David Kinny
Jane Yung-jen Hsu
Guido Governatori
Aditya Ghose (Eds.)

LNAI 7047

Agents in Principle, Agents in Practice

14th International Conference, PRIMA 2011
Wollongong, Australia, November 2011
Proceedings

Springer

David Kinny   Jane Yung-jen Hsu
Guido Governatori   Aditya Ghose (Eds.)

Agents in Principle, Agents in Practice

14th International Conference, PRIMA 2011
Wollongong, Australia, November 16-18, 2011
Proceedings

Springer

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

David Kinny
Kyoto University, Department of Social Informatics
Yoshida Honmachi, Sakyo-ku, Kyoto 606-8501, Japan
E-mail: dnk@i.kyoto-u.ac.jp

Jane Yung-jen Hsu
National Taiwan University
Department of Computer Science and Information Engineering
1 Roosevelt Road, Taipei 106, Taiwan
E-mail: yjhsu@csie.ntu.edu.tw

Guido Governatori
NICTA, Queensland Research Laboratory
PO Box 6020, St. Lucia, QLD 4067, Australia
E-mail: guido.governatori@nicta.com.au

Aditya Ghose
University of Wollongong
School of Computer Science and Software Engineering
Wollongong, NSW 2522, Australia
E-mail: aditya@uow.edu.au

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-25043-9 e-ISBN 978-3-642-25044-6
DOI 10.1007/978-3-642-25044-6
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011940004

CR Subject Classification (1998): I.2, I.2.11, I.2.9, C.2.4, K.4, D.2, H.3-5

LNCS Sublibrary: SL 7 – Artificial Intelligence

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)

Preface

Agent computing is a remarkably successful and transformational approach to developing computer systems that can rapidly and reliably solve real-world problems that usually demand human knowledge and expertise. The value, power and flexibility of agent and multi-agent systems have been demonstrated in application areas such as logistics, manufacturing, simulation, robotics, decision-support, entertainment, online markets, as well as disaster management and military applications. As one of the largest and fastest growing research fields of computer science, research into agent computing today spans a wealth of topic areas; this will be apparent to the reader of this volume, which contains the 39 papers presented at PRIMA 2011 in Wollongong, Australia.

PRIMA is one of the oldest active agent computing forums, beginning in 1998 as a regional agent workshop (the Pacific Rim Workshop on Multi-Agents). PRIMA has since grown into an international scientific conference on agents and multi-agent systems which attracts high-quality, cutting-edge research from all over the world, and is now known as the International Conference on Principles and Practice of Multi-Agent Systems. The 2011 conference built upon the success of its 13 predecessors, and attracted contributions from 29 countries. Of these, 24% were accepted as full papers, and a further 22 interesting and promising but not fully mature contributions were accepted at reduced length as *early innovation* papers.

Throughout its history PRIMA has focused on showcasing the impact of agents on the real world, across the spectrum from early research and prototypes to mature, deployed systems. The conference theme for PRIMA 2011 of *Agents for Sustainability* sought to encourage thought leadership in the agent community as to how agent computing can be applied to enhance sustainable practices in today's world. We are pleased that the authors of several contributions and an invited talk in this volume have taken up the challenge of addressing this vital topic, and we hope also to see the growth of research focused on this theme within the broad agent community, as we believe it offers a unique set of challenges and opportunities for the community to deliver innovative solutions and lasting value.

We thank the PRIMA Steering Committee, especially the Chair, Makoto Yokoo, and all others who contributed to the event's success.

November 2011

David Kinny
Jane Yung-jen Hsu
Guido Governatori
Aditya Ghose

Organization

PRIMA 2011 was organized by the following people and held at the University of Wollongong, Australia.

General Chairs

Aditya Ghose	University of Wollongong, Australia
Guido Governatori	NICTA, Australia

Program Chairs

Jane Hsu	National Taiwan University, Taiwan
David Kinny	Kyoto University, Japan

Deputy Program Chairs

Hoa Dam	University of Wollongong, Australia
Serena Villata	INRIA, Sophia Antipolis, France

Publicity Chairs

Wayne Wobcke	University of New South Wales, Australia
Pascal Perez	University of Wollongong, Australia
Louise Leenen	CSIR, South Africa

Workshop Chairs

Stephen Cranefield	University of Otago, New Zealand
Insu Song	James Cook University, Australia

Local Arrangements Chairs

Minjie Zhang	University of Wollongong, Australia
Hoa Dam	University of Wollongong, Australia

Webmaster

Graham Billiau	University of Wollongong, Australia

Senior Program Committee

Stephen Cranefield	University of Otago, New Zealand
Michael Luck	King's College London, UK
John-Jules Meyer	Utrecht University, The Netherlands
Iyad Rahwan	Masdar Institute of Science and Technology, UAE
Paul Scerri	Carnegie Mellon University, USA
Munindar P. Singh	North Carolina State University, USA
Von-Wun Soo	National Tsing Hua University, Taiwan
Wiebe Van Der Hoek	University of Liverpool, UK

Program Committee

Mohan Baruwal Chhetri	Yichuan Jiang	Nir Oren
Alexandros Belesiotis	Wan-Rong Jih	Juan Pavón
Ghassan Beydoun	Zhi Jin	Henry Prakken
Frances Brazier	Benjamin Johnston	Michael Rovatsos
Longbing Cao	Kee-Eung Kim	Ji Ruan
Brahim Chaib-Draa	Yasuhiko Kitamura	Yuko Sakurai
Sanjay Chaudhary	Kazuhiro Kuwabara	Bastin T.R. Savarimuthu
Shih-Fen Cheng	Jérôme Lang	Tino Schlegel
Sung-Bae Cho	Habin Lee	Murat Sensoy
Mehdi Dastani	Jaeho Lee	Kiam Tian Seow
Frank Dignum	Ho-Fung Leung	Biplav Srivastava
Edith Elkind	Lin Liu	Eugen Staab
Marc Esteva	Graham Low	Toshiharu Sugawara
Joseph Giampapa	Beatriz López	Nicolas Troquard
Chung-Wei Hang	Xinjun Mao	M. Birna Van Riemsdijk
Hiromitsu Hattori	Shigeo Matsubara	Pradeep Varakantham
Christopher Hazard	Felipe Meneguzzi	Yonghong Wang
Sarah Hickmott	Yohei Murakami	Yang Xu
Koen Hindriks	Hideyuki Nakanishi	Yifeng Zeng
Michal Jakob	Mariusz Nowostawski	

Additional Reviewers

Karthik Aadithya	Jianye Hao	Inmaculada Rodríguez
Reyhan Aydogan	Zhichuan Huang	Ruben Stranders
Georgios Chalkiadakis	Anup Kalia	Toru Takahashi
Chi-Kong Chan	Stepan Kopriva	Tomas Trescak
Jaedeug Choi	Viliam Lisy	Jos Vrancken
Michael Da Costa Mora	Motohiro Mase	Makoto Yokoo
Christopher Frantz	Reshef Meir	Zhe Zhang
Enrico Gerding	Valentin Robu	

Table of Contents

Invited Talks

Game Theory for Security: Lessons Learned from Deployed Applications 1
Milind Tambe

Tools for a Robust, Sustainable Agent Community 2
Sandip Sen

From Notions to Models and Back Again, Again 3
Liz Sonenberg

Full Papers

Coalitions and Teamwork

A Compact Representation Scheme of Coalitional Games Based on Multi-Terminal Zero-Suppressed Binary Decision Diagrams 4
Yuko Sakurai, Suguru Ueda, Atsushi Iwasaki, Shin-Ichi Minato, and Makoto Yokoo

Environment Characterization for Non-recontaminating Frontier-Based Robotic Exploration 19
Mikhail Volkov, Alejandro Cornejo, Nancy Lynch, and Daniela Rus

Learning

Aspects of Active Norm Learning and the Effect of Lying on Norm Emergence in Agent Societies 36
Bastin Tony Roy Savarimuthu, Rexy Arulanandam, and Maryam Purvis

Mechanisms and Voting

Strategy-Proof Mechanisms for Interdependent Task Allocation with Private Durations 51
Ayman Ghoneim

Costly Voting with Sequential Participation 68
Ryuya Kagifuku and Shigeo Matsubara

Modelling and Simulation

An Agent-Based Model for Integrated Contagion and Regulation of Negative Mood 83
Azizi Ab Aziz, Jan Treur, and C. Natalie van der Wal

A Framework for Agent-Based Modeling of Intelligent Goods 97
Åse Jevinger, Paul Davidsson, and Jan A. Persson

Multi-Agent Systems for Biomedical Simulation: Modeling Vascularization of Porous Scaffolds 113
Hamidreza Mehdizadeh, Arsun Artel, Eric M. Brey, and Ali Cinar

Group Abstraction for Large-Scale Agent-Based Social Diffusion Models with Unaffected Agents 129
Alexei Sharpanskykh and Jan Treur

Negotiation

Towards a Quantitative Concession-Based Classification Method of Negotiation Strategies 143
Tim Baarslag, Koen Hindriks, and Catholijn Jonker

Consensus Policy Based Multi-agent Negotiation 159
Enrique de la Hoz, Miguel A. Lopez-Carmona, Mark Klein, and Ivan Marsa-Maestre

Optimization

Distributed Lagrangian Relaxation Protocol for the Over-constrained Generalized Mutual Assignment Problem 174
Kenta Hanada and Katsutoshi Hirayama

The Effect of Congestion Frequency and Saturation on Coordinated Traffic Routing 187
Melanie Smith and Roger Mailler

Sustainability

Coordination, Conventions and the Self-organisation of Sustainable Institutions 202
Jeremy Pitt, Julia Schaumeier, and Alexander Artikis

An Agent-Based Extensible Climate Control System for Sustainable Greenhouse Production 218
Jan Corfixen Sørensen, Bo Nørregaard Jørgensen, Mark Klein, and Yves Demazeau

Applications

ACTraversal: Ranking Crowdsourced Commonsense Assertions and Certifications 234
Tao-Hsuan Chang, Yen-Ling Kuo, and Jane Yung-jen Hsu

Automated Adaptation of Strategic Guidance in Multiagent Coordination 247
Rajiv T. Maheswaran, Pedro Szekely, and Romeo Sanchez

Early Innovation Papers

Agent Societies and Frameworks

Weaving a Fabric of Socially Aware Agents 263
Mark d'Inverno, Michael Luck, Pablo Noriega, Juan A. Rodriguez-Aguilar, and Carles Sierra

Dynamic Ad Hoc Coordination of Distributed Tasks Using Micro-Agents 275
Christopher Frantz, Mariusz Nowostawski, and Martin K. Purvis

Programming Dynamics of Multi-Agent Systems 287
Cuiyun Hu, Xinjun Mao, and Huiping Zhou

Capability Modeling of Knowledge-Based Agents for Commonsense Knowledge Integration 299
Yen-Ling Kuo and Jane Yung-jen Hsu

Producing Enactable Protocols in Artificial Agent Societies 311
George K. Lekeas, Christos Kloukinas, and Kostas Stathis

Argumentation

Argumentation Schemes for Collaborative Planning 323
Alice Toniolo, Timothy J. Norman, and Katia Sycara

Preference-Based Argumentation Handling Dynamic Preferences Built on Prioritized Logic Programming 336
Toshiko Wakaki

Learning

Adaption of Stepsize Parameter Using Newton's Method 349
Itsuki Noda

A Health Social Network Recommender System 361
Insu Song, Denise Dillon, Tze Jui Goh, and Min Sung

Modelling

Learning Belief Connections in a Model for Situation Awareness 373
Maria L. Gini, Mark Hoogendoorn, and Rianne van Lambalgen

An Integrated Agent Model Addressing Situation Awareness and Functional State in Decision Making 385
Mark Hoogendoorn, Rianne van Lambalgen, and Jan Treur

Agent-Based Modelling for Understanding Sustainability 398
Sonja Pedell and Leon Sterling

Modelling Joint Decision Making Processes Involving Emotion-Related Valuing and Empathic Understanding 410
Jan Treur

Negotiation and Coalitions

Negowiki: A Set of Community Tools for the Consistent Comparison of Negotiation Approaches.. 424
Ivan Marsa-Maestre, Mark Klein, Enrique de la Hoz, and Miguel A. Lopez-Carmona

A Stochastic Negotiation Approach to Power Restoration Problems in a Smart Grid 436
Von-Wun Soo and Yen-Bo Peng

Coalition-Oriented Sensing in Wireless Sensor Networks............... 448
M. del Carmen Delgado-Roman and Carles Sierra

Simulation

Reducing the Environmental Impact of New Construction Projects through General Purpose Building Design and Multi-agent Crowd Simulation 460
Kieron Ekron, Jaco Bijker, and Elize Ehlers

An Investigation of Emergent Collaboration under Uncertainty and Minimal Information in Energy Domains........................... 472
Radu-Casian Mihailescu, Matteo Vasirani, and Sascha Ossowski

A Simulator Integration Platform for City Simulations................ 484
Yuu Nakajima and Hiromitsu Hattori

Human Relationship Modeling in Agent-Based Crowd Evacuation Simulation 496
Masaru Okaya and Tomoichi Takahashi

Applications

A Web Service Recommendation System Based on Users' Reputations 508
Yu Furusawa, Yuta Sugiki, and Reiko Hishiyama

Human-Centered Planning for Adaptive User Situation in Ambient Intelligence Environment 520
Makoto Sando and Reiko Hishiyama

Author Index 533

Game Theory for Security: Lessons Learned from Deployed Applications[*]

Milind Tambe

Computer Science and Industrial & Systems Engineering Departments
University of Southern California
Los Angeles, California, USA
tambe@usc.edu

Abstract. Security at major locations of economic or political importance or transportation or other infrastructure is a key concern around the world, particularly given the threat of terrorism. Limited security resources prevent full security coverage at all times; instead, these limited resources must be deployed intelligently taking into account differences in priorities of targets requiring security coverage, the responses of the adversaries to the security posture and potential uncertainty over the types of adversaries faced. Game theory is well-suited to adversarial reasoning for security resource allocation and scheduling problems. Casting the problem as a Bayesian Stackelberg game, we have developed new algorithms for efficiently solving such games to provide randomized patrolling or inspection strategies: we can thus avoid predictability and address scale-up in these security scheduling problems, addressing key weaknesses of human scheduling. Our algorithms are now deployed in multiple applications. ARMOR, our first game theoretic application, has been deployed at the Los Angeles International Airport (LAX) since 2007 to randomize checkpoints on the roadways entering the airport and canine patrol routes within the airport terminals. IRIS, our second application, is a game-theoretic scheduler for randomized deployment of the Federal Air Marshals (FAMS) requiring significant scale-up in underlying algorithms; IRIS has been in use since 2009. Similarly, a new set of algorithms are deployed in Boston for a system called PROTECT for randomizing US coast guard patrolling; PROTECT is intended to be deployed at more locations in the future, and GUARDS is under evaluation for national deployment by the Transportation Security Administration (TSA). These applications are leading to real-world use-inspired research in scaling up to large-scale problems, handling significant adversarial uncertainty, dealing with bounded rationality of human adversaries, and other fundamental challenges. This talk will outline our algorithms, key research results and lessons learned from these applications.

[*] This is joint work with several researchers, including former and current members of the Teamcore group, please see http://teamcore.usc.edu/projects/security

Tools for a Robust, Sustainable Agent Community

Sandip Sen

Department of Mathematical & Computer Sciences,
The University of Tulsa
Tulsa, Oklahoma, USA

Abstract. We believe that intelligent information agents will represent their users interest in electronic marketplaces and other forums to trade, exchange, share, identify, and locate goods and services. Such information worlds will present unforeseen opportunities as well as challenges that can be best addressed by robust, self-sustaining agent communities. 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. More specifically, agents will need mechanisms to benefit from complementary expertise in the group, pool together resources to meet new demands and exploit transient opportunities, negotiate fair settlements, develop norms to facilitate coordination, exchange help and transfer knowledge between peers, secure the community against intruders, and learn to collaborate effectively. In this talk, I will summarize some of our research results on trust-based computing, negotiation, and learning that will enable intelligent agents to develop and sustain robust, adaptive, and successful agent communities.

From Notions to Models and Back Again, Again

Liz Sonenberg

Department of Information Systems
The University of Melbourne
Parkville, Victoria, Australia

Abstract. A typical thread in research relies on the maturing of ideas through an iterative process of construction, testing and refinement. In this talk I will trace some such trajectories of ideas by illustration from some of my own and others' experiences in agent-based modelling. I draw inspiration from previous commentaries, including from those who generated the mottos: "No Notation without Denotation" [1]; "No Notation without Exploitation" [2]; and "No experimentation without explanation" [3].

References

1. McDermott, D.: Tarskian Semantics, or No Notation without Denotation! Cognitive Science 2, 277–282 (1978)
2. Shoham, Y.: Agent Oriented Programming. Artificial Intelligence 60, 51–92 (1993)
3. Dignum, F., Sonenberg, L.: A dialogical argument for the usefulness of logic in MAS. Discussion contribution in Journal of Artificial Societies and Social Simulation 7(4) (2004), http://jasss.soc.surrey.ac.uk/7/4/8/logic-in-abss/Dignum&Sonenberg-reply.html

A Compact Representation Scheme of Coalitional Games Based on Multi-Terminal Zero-Suppressed Binary Decision Diagrams

Yuko Sakurai[1], Suguru Ueda[1], Atsushi Iwasaki[1], Shin-Ichi Minato[2], and Makoto Yokoo[1]

[1] Kyushu University
{ysakurai@,ueda@agent.,iwasaki@,yokoo@}inf.kyushu-u.ac.jp
[2] Hokkaido University
minato@ist.hokudai.ac.jp

Abstract. Coalitional games, including Coalition Structure Generation (CSG), have been attracting considerable attention from the AI research community. Traditionally, the input of a coalitional game is a black-box function called a characteristic function. Previous studies have found that many problems in coalitional games tend to be computationally intractable in this black-box function representation. Recently, several concise representation schemes for a characteristic function have been proposed. Among them, a synergy coalition group (SCG) has several good characteristics, but its representation size tends to be large compared to other representation schemes.

We propose a new concise representation scheme for a characteristic function based on a Zero-suppressed Binary Decision Diagram (ZDD) and a SCG. We show our scheme (i) is fully expressive, (ii) can be exponentially more concise than the SCG representation, (iii) can solve core-related problems in polynomial time in the number of nodes, and (iv) can solve a CSG problem reasonably well by utilizing a MIP formulation. A Binary Decision Diagram (BDD) has been used as unified infrastructure for representing/manipulating discrete structures in such various domains in AI as data mining and knowledge discovery. Adapting this common infrastructure brings up the opportunity of utilizing abundant BDD resources and cross-fertilization with these fields.

1 Introduction

Forming effective coalitions is a major research challenge in AI and multi-agent systems (MAS). A coalition of agents can sometimes accomplish things that individual agents cannot or can do things more efficiently. There are two major research topics in coalitional games. The first involves partitioning a set of agents into coalitions so that the sum of the rewards of all coalitions is maximized. This is called the Coalition Structure Generation problem (CSG) [19]. The second topic involves how to divide the value of the coalition among agents. The theory of coalitional games provides a number of solution concepts, such as the core, the Shapley value, and the nucleolus.

Previous studies have found that many problems in coalitional games, including CSG, tend to be computationally intractable. Traditionally, the input of a coalitional game is a black-box function called a characteristic function that takes a coalition as an input and returns its value. Representing an arbitrary characteristic function explicitly requires $\Theta(2^n)$ numbers, which is prohibitive for large n.

Recently, several concise representation schemes for a characteristic function have been proposed [8–11, 13, 20, 21]. Among them, the synergy coalition group (SCG) [8] has several good characteristics: (i) it is a general representation scheme that can represent any characteristic function, (ii) its representation is simple and intuitively natural, (iii) core-related problems can be solved efficiently, and (iv) although solving a CSG is NP-hard, it can be solved reasonably well by utilizing a MIP formulation. However, a SCG tends to require more space than other representation schemes such as marginal contribution networks [13].

In this paper, we propose a new concise representation scheme for a characteristic function, based on the idea of *Binary Decision Diagram* (BDD) [3, 7]. A BDD is graphical representations that can compactly represent a boolean function. We use a variant of BDD called a Zero-suppressed BDD (ZDD) [16] that can compactly represent a set of combinations. More specifically, we use a Multi-Terminal ZDD (MTZDD), which can compactly represent a SCG. This representation preserves the good characteristics of a SCG. The following are the features of our scheme: (i) it is fully expressive, (ii) it can be exponentially more concise than a SCG, (iii) such core-related problems as core-non-emptiness, core-membership, and finding a minimal non-blocking payoff vector (cost of stability) can be solved in polynomial time in the number of nodes in a MTZDD, and (iv) although solving a CSG is NP-hard, it can be solved reasonably well by utilizing a MIP formulation.

A BDD was originally developed for VLSI logic circuit design. Recently, A BDD has been applied to various domains in AI, including data mining and knowledge discovery [15]. In these domains, we need to handle logic functions or combination sets efficiently. A BDD has been used as unified infrastructures for representing/manipulating such *discrete structures*. A vast amount of algorithms, software, and tools related to a BDD already exist, e.g., an arithmetic boolean expression manipulator based on a BDD, and a programs for calculating combination sets based on a ZDD [17]. Adapting this common infrastructure for coalitional game theory brings up the opportunity to utilize these abundant resources and for cross-fertilization with other related fields in AI.

The rest of this paper is organized as follows. First, we describe our model and background (Section 2). Next, we define our MTZDD representation and evaluate its conciseness (Section 3). Then we examine the computational complexity of coalitional games including CSG (Section 4) and core-related problems (Section 5). Finally, we show the results of experimental simulations (Section 6) and conclude this paper and describe future works (Section 7).

2 Preliminaries

2.1 Coalitional Games

Let $A = \{1, 2, \ldots, n\}$ be the set of agents. Since we assume a characteristic function game, the value of coalition S is given by characteristic function v, which assigns a value to each set of agents (coalition) $S \subseteq A$. We assume that each coalition's value is non-negative.

Coalition structure CS is a partition of A into disjoint and exhaustive coalitions. To be more precise, $CS = \{S_1, S_2, \ldots\}$ satisfies the following conditions: $\forall i, j\ (i \neq j),\ S_i \cap S_j = \emptyset, \bigcup_{S_i \in CS} S_i = A$. The value of coalition structure CS, denoted as $V(CS)$, is given by: $V(CS) = \sum_{S_i \in CS} v(S_i)$. Optimal coalition structure CS^* is a coalition structure that satisfies the following condition: $\forall CS, V(CS^*) \geq V(CS)$.

We say a characteristic function is super-additive, if for any disjoint sets S_i, S_j, $v(S_i \cup S_j) \geq v(S_i) + v(S_j)$ holds. If the characteristic function is super-additive, solving CSG becomes trivial; the grand coalition (the coalition of all agents) is optimal. We assume a characteristic function can be non-super-additive.

The core is a prominent solution concept focusing on stability. When a characteristic function is not necessarily super-additive, creating a grand coalition does not make sense. As discussed in [4], we need to consider the stability of a coalition structure. The concept of the core can be extended to the case where agents create an optimal coalition structure. Assume $\pi = (\pi_1, \ldots, \pi_n)$ describes how to divide the obtained reward among agents. We call π a *payoff vector.*

Definition 1 (Core (for CS^*)). *The core is the set of all payoff vectors π that satisfy the feasibility condition: $\sum_{i \in A} \pi_i = V(CS^*)$, and non-blocking condition: $\forall S \subseteq A, \sum_{i \in S} \pi_i \geq v(S)$.*

If for some set of agents S, the non-blocking condition does not hold, then the agents in S have an incentive to collectively deviate from CS^* and divide $v(S)$ between themselves. As discussed in [2], there are two alternative definitions of the feasibility condition: (i) $\sum_{i \in A} \pi_i = V(CS^*)$, and (ii) $\forall S \in CS^*, \sum_{i \in S} \pi_i = v(S)$. If (ii) holds, then (i) holds, but not vice versa. Condition (ii) requires that no monetary transfer (side payment) exists across different coalitions. However, as shown in [4], if a payoff vector satisfies both condition (i) and the non-blocking condition, it also satisfies condition (ii). Thus, we use condition (i) as the feasibility condition.

In general, the core can be empty. The ϵ-core can be obtained by relaxing the non-blocking condition as follows: $\forall S \subseteq A, \sum_{i \in S} \pi_i + \epsilon \geq v(S)$. When ϵ is large enough, the ϵ-core is guaranteed to be non-empty. The smallest non-empty ϵ-core is called the least core.

Alternatively, we can relax the feasibility condition as follows: $\sum_{i \in A} \pi_i = V(CS^*) + \Delta$. This means that an external party is willing to pay amount Δ as a subsidy to stabilize the coalition structure. The minimal amount of Δ is called the *cost of stability* [5].

2.2 SCG

Conitzer and Sandholm (2006) [8] introduced a concise representation of a characteristic function called a *synergy coalition group (SCG).* The main idea is to explicitly represent the value of a coalition only when some *positive* synergy exists.

Definition 2 (SCG). *An SCG consists of a set of pairs of the form:* $(S, v(S))$*. For any coalition* S*, the value of the characteristic function is:* $v(S) = \max_{p_S} \{\sum_{S_i \in p_S} v(S_i)\}$*, where* p_S *is a partition of* S*; all* S_i*s are disjoint and* $\bigcup_{S_i \in p_S} S_i = S$*, and for all the* S_i*,* $(S_i, v(S_i)) \in SCG$*. To avoid senseless cases without feasible partitions, we require that* $(\{a\}, 0) \in SCG$ *whenever* $\{a\}$ *does not receive a value elsewhere in SCG.*

If the value of coalition S is not given explicitly in SCG, it is calculated from the possible partitions of S. Using this original definition, we can represent only super-additive characteristic functions. To allow for characteristic functions that are not super-additive, we add the following requirement on the partition p_S: $\forall p'_S \subseteq p_S$, where $|p'_S| \geq 2$, $(\bigcup_{S_i \in p'_S} S_i, v(\bigcup_{S_i \in p'_S} S_i))$ is not an element of SCG.

This additional condition requires that if the value of a coalition is explicitly given in SCG, then we cannot further divide it into smaller subcoalitions to calculate values. In this way, we can represent *negative* synergies.

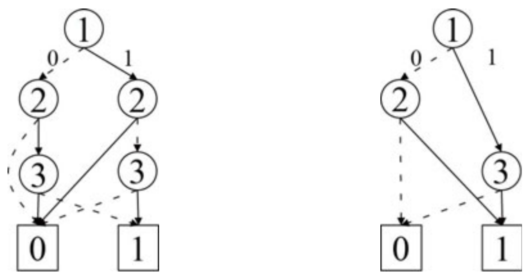
Example 1. Let there be five agents $1, 2, 3, 4, 5$ and let $SCG = \{(\{1\}, 0), (\{2\}, 0), (\{3\}, 1), (\{4\}, 2), (\{5\}, 3), (\{1, 2\}, 3), (\{1, 2, 3\}, 3)\}$. In this case, $v(\{4, 5\}) = v(\{4\}) + v(\{5\}) = 5$, and $v(\{1, 2, 3, 4, 5\}) = v(\{1, 2, 3\}) + v(\{4\}) + v(\{5\}) = 8$. For $v(\{1, 2, 3, 4, 5\})$, we cannot use $v(\{1, 2\}) + v(\{3\}) + v(\{4\}) + v(\{5\}) = 9$, because $\{1, 2\} \cup \{3\} = \{1, 2, 3\}$ appears in SCG.

2.3 BDD and ZDD

A BDD represents boolean functions as a rooted, directed acyclic graph of internal nodes and two 0/1-terminal nodes. Each internal node represents a variable and has two outgoing edges: a high-edge and a low-edge. The high-/low-edge means that the value of the variable is true/false. A path from the root node to the 1-terminal node represents that the corresponding value assignment to the path makes the boolean function true.

A ZDD is a variant of BDD that can efficiently represent a set of combinations. The high-/low-edge means the presence/absence of an element in a combination. In a ZDD, a path from the root node to the 1-terminal node represents that the corresponding value assignment to the path is included in the set. On the other hand, a path from the root node to the 0-terminal node represents that such a combination does not exist in a set of combinations.

Consider boolean function $((x_1 \bar{x}_2 x_3) \vee (\bar{x}_1 x_2 \bar{x}_3))$, which can be equivalently represented by using a set of combinations $(\{\{1, 3\}, \{2\}\})$. Figure 1 shows the BDD/ZDD representation for this function/set of combinations. In a tree, a node with x_i represents i. A ZDD is more concise than a BDD. If a variable never



(a) BDD (b) ZDD

Fig. 1. Examples of a BDD and a ZDD

appears within any elements in a set of combinations, a node that represents the variable is removed from the ZDD. If the sum of elements contained in all combinations in a set is k, the number of nodes in a ZDD is at most $O(k)$.

Quite recently, two different BDD-based representation schemes for a characteristic function have been developed independently from our work [1, 6]. While Berghammer and Bolus (2010) deals with simple games, Aadithya *et al.* (2011) considers general games. Both schemes try to represent a characteristic function directly, while our scheme represents SCGs. As discussed in the next section, our ZDD-based representation can be exponentially more concise than a BDD-based representation scheme. Also, solving a CSG problem is not considered in these works.

3 New Concise Representation Scheme

We propose our new representation scheme for a characteristic function based on a SCG and a ZDD. Although a ZDD can only represent whether a combination exists in a set, a SCG is not just a set of coalitions, because each coalition S in a SCG is associated with its value $v(S)$. Thus, we use a multi-terminal ZDD (MTZDD) representation described as follows.

3.1 MTZDD Representation Based on SCG

A MTZDD G is defined by (V, T, H, L), where V is a set of internal (non-terminal) nodes, T is a set of terminal nodes, H is a set of high-edges, and L is a set of low-edges. Each internal node $u \in V$ is associated with one agent, which we denote as $agent(u)$. u has exactly two outgoing edges, $h(u) = (u, u')$ and $l(u) = (u, u'')$, where $h(u) \in H$ and $l(u) \in L$. Each terminal node $t \in T$ is associated with a non-negative value, which we denote as $r(t)$. Root node u_0 has no incoming edges. For each node $u \in V \setminus \{u_0\} \cup T$, at least one incoming edge exists. We denote the parents of u as $Pa(u)$, $Pa(u) = \{u' \mid (u', u) \in H \cup L\}$.

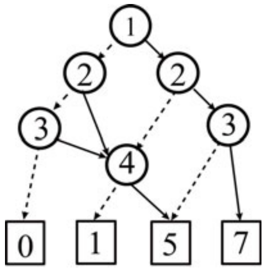


Fig. 2. MTZDD representation in Example 2

Path p from root node u_0 to terminal node t is represented by a sequence of edges on path $p = ((u_0, u_1), (u_1, u_2), \dots, (u_k, t))$. For p, we denote $S(p) = \{agent(u_i) \mid h(u_i) \in p\}$, because $S(p)$ denotes a coalition represented by path p. Also, we denote the value of path p as $r(p)$, which equals $r(t)$: $v(S(p)) = r(t)$. In a MTZDD, a particular ordering among agents is preserved. In path p from root node u_0 to terminal node t, agents associated with nodes in p appear in the same order. More specifically, if node u appears before node u' in p, then $agent(u) \neq agent(u')$. Also, there exists no path p', in which node u appears before node u', where $agent(u) = agent(u')$. For each agent $i \in A$, $nodes(i)$ denotes a set of nodes that are associated with agent i, i.e., $nodes(i) = \{u \mid u \in V \land agent(u) = i\}$.

Here, we present an example for a MTZDD representation.

Example 2. Let there be four agents: 1, 2, 3, and 4. Let $SCG = \{(\{1\}, 1), (\{2\}, 1), (\{3\}, 1), (\{4\}, 0), (\{1, 2\}, 5), (\{1, 4\}, 5), (\{2, 4\}, 5), (\{3, 4\}, 5), (\{1, 2, 3\}, 7)\}$. This MTZDD representation appears in Figure 2. For example, the rightmost path of the tree represents a coalition $\{1, 2, 3\}$ and its value 7.

3.2 Conciseness of MTZDD Representation

Theorem 1. *MTZDD can represent any characteristic function represented in a SCG using at most $O(n|SCG|)$ nodes, where n is the number of agents and $|SCG|$ is the number of elements in a SCG.*

Proof. In a MTZDD, for each agent i, $|nodes(i)|$ is at most $|SCG|$ because $|nodes(i)|$ represents the number of different contexts that result in different outcomes. This number is bounded by the number of different combinations of agents, which appear before i in the ordering among agents. Clearly, this number is at most $|SCG|$. Thus, the number of non-terminal nodes, i.e., $\sum_{i \in A} |nodes(i)|$, is at most $n|SCG|$. Also, the number of terminal nodes is at most $|SCG| + 1$. As a result, the total number of nodes is $O(n|SCG|)$. □

Theorem 2. *A MTZDD representation is exponentially more concise than a SCG for certain games.*

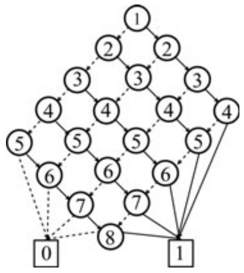


Fig. 3. MTZDD representation in Theorem 2

Proof. Consider a coalitional game with $2m$ agents, where the value of characteristic function $v(S)$ is 1 if $|S| \geq m$, and 0 otherwise. A SCG must include each coalition with size m. The number of such coalitions is given as $\binom{2m}{m}$, which is $O(2^n)$ using Stirling's approximation.

On the other hand, we can create a MTZDD that counts the number of agents in a coalition and returns 1 when the number reaches m. Such MTZDD requires $m(m+1)$ nodes, i.e., $O(n^2)$. $\square$

As shown in the proof of Theorem 2, when some agents are *symmetric*, the MTZDD representation can be much more concise than a SCG. Figure 3 shows a MTZDD when we set $m = 4$. The number of nodes is 20, but a SCG requires 70 coalitions.

Instead of representing a SCG with a MTZDD, we can directly represent a characteristic function using a MTBDD (such an approach is considered in [1, 6]). In a MTBDD, an agent that does not appear in a path is considered irrelevant; if $v(S \cup \{i\}) = v(S)$, we only need to describe S in a MTBDD[1]. Thus, we can reduce the representation size to a certain extent by using a MTBDD. However, this MTBDD representation for a characteristic function is not as concise as the MTZDD representation. The following theorem holds.

Theorem 3. *A MTZDD representation of a SCG is always as concise as a MTBDD representation of a characteristic function. Also, it is exponentially more concise than a MTBDD representation for certain games.*

Proof. The worst case occurs when a SCG contains all possible coalitions. In this case, the representation sizes of the MTZDD and MTBDD are the same.

Then, we show the case where the MTZDD representation is exponentially more concise. Consider a coalitional game with agents $1, 2, \ldots, n$, where $v(\{i\}) = 2^i$, and $v(S) = \sum_{i \in S} v(\{i\})$. $v(S)$ can take any integer value from 1 to $2^{n+1} - 1$. Thus, the number of terminal nodes in the MTBDD becomes $O(2^n)$. On the other hand, the number of elements in a SCG is n, the number of internal nodes in the MTZDD is n, and the number of terminal nodes is $n + 1$. Thus, the total number of nodes is $O(n)$. $\square$

[1] Note that such an irrelevant agent is not included in a SCG.

In this thorem, we compare the representation complexity of (a) our MTZDD representation of a SCG and (b) a MTBDD representation of a standard characteristic function. The representation sizes of (MT)ZDD and (MT)BDD should be within the factor of n, if we represent the same thing [14], but these two approaches are based on different representations. Thus, MTZDD representation of a SCG is exponentially more concise than a MTBDD representation for certain games, since the SCG representation can be exponentially more concise than the standard characteristic function representation.

3.3 Procedure of Constructing a MTZDD Representation

Let us consider how a person, who has knowledge of a coalitional game, can describe our MTZDD-based representation. We assume she is aware of symmetry among agents. She constructs a MTZDD by using the following procedure.

1. She describes several partial MTZDDs considering the symmetry among agents. Each partial MTZDD can correspond to multiple (possibly exponentially many) elements in a SCG. We assume each partial MTZDD is reasonably small so that she can describe it manually (possibly with the help of an automated translation tool and/or templates). For example, if she is describing the characteristic function used in the proof of Theorem 2, we can assume she describes multiple partial MTZDDs, each of which corresponds to coalitions of k agents (where k varies from m to $2m$). She can use a template of a MTZDD for a k-out-of-n function.
2. She integrates these partial MTZDDs into a single MTZDD by applying a Union operation [16] and reduction rules described in Section 2.3. By applying the reduction rule, all equivalent subgraphs will be shared.

4 Coalition Structure Generation

We propose a new mixed integer programming formulation for solving a CSG problem in the MTZDD representation.

Various logical operations for a BDD/ZDD have previously been implemented as polynomial-time graph manipulation algorithms [7, 17]. However, as far as the authors aware, none of standard BDD/ZDD package is able to perform complex combinatorial optimization operations such as solving a linear or integer programming so far. Thus, the method described in this section relies on external codes/routines. In our MTZDD representation, a path from the root node to a terminal node represents a coalition that is included in a SCG. We define a condition where a set of paths, i.e., a set of coalitions, is *compatible.*

Definition 3 (Compatible paths). *Two paths, p and p', are compatible if $S(p) \cap S(p') = \emptyset$. Also, set of paths P is compatible if $\forall p, p' \in P$, where $p \neq p'$, p and p' are compatible.*

Finding optimal coalition structure CS^* is equivalent to finding set of paths P^*, which is compatible, and $\sum_{p \in P^*} r(p)$ is maximized. We show that P^* is NP-complete and inapproximable.

Theorem 4. *When the characteristic function is represented as a MTZDD, finding an optimal coalition structure is NP-hard. Moreover, unless P = NP, there exists no polynomial-time $O(|SCG|^{1-\epsilon})$ approximation algorithm for any $\epsilon > 0$.*

Proof. The maximum independent set problem is to choose $V' \subseteq V$ for a graph $G = (V, E)$ such that no edge exists between vertices in V', and $|V'|$ is maximized under this constraint. It is NP-hard, and unless $\mathcal{P} = \mathcal{NP}$, there exists no polynomial-time $O(|V|^{1-\epsilon})$ approximation algorithm for any $\epsilon > 0$ [12]. We reduce an arbitrary maximum independent set instance to a CSG problem instance as follows. For each $e \in E$, let there be agent a_e. For each $v \in V$, we create an element of SCG, where the coalition is $\{a_e \mid e \ni v\}$ and its value is 1. Thus, two coalitions have a common element only if they correspond to neighboring vertices. Coalition structures correspond exactly to independent sets of vertices. Furthermore, we transform this SCG representation to a MTZDD representation in polynomial time [16]. As a result, the number of internal nodes in a MTZDD is at least $|E|$ and at most $2|E|$, since an agent appears in exactly two coalitions. $\square$

The conciseness of a SCG-base representation incurs some cost, i.e., obtaining the value of the characteristic function for a single coalition can be NP-hard. However, we assume a characteristic function can be non-super-additive (or even non-monotonic). Thus, obtaining a coalition value of S is not very meaningful, since it might not be optimal for S to work as a single team. It is more important to obtain an optimal coalition structure (and its value) that consists of agents in S. Although this computation is NP-hard, we provide a MIP formulation for solving this computation as follows.

Ohta *et al.* (2009) [18] developed a MIP formulation for a CSG problem when a characteristic function is represented by a SCG. If we enumerate paths (i.e., all elements of SCGs), we can use their results. However, the number of paths can be exponential to the number of nodes in a MTZDD. Thus, we need to find P^* without explicitly enumerating all possible paths. We first identify the maximal number of paths within P^*, which leads to one terminal node $r(t)$, using a concept called *minimal required high-edge set* that is concisely described *minimal set.*

Definition 4 (minimal set). *For each terminal node $t \in T$, where $r(t) > 0$, $E \subseteq H$ is a required high-edge set if for all paths p, where t is p's terminal node, there exists $h \in E$ such that h is included in p. E is a minimal set, if E is a required high-edge set, and there exists no proper subset of E that is a required high-edge set.*

There can be multiple minimal sets. We can find one minimal set using backtrack search starting from the terminal node. The complexity of this procedure is

$O(|V|)$. We denote one minimal set of t as $min(t)$. It is clear that the number of paths within P^*, which leads to terminal node $r(t)$, is at most $|min(t)|$.

A MIP formulation of finding P^* is defined as follows. We define some terms and notations. For each terminal node t, where $r(t) > 0$, we create one goal for each element in $min(t)$ and denote the set of goals created from t as $goals(t)$. For each goal $g \in goals(t)$, we denote the corresponding element in $min(t)$ as $h(g)$ and the value of g as $r(g)$, which equals $r(t)$. Let $GS = \bigcup_{t \in T | r(t) > 0} goals(t)$. For each $g \in GS$, $x(g)$ is a 0/1 decision variable that denotes whether g is active ($x(g) = 1$ means g is active). For each goal $g \in GS$ and for each edge (u, u'), $x(g, (u, u'))$ is a 0/1 decision variable that denotes that the edge (u, u') is used for goal g.

Definition 5 (MIP formulation of CSG for a MTZDD). *The problem of finding P^* can be modeled as follows.*

$$\begin{aligned}
\max \quad & \textstyle\sum_{g \in GS} x(g) \cdot r(g) \\
s.t. \quad & \forall g \in GS, x(g) = x(g, h(g)), \text{ --- (i)} \\
& \forall t \in T, \text{ where } r(t) > 0, \forall g \in goals(t), \\
& \quad x(g) = \textstyle\sum_{u \in Pa(t)} x(g, (u, t)), \text{ --- (ii)} \\
& \forall u \in V \setminus \{u_0\}, \forall g \in GS, \\
& \quad x(g, h(u)) + x(g, l(u)) \\
& \quad = \textstyle\sum_{u' \in Pa(u)} x(g, (u', u)), \text{ --- (iii)} \\
& \forall i \in A, \textstyle\sum_{u \in nodes(i)} \sum_{g \in GS} x(g, h(u)) \le 1, \text{ --- (iv)} \\
& x(\cdot), x(\cdot, \cdot) \in \{0, 1\}.
\end{aligned}$$

Constraint (i) ensures that if goal g is selected, its required high-edge must be selected. Constraint (ii) ensures if one of its goal g is selected for terminal node t, then an edge must exist that is included in a path for g. Constraint (iii) ensures that for each non-terminal, non-root node, correct paths are created (the numbers of inputs and outputs must be the same). Constraint (iv) ensures that one agent can be included in at most one path. In this MIP formulation, the number of constraints is linear to the number of nodes in a MTZDD.

Example 3. We consider a MIP problem of a MTZDD representation in Example 2.

First, we create a minimal set for a non-zero-terminal node. As shown in Figure 4, we denote each non-zero terminal node as t_1, t_2, and t_3 from the left. No high-edge directly points to t_1, but using backtracking search, we find three high-edges labeled $h(g_1)$, $h(g_2)$, and $h(g_3)$ as elements of $min(t_1)$. t_2 has both incoming high-edge and low-edge, and so we obtain $min(t_2) = \{h(g_4), h(g_5)\}$. t_3 only has an incoming high-edge, i.e., $min(t_3) = \{h(g_6)\}$. Thus, we obtain $\{g_1, \ldots, g_6\}$ as GS.

Next, we solve a MIP defined by Definition 5 and obtain optimal set of paths P^* that consists of two paths that represent coalitions $\{1, 2\}$ and $\{3, 4\}$ (Figure 5). The value of P^* is calculated as 10.

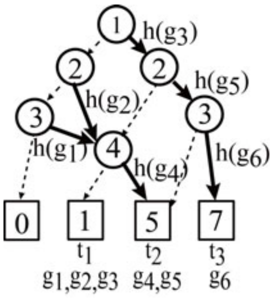


Fig. 4. GS in Example 3

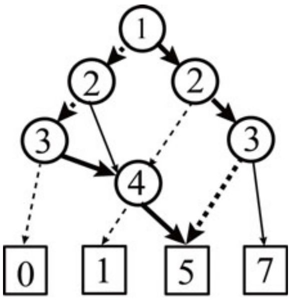


Fig. 5. P^* in Example 3

5 Core-Related Problems

We show that a MTZDD representation can efficiently solve core-related problems.

5.1 Core-Non-emptiness

We show a LP formulation of the problem of checking core-non-emptiness in a MTZDD representation. By assuming that the value of an optimal coalition structure $V(CS^*)$ is given, checking the core-non-emptiness for CS^* can be done in a polynomial time in the number of nodes in a MTZDD. We represent the payoff of an agent as the distance of its high edge. For terminal node t, its shortest distance to the root node represents the minimal total reward of coalition S, where $v(S) = r(t)$. The non-blocking condition requires that, for each terminal node t, its shortest distance to the root node is at least $r(t)$. Let $dis(u)$ represent the shortest distance from root node u_0 to node u.

Definition 6 (LP formulation for the least core). *The following LP formulation gives an element in the ϵ-core:*

$$\begin{aligned}
\min \quad & \epsilon \\
s.t. \quad & dis(u_0) = 0, \\
& \textstyle\sum_{i \in A} \pi_i = V(CS^*), \\
& \forall u \in V \setminus \{u_0\} \cup T,\ \forall u' \in Pa(u), \\
& \quad dis(u) \le dis(u') + \pi_{agent(u')} \qquad \text{— } \textit{if } (u', u) \in H, \\
& \quad dis(u) \le dis(u') \qquad \text{— } \textit{otherwise}, \\
& \forall t \in T,\ dis(t) + \epsilon \ge r(t).
\end{aligned}$$

Theorem 5. *Using a MTZDD representation, determining whether the core is non-empty can be done in polynomial time in the number of nodes in a MTZDD, assuming that the value of an optimal coalition structure $V(CS^*)$ is given.*

Proof. To examine whether the core is non-empty, it is sufficient to check whether a solution of the above LP problem is 0 or less. The LP can be solved in polynomial time in the number of its constraints, which is given as $2|V| + |T|$. □

5.2 Core-Membership

For given payoff vector π, we need to examine whether π is in the core. Assuming the value of an optimal coalition $V(CS^*)$ is given, checking the feasibility condition is easy. For each terminal node $t \in T$, where $r(t) > 0$, similar to checking the core-non-emptiness, the non-blocking condition holds if the shortest path $dis(t)$ from the root node to terminal node t is the value of path $r(t)$ or more.

Theorem 6. *By using a MTZDD representation, determining whether a payoff vector π is in the core can be done in $O(|V|)$ time, assuming the value of an optimal coalition structure $V(CS^*)$ is given.*

Proof. A MTZDD is a single-source directed acyclic graph (DAG). Thus, for each terminal node, we can find the distance from the root node using the DAG-shortest paths algorithm, which requires $O(|V| + |H| + |L|)$ time. In a MTZDD, since each internal node has one high-edge and one low-edge, $|V| = |H| = |L|$ holds. It requires $O(|V|)$ time. □

5.3 The Cost of Stability

Definition 7 (LP formulation for the cost of stability). *The following LP formulation gives the cost of stability Δ:*

$$\begin{aligned}
\min\quad & \Delta, \\
s.t.\quad & dis(u_0) = 0, \\
& \textstyle\sum_{i \in A} \pi_i = V(CS^*) + \Delta, \\
& \forall u \in V \setminus \{u_0\} \cup T,\ \forall u' \in Pa(u), \\
& \qquad dis(u) \le dis(u') + \pi_{agent(u')} \qquad \text{— if } (u', u) \in H, \\
& \qquad dis(u) \le dis(u') \qquad \text{— otherwise}, \\
& \forall t \in T,\ dis(t) \ge r(t).
\end{aligned}$$

Theorem 7. *By using a MTZDD representation, the cost of stability can be obtained in polynomial time in the number of nodes in a MTZDD, assuming that the value of optimal coalition structure $V(CS^*)$ is given.*

Proof. The cost of stability can be obtained by solving the above LP formulation. The LP can be solved in polynomial time in the number of its constraints, which is given as $2|V| + |T|$. □

6 Experimental Simulations

While core-related problems can be solved in polynomial time, we need to solve a MIP problem for CSG. In this section, in order to show that our proposed CSG algorithm is reasonably efficient and scalable, we experimentally evaluate its performance, in comparison with the MIP formulation using a SCG representation [18]. The simulations were run on a Xeon E5540 processor with 24-GB RAM. The test machine ran Windows Vista Business x64 Edition SP2. We used CPLEX 12.1, a general-purpose mixed integer programming package.

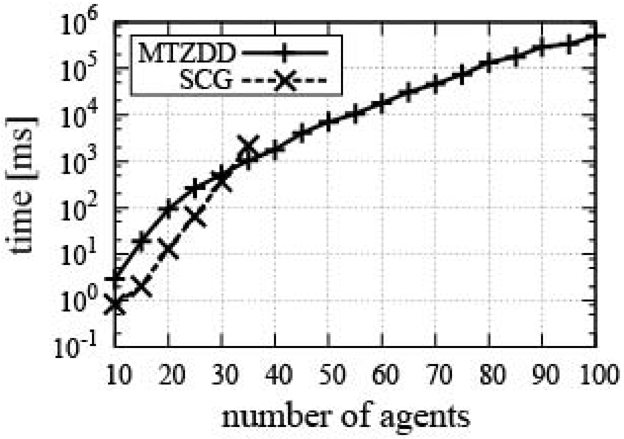


Fig. 6. Computation time

We generated problem instances with 5 different groups of symmetric agents. First, we created a set of *abstract rules*. Each rule specifies the required number of agents in each group, which is generated using a decay distribution as follows. Initially, the required number of agents in each group is set to zero. First, we randomly chose one group and incremented the required number of agents in it by one. Then we repeatedly chose a group randomly and incremented its required number of agents with probability α until a group is not chosen or the required number of agents exceeds the limit. In this simulation, we used $\alpha = 0.55$. For each rule, we randomly chose an integer value between 1 and 10 as the value of the coalition. The number of abstract rules is set equal to the number of agents. Then we translated these abstract rules into a MTZDD representation. The MIP formulation using a SCG representation is also generated from these abstract rules. Figure 6 shows the median computation times for solving the generated 50 problem instances.

When $n \leq 30$, a SCG representation is more efficient than a MTZDD representation for finding an optimal coalition structure, while a MTZDD representation eventually outperforms the SCG for $n > 30$. When the number of coalitions in a SCG is relatively small, the MIP formulation of a SCG representation is simple and CPLEX can reduce the search space efficiently. However, the number of coalitions in a SCG grows exponentially based on the increase of the number of agents/rules. For example, when k agents among n symmetric agents are required, the number of coalitions in a SCG is given as $\binom{n}{k}$. Thus, for $n \geq 40$, generating problem instances becomes impossible due to insufficient memory. On the other hand, the number of nodes in a MTZDD grows linearly based on the increase of the number of agents/rules. As a result, the computation time for a MTZDD representation grows more slowly compared to the SCG.

7 Conclusion

We developed a new representation scheme by integrating a ZDD data structure and an existing compact representation scheme called SCG. A ZDD is an efficient data structures applied in various domains in AI. We showed that our MTZDD representation scheme (i) is fully expressive, (ii) can be exponentially more concise than SCG representation, (iii) can solve core-related problems in polynomial time in the number of nodes, and (iv) can solve a CSG problem reasonably well by utilizing a MIP formulation.

Future work includes overcoming the complexity of solving other problems including the Shapley value in coalitional games. We will also consider applying BDD/ZDD-based graphical representation for characteristic functions in non-transferable utility coalitional games. Furthermore, we hope to implement the optimization routines used in this paper as general purpose operations equipped in a MTZDD package.

Acknowledgements. This research was partially supported by Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research (C), 23500166, 2011. The authors would like to thank anonymous reviewers for helpful discussions.

References

1. Aadithya, K., Michalak, T., Jennings, N.: Representation of Coalitional Games with Algebraic Decision Diagrams. In: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), pp. 1121–1122 (2011)
2. Airiau, S., Sen, S.: On the stability of an optimal coalition structure. In: Proceeding of the 19th European Conference on Artificial Intelligence (ECAI 2010), pp. 203–208 (2010)
3. Akers, S.B.: Binary decision diagrams. IEEE Transactions on Computers C-27(6), 509–516 (1978)
4. Aumann, R.J., Dreze, J.H.: Cooperative games with coalition structures. International Journal of Game Theory 3, 217–237 (1974)
5. Bachrach, Y., Elkind, E., Meir, R., Pasechnik, D., Zuckerman, M., Rothe, J., Rosenschein, J.S.: The Cost of Stability in Coalitional Games. In: Mavronicolas, M., Papadopoulou, V.G. (eds.) SAGT 2009. LNCS, vol. 5814, pp. 122–134. Springer, Heidelberg (2009)
6. Berghammer, R., Bolus, S.: Problem Solving on Simple Games via BDDs. In: Proceedings of the 3rd International Workshop on Computation Social Choice, COMSOC 2010 (2010)
7. Bryant, R.E.: Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams. ACM Computing Surveys 24(3), 293–318 (1992)
8. Conitzer, V., Sandholm, T.: Complexity of constructing solutions in the core based on synergies among coalitions. Artificial Intelligence 170(6), 607–619 (2006)
9. Elkind, E., Goldberg, L.A., Goldberg, P.W., Wooldridge, M.: A tractable and expressive class of marginal contribution nets and its applications. In: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), pp. 1007–1014 (2008)

10. Engel, Y., Wellman, M.P.: Generalized value decomposition and structured multiattribute auctions. In: Proceedings of the 8th ACM Conference on Electronic Commerce (EC 2007), pp. 227–236 (2007)
11. Faigle, U., Fekete, S.P., Hochstättler, W., Kern, W.: On the complexity of testing membership in the core of min-cost spanning tree games. International Journal of Game Theory 26, 361–366 (1997)
12. Håstad, J.: Clique is hard to approximate within $n^{1-\epsilon}$. Acta Mathematica 182, 105–142 (1999)
13. Ieong, S., Shoham, Y.: Marginal contribution nets: a compact representation scheme for coalitional games. In: Proceedings of the 5th ACM Conference on Electronic Commerce (EC 2005), pp. 193–202 (2005)
14. Knuth, D.: The art of computer programming 4, Fascicle 1 (2009)
15. Loekito, E., Bailey, J.: Fast mining of high dimensional expressive contrast patterns using zero-suppressed binary decision diagrams. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006), pp. 307–316 (2006)
16. Minato, S.: Zero-suppressed bdds for set manipulation in combinatorial problems. In: Proceedings of the 30th Design Automation Conference (DAC 1993), pp. 272–277 (1993)
17. Minato, S.: Arithmetic boolean expression manipulator using bdds. Formal Methods in System Design 10(2/3), 221–242 (1997)
18. Ohta, N., Conitzer, V., Ichimura, R., Sakurai, Y., Iwasaki, A., Yokoo, M.: Coalition Structure Generation Utilizing Compact Characteristic Function Representations. In: Gent, I.P. (ed.) CP 2009. LNCS, vol. 5732, pp. 623–638. Springer, Heidelberg (2009)
19. Sandholm, T., Larson, K., Andersson, M., Shehory, O., Tohmé, F.: Coalition structure generation with worst case guarantees. Artificial Intelligence 111(1-2), 209–238 (1999)
20. Shrot, T., Aumann, Y., Kraus, S.: On agent types in coalition formation problems. In: Proceedings of the 9th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), pp. 757–764 (2010)
21. Uckelman, J., Chevaleyre, Y., Endriss, U., Lang, J.: Representing utility functions via weighted goals. Mathematical Logic Quarterly 55(4), 341–361 (2009)

Environment Characterization for Non-recontaminating Frontier-Based Robotic Exploration

Mikhail Volkov, Alejandro Cornejo, Nancy Lynch, and Daniela Rus

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology, Cambridge, MA 02139
{mikhail,acornejo,lynch,rus}@csail.mit.edu

Abstract. This paper addresses the problem of obtaining a concise description of a physical environment for robotic exploration. We aim to determine the number of robots required to clear an environment using non-recontaminating exploration. We introduce the medial axis as a configuration space and derive a mathematical representation of a continuous environment that captures its underlying topology and geometry. We show that this representation provides a concise description of arbitrary environments, and that reasoning about points in this representation is equivalent to reasoning about robots in physical space. We leverage this to derive a lower bound on the number of required pursuers. We provide a transformation from this continuous representation into a symbolic representation. Finally, we present a generalized pursuit-evasion algorithm. Given an environment we can compute how many pursuers we need, and generate an optimal pursuit strategy that will guarantee the evaders are detected with the minimum number of pursuers.

Keywords: swarm robotics, frontier-based exploration, distributed pursuit-evasion, environment characterization, mathematical morphology, planar geometry, graph combinatorics, game determinacy.

1 Introduction

This paper deals with the problem of developing a concise representation of a physical environment for robotic exploration. We address a specific type of exploration scenario called pursuit-evasion. In this scenario, a group of robots are required to sweep an unexplored environment and detect any intruders that are present. Pursuit-evasion is an example of non-recontaminating exploration, whereby an initially unexplored and contaminated region is cleared while ensuring that the cleared region does not become contaminated again. Pursuit-evasion is a useful model for many applications such as surveillance, security and military operations. Aside from the classical pursuit-evasion problem, there are other scenarios that motivate non-recontaminating exploration. One example is cleaning up an oil spill in the ocean using robots, where the oil can leak into previously decontaminated water whenever part of the frontier is not guarded by a robot. Another example is a disaster response scenario such as after an earthquake, where a group of robots is required to locate all survivors in some inaccessible area, while the survivors are possibly moving in the environment.

1.1 Related Work

Visibility-based pursuit-evasion in continuous two-dimensional space was first introduced in [25]. Frontier-based exploration was introduced in [29] and extended to multiple robots in [30]. In [7] the authors consider limited visibility frontier-based pursuit-evasion in non-polygonal environments, making use of the fact that not allowing recontamination means we do not need to store a map of the environment; here a distributed algorithm is presented which works by locally updating the frontier formed by the sensor footprints of the robots. Another distributed model was more recently considered in [3]. Limited visibility was also considered in [23] which presents an algorithm for clearing an unknown environment without localization. In [8] the problem was considered for a single searcher in a known environment. Generally speaking, these algorithms do the correct thing locally, but do not rely on, or make any guarantees for, a global description of an environment. As such, they may not be guaranteed to terminate.

Environment characterization for pursuit-evasion has been considered for polygonal spaces. In [10], [12] pursuit-evasion in connected polygonal spaces is investigated, and tight bounds derived on the number of pursuers necessary based on the number of polygon edges and holes. In [19] basic environment characterization is established, by means of a general decomposition concept based on a finite complex of conservative cells. In [28] similar ideas were explored, and several metrics proposed for primitive characterization of polygonal spaces. We highlight that these studies considered robots equipped with infinite range visibility sensors deployed in polygonal spaces. Consequently the scope of environment characterization was limited. By contrast we consider limited visibility sensors, and we do not require the environment to be polygonal.

The medial axis has previously been studied in the context of robotic navigation. For example, in [11], [13], [27] the medial axis was used as a heuristic for probabilistic roadmap planning. One of the key features that makes the medial axis attractive for such applications is that it captures the connectivity of the free space. In this work, we use the medial axis to similarly capture the underlying topological and geometric properties of our environment, and use it to transform the problem into a graph formulation.

Pursuit-evasion on graphs that are representations of some environment goes back as early as [20], [21]. Randomized pursuit strategies on a graph are considered in [1]. Roadmap-based pursuit-evasion is considered in [14] and [22] where the pursuer and evader share a map and act according to different rules. In [22] a graph-based representation of the environment is used to derive heuristic policies in various scenarios. More recently, [17] presents a graph-based approach to the pursuit-evasion problem whereby agents use blocking or sweeping actions to detect all intruders in the environment. In [15] and [16] the more general graph variant of the problem was reduced to a tree by blocking edges. Although discrete graph-based models offer termination and correctness guarantees, they assume the world is suitably characterized and make no reference to the underlying physical geometry of the environment that is being represented.

The aim of this paper is to bridge the different levels of abstraction that work to date has been grounded in. We make use of the medial axis as a configuration space to derive a robust representation of a continuous environment that captures its underlying properties. We use the medial axis leveraged by existing exploration models to build a concise representation of arbitrary environments in continuous two-dimensional space.

We then transform this representation into the discrete domain. First, this allows us to calculate bounds on the number of pursuers required to clear an environment, and to provide termination guarantees for existing pursuit-evasion algorithms. Second, this establishes an application platform for existing graph-based algorithms making them applicable to continuous environment descriptions.

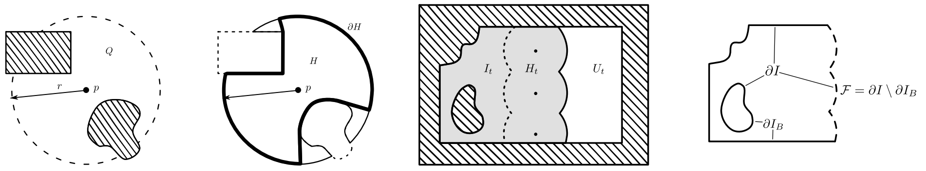
This paper is organized as follows. Section 2 provides a formal model for non-recontaminating exploration and states the problem being addressed. Section 3 introduces the medial axis as a configuration space and shows that reasoning about points in this space is equivalent to reasoning about robots in the physical world. We formalize the notion of width, corridors and junctions and derive bounds on the number of robots required to traverse a junction. Section 4 presents a transformation from this continuous configuration space into a symbolic representation in the discrete domain. Finally, we present an optimal pursuit-evasion algorithm, and prove correctness for this algorithm.

2 Problem Formulation

We now present a formal model of the problem we are addressing. Our model builds on the notation and terminology introduced in [7]. We have a team of n exploration robots deployed in the Euclidean plane $\mathbb{R}^2$. Each robot is equipped with a holonomic (uniform in all orientations) sensor that records a line of sight perception of the environment within a maximum sensing radius r. We assume that two robots can reliably communicate if they are within line of sight of each other and if the distance between their positions is less than or equal to $2r$.

The position of a robot is constrained to be within some *free region* Q, which is a closed compact subset of $\mathbb{R}^2$. The *obstacle region* B makes up the rest of the world, and is defined as the complement of Q. In this paper we require both Q and B to be connected spaces, which means there are no holes in the environment. We define the *obstacle boundary* ∂B as the oriented boundary of the obstacle region (which by definition is the oriented boundary of the free region).

We assume a continuous time model, i.e. time $t \in \mathbb{R}_{\geq 0}$. Let H_t^i be the holonomic *sensor footprint* of robot i at time t, which is defined as the subset of Q that is within direct line of sight of robot i and within distance r of robot i. Formally, if $p \in \mathbb{R}^2$ is the position of robot i at time t, then $H_t^i = \{x \in Q \ \mid d(p,x) \leq r \wedge \forall y \in [px], y \in Q\}$ where $d(x,y)$ is the Euclidean distance between x and y (see Fig. 1a). Let H_t be the union of the sensor footprints of all robots at some time t, given by $H_t = \bigcup_{i=0}^{n} H_t^i$. This corresponds to the region being sensed by the robots at time t. We define the *inspected region* $I_t \subseteq Q$ as the union at time t of all previously recorded sensor footprints, given by $I_t = \{p \in \mathbb{R}^2 \mid \exists t_0 \in [0,t]$ such that $p \in H_{t_0}\}$. The *contaminated region* (or unexplored region) U_t is defined as the free space that has not been inspected by time t, given by $U_t = Q \setminus I_t$ (see Fig. 1b). Note that at time $t = 0$ the contaminated region is given by $Q \setminus H_0$. We define the *cleared region* $C_t \subseteq I_t$ as the inspected region that is not currently being sensed, given by $C_t = I_t \setminus H_t$. We say that *recontamination* occurs at time t if the cleared region C_t comes in contact with the contaminated region U_t (we understand two regions to be in contact if the intersection of their closure is non-empty, i.e. $\mathrm{cl}(C_t) \cap \mathrm{cl}(U_t) \neq \emptyset$).



(a) Sensor Footprint **(b)** Inspected region **(c)** Boundaries

Fig. 1. Fig. 1a shows a robot located at some position p within the free region Q. The sensor footprint H and its oriented boundary ∂H are shown on the right. Fig. 1b shows the inspected region I_t and the contaminated region U_t at some time t. Fig. 1c shows the inspected region boundary ∂I, the inspected obstacle boundary ∂I_B and the frontier $\mathcal{F}$.

When time is clear from context, we understand I and U to mean the current inspected region and the current contaminated region, respectively. We define the *inspected region boundary* ∂I as the oriented boundary of the inspected region I. We define the *inspected obstacle boundary* ∂I_B as the intersection of the inspected region boundary and the obstacle boundary, given by $\partial I_B = \partial I \cap \partial B$. We define the *frontier* $\mathcal{F}$ as the free (non-obstacle) boundary of the inspected region, given by $\mathcal{F} = \partial I \setminus \partial I_B$ (see Fig. 1c). Observe that by definition the frontier $\mathcal{F}$ separates the free region Q into the inspected region I and the contaminated region U. Observe also that the frontier need not be connected, and is in general the union of one or more disjoint maximally connected arcs. We understand the frontier of a group of robots $\mathcal{F}' \subseteq \mathcal{F}$ to mean a single maximally connected arc of the total frontier formed by the exterior boundary of the sensor footprints of that group of robots.

The goal of exploration algorithms is to inspect the entire free region. For non-recontaminating exploration the goal is to inspect the entire fee region without admitting recontamination. In both cases we say that an environment has been successfully explored if $I_t = Q$ at some time t. In this paper we deal specifically with non-recontaminating exploration and present an algorithm that is guaranteed to explore a space without admitting recontamination.

An algorithm for exploration relies on robots to "expand" the frontier boundary until the entire free region becomes inspected. However, in a non-recontaminating exploration algorithm the goal is not only to expand, but also to "guard" the frontier, ensuring that the inspected region does not become contaminated again. This difference makes non-recontaminating exploration more restrictive than conventional exploration. For example, observe that regardless of the size of the sensing radius of the robots (as long as $r > 0$), or the properties of the world (as long as Q is a connected space), a single robot can always explore the world. However, in non-recontaminating exploration this is not true in general. Informally speaking, if the "width" of the corridors in the free region is larger than the sensing radius of a robot, then it should appear obvious that a single robot cannot simultaneously expand and guard the frontier to inspect the entire free region.

Consider the simple rectangular free region Q shown in Fig. 1b. We can reason that if the width of Q is less than the sum of the sensor diameters of the n robots, then the environment can be explored without admitting recontamination. However, even in this simple example it is not completely clear what is meant by width. Notice that if we consider width to be the distance from the left to the right border then this reasoning fails — in this case width would specifically mean the smaller of the two dimensions. So it is already non-trivial how to characterize a very simple environment, and things become much more complicated in non-rectangular environments.

In this paper we study the relationship between an environment Q, the sensor radius r, and the number of robots n required for non-recontaminating exploration of Q. Intuition tells us that corridor width and junctions are important features. We formalize the notion of corridors and junctions and present a general method for computing a configuration space representation of the environment that captures this intuition. We show that this representation provides a concise description of arbitrary environments.

A canonical example of non-recontaminating exploration is pursuit-evasion. In this scenario there is a group of robot pursuers and a group of robot evaders deployed in the free region Q. The evaders are assumed to be arbitrarily small and fast. The goal of the pursuers is to catch the evaders (by detecting their presence within the sensor footprint), and the goal of the evaders is to avoid getting caught. Whenever part of the frontier is not being guarded by the pursuers, the evaders can move undetected from the contaminated region to the previously inspected region, thereby recontaminating it.

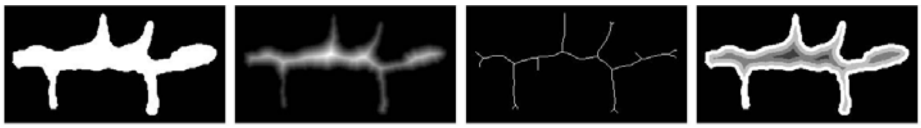
3 Environment Analysis

In this section we present the medial axis as a configuration space and show that reasoning about points in this configuration space is equivalent to reasoning about robots in physical space. First, we establish the necessary geometric framework, accompanied by a series of definitions and claims. Second, we introduce an exploration model in this configuration space and justify that it allows us to reason about the physical movement of the robots in the environment.

3.1 Environment Geometry

The *distance transform* is a mapping $\mathcal{D} : \mathbb{R}^2 \to \mathbb{R}$ where $\mathcal{D}(x) = \min_{y \in B} \{d(x, y)\}$ and $d(x, y)$ is the Euclidean distance between x and y (extending definition in [5] to the continuous domain) (see Fig. 2b). Observe that by definition if $x \notin Q$ then $\mathcal{D}(x) = 0$. The distance transform of a point $x \in Q$ captures the notion of “undirected width” of a region around a point x in free space, that is we get a measure of how wide or narrow a region is without being explicit about orientation.

The *medial axis* or *skeleton* S of a free space is defined as the locus of the centers of all maximal inscribed circles in the free space [4] (see Fig. 2c). Equivalently, the skeleton can be defined as the locus of quench points of a fire that has been set to a grass meadow at all points along its boundary [2], [24]. The skeleton captures the topology of the free space, and aids us in determining which parts of an environment should be considered “corridors” and which parts should be considered “junctions” of multiple corridors.



(a) Environment image **(b)** Distance transform **(c)** Skeleton **(d)** Relief map

Fig. 2. The environment is represented by a binary image in Fig. 2a. Fig. 2b shows the distance transform $\mathcal{D}$ of the environment. Fig. 2c shows the skeleton S. Fig. 2d shows the relief map quantization. The relief contours indicate multiples of the sensing radius r.

The *degree* of a point $x \in S$ is given by the function $\theta : S \to \mathbb{N}_{>0}$ which maps every point on the skeleton to a natural number k. Specifically, we define a point $x \in S$ to have degree $\theta(x) = k$ if there exists an $a \in \mathbb{R}_{>0}$ such that $\forall \varepsilon \in (0, a]$ a circle centered at x of radius ε intersects the skeleton S at exactly k points.

Borrowing notation from [6], we use the degree of a point $x \in S$ to distinguish between three types of points on the skeleton: corridor points, end points and junction points. Specifically, for a point $x \in S$, we say x is an *end point* if $\theta(x) = 1$, x is a *corridor point* if $\theta(x) = 2$, and x is a *junction point* if $\theta(x) > 2$ (see Fig. 3a). We refer to a continuous arc of corridor points on the skeleton simply as a *corridor*.

An alternative definition for $\theta(\cdot)$ can be stated as follows. For a point $x \in S$ let C be the maximal inscribed circle centered at x, and let G be the intersection of this circle with the obstacle boundary, given by $G = C \cap \partial B$. (Observe that by definition C has radius $\mathcal{D}(x) \neq 0$, and since C is maximal, G is non-empty.) Then $\theta(x)$ is defined as the number of maximally connected arcs in G. This definition for for $\theta(\cdot)$ is equivalent to the previous one [4], [9], [18].

Let $G_1, G_2, \ldots, G_{\theta(x)}$ be the set of maximally connected arcs of G. Note that in most cases these arcs are in fact just single points, which corresponds to the intuitive notion of the circle being tangent to the boundary at these points. A cursory glance reveals that this is the case for most corridor points and junction points. For end points that lie on the obstacle boundary, the tangent point coincides with the end point itself. However, the generality is necessary in a few special cases, such as end points of regions that taper off in a sector. In these cases the maximal inscribed circle C will be tangent to the obstacle boundary at a continuous arc segment of points. In order to simplify the discussion we define the tangent points $\tau_1(x), \tau_2(x), \ldots, \tau_{\theta(x)}(x)$ of a point $x \in S$ as the midpoints of the tangent arcs $G_1, G_2, \ldots, G_{\theta(x)}$ (see Fig. 3b).

We define a *boundary wall* as a maximally connected arc segment of the obstacle boundary ∂B that does not contain a tangent point of any end point. Formally $\partial B_0 \subset \partial B$ is a boundary wall if it is a maximally connected arc segment such that $\forall e \in S \mid \theta(e) = 1, \tau(e) \notin \partial B_0$.

Lemma 1. *For a junction point $j \in S$, the $\theta(j)$ tangent points of j are located on $\theta(j)$ distinct boundary walls.*

Proof. See full version [26].

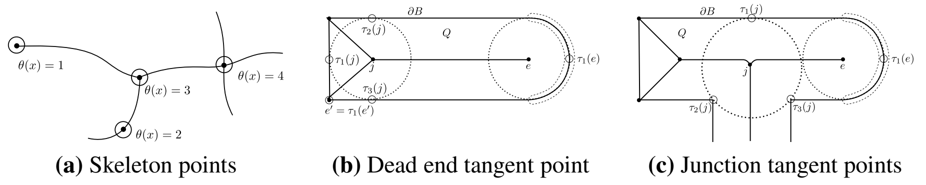


Fig. 3. Fig. 3a shows the three types of points on the skeleton: end points, corridor points, and junction points. A corridor is a continuous arc of corridor points. Fig. 3b shows three points on the skeleton e, e', j and their respective tangent points. For end point e the tangent point $\tau_1(e)$ is the midpoint of the tangent arc segment shown (dotted outline). For end point e' the tangent point $\tau_1(e')$ coincides with the end point itself. For junction point j there are $\theta(j) = 3$ tangent points τ_1, τ_2, τ_3. Fig. 3c shows a junction point j of degree $\theta(j) = 3$. All 3 tangent points τ_1, τ_2, τ_3 are located on distinct boundary walls.

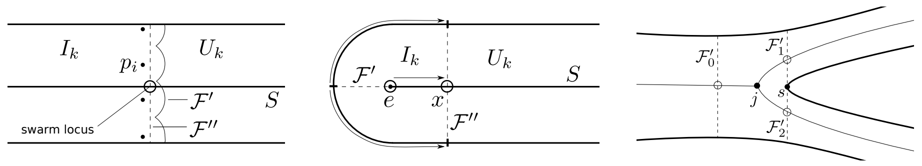
3.2 Exploration Model

We now show how we can use the preceding definitions and geometric claims to form a model for frontier-based exploration and establish an equivalence between the the medial axis configuration space and the physical environment.

We claim that reasoning about a single point moving along the skeleton allows us to reason about a group of robots that form a frontier with their end to end sensor footprints moving through physical space. We call this point the *swarm locus*. By definition a corridor point $x \in S$ has exactly two tangent points. For a swarm locus stationed at x we call these two points the frontier *anchor points*. The frontier of a group of robots is represented by a corresponding frontier formed by two line segments joining the swarm locus to its anchor points. A group of robots engaged in non-recontaminating exploration will form a frontier arc subtended between two obstacle boundary walls in physical space; the frontier arc separates the inspected region on one side from the contaminated region on the other side. Our abstraction allows us to reason in similar terms: a swarm locus stationed at a point x on the corridor of the skeleton will similarly form a frontier arc consisting of two line segments subtended between two obstacle boundary walls; the frontier arc transposed onto Q likewise separates the inspected region from the contaminated region (see Fig. 4a). We understand the frontier of a swarm locus to mean the frontier $\mathcal{F}' \subseteq \mathcal{F}$ of a group of robots represented by a swarm locus stationed at a point $x \in S$.

Note that we are making a simplifying abstraction in representing the frontier $\mathcal{F}' \subseteq \mathcal{F}$ of a group of n_0 robots by two end to end line segments subtended between two obstacle boundary walls. Observe that as n grows, the abstraction becomes more accurate as the periodic protrusion of the frontier due to the curvature of sensor footprints becomes finer-grained and less prominent with respect to its length. In general, this abstraction is justified as we are usually interested in characterizing environments where $n \gg 0$.

For the purposes of introducing the exploration model we assume that the swarm locus always begins at an end point. (Note that this assumption only serves to simplify



(a) Swarm locus **(b)** Initial frontier **(c)** Split frontier

Fig. 4. Fig. 4a shows a group of robots at positions p_i forming a frontier $\mathcal{F}'$ with the exterior boundary of their sensor footprints. Superimposed is the corresponding swarm locus stationed at a point $x \in S$ forming a frontier $\mathcal{F}''$ with two line segments subtended between two obstacle boundary walls. Fig. 4b shows the initial frontier $\mathcal{F}'$ of a swarm locus stationed at an end point e, separating the environment into $I_0 = \emptyset$ and $U_0 = Q$. As the swarm locus moves along the skeleton to reach a point $x \in S$, it forms a frontier $\mathcal{F}''$. The swarm locus has swept across the environment and cleared the region to the left of $\mathcal{F}''$. Fig. 4c shows the configuration of a split frontier. A swarm locus is traversing a junction point j with $\theta(j) = 3$. The ingoing frontier $\mathcal{F}'_0$ splits, producing 1 split point s and 2 outgoing frontiers $\mathcal{F}'_1, \mathcal{F}'_2$.

the discussion, and can be removed easily by introducing several special cases.) From the definition of the degree of a point on the skeleton, a maximal inscribed circle C centered at an end point $e \in S$ will be tangent to the obstacle boundary at a single point $\tau(e)$. Thus both anchor points are the same point $\tau(e)$ and the frontier $\mathcal{F}' \subseteq \mathcal{F}$ of a swarm locus stationed at e is formed by two identical line segments $[e\ \tau(e)]$. In this configuration, $\mathcal{F}'$ separates the environment Q into the inspected region $I_0 = \emptyset$ and the contaminated region $U_0 = Q$, corresponding to the fact that the swarm locus has not yet explored any of the environment. As the swarm locus starts moving along the skeleton, the anchor points will move along ∂B on either side of the corridor and the frontier will "sweep" across the environment. The frontier now separates Q into two disjoint nonempty regions. The inspected region begins growing, while the contaminated region begins shrinking, corresponding to the fact that the robots have begun clearing the environment (see Fig. 4b).

Moving Through Corridors. We define the *relief map* $\mathcal{R} : \mathbb{R}^2 \to \mathbb{N}$ as the quantization of the distance transform using the sensing radius r, given by $\mathcal{R}(x) = \lceil \mathcal{D}(x)/r \rceil$ (see Fig. 2d). The relief map uses the distance transform to similarly capture the notion of width, expressing the same information in terms of the number of robots required at a point x to reach the closest point on the obstacle boundary.

Lemma 2. *A group of n_0 robots represented by a swarm locus that reaches a corridor point $x \in S$ prevents recontamination if and only if $n_0 \geq \mathcal{R}(x)$.*

Proof. See full version [26].

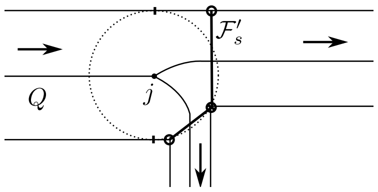
Corollary 1. *A group of n_0 robots represented by a swarm locus moving along a corridor $G = [a\ b] \subset S$ prevents recontamination at all points $x \in G$ if and only if $n_0 \geq \max_{x \in G} \{\mathcal{R}(x)\}$.*

When a swarm locus reaches an end point, the situation is the reverse of that at the beginning. From the definition of the degree of a point on the skeleton, a maximal inscribed circle C centered at an end point $e \in S$ will be tangent to the obstacle boundary at a single point $\tau(e)$. Thus both anchor points are the same point $\tau(e)$ and the frontier $\mathcal{F}' \subseteq \mathcal{F}$ of a swarm locus stationed at e is formed by two identical line segments $[e\,\tau(e)]$. In this configuration, $\mathcal{F}'$ separates the environment Q into the inspected region I_t and some part of the contaminated region $\emptyset$, corresponding to the fact that the swarm locus has cleared a particular corridor. In the case where there is only one swarm locus, $\mathcal{F}'$ separates the environment Q into the inspected region $I_t = Q$ and the entire contaminated region $U_t = \emptyset$, corresponding to the fact that the entire environment has been cleared.

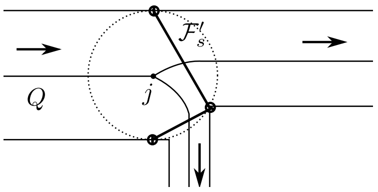
Traversing Junctions. When a group of robots reaches a junction in physical space, they should split and explore the outgoing junction corridors separately. If the robots do not split then recontamination will occur due to any of the unattended outgoing corridors coming in contact with the inspected region. Upon reaching a physical junction with some number of outgoing corridors, a single group of robots forms more than one group of robots with that number of disjoint maximally connected arcs making up the exterior boundary of their sensor footprints. Correspondingly, we define the frontier $\mathcal{F}'_0 \subseteq \mathcal{F}$ of a group of robots to *split* when the exterior boundary of the sensor footprints of the robots is no longer connected and becomes the union of two or more disjoint maximally connected arcs. Thus a group of robots splits when their frontier splits. For a junction point j with $\theta(j) - 1$ outgoing corridors, the junction is considered *traversed* when the frontier $\mathcal{F}'_0 \subseteq \mathcal{F}$ of the group of robots splits to form $\theta(j) - 1$ outgoing frontiers $\mathcal{F}'_1, \mathcal{F}'_2, \ldots, \mathcal{F}'_{\theta(j)-1} \subset \mathcal{F}$.

Observe that since the total frontier is at all times given by $\mathcal{F} = \partial I \setminus \partial I_B$, if one or more new disjoint maximally connected frontier arcs form, then by necessity one or more new disjoint maximally connected inspected obstacle boundary arcs also form. At the time t_0 that a frontier $\mathcal{F}'_0 \subseteq \mathcal{F}$ of a group of robots splits to form $\theta(j) - 1$ frontiers, it will have $\theta(j) - 2$ points of contact with the obstacle boundary. We call a frontier $\mathcal{F}'_0$ in such a configuration a *split frontier* and we call these points *split points* (see Fig. 4c). For $t > t_0$ the split points on the obstacle boundary grow into the $\theta(j) - 2$ new disjoint maximally connected inspected obstacle boundary arcs.

Split Frontier Bounds. We establish the lower bound as follows. To traverse a junction we require a split frontier $\mathcal{F}'_s$ to be formed with $\theta(j) - 2$ split points. The ingoing frontier $\mathcal{F}'_0$ subtends between two boundary walls $\partial B_1, \partial B_{\theta(j)}$, intersecting them at the frontier anchor points c_1, c_2. Therefore the split points $s_1, s_2, \ldots, s_{\theta(j)-2}$ are located on each of the other $\theta(j) - 2$ boundary walls $\partial B_2, \partial B_3, \ldots, \partial B_{\theta(j)-1}$. Without a split point on each of these boundary walls, the split frontier cannot be formed and the junction cannot be traversed. Hence, a lower bound on the number of robots required to traverse a junction is given by minimizing the length of the split frontier over all possible points on the respective boundary walls. Thus we can never form a split frontier of total length less than



(a) Lower bound split frontier



(b) Upper bound split frontier

Fig. 5. Frontier configurations for lower and upper bound derivations

$$\left\|\mathcal{F}'_{s,min}\right\| = \min\left\{\|c_1\ s_1\| + \|s_1\ s_2\| + \ldots + \left\|s_{\theta(j)-3}\ s_{\theta(j)-2}\right\| + \left\|s_{\theta(j)-2}\ c_2\right\|\right\},$$
$$\text{for } c_1 \in \partial B_1, s_1 \in \partial B_2, \ldots, s_{\theta(j)-2} \in \partial B_{\theta(j)-1}, c_2 \in \partial B_{\theta(j)}\ . \quad (1)$$

If the number of robots is insufficient to form a split frontier of the smallest possible length, then it will certainly be insufficient to form a split frontier of any other length. But a split frontier must be formed in order to traverse a junction, irrespective of the exploration model. Thus no exploration model can dictate a configuration of robots that is able to traverse a given junction with fewer than $\|\mathcal{F}'_{s,min}\|$ robots. This establishes the lower bound (see Fig. 5a).

We establish the upper bound as follows. Assume we have some number of robots at a junction of degree $\theta(j)$. The ingoing frontier $\mathcal{F}'_0$ subtends between two boundary walls $\partial B_1, \partial B_2$, intersecting them at the frontier anchor points c_1, c_2. The ingoing frontier can always be aligned, without admitting recontamination, such that its anchor points correspond to the tangent points $\tau_1, \tau_{\theta(j)}$ on the two boundary walls that it subtends. By Lemma 1, a junction of degree $\theta(j)$ will have tangent points on $\theta(j)$ distinct boundary walls. Thereafter the ingoing frontier can always be extruded toward each of the other $\theta(j) - 2$ tangent points in turn, likewise without admitting recontamination. Thus for a junction j with tangent points $\tau_1, \tau_2, \ldots, \tau_{\theta(j)}$ we can always form a split frontier of total length

$$\begin{aligned}\left\|\mathcal{F}'_{s,max}\right\| &= \|c_1\ s_1\| + \|s_1\ s_2\| + \ldots + \left\|s_{\theta(j)-3}\ s_{\theta(j)-2}\right\| + \left\|s_{\theta(j)-2}\ c_2\right\| \\ &= \|\tau_1\ \tau_2\| + \|\tau_2\ \tau_3\| + \ldots + \left\|\tau_{\theta(j)-1}\ \tau_{\theta(j)}\right\|\ . \end{aligned} \quad (2)$$

We can traverse any junction in this way, thus no junction will ever require more than $\|\mathcal{F}'_{s,max}\|$ robots to traverse. This establishes the upper bound (see Fig. 5b). Observe that since all the tangent points are on the boundary of a maximal inscribed circle centered at j, the ingoing frontier $\mathcal{F}'_0$, aligned such that its anchor points correspond to $\tau_1, \tau_{\theta(j)}$, is at most $2\mathcal{D}(j)$ in length, i.e. the diameter of the circle. For each of the $\theta(j) - 2$ split points, the frontier gains an additional line segment, likewise of at most $2\mathcal{D}(j)$ in length. Thus we get a numeric upper bound on the maximum length of the split frontier, given by

$$\left\|\mathcal{F}'_{s,max}\right\| \leq 2\big(\theta(j) - 1\big)\mathcal{D}(j)\ . \quad (3)$$

We have now established an equivalence between the medial axis configuration space and the physical movement and frontier expansion of robots in physical space. We introduced an exploration model whereby a swarm locus moving along the skeleton allows us to reason about a group of robots moving through physical space. We defined what it means for a frontier to split and for a group of robots to traverse a junction, and derived lower and upper bounds on the number of robots required to traverse a junction.

4 Topology Tree

In this section we present a series of steps that will transform our continuous configuration space into a symbolic representation of the environment in the discrete domain. We establish a set of rules for navigating the environment in this discrete representation, that allow us to develop an algorithmic pursuit strategy. Using the junction lower bound from Result (1) we derive a lower bound on the total number of pursuers necessary to clear the environment, showing that no fewer than this number can possibly clear the environment regardless of the exploration model or pursuit strategy. Using the junction upper bound from Result (2) we develop an upper bound on the total number of pursuers that will always be sufficient to clear the environment, for any pursuit strategy. Finally, we derive an optimal pursuit strategy and prove that it guarantees we can clear the environment with the minimum number of pursuers for a given exploration model.

The most natural representation of the skeleton of an environment where both Q and B are connected is a tree. Since we are also given a starting point on the skeleton, we consider a directed rooted tree (rooted at the start node). We refer to this as the environment *topology tree*, denoted by $T = (V, E)$, where V is the set of vertices (nodes) and E is the set of edges of T. Let $s \in V$ be the root node of T. Nodes correspond to end points and junction points, and edges correspond to corridors connecting these points on the skeleton.

Let $\gamma : V \to \mathbb{N}$ denote the out-degree of a node. There are four types of nodes on the topology tree. The swarm locus starts at an end point on the skeleton which corresponds to the root node s of out-degree $\gamma(s) = 1$. Every other end point corresponds to a leaf node w of out-degree $\gamma(w) = 0$. Each junction point j is represented by a unique junction *entry node* u of out-degree $\gamma(u) = \theta(j) - 1$, connected to an associated set of distinct junction *exit nodes* $\{v_1, v_2, \dots, v_{\gamma(u)}\}$ of out-degree $\gamma(v) = 1$. (Observe that since T is a directed rooted tree, every node has in-degree 1, except for the root node s which has in-degree 0.)

The root node is connected to some junction entry node, while junction exit nodes are connected to leaf nodes and other junction entry nodes, as determined by the corridors connecting these points on the skeleton. Every edge $e \in E$ on the topology tree is assigned a weight $\alpha : E \to \mathbb{N}$. There are two fundamentally different types of edges: edges that represent a corridor on the skeleton and edges that represent the split frontier at a junction point.

Motivated by Corollary 1, for an edge e connecting node u of out-degree $\gamma(u) = 1$ to a node v, the edge weight $\alpha(e)$ is determined by the number of robots necessary and sufficient to advance from u to v, given by the maximum relief value $\mathcal{R}(x)$ at some point $x \in S$ amongst the points along that corridor.

Motivated by the reasoning in Section 3.2.2, for junction nodes the representation is as follows. For each junction entry node $u \in V$ let $E_u \subset E$ be the set of edges $\{e_1, e_2, \ldots, e_{\gamma(u)}\}$ connecting u to its associated set of junction exit nodes $\{v_1, v_2, \ldots, v_{\gamma(u)}\}$. For each junction, we define a traversal function $\delta : E_u \rightarrow \mathbb{N}$, where $\sum_i \delta(e_i)$ is the number of robots required to traverse the junction. This corresponds to the minimum number of robots required to form a split frontier at the junction, given by its ceiling length $\lceil \|\mathcal{F}'_s\| \rceil$. Since we do not have tight bounds on the length of the split frontier, the traversal function depends on the the context of the analysis. Namely, if the goal is to derive a lower bound on the number of robots required to clear an environment, then we consider the length of the split frontier $\lceil \|\mathcal{F}'_{s,min}\| \rceil$ given by Result (1). If the goal is to derive an upper bound then we consider the length of the split frontier $\lceil \|\mathcal{F}'_{s,max}\| \rceil$ given by Result (2). Each edge e_i is assigned weight $\alpha(e_i) = \delta(e_i)$ which corresponds to the number of robots n_i that are are required to form a frontier $\mathcal{F}'_i$ at each outgoing corridor, given by the ceiling length of each outgoing frontier $\lceil \|\mathcal{F}'_i\| \rceil$.

4.1 Exploration Rules

We consider exploration of the topology tree to be a game. We start the game with a single group of n_0 robots stationed at the root node s. Every node $v \in V$ on the topology tree is marked with a label λ, which can have one of three values: CONTAMINATED, EXPLORED and CLEARED. Initially, the root node is marked EXPLORED and all other nodes are marked CONTAMINATED.

We play the game in rounds, each round moving some number of robots from one node to another. If a group of robots is unable to move from one node to another on some round, then the robots are "stuck" at that node. This corresponds to the fact that if there are insufficient robots to clear a corridor, they will remain stuck guarding the corridor, unable to retreat without allowing recontamination. Let $\lambda_k(v)$ denote the labeling of a node $v \in V$ on round k. We win the game if the tree is cleared on some round k_0, that is if $\exists k_0 \in \mathbb{N} \mid \forall k > k_0 \; \forall v \in V, \lambda_k(v) = \text{CLEARED}$. We lose the game if all robots are stuck at some node but the tree has not been cleared by some round k_0, that is if $\exists k_0 \in \mathbb{N} \mid \forall k > k_0 \; \exists v \in V, \lambda_k(v) \neq \text{CLEARED}$.

Robots can split into smaller groups and join to form larger groups. In general, we are free to choose how we move the robots on the topology tree, provided that we obey the following transition rules.

1. If a group of n_0 robots reaches a node u where $\gamma(u) > 0$, then the group splits into some permutation of $\gamma(u)$ groups of n_i robots advancing to each of the children nodes $\{v_1, v_2, \ldots, v_{\gamma(u)}\}$. We are free to choose this permutation, subject to the following restrictions:
   (a) If $\lambda(v_i) = \text{CONTAMINATED}$, then $n_i \geq \alpha(e_i(u))$.
   (b) If $\lambda(v_i) = \text{EXPLORED}$, then $n_i \geq 0$.
   (c) If $\lambda(v_i) = \text{CLEARED}$, then $n_i = 0$.

   If no such permutation exists, then the group remains stuck at node u.
2. If a group of robots is stationed at a node u where $\lambda(u) = \text{CLEARED}$, then the group backtracks to the parent node.

3. If a group of robots reaches a node u that is marked CONTAMINATED, then u is marked EXPLORED.
4. If a group of robots reaches a leaf node u or a node u where all children of u are marked CLEARED, then u is also marked CLEARED.
5. If two or more groups of $n_1, n_2, \ldots, n_k$ groups of robots are stationed at the same node, then they form a single group of $n_1 + n_2 + \ldots + n_k$ robots.

The reasoning behind these rules follows from the problem formulation and results in Section 3. Rule 1(a) enforces that our exploration is non-recontaminating. Rule 1(b) allows robots to move to explored nodes and join other robots. Rules 3 and 4 define the progression of the game, and Rules 1(c) and 2 ensure that exploration is always progressive (the latter ensures a group of robots leaves a region once it has been cleared, while the former ensures that no group of robots re-enters that region unnecessarily). Rule 5 ensures that robots always act in a single group when stationed at a node.

We understand a state of the topology tree to mean the labeling of each node and the number of robots stationed at each node on a given round. We call a sequence of transitions between states of the topology tree a *pursuit strategy*. (We omit a formal definition for brevity.) A pursuit strategy is like a written record of a game of chess that allows the game to be replayed by carrying out the recorded sequence of transitions. Observe that the only degree of freedom in choosing a pursuit strategy is what to do at a given junction entry node u. We can always choose what junction exit node to send a group of robots to as long as it is not marked CLEARED. If the junction exit nodes are marked CONTAMINATED then the group traverses the junction if and only if $n_0 \geq \sum_i \alpha(e_i(u))$. If the junction is traversed, then the group splits into some permutation of $\gamma(u)$ groups of n_i robots advancing to each of the associated junction exit nodes $\{v_1, v_2, \ldots, v_{\gamma(u)}\}$. We are free to choose this permutation, provided that $\forall i, n_i \geq \alpha(e_i(u))$. The choice we make in selecting this permutation may affect the outcome of the game. (Note also that a node u is only marked CLEARED once the entire subtree $T(u)$ is marked cleared. Thus robots are forced to clear subtrees recursively, and can only backtrack once a given subtree is cleared.)

We also note that because n_0 and $|V|$ are finite, there are a finite number of possible pursuit strategies for a given topology tree. Intuition tells us that if n_0 is too low, every pursuit strategy will be a losing strategy, whereas if n_0 is sufficiently high then any pursuit strategy will be a winning strategy. We now formalize this intuition, and derive lower and upper bounds on the total number of robots that can clear the topology tree.

4.2 Environment Bounds

Consider the topology tree T with root node s. Let $T(q)$ be the tree obtained by considering node q as the root node and removing nodes that are not descendants of q. Let $n(T(q))$ be the number of robots required to clear $T(q)$. Let $P(v)$ be the set of nodes $\{p_1, p_2, \ldots, p_{\gamma(v)}\}$ that are children of $v \in V$, enumerated in order of ascending $n(T(p_i))$.

We motivate the lower bound as follows. At each node s we consider whether more robots are required to advance to the child node p_1 than are required to clear the rest of the subtree $T(p_1)$, and apply this recursively for the entire tree. At each junction entry node we consider the maximum number of robots required to clear a given subtree $T(p_i)$ while guarding the remaining junction exit nodes that have not been cleared. Formally, let $n_{min}(T(s))$ be the total number of robots necessary to clear the environment with topology tree T, given by

$$n_{min}(T(s)) =$$

$$\begin{cases} 0 & \text{if } \gamma(s) = 0 \\ \max\left\{ \sum_{i=1}^{\gamma(s)} \alpha\big(e_i(s)\big),\ \max_{i=1,\ldots,\gamma(s)} \Big\{ n_{min}\big(T(p_i)\big) + \sum_{j=i+1}^{\gamma(s)} \alpha\big(e_j(s)\big) \Big\} \right\} & \text{otherwise}\ . \end{cases} \tag{4}$$

Lemma 3. *$n_{min}(T(s))$ robots are necessary to clear an environment with topology tree T, regardless of exploration model or pursuit strategy.*

Proof. See full version [26].

We motivate the upper bound as follows. We imagine an adversary that dictates the pursuit strategy of a number of robots, with the goal of placing the maximum number of them on the topology tree T, while preventing T from being cleared. Then we argue that given any such adversarial configuration of $n^\star(T(s))$ robots, $n^\star(T(s)) + 1$ robots will always be able to clear T, regardless of the pursuit strategy chosen by the adversary. Formally, let $n_{max}(T(s))$ be the total number of robots sufficient to clear the environment with topology tree T, given by

$$n_{max}(T(s)) = n^\star(T(s)) + 1\ ,$$
$$n^\star(T(s)) = \begin{cases} 0 & \text{if } \gamma(s) = 0 \\ \max\left\{ \Big(\sum_{i=1}^{\gamma(s)} \alpha\big(e_i(s)\big) \Big) - 1\ ,\ \sum_{i=1}^{\gamma(s)} n^\star\big(T(p_i)\big) \right\} & \text{otherwise}\ . \end{cases} \tag{5}$$

Lemma 4. *$n_{max}(T(s))$ robots are sufficient to clear an environment with topology tree T, for a given exploration model, regardless of pursuit strategy.*

Proof. See full version [26].

4.3 Optimal Pursuit Strategy

We now present an optimal pursuit strategy that guarantees that the environment is cleared with the minimum number of robots for a given exploration model. Consider the topology tree T with root node s. We know that $n_{min}(T(s))$ robots are necessary to clear T, given by Result (4). The following algorithm guarantees that T will be cleared with $n_{min}(T(s))$ robots. (We use the same notation as in Section 4.2.)

Algorithm 1. – `Clear`$\big(T(s),\ n_0 \big)$

```
Given $n_0$ robots located at root node $s$ of topology tree $T = (V, E)$:
1. If $s$ is a leaf node or if all children of $s$ are marked CLEARED, then:
    (a) mark $s \leftarrow$ CLEARED.
    (b) Backtrack to parent node. If parent node does not exist, terminate.
2. If $\lambda(s) =$ CONTAMINATED, mark $s \leftarrow$ EXPLORED.
3. If $\gamma(s) = 1$, Clear$\big( T(p_1),\ n_0 \big)$.
4. If $\gamma(s) > 1$ and if all children of $s$ are marked CONTAMINATED, then:
    (a) for $i = 2, \ldots, \gamma(s)$: Clear$\big( T(p_i),\ \alpha(e_i(s)) \big)$.
    (b) Clear$\big( T(p_1),\ n0 - \sum_{i=2}^{\gamma(s)} \alpha(e_i(s)) \big)$.
5. If $\gamma(s) > 1$ and if all children of $s$ are not marked CONTAMINATED, then:
    (a) let $i = \min_{2,\ldots,\gamma(s)} \{ i \mid \lambda(p_i) \neq$ CLEARED$\}$.
    (b) Clear$\big( T(p_i),\ n_0 \big)$.
```

Lemma 5. $n_{min}(T(s))$ *robots are necessary and sufficient to clear an environment with topology tree* T*, for a given exploration model.*

Proof. See full version [26].

Using the junction upper bound from Result (2) to obtain the traversal function δ_{max} for each junction, we know that no more than $\delta_{max}(e_i(u))$ robots are required to traverse the junction point j corresponding to the junction entry node u for the given exploration model. Thus, using $\alpha(e_i(u)) = \delta_{max}(e_i(u))$ for each junction entry point u, the `Clear` algorithm gives an optimal pursuit strategy for clearing an environment with topology tree T, for a given exploration model.

5 Conclusion

The problem of obtaining a concise characterization of a physical environment in the context of frontier-based non-recontaminating exploration was considered. We introduced the medial axis as a configuration space and showed that reasoning about points in this configuration space is equivalent to reasoning about robots in physical space. We formalized the notion of width, corridors and junctions and derived lower and upper bounds on the number of robots required to traverse a junction. We presented a transformation from this continuous configuration space into a symbolic representation in the discrete domain. We cast the exploration problem as a game, established rules for playing this game, and derived bounds on the number of robots necessary and sufficient to clear the environment. Finally we presented an optimal pursuit strategy that guarantees that we can clear the environment with the minimum number of robots.

There are a number of interesting future lines of research for this work. First, the establishment of tight bounds on the number of robots required to traverse a junction — we suspect that the lower bound given by Result (1) in Section 3.2 is in-fact necessary and sufficient. A rigorous proof of this fact would have to generalize to accommodate a number of special cases.

Second, the extension of this work to environments with holes would be a significant contribution. The medial axis configuration space was chosen with this in mind, and the model presented in Section 3 soundly generalizes the characterization to arbitrary connected environments. A number of issues need to be addressed in transforming this representation into the discrete domain. One would need to consider an undirected graph and junctions would need to be represented accordingly. It is known that computing the number of searchers required to clear a general graph is NP-hard [21], so suitable heuristics or approximations would need to be employed. Alternatively, the graph could be converted into a tree such as in [15], [16]; however the non-isotropic nature of the junction transition function would demand a judicious approach to blocking cycles.

Acknowledgements. This research was supported in part by the Future Urban Mobility project of the Singapore-MIT Alliance for Research and Technology (SMART) Center, with funding from Singapore's National Research Foundation. It was also supported in part by AFOSR Award Number FA9550-08-1-0159, NSF Award Number CNS-0715397, NSF Award Number CCF-0726514, ONR grant N000140911051 and NSF grant 0735953. We are grateful for this support.

References

1. Adler, M., Racke, H., Sivadasan, N., Sohler, C., Vocking, B.: Randomized pursuit-evasion in graphs. Combinatorics, Probability and Computing 12(03), 225–244 (2003)
2. Blum, H.: A transformation for extracting new descriptors of shape. Models for the Perception of Speech and Visual Form 19, 362–380 (1967)
3. Burgos, J.L.: Pursuit/evasion behaviors for multi-agent systems (2010)
4. Choi, H., Choi, S., Moon, H.: Mathematical theory of medial axis transform. Pacific Journal of Mathematics 181(1), 57–88 (1997)
5. Danielsson, P.: Euclidean distance mapping. Computer Graphics and Image Processing 14(3), 227–248 (1980)
6. Dimitrov, P., Phillips, C., Siddiqi, K.: Robust and efficient skeletal graphs. In: CVPR, p. 1417. IEEE Computer Society (2000)
7. Durham, J.W., Franchi, A., Bullo, F.: Distributed pursuit-evasion with limited-visibility sensors via frontier-based exploration. In: 2010 IEEE International Conference on Robotics and Automation (ICRA), pp. 3562–3568. IEEE (2010)
8. Gerkey, B., Thrun, S., Gordon, G.: Visibility-based pursuit-evasion with limited field of view. The International Journal of Robotics Research 25(4), 299 (2006)
9. Giblin, P.J., Kimia, B.B.: Local forms and transitions of the medial axis. In: Siddiqi, K., Pizer, S.M. (eds.) Medial Representations, Computational Imaging and Vision, vol. 37, pp. 37–68. Springer, Netherlands (2008)
10. Guibas, L., Latombe, J., Lavalle, S., Lin, D., Motwani, R.: Visibility-based pursuit-evasion in a polygonal environment. Algorithms and Data Structures 1272, 17–30 (1997)
11. Guibas, L.J., Holleman, C., Kavraki, L.E.: A probabilistic roadmap planner for flexible objects with a workspace medial-axis-based sampling approach. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 1999, vol. 1, pp. 254–259. IEEE (1999)
12. Guibas, L., Latombe, J., LaValle, S., Lin, D., Motwani, R.: A visibility-based pursuit-evasion problem. Intnl. Journal of Computational Geometry and Applications 9(4/5), 471 (1999)

13. Holleman, C., Kavraki, L.E.: A framework for using the workspace medial axis in prm planners. In: Proceedings of IEEE International Conference on Robotics and Automation, ICRA 2000, vol. 2, pp. 1408–1413. IEEE (2000)
14. Isler, V., Sun, D., Sastry, S.: Roadmap based pursuit-evasion and collision avoidance. In: Proc. Robotics, Systems, & Science (2005)
15. Kolling, A., Carpin, S.: The graph-clear problem: definition, theoretical properties and its connections to multirobot aided surveillance. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2007, pp. 1003–1008. IEEE (2007)
16. Kolling, A., Carpin, S.: Multi-robot surveillance: an improved algorithm for the graph-clear problem. In: IEEE International Conference on Robotics and Automation, ICRA 2008, pp. 2360–2365. IEEE (2008)
17. Kolling, A., Carpin, S.: Pursuit-evasion on trees by robot teams. IEEE Transactions on Robotics 26(1), 32–47 (2010)
18. Kuijper, A.: Deriving the medial axis with geometrical arguments for planar shapes. Pattern Recognition Letters 28(15), 2011–2018 (2007)
19. LaValle, S., Lin, D., Guibas, L., Latombe, J., Motwani, R.: Finding an unpredictable target in a workspace with obstacles. In: Proceedings of IEEE International Conference on Robotics and Automation, vol. 1, pp. 737–742. IEEE (1997)
20. Nowakowski, R., Winkler, P.: Vertex-to-vertex pursuit in a graph. Discrete Mathematics 43(2-3), 235–239 (1983)
21. Parsons, T.: Pursuit-evasion in a graph. Theory and Applications of Graphs 642, 426–441 (1978)
22. Rodriguez, S., Denny, J., Zourntos, T., Amato, N.M.: Toward Simulating Realistic Pursuit-Evasion using a Roadmap-Based Approach. In: Boulic, R., Chrysanthou, Y., Komura, T. (eds.) MIG 2010. LNCS, vol. 6459, pp. 82–93. Springer, Heidelberg (2010)
23. Sachs, S., LaValle, S., Rajko, S.: Visibility-based pursuit-evasion in an unknown planar environment. The International Journal of Robotics Research 23(1), 3 (2004)
24. Serra, J.: Image analysis and mathematical morphology. Academic Press, London (1983); [Review by Fensen, EB in: J. Microsc. 131 (1983) 258.] Technique Staining Microscopy, Review article General article, Mathematics, Cell size (PMBD, 185707888) (1982)
25. Suzuki, I., Yamashita, M.: Searching for a mobile intruder in a polygonal region. SIAM Journal on Computing 21, 863 (1992)
26. Volkov, M., Cornejo, A., Lynch, N., Rus, D.: Environment characterization for non-recontaminating frontier-based robotic exploration (full version) (2011), http://people.csail.mit.edu/mikhail/volkov2011environment-full.pdf
27. Wilmarth, S., Amato, N., Stiller, P.: Maprm: A probabilistic roadmap planner with sampling on the medial axis of the free space. In: Proceedings of IEEE International Conference on Robotics and Automation, vol. 2, pp. 1024–1031. IEEE (1999)
28. Yamashita, M., Umemoto, H., Suzuki, I., Kameda, T.: Searching for mobile intruders in a polygonal region by a group of mobile searchers. Algorithmica 31(2), 208–236 (2001)
29. Yamauchi, B.: A frontier-based approach for autonomous exploration. In: Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA 1997, pp. 146–151. IEEE (1997)
30. Yamauchi, B.: Frontier-based exploration using multiple robots. In: Proceedings of the Second International Conference on Autonomous Agents, pp. 47–53. ACM (1998)

Aspects of Active Norm Learning and the Effect of Lying on Norm Emergence in Agent Societies

Bastin Tony Roy Savarimuthu, Rexy Arulanandam, and Maryam Purvis

University of Otago, Dunedin, P.O. Box 56, Dunedin, New Zealand
{tonyr,tehrany}@infoscience.otago.ac.nz,
rexi1010@gmail.com

Abstract. Norms have facilitated smoother functioning in human societies. In the field of normative multi-agent systems researchers are interested in investigating how the concept of social norms can be used to facilitate social order in electronic agent societies. In this context, the area of norm emergence has attracted a lot of interest among researchers. The objectives of this paper are two-fold. First, we discuss the norm learning approaches in agent societies and discuss the three aspects of active norm learning (experiential, observational and communication-based learning) in agent societies. Using an example we demonstrate the usefulness of combining these three aspects of norms learning. Second, we investigate the effect of the presence of liars in an agent society on norm emergence. Agents that lie distort truth when they are asked about the norm in an agent society. We show that lying has deleterious effect on norm emergence. In particular, using simulations we identify conditions under which the norms that have emerged in a society can be sustained in the presence of liars.

1 Introduction

Norms have been of interest to researchers in multi-agent systems because they enable cooperation and coordination among software agents. They are also light-weight mechanisms for enabling social control. Agents that know about norms in agent societies do not need to recompute what the norms of the society are and also do not often need to spend time in contemplating actions that are forbidden and obliged as they are aware of these norms. Also, agents that are aware of norms know that violating them will have consequences for them. However, this is true only when the agents know what the norms are. A new agent joining an open society may not know what the norms of the society are. This agent will need to be equipped with some mechanism for learning the norms in the society.

Researchers have employed several mechanisms for the learning of norms. These include imitation, normative advice from leaders, machine learning and data-mining [15]. Section 2 provides a brief background on norm research in the field of multi-agent systems. Section 3 discusses different mechanisms used by researchers for norm learning. In Section 4 we discuss three aspects of active learning namely experiential learning, observational learning and communication-based learning and compare the use of these three aspects in the existing works on norm learning. We also discuss how these three aspects can be integrated in the context of a simple example. In Section 5 we investigate the effect of lying on norm emergence when communication-based learning is used. We identify the tipping points (i.e. number of liars in the society) after which the norms are no longer sustained in an agent society.

Thus, this paper makes two contributions to normative multi-agent systems. First, it discusses how active learning on the part of the agent helps in the process of expediting norm learning in agent society. Second, it discusses the impact of lying on norm emergence in an agent society.

2 Background

Researchers in multi-agent systems have studied how the concept of norms can be applied to artificial agents. Norms are of interest to multi-agent system (MAS) researchers as they help in sustaining social order and increase the predictability of behaviour in the society. Researchers have shown that norms improve cooperation and collaboration [21,25]. Epstein has shown that norms reduce the amount of computation required to make a decision [7]. However, software agents may tend to deviate from norms due to their autonomy. So, the study of norms has become important to MAS researchers as they can build robust multi-agent systems using the concept of norms and also experiment on how norms may evolve in response to environmental changes.

Research in normative multi-agent systems can be categorized into two branches. Researchers have worked on both prescriptive (top-down) and emergent (bottom-up) approaches to norms.The first branch focuses on normative system architectures, norm representations, norm adherence and the associated punitive or incentive measures. Several architectures have been proposed for normative agents (refer to [13] for an overview). Researchers have used deontic logic to define and represent norms [10]. Several researchers have worked on mechanisms for norm compliance and enforcement such as sanctioning mechanisms [2] and reputation mechanisms [5].

The second branch of research is related to emergence of norms [20, 21]. In the bottom-up approach, the agents infer a norm through learning mechanisms [20,21] and cognitive approaches [1]. This paper contributes to this branch of research by providing an overview of norm learning mechanisms.

3 Approaches to the Learning of Norms

Researchers have employed four types of mechanisms for an individual agent to learn norms: imitation, machine learning, data mining and advice-based learning. The learning mechanisms identified in this paper are extensions to the learning mechanisms discussed in the categorization presented in a previous work [15]. Since the imitation, machine learning and advise-based learning mechanisms (also called as leadership mechanisms) are explained in the previous work, we only provide a brief summary of these three approaches and provide a longer discussion on the data mining approaches.

Imitation Mechanisms. The philosophy behind an imitation-based learning mechanism is *When in Rome, do as the Romans do* [7]. Models based on imitation are characterised by agents first observing and then mimicking the behaviour of what the majority of the agents do in a given agent society (following the crowd). Epstein's main argument [7] for an imitation mechanism is that individual thought (i.e. the amount of computing needed by an agent to infer what the norm is) is inversely related to the strength of a social norm. This implies that when a norm becomes entrenched the agent can follow it without much thought. An issue for debate is whether imitation-based behaviour (solely) really leads to norms as there is no notion of generalized expectation. Imitation mechanism also does not consider sanctions or rewards thereby focus only on conventions and not norms[1]. The direct utility from conforming to a particular behaviour is not modelled in some cases (i.e. blindly imitating what the crowd does without carefully considering the impact of the actions either for the agent or the society).

Works Based on Machine Learning. Several researchers have experimented with agents finding a norm based on learning on the part of an agent when it interacts with other agents in the society by performing some actions [20,21,25]. Researchers have used simple reinforcement algorithms for norm learning. The reinforcement learning algorithms identify a strategy that maximizes an agent's utility and the chosen strategy is declared as the norm. Since all agents in the society make use of the same algorithm, the society stabilises to an uniform norm. Agents using this approach cannot distinguish between a strategy and a norm. These agents accept the strategy that maximizes its utility as its norm. However, the agents do not have a notion of normative expectation associated with a norm (i.e. agents do not expect certain behaviour on the part of other agents). Another weakness is that agents in machine learning approach do not have a mental notion of norms (i.e. the ability to reason about why norms have to be followed and the consequences for not following norms) as they are mainly utilitarian agents. These limitations are addressed by the works that employ cognitive approaches where the norms learnt affect an agent's future decision making by influencing its beliefs, intentions and goals [1].

Advice-Based Learning. Boman [3] has used a centralised approach, where agents consult with a normative advisor before they make a choice on actions to perform. Verhagen [23] has extended this notion of normative advice to obtaining normative comments from a centralized normative advisor (e.g. the leader of the society) on an agent's previous choices. Savarimuthu et al. [16] have adopted a distributed approach approach for normative advice. In their mechanism, there could be several normative advisors (called role models) from whom other agents can request advice. Hoffmann [9] has experimented with the notion of norm entrepreneurs who think of a norm that might be beneficial to the society. An entrepreneur can recommend a norm to a certain percentage of the population (e.g. 50%) which leads to varying degrees of establishment of a norm. The models based on advice assume that a powerful authority is present in the society and all agents in the society acknowledge the power of such agents. Both centralised and distributed notions of norm spreading using *power* have been employed. The centralised approach is suitable for closed societies. However, this might not work well for open, flexible and dynamic societies. Distributed approaches for norm spreading and emergence are promising because the computational cost required to spread, monitor and control a norm is distributed to all the members of the society.

[1] Many sociologist consider sanctions and/or rewards a core part of the norm.

Data Mining Mechanisms. Agents can use data mining approach to identify norms in agent societies. Agents in open agent societies can learn norms based on what they infer based on their observations of the society. The repository of an agent's observations can be mined for patterns of behaviour. There has been a proposal of an agent architecture for normative systems to employ data mining for citizens of a country to find information and norms from official documents [22]. However, the work does not describe what types of norms are discovered and also the mechanisms used in the identification of norms.

Savarimuthu et al. [17, 18] have proposed an architecture for norm identification which employs association rule mining, a data mining approach. The architecture makes use signals (sanctions and rewards) as the starting points for norm identification. Mechanisms for identifying two types of norms, prohibition norms and obligations norms have been studied. The details on how an agent identifies a prohibition norm are explained in the context of a public park scenario, where the norm against littering is identified by the agent. The obligation norm inference is explained in the context of a tipping norm in a restaurant scenario. They have demonstrated that an agent using the proposed architecture can dynamically add, remove and modify norms based on mining the interactions that take place between agents. They have shown that agents can identify co-existing norms. The agents can also identify conditional norms (e.g. identification of *normative pre-conditions* which are conditions that have to be true for the norm to hold).

In the work of Lotzmann et al. [12] an agent constructs a decision tree of events that take place. It learns norms by considering the occurrence probabilities of those events that take place. For example, an agent participating in a traffic scenario either as a pedestrian or a car driver, decides about which action to perform, based on the probability of events represented as nodes of a decision tree. Based on these probabilities, a pedestrian agent learns that if it jaywalks instead of using the pedestrian crossing, it has a high probability of being run over by a car. A car driver learns to stop in the pedestrian crossing area.

Data mining is a promising approach for the identification of some types of norms that can be inferred based on observing the interactions between agents in the society. However, if actions that explicitly signal a sanction or reward are absent or other mechanisms such as reputation are used instead of explicit signals (i.e. reduction in the reputation score of a rogue agent instead of explicit sanctioning that is visible to other agents), then it is difficult to identify norms.

4 Aspects of Active Learning of Agents

Hamada et al. [8] note that *active learning is learning with learners involved in the learning process as active partners: meaning they are "doing", "observing" and "communicating" instead of just "listening" as in the traditional learning style*. An actively learning agent can thus learn about norms in the following three ways.

– **Experiential learning** - This is the ability of an agent *learning by doing*. For example, an agent may litter a park. It may be sanctioned by some other agent(s).

Through the sanction experienced as a result of the littering action, the agent can learn about the norm. Thus, an agent can learn from its personal experience based on sanctions and rewards.
– **Observational learning** - This is the ability of an agent *learning by observing*. For example, an agent may observe littering agents being sanctioned in a society. Through the observation of the sanction on others, an agent can learn about the norm.
– **Communication-based learning** - This is the ability of an agent *learning by communicating* with other agents. For example, an agent may ask another agent in the park what the norms of the park are and that agent may communicate the norm to the agent. Norm communication can happen at a peer-to-peer level or from leaders to follower agents.

Table 1 shows the aspects of learning used by different research works investigating norms. It can be noticed that not all the three types of learning have been investigated by many research works. Only some of the recent research works have considered all the three types of learning [6,18].

Table 1. Comparison of the types of learning employed by different research works (Yes - considered, No - not considered)

Model	**Experiential learning**	**Observational learning**	**Communication-based learning**
Axelrod, 1986 [2]	No	Yes	No
Shoham and Tennenholtz, 1992 [21]	Yes	No	No
Kittock, 1993 [11]	Yes	No	No
Walker and Wooldridge, 1995 [25]	Yes	No	Yes
Verhagen, 2001 [23]	No	No	Yes
Epstein, 2001 [7]	No	Yes	No
Hoffmann, 2003 [9]	No	Yes	No
Pujol, 2006 [14]	Yes	No	No
Sen and Airiau, 2007 [20]	Yes	No	No
Savarimuthu et al., 2010 [17,18]	Yes	Yes	Yes
EMIL Project, 2006-2010 [6]	Yes	Yes	Yes

4.1 Comparing Different Combinations of Learning

In this section we demonstrate how the three types of learning can be carried out together in the context of an example that uses the machine learning approach. Most research works using machine learning mechanisms have investigated only the experiential learning aspect [14,20].

Consider the scenario where agents strive to establish a convention of driving either on the left (L) or the right (R) of the road. The payoff matrix for this coordination game is given in Table 2. The goal of the learning task is to enable the agents in a society to drive either on the right or the left. This goal can be achieved through several combinations of three aspects of learning[2]. In this work, we will compare three combinations just to demonstrate that the use of more than one aspect of learning improves the rate of norm emergence[3]. These three combinations are 1) learning by doing, 2) learning by doing and observing and 3) learning by doing, observing and communicating.

Table 2. Payoff matrix

	L	R
L	1, 1	-1, -1
R	-1, -1	1, 1

Assume that there are 100 agents in the system. In each iteration the agents randomly interact with one other agent by choosing an action (L or R). Based on the outcome of the interaction, the agent learns which action to choose for the next iteration. We use a simple Q-Learning algorithm [26] to facilitate learning similar to other works on norm emergence [24]. The Q-value is calculated using the formula

$$Q^t(a) = (1 - \alpha) * Q^{t-1}(a) + \alpha * R \qquad (1)$$

where $Q^t(a)$ is the utility of the action a chosen by an agent after t times, R is the reward for performing an action and α is the learning rate. Agents choose actions that yield the highest utility.

The pseudocode described in Algorithm 1 provides the operational details of an agent in learning a norm. An agent may learn based on a combination of different aspects. If an agent learns by doing, then, when it interacts with another agent (in the context of playing the coordination game), it chooses the action for which it has the highest Q-value. It then updates the Q-value of the chosen action based on the reward obtained. If an agent chooses to learn by observing, it learns by observing a randomly chosen agent X interacting with another agent Y. It then updates the Q-value for the action that was chosen by X, based on the reward obtained by X. If an agent learns through communication, the agent asks another agent for the action that it considers to be the norm. It then updates the Q-value of the action recommended as the norm with a reward of one.

[2] Seven combinations are possible without considering repetition. These are *doing, observing, communicating, doing-observing, observing-communicating, communicating-doing, doing-observing-communicating*.

[3] Conventions and norms are broadly considered under the same umbrella of norms in this paper, similar to some other works in this field [20] on norm emergence and the terminologies have been used interchangeably in this paper. We recognize that the distinction between norms and conventions exist in multi-agent systems by the fact that norms have explicit sanctions associated with them.

Algorithm 1.1. Psuedocode for agent learning

```
1  while iterationNumber $\leq$ totalNoOfIterations do
2      if learningFromDoingEnabled then
3          Each agent interacts with one other agent by choosing an action that has the
           highest Q-value;
4          Each agent updates the Q-value of the chosen action based on the reward
           obtained;
5      if learningFromObservingEnabled then
6          Each agent observes the action chosen by one other agent (X) interacting with
           another agent (Y);
7          Each agent updates the Q-value of the action chosen by X based on the reward
           obtained by X;
8      if learningFromCommunicationEnabled then
9          Each agent asks another agent for the action that is considered to be the norm;
10         Each agent updates the Q-value of the action recommended as the norm with a
           reward value of one;
11
```

We have conducted three experiments by fixing the value of α to 0.3. In the first experiment, an agent learns only through its experience (i.e. based on the result of their interaction with other agents). In the second experiment, in addition to experiential learning they also observe one other agent's action and learn from the result of that action (experiential + observational learning). In the third experiment, in addition to the set-up of the second experiment, an agent also learns from the experience of one other agent (i.e. by asking about the action performed by the agent and the reward it obtained). It should be noted that the third experiment involves all the three aspects of learning.

Figure 1 shows the results of the three experiments as three lines. The results are based on the average of 500 runs per experiment. It demonstrates that experiment three that uses experiential, observational and communication based learning results in the fastest convergence of norms. It is intuitive that an agent that makes use of the three aspects of learning performs better since more information is available to the agent from all the three learning channels. However, it is interesting to note that not all the three aspects have been considered by many research works as shown in Table 1. Even though communication is a fundamental aspect of agent systems, many research works have not considered the possibility of agents learning from others. The reason for this may include the lack of trust on other agents (i.e. agents may lie when asked for advice about a norm).

4.2 The Need for Integrating the Three Aspects of Learning

We note that the future research works on norm learning should consider integrating these three aspects where ever possible. The reasons are outlined below.

1. One of the drawbacks on the experiential learning of norms in an agent society is that an agent cannot perform all possible actions in order to find out what the norms

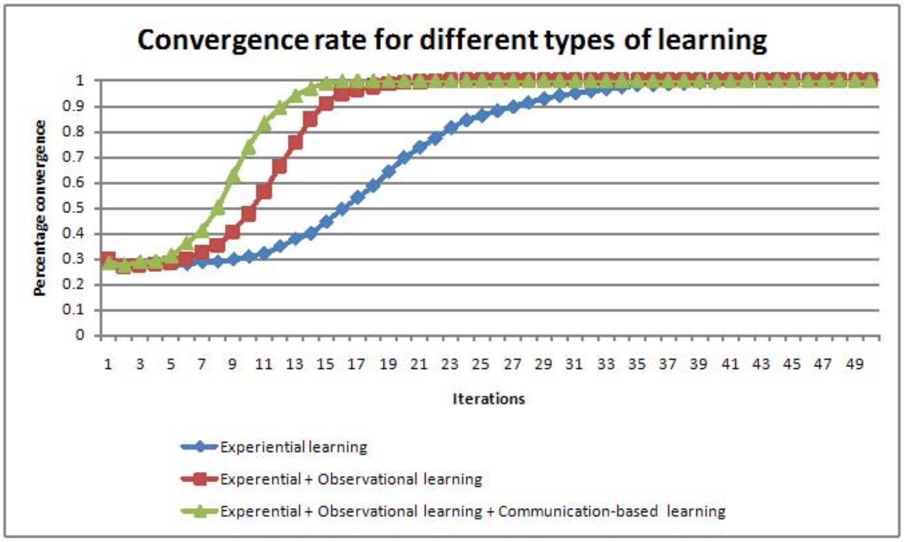


Fig. 1. Comparison of convergence rate in three experiments (varying different aspects of learning)

of the society are. For example, a new agent in a society may not know what the norms are and it may not be desirable to perform all actions to see whether any of those actions result in a sanction by performing it. The state space of actions can be large. Hence, this approach can be computationally expensive. However, if an agent does not actively search for an action that might be sanctioned, but only learns based on receiving a sanction for an action that it performed accidentally, it can use that sanction as a starting point to infer a norm. For example, when it is sanctioned for littering, it can flag the littering action as the potential norm and then check to see in its future interactions with other agents whether littering causes a sanction.

2. Using just the observational learning for learning norms might also cause problems. Assume that agent Z observes agent X punishing Y. Only if Z observes both a) the action responsible for the sanction and b) the sanction itself, it can learn from the observation. However, if the observer (agent Z) does not know which of the actions agent Y had done in the immediate past had caused this sanction, this approach will not be useful. In this case, it has to learn by asking about a norm from an agent in the society (i.e. communication-based learning).
3. Using just the communication learning may be sufficient in regimented societies where norms are prescribed by the organization and in societies where there is no lying. However, in open agent societies it may not be possible to rule out lying. In this case an agent may have to engage in observational learning and/or experiential learning.

5 Lying in Agent Societies

One of the issues with communication-based learning is the ability of agents to lie. Agents being autonomous entities may not be truthful when communicating. For example, an agent that is anti-social may not want a norm to emerge in a society. Therefore, it may try to thwart the process by spreading false information about the norm when asked for advice from other agents.

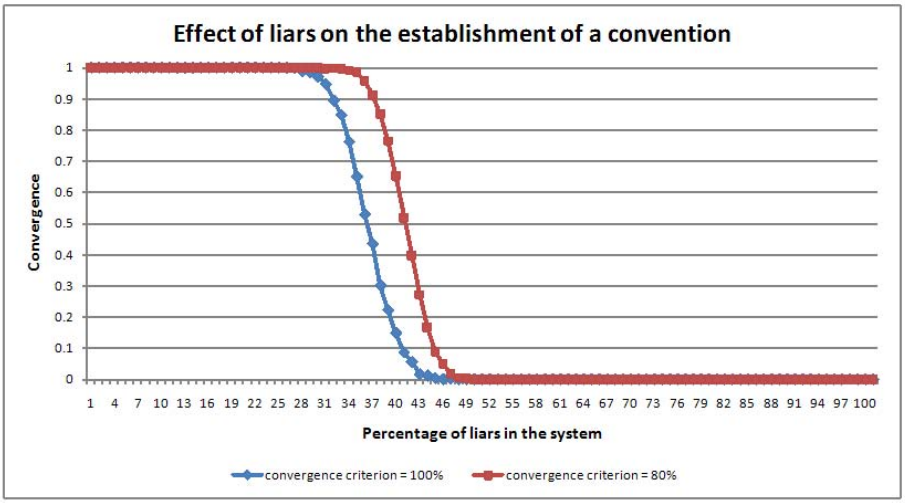


Fig. 2. Effect of liars on convention emergence (observation and communication-based learning)

In this section, we discuss our investigation of the effect of lying in communication-based learning. In Section 5.1, we discuss the effect of lying when communication-based learning is used in conjunction with observation based learning. In Section 5.2 we discuss the effect of lying when communication-based learning is used in conjunction with observation and experiential learning. Our aim is to determine how much lying (i.e. the percentage of liars in a society) would thwart the process of norm emergence. In other words, we aim at investigating how much lying can still be allowed in the society that would not disrupt the norm emergence process.

5.1 Impact of Lying When Observational and Communication-Based Learning Is Used

In a society of 100 agents, by keeping all the other parameters constant, we varied the number of liars in a society from 1 to 100. We conducted 1000 runs of each experiment, with each run spanning 200 iterations. At the end of the runs we calculated the percentage of convergence of the society to one of the conventions (either left or right). Figure 2 shows the effect of liars on norm emergence. The two lines correspond to different

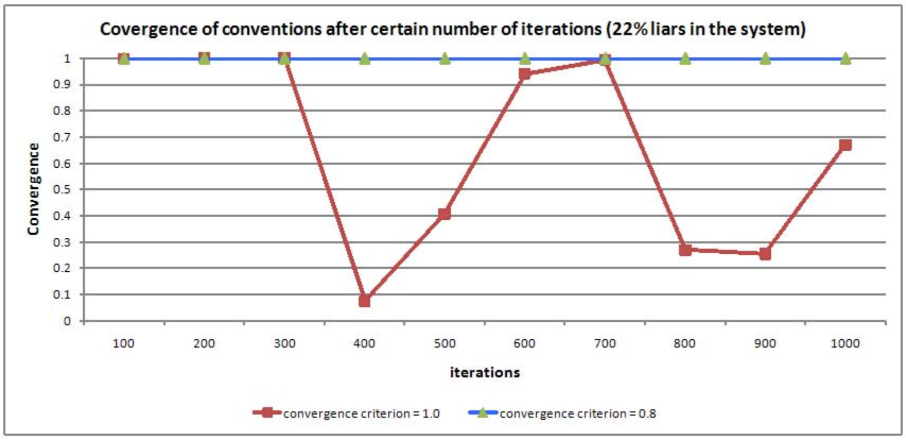


Fig. 3. Effect of liars on convention emergence by varying the number of iterations before convergence is checked

convergence criteria, 80% and 100% in the society respectively[4]. It can be inferred that the system always converges when there are less than or equal to 25% liars in the system when the convergence criterion is 100% and it always converges when there are less than or equal to 29% liars when the convergence criterion is 80%. These points can be considered as tipping points after which there is a rapid drop in the convergence to a norm. Once the system has 46% and 49% liars respectively for convergence criteria of 100% and 80%, the system never converges to a stable norm (i.e. probability of convergence to a norm is 0%).

Since we had only investigated convention emergence after certain number of iterations (i.e. 200 iterations), we investigated convention emergence further systematically at the end of every 100 iterations to a maximum of 1000 iterations (i.e. by conducting 10 experiments). Each experiment was repeated 1000 times. The number of liars was kept to 22. Since the system converged to a convention at 200 iterations, we expected all the experiments after 200 iterations would result in 100% convergence. However, the results that we obtained were surprising (see Figure 3). It can be seen that even though with 22 liars the system had converged in iterations 100, 200 and 300, it did not converge from iterations 400 to 600. It converged in iteration 700 and then it again it did not converge after that. So, there appears to be *cycles* in convergence which we call as *cycle effect*. The cycle is caused by all agents converging to a convention because of learning by observation initially, but once the system has converged to a norm, non-liars are conned by liars and this leads to some of the non-liars moving away from the convention. Soon those non-liar agents that deviated from the convention realize that they have been conned (i.e. through their reduction in utility) and they choose the

[4] 100% convergence means that in all the 1000 runs of the experiment, the system converged to a norm at the end of certain number of iterations (e.g. 200 iterations). 80% convergence criteria means that in all the 1000 runs of the experiment, at least 80% of the agents have converged to a norm.

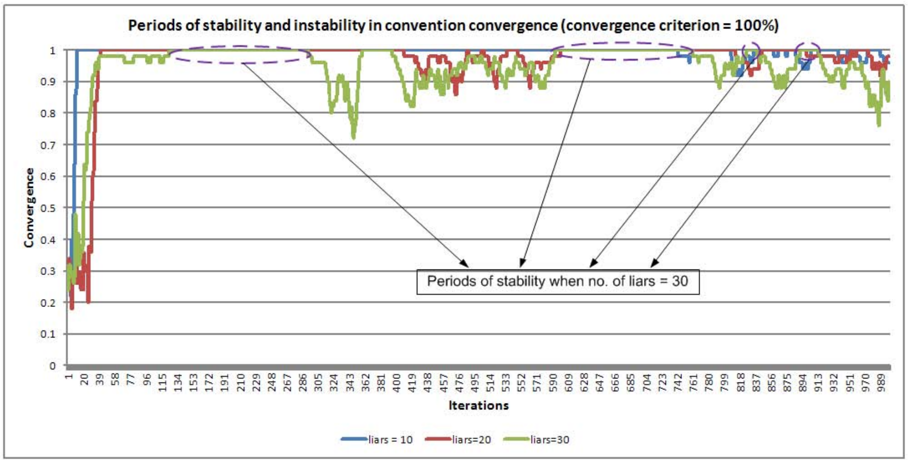


Fig. 4. Periods of stability and instability when the number of liars were varied (convergence criterion = 100%)

appropriate convention. This cycle repeats again. It should be noted that the cycle effect is seen only for 100% convergence and not for 80% convergence which called for further investigation.

In order to investigate the cycle effect further, we examined sample runs (one sample each) from the system where the number of liars were 10%, 20%, and 30% respectively. The convergence criterion was set to 100%. The experiments were conducted for 1000 iterations. It can be observed from Figure 4 that there are periods of stability and instability in convergence (i.e. 100% convergence) in the system. The periods of instability for low liar numbers was smaller than the periods of instability for larger number of liars. These lines also show the cycle associated with convergence.

Figure 5 shows the periods of stability and instability in convergence emergence when the convergence criterion was set to 80% in the system. It is intuitive that there are fewer periods of instability than when the convergence criterion was set to 100% for the same number of liars in the system. Note that there were no periods of instability when the number of liars were 20. This is in agreement with the result shown for approximately same number of liars in Figure 2.

In the light of the result shown in Figure 4 we note that the result shown in Figure 2 still holds since we had investigated convergence in 1000 runs after iteration 200 in each of the runs (in Figure 2). In all 1000 runs the system always converged since iteration 200 falls in the period of stability for 22 liars. However, since the experiment was not run for large number of iterations, it did not capture the dynamics that could have ensued. So, what can be inferred from this result is that we need to measure convergence throughout the entire simulation period instead of just measuring convergence at certain points in time such as iteration 200. Additionally the mechanism employed should check whether a convention is sustained once it has converged. In other words, the investigations should examine whether the system upon reaching the desired convergence level (80% or 100%) becomes unstable again. If it becomes unstable then the convention is said to be unsustained.

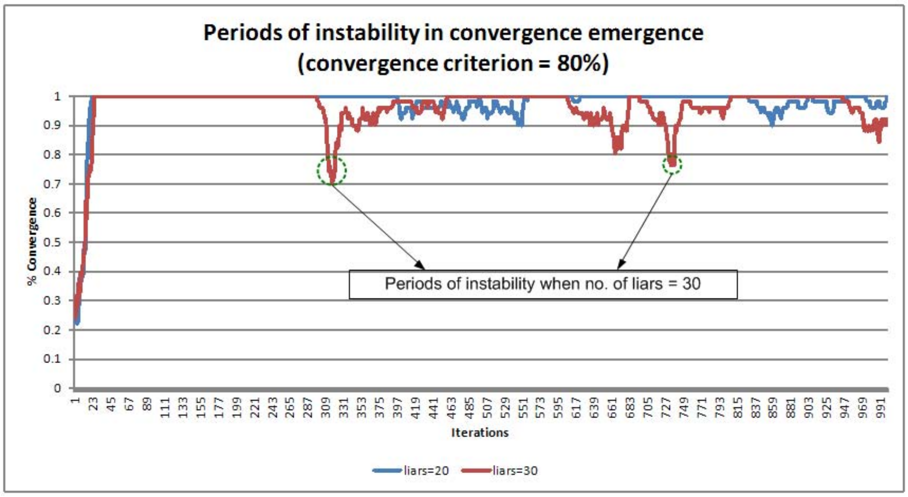


Fig. 5. Periods of stability and instability when the number of liars were varied (convergence criterion = 80%)

The experimental results discussed so far checked for convergence after certain number of iterations. We modified this process by making a simple change where the stability of convention convergence is checked once the initial convergence is reached (i.e. when all the agents converge to a norm). We allowed the agents to interact for certain number of iterations initially before we started measuring convergence. This value was set to 500 in our experiments[5]. Figure 6 shows the results on sustaining norms in an agent society. We have found that in order to sustain convergence there cannot be more than two liars in the society for 100% convergence criterion, and there cannot be more than 11 liars in the society for 80% convergence criterion. These are the tipping points for the number of liars a system can have above which the probability of the system sustaining the emerged norm is less than one. The results reported in this experiment are based on running the experiment 10,000 times (i.e. for each number between 1 to 100 that represents the number of liars in the system, the experiment was repeated 10,000 times).

It can be observed from Figure 6 that there is a considerably long period where the convergence is closer to 100% (percentage of liars from 12 to 22) for 100% convergence criterion. Once the number of liars in the system reach 5 and 33 respectively for 100% and 80% convergence criteria, the probability of system converging to a norm is zero.

5.2 Impact of Lying When All the Three Aspects of Learning Are Considered

We also conducted experiments to investigate the impact of lying when all the three aspects of learning are used in an agent society. For the same parameter settings of the previous experiment (results shown in Figure 6), the tipping points obtained for

[5] We note this value can be changed.

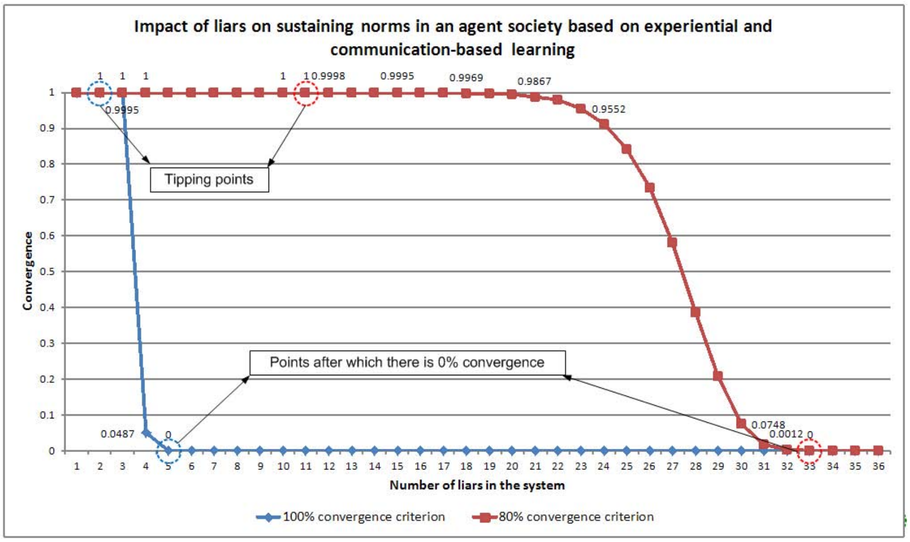


Fig. 6. Tipping points of norm sustainability on varying convergence criteria (80% and 100%)

both 100% and 80% convergence criteria were the same. In both cases, any society that has more than two liars resulted in the destabilization of a norm in the society (not shown here). It is an interesting result because one would assume that the tipping point would be higher (i.e. the set-up can sustain more lying than the result presented in Figure 6) because the impacts of experiential learning and observational learning are the same, hence their additive effect should help in norm stabilization (as opposed to an agent using just the experiential learning). However, it is not the case because once the destabilization phase starts (i.e. liars start impacting the non-liars), the system quickly becomes unstable because the rate of destabilization is twice faster than the previous experiment. It should be noted that the rates of both stabilization and destabilization of a norm are faster when experiential and observational learning are used in conjunction with communication-based learning that permits lying.

6 Discussion

We note that there is scope for further investigation on the lying aspect on norm emergence. In the future the following extensions will be considered.

– The impact of assigning certain weights to each of the three types of learning on the time taken to reach convergence with and without the presence of liars can be investigated.
– The role of network topologies on lying can be investigated in the future. If the agents that are hubs are the liars, then the impact of the lie spreading would be more pronounced than the agent being a leaf node in a network topology.

– Mechanisms for preventing lying in an agent society can be investigated. For example, spreading information about liars using decentralized mechanisms such as gossips [19] can be undertaken.

In this work, we have arrived at the tipping points using a simulation-based approach. It would be desirable to mathematically model scenarios such as the approach used in the work of Brooks et al. [4] to accurately estimate when the liars can thwart the emergence of a norm.

7 Conclusion

The objectives of this paper were two fold. First, in the context of discussing the approaches to norm learning in agent societies, it discussed how three aspects of active learning (learning by doing, observing and communicating) can be integrated to facilitate better norm learning in agent societies. It also demonstrated using a simple example how the three aspects can be integrated. Second, it demonstrated what the effect of liars are on convention emergence when communication-based learning mechanism is used. It identified the tipping points where the convention can no longer be sustained in an agent society.

References

1. Andrighetto, G., Conte, R., Turrini, P., Paolucci, M.: Emergence in the loop: Simulating the two way dynamics of norm innovation. In: Boella, G., van der Torre, L., Verhagen, H. (eds.) Normative Multi-agent Systems. No. 07122 in Dagstuhl Seminar Proceedings, Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany (2007)
2. Axelrod, R.: An evolutionary approach to norms. The American Political Science Review 80(4), 1095–1111 (1986)
3. Boman, M.: Norms in artificial decision making. Artificial Intelligence and Law 7(1), 17–35 (1999)
4. Brooks, L., Iba, W., Sen, S.: Modeling the emergence and convergence of norms. In: International Joint Conference on Artificial Intelligence (IJCAI), pp. 97–102 (2011)
5. Castelfranchi, C., Conte, R., Paolucci, M.: Normative reputation and the costs of compliance. Journal of Artificial Societies and Social Simulation 1(3) (1998)
6. EMIL project report: Emil - emergence in the loop: simulating the two way dynamics of norm innovation. Emil-t, deliverable 5.1, final project report, EMIL consortium (2010), `http://cfpm.org/EMIL-D5.1.pdf`
7. Epstein, J.M.: Learning to be thoughtless: Social norms and individual computation. Computational Economics 18(1), 9–24 (2001)
8. Hamada, M.: Web-based Environment for Active Computing Learners. In: Gervasi, O., Murgante, B., Laganà, A., Taniar, D., Mun, Y., Gavrilova, M.L. (eds.) ICCSA 2008, Part I. LNCS, vol. 5072, pp. 516–529. Springer, Heidelberg (2008)
9. Hoffmann, M.J.: Entrepreneurs and Norm Dynamics: An Agent-Based Model of the Norm Life Cycle. Tech. rep., Department of Political Science and International Relations, University of Delaware, USA (2003)
10. Jones, A.J.I., Sergot, M.: On the characterisation of law and computer systems: The normative systems perspective. In: Deontic Logic in Computer Science: Normative System Specification, pp. 275–307. John Wiley and Sons (1993)

11. Kittock, J.E.: Emergent conventions and the structure of multi-agent systems. In: Nadel, L., Stein, D.L. (eds.) 1993 Lectures in Complex Systems, Santa Fe Institute Studies in the Sciences of Complexity. Addison-Wesley (1995)
12. Lotzmann, U., Möhring, M.: Simulating norm formation: an operational approach. In: Proceedings of the Eighth International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 1323–1324. International Foundation for Autonomous Agents and Multiagent Systems (2009)
13. Neumann, M.: A classification of normative architectures. In: Takadama, K., Cioffi-Revilla, C., Deffuant, G. (eds.) Simulating Interacting Agents and Social Phenomena: The Second World Congress, Agent-Based Social Systems, vol. 7, pp. 3–18. Springer, Heidelberg (2010)
14. Pujol, J.M.: Structure in Artificial Societies. Ph.D. thesis, Departament de Llenguatges i Sistemes Informàtics, Universitat Politènica de Catalunya (2006)
15. Savarimuthu, B.T.R., Cranefield, S.: Norm creation, spreading and emergence: A survey of simulation models of norms in multi-agent systems. Multiagent and Grid Systems 7(1), 21–54 (2011)
16. Savarimuthu, B.T.R., Cranefield, S., Purvis, M., Purvis, M.K.: Role model based mechanism for norm emergence in artificial agent societies. In: Coordination, Organizations, Institutions, and Norms in Agent Systems III, pp. 203–217 (2007)
17. Savarimuthu, B.T.R., Cranefield, S., Purvis, M.A., Purvis, M.K.: Norm identification in multi-agent societies. Discussion Paper 2010/03, Department of Information Science, University of Otago (2010), http://eprints.otago.ac.nz/873/
18. Savarimuthu, B.T.R., Cranefield, S., Purvis, M.A., Purvis, M.K.: Obligation norm identification in agent societies. Journal of Artificial Societies and Social Simulation 13(4), 3 (2010), http://jasss.soc.surrey.ac.uk/13/4/3.html
19. Savarimuthu, S., Purvis, M.A., Purvis, M.K., Savarimuthu, B.T.R.: Mechanisms for the Self-Organization of Peer Groups in Agent Societies. In: Bosse, T., Geller, A., Jonker, C.M. (eds.) MABS 2010. LNCS, vol. 6532, pp. 93–107. Springer, Heidelberg (2011)
20. Sen, S., Airiau, S.: Emergence of norms through social learning. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI), pp. 1507–1512. AAAI Press (2007)
21. Shoham, Y., Tennenholtz, M.: Emergent conventions in multi-agent systems: Initial experimental results and observations. In: Proceedings of the Third International Conference on the Principles of Knowledge Representation and Reasoning (KR), pp. 225–231. Morgan Kaufmann, San Mateo (1992)
22. Stane, S.A.B., Zytniewsk, M.: Normative multi-agent enriched data mining to support e-citizens. In: Cao, L. (ed.) Data Mining and Multi-Agent Integration, pp. 291–304. Springer, Heidelberg (2009)
23. Verhagen, H.: Simulation of the Learning of Norms. Social Science Computer Review 19(3), 296–306 (2001)
24. Villatoro, D., Sabater-Mir, J., Sen, S.: Interaction, observance or both? study of the effects on convention emergence. In: Proceeding of the 2009 Conference on Artificial Intelligence Research and Development: Proceedings of the 12th International Conference of the Catalan Association for Artificial Intelligence, pp. 189–196. IOS Press (2009)
25. Walker, A., Wooldridge, M.: Understanding the emergence of conventions in multi-agent systems. In: Lesser, V. (ed.) Proceedings of the First International Conference on Multi-Agent Systems (ICMAS), pp. 384–389. AAAI Press, Menlo Park (1995)
26. Watkins, C.J.C.H., Dayan, P.: Q-learning. Machine Learning 8(3-4), 279–292 (1992)

Strategy-Proof Mechanisms for Interdependent Task Allocation with Private Durations

Ayman Ghoneim

The Australian National University and NICTA
Canberra ACT, Australia
ayman.ghoneim@anu.edu.au

Abstract. Classical mechanism design assumes that an agent's value of any determined outcome depends only on its private information. However in many situations, an agent's value of an outcome depends on the private information of other agents in addition to its private information. In such settings where agents have interdependent valuations, strategy-proof mechanisms have not been proposed yet, and when these mechanisms are possible is still an open research question. Toward addressing this question, we consider the interdependent task allocation (ITA) problem, where a set of tasks with predefined dependencies is to be assigned to self-interested agents based on what they report about their privately known capabilities and costs. We consider here the possibility that tasks may fail during their executions, which imposes interdependencies between the agents' valuations. In this study, we design mechanisms and prove their strategy-proofness along with other properties for a class of ITA settings where an agent's privately known costs are modeled as privately known durations.

1 Introduction

Given a social choice problem, mechanism design [5] aims to determine the outcome that maximizes the social welfare by designing a payment schema that guarantees that each agent reports its private information truthfully. Classical mechanism design assumes that an agent's value of any determined outcome depends only on its private information. However in many situations, an agent's value of an outcome depends on the private information of other agents in addition to its private information. In such situations, agents have interdependent valuations. Among the many possible examples that involve interdependent valuations, consider the following two examples. When a seller who has private information about the commodity he is selling to a buyer, the buyer's value of the commodity depends on the seller's private information [6]. Or when an agent (e.g., technician, programmer, lawyer, etc.) who is paid to perform a task that depends on a predecessor task assigned to another colleague, the agent's value depends on the colleague's private information of whether the colleague can perform the predecessor task or not. When valuations are interdependent, mechanisms that achieve truthfulness in ex-post incentive compatibility have been introduced (e.g., [6]). However, mechanisms that achieve the strongest and most preferable form of truthfulness in dominant strategy (i.e., strategy-proof) have not been proposed yet for any domain, and when such mechanisms are possible is still an open research question.

Intuitively, and unlike classical mechanism design, strategy-proof mechanisms for interdependent valuations can only be provided for a particular domain, because the interdependencies between the agents' valuations vary depending on the problem, and this directly affects how strategy-proofness is established. Thus, to address this open research question, we need to focus our attention on a particular problem. We study here the interdependent task allocation (ITA) problem, which occurs in various real-life applications, ranging from construction, service providing, to computing and research projects. Adopting a general ITA model, a center wants to assign a set of tasks with predefined dependencies to a number of self-interested agents. Each agent has its own private information that describes which tasks it can execute and the associated costs. Our ITA model considers the possibility that agents may fail in executing their assigned tasks. There are two execution failure models in the literature. The first (e.g., [9]) assumes that agents will make a full effort to execute their assigned tasks, but still may fail accidentally. The second model (e.g., [12]) - and the more nefarious - which we consider here assumes that agents may intentionally fail in their tasks. This happens if agents claim the ability to perform tasks that they actually can't perform in order to increase their payments, if some agents behave irrationally by refusing to execute their tasks even against their best interest, and/or if agents find incentives to refuse to execute their tasks because other agents are causing failures. Intentional failures mimic the one-shot interaction situations where agents don't care much about future implications (e.g., reputation or future opportunities).

Given the interdependencies between tasks and execution failures, an agent may not be able to execute its assigned tasks if their predecessor tasks have failed. This implies that an agent's actual value (i.e., costs) of its assigned tasks may depend on other agents' true private information, and that agents in such settings have *interdependent valuations*. This study advances the state of art by showing that designing strategy-proof mechanisms is possible for the considered ITA model, if we can factorize the agent's privately known cost for performing a task into two components: a privately known duration in which the agent can perform that task, and a publicly known unit cost associated with each duration unit. In particular, we contribute three strategy-proof mechanisms for the ITA problem with intentional failures. The proposed mechanisms differ based on whether the mechanism allocates a task that incurs a negative social welfare or not, and whether the mechanism guarantees profit for the center or not. In the next section, we formulate the task allocation problem as a mechanism design problem. In Sections 3 and 4, we introduce the mechanisms and prove their properties. Section 5 discusses related work, and Section 6 concludes the study and discusses future work.

2 Task Allocation and Preliminary Concepts

Basic Model. Assume a center which has a set $T = \{t_1, \ldots, t_m\}$ of m tasks. There are *predefined* interdependencies (i.e., a partial order) between these tasks, where some tasks can't be executed unless their predecessor tasks were executed successfully. Thus, each task t may have a set of successor tasks $t_{\succ}$, a set of predecessor tasks $t_{\prec}$, and a set of immediate predecessors tasks $t_{\prec}^{im} \subseteq t_{\prec}$ (i.e., t can start immediately after the completion of the tasks in $t_{\prec}^{im}$). The center will gain a reward $R(t)$ (e.g., a market value) for each successful task t. The center wants to allocate the tasks to a set α of n self-interested agents, where each agent has its own private information (i.e., type) and knows nothing about other agents' types. The type $\theta_i = \langle T_i; \{D_i(t), \forall t \in T_i\}\rangle$ of agent i consists of: *1.* the set of tasks $T_i \subseteq T$ that the agent can perform, and *2.* the set $\{D_i(t), \forall t \in T_i\}$ that holds the duration $D_i(t)$ in which the agent can execute each task $t \in T_i$. We assume that agent i has for each task $t \in T_i$ a non-negative *publicly known* unit cost $c_i(t)$ associated with each unit of duration in $D_i(t)$, where $c_i(t)D_i(t)$ is the agent's cost for performing t. This models numerous kinds of agents in real-life (e.g., technicians, programmers, lawyers, etc.), where their costs per hour are publicly known, however, the durations they need to accomplish their tasks are privately known (i.e., depends on private technologies, experience and skill).

Outcome. The center wants to determine an assignment (i.e., outcome) $o = \{(t_1, i, S(t_1)), (t_2, j, S(t_2)), \ldots\}$, where each triple indicates the agent who is assigned a certain task and the starting time for that task, e.g., $(t_1, i, S(t_1))$ means that agent i is assigned t_1 that starts at time $S(t_1)$. For a task t that has no predecessors (i.e., $t_{\prec} = \emptyset$), we set $S(t) = 0$ (any arbitrary initial time representing a point in the future will do). For a task t that has predecessors, $S(t) = max_{t' \in t^{im}_{\prec}} S(t') + D(t')$, where $D(t')$ is the execution duration of t'. Under an outcome o, agent i is assigned the tasks in $T_i(o) = \{t_k | (t_k, i, S(t_k)) \in o\}$. We assume that agent i has enough resources to perform all its assigned tasks in $T_i(o)$, and perform them concurrently if required. Given an outcome o, $T_A(o) = \bigcup_{i \in \alpha} T_i(o)$ is the set of assigned tasks. We stress that an assignment may not include all the tasks in T, e.g., if no agent reported its ability to perform a certain task t, this task and its successor tasks $t_{\succ}$ will not be assigned.

Tentative Values and Efficiency. To perform a task t, agent i will bear a non-negative cost $c_i(t)D_i(t)$, such a cost will be compensated by a payment from the center. Thus, an agent i's tentative value of an outcome o is $v_i(o, \theta_i) = -\sum_{t \in T_i(o)} c_i(t)D_i(t)$. The center's tentative value of an outcome o is $V(o) = \sum_{t \in T_A(o)} R(t)$. Given an outcome o, its social welfare - considering the center and the agents - is $SW(o) = V(o) + \sum_{i \in \alpha} v_i(o, \theta_i)$. Alternatively, the social welfare $SW(o)$ can be viewed as the summation of the social welfare of each assigned task in o, i.e., $SW(o) = \sum_{t \in T_A(o)} SW(t)$, where $SW(t) = R(t) - c_i(t)D_i(t)$ is the social welfare of assigning task t to agent i. Given the vector $\theta = (\theta_1, \ldots, \theta_n)$ of the agents' reported types, the center will determine an efficient outcome $o_d \equiv o_d(\theta)$ from the set of possible outcomes O as follows.

Definition 1. *The determined outcome o_d is* efficient *if o_d maximizes the social welfare, i.e., $o_d = argmax_{o \in O} SW(o)$, and $SW(o_d) \geq 0$.*

Under o_d, each task t is assigned to agent i who can perform it for the lowest cost (i.e., highest $SW(t)$), given the assignment of the task's predecessors. We stress that if the agents who reported their ability to perform a task t have the same unit cost associated with each duration unit, then this task t will be assigned to agent i who can perform t in the shortest duration $D_i(t)$ (i.e., the minimum makespan of t).

Utilities and Mechanism Design. Given the determined efficient outcome o_d, the center pays each agent i a payment $p_i(o_d) \equiv p_i(o_d, \theta)$ for its contributions in o_d. Both the center and the agents have quasi-linear utilities, where the utility of agent i is

$u_i(o_d, \theta_i) = v_i(o_d, \theta_i) + p_i(o_d)$, while the center's utility is $U(o_d, \theta) = V(o_d) - \sum_{i \in \alpha} p_i(o_d)$. To guarantee the efficiency of o_d, the center must propose a payment schema $p_i(o_d)$ for each agent i that guarantees truthful reporting. Clearly, this is a mechanism design problem [5]. We will focus our attention here on *direct revelation* (DR) mechanisms, where an agent reports all its private information to the center that determines o_d and organizes payments to the agents. The revelation principle [7] states that the properties of any mechanism can be replicated by a DR mechanism, and thus, any obtained results here immediately generalize to other indirect mechanisms. The mechanism needs primarily to establish truthfulness under some solution concept (Definition 2), either in *dominant strategies* (i.e., *strategy-proof*) or in *ex-post incentive compatibility*. Dominant strategy implementation is the strongest and most preferable truthfulness form, as an agent reports truthfully irrespective of other agents' reports, and of whether other agents are rational (i.e., never behave in a way that decreases their utilities) or not.

Definition 2. *Given a true type θ_i of agent i, a strategically misreported type θ'_i of agent i, a vector of reported types $\theta_{-i} = (\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_n)$ of other agents except agent i, a determined outcome $o_d \equiv o_d(\theta_i, \theta_{-i})$, and a determined outcome $o'_d \equiv o'_d(\theta'_i, \theta_{-i})$, a DR mechanism achieves* truthfulness *in*

Dominant Strategy: *For any rational agent i, reporting truthfully is always an optimal strategy regardless of whether other agents are reporting truthfully or not, i.e., $\forall i \in \alpha$, $u_i(o_d, \theta_i) \geq u_i(o'_d, \theta_i)$ for any reported θ_{-i}.*

Ex-Post Incentive Compatibility: *For any rational agent i, reporting truthfully is always an optimal strategy given that other agents are reporting truthfully and rational, i.e., $\forall i \in \alpha$, $u_i(o_d, \theta_i) \geq u_i(o'_d, \theta_i)$, given rationality and a truly reported θ_{-i}.*

DR mechanisms are usually required to possess other desirable properties such as *individual rationality* (Definition 3) and *center rationality* (Definition 4).

Definition 3. *A DR mechanism is* individually rational *if every rational and truthful agent i has a non-negative utility, i.e., $u_i(o_d, \theta_i) \geq 0$ given any $o_d \in O$.*

Definition 4. *A DR mechanism is* center rational *if in the truth-telling equilibrium, the center has a non-negative utility, i.e., $U(o_d, \theta) \geq 0$) given any outcome $o_d \in O$.*

Strategic Misreporting. Agent i may increase its utility by strategically misreporting its type to the center. Recalling the type $\theta_i = \langle T_i; \{D_i(t), \forall t \in T_i\}\rangle$ of agent i, agent i can strategically misreport its type in two ways: *1.* over-report its ability to perform more tasks than its actual ability (i.e., over-report $T'_i \supset T_i$), and/or over-report its ability to perform the tasks it actually can perform by reporting a shorter duration than the actual duration (i.e., over-report $D'_i(t) < D_i(t)$ for any task $t \in T_i$); and *2.* under-report its ability to perform tasks that it actually can perform (i.e., under-report $T'_i \subset T_i$), and/or under-report its ability to perform the tasks it actually can perform by reporting a longer duration than the actual duration (i.e., under-report $D'_i(t) > D_i(t)$ for any task $t \in T_i$). Over-reporting implies a larger set of outcomes $O' \supset O$ from which the center will choose an outcome, while under-reporting implies a smaller set of outcomes $O' \subset O$. In the event of over-reporting and under-reporting, the same assignments in O' and O may correspond to different social welfare.

Execution and Schedule Failures. We assume here that a truthful agent will succeed in executing its assigned tasks on schedule if the agent tried to execute them (i.e., no accidental failures), and that any failures happen intentionally. In our ITA model, a failure can be either an *execution failure* where a task is not executed successfully, or a *schedule failure* where a task is executed successfully but didn't finish on schedule. An execution failure happens: *1.* if an agent has an incentive to over-report its ability to perform a task that it can't actually perform, or an incentive to refuse to execute its assigned tasks because other agents are causing failures; and/or *2.* if an agent behaves irrationally and refuses to execute its assigned tasks even though this is not beneficial. On the other hand, a schedule failure happens if an agent has an incentive to over-report its ability to perform a task in a shorter duration than the actual duration it needs to finish the task. An agent causes a failure if it causes an execution failure and/or a schedule failure. However, the two types of failure have different consequences on the successor tasks. Once a task suffered an execution failure, none of its successor tasks will be executed. On the other hand, if a task suffered a schedule failure, its successor tasks can still be executed but they will not start as scheduled because of the delay in their predecessor task. We acknowledge that some ITA applications are time critical (i.e., each task must start at its scheduled time), and a schedule failure will have the same effect as an execution failure. In order to maintain a more general ITA model, we assume in this study that the successor tasks of a task that suffered a schedule failure can still be executed, i.e., we will not assume that execution failures and schedule failures have the same effect. However, we will assume - for now - that agents can handle any delays in the starting time of their assigned tasks without bearing extra costs. We will discuss the scenario in which agents will bear extra costs for delays in Section 4.

Executed Outcome and Actual Values. Given that the tasks not executed due to an execution failure may include tasks that belong to agent i and that agent i bears a task's cost only if the agent executed it, the actual value of agent i is $v_i(o_e, \theta_i) = -\sum_{t \in T_i(o_e)} c_i(t) D_i(t)$. The actual value of agent i may differ from its tentative value $v_i(o_d, \theta_i) = -\sum_{t \in T_i(o_d)} c_i(t) D_i(t)$. We denote by o_e the part of the determined outcome o_d that was successfully executed while neglecting schedule failures, $T(o_e)$ as the set of successful tasks, and $T_i(o_e)$ as the set of successful tasks executed by agent i.

Investment Example. An investment company wants to improve a land's suitability for construction in order to sell it for a higher price. Possible interdependent tasks for the land improvement are site clearing, removal of trees, general excavation, installation of sewer pipes, etc. Assume that the company decided on nine tasks that have the interdependencies $t_1 \prec t_2 \prec \ldots \prec t_9$. The company gets a reward of 10 from each task (i.e., the company can increase the land's selling price by 10 after each task), and wants to assign the tasks to two contactors i and j. As a special case of the general model, we assume here that both contractors have the same publicly known unit cost of 1 for any task t, and thus, the lowest cost for a task corresponds to the shortest duration.

Table 1 includes the true types of the contractors (i.e., θ_i and θ_j), and two misreported types θ'_j and θ''_j of contractor j. We use ∞ to denote the inability of performing a task. If θ'_j was reported, then $o_d = \{(t_1, i, 0), (t_2, j, 3), (t_3, j, 7), (t_4, j, 10), (t_5, i, 15), (t_6, j, 23), (t_7, j, 28)\}$. If contractor j reported its true type θ_j, then t_3 would have been

Table 1. Investment Example

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
θ_i	3	6	6	7	4	∞	6	∞	7
θ_j	15	4	7	5	8	∞	3	∞	3
θ'_j	15	4	3	5	8	5	3	∞	3
θ''_j	5	4	7	5	8	∞	3	4	3

assigned to contractor i, while t_6 and t_7 wouldn't have been assigned at all. Based on o_d, $T_A(o_d)$ will contain the tasks $\{t_1, \ldots, t_7\}$. o_d will suffer a schedule failure at task t_3 because it didn't finish on schedule, and will suffer an execution failure at task t_6 because contractor j can't perform it. Thus, the successfully executed tasks are $o_e = \{t_1, t_2, t_3, t_4, t_5\}$, where $T_i(o_e) = \{t_1, t_5\}$ and $T_j(o_e) = \{t_2, t_3, t_4\}$.

Interdependent Valuations and Profit. Our problem differs from a classical mechanism design problem in two main aspects. *First, interdependent valuations.* Classical mechanism design normally assumes that the value $v_i(o_d, \theta_i)$ of an agent i of o_d depends only on its type θ_i (i.e., independent valuations). Here valuations are interdependent, as the actual value $v_i(o_e, \theta_i)$ of agent i clearly depends on its type, and the actual types of other agents who may cause the failure of the outcome (i.e., $o_d \neq o_e$). *Second, profit.* Normally, classical mechanism design assumes that the central authority that determines the problem's outcome has no value of the determined outcome and solves a social choice problem that involves only the agents. Thus, it is not preferred that this central authority ends up with any left-overs from the agents' payments (which happens if a weakly budget balanced mechanism is used), and redistributing the left-overs using redistribution mechanisms is required. This is not the case in our ITA model, as the center has its value of the determined outcome and any left-over amount contributes toward the center's utility (i.e., profit). Thus, we follow Porter et al. [9] in denoting budget balance as center rationality to point out this issue. However, we stress that our ITA model is not a two-sided (double) auction, because the center doesn't act strategically.

Failure Detection. Although the existence of interdependent valuations complicates the process of designing strategy-proof mechanisms, we can take the advantage of focusing particularly on the ITA problem and make use of the fact that the determined outcome will be executed by the agents. This naturally allows the center to verify some of the agents' reported information during the execution phase. If any task was not executed successfully (i.e., an execution failure), then this will be detected by the center. This was considered in all similar studies [9,10] that deal with the task allocation problem. Moreover, if a task took longer than scheduled (i.e., a schedule failure), then the center can detect that. For instance, assume an online freelance programmer who promised to finish a program in 4 hours. If the program wasn't delivered at all or was delivered after 5 hours, then this can be clearly observed. However, if the programmer finishes the program in 3 hours instead of 4 hours, he still can deliver the program after 4 hours without being detected. Assuming private duration in our ITA model rather than private costs gives us the additional advantage of detecting schedule failures. Under private costs, the center can only detect unexecuted tasks, but an agent can execute a task for a higher or lower cost than the agent's actual cost without being detected. We define the failure-detection assumption that provides a task-by-task monitoring as follows.

Assumption I. Failure-Detection: *If any task t fails - either because it took longer time in execution than scheduled or wasn't executed at all, then the mechanism can detect this failure and identify the responsible agent.*

3 Non-Negative ITA Mechanisms

Unfortunately, it is impossible to achieve center rationality - see the impossibility result of Theorem 7 - if we allow the center to assign a task for a negative social welfare, i.e., the task is assigned for a cost higher than the center's reward of that task. Given that our primary concern in this study is to investigate the conditions under which strategy-proofness can be established, we consider the following two options. In this section, we consider a non-negative ITA (NN-ITA) model where each assigned task must incur a non-negative social welfare (Assumption II), and propose two strategy-proof mechanisms that achieve center rationality. In Section 5, we will allow the center to assign a task even if it incurs a negative social welfare, and propose a strategy-proof mechanism which is not center rational. The NN-ITA model works under the following assumption.

Assumption II. Non-Negative $SW(t)$: *The center will assign a task $t \in T$ only if it incurs a non-negative social welfare, i.e., $SW(t) \geq 0$.*

Assumption II narrows down the situations where an efficient outcome is determined, by eliminating the scenario in which the center should assign a task t for a negative social welfare if this will allow assigning its successor tasks, and these successor tasks have a positive social welfare that compensates the negative social welfare of t. This assumption suggests that we can eliminate the agents' interdependent valuations if we apply a sequence of auctions, and start auctioning a task if its predecessors were executed successfully. This solution works only under Assumption II and has several disadvantages from the practical point of view. For instance in our investment example, the investment company can't know in advance which improvement tasks will be performed to decide the land's final price and start finding a buyer, or when the improvement tasks will finish to set a selling date for the land. Another issue is that we introduce unnecessary time gaps between the execution of tasks by making intermediate auctions/assignments, which can affect the project's duration. For instance, assume that a contractor needs two days to travel to a site, by knowing in advance about his assignment he can arrive to the site when the predecessor tasks were just finished to start his assigned task immediately. This saves a delay of two days in starting his task if the assignment was made after the execution of the predecessor tasks. Thus, we will provide here the more general solution where the whole assignment needs to be determined in advance, which creates interdependencies between the agents' valuations.

3.1 Non-profitable NN-ITA

Given Assumptions I and II, we will define the non-profitable NN-ITA mechanism and prove its properties.

Definition 5. *A* non-profitable NN-ITA mechanism *is defined as follows.*

1. *The center announces the set of the offered tasks T. Then, agents report their types $\theta = (\theta_1, \ldots, \theta_n)$ to the center that will determine an efficient outcome o_d (according to Definition 1, and under Assumption II).*
2. *o_d will be executed resulting in o_e. Each agent i will be paid as follows.*
   a. If agent i caused any execution or schedule failures, then agent i will get no payment, i.e., $p_i(o_e) = 0$.
   b. If the outcome was executed successfully or failed because of another agent $j \neq i$, then agent i will be paid $p_i(o_e) = \sum_{t \in T_i(o_e)} R(t)$.

Theorem 1. *The non-profitable NN-ITA mechanism is individually rational for every rational and truthful agent.*

Proof. The utility of agent i (i.e., $u_i(o_d, \theta_i) = v_i(o_d, \theta_i) + p_i(o_d)$) differs depending on the execution. If agent i caused any execution or schedule failures, then its utility is

$$u_i(o_d, \theta_i) = -\sum_{t \in T_i(o_e)} c_i(t) D_i(t), \tag{1}$$

which is negative if agent i executed any tasks, or 0 if agent i didn't execute any tasks (i.e., $T_i(o_e) = \emptyset$). If o_d was successful (i.e., $o_d = o_e$) or failed because of another agent $j \neq i$, the utility of agent i is

$$u_i(o_d, \theta_i) = \sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t). \tag{2}$$

For every rational and truthful agent i, its utility is as expressed in Eq. 2. It can be rewritten as $u_i(o_d, \theta_i) = \sum_{t \in T_i(o_e)} SW(t)$. Under Assumption II, the $SW(t)$ of any assigned task t is non-negative, implying $u_i(o_d, \theta_i) \geq 0$ and individual rationality. □

Theorem 2. *The non-profitable NN-ITA mechanism is strategy-proof and efficient after imposing Assumption II.*

Proof outline. We classify over-reporting into detected over-reporting (i.e., a task wasn't executed or didn't finish in its schedule time and agent i was detected under Assumption I) and undetected over-reporting (i.e., the outcome o'_d failed due to another agent $j \neq i$ before agent i fails to execute its over-reported tasks). To prove strategy-proofness based on Definition 2, we will prove that $u_i(o_d, \theta_i) \geq u_i(o'_d, \theta_i)$ holds for any θ_{-i}, given that agent i may practise each type of strategic misreporting (i.e., detected over-reporting, undetected over-reporting, and under-reporting) separately. By showing that practicing each lying type separately decreases the agent's utility under o'_d, we will have shown that any combined strategic misreporting that involves more than one lying type may further decrease the agent's utility under o'_d. Once strategy-proofness is established, efficiency directly follows from step 1 in Definition 5. We stress that irrational behaviors by other agents during execution have the same effect of initially misreporting their types θ_{-i}. We stress that the payment applies for all the agents who reported their information, and we don't assume that each agent is necessarily assigned tasks under o_d. Let o'_e be the successfully executed part of o'_d.

Proof. *Detected over-reporting:* The agent's utility is negative or 0 (as in Eq. 1), and $u_i(o_d, \theta_i) = \sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t) \geq u_i(o'_d, \theta_i) = -\sum_{t \in T_i(o'_e)} c_i(t) D_i(t)$ holds because $u_i(o_d, \theta_i)$ (Eq. 2) corresponds to a non-negative utility (established in Theorem 1). *Undetected over-reporting and under-reporting:* If o_d and/or o'_d failed during execution, then this happens by another agent $j \neq i$, and the utility of agent i is as expressed in Eq. 2, and $u_i(o_d, \theta_i) = \sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t) \geq u_i(o'_d, \theta_i) = \sum_{t \in T_i(o'_e)} R(t) - \sum_{t \in T_i(o'_e)} c_i(t) D_i(t)$ holds as follows. *Undetected over-reporting:* This can never increase the agent's utility under o'_d because $T_i(o'_e)$ is not affected, and the agent is only paid the center's rewards for its executed tasks. *Under-reporting:* If agent i denies the ability to perform a task and/or reported longer durations for executing its tasks, then this may decrease the number of assigned tasks to agent i under o'_d. This decreases the utility of agent i, as any task agent i executes successfully contributes a non-negative increase to its utility under Assumption II. □

Theorem 3. *The non-profitable NN-ITA mechanism is center rational.*

Proof. In the truth-telling equilibrium, the center pays each agent the whole reward of its executed tasks, and thus, $U(o_e, \theta) = V(o_e) - \sum_{i \in \alpha} p_i(o_e) = \sum_{t \in T(o_e)} R(t) - \sum_{i \in \alpha} \sum_{t \in T_i(o_e)} R(t) = 0$, which implies center rationality and zero profit. □

3.2 Profitable NN-ITA

We will now propose a profitable NN-ITA mechanism that achieves all the properties possessed by the non-profitable NN-ITA, but in addition, it may possibly yield some profit for the center. We denote α_{-i} as the set of agents without agent i. Given the executed outcome o_e, we define $SW_{-i}(o_e)$ as the social welfare of o_e without the social welfare of the executed tasks by agent i, i.e., $SW_{-i}(o_e) = \sum_{j \in \alpha_{-i}} \sum_{t \in T_j(o_e)} SW(t)$. As well, we define $SW(o^{-i}(o_e))$ as the social welfare of a virtual outcome $o^{-i}(o_e)$ that maximizes the social welfare given the reported types of other agents $j \neq i$. The outcome $o^{-i}(o_e)$ is an efficient assignment of the successfully executed tasks in $T(o_e)$ to agents $j \neq i$, while respecting Assumption II and neglecting the dependencies between the tasks (i.e., a task is assigned even if some of its predecessors are not). *For instance, if θ_i and θ'_j are reported in the investment example: 1. $SW_{-i}(o_e) = SW(t_2) + SW(t_3) + SW(t_4) = 6 + 7 + 5 = 18$; and 2. $SW(o^{-i}(o_e)) = SW(t_2) + SW(t_3) + SW(t_4) + SW(t_5) = 6 + 7 + 5 + 2 = 20$ because $T(o_e) = \{t_1, \ldots, t_5\}$ and when assigning them to contractor j, t_1 is not assigned because of its negative social welfare, and t_2, t_3, t_4 and t_5 are assigned because we neglected their dependency on t_1.*

Definition 6. *A* profitable NN-ITA mechanism *is defined as follows.*

1. *The center announces the set of the offered tasks T. Then, agents report their types $\theta = (\theta_1, \ldots, \theta_n)$ to the center that will determine an efficient outcome o_d (according to Definition 1, and under Assumption II).*
2. *o_d will be executed resulting in o_e. Each agent i will be paid as follows.*
   a. If agent i caused any execution or schedule failures, then agent i will get no payment, i.e., $p_i(o_e) = 0$.

b. If the outcome was executed successfully or failed because of another agent $j \neq i$, then agent i will be paid $p_i(o_e) = \sum_{t \in T_i(o_e)} R(t) + SW_{-i}(o_e) - SW(o^{-i}(o_e))$.

Theorem 4. *The profitable NN-ITA mechanism is individually rational for every rational and truthful agent.*

Proof. The utility of agent i will differ depending on the execution. If agent i caused any execution or schedule failures, then its utility is as expressed in Eq. 1, which is negative or 0. If the execution was successful (i.e., $o_d = o_e$) or failed because of another agent $j \neq i$, then the utility of agent i is

$$u_i(o_d, \theta_i) = \sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t) \\ + SW_{-i}(o_e) - SW(o^{-i}(o_e)). \tag{3}$$

For every rational and truthful agent i, its utility is as expressed in Eq. 3, which can be rewritten as $u_i(o_d, \theta_i) = \sum_{t \in T_i(o_e)} SW(t) + SW_{-i}(o_e) - SW(o^{-i}(o_e)) = SW(o_e) - SW(o^{-i}(o_e))$. Given that $o_{-i}(o_e)$ is determined by assigning the executed tasks $T(o_e)$, $SW(o_e) \geq SW(o^{-i}(o_e))$ holds. This is because agent i executes its tasks in o_e for the lowest possible cost (i.e., highest social welfare), but these tasks are assigned in $o^{-i}(o_e)$ for other agents $j \neq i$ for the second-lowest costs. □

Theorem 5. *The profitable NN-ITA mechanism is strategy-proof and efficient after imposing Assumption II.*

Proof outline. We use here the same proof outline as in Theorem 2. But in addition, we classify under-reporting into under-reporting abilities (i.e., denying the ability to perform a task, or report a longer duration for executing a task that makes the agent's cost for performing the task higher than the task's reward), and under-reporting durations (i.e., report longer duration for executing a task but the agent's cost is still lower than the task's reward).

Proof. *Detected over-reporting:* The agent's utility will be negative or 0 (as in Eq. 1), and thus, $u_i(o_d, \theta_i) \geq u_i(o'_d, \theta_i)$ holds because $u_i(o_d, \theta_i)$ (as in Eq. 3) corresponds to a non-negative utility (established in Theorem 4). *Undetected over-reporting, under-reporting abilities and under-reporting durations:* The utility of agent i is as expressed in Eq. 3, and $u_i(o_d, \theta_i) = \sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t) + SW_{-i}(o_e) - SW(o^{-i}(o_e)) \geq u_i(o'_d, \theta_i) = \sum_{t \in T_i(o'_e)} R(t) - \sum_{t \in T_i(o'_e)} c_i(t) D_i(t) + SW_{-i}(o'_e) - SW(o^{-i}(o'_e))$ holds as follows. *Undetected over-reporting:* This has no effect and can't increase the agent's utility under o'_d because all the payment terms depend on the executed tasks by agent i and other agents. *Under-reporting abilities:* If agent i was the only one capable of performing the task t it under-reported or reported a cost that is higher than the task's reward, then this implies that t will not be assigned under o'_d as well as its successor tasks because no agent can perform it or because of Assumption II, and we have three cases. *Case 1:* If these unassigned tasks were assigned to other agents $j \in \alpha_{-i}$ under o_d, this implies a decrease in the payment that agent i pays the center (i.e., $SW(o^{-i}(o'_e)) < SW(o^{-i}(o_e))$), however, this decrease corresponds to an equal decrease in the agent's received payment from the center (i.e., $SW_{-i}(o_e) < SW_{-i}(o'_e)$). *Case 2:* If these unassigned tasks were assigned to agent i under o_d, then $\sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t)D_i(t) > \sum_{t \in T_i(o'_e)} R(t) - \sum_{t \in T_i(o'_e)} c_i(t)D_i(t)$ holds, as any executed task by agent i corresponds to non-negative increase in its utility under Assumption II. *Case 3:* If these unassigned tasks were assigned to agent i and other agents $j \in \alpha_{-i}$ under o_d, then a combined effect of cases 1 and 2 will happen. *Under-reporting durations:* This doesn't affect the set of tasks that are successfully executed (i.e., $T(o_e) = T(o'_e)$), and thus, $SW(o^{-i}(o'_e)) = SW(o^{-i}(o_e))$. $u_i(o_d, \theta_i) = \sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t)D_i(t) + SW_{-i}(o_e) = SW(o_e)$, and $u_i(o'_d, \theta_i) = \sum_{t \in T_i(o'_e)} R(t) - \sum_{t \in T_i(o'_e)} c_i(t)D_i(t) + SW_{-i}(o'_e) = SW(o'_e)$. $SW(o_e) \geq SW(o'_e)$ holds because the center determines an efficient outcome that maximizes the social welfare. □

Theorem 6. *The profitable NN-ITA mechanism is center rational, and provides a lower-bound on the center's profit.*

Proof. In the truth-telling equilibrium, the center pays $p_i(o_e) = \sum_{t \in T_i(o_e)} R(t) + SW_{-i}(o_e) - SW(o^{-i}(o_e))$ for each agent i. The center's utility of the executed outcome is $U(o_e, \theta) = V(o_e) - \sum_{i \in \alpha} p_i(o_e) = \sum_{t \in T(o_e)} R(t) - \sum_{i \in \alpha} p_i(o_e)$, and we need to show that $U(o_e, \theta) \geq 0$ holds. The term $\sum_{i \in \alpha} \sum_{t \in T_i(o_e)} R(t)$ aggregates the first term $\sum_{t \in T_i(o_e)} R(t)$ of each payment $p_i(o_e)$, and $\sum_{t \in T(o_e)} R(t) = \sum_{i \in \alpha} \sum_{t \in T_i(o_e)} R(t)$. Thus, we can represent the center's utility by the remaining terms in each $p_i(o_e)$, i.e., $U(o_e, \theta) = \sum_{i \in \alpha} SW(o^{-i}(o_e)) - SW_{-i}(o_e)$, and we need to prove that $SW(o^{-i}(o_e)) \geq SW_{-i}(o_e)$ holds for each agent i. Recalling that if a task t was assigned to agent i, then agent i has the lowest cost for performing it, and thus, this assignment leads to the best social welfare from t. Let $SW'(t)$ be the second best social welfare, i.e., assign t to the agent who has the second-lowest cost. $SW(o^{-i}(o_e)) \geq SW_{-i}(o_e)$ holds because $SW(o^{-i}(o_e))$ contains $SW_{-i}(o_e)$, in addition to the second best social welfare $SW'(t)$ from each task t that was executed by agent i in o_e. This guarantees center rationality, and guarantees that the center gains a lower-bound profit of $SW'(t)$ for each successfully executed task t, given that a second-lowest cost exists. □

4 Interdependent Task Allocation Mechanisms

Relaxing Assumption II leads to this impossibility result.

Theorem 7. *If tasks can be assigned for a negative social welfare, no efficient mechanism can achieve center rationality, even under ex-post incentive compatibility and if agents can only under-report abilities.*

Proof. We prove this by an example that shows that if for each offered task there is only one agent who is capable of performing it, then achieving truthfulness in ex-post incentive compatibility contradicts center rationality, because any efficient mechanism - to guarantee truthfulness in ex-post incentive compatibility - must pay each agent its cost for performing a task in addition to the net social welfare of the outcome, and thus, center rationality is lost. Assume that the center has two tasks t_1 and t_2, t_2 depends on t_1, and the center gets a reward of 10 from each (i.e., $R(t_1) = R(t_2) = 10$). Assume that agent i is the only agent who can perform t_1 for $c_i(t_1)D_i(t_1) = 12$, while agent j is the only agent who can perform t_2 for $c_j(t_2)D_j(t_2) = 5$. Assigning t_1 leads to a social welfare of $SW(t_1) = 10 - 12 = -2$, while assigning t_2 leads to a social welfare of $SW(t_2) = 10 - 5 = 5$. The outcome will have a positive social welfare of $SW(o) = SW(t_1) + SW(t_2) = 3$. Now, consider agent i and assume that agent j will report truthfully (i.e., ex-post incentive compatibility). If the center pays agent i an amount less than 12 (the agent's cost), then agent i will under-report its ability to perform t_1. This implies that agent i must be paid an amount more than its cost. Given agent j reported cost, the highest cost the center can allocate t_1 for is 15, otherwise, the $SW(o)$ will be negative and allocating no tasks will be the solution. This implies that agent i can report a cost as high as 15 (i.e., agent i claims a longer duration for performing t_1 and will not be detected under Assumption I). The only way to guarantee a truthful reporting from agent i is to pay the agent an amount equal to or greater than 15. Now, consider agent j and assume that agent i will report truthfully. Repeating the previous analysis, agent j must be paid an amount greater than 5 to prevent it from under-reporting, and an amount equal to or greater than 8 (i.e., the highest cost the center can allocate t_2 for) to prevent the agent from reporting a higher cost than the actual. Given the previous, the center must pay agent i an amount of 15 and agent j an amount of 8, with a total payments of 23. Given that the center gets a total reward of 20 from getting the two tasks performed, the center's payments to the agents are greater than its gained rewards, and center rationality is lost. □

Once center rationality is lost, we can't assume any commercial usage of the mechanism. We shift to a social ITA model that can still be meaningful if center rationality is lost. Assume an unbiased center (e.g., a government) that wants to allocate some tasks (e.g., projects) to the agents in order to maximize their social welfare. Here, the central authority is ready to subsidize the problem in order to maximize the social impact of the outcome. We propose an ITA mechanism that is strategy-proof, individually rational, and efficient, while dismissing Assumption II. Given any task t, we define $R_{\succ}(t)$ as the summation of the rewards for all the successor tasks of t (i.e., $R_{\succ}(t) = \sum_{t' \in t_{\succ}} R(t')$). Given Assumption I, the ITA mechanism works as follows.

Definition 7. *An* **ITA mechanism** *is defined as follows.*

1. *The center announces the set of the offered tasks T. Then, agents report their types $\theta = (\theta_1, \ldots, \theta_n)$ to the center that will determine an efficient outcome o_d (according to Definition 1).*
2. *o_d will be executed resulting in o_e. Each agent i will be paid as follows.*
   a. If agent i caused any execution or schedule failures, then agent i will get no payment, i.e., $p_i(o_e) = 0$.
   b. If the outcome was executed successfully or failed because of another agent $j \neq i$, then agent i is paid $p_i(o_e) = \sum_{t \in T_i(o_e)} R(t) + \sum_{t \in T_i(o_e)} R_{\succ}(t)$.

Theorem 8. *The ITA mechanism is individually rational for every rational and truthful agent.*

Proof. The utility of agent i will differ depending on the execution. If agent i caused any execution or schedule failures, then its utility is as expressed in Eq. 1, which is negative or 0. If the execution was successful (i.e., $o_d = o_e$) or failed because of another agent $j \neq i$, then the utility of agent i will be

$$\begin{aligned} u_i(o_d, \theta_i) = \textstyle\sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t) \\ + \textstyle\sum_{t \in T_i(o_e)} R_{\succ}(t). \end{aligned} \tag{4}$$

The utility of every rational and truthful agent i is as expressed in Eq. 4, and to prove individual rationality, we need to show that Eq. 4 corresponds to a non-negative utility (i.e., $u_i(o_d, \theta_i) \geq 0$). Given that tasks may incur negative social welfare (i.e., $\sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t)$ may be negative), we need to prove that by adding the term $\sum_{t \in T_i(o_e)} R_{\succ}(t)$ which is positive by definition, Eq. 4 will correspond to a non-negative utility. Given that the determined efficient outcome o_d must correspond to a non-negative social welfare, the only benefit from assigning a task that incurs a negative social welfare is to allow the assignment of a portion of its successor tasks, this portion will yield an additional positive increase in the social welfare that compensates the preceding negative social welfare. Given that $R_{\succ}(t)$ serves as an upper bound for any possible positive social welfare resulting from the successor tasks of any task t, and that agent i is paid $R_{\succ}(t)$ for each of its executed task (i.e., $\sum_{t \in T_i(o_e)} R_{\succ}(t)$), the term $\sum_{t \in T_i(o_e)} R_{\succ}(t)$ will compensate any possible negativity in $\sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t)$, and Eq. 4 will correspond to a non-negative utility. □

Theorem 9. *The ITA mechanism is strategy-proof and efficient.*

Proof Outline. Same as Theorem 2.

Proof. *Detected over-reporting:* The agent's utility will be negative or 0 (as in Eq. 1), and $u_i(o_d, \theta_i) = \sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t) + \sum_{t \in T_i(o_e)} R_{\succ}(t) \geq u_i(o'_d, \theta_i) = -\sum_{t \in T_i(o'_e)} c_i(t) D_i(t)$ holds because $u_i(o_d, \theta_i)$ (Eq. 4) corresponds to a non-negative utility (established in Theorem 8). For undetected over-reporting and under-reporting, if o_d and/or o'_d failed during execution, then this happens by another agent $j \neq i$, and the utility of agent i is always as expressed in Eq. 4. We prove that $u_i(o_d, \theta_i) = \sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t) + \sum_{t \in T_i(o_e)} R_{\succ}(t) \geq u_i(o'_d, \theta_i) = \sum_{t \in T_i(o'_e)} R(t) - \sum_{t \in T_i(o'_e)} c_i(t) D_i(t) + \sum_{t \in T_i(o'_e)} R_{\succ}(t)$ holds as follows. *Undetected over-reporting:* This can never increase the agent's utility under o'_d because $T_i(o'_e)$ is not affected, and the agent's utility depends on its executed tasks. *Under-reporting:* If agent i denies the ability of performing a task and/or reported longer durations for executing its tasks, then this may decrease the number of assigned tasks to agent i under o'_d. This can make $\sum_{t \in T_i(o'_e)} R(t) - \sum_{t \in T_i(o'_e)} c_i(t) D_i(t) > \sum_{t \in T_i(o_e)} R(t) - \sum_{t \in T_i(o_e)} c_i(t) D_i(t)$, if both sides are negative. But this corresponds to a greater decrease in $\sum_{t \in T_i(o'_e)} R_{\succ}(t)$. □

Compensation for Delays. Our proposed mechanisms are strategy-proof. However, we assumed that an agent will handle delays caused by schedule failures without bearing extra costs. If agent i will bear extra costs for delays, then its actual value can be rewritten as $v_i(o_e, \theta_i) = -\sum_{t \in T_i(o_e)} c_i(t) D_i(t) - Delay_i(o_e)$, where $Delay_i(o_e)$ is the cost for delays bore by agent i due to schedule failures under the executed outcome o_e. For our mechanisms to maintain strategy-proofness, each agent i must be paid an amount $Comp_i$ as a compensation for delays, where this amount must not depend on the reported information of agent i (e.g., it can be a fixed amount). Given that agent i should not be compensated for delays unless a delay occurred, agent i can under-report to make other agents cause schedule failures [1]. The only way for the center to stop agent i from under-reporting is to pay the agent the compensation amount whether there delays occurred or not, i.e., the term $Comp_i$ will be added to the payment equation of Step 2(b) in Definitions 5, 6 and 7. This makes our proposed mechanisms strategy-proof, efficient and individually rational. However, center rationality will be lost even when considering the NN-ITA model. We omit the formal analysis of this result due to space constraints.

5 Discussion and Related Work

In this section, we discuss previous efforts in designing mechanisms for interdependent valuations, and clarify the differences between this study and related studies.

Interdependent Valuations. We stress that outcome failure problems (e.g., task allocation, multiagent planning) are not the only type of problem that involves interdependent valuations (see [6] for other examples), and if tasks are not interdependent (i.e., independent valuations), strategy-proof mechanisms already exist (e.g., [8]). When valuations are interdependent, a Groves mechanism [3] loses its strategy-proofness, because its payment depends on the agents' tentative values, i.e., its payment schema for our ITA model for any agent i will be $p_i(o_d) = \sum_{t \in T_i(o_d)} R(t) + SW_{-i}(o_d) - h_i(\theta_{-i})$, where o_d is the tentative outcome and $h_i(\theta_{-i})$ is a term that doesn't depend on the reported type of agent i. Clearly here, agent i will over-report in order to make more tasks assigned under o'_d, and this will increase its payment, i.e., $\sum_{t \in T_i(o'_d)} R(t) + SW_{-i}(o'_d) \geq \sum_{t \in T_i(o_d)} R(t) + SW_{-i}(o_d)$.

All previous efficient mechanisms for interdependent valuations settings achieve truthfulness at ex-post incentive compatibility [2]. Mezzetti [6] introduced a two-stage Groves mechanism, which works for any interdependent valuations problem under the assumption that the agents will realize their actual values after the outcome is determined (e.g., additional information is revealed). The two-stage Groves mechanism is identical to a Groves mechanism, except for a second reporting phase, where agents report their actual values of the determined outcome, and the Groves payment is made based on these actual values. This second reporting phase can be eliminated under Assumption I, as the center is monitoring the outcome and knows the agents' actual values.

[1] Contractor i can create a delay in the investment example as follows. If contractors i and j reported the types θ_i and θ''_j as in Table 1, then o_d will be successful. However, if contractor i reported θ'_i that indicates a duration of 7 for t_1 instead of 3, then t_1 will be assigned to contractor j in o'_d who claimed finishing t_1 in 5 instead of 15 according to its true type θ_j. Thus, t_1 will suffer from a schedule failure, and agent i will get the compensation amount.

[2] An efficient mechanism that achieves truthfulness in dominant strategy was proposed for multiagent planning [4], and then was shown to be incorrect and was dismissed in [12].

Given the Groves uniqueness [2] (i.e., under mild assumptions, the only efficient and strategy-proof mechanisms among the DR mechanisms), the fact that Mezzetti's mechanism and our mechanisms have similarities with the Groves mechanism is expected and unavoidable. Both Mezzetti's mechanism and all our mechanisms calculate all the payment terms based on the actual values (not the tentative values). In Mezzetti's mechanism and our profitable NN-ITA mechanism (Definition 6), the Groves payment is used based on the actual values of the agents (i.e., $p_i(o_e) = \sum_{t \in T_i(o_e)} R(t) + SW_{-i}(o_e) - h_i(\theta_{-i})$). Our profitable NN-ITA mechanism is similar to the Vickrey-Clarke-Groves (VCG) mechanism [11,1,3], which uses the Clarke pivot term [1] to define the $h_i(\theta_{-i})$ function in the Groves mechanism. However, in the profitable NN-ITA mechanism, the Clarke pivot term $h_i(\theta_{-i}) = SW(o^{-i}(o_e))$ depends on the private information of agent i, and is calculated based on the executed outcome while neglecting the interdependencies between tasks and imposing Assumption II.

Domain specific mechanisms for outcome failure problems can handle failures easily, as agent i can be the only agent behind the outcome failure (i.e., other agents are reporting truthfully under ex-post incentive compatibility). In [9,10], ex-post incentive compatible mechanisms were proposed for task allocation, where valuations were interdependent in the first because of the interdependencies between tasks, while in the second because of assuming a trust-based model. In [12], an ex-post incentive compatible mechanism was proposed for multiagent planning (MAP), where valuations were interdependent because of the interdependencies between the plans executed by different agents. The MAP model is more complicated than an ITA model, as interdependencies between actions are not pre-defined, and agents report their own goals and the rewards associated with their goals.

Accidental Failures. Previous studies (e.g., [9]) assume that an outcome may fail accidentally by assuming that an agent privately knows its probability of success (PoS) when performing a particular task. An accidental ITA model (e.g., [9,10]) assumes the following: *1.* Agents may fail in executing their tasks even if they reported their PoS truthfully; and *2.* If a task failed, then it will not be repeated again (i.e., an agent will not keep repeating the task until it succeeds). We consider intentional failure here because it has the same effect of the previous two points, i.e., in both accidental and intentional models, valuations are interdependent because the successor tasks of any failing task will not be executed. The work presented here can be easily extended to an accidental failure ITA model, but we leave that for future work.

Private Costs and Resources. Previous studies [9,10,12] assume private costs instead of durations as we consider here. Assuming private durations gives us an additional advantage under Assumption I, because if an agent claims the ability to perform a task in a shorter period than its actual capability, then this agent will be detected. This is not possible when assuming private costs, as the center can't verify the cost for which an agent executed a task. With private costs, an agent can execute a task for a higher or lower cost than the agent's actual cost without being detected. When assuming private costs, it is impossible to design efficient and strategy-proof mechanisms for the ITA problem that use a single-allocation round. However, we stress that designing efficient and strategy-proof mechanisms when assuming private costs is possible if we reassigned the failing tasks. Considering resources for executing tasks, we assumed here that agents have sufficient resources. This is convenient in scenarios where an agent may acquire the required resources to perform its allocated tasks after the allocation phase by - for example - hiring personal and buying raw material. When agents can't acquire additional resources, agents are considered to have limited resources, e.g., an agent can perform t_1 and t_2, but has resources only to perform one of them. Assuming limited resources adds more complications, and for instance, it is impossible to achieve center rationality even under ex-post incentive compatibility in a NN-ITA model which uses Assumption II. We will propose the reassignment mechanisms that handle private costs and investigate the limited resources ITA problem in a future study.

6 Conclusions and Future Work

We proposed three mechanisms and showed that designing efficient and strategy-proof mechanisms is possible when valuations are interdependent. Interdependent valuations introduce a lot of complexities to the classical mechanism design problem, which only can be handled by designing domain specific mechanisms especially if maintaining truthfulness in dominant strategy is required. We stress that achieving truthfulness in dominant strategy while sacrificing other properties (e.g., center rationality) may not always be the best choice. On the other hand, assuming that an agent will report truthfully because it believes that other agents are doing the same (i.e., ex-post incentive compatibility) can also be unreasonable in many situations (e.g., allocating tasks in a competitive market where naturally an agent will not trust that its rivals are reporting truthfully). Moreover, achieving truthfulness in dominant strategy is motivated by the fact that it is not always possible to assume that all involved agents are fully rational. We argue that having different mechanisms that possess different properties (e.g., achieve different forms of truthfulness) is vital for allowing us to handle various real-life applications, each with its requirements. As this was the first step in designing such mechanisms for applications where valuations are interdependent, extending the current work to consider combinatorial values in ITA, and to multiagent planning appear fruitful avenues of pursuit.

Acknowledgment. The author is indebted to Jussi Rintanen and Patrik Haslum for helpful discussions and suggestions for improving the exposition of the material in this paper. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

1. Clarke, E.H.: Multipart pricing of public goods. Public Choice 11, 17–33 (1971)
2. Green, J.R., Laffont, J.J.: Characterization of satisfactory mechanisms for the revelation of preferences for public goods. Econometrica 45, 427–438 (1977)
3. Groves, T.: Incentives in teams. Econometrica 41, 617–631 (1973)
4. van der Krogt, R., de Weerdt, M.M., Zhang, Y.: Mechanism design and multiagent planning. In: The 18th ECAI, pp. 423–427 (2008)

5. Mas-Colell, A., Whinston, M.D., Green, J.R.: Microeconomic Theory. Oxford Uni. Press (1995)
6. Mezzetti, C.: Mechanism design with interdependent valuations: Efficiency. Econometrica 72(5) (2004)
7. Myerson, R.B.: Incentive compatibility and the bargaining problem. Econometrica 47(1), 61–73 (1979)
8. Nisan, N., Ronen, A.: Algorithmic mechanism design. Games and Economic Behavior 35 (2001)
9. Porter, R., Ronen, A., Shoham, Y., Tennenholtz, M.: Fault tolerant mechanism design. Artificial Intelligence 172(15), 1783–1799 (2008)
10. Ramchurn, S.D., Mezzetti, C., Giovannucci, A., Rodriguez-Aguilar, J.A., Dash, R.K., Jennings, N.R.: Trust-based mechanisms for robust and efficient task allocation in the presence of execution uncertainty. Journal of Artificial Intelligence Research 35, 119–159 (2009)
11. Vickrey, W.: Counterspeculation, auctions, and competitive sealed tenders. The Journal of Finance 16, 8–37 (1961)
12. Zhang, Y., de Weerdt, M.M.: Creating incentives to prevent intentional execution failures. In: IEEE/WIC/ACM, pp. 431–434 (2009)

Costly Voting with Sequential Participation

Ryuya Kagifuku and Shigeo Matsubara

Department of Social Informatics, Kyoto University,
Yoshida-honmachi, Sakyo-ku, Kyoto, 606-8501, Japan
kagifuku@ai.soc.i.kyoto-u.ac.jp, matsubara@i.kyoto-u.ac.jp

Abstract. This paper examines the property of the m votes to win mechanism. Voting is an effective way to make a collective decision but voting behaviors, e.g., monitoring the voting process, may incur a cost, that is, voting is often costly. In this case, compulsory voting incurs a larger cost. Random decision making can reduce the cost for voting but is skeptical in the quality of decision making. That is, we face the problem of how to balance the quality of collective decision making with the reduction of the cost for voting. To solve this problem, this paper focuses on the m votes to win mechanism, in which voters sequentially vote and if an alternative receives m votes, the voting process immediately terminates and the alternative received m votes wins. The similar voting mechanism is actually used in the Apache projects. However, the property of the m votes to win mechanism has not sufficiently studied. The questions include how to find a desirable value of m and what situation this mechanism is superior to other mechanisms. To answer these question, we create the discussion model where two alternatives is included, and analyze what voting strategy is rational. Based on the analysis, we examine what factors affects the social surplus, i.e., to what extent the quality of collective decision making and the reduction of the cost for voting are well balanced, and clarify whether the m votes to win mechanism is superior to the compulsory voting or the random decision making in terms of social surplus.

Keywords: Mechanism design, Game theory, Social surplus, Collective decision making.

1 Introduction

Voting is an effective way of collective decision making. The nature of the various voting mechanisms have been studied in economics and the game theory, and have recently been actively studied in the multi-agent systems researches [8,11, 4,7]. Furthermore, the areas of employing voting to aggregate various opinions have been widespread in the real field. For example, in the Apache projects whether the package will be released or not is determined by voting.

The characteristics of the voting mechanism in Apache are (1) sequential participation and (2) the mechanism of m votes to win[1]. Voters sequentially vote

[1] http://www.apache.org/foundation/voting.html

during the voting period and they can know how many votes each alternative obtains so far. In addition, the voting process immediately ends if an alternative obtains m votes, and whether the proposal is passed or rejected is determined. For example, an Apache project, which includes around twenty members, employs the voting mechanism of three votes to win.

The voting mechanism of m votes to win is different from the simple majority voting. Suppose there are two alternatives, A and B. In the m votes to win mechanism, once alternative A has m votes, the voting process is immediately terminated and alternative A wins. Even if other $m + 1$ voters claim that they are in favor of alternative B, they cannot reverse the decision of A's win in the m votes to win mechanism, while it may happen in the simple majority voting.

Another characteristic of the voting mechanism discussed in this paper is sequential voting. Voters do not vote at the same time but vote sequentially, and they can observe the preceding results of other voters' votes. In Apache projects, a vote is done by sending an e-mail to the mailing list, which enables voters to observe the votes by the preceding voters.

Note that the term of sequential voting are used in different way. One is that voters vote sequentially. For example, suppose that three voters a, b, c exist. Voters a, b, c express their preferences one by one and the alternative which finally gets more votes than others wins. The other is that the alternatives are compared incrementally, which is related to the well-known Condorcet paradox. For example, suppose that three alternatives A,B,C exist. First, the organizer asks voters to vote the preferred one between A and B, then the organizer asks voters to vote the preferred one between the winner in the firs vote and C. This paper deals with the former one.

In this paper we assume that voting is costly. Because it is true in the real world and it enables us to explain that the m votes to win system is better than the compulsory voting. Costly voting means that the disutility is associated with voting behavior. It corresponds to the burden that voters have to go to the voting place in the specific day if voting is carried out in the off-line environments. The cost for voting also exists in the online environments. Voters have to keep it constantly monitors when that vote is accepted.

The readers may consider that the problem of costly voting can be solved if computational agents are introduced and behave on behalf of a human. In such a case, the monitoring costs can be negligible. However, acquiring an accurate preference from a human is still difficult. This means that we have to consider the situation that agents and humans are simultaneously included in the voting process, which makes the discussion about the costly voting significant. Actually, in some cases in the Apache voting, exercise of a vote carries some responsibilities. That is, a favorable vote carries the implied message "I approve and I am willing to help." Also, an unfavorable vote may imply "I disapprove, but I have an alternative and will help with that alternative." These are the reasons that we consider costly voting.

Another type of cost for voting is the cost for examining the alternatives. For example, the nominees have to spend their time for reading the candidate papers to nominate the best paper. This paper, however, does not consider this type of voting cost but focuses on the situation that voters know their preferred alternative but expressing it incurs some cost.

The question here is the quality of decision making, i.e., whether the result of votes is consistent with the alternative that the majority of the community members support. The m votes to win mechanism does not guarantee that the alternative supported by the majority is always elected. On the other hand, if the organizer forces all members to vote and the number of the alternatives are two, the alternative supported by the majority is always elected. However, especially in online communities, implementing compulsory voting is difficult because it causes the burden to the members. Here, we face the problem of how to balance out the quality of collective decision making and the costs for voting. The objective of this paper is to examine this issue.

As mentioned above, the m votes to win mechanism is not our invention but has already been used in the real world. However, it has not been sufficiently examined whether setting $m = 3$, which is used in the voting in Apache projects, is effective or not to balance the quality of collective decision making and the reduction of the voting cost. If we apply the same voting mechanism to other fields, it is not clear what factors such as the number of community members, the distribution of the preferences to each alternative affect the efficiency. The contributions of this paper is (1) provisioning the model of the costly voting with sequential participation, (2) developing a method of calculating the optimal voting behavior based on dynamic programming, (3) providing the information for selecting the design parameters such as m, the length of the voting period, and (4) clarifying what conditions the m votes to win mechanism is superior to the compulsory voting or the random decision making to answer the above questions.

2 Related Work

The costly voting has been studied by Börgers [3]. His question is whether participation in votes should be voluntary or compulsory and how much pressure should be exerted on individuals to participate in votes. He compared three voting mechanisms: compulsory voting, voluntary majority voting, and random decision making. Compulsory voting can attain the highest quality of decision making, i.e., the alternative that the majority of the voters prefer is selected, but the costs for voting is worst. In contrast, random decision making that picks a voter and obeys his/her preference, i.e., a dictatorial decision making. This can minimize the cost for voting but the quality of the collective decision making is not clear. Börgers showed that voluntary majority voting is superior to compulsory voting and social surplus obtained by the majority voting is always larger than that obtained by the compulsory voting. Our study is different from Börgers's study in that our study deals with m votes to win mechanism with the sequential votes, while Börgers deals with the majority voting with simultaneous votes.

Another related study is done by Battaglini [2]. They examined majority voting mechanism with voters' sequential participation. In national elections, the early voting is often allowed, that is, eligible voters can vote in the early day as well as in the day of election. The persons voting in the day of election can obtain rich information than the persons voting in the early day because the news media report their analysis during the election campaign period. This is a reason that Battaglini examined the sequential voting. They argued how the cost of voting affects the winning alternative. Our study is different from Battaglini's study in that our study deals with the m votes to win mechanism, while Battaglini studied the majority voting mechanism.

Desmedt and Elkind studied the properties of equilibrium outcomes of plurality voting with abstentions, for both simultaneous and sequential voting [6]. They showed that an important advantage of sequential voting is that it always has an equilibrium in pure strategies. They examined plurality voting with abstentions by assuming that the voters know each others' true preferences and the voters vote one by one in an exogenously determined order, which seems too restrictive. On the other hand, our study examines the m votes to win mechanism by assuming that voters know only the probability of preferring alternative A. This is the difference from Desmedt's study.

Sequential voting mechanisms have also been studied by the economists [12, 10,5] and by the computer scientists [1,13,9], but these studies do not focus on abstentions. Our study is different from these studies in that we deal with the m votes to win mechanism.

3 Model

3.1 Model of Voting

To make rigorous discussions, this section gives a model of voting. In the community, n agents $(i = 1, 2, ..., n)$ exist. To avoid trivial case distinctions, we assume $n \geq 3$. The community has to choose one of two alternatives: $a = \{A, B\}$. This is a collective decision making problem. One alternative must be chosen, and this alternative will apply to all members of the community. An example would be that the community (workers in the same room) faces the energy problem and has to select A (turn on the air conditioner) or B (turn off the air conditioner).

Here, it may happen that neither A nor B obtain m votes in the m votes to win mechanism. In such a case, we cannot determine the outcome. To avoid this situation, we assume that B is an alternative of the status quo and B is selected if neither A nor B obtain m votes in the m votes to win mechanism.

The relevant characteristics of an agent are summarized in that agent's "type" $\theta_i = (a_i, c_i)$ in $\{A, B\} \times \{c_H, c_L\}$ where the first component, a_i, is the alternative which agent i favors, and the second component, c_i, indicates agent i's costs of participating in a collective decision making process. To make discussion simple, we assume only two cases exist about the cost: high cost and low cost. c_H corresponds to the high cost and c_L corresponds to the low cost and $c_H > c_L$ holds.

If agent i is of type $\theta_i = (a_i, c_i)$, then i's von Neumann Morgenstern utility is highest if i's most favored alternative, a_i, is chosen, but agent i does not participate in the decision making process. In that case, agent i's utility is normalized to be equal to 1. If the alternative which i ranks second is chosen, and i does not participate, then i's utility is equal to 0. Now consider the utility of agent i if i does participate in the decision making process. In this case we simply subtract from the utilities described so far agent i's participation costs c_i. Hence, if a_i is chosen and agent i does vote, then his/her utility is $1 - c_i$, and if a_i is not chosen, and i does vote, then his/her utility is $-c_i$. We assume that $1 - c_i \geq 0$. If $1 - c_i \geq 0$ does not hold, no one votes because we do not assume the monetary transfer between voters is available. Note that we are assuming that the costs of participation are independent of whether agent i's participation is compulsory or voluntary. The costs are also independent of the decision making mechanism which society uses, of the strategy which agent i chooses in that mechanism, and of the alternative which society chooses. These assumptions are made for simplicity.

Social surplus is the sum of all the voters' utility. Voting mechanisms can be evaluated from the various viewpoints. This paper focuses on how to balance the quality of collective decision making and the reduction of the cost for voting. Thus, we use social surplus as a measure for evaluating voting mechanisms.

We assume p designates the ratio of preferring the alternative A and q designates the ratio of having the high cost for voting. Therefore, the probability that a voter prefers alternative A and has the high cost is pq, the probability that a voter prefers alternative A and has the low cost is $p(1 - q)$, the probability that a voter prefers alternative B and has the high cost is $(1 - p)q$ and the probability that a voter prefers alternative B and has the low cost is $(1 - p)(1 - q)$. We assume that all the voters know the value of p and q. In the real world, it is often difficult to obtain the accurate values of p and q. We will discuss later what outcome is obtained if m is chosen without knowledge about the accurate values of p and q.

The voting period has been predetermined. In the Apache projects, the voting period is usually set to 72 hours. Voters will have the opportunity to vote in an exogenously determined order. This models that different voters find that voting is held at the different time. Voters are allowed at his/her turn to vote for A, vote for B, or do nothing. Once he/she passes the opportunity to vote, we assume that he/she does not come back to the voting process nor vote an alternative, that is, he/she does not postpone the decision making. This is because examining when to come back to the voting process incurs additional costs.

When the voters vote, they can observe how many votes each alternative receives until now. We introduce the notation of $\#A$ and $\#B$ to represent these values.

3.2 Voting Mechanisms

We compare three voting mechanisms: compulsory, m votes to win, and random decision making. Compulsory voting forces all the voters to reveal their preferences. We restrict the discussion to that the number of alternatives is two. Thus, we do not have to consider the strategic manipulation such as viewed in the Condorcet paradox. In the m votes to win mechanism, participations are voluntary rather than mandatory and once an alternative receives m votes, the voting process immediately terminates and the alternative that receives m votes wins. In the random decision making, it picks a voter and obeys his/her preference without voting. Note that this definition is different from the Börgers' paper. In his paper, the random decision making means the mechanism in which no agent is invited, nor indeed allowed, to participate in the decision making. Each of the two alternatives is selected with probability 1/2, while in our study, the probability of selecting alternative A depends on the value of p.

4 Optimal Voting Strategies in the m Votes to Win Mechanism

This section examines an optimal voting strategy in the m votes to win mechanism. An optimal strategy of the voter can be obtained by dynamic programming, i.e., by thinking backward from the n-th voters. Consider the k-th voter. From the assumptions, the k-th voter cannot manipulate the first to $(k-1)$-th voters' behaviors, which includes the behavior of abstention as well as voting for A and voting for B. Thus, k-th voter is sufficient to consider the $(k+1)$-th to n-th voters' behaviors to determine his/her behavior. Here, we show the behaviors of $(k+1)$-th to n-th voters' behaviors can be aggregated as the probabilities of alternative A's win, alternative B's win, and no alternative's win at the $(k+1)$-th voter's turn. From the assumption, the case of no alternative's win is dealt with the alternative B (the status quo)'s win.

First, consider the behavior of the last voter, i.e., the n-th voter. The optimal behavior of the n-th voter can simply be described as follows. If he/she prefers alternative A and $\#A = m-1$, that is, it can reach m votes by his/her vote for A, he/she should vote for alternative A because his/her vote for A is an only way to win A and we assume that $1 - c_n \geq 0$. Otherwise, he/she should choose abstention because he/she suffers a loss of $-c_n$ by his/her vote for A. On the other hand, if he/she prefers alternative B, he/she should choose abstention because no other voters exist after his/her turn and alternative A no longer wins even if he/she chooses abstention.

Next, consider the behavior by the second last voter, i.e., the $(n-1)$-th voter preferring alternative A. First, consider the case that $\#A = m-1$. If he/she votes for A, he/she obtains the utility of $1 - c_{n-1}$, otherwise he/she obtains the utility p. This is equal to the probability that the n-th voter prefers alternative A. Therefore, if $1 - c_{n-1} \geq p$ holds, the $n-1$ voter preferring A should vote for A, otherwise he/she should choose abstention. Second, consider the case that $\#A = m-2$. If he/she votes for A, he/she obtains the utility of $p \times (1 - c_{n-1}) + (1-p) \times (-c_{n-1})$=$p - c_{n-1}$, otherwise 0 utility. Thus, if $p - c_{n-1} \geq 0$ holds, the $(n-1)$-th voter preferring A should vote for A, otherwise he/she should choose abstention. Third, consider the case that $\#A < m-2$. In this case, even if both

$(n-1)$-th and n-th voters vote for A, the number of votes for A cannot reach m. Thus, the $(n-1)$-th voter should choose abstention. The behavior of the $(n-1)$-th voter preferring alternative B can be analyzed in the same manner.

Here, we can represent the probability of A's win if the opportunity of vote comes to the $(n-1)$-th voter preferring A. If $\#A = m-1$ and $1 - c_{n-1} \geq p$, the probability of A's win is 1, else if $\#A = m-1$ and $1 - c_{n-1} < p$, the probability of A's win is p, else if $\#A = m-2$ and $p - c_{n-1} \geq 0$, the probability of A's win is p, otherwise the probability of A's win is 0. We can also calculate the probability that the number of votes for B will reach m and the probability that no alternative will receive m by subtracting from 1 the probability of A's win and the probability that the number of votes for B will reach m. Thus, the probability of B's win can be calculated. The set of the probabilities of A's win and B' win at the $(n-1)$-th voters' turn enables us to calculate the $(n-2)$-th voters optimal behavior without directly examining the n-th voter's behavior.

By using $prob_{a_i,c_i}(X, i, s, t)$ we represent the probability of alternative X's win if the opportunity of vote comes to the i-th voter whose type is $(a_i,\, c_i)$. Here, $X = \{A, B, \emptyset\}$. $\emptyset$ corresponds to the case that no alternative wins. Such a case will be dealt with as B's win, but we distinguish the case that alternative B receives m votes and B wins and the case that no alternative receives m votes and B wins.

Now, we can calculate the optimal behavior of the k-th voter preferring alternative A, $behavior_{A,c_k}(k, s, t)$ conditioned by $\#A = s$ and $\#B = t$. Here, $behavior_{A,c_k}(k, s, t) = 1$ means that the k-th voter should vote for A, while $behavior_{A,c_k}(k, s, t) = 0$ means that the k-th voter should choose abstention. In the following expressions, $u01$ and $u02$ represent the probability of A's win and B's win, respectively, if the k-th voter chooses abstention, while $u1$ and $u2$ represent the probabilities of A's win and B's win, respectively, if the k-th voter votes for alternative A.

$$\begin{aligned}
u01 &= p * q * prob_{A,c_k}(A, k+1, s, t) + p * (1-q) * prob_{A,c_k}(A, k+1, s, t) \\
&\quad + (1-p) * q * prob_{B,c_k}(A, k+1, s, t) \\
&\quad + (1-p) * (1-q) * prob_{B,c_k}(A, k+1, s, t) \\
u02 &= p * q * prob_{A,c_k}(B, k+1, s, t) + p * (1-q) * prob_{A,c_k}(B, k+1, s, t) \\
&\quad + (1-p) * q * prob_{B,c_k}(B, k+1, s, t) \\
&\quad + (1-p) * (1-q) * prob_{B,c_k}(B, k+1, s, t) \\
u1 &= p * q * prob_{A,c_k}(A, k+1, s+1, t) + p * (1-q) * prob_{A,c_k}(A, k+1, s+1, t) \\
&\quad + (1-p) * q * prob_{B,c_k}(A, k+1, s+1, t) \\
&\quad + (1-p) * (1-q) * prob_{B,c_k}(A, k+1, s+1, t) \\
u2 &= p * q * prob_{A,c_k}(B, k+1, s+1, t) + p * (1-q) * prob_{A,c_k}(B, k+1, s+1, t) \\
&\quad + (1-p) * q * prob_{B,c_k}(B, k+1, s+1, t) \\
&\quad + (1-p) * (1-q) * prob_{B,c_k}(B, k+1, s+1, t)
\end{aligned}$$

In the expression of $u01$, the first term corresponds to the case that the $(k+1)$-th voter prefers A and has the high voting cost and $\#A = s, \#B = t$. The second term corresponds to the case that the $(k+1)$-th voter prefers A and has the low voting cost and $\#A = s, \#B = t$. The third term corresponds to the case that the $(k+1)$-th voter prefers B and has the high voting cost and $\#A = s, \#B = t$. The fourth term corresponds to the case that the $(k+1)$-th voter prefers B and has the low voting cost and $\#A = s, \#B = t$. In the expressions of $u02$, $u1$ and $u2$ can be read as the same manner.

- if $s = m - 1$,
  - if $1 - c_k \geq u01$,

$$\begin{aligned} behavior_{A,c_k}(k, s, t) &= 1 \\ prob_{A,c_k}(A, k, s, t) &= 1 \\ prob_{A,c_k}(B, k, s, t) &= 0 \\ prob_{A,c_k}(\emptyset, k, s, t) &= 0 \end{aligned}$$

  - otherwise

$$\begin{aligned} behavior_{A,c_k}(k, s, t) &= 0 \\ prob_{A,c_k}(A, k, s, t) &= u01 \\ prob_{A,c_k}(B, k, s, t) &= u02 \\ prob_{A,c_k}(\emptyset, k, s, t) &= 1 - u01 - u02 \end{aligned}$$

- if $s < m - 1$
  - if $u1 - c_k \geq u01$,

$$\begin{aligned} behavior_{A,c_k}(k, s, t) &= 1 \\ prob_{A,c_k}(A, k, s, t) &= u1 \\ prob_{A,c_k}(B, k, s, t) &= u2 \\ prob_{A,c_k}(\emptyset, k, s, t) &= 1 - u1 - u2 \end{aligned}$$

  - otherwise

$$\begin{aligned} behavior_{A,c_k}(k, s, t) &= 0 \\ prob_{A,c_k}(A, k, s, t) &= u01 \\ prob_{A,c_k}(B, k, s, t) &= u02 \\ prob_{A,c_k}(\emptyset, k, s, t) &= 1 - u01 - u02 \end{aligned}$$

We can calculate the optimal behavior of the k-th voter preferring alternative B as in the same manner as the above calculation for the voter preferring A.

The optimal behavior of k-th voter can be calculated based on the probability of alternative X' win at $(k+1)$-th voter. In addition, the behavior of the last n-th voter can be characterized as discussed above. Thus, we can attain to characterize the optimal voting strategies of all the voters.

5 Efficiency of the Voting Mechanisms

5.1 Experimental Setup

From the above discussion, we can calculate the voting behaviors of the rational voters and what outcome can be obtained by using the m votes to win mechanism. We examine the property of the m votes to win mechanism by the following three steps.

- What value is the optimal for m in the m votes to win mechanism?
- When do the voters actually vote in the m votes to win mechanism?
- Which mechanism is superior to other mechanisms in terms of social surplus, the m votes to win mechanism, compulsory voting, or random decision making?

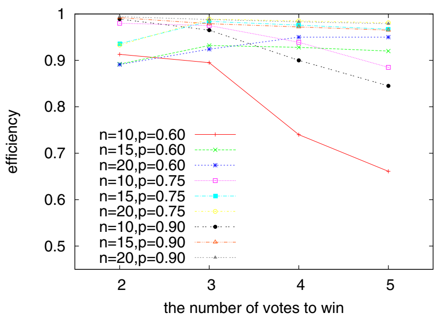
A design parameter in the m votes to win mechanism is m. If we employ the m votes to win mechanism, we have to select m suitable for the situation on the question. To examine this issue, we calculate the social surplus for the combinations of the following parameters.

- The number of voters: $n = \{10, 15, 20\}$
- The probability of preferring alternative A: $p = \{0.6, 0.75, 0.9\}$
- The probability of having the high voting cost: $q = \{0.1, 0.5, 0.9\}$

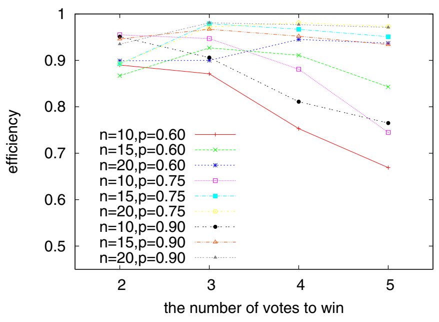
These values of the parameters are chosen to cover various cases and clarify the general properties of the m votes to win mechanism. This is the same as in the following two experiments.

Next, we examine the frequency of voting by the k-th voter on the various conditions. The combinations of the parameter are the same as the first experiment except setting $m = 3$. This is related to the issue of how long the voting period should be. Each voter faces the problem of choosing one of the following two behaviors, acting by itself or leaving the vote to the voters appearing in the succeeding turns. If the voting behaviors occur only in the late turns, setting the long period for voting has little significance. This is because even if the voting period is long, the organizer is likely to receive few votes in the period except the last several turns, which will give the same outcome if the voting period is shortened.

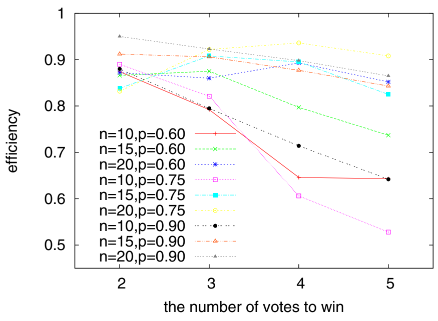
Lastly, we compare the performance of the three voting mechanisms: the m votes to win mechanism, compulsory voting, and random decision making in terms of social surplus. Here, we have to consider the problem of how to know the parameters of p and q. We may be able to estimate these values but are difficult to obtain the accurate values in the real fields. Thus, tuning the value of m for each instance case is not realistic. Based on the results of the first experiment, we choose $m = 3$ and examined the performance of the m votes to win mechanism. The parameter setting of p is different from the first and the second experiment, i.e., $p = \{0.6, 0.7, 0.8, 0.9\}$ is used to make more detailed examination.



(a) q=0.1



(b) q=0.5



(c) q=0.9

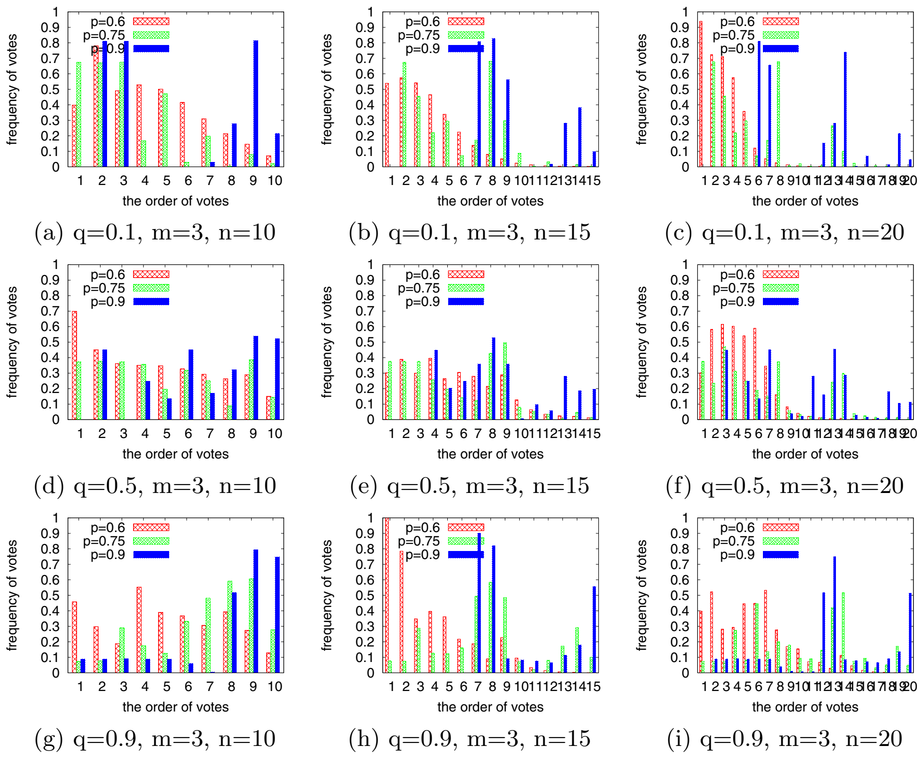
Fig. 1. The effect of changing m to social surplus

In these three experiments, we set $c_H = 0.2$ and $c_L = 0.05$ and create 100000 instances that are consistent with the probability distributions, and then calculate the average values. Social surplus of each instance can be calculated for the each voting system as followings.

5.2 Experimental Results

The Optimal Value of m. Fig. 1 shows how social surplus is affected by changing the number of votes to win, i.e., m, in the m votes to win mechanism. Figures (a),(b),(c) correspond to the cases of $q = 0.1, 0.5, 0.9$, respectively and the nine lines in each figure corresponds to the combinations of $n = \{10, 15, 20\}$ and $p = \{0.6, 0.75, 0.9\}$. Each horizontal axis represents the value of m, while the vertical axis represents the social surplus normalized by the optimal social surplus. Here, the optimal social surplus is the social surplus obtained by collecting all voters' preferences without causing any cost to the voters and choosing the alternative supported by the majority. Thus, this is an ideal value but cannot be realized in the real world. From these figures, we can elucidate the followings.

– When the probability of preferring alternative A, p, is 0.6 or 0.75, the optimal value of m increases as the number of voters increases. If the total number



(a) q=0.1, m=3, n=10 (b) q=0.1, m=3, n=15 (c) q=0.1, m=3, n=20

(d) q=0.5, m=3, n=10 (e) q=0.5, m=3, n=15 (f) q=0.5, m=3, n=20

(g) q=0.9, m=3, n=10 (h) q=0.9, m=3, n=15 (i) q=0.9, m=3, n=20

Fig. 2. The occurrence of vote/abstention

of voters increases, the number of voters preferring B also increases, which increases the risk to accidentally win alternative B which is not preferred in the majority of the community. Such risk can be reduced by setting m to the larger values.

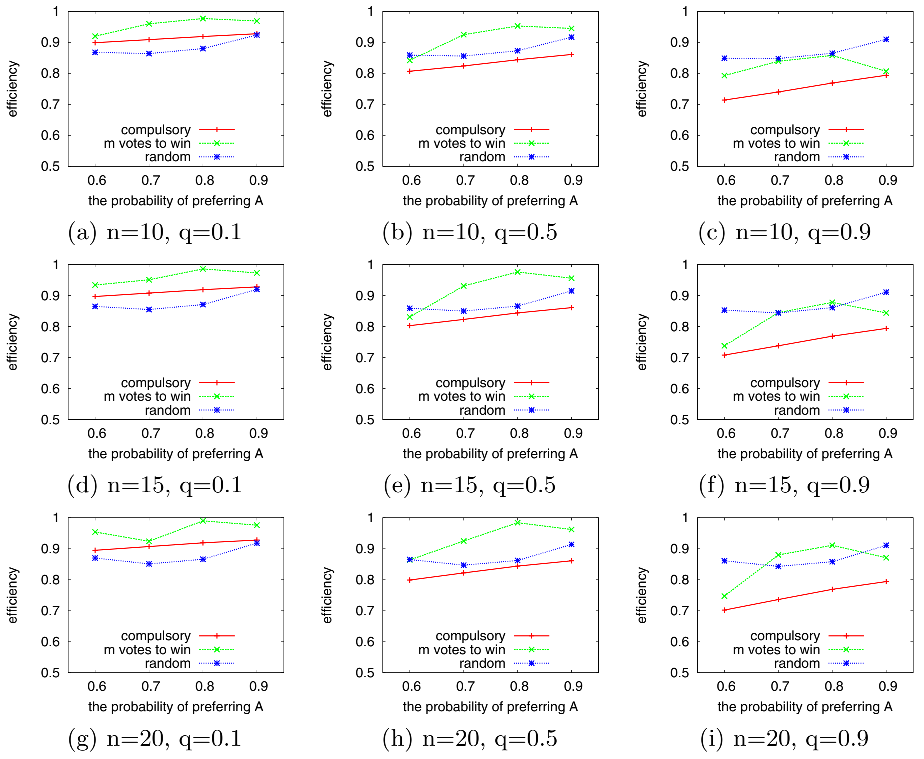
– When the probability of preferring alternative A, p, is 0.9, the optimal value of m does not change so much compared to the case of $p = 0.6, 0.75$ and $m = 2$ is optimal in many cases. This is because a lot of voters prefer A, and specifically in the case of $n = 10$, the probability of B's win is considerably small even if m is set to the small value. Thus, setting m to the small value can reduce the overall cost for voting without increasing the risk of B's win, which contributes to increase social surplus.
– In terms of the probability of having the large voting cost, q, no major difference cannot be found in selecting the optimal value of m.

When Vote/Abstention Occurs. Fig. 2 shows when voters choose vote or abstention. Figures (a) to (i) correspond to the cases of the combinations of $q = \{0.1, 0.5, 0.9\}$ and $n = \{10, 15, 20\}$. Each horizontal axis represents the turns of the first to the last voters, while the vertical axis represents the frequency of votes. It shows how often the voter chooses vote for A or B among the 100000 trials. From these figures, we can elucidate the followings.

- If the probability of preferring alternative A, p, is 0.6, the voters in the early turns vote for their preferred alternative. In this setting the number of voters preferring B is not small compared to that preferring A. Thus, taking the first vote is very valuable. Abstentions by the voters preferring A in the first several turns bring a disadvantageous situation to the group of voters preferring A, although they are majority.
- If the probability of preferring alternative A, p, is 0.9, the voters in the middle turns often vote. Opposed to the above case, the voters preferring A in the early turns do not have to vote for A because there are not so many voters preferring B. However, an interesting point is that the voting behaviors can be found in the early and middle turns as well as in the last several turns in many settings. The excessive postponing of voting results in the failure of obtaining three votes for A. To avoid this failure, voting by the voters in the middle turns is effective.
- The case of $p = 0.75$ is in the intermediate situation between the above two cases.

Comparing the Three Voting Mechanisms. Fig. 3 shows the performance of the three voting mechanisms in terms of social surplus: the m votes to win mechanism, compulsory voting, and random decision making. Figures (a) to (i) correspond to the cases of the combinations of $n = \{10, 15, 20\}$ and $q = \{0.1, 0.5, 0.9\}$. Each horizontal axis represents the probability of preferring alternative A, while the vertical axis represents the social surplus normalized by the optimal social surplus as explained in the first experiment. From these figures, we can elucidate the followings.

- If the probability of having the high voting cost, q, is 0.1 or 0.5, the m votes to win mechanism is superior to compulsory voting or random decision making. In the compulsory voting, the quality of collective decision making can be maximized. However, it is worst in terms of the cost for voting. On the other hand, in the random decision making, the cost for voting can be minimized. However, it has a problem in the quality of the decision making. The m votes to win mechanism can overcome these drawbacks of both of these mechanisms.
- In the m votes to win mechanism, as the probability of preferring A, p, increases, the social surplus first increases, and then decreases. This is a characteristic of this mechanism compared to the compulsory voting. In the m votes to win mechanism, if p is close to 0.9, both alternatives fail to obtain m votes, which results in B (the status quo alternative)'s win. The voters who prefer alternative A and have the high voting cost take the strategy of voting only in the last few turns. This results in reducing the the patterns of the voters' sequence that leads to A's win.



(a) n=10, q=0.1 (b) n=10, q=0.5 (c) n=10, q=0.9
(d) n=15, q=0.1 (e) n=15, q=0.5 (f) n=15, q=0.9
(g) n=20, q=0.1 (h) n=20, q=0.5 (i) n=20, q=0.9

Fig. 3. Comparing the performance of the three voting mechanism

– The failure of A (the alternative supported by the majority)'s win will become worse when $q = 0.9$, i.e., in the case that few voters prefer alternative A and have the low voting cost. This means that the first vote for A has little chance to occur. In addition, the voters preferring A have the strategy that voting for A should be done only in the turns of the $n - 2$ to n turns. Here, the probability that all the voters in the $n - 2$ to n turns prefer alternative A is calculated to be $0.9^3 = 0.729$. This means that the failure of A's win occurs with the probability of around 1/4. This is rather pessimistic estimation, but there is a risk that social surplus is significantly reduced because the almost all voters prefer alternative A but alternative A loses.

6 Discussions

By comparison in terms of social surplus, we verified that the m votes to win mechanism is often superior to the other mechanisms but random decision making sometimes is superior to the m votes to win mechanism. That is, the m votes to win mechanism can balance well the quality of collective decision making with the reduction of the cost for voting but it may happen that random decision making outperforms the m votes to win mechanism. Random decision making, however, is problematic in embedding the voting mechanism in the real fields. A reason is that a selected voter feels more responsibility for the decision making, even if the selection is done at random. Thus, voting will become more costly for the selected voter compared to voting in the other voting mechanisms.

Another reason is that the decision making is dictatorial in random decision making, which is more vulnerable to strategic manipulation. For example, if a voter intends to suborn other voters, suborning only one voter is sufficient in random decision making, whereas suborning m voters is needed in the m votes to win mechanism.

From the above discussion, we can conclude that the m votes to win mechanism is a considerably desirable mechanism for balancing the quality of collective decision making with the reduction of the cost for voting.

7 Concluding Remarks

This paper examined the properties of the m votes to win mechanism as an online voting mechanism that balances the quality of collective decision making with the reduction of the cost for voting. By analyzing the optimal voting strategy, we clarified the followings. First, the optimal value of m increases as the number of voters increases if the probability of preferring an alternative is not large (e.g., around 0.6 to 0.75), while the optimal value of m does not change so much if the probability of preferring an alternative is large (e.g., around 0.9). Second, if the probability of preferring an alternative is not large (e.g., 0.6), the voters in the early turns vote for their preferred alternative, while if the probability of preferring an alternative is large (e.g., 0.9), the voters in the middle turns often vote. Third, if the probability of having the high voting cost is not large (e.g., 0.1 or 0.5), the m votes to win mechanism is superior to compulsory voting or random decision making, while if the probability of having the high voting cost is large (e.g., 0.9), it happens that random decision making is superior to the m votes to win mechanism. These findings are helpful to learn whether the m votes to win mechanism performs well and to determine the optimal value of m in the new application fields.

In this paper, we assume that the number of alternatives is two for making discussions simple. Extending the discussion to the case of more than two alternatives is included in our future work.

Acknowledgments. This research was partially supported by a Grant-in-Aid for Scientific Research (B) (22300052, 2010-2012) from Japan Society for the Promotion of Science (JSPS).

References

1. Airiau, S., Endriss, U.: Iterated Majority Voting. In: Rossi, F., Tsoukias, A. (eds.) ADT 2009. LNCS, vol. 5783, pp. 38–49. Springer, Heidelberg (2009)
2. Battaglini, M.: Sequential voting with abstention. Games and Economic Behavior 51(2), 445–463 (2005)

3. Börgers, T.: Costly voting. American Economic Review 94, 57–66 (2004)
4. Conitzer, V.: Making decisions based on the preferences of multiple agents. Communications of the ACM 53(3), 84–94 (2010)
5. Dekel, E., Piccione, M.: Sequential voting procedures in symmetric binary elections. Journal of Political Economy 108(1), 34–55 (2000)
6. Desmedt, Y., Elkind, E.: Equilibria of plurality voting with abstentions. In: Proceedings of the Eleventh ACM Conference on Electronic Commerce (EC 2010), pp. 347–356 (2010)
7. Elkind, E., Lang, J.: Guest editorial: special issue on computational social choice. In: Autonomous Agents and Multi-Agent Systems, vol. 22(1), pp. 1–3 (2011)
8. Farquharson, R.: Theory of Voting. Yale University Press (1969)
9. Meir, R., Polukarov, M., Rosenschein, J.S., Jennings, N.R.: Convergence to equilibria in plurality voting. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI 2010), pp. 823–828 (2010)
10. Morton, R.B., Williams, K.C.: Information asymmetries and simultaneous versus sequential voting. The American Political Science Review 93(1), 51–67 (1999)
11. Poundstone, W.: Gaming the Vote: Why Elections Aren't Fair. Hill and Wang (2008)
12. Sloth, B.: The theory of voting and equilibria in noncooperative games. Games and Economic Behavior 5(1), 152–169 (1993)
13. Xia, L., Conitzer, V.: Stackelberg voting games: Computational aspects and paradoxes. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010 (2010)

An Agent-Based Model for Integrated Contagion and Regulation of Negative Mood

Azizi Ab Aziz, Jan Treur, and C. Natalie van der Wal

VU University Amsterdam, Agent Systems Research Group
De Boelelaan 1081, 1081HV Amsterdam, The Netherlands
{azizi.binabaziz,j.treur,c.n.vander.wal}@vu.nl
http://www.few.vu.nl/~{mraaziz,treur,cwl210}

Abstract. Through social interaction, the mood of a person can affect the mood of others. The speed and intensity of such mood contagion can differ, depending on the persons and the type and intensity of their interactions. Especially in close relationships the negative mood of a depressed person can have a serious impact on the moods of the ones close to him or her. For short time durations, contagion may be the main factor determining the mood of a person; however, for longer time durations individuals also apply regulation mechanisms to compensate for too strong deviations of their mood. Computational contagion models usually do not take into account such regulation. This paper introduces an agent-based model that simulates the spread of negative mood amongst a group of agents in a social network, but at the same time integrates elements from Gross' emotion regulation theory, as the individuals' efforts to avoid a negative mood. Simulation experiments under different group settings pointed out that the model is able to produce realistic results, that explain negative mood contagion and emotion regulation behaviours posed in the literature.

Keywords: emotion contagion and regulation, agent-based model.

1 Introduction

There is a wide consensus in sociological literature that human mood spreads through social networks [9, 11]. This social phenomenon is known as contagion. Especially negative moods are strongly influenced by social contacts (e.g., family, friends, colleagues, and neighbours), for example, when the social interaction involves conflict issues or stressful events [4, 15]. Agent-based computational models for contagion of different types of mental states can be found, for example, in [1, 10]. However, in addition to contagion at the social level, also emotion regulation within individuals plays an important role [3]. Emotion regulation is a process through which individuals balance their emotions by exerting forms of control on how they feel [8]. For instance, by avoiding situations or persons who trigger negative emotions, or suppressing anger when receiving bad comments from interviewers. By such emotion regulation mechanisms, persons have the ability to suppress negative influences from interaction with others and maintain a form of emotional homeostasis [7, 8]. For example, if a partner of a depressed person has regulation mechanisms that are strong enough, he or she does not need to become depressed, but if the mechanisms are less strong, there is a serious risk that the partner also becomes depressed.

In recent years researchers have focused on understanding the mechanisms of emotion regulation, and social contagion separately [2, 13, 15]. However, little information is available to explain how these processes work in an integrated manner by means of computational models. In this paper, an agent-based model is proposed that formalizes and simulates the integrated contagion and regulation of negative mood. In order to exemplify the proposed model, simulation experiments have been performed with a variety of scenarios that include varying personal characteristics and group or network compositions. Attributes were configured, to represent the personality and social characteristics of different individuals. Simulation traces were generated, to show behaviour of these individuals over time, under multiple conditions.

2 Mood Contagion and Regulation

In this section, important ideas and concepts in negative mood contagion and emotion regulation research are addressed. These ideas form the basis of the current computational model that will be formally described in the next section. As described in [5], the degree of mood contagion in groups is influenced by the valence and energy of the mood. One of the fundamental components in mood contagion is the *contagion strength* between individuals within a group [6]. It involves the type of interaction between individuals (*channel strength* from sender to receiver) and personality characteristics of the sender (*expressiveness*) and receiver (*openness*). For negative mood contagion, channel strength can be defined as the intensity of the social interaction, either via *physical contact* (i.e, face-to-face), or *virtual interaction* (i.e, text message, social networking) [16]. Neighbourhood and personality characteristics, affect the openness for mood contagion of a person [11, 12]. For example, a neurotic individual tends to aggravate negative perception towards incoming mood [14]. In addition to this, a bad neighbourhood (physical or social) also creates a negative influence towards individual's perception in social interaction [12]. Expressiveness is related to the ability of an individual to induce contagion, where an extravert individual can induce a stronger contagion of a negative mood than an introvert individual, because an extravert person expresses his or her internal feelings stronger than an introvert person [1].

Besides mood contagion, emotion regulation plays a role in the experience and transfer of moods. It is important to understand the emotion regulation process, by knowing which different strategies individuals use to exert control over their moods [2]. To serve this purpose, Gross' emotion regulation theory provides a number of strategies to affect individuals' level of emotion [7]. This theory differentiates these strategies into *antecedent-focused strategies* and *response-focused strategies*. The former type of strategies refer to the process preparing for response tendencies before they are (fully) activated, and the latter deal with the actual activation or suppression of the expression of emotional responses [13]. Antecedent-focused strategies can involve the external situation of the person (e.g., avoiding certain places or persons), or the internal processes (e.g., redirecting attention or cognitive interpretation). Gross [7, 8] mentions four examples of antecedent-focused strategies: *situation selection, situation modification, attentional deployment,* and *cognitive change*. In a response-focused strategy, response modulation is used (e.g., suppressing expressing of negative emotions, or amplifying expression of positive emotions).

Situation selection involves selecting a situation that supports the individual's emotional well-being. This may involve physical and/or social aspects. For example, if a person has a bad response on low light intensity, a form of regulation is to increase this intensity. Especially relevant to the integration with social contagion processes, is the regulation of the social situation. For example, if a person feels bad in a certain social environment, he/she can decrease his/her openness for and intensity of social interaction. Situation modification is similar to selection, but addresses only some aspects of a situation. Attentional deployment includes redirection of attention, for example, on more neutral or positive elements [7]. Cognitive change refers to change in how an individual interprets the situation. Response modulation refers to physical or behavioural actions that decrease the expression of negative emotions [8].

3 The Agent-Based Model

The agent-based model introduced in this section combines knowledge on mechanisms for mood contagion and emotion regulation, as briefly introduced above. In this computational model these mechanisms are encapsulated, allowing the simulation of how fragile individuals in their social environment are, towards negative mood contagion. The model describes a process to maintain homeostasis for mood. Through social interaction, there is a habitual tendency of an individual to perceive the negative mood of others and to regulate his or her own moods. Both processes are governed by individual's socio-culture, default (norm) personality, and his or her negative mood. In the formalized model, all nodes are designed to have values ranging from *0* (low) to *1* (high). The interaction will determine the new value for each node, either by a series of accumulations or an instantaneous interaction. To represent these relationships in agent terms, each variable will be coupled with an agent's name (*A* or *B*) and a time variable *t*. The description of these formalizations is described below. For a global overview, see Fig. 1.

3.1 Norm Values

Norm values indicate which level each individual is inclined to approximate during the process: an individual tries too keep itself within safe boundaries around these values. These norm values can be seen as a basis for 'default behavioural patterns'; e.g., the openness a person tends to have, based on neighbourhood characteristics and level of neuroticism, or a default level of expressiveness, based on personality characteristics. These norm values are also the natural initial settings of the persons in scenarios. The norm value C_{normAB} at some point in time *t* for the channel of agent *A* to agent *B*, can be related to the amount of physical (PI_{AB}) and virtual (VI_{AB}) interactions that take place**,** where *0* means no physical or virtual interaction with others, and *1* means a lot of physical interaction [12]. This interaction is regulated by the proportional parameter α. If $\alpha = 0.5$, both types of interactions have the same effect, otherwise, one of these types of interactions has more effect on the channel norm value.

$$C_{normAB}(t) = \alpha.\ PI_{AB}(t) + (1\text{-}\alpha).VI_{AB}(t) \qquad (1)$$

Note that the interaction can be bidirectional, so that $C_{normAB}(t) = C_{normBA}(t)$, but this is not assumed to be always the case; the model also covers asymmetric cases, for example, where frequently text messages are sent from *A* to *B* but not conversely, or *B* follows *A* on Twitter but not the other way around.

Next, the openness norm value O_{normA} of agent *A*, first relates to the (bad) neighbourhood circumstances of *A* expressed in a concept NH_A, where a value of *1* means a very 'bad' neighbourhood, which makes a person vulnerable to negative mood, and the value *0* means the neighbourhood does not make a person more susceptible to negative mood of others. NH_A is modelled as the product of the social (SNH_A) and physical (PNH_A) neighbourhood and of the person. If $PNH_A = 1$, then the physical neighbourhood is very 'bad', and it will have a negative effect on the person's susceptibility. By multiplication of the social and physical neighbourhood in (2), a more 'positive' social neighbourhood (with a low value), will make the impact of the 'bad' physical neighbourhood smaller [12].

$$NH_A(t) = SNH_A(t).PNH_A(t) \qquad (2)$$

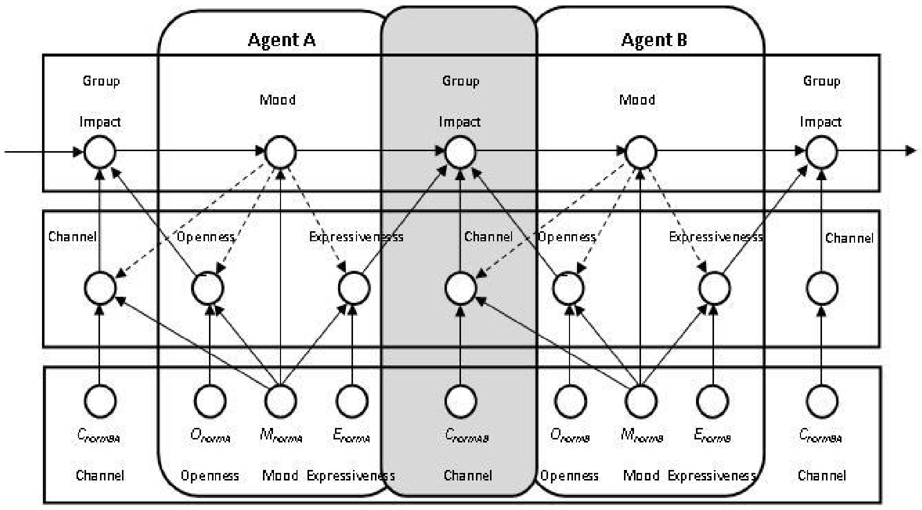


Fig. 1. Overview of the Agent-Based Model Integrating Mood Contagion and Regulation

The openness norm value O_{normA} of agent *A,* combines the concepts of a bad neighbourhood NH_A, with the concepts friends ratio NF_A and neuroticism N_A. In [12] it is described that the more friends you have, the less prone you are to negative mood contagion. The quantity NF_A is defined as a number between *0* and *1* (a 'friend ratio'): the number of friends is divided by a fixed number (serving as an upper bound) to normalise it. For example, if the upper bound taken is *10* (as in the simulations discussed in Section 4) then one friend will give $NF_A = 0.1$, whereas *7* friends will give $NF_A = 0.7$. Parameter φ regulates the equation; so that it can be modelled which concept can have more effect on the openness norm value than the other. In addition to this, [11] put forward that the more neurotic you are, the more susceptible you are to negative mood of others. Therefore, the level of neuroticism N_A can amplify or reduce the positive effects of having such as a high number of friends and/or a not bad neighbourhood.

$$O_{normA}(t) = [\ \varphi.(1-NF_A(t))+ (1- \varphi).NH_A(t)].N_A(t) \tag{3}$$

Finally, in the current model, the expressiveness norm value E_{normA} of agent A is initialised by a number between *0* and *1*, not a formula. The number represents the level of expressiveness a person tends to approximate in daily life, where *0* means low expressiveness and *1*, high expressiveness.

3.2 The Dynamics of Mood Contagion and Emotional Regulation

In this section the dynamical model for mood contagion and regulation is introduced. A summary of the parameters and state variables of the model is shown in Table 1.

For the mechanisms behind mood contagion, elements from the model presented in [1] have been adopted. The main building block of mood contagion in this model is the contagion strength CS_{AB} from agent A to agent B, where it represents the type and intensity of the contact between agent A and agent B. The higher the value of CS_{AB}, the more contagion will take place.

$$CS_{AB}(t)= E_A\,(t).C_{AB}\,(t).O_B(t) \qquad \text{where } A\neq B \tag{4}$$

Here, E_A is the personal characteristic expressiveness (the degree in which a person can express his/her mood), C_{AB} the channel strength (intensity of contact, depending on the social relation) from A to B, and O_B the openness (the degree of susceptibility) of the receiver B. Using this equation, the group contagion strength is computed. The group contagion strength $CS_A*(t)$ towards A is the overall strength by which the negative mood of all other group members is received by A:

$$CS_A*(t) = \textstyle\sum_{B\neq A} CS_{BA}(t) \tag{5}$$

Note that for the sake of simplicity here a linear (sum) combination is used. Alternatively, also a logarithmic or logistic combination function might be used. Given the mood levels $M_B(t)$ of the agents $B\neq A$ at time t, the weighted group impact $M_A*(t)$ of all other agents in the group towards agent A is modelled as:

$$M_A*(t)= \textstyle\sum_{B\neq A} CS_{BA}(t).\ M_B(t)\,/\,CS_A*(t) \tag{6}$$

More details of this model for contagion can be found in [1]. Next the dynamics of the mechanisms for integrated emotion regulation and negative mood contagion are modelled in (7), (8), (9), and (10). The general pattern underlying these dynamical relationships is

$$Y_A(t+\Delta t) = Y_A(t) + \tau.\ \textit{<change\_expression>}.\ \Delta t$$

Here the change of Y is specified for a time interval between t and $t+\Delta t$; the τ are personal flexibility parameters that represent the speed of the cognitive adjustment processes. Within <*change_expression*> two cases are considered: upward (positive) change <*upward_change*>, and downward (negative) change <*downward_change*>.

$$\textit{<change\_expression>} = (1-Y_A(t)).\ \textit{<upward\_change>} + Y_A(t).\ \textit{<downward\_change>}$$

The upward and downward change expressions are determined using the operator Pos(x) defined as Pos(x) = x when $x \geq 0$, else 0.

$$<upward\ change> = \text{Pos}(<basic\ change>)$$
$$<downward\ change> = -\text{Pos}(-<basic\ change>)$$

Within the basic change expression for (7), (8), and (9), two parts are considered. The first part incorporates the emotion regulation, and the second part the maintenance of homeostasis.

$$<basic\_change> = <regulation\_change> + <maintenance\_change>$$

The latter change expressions were taken linear in the deviation:

$$<regulation\_change> = \zeta \,.\, [M_{normA} - M_A(t)]$$
$$<maintenance\_change> = \upsilon \,.\, [Y_{normA} - Y_A(t)]$$

Here ζ and υ are more specific flexibility parameters, for regulation and maintenance. Next it is shown how this general pattern was applied for channel strength (7), openness (8), and expressiveness (9). Firstly, the concepts of emotion regulation are represented in the dynamic adjustment of the strength of the channel from agent A to B. In (7) this occurs by comparing the current mood level to the mood norm value and comparing the current channel level with the channel norm value. These possible deviations influence the adjustment in the strength of the channel that the agent makes. This covers situations in which a person is infected by negative mood from other persons and directs his/her attention away, or physically moves to another place.

$$\begin{aligned} C_{BA}(t+\Delta t) = C_{BA}(t) + &\\ \tau_{CA}.[\ (1 - C_{BA}(t)).\ \text{Pos}(\zeta_{CA}\,[M_{normA} - M_A(t)] + \upsilon_{CA.}\ [C_{normBA} - C_{BA}(t)]) - &\\ C_{BA}(t).\ \text{Pos}(-\ \zeta_{CA}\,[M_{normA} - M_A(t)] - \upsilon_{CA.}\ [C_{normBA} - C_{BA}(t)])]\ .\Delta t & \end{aligned} \tag{7}$$

The dynamic relation for the openness O_A of agent A models another antecedent-focused emotion regulation mechanism [7].

$$\begin{aligned} O_A(t+\Delta t) = O_A(t) + &\\ \tau_{OA}.[(1 - O_A(t)).\text{Pos}(\zeta_{OA}\ [M_{normA} - M_A(t)] + \upsilon_{OA}.\ [O_{normA} - O_A(t)]) - &\\ O_A(t).\ \text{Pos}(-\zeta_{OA}\ .[M_{normA} - M_A(t)] - \upsilon_{OA}\ .[O_{normA} - O_A(t)])].\Delta t & \end{aligned} \tag{8}$$

The expressiveness E_A of agent A involves a response-based emotion regulation mechanism [7, 8]. In (9), expressiveness is adjusted towards the norm value, but also adjusted to decrease expression of negative mood.

$$\begin{aligned} E_A(t+\Delta t) = E_A(t) + &\\ \tau_{EA}.[(1 - E_A(t)).\ \text{Pos}(\zeta_{EA.}\ [M_{normA} - M_A(t)] + \upsilon_{EA}.\ [E_{normA} - E_A(t)]) - &\\ E_A(t).\ \text{Pos}(-\zeta_{EA.}\ [M_{normA} - M_A(t)] - \upsilon_{EA.}\ [E_{normA} - E_A(t)])].\Delta t & \end{aligned} \tag{9}$$

Finally in (10), an internal antecedent-focused emotion regulation mechanism called re-appraisal [8] is modelled. Here within the generic pattern discussed above the expression *<basic_change>* is instantiated as follows.

$$<basic\_change> = <contagion\_change> + <reappraisal\_change>$$

Table 1. Parameters and state variables of the model

Concepts	***Formalization***
negative mood of agent *A*	M_A
norm value for the negative mood of agent *A*	M_{normA}
weighted group impact	M_A*
expressiveness of agent *A* (sending side)	E_A
norm value for expressiveness of agent *A*	E_{normA}
channel strength from agent *A* to agent *B*	C_{AB}
norm value for channel from agent *A* to agent *B*	C_{normAB}
contagion strength from agent *A* to agent *B*	CS_{AB}
overall group contagion strength towards agent *A*	CS_A*
openness of agent *A* (receiving side)	O_A
norm value for openness of agent *A*	O_{normA}
physical interaction from *A* to *B* (face-to-face)	PI_{AB}
virtual interaction from *A* to *B*	VI_{AB}
number of friends 'friend ratio' of agent *A*	NF_A
bad neighbourhood of agent *A*	NH_A
level of neuroticism of agent *A*	N_A
bad social neighbourhood of *A*	SNH_A
bad physical neighbourhood of *A*	PNH_A
proportional parameter for C_{normA}	α
proportional parameter for O_{normA}	φ
flexibility parameter for *Y (*regulation_change);	ζ_{YA}
flexibility parameter for *Y* (maintenance_change);	υ_{YA}
flexibility parameter of agent *A* for the re-appraisal emotion regulation in (10)	λ_A
bias of agent *A*	β_A
flexibility parameter of *Y* (in a change expression); see (7), (8), (9), (10)	τ_{YA}

where

$$<reappraisal\_change> = \lambda_A.\ [M_{normA}-M_A(t)]$$
$$<contagion\_change> = CS_A*(t).\ [\ \beta_A\ .(1-(1-M_A(t))\ (1-M_A*(t))) + (1-\beta_A)\ .M_A(t).\ M_A*(t) - M_A(t)]$$

The latter expression was adopted from [1]. This provides the following mood dynamics relation:

$$\begin{aligned} M_A(t+\Delta t) = {} & M_A(t) + \\ & \tau_{MA}\,.[(1-M_A(t)).\ \mathrm{Pos}(CS_A*(t)\ [\ \beta_A.(1-(1-M_A(t)).(1-\ M_A*(t))) + \\ & (1-\beta_A).\ M_A(t).M_A*(t)\ - M_A(t)] -\ \lambda_A.[M_{normA}-M_A(t)])] - \\ & M_A(t).\ \mathrm{Pos}(-\ CS_A*(t).[\ \beta_A\ .(1-(1-M_A(t)).(1-M_A*(t))) + \\ & (1-\beta_A)M_A(t)M_A*(t) -\ M_A(t)] + \lambda_A[M_{normA}-M_A(t)])\ ]\ .\Delta t \end{aligned} \qquad (10)$$

4 Simulation Results

The model was implemented in different numerical software environments, one of which was Matlab. Multiple compositions of groups and networks were simulated, but for the sake of brevity, in this section the simulation scenario with only three agents are considered: namely; (**A**) a 'depressed' person with a very negative mood, (**B**) his/her life partner, and (**C**) his/her friend. Through this scenario, it is explored how the negative mood of a person can spread through his/her social network and can be controlled by emotion regulation mechanisms in the receiving persons. For all scenarios, the current simulations used the following parameters settings; t_{max}=*1000*, Δt = *0.1*, flexibility parameters τ_{YA} = *0.5* for *openness*, *channel strength*, *expressiveness,* and *0.1* for negative mood. These settings were obtained from previous systematic experiments to determine to the most suitable parameters values in the model. It means, several experiments were conducted to determine how a reasonable time scale and grain size of the simulation could be obtained. In this way, an appropriate setting for the parameters for speed of change, and of the time step Δt was chosen. The other parameters in principle can be chosen in any form as they reflect characteristics of the situation modelled. Table 2 summarizes the (initial) settings for the different agents.

Table 2. Individual Profiles for Each Agent

	Scenario #1			**Scenario # 2**			**Scenario # 3**			**Scenario # 4**		
	A	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**
Initial M	0.9	0.4	0.2	0.9	0.4	0.2	0.9	0.4	0.2	0.9	0.4	0.2
M_{norm}	0	0	0	0	0	0	0	0	0	0	0	0
O_{norm}	1	1	1	1	1	1	1	1	1	1	1	1
E_{norm}	1	1	1	1	1	1	1	1	1	1	1	1
C_{norm}	1	1	1	1	1	1	1	1	1	1	1	1
λ	0	0	0	0	1	1	0	0	0	0.5	0.5	0
B	1	0.5	0	1	1	1	0.5	0.5	0.5	1	0.5	0
υ (for all opennes *O*, channels *C* and expressiveness *E*)	0	0	0	0	0.1	0.1	0	0	0	0.1	0.1	0
ζ (for all opennes *O*, channels *C* and expressiveness *E*)	0	0	0	0.1	0.1	0.1	0	0	0	0.1	0.1	0.1

Scenario # 1

The results of this scenario are shown in Fig. 2. During the simulation, the agent *A* stays on his negative initial mood. He is not capable of regulating his mood (since he is too depressed; his emotion regulation mechanisms do not work) and transmits his negative mood to his partner and friend.

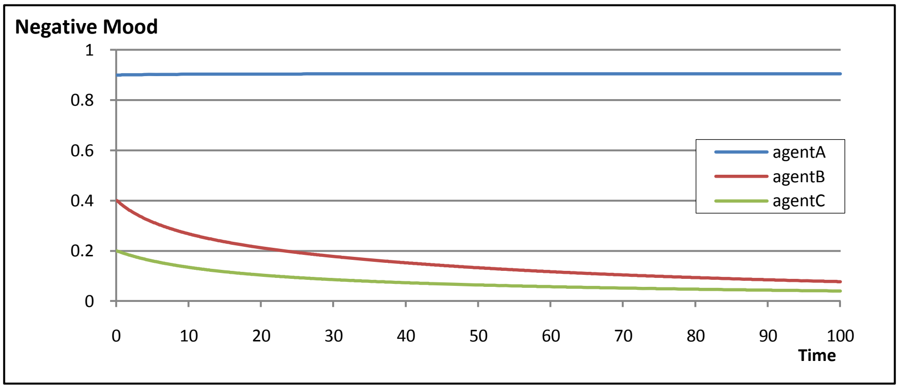


Fig. 2. Simulation results scenario 1

Because the partner and friend do have intact emotion regulation mechanisms, they are not infected to the level of the 'depressed' person's negative mood. The stronger their emotion regulation mechanisms are, the less the 'depressed' person can infect them with his negative mood. Furthermore, agent *B* has a higher negative mood bias ($\beta = 0.5$), than agent A ($\beta = 0$), therefore, agent *B*'s negative mood decreases less fast than for agent *C*.

Scenario # 2

Here all agents have a maximum negative mood bias ($\beta = 1$), by which they all approximate the highest initial negative mood (in this case that of the 'depressed' person, agent *A*). If no agent would have working emotion regulation capacities, all agents would increase to a negative mood level of 0.9. Now agent *B* and *C* have small emotion regulation capacities and therefore, they do not fully increase to the initial mood level of agent *A*. Fig. 3 depicts the results for this scenario.

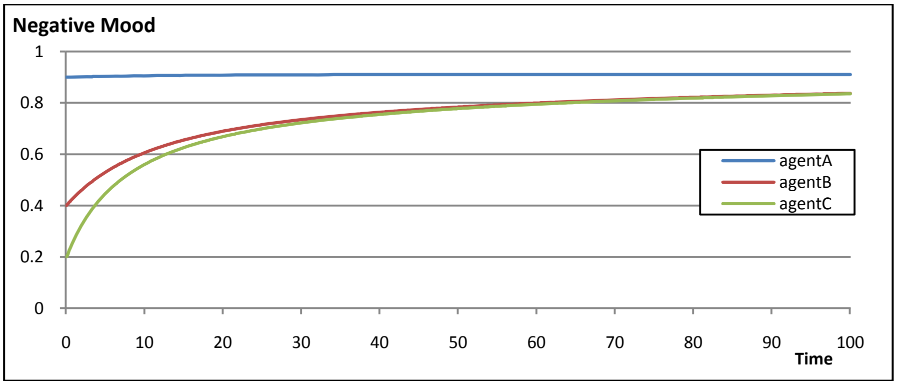


Fig. 3. Simulation results scenario 2

Scenario #3
This scenario represents the baseline where no emotion regulation mechanisms exist in the three agents. In this case, all agents have a negative mood bias (β = 0.5), which has the effect that all the agent's mood levels approximate the average initial mood setting (see Fig. 4).

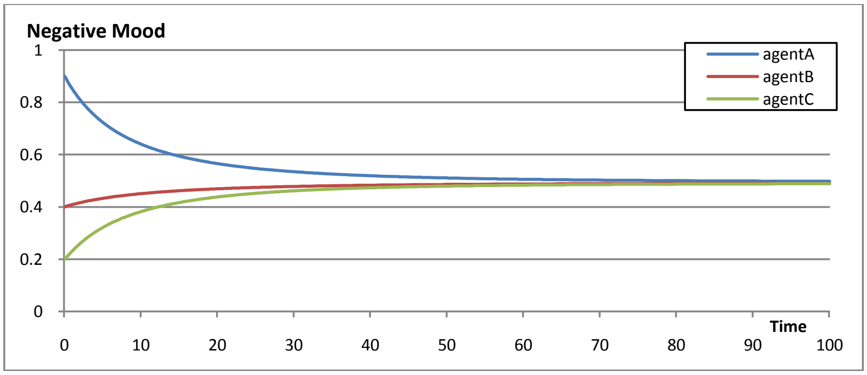


Fig. 4. Simulation results for scenario 3

The emotion regulation mechanisms in agent *A* and *B*, let the negative mood levels of agent *A* and *B* increase to a lesser extent. As can be seen from Fig. 4, this scenario shows how the negative bias β and emotion regulation mechanism have opposite effects.

Scenario #4
In this scenario, agent *C* does not have working emotion regulation mechanisms, but agent *A* and *B* do. In Fig. 5 it is shown that the emotion regulation mechanisms in agent *A* and *B*, let the negative mood levels of agent *A* and *B* decrease to a lesser extent, than that of Agent *C*, compared with scenario 3 (Fig. 5), where no agent had emotion regulation mechanisms that work. This shows how the negative bias β and emotion regulation mechanism have opposite effects: A high negative bias ($\beta>0.5$) can increase the negative mood of the agent, intact emotion regulation mechanisms (λ_A or υ of openness *O*, channel strength *C* or expressiveness *E* nonzero) will reduce this effect.

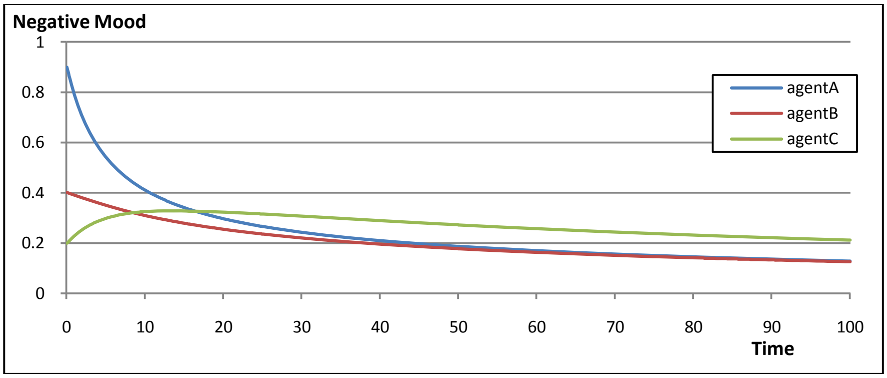


Fig. 5. Simulation results for scenario 4

5 Mathematical Analysis

In this section, an analysis is made of possible equilibria of the model. These are values for the variables of the model for which no change occurs. Taking as a point of departure the generic pattern,

$$Y_A(t+\Delta t) = Y_A(t) + \tau.\langle change\_expression\rangle.\Delta t$$

and assuming τ nonzero, this is equivalent to $\langle change\ expression\rangle = 0$ for all variables Y_A. Moreover, as

$$\langle change\_expression\rangle = (1-Y_A(t)).\mathrm{Pos}(\langle basic\_change\rangle) - Y_A(t).\mathrm{Pos}(-\langle basic\_change\rangle)$$

the criterion for an equilibrium is:

$$(1-Y_A(t)).\mathrm{Pos}(\langle basic\_change\rangle) - Y_A(t).\mathrm{Pos}(-\langle basic\_change\rangle) = 0$$

Note that always $Pos(x) = 0$ or $Pos(-x) = 0$; this implies the following lemma:

Lemma 1
For any nonzero η_1 and η_2 it holds

$$\eta_1 Pos(x) + \eta_2 Pos(-x) = 0 \quad \text{iff} \quad x = 0.$$

By Lemma 1 it follows that for cases that $Y_A(t)$ is nonzero and < 1, the equilibrium criterion is

$$\langle basic\_change\rangle = 0$$

If this is applied to dynamic relations (7) to (10) the following four equilibrium equations are obtained:

$$\zeta_{CA}.[M_{normA}-M_A] + \upsilon_{CA}.[C_{normBA}-C_{BA}] = 0 \quad (11)$$
$$\zeta_{OA}.[M_{normA}-M_A] + \upsilon_{OA}.[O_{normA}-O_A] = 0 \quad (12)$$
$$\zeta_{EA}.[M_{normA}-M_A] + \upsilon_{EA}.[E_{normA}-E_A] = 0 \quad (13)$$
$$\beta_A.(1-(1-M_A).(1-M_A^*)) + (1-\beta_A).M_A.M_A^* - M_A + \lambda_A.[M_{normA}-M_A] = 0 \quad (14)$$

The first three equations are equivalent to (here the following short notation is used: $devY = Y_{norm} - Y$ (deviation of Y from norm value)):

$$devC_{BA} = -(\zeta_{CA}/\upsilon_{CA}).\ devM_A \quad \text{from (11)}$$
$$devO_A = -(\zeta_{OA}/\upsilon_{OA}).\ devM_A \quad \text{from (12)}$$
$$devE_A = -(\zeta_{EA}/\upsilon_{EA}).\ devM_A \quad \text{from (13)}$$

In particular, it follows that either none of C_{BA}, O_A, E_A, M_A deviates from its norm, or all of them deviate from their norm (in a proportional manner). For the special case $M_{normA} = 0$ used in the experiments, it holds $devM_A = -M_A$, and therefore the equations are:

$$devC_{BA} = (\zeta_{CA}/\upsilon_{CA}).\ M_A$$
$$devO_A = (\zeta_{OA}/\upsilon_{OA}).\ M_A$$
$$devE_A = (\zeta_{EA}/\upsilon_{EA}).M_A$$

Having exploited the first three equations, what remains is the fourth one. To analyse this one, the following lemma is useful.

Lemma 2
For any A it holds:
$M_A{}^* = 0$ iff $M_B = 0$ for all $B \neq A$ with nonzero CS_{BA}
$M_A{}^* = 1$ iff $M_B = 1$ for all $B \neq A$ with nonzero CS_{BA}

As the fourth equation is rather complex in its general form, it is analysed for a number of special cases. In particular, assume $\lambda_A = 0$ (no re-appraisal). Then the fourth equation can be rewritten as follows:

$$\beta_A .M_A{}^* - M_A .[1 - \beta_A - M_A{}^* + 2\beta_A .M_A{}^*] = 0$$
$$M_A = \beta_A .M_A{}^* / [(1 - \beta_A).(1 - M_A{}^*) + \beta_A .M_A{}^*],$$
$$\text{if } (1 - \beta_A).(1 - M_A{}^*) + \beta_A .M_A{}^* \neq 0$$

For this case, equilibria can occur with values different from 0 and 1, which may depend on the initial values. In addition, three special cases for β_A are considered: $\beta_A = 0, \beta_A = 0.5, \beta_A = 1.$

***Case I.* $\lambda_A = 0,\ \beta_A = 0$**
In this case the fourth equation can be rewritten into

$$M_A.M_A{}^* - M_A = 0,$$

which is equivalent to

$$M_A = 0 \text{ or } M_A{}^* = 1$$

By Lemma 2 this is equivalent to

$$M_A = 0 \text{ or } M_B = 1 \text{ for all } B \neq A \text{ with nonzero } CS_{BA}$$

This implies that for this case no equilibria exist with values different from 0 and 1.

***Case II.* $\lambda_A = 0,\ \beta_A = 0.5$**
In this case the fourth equation can be rewritten into

$$0.5.(M_A + M_A{}^* - M_A.M_A{}^*) + 0.5.M_A.M_A{}^* - M_A = 0,$$

which is equivalent to $M_A = M_A{}^*$
For this case equilibria can occur with values different from 0 and 1, which may depend on the initial values.

Case III. $\lambda_A = 0,\ \beta_A = 1$
In this case the fourth equation can be rewritten into

$$M_A - M_A . M_A^* = 0$$

which is equivalent to

$$M_A = 1 \text{ or } M_A^* = 0$$

By Lemma 2 this is equivalent to

$$M_A = 1 \text{ or } M_B = 0 \text{ for all } B \neq A \text{ with nonzero } CS_{BA}$$

As for Case I, this implies that for this case no equilibria exist with values different from *0* and *1*.

6 Discussion

Research into the mechanisms of emotion regulation and social contagion has mainly been conducted separately [2, 13, 15]. In the current work, it was investigated how these processes work in an integrated manner, by means of a computational model. An agent-based model is proposed, that formalizes and simulates the integrated contagion and regulation of negative mood. The current model was inspired by a number of theories, namely emotion contagion and Gross' emotion regulation theory [1, 2, 5, 7]. For short time durations, contagion may be the main factor determining the mood of a person; however, for longer time durations individuals also apply regulation mechanisms to compensate for too strong deviations of their mood. Computational contagion models usually do not take into account such regulation.
Simulation results show interesting patterns that illustrate the combined effect of negative mood contagion and emotion regulation. Together, these elements can be used to understand how a person is capable to maintain his or her mood, while maintaining social interactions with another person. For this model, a mathematical analysis shows how such equilibria are indeed possible for the model. Note that for the sake of simplicity mood affecting external events during a simulated process have been left out of consideration. However, it is not difficult to include them too.

In follow up research, more attention will be focused to implement this model in a large scale social networks and to see important emergent behaviours that possibly exist when more agents are involved. Furthermore, it would be interesting to study a situation at a societal level where agents can also change their behaviours (such as relapse, recovery, and susceptibility), by introducing additional attributes and parameters into the model. In addition, this model can be used as a foundation to design software agents that capable to understand and aware about humans and their interactions. By using this model, software agents will use this as knowledge to provide appropriate actions to support humans pertinent to their predicted states (e.g. the level of negative mood). Future work of this model can be extended to incorporate multiple types of emotion and their interaction. Moreover, this model has a potential to be useful to provide a foundation to understand how negative mood can be propagated via social media (e.g., Facebook, MySpace, Twitter).

References

1. Bosse, T., Duell, R., Memon, Z.A., Treur, J., van der Wal, C.N.: Multi-Agent Model for Emotion Contagion Spirals Integrated within a Supporting Ambient Agent Model. In: Yang, J.-J., Yokoo, M., Ito, T., Jin, Z., Scerri, P. (eds.) PRIMA 2009. LNCS, vol. 5925, pp. 48–67. Springer, Heidelberg (2009)
2. Bosse, T., Pontier, M.A., Treur, J.: A Computational Model based on Gross' Emotion Regulation Theory. Cognitive Systems Research Journal 11, 211–230 (2010)
3. Cote, S.: A Social Interaction Model of the Effects of Emotion Regulation on Work Strain. Academy of Management Review 30, 509–530 (2005)
4. Fowler, J.H., Christakis, N.A.: Dynamic Spread of Happiness in a Large Social Network: Longitudinal Analysis Over 20 Years in the Framingham Heart Study. British Medical Journal 338, 768 (2008)
5. Frederickson, B.L.: The role of positive emotions in positive psychology: The broaden-and-build theory of positive emotions. American Psychologist 56, 218–226 (2001)
6. Fredrickson, B.L., Branigan, C.: Positive Emotions Broaden the Scope of Attention and Thought-Action Repertoires. Cognition and Emotion 19, 313–332 (2005)
7. Gross, J.J.: Antecedent- and response-focused emotion regulation: divergent consequences for experience, expression, and physiology. J. of Pers. and Soc. Psych. 74, 224–237 (1998)
8. Gross, J.J.: Emotion Regulation: Affective, Cognitive, and Social Consequences. Pyschophysiology 39, 281–292 (2002)
9. Hill, A.L., Rand, D.G., Nowak, M.A.: Emotions as Infectious Diseases in a Large Social Network: The SISa Model. Proc. Royal British Society 277, 3827–3835 (2010)
10. Hoogendoorn, M., Treur, J., van der Wal, C.N., van Wissen, A.: An Agent-Based Model for the Interplay of Information and Emotion in Social Diffusion. In: Proc. of the International Conference on Intelligent Agent Technology, IAT 2010, pp. 439–444 (2010)
11. Joiner, T.E., Katz, J.: Contagion of Depressive Symptoms and Mood: Meta-analytic Review and Explanations From Cognitive, Behavioral, and Interpersonal Viewpoints. Clinical Psychology: Science and Practice 6, 149–164 (1999)
12. Kim, D.: Blues from the Neighbourhood? Neighbourhood Characteristics and Depression. Epidemiology Rev. 30, 101–117 (2008)
13. Lopes, P., Salovey, P., Beers, M., Cote, S.: Emotion Regulation Abilities and the Quality of Social Interaction. Emotion 5, 113–118 (2005)
14. Paukert, A.L., Pettit, J.W., Amacker, A.: The Role of Interdependence and Perceived Similarity in Depressed Affect Contagion. Behaviour Therapy 39, 277–285 (2008)
15. Rosenquist, J.N., Fowler, J.H., Christakis, N.A.: Social Network Determinants of Depression. Mol. Psychiatry 16, 273–281 (2010)
16. Ueno, K.: The Effects of Friendship Networks on Adolescent Depressive Symptoms. Social Science Research 34, 484–510 (2005)

A Framework for Agent-Based Modeling of Intelligent Goods

Åse Jevinger[1], Paul Davidsson[1,2], and Jan A. Persson[1,2]

[1] Blekinge Institute of Technology, School of Computing, P.O. Box 214, SE-374 24 Karlshamn, Sweden
{Ase.Jevinger,Paul.Davidsson,Jan.Persson}@bth.se
[2] Malmö University, School of Technology, SE-205 06 Malmö, Sweden

Abstract. The purpose of this paper is to present a framework for intelligent goods and illustrate how it can be applied when modeling intelligent goods as agents. It includes a specification of different levels of capability connected to the goods, which is based on a number of capability dimensions. Additionally, three specific intelligent goods services related to transport are presented. We show how these services can be modeled as agents and how they relate to the intelligent goods framework. A discussion of different physical locations of service information and processing is also included.

Keywords: Intelligent goods, Smart goods, Distributed intelligence, Agents, Transportation.

1 Introduction

The increasing demands on transports related to global flows, just-in-time deliveries, intermodality etc., call for more and more complex logistics solutions. In order to cope with this, new instruments are continuously being developed and one of the concepts sometimes mentioned in this context is "intelligent goods". Generally speaking, this concept usually refers to goods equipped with some information and/or communication technology, giving it capabilities beyond traditional goods. The research on intelligent goods is continuously progressing and as a result of this, a great number of test cases and pilots are being launched to show the potential of the concept [5, 8, 13]. Moreover, a few solutions related to intelligent goods have also actually entered the transport market [8]. In this paper we present a way to categorize the different types and levels of capability that are relevant in the context of intelligent goods.

Radio Frequency Identification (RFID) is usually assumed to be involved in solutions based on intelligent goods. Some of the previous studies within RFID have focused on the feasibility of mobile RFID solutions in supply chain management [7, 17]. Furthermore, the idea of combining RFID and agent technology has recently also attracted research efforts. However, we believe that the question of how to model intelligent goods as agents still needs further investigations. The EU funded project EURIDICE (EURopean Inter-Disciplinary research on Intelligent Cargo for efficient, safe and Environment-friendly logistics) has specified a number of general agents that can be used as an instrument for implementing solutions based on intelligent goods [6]. This paper aims at contributing to this area by modeling three specific intelligent goods services as agents. Both single agent and multi-agent solutions are given and compared. The agent solutions are furthermore categorized according to the intelligent goods framework. Different locations of the agents are also discussed.

The paper is structured as follows. The next section outlines the intelligent goods framework. Section 3 presents the three intelligent goods services and section 4 describes the agent models. Finally, in section 5, some conclusions are drawn.

2 Intelligent Goods

Just as Artificial Intelligence (AI) research initially was focused on how to replicate human intelligence in a machine, intelligent goods derive from the idea of making the goods, to some extent, act as human travellers. Intelligent goods is an established concept that, however, generally comprises goods that would not be considered as intelligent within AI. It is hereby important to separate intelligence in the AI sense (which relates to the human intelligence), from intelligent goods (which indicates goods provided with electronically enhanced capabilities, usually used during transport). This confusion further motivates the need for a specific definition of the concept.

2.1 Definition

Research within the area of intelligent goods, use a number of different denotations for identical or similar concepts, e.g. intelligent cargo [6], smart goods [7], smart freight [12], intelligent packaging [11] etc. The meanings of these concepts are usually not identical, and often not precisely defined, but they do strive in the same direction. In contrast to previous definitions and descriptions, we include several levels of capability in the definition, in order to reflect a potential for different levels of (enhanced) behavior, e.g. from simply knowing its own identity to autonomous decision making. The higher capability levels indicate a potential for intelligent operations in the more traditional sense whereas the lower do not. However, only providing the goods with a relatively high level of capability is not enough to make the goods intelligent. The added value is created when these capabilities are used in a purposive way.

We suggest that the capability levels should be characterized based on a number of dimensions, corresponding to different types of capabilities. These dimensions differ from the ones traditionally used within AI since they are partly motivated by the requirements from intelligent goods services [10]. These dimensions and their different values, ordered according to the capability degree, are shown in Table 1. The list has been developed based on the definition of smart freight [12], the requirements from potential intelligent goods services [10] and a number of different agent definitions and levels of agent capabilities [3] [15].

Table 1. Capability dimensions related to goods

Dimension	Capability
A. Memory storage	1. Ability to store ID 2. Ability to store goods data (other than ID) 3. Ability to store algorithms/decision rules
B. Memory dynamics	1. Static memory 2. Ability to change/add/delete data (other than ID) 3. Ability to change/add/delete algorithms/decision rules
C. Communication out	1. Data (including ID) can be read by external unit (e.g. barcode, simple RFID) 2. Ability to send short-range communication messages 3. Ability to send long-range communication messages
D. Communication in	1. None 2. Ability to receive short-range communication messages 3. Ability to receive long-range communication messages
E. Processing	1. None 2. Ability to execute decision rules (e.g. If–Then statements) 3. Ability to execute algorithms (e.g. planning capability, optimization algorithms)
F. Autonomy	1. None 2. Reactive capability (actions must be triggered by an external unit) 3. Proactive capability (no external trigger needed)
G. Sensor	1. None 2. Sensor capability incl. ability to read sensor data (e.g. enclosed position or temperature sensors)
H. Time	1. None 2. Ability to measure time intervals 3. Ability to determine actual time

In Table 1, a lower number indicates a lower capability level within the same dimension. We assume that the ID of the goods is static and that the goods always store at least an ID. Please note that the implementations of the capability dimensions do not have to be physically located at the goods. For instance, the decision rules may be stored on the goods whereas the processing of the rules and the sensor management may be performed by two separate nearby units.

The dimensions show that the lowest capability level (i.e. capability 1 in all dimensions) corresponds to the simple bar codes often used in retail today. This level also includes the simplest forms of RFID tags. We suggest that goods should be included in the intelligent goods concept if at least one of the capabilities lies above 1, thus implying an extension of the bar code situation. The concept would hereby represent goods with varying capability levels but that do fulfill this requirement.

A potential usage of the capability dimensions is to map different services based on intelligent goods, to their implementation requirements. A higher level implies a higher requirement on the implementation. In particular, it is of interest to use these dimensions when multiple services should coexist. For instance, consider a set of services that are supposed to be implemented using agents. If the capabilities of both agents and services are mapped to the capability dimensions, a comparison between these two mappings will reveal what services can be supported by a specific set of agents. Alternatively, the total requirements of the services can be used to specify the capabilities of the agents needed to support the complete set of services.

A practical example of areas that potentially might benefit from using the intelligent goods framework is the RFID technological development. Depending on application and development, different capability levels are today implemented on the RFID tags. Here, the framework could be used as a way to classify the different RFID tags (e.g. active and passive tags).

2.2 Capability Dependencies

There are several dependencies between the different capability dimensions. For instance, a capability of only being able to store an ID and not any data (A1) can't be combined with the capability of changing data (B2), since there is no data to change. Naturally, different services put different requirements in terms of capability dimensions. It is thereby relevant to present the dependencies between the different capability dimensions, i.e. how the requirement of a capability propagates to other dimensions (see Table 2).

According to Table 2, a reactive capability (F2) requires the ability to store algorithms/decision rules (A3) and at least the ability to execute decision rules (E2).

Table 2. Dependencies between dimensions

Capability	**Required capability**
Memory dynamics: B	Memory storage: A >= B
Communication out: C > 1	Memory storage: A = 3, Processing: E > 1
Communication in: D > 1	Memory storage: A = 3, Memory dynamics: B > 1, Processing: E > 1
Processing: E > 1	Memory storage: A = 3, Memory dynamics: B > 1, Autonomy: F > 1
Autonomy: F = 2	Memory storage: A = 3, Processing: E > 1, Communication in: D > 1
Autonomy: F = 3	Memory storage: A = 3, Memory dynamics: B > 1, Processing: E > 1, Time: H > 1 or Sensor: G = 2
Sensor: G = 2	Memory storage: A = 3, Memory dynamics: B > 1, Processing: E > 1
Time: H > 1	Memory storage: A = 3, Memory dynamics: B > 1, Processing: E > 1

This indicates that the capability of being reactive requires some kind of processing. Thus, we do not consider for instance bar codes or passive RFID tags as reactive, since these are simply scanned by a reader, without being able to for instance decide for themselves which information to send or whether to send any information at all.

Based on the capability dimensions and dependencies it is possible to create a list of all possible combinations of capabilities included in the intelligent gods concept.

3 Services

The list of services that might benefit from being realized based on intelligent goods is extensive. In this paper we focus on services useful during loading, transport and unloading. In order to show how intelligent goods can be modeled as agents and what capability levels are required for different types of services, we present three concrete and illustrative example services below. These services originate from a list of primitive services that may be used as building blocks when creating more complex intelligent goods services [10]. The main reason for selecting these services is that they are simple, which makes the principles behind the analysis clear and easy to understand. The benefits of modeling intelligent goods as agents will however be more apparent with more advanced services.

1. **Delay notification service:** notifies designated receivers about actual arrival times when the goods are arriving at a stop after the specified delivery time window.
2. **Priority service:** informs about the priority of the goods, as well as calculates the priority based on the customer class and estimated time of arrival to the final destination (or at least to one of the destinations ahead).
3. **Transport conditions service:** records condition values (e.g. temperature) at specific time intervals, and sends these to external units upon request.

The above services may be implemented using intelligent goods (e.g. instead of centralized configuration) and the intelligent goods may in turn be modeled as agents and possibly implemented using agent technology. Next we will show which requirements these services put on such agents. These requirements can then be expressed in terms of the presented capability dimensions of intelligent goods.

Please note that there might be alternatives to using agent technology when implementing a service. For instance, the former company Bioett [2] used a biosensor to track the accumulated temperature, which could be read by a handheld scanner. A mapping of this solution to the intelligent goods framework shows that it falls within the category of intelligent goods (since it has the capability of adding data). In general, though, using agent technology for intelligent goods services has many advantages. For instance, agents represent a natural way of modeling intelligent goods, and they allow for autonomous solutions that decreases the need for human intervention. Furthermore, agents enable the goods to interconnect which may be useful in situations where goods need to coordinate with other goods, for instance to make sure that only goods that are allowed to be grouped together are loaded onto the same vehicle (e.g. certain types of reacting dangerous goods are forbidden to be grouped together). Finally, agent technology typically provides benefits in terms of computational efficiency, extensibility, robustness, responsiveness, flexibility and reusability.

4 Agents

There are a number of different approaches to the formal modeling of agent systems. We have chosen to base the modeling of our three services on the Agent Unified Modeling Language (AUML), as described in [14]. AUML represents an extension to UML and is a well-established modeling approach for multi-agent systems. It is supported by FIPA (the Foundation for Intelligent Physical Agents) and OMG (Object Management Group). AUML allows describing both inter- and intra-agents behavior and is therefore well-suited for our purposes. We use sequence diagrams to illustrate the inter-agent behaviors, and activity diagrams for intra-agent behaviors. The methodology chosen for the design of the agents is GAIA [16], which is a general methodology that supports both the micro-level (agent structure) and macro-level (agent society and organization structure) aspects of systems [16]. In this paper we use it for the system analysis and high-level design of the agents that are subsequently described in more detail using AUML. Since we only use GAIA for a high-level design, we will follow the methodology described in [16] and, for instance, not the more extensive one presented in [18]. We regard a high-level design as sufficient in this case since the multi-agent system is of limited size and has relatively straightforward functionalities.

Below we will initially assume we only have one single agent for each service, i.e. one agent is responsible for all functionalities related to a service. In practice however, different parts of the functionality may be placed on different units. Based on the GAIA methodology, the agents presented in section 4.3 show one way of dividing the service functionality between different agents. In these solutions functionality that can be reused by more than one service, have been extracted and placed in separate agents. The structure of this result naturally depends on the set of services considered. Following this approach, a separate investigation of a large number of potential intelligent goods services should hereby result in, apart from the service specific agents, a number of agents with basic functionality forming a foundation for service implementation. This type of investigation represents future work.

In a real world case, some form of security mechanism will most probably be needed in order to restrict who is allowed to access the involved information entities and make use of the service functionalities. This paper, does not address these issues.

4.1 Information Entities

In order to realize the services presented above, some information related to the goods are needed. We refer to this information as "information entities". For instance, the Delay notification service needs information about the specified delivery time window, which is unique for each transport item. Additionally, information related to the "housing", e.g. transport container, terminal or vehicle, of the goods is also needed. Table 3 lists the information entities required to fulfill the three services in question. These information entities originate from a more extensive set presented in [10].

Table 3. Required information entities for the services

No	Goods information entities (GE)	
1	GE-ID	<ID>, where ID is the unique goods identity, e.g. SGTIN, SSCC, GRAI [4]
2	GE-Next Destination	<position>, where position is an address, SGLN [4] or a set of coordinates
3	GE-Itinerary	Sequence of <position, accountable ID, time 1, time 2>, where position is an address, SGLN or a set of coordinates, accountable ID is the organization currently accountable for the goods (usually a transport company), and time 1 and time 2 represent the delivery time window
4	GE-Priority	<priority>, where priority is the delivery priority of the goods
5	GE-Customer Class	<class>, where class indicates the priority ranking of a transport customer (for instance based on the amount paid for a reliable delivery time of the goods)
6	GE-Recorded Conditions	Set of <condition type, a sequence of <time stamp, value>>, where condition type is the stored condition (e.g. temperature), time stamp is the time of measurement, and value is the condition value (e.g. temperature value)
7	GE-Set of Information Receivers	Set of <service no, receiver contact>, where the service number is a predefined number identifying the service (e.g. 1, 2 or 3 in our list) and receiver contact is the contacting details of the receiver of the information (e.g. telephone number)
No	**Housing information entities (HE)**	
8	HE-Accountable	<accountable ID>, where accountable ID is the organization (currently) accountable for the housing
No	**Housing information entities (HE) – vehicle specific**	
9	HE-Itinerary	Sequence of <position>, where position is an address, SGLN or a set of coordinates
10	HE-ETA	Estimated time of arrival of a vehicle, a set of pairs <position, ETA>, where position corresponds to the positions in HE-Itinerary

Table 3 relates the information entities to either the goods or the housing level but it says nothing about actual location of the information entities. For our purposes however, the goods information entities are assumed to be stored inside the agents, whereas the housing information entities are considered to be parts of the input data messages. In particular, HE-ETA denotes the estimated time of arrival to the different stops of a vehicle, i.e. it is vehicle specific but in theory it can be stored anywhere. We assume that this particular information may be aggregated and stored in the terminals/warehouses. The goods may hereby acquire information about possible ETAs to a number of destinations of different vehicles, when arriving to a stop. This information may furthermore contain several ETAs for each specified position.

4.2 Single Agents

This section illustrates how the services presented in section 3 can be modeled as one agent per service. For each model, the required capabilities, in terms of the identified capability dimensions connected to intelligent goods, are also presented. In a real world case, these models are useful when only one or a few intelligent goods services are implemented, i.e. when there is no use of a multi-agent system.

Delay Notification Service

Agent description: The agent, denoted as A1, is triggered by an external unit notifying the agent about the arrival to a stop (e.g. an RFID sender at the gateway). The notification includes the stop position. In order to find the required delivery time window to the next stop, the information entity GE-Next Destination is used to search through GE-Itinerary. Furthermore, GE-Set of Information Receivers is used to find the recipients of the notifications.

Required capabilities: A3, B2, C3, D2, E2, F2, G1, H3

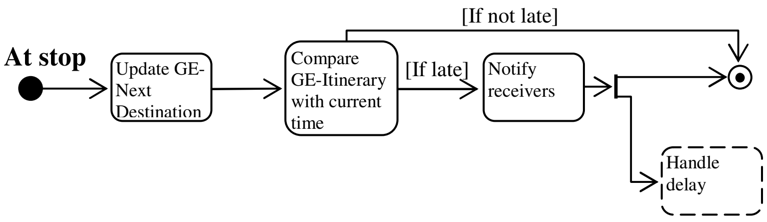

[If not late]
At stop
Update GE-Next Destination
Compare GE-Itinerary with current time
[If late]
Notify receivers
Handle delay

Fig. 1. Activity diagram for the Delay notification service agent model (A1)

Priority Service

Agent description: As in the Delay notification service, the agent, denoted as A2, is triggered by an external unit notifying the agent about the arrival to a stop. The agent thereafter asks for information about all available values of ETA to the different stops ahead (stored in GE-Itinerary). Based on the ETAs and information about the customer priority class (i.e. GE-Customer Class), the agent may calculate a new priority of the goods (GE-Priority). For instance, if the goods are delayed and the class of priority ranking is high, the priority of the goods is increased. The information provision about the goods priority is based on the itinerary and who is accountable for the transport. This means that an answer to a priority question is only given if the accountable ID matches the accountable ID in the question (HE-Accountable) and if the next stop of the goods can be found within the vehicle itinerary (HE-Itinerary), also included in the question.

Required capabilities: A3, B2, C2, D2, E3, F2, G1, H1

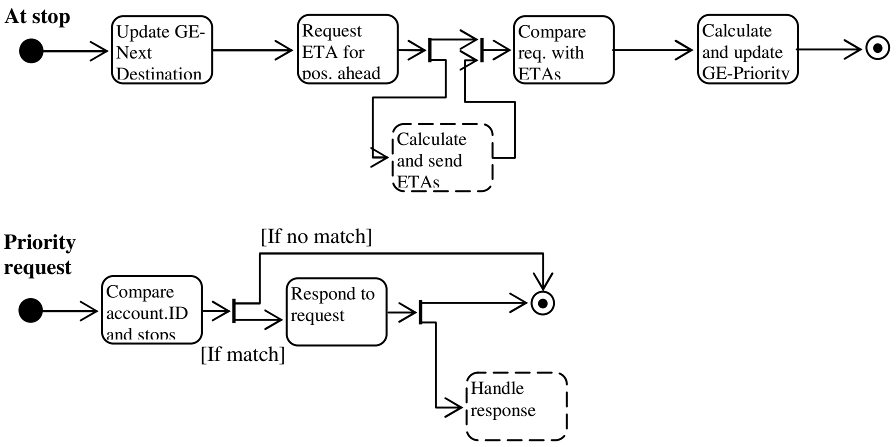


Fig. 2. Activity diagram for the Priority service agent model (A2)

Transport Conditions Service

Agent description: The agent, denoted as A3, retrieves sensor data from its condition sensors at time intervals. It uses the information entity GE-Recorded Conditions to store the values and put a time stamp, reflecting the actual time, on each of them. The agent responds to requests of stored sensor data for a certain time period.

Required capabilities: A3, B2, C2, D2, E2, F3, G2, H3

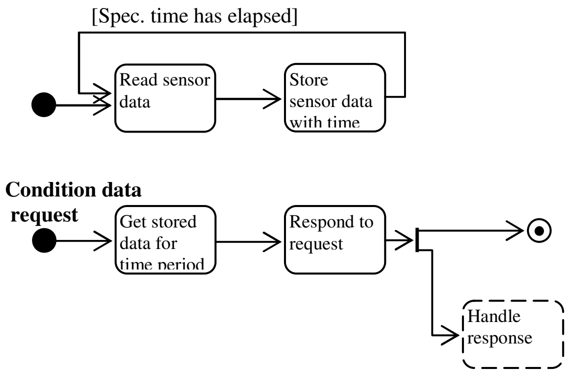


Fig. 3. Activity diagram for the Transport conditions service agent model (A3)

Discussion. Since the three services differ in functionality, the corresponding agents have different capability levels. They do have a few things in common though. They all need the ability to store algorithms/decision rules and to change/add/delete data. They also require the ability to receive short-range communication messages. These seem to be fundamental capabilities, at least for these three services.

From a functional perspective, they also have a few things in common. All agents include one internal database each, which is used for storing the service specific goods information entities. Two of the agents are furthermore dependent on a correct update of GE-Next Destination. These duplicated functionalities can be extracted into separate agents. However, as mention before, the value of this naturally depends on the set of services to be implemented. In particular, storing two copies of the same goods information entity in two different agents might involve some risks, especially if it is a dynamic information entity that needs to be updated from time to time. In some cases though, redundant information entities might give the agent control over the information. For instance, the Delay notification service updates the information entity GE-Next Destination whenever the goods reach a new stop. This implementation of the service is dependent on a correct update of GE-Next Destination, which triggers the other parts of the service. In this case, storing and updating the information entity inside the agent is beneficial since it gives the agent complete control of when it is updated. However, this can be solved in a distributed solution as well, which will be shown in the next section.

4.3 Multiple Agents

A multi-agent system is beneficial when several intelligent goods services are implemented, in particular when they require similar functionality. This section presents such a multi-agent system based on the three services presented in section 3. Following GAIA we have, as part of the first analysis phase, identified the roles[1] of the system, presented below:

- DelayNotifier, who is responsible for notifying designated receivers about actual arrival times when the goods are arriving at a stop after the delivery time window;
- PriorityHandler, who is responsible for answering priority requests as well as calculating and updating the goods priority;
- ConditionHandler, who is responsible for updating condition values at specific time intervals and send these to external units upon request;
- Database, who handles information entities;
- Sensor, who handles a sensor; and finally
- DestinationUpdater, who is responsible for updating the goods information entity GE-Next Destination.

Based on these roles, we have created a GAIA role model, with one role schema for each role. As an example, Table 4 shows the schema for the DelayNotifier role.

In the interaction model, which is the last part of the analysis phase of GAIA, all interactions between the different roles are identified. Due to limited space, the interaction model will not be presented in this paper. However, both the acquaintance model and, in particular, the AUML diagrams will show all necessary interactions in the multi-agent system.

[1] "Here a role can be viewed as an abstract description of an entity's expected function." [16]

Table 4. GAIA role schema for the DelayNotifier role

Role Schema: DelayNotifier

Description:
Generates a delay notification if the time of arrival to a stop is later than specified in itinerary.

Protocols and Activities:
AwaitStop, GetInformation, <u>ProduceNotification</u>, InformReceivers

---

Permissions:

Reads	*goodsID*	*//ID of the goods*
Reads	*goodsItinerary*	*//Itinerary of the goods*
Reads supplied	*atStop*	*//current stop, if any*
Reads	*informationReceivers*	*//receivers of notification*
Generates	*delayNotification*	*//information about delay*

Responsibilities
Liveness:
DelayNotifier = (AwaitStop, GetInformation, <u>ProduceNotification</u>, InformReceivers)$^{\omega}$
Safety:
Time at stop > upper limit in deliv. time window => notification

The design phase of GAIA involves three different models: the agent model, the service model and the acquaintance model. The purpose of the agent model is to identify the agent types connected to the agent roles, and the number of agent instances that will appear in the system. In our agent model there is a one-to-one correspondence between the roles and the agent types for all roles except one; the sensor role. The system requires one or several sensor agents since the condition service might make use of more than one condition sensor. We have hereby identified the following agents: DatabaseAgent, SensorAgent, DestinationUpdaterAgent, DelayNotifierAgent, PriorityHandlerAgent and ConditionHandlerAgent.

Fig. 4 shows the acquaintance model, which defines the communication links that exists between the different agent types in the system. Following GAIA, we have also identified the services related to each role, according to the service model description. These services derive from the list of protocols, activities, responsibilities and liveness properties of the roles [16]. For example, the GetInformation protocol corresponds to one of the services. The identified services will, however, not be described according to the GAIA methodology in this paper, but examples are presented as a part of the AUML diagrams instead.

In general, sensors and databases may be modeled either as agents or simply as resources of the environment [18]. We have chosen to model them as agents since

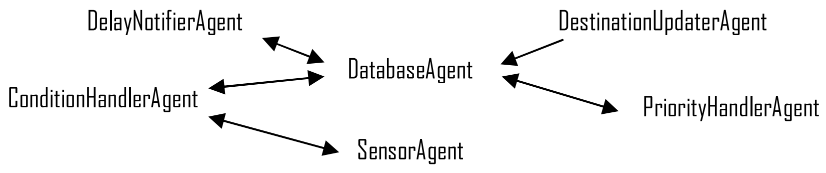


Fig. 4. GAIA acquaintance model

they perform more complex operations than just influencing the mechanisms by which the other agents retrieve resources (according to guidelines presented in [18]). These database and sensor agents are actively involved in the agent interactions, for instance by using event-notification mechanisms.

In the resulting multi-agent system, we have extracted two functionalities from the single agents, into separate agents types; the DatabaseAgent responsible for the database holding the relevant goods information entities and the SensorAgent, responsible for the sensors. Even though the sensor functionality is only present in one of the single agents, we have extracted this into a separate agent type since we regard this as a general functionality needed by several intelligent goods services (e.g. track and trace, geofencing, real time condition control). Similarly, the DestinationUpdaterAgent is in this system responsible for a relatively small task. However, we believe that several services are dependent on the correct update of different information entities and therefore it is convenient to place such functionality in a separate agent.

We use sequence diagrams to illustrate the interactions between the agent types, and activity diagrams to describe the intra-agent behaviors, as defined in AUML.

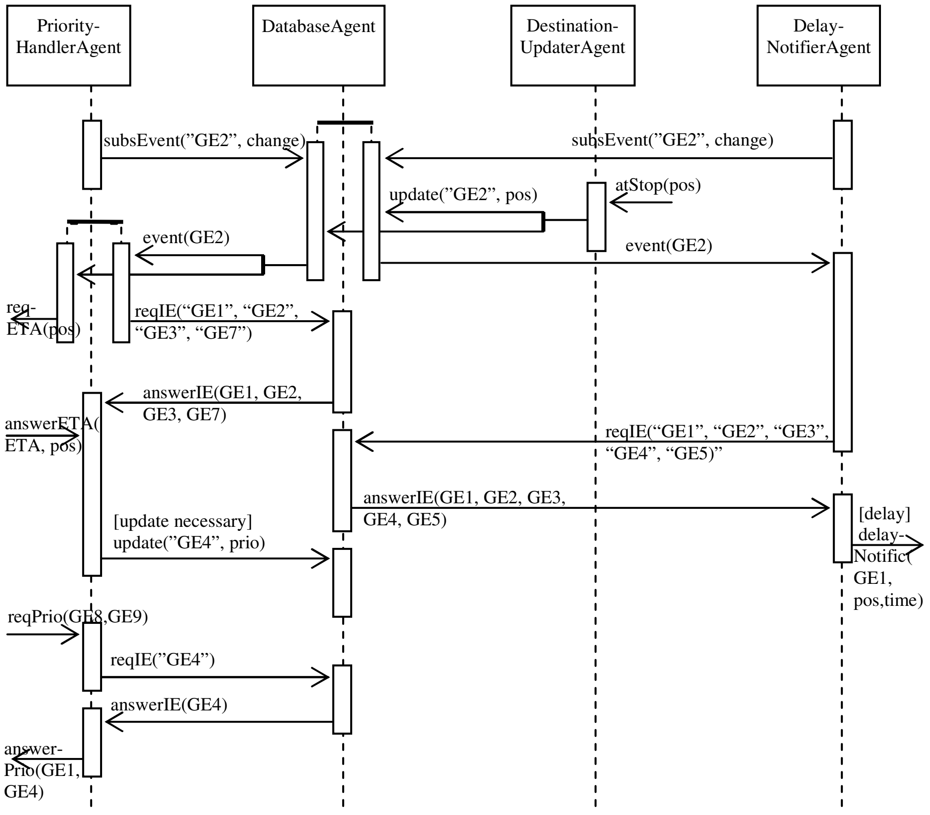


Fig. 5. Sequence diagram showing the inter-agent behaviors (1)

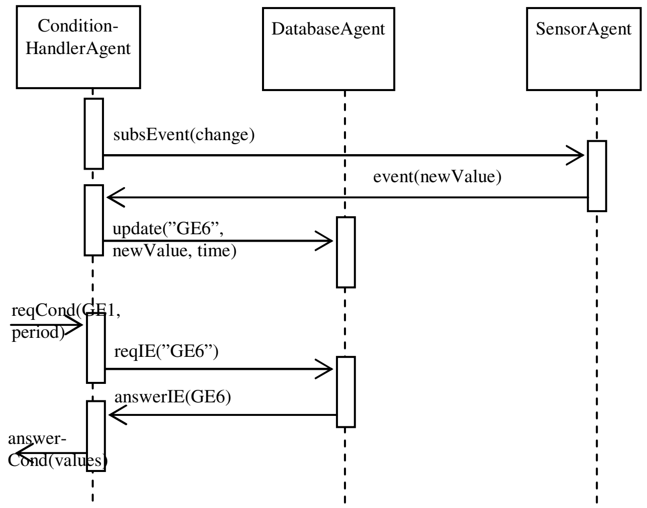


Fig. 6. Sequence diagram showing the inter-agent behaviors (2)

Fig. 5 and 6 shows the interaction between the reduced service specific agents, i.e. PriorityHandlerAgent, ConditionHandlerAgent, DealyNotifierAgent (denoted PHA, CHA, DNA below) and the basic agents, i.e. DatabaseAgent, SensorAgent (denoted DA and SA). The service specific agents are assumed to hold the service specific functionalities that are not supported by the basic agents. Fig. 5 also includes an additional agent, DestinationUpdaterAgent (denoted as DUA). This agent represents an extraction of the previous single agents, but it should rather be seen as a subservice using the other basic agents while assisting the three service specific agents. It thereby both uses the other basic agents and is used by the service specific agents.

Using more than one agent to implement an intelligent goods service usually provides a higher level of flexibility in the sense that the agents can be spread out in an optimized way. For instance, the physical location (goods level, vehicle level etc.) of an agent might influence the decision of what functionality to include in that agent.

Table 5. Services modeled as agents are mapped to intelligent goods dimensions

	A1	A2	A3	DA	SA	DNA	PHA	CHA	DUA
Memory storage	3	3	3	3	3	3	3	3	3
Memory dynamics	2	2	2	2	2	2	2	2	2
Com. out	3	2	2	2	2	3	2	2	2
Com. in	2	2	2	2	2	2	2	2	2
Processing	2	3	2	2	2	2	3	2	2
Autonomy	2	2	3	2	3	2*	2*	2*	2
Sensor	1	1	2	1	2	1	1	1	1
Time	3	1	3	1	2	3	1	3	1

*) The initial subscribe-call does not make the agent proactive since it is only called when the service is activated and may be triggered by the activation.

Table 5 shows the mappings of the agents representing the divided functionality to the capability dimensions presented in section 2. The table also includes the corresponding mappings of the original service agents (A1-A3), presented in section 4.2, as reference material. The fundamental capability dimensions, such as memory storage and memory dynamics remain the same for all agents, but the rest of the dimension levels differ.

4.4 Locations of Agents

There are several ways of dividing the functionality of services between different parts of a system. The information entities related to the goods and the algorithms/decision rules may be stored for instance either on the goods, on the transport container/vehicle/terminal or centrally. Moreover, this data may also be distributed between different parts of the system. Different parts of the processing of a service may similarly be conducted on various units. Furthermore, the data storage might very well be separated from the processing. For instance, the data needed by an intelligent goods service might be located on the goods, whereas the service processing might be conducted on for instance the vehicle or central level.

Fig. 7 shows the context of the goods. In particular, the figure shows the main alternatives for placing data, rules and processing connected to a service. There are three different information processing levels; the ERP (Enterprise Resource Planning), the Housing and the Goods level. The figure furthermore shows the stages the goods go through during transport, and the different communication paths that might exist between the information processing levels. IS denotes the information system, which is responsible for any communication and processing that might exist on a level. An RFID tag might for instance represent the IS on goods level.

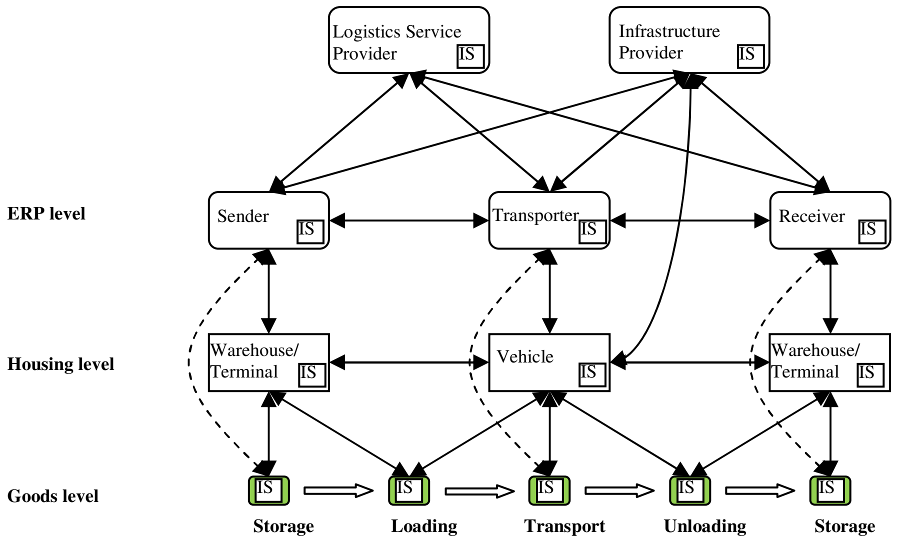


Fig. 7. The context of the goods

The optimal solution to the problem of where to place data, rules and processing is dependent on things like the service in question, communication link availability, costs etc. Generally, from a strict functional perspective, if the complete service functionality can be conducted locally (i.e. on goods, transport container and/or vehicle/terminal level), the service becomes independent of higher levels. It may thereby be performed in situations where the communications towards other units are cut off. A higher level of autonomy is obtained. Another advantage with local services is that the closer to the goods the sensors are placed, the more precise sensor data can be obtained. All these advantages must however, as mentioned above, be valued against all other factors specific for each situation and service.

5 Conclusions

We have used three specific intelligent goods services to illustrate how intelligent goods can be modeled as agents. The capabilities of the agents have been related to a novel intelligent goods framework, which is characterized by a number of capability dimensions. In relation to this, we have suggested a lowest requirement for when the goods should be included in the intelligent goods concept. Finally, we have discussed some considerations and effects of different placements of the data, decision rules, algorithms and processing corresponding to a service.

As a part of our future work, the model will be evaluated through simulation experiments and further through real test cases.

Acknowledgements. This research has been carried out within two projects called “Intelligent industrial goods and ERP systems” [9], funded by the Swedish Road Administration, and “ARENA – electronic fee collection for heavy vehicles” [1], funded by the Swedish Road Administration and Vinnova.

References

1. ARENA, project, `http://www.arena-ruc.com/?info=star`
2. Coenen, L., Moodysson, J.: Putting Constructed Regional Advantage into Swedish Practice. European Planning Studies 17(4), 587–604 (2009)
3. Davidsson, P., Johansson, S.J.: On the Metaphysics of Agents (extended version). BTH research report 2005:4 (2005) ISSN: 1103-1581
4. EPCglobal, `http://www.epcglobalinc.org`
5. EURIDICE, project, `http://www.euridice-project.eu`
6. EURIDICE WP 14 D14.1: Preliminary User, System and Communication Services Definition (2008), `http://www.euridice-project.eu`
7. Holmqvist, M., Stefansson, G.: Mobile RFID A case from Volvo on innovation in SCM. In: Proceedings from the 39th Hawaii International Conference on System Sciences (2006)
8. Huschebeck, M., Piers, R., Mans, D., Schygulla, M., Wild, D.: Intelligent Cargo Systems Study (ICSS) - Impact assessment study on the introduction of intelligent cargo systems in transport logistics industry (2009), `http://www.intelligentcargo.eu/`
9. Intelligent industrial goods and ERP systems, project, `http://www.bth.se/tek/intelligent_gods`

10. Jevinger, Å., Persson, J.A., Davidsson, P.: Analysis of transport services based on intelligent goods. In: NOFOMA Conference, Kolding, Denmark (2010)
11. Johansson, O.: On the value of intelligent packaging - a packaging logistics perspective. PhD dissertation, Division of Packaging Logistics, Lund University of Technology, Lund, Sweden (2009)
12. Lumsden, K., Stefansson, G.: Smart freight to enhance control of supply chains. International Journal of Logistics Systems and Management 3(3), 315–329 (2007)
13. Mirzabeiki, V., Ringsberg, H., Lumsden, K.: Effects of Using Smart Goods on Traceability Information and Carried out Activities in Supply Chains of Fresh Food – A Cross Case Analysi. In: Nordic Logistics Research Networks (NOFOMA) Conference, Kolding, Denmark (2010)
14. Odell, J.J., Van Dyke Parunak, H., Bauer, B.: Representing Agent Interaction Protocols in UML. In: Ciancarini, P., Wooldridge, M.J. (eds.) AOSE 2000. LNCS, vol. 1957, pp. 121–140. Springer, Heidelberg (2001)
15. Wooldridge, M.: An introduction to multiagent systems, 2nd edn. John Wiley & Sons Ltd. (2009) ISBN 9780470519462
16. Wooldridge, M., Jennings, N.R., Kinny, D.: The Gaia methodology for agent-oriented analysis and design. Journal of Autonomous Agents and Multi-Agent Systems 3(3), 285–312 (2000)
17. Wu, J., Wang, D., Sheng, H.: Design an OSGi extension service for mobile RFID applications. In: IEEE International Conference on e-Business Engineering, Hong Kong, pp. 323–326 (2007)
18. Zambonelli, F., Jennings, N., Wooldridge, M.: Developing multiagent systems: The gaia methodology. ACM Transactions on Software Engineering and Methodology 12(3), 317–370 (2003)

Multi-Agent Systems for Biomedical Simulation: Modeling Vascularization of Porous Scaffolds

Hamidreza Mehdizadeh[1], Arsun Artel[1], Eric M. Brey[1,2], and Ali Cinar[1]

[1] Illinois Institute of Technology, Chicago IL 60616, USA
{mehdizadeh,brey,cinar}@iit.edu, arsun.artel@mam.gov.tr
[2] Research Service, Hines VA Hospital, Hines IL 60141, USA

Abstract. An interesting application of multi-agent systems (MAS) is in modeling systems that can be represented by independent entities interacting together, the so-called agent-based modeling (ABM). In this paper MAS paradigm is used as a promising technique for representing complex biomedical systems. A brief survey of some ABM of biomedical systems is presented, followed by the description of a multi-layered agent-based framework developed in our own labs to model the process of sprouting angiogenesis (blood vessel formation) within polymeric porous scaffolds used for regenerative medicine. The ABM structure developed and challenges in modeling systems with a large number of rapidly increasing interacting agents are discussed. 2D and 3D case studies are presented to investigate the impact of scaffold pore structure on vessel growth. MAS provides a valuable tool for studying highly complex biological and biomedical systems, and for investigating ways of intervening in such systems.

Keywords: agent-based simulation, agent-based modeling, emergent behavior, angiogenesis.

1 Introduction

The number and diversity of the fields that use multi-agent systems (MAS) is rapidly growing. Computational biology and biomedical engineering communities have started using multi-agent modeling techniques to study complex systems with many interacting elements. During its relatively short but fast developing lifetime, agent-based modeling (ABM) paradigm has shown to be a promising way of representing biological and biomedical systems and describing their dynamic behavior.

The central idea in ABMs is to define agents that represent the building blocks of a system and to develop rules that regulate their interactions. The rules originate from the vast knowledge gained through many years of studying the individual components of these biological systems. Agent-based systems (ABS) are naturally suitable for modeling biological systems as they are comprised of individual discrete micro-scale constituents (e.g. cells) that interact with each other to form non-homogeneous macro-scale bodies (e.g. tissues and organs). Table 1 provides an illustrative list of applications of ABM in biomedical engineering. A brief review highlights how ABM is used in developing biomedical models and displays the diversity of the applications.

Table 1. An illustrative list of biological and biomedical applications of ABM

Field	**Example**	**Reference**
Cancer	Multi-scale, multi-resolution brain cancer modeling	[27]
Tissue engineering	ABM of neovascularization within porous scaffolds	[3]
Angiogenesis	Module-based multiscale simulation of angiogenesis in skeletal muscle	[16]
Lung disease	ABM of inflammation and fibrosis following particulate exposure in the lung	[8]
Clinical	In silico experiments of existing and hypothetical cytokine-directed clinical trials using ABM	[1]
Morphogenesis	Simulating properties of in vitro epithelial cell morphogenesis	[13]
Bone	ABM of real-time signaling induced in osteocytic networks by mechanical stimuli	[4]
Epidemiology	ABM of the spread of the 1918-1919 flu in three Canadian fur trading communities	[19]

Zhang et al. developed a multi-scale agent-based framework for modeling cancerous brain tumors [27]. They simulated the growth and expansion of brain tumors, utilizing a novel multi-resolution approach to substantially reduce the computation time of the individual-based model while maintaining a comparably high predictive power. Artel et al. developed an agent-based model to investigate the effects of porous scaffold structure on rate and characteristics of angiogenesis (blood vessel formation) [3]. Liu et al. used an ABM with complex logical rules and equation-based models integrated into a uniform framework for studying angiogenesis in skeletal muscle [16].

Brown et al. developed an ABM to capture the features of inflammation caused by particulate exposure in the lung [8]. Their model considered a limited number of relevant interactions among different cell types and their effect on tissue damage. An used a simple ABM that could evaluate the dynamics of the innate immune response and demonstrated counterintuitive outcomes of the system and the difficulty of effective manipulation of complex multi-component systems [1]. Grant et al. developed a model that could capture the interconversion of different cell types, with behavior of cell agents being governed by a set of axioms derived from observation of real cell behaviors and a decision process embedded within each agent [13]. Ausk et al. developed an ABM for studying networks of bone cells when affected by mechanical stimuli [4].

O'Neil and Sattenspiel modeled the epidemic spread of the flu between three real communities by means of stochastic agent-based computer simulation [19]. Their model was successful in describing the course of the flu spread in small communities. Thorne et al. have reviewed biological ABMs and compared them with continuum computational models [24]. In another work, they emphasized the importance of combining ABMs with experimental work to gain new understanding from the experiments and presented a number of such models [25].

Recent advances in genetics and in molecular and cellular biology have generated a vast amount of biological information. A systematic approach is needed to integrate this information in a unified framework, and to facilitate its use for multi-scale modeling. Traditionally, the evolution and patterning of the naturally discrete cells were modeled using a set of continuous differential equations. Even simple ABMs can develop complicated behavioral patterns and provide valuable insight about the dynamics of real-world systems. The MAS paradigm provides a versatile modeling environment where each cell can be represented by a software agent that has a set of states and behaviors and an internal rule-based logic to relate them. A hierarchical MAS can link systematically the effect of mechanisms in a lower level of biological scale (e.g. molecular or cellular levels) with higher level (e.g. tissue level) patterns and behaviors [25]. The flexibility and modularity of MAS enable the development of a model that can be modified, extended, or expanded with a minimum amount of effort.

2 Use of Agents in Biological Systems

Agents are autonomous software entities designed to perform proactively or reactively various tasks related to the specific system they represent. In medical system simulation and modeling, agents usually represent cells or cellular components (such as cell receptors or integrins) that interact with each other and their local environment. These interactions lead to higher-level emergent phenomena, such as blood vessels or tissue formation, or cancerous tumor invasion.

A list of the characteristics of ABS that are important in simulating biomedical systems include:

1. *Modularity:* The behavior of different agents are determined at execution time based on rules and parameter values computed dynamically during the simulation. It is possible to add new rules at later stages of model development. It is also possible to add different types of agents to the model. The effects of new rules and agents on current rule-bases should be carefully considered for conflict resolution.
2. *Abstraction:* ABMs can be constructed with less quantitative information about the behavior of a system compared to models based exclusively on fundamental equations.
3. *Multi-scale modeling:* ABMs provide a powerful structure to describe phenomena occurring in different scales of biological systems in a single model. One model can include details in the molecular level, and produce results in the tissue or organ level.

4. *Randomness:* There are many unknowns in biological and biomedical systems due to lack of measurement or precision. By adding randomness to models, it is possible to accommodate the lack of knowledge and to account for the stochastic nature of phenomena at various physical or biological scales. ABM rules can represent the stochastic and deterministic phenomena.
5. *Decision-making at runtime:* An agent is capable of deciding about its behavior during runtime based on the dynamic behavior of its surroundings as the simulation is progressing. Hence, ABMs incorporate dynamically the effects of interactions with an agent's environment and neighbors, and conveniently account for the influence of random variations at agent level.
6. *Emergent behavior:* ABMs can generate complex emergent behaviors even with a few rules. They provide a powerful environment to describe emergence of behavior at one scale based on the events occurring at another scale.

Software agents can be equipped with different types of goals. These goals form the motivational component of agents and enable them to act proactively. Agent goals can be classified into the following taxonomy [26]:

1. *Performance goals* that the agents that are programmed to perform certain tasks must achieve
2. *Maintenance goals* which represent a state that agents want to maintain
3. *Achievement goals* which represent a desired state that the agent wants to reach
4. *Query (or test) goals* which represent an intention to obtain certain information

Additional goal types include combined goal types, such as "achieve then maintain" goals, which force an agent to reach a state and then to keep that state in later times [12]. Agents used for biomedical systems modeling usually possess only performance goals. Performance goals are a procedural type of goal and indicate that the goal of the agent is to execute certain well-defined actions. This is due to the fact that each agent in these systems is only aware of the functions it is entitled to perform, without knowing the higher level goal that the combination of agents, or the system, needs to pursue. This means that biological systems are modeled using individual goals for agents, rather than system goals for the entire system. Achievement of system goals is assessed by inspecting the emergent behavior, for example at tissue level.

Agents in medical systems are usually independent, intelligent, and mobile entities that perform specific tasks one expect from a cell (or its components). These agents are programmed to adopt any of the phenotypes a cell can posses as its state. These phenotypes may include motile, quiescent, proliferative, and apoptotic cells.

3 Agent-Based Modeling of Angiogenesis

The multi-layered ABM developed to simulate the process of angiogenesis can be used to investigate the effects of various factors on angiogenesis. The case studies reported illustrate its use for identifying the role of pore size of a polymeric scaffold on implant vascularization. Most existing computational angiogenesis models consider solely the effect of soluble factors in the environment. A scaffold is desirable if the tissue volume to be replaced by engineered tissue is large.

3.1 Angiogenesis

Angiogenesis is the process of formation of new blood vessels from pre-existing vasculature [15]. It is a complex phenomenon that plays an important role in organ growth, healing and reproduction, and also in vascularizing tissue-engineered constructs. Abnormal angiogenesis can be initiated by diseases such as cancer and cardiovascular disease [9]. Consequently, angiogenesis has been the focus of a large number of experimental and theoretical studies that have provided information leading to successful outcomes such as recent anti-angiogenesis drugs.

Angiogenesis results in response to signalling of naturally occurring soluble factors, referred to as growth factors (GF), on endothelial cells (ECs), which are the cells lining inside of blood vessel capillaries. ECs become activated when they sense high concentrations of pro-angiogenic GFs. Through complex molecular and cellular mechanisms, some of the activated ECs differentiate into a special cell type, called tip cell. Tip cells start invading the surrounding tissue, following the direction of the GF gradient. Other factors such as insoluble ligand concentrations also play an important role in the process.

ECs behind the tip cell proliferate (cell division) and elongate (cell expansion) to fill the gap created between the motile tip cell and the host blood vessel. Combination of tip cell migration and EC proliferation and elongation results in formation of new capillaries. The newly formed capillaries connect to each other to make loops, a process called anastomosis, and once these loops are formed, blood flow starts in the new network. One of the features observed in sprouting capillaries is the persistence in branching and their migration direction: a new branch does not branch again immediately, and continues to move in the same direction for a time period known as its persistence.

Angiogenesis is a critical aspect of both tissue engineering and regenerative medicine as rebuilding tissues deeper than a few millimeters will not be successful without a healthy blood vessel network to supply nutrients, oxygen and other required factors [23]. Studying vascularization of synthetic scaffolds and implants can guide researchers in designing improved tissue engineering strategies that will lead to clinical success [7,20,10]. Computational models would enable researchers to study the effect of important factors while reducing the number of costly trial-and-error laboratory experiments.

3.2 Model Description

Our model [2] is implemented in Java and uses Repast (REcursive Porous Agent Simulation Toolkit), which is a Java-based open source agent modeling toolkit with features such as event scheduling procedures and visualization tools [22,18]. Repast includes packages of Java libraries that allow modelers to build simulation

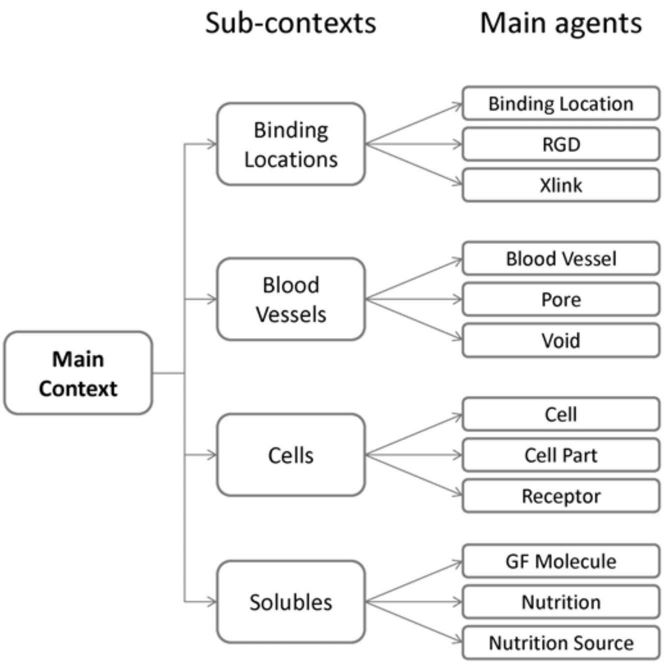


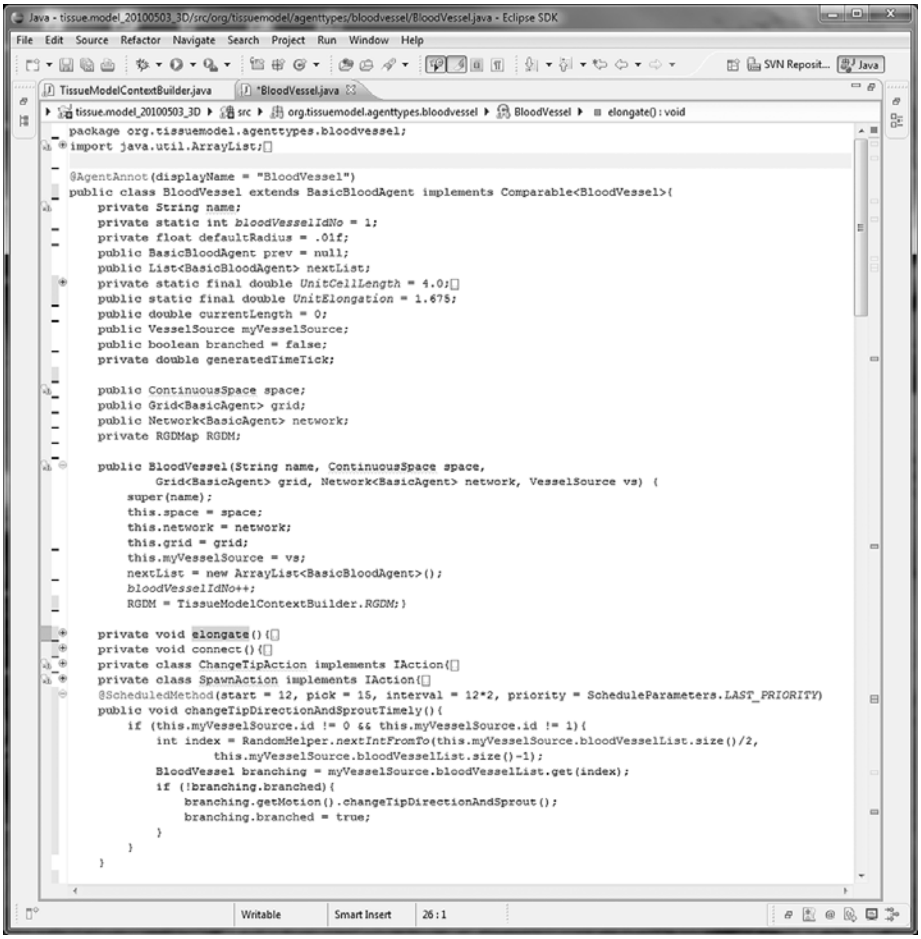
Fig. 1. Context structure of the model. The model includes four sub-contexts that contain different agent types. Sample agent types for each sub-context are shown.

environments, create agents in these environments, define relationships among the created agents, automatically collect data from simulation runs, and build user interfaces and displays in an easy fashion.

Repast is a widely used and complete Java-based platform [21], managed by the non-profit volunteer organization ROAD (Repast Organization for Architecture and Development). Repast has major benefits such as fast execution speed, inclusion of classes for network modeling, and ease of integration with the Eclipse IDE. Repast has the capability of developing multilevel models, utilizing the concept of context in designing the ABMs.

In Repast, different types of agents are organized by placing them in separate virtual buckets, referred to as *contexts* that enable hierarchical organization of the agents. Each context holds certain types of agents, and supports the control of relationships among the agents it holds through *projections*. A projection is a structure defined upon the agents of a context and its use enables the modeler to identify relationships among the agents in that context. The projections in Repast that are useful in modeling biomedical systems include network, grid, and continuous space projections.

The main context, which is the core object in Repast, holds all sub-contexts and their corresponding agents. Figure 1 shows the sub-contexts along with some of their main agent types. Our model structure includes four sub-contexts,


Java - tissue.model_20100503_3D/src/org/tissuemodel/agenttypes/bloodvessel/BloodVessel.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
SVN Reposit...
Java
TissueModelContextBuilder.java
*BloodVessel.java
tissue.model_20100503_3D src org.tissuemodel.agenttypes.bloodvessel BloodVessel elongate() : void
Writable
Smart Insert
26 : 1

```java
package org.tissuemodel.agenttypes.bloodvessel;
import java.util.ArrayList;□

@AgentAnnot(displayName = "BloodVessel")
public class BloodVessel extends BasicBloodAgent implements Comparable<BloodVessel>{
    private String name;
    private static int bloodVesselIdNo = 1;
    private float defaultRadius = .01f;
    public BasicBloodAgent prev = null;
    public List<BasicBloodAgent> nextList;
    private static final double UnitCellLength = 4.0;□
    public static final double UnitElongation = 1.675;
    public double currentLength = 0;
    public VesselSource myVesselSource;
    public boolean branched = false;
    private double generatedTimeTick;

    public ContinuousSpace space;
    public Grid<BasicAgent> grid;
    public Network<BasicAgent> network;
    private RGDMap RGDM;

    public BloodVessel(String name, ContinuousSpace space,
            Grid<BasicAgent> grid, Network<BasicAgent> network, VesselSource vs) {
        super(name);
        this.space = space;
        this.network = network;
        this.grid = grid;
        this.myVesselSource = vs;
        nextList = new ArrayList<BasicBloodAgent>();
        bloodVesselIdNo++;
        RGDM = TissueModelContextBuilder.RGDM;}

    private void elongate(){□
    private void connect(){□
    private class ChangeTipAction implements IAction{□
    private class SpawnAction implements IAction{□
    @ScheduledMethod(start = 12, pick = 15, interval = 12*2, priority = ScheduleParameters.LAST_PRIORITY)
    public void changeTipDirectionAndSproutTimely(){
        if (this.myVesselSource.id != 0 && this.myVesselSource.id != 1){
            int index = RandomHelper.nextIntFromTo(this.myVesselSource.bloodVesselList.size()/2,
                    this.myVesselSource.bloodVesselList.size()-1);
            BloodVessel branching = myVesselSource.bloodVesselList.get(index);
            if (!branching.branched){
                branching.getMotion().changeTipDirectionAndSprout();
                branching.branched = true;
            }
        }
    }
```

Fig. 2. Structure of BloodVessel class for creating EC agent objects. Only sample parts of the whole class are shown, including a number of instance variables, sample methods, and one scheduled method.

Binding Locations Context (for holding agents related to polymer molecules in the scaffold that ECs can attach to), Blood Vessel Context (that contains all the agents related to ECs), Cell Context (that includes all cell types other than blood vessel cells, and their parts and receptors), and Solubles Context (containing agents related to different soluble factors and the important soluble containers that hold the amount of each soluble at each grid point).

To illustrate the structure of agents, we have shown parts of an agent used in our model (Fig. 2). In this figure, selected parts of the BloodVessel class, which is the blueprint for creating BloodVessel agents, is depicted. This agent encapsulates a number of instance variables that determine the internal state of the agent. As an example, the currentLength variable, which is zero initially, holds the length of each EC agent. When an EC elongates, the value of this parameter is updated, and controlled not to exceed the maximum allowable length, L_{max}. Each EC agent holds information of its previous (parent) EC agent, and a list of the next BloodVessel agents, in prev and nextList variables, respectively. The constructor, creates the BloodVessel object, and adds the space, network, and grid projections to the object. A number of methods define the behaviors the agent is capable of performing, such as connect (for connecting to other EC agents), and elongate (for increasing the length of the agent, EC growth). The method changeTipDirectionAndSproutTimely is an special method as it includes an annotation before its definition. This type of scheduled methods are used in Repast to perform tasks on a timely basis, as defined in the annotation.

The model developed includes a number of rules that govern the behavior of cells. The agents created interact with a virtual representation of the physical environment. Two types of agents have been considered to represent the two different cell types. However, in simulation runs we have only activated the EC agents to observe angiogenesis. Agents representing ECs mimic the actions that ECs perform during angiogenesis, which include elongation, proliferation, tip cell migration, and anastomosis, as governed by the rule base. Based on these rules, first the neighborhood of an agent is defined, and then the agents perceive their local environment and act based on their internal state and according to their embedded internal logic.

Figure 3 shows the flow chart of the time-driven processes, their connectivity, and the main model parameters [2]. EC agents conduct their actions either based on conditions in the environment (event-driven actions) or randomly based on time intervals (time-driven random actions). EC agents are proactive, as their internal state affects their behavior, and are non-adaptive because they have a determinate set of rules that are not being updated or altered during the simulation. In our model, an EC agent interacts directly only with its two neighboring EC agents in a blood vessel branch. A tip cell agent is an exception as it can move in the pore areas of the scaffold and has the ability to connect to another EC agent if they are in immediate neighborhood of each other.

Several layers in the model have been utilized to enable flexibility in handling the agents and representing their environment and neighborhoods [2]. A rectangular *grid* layer, defined using the respective projection in Repast, enables storing all the environmental information, including soluble concentrations and the details of porous scaffold structure at different grid points. Grid layer facilitates calculating gradients of solubles. Agents can find their neighbors by using this grid layer. Grid positions are discrete integer-valued coordinates. In contrast, agents are located and act on a continuous real-valued *space* layer, defined using the same projection of Repast. This combination of layers provides the tools to handle agents positioning and provide them with the ability to access information by querying their neighborhood. In addition to these two layers for handling agent positions, we use a *network* layer to keep track of the relationship between agents. Using this layer, we create a network of the agents that are connected to each other through parent and child relationship, and we save

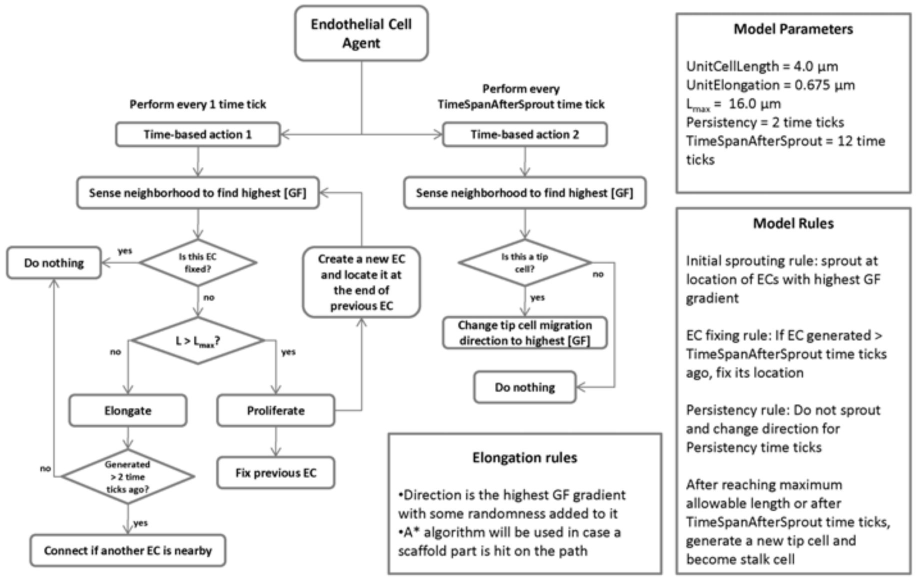

Endothelial Cell Agent
Perform every 1 time tick
Time-based action 1
Perform every TimeSpanAfterSprout time tick
Time-based action 2
Sense neighborhood to find highest [GF]
Do nothing
yes
Is this EC fixed?
no
Create a new EC and locate it at the end of previous EC
Is this a tip cell?
Change tip cell migration direction to highest [GF]
L > L_max?
Elongate
Proliferate
Fix previous EC
Generated > 2 time ticks ago?
Connect if another EC is nearby
Model Parameters
UnitCellLength = 4.0 µm
UnitElongation = 0.675 µm
L_max = 16.0 µm
Persistency = 2 time ticks
TimeSpanAfterSprout = 12 time ticks
Model Rules
Initial sprouting rule: sprout at location of ECs with highest GF gradient
EC fixing rule: If EC generated > TimeSpanAfterSprout time ticks ago, fix its location
Persistency rule: Do not sprout and change direction for Persistency time ticks
After reaching maximum allowable length or after TimeSpanAfterSprout time ticks, generate a new tip cell and become stalk cell
Elongation rules
•Direction is the highest GF gradient with some randomness added to it
•A* algorithm will be used in case a scaffold part is hit on the path

Fig. 3. Flowchart showing the main rules governing the behavior of tip cell and stalk cell ECs. Arrows indicate the connectivity of the rules.

these connection information to populate the number of ECs and sprouts in each of the branches at each time step in the simulation.

In our model, ECs are identified by two spherical nodes attached to each other with a connection in the network layer of the model. One of these nodes is designated to act for the EC itself, meaning that it is the agent assigned to represent that EC. The other node represents the parent EC of the current EC. The position of the EC node in the space layer is in fact the position of the front end of an EC. The rear end position of each EC is the location of its previous EC agent. As a result, the length of each EC would be the distance between its node and the node corresponding to its previous EC in the network layer. Using this abstraction method, we have been able to simply exhibit elongation of an EC by movement of its front node.

As a result of the time-based action 1 (Fig. 3), which happens at every time tick, tip cell node at each blood vessel sprout continues elongating until it reaches L_{max}. Once the length of an EC reaches L_{max}, proliferation occurs and another node is generated. The newly generated node becomes the leading node (tip cell) and continues elongation, while the previous node becomes fixed, which means it would not be able to move any further.

Fixed nodes can only become activated and create new leading nodes, causing branch formation and a new capillary vessel sprout, but they are not able to move themselves. Hence, blood vessels are able to change their direction either by proliferation and generation of new leading nodes or by branching. After a tip cell proliferates, the newly generated leading node has several options, including elongation, proliferation, generation of new sprouts (branching), or connecting with other blood vessels based on its proximity to other vessel segments (anastomosis).

Proliferation has been modeled using the following algorithm: A tip cell EC proliferates either when it's length reaches L_{max}, or after a certain amount of time (TimeSpanAfterSprout) has passed from its generation. To create a new EC, first a new node is created at the location of current node, and then the current node retreats by half of the length of the proliferating EC. This mechanism leads to creation of two ECs of equal size. The previous node will then become fixed, and the newly generated node will become the tip cell and continue elongation in the same direction as the previous tip cell.

We have utilized a modified version of the A* search algorithm [14] for leading node pathfinding [2]. The leading node searches its local environment to find the location with the "lowest cost" among all the possible neighboring locations, while avoiding collisions with the scaffold. This cost is inversely proportional to the magnitude of GF concentration gradient. However, if the candidate location happens to be part of the scaffold, then a node cannot move to that location. We modified the algorithm so that the leading node does not compute its whole migration path from the beginning to the end *a priori*. The modified algorithm determines the node's path at each time step based on new information generated by the simulation. This enables the nodes to consider the effect of dynamic changes in environment (such as variations in GF concentration by production or consumption) as a result of presence of different cell types, and saves significant computation resources without reducing the efficiency of the algorithm as one tip cell will not be able to follow the whole path due to proliferation and the randomness embedded in it.

As a result of time-based action 2 (Fig. 3), at each time tick a certain number of stalk ECs (stalk cells are the cells that are not at the tip of a sprout, represented by fixed nodes in our model) will be chosen randomly and sprouting will happen at their location. The direction of the new sprouts will be in the direction of highest GF gradient at the location of the selected stalk ECs. These nodes will not be able to change their migration direction without making a new sprout. To consider the persistency of the sprouting branches, we have included a persistency rule in the rule base. According to this rule, a branch does not change its growth direction for a certain time, specified in the model with a persistency time parameter. In addition, a newly created branch cannot sprout to create another branch for the same persistency time.

The scaffold is modeled to have circular pores of equal size [2]. The diameter and shape of the pores can be varied to investigate angiogenesis in different pore sizes and structures. In the control case, all the space is pore region and hence available to the ECs. Vessel networks are limited to grow within the pores only and the scaffold locations are unavailable to EC agents. GF concentration (C_{GF}) profile in 3D runs (Equation 1) is a function of y and z directions. $Y_{max}/2$ and $Z_{max}/2$ are the maximum concentration locations, chosen to be the center of the scaffold (middle of the two existing blood vessels), and ε is a small positive number to make the concentration at the initial locations greater than zero. A similar concentration profile have been used for 2D runs, which lack the z direction parameters. During simulation initialization, a random change of less than or equal to 0.01% of C_{GF} is added to the values of GF concentration to include stochastic behavior.

$$C_{GF}(y) = -(Y_{max}/2 - y)^2 + Y_{max}^2/4 - (Z_{max}/2 - z)^2 + Z_{max}^2/4 + \varepsilon \qquad (1)$$

In 2D runs, we have simulated angiogenesis occurring on a 800×800 μm surface located in the first quadrant of the coordinate system. We initially place two main blood vessels, located at $y = 100$ and $y = 700$ μm. In 3D runs, due to memory limitations resulting from the presence of another dimension, scaffold size is reduced to 400×400×400 μm and four initial blood vessels are located at $y = 50$ and $z = 25$ μm, $y = 350$ and $z = 25$ μm, $y = 50$ and $z = 375$ μm, and $y = 350$ and $z = 375$ μm. These initial vessels represent the host vasculature, and the scaffold is implanted between them. The implanted scaffold is vascularized via sprouting angiogenesis from the surrounding host vasculature. EC agents possess an initial speed when they are first created which varies based on the local availability of scaffold attachment proteins (ligands), and based on the natural logarithm of the number of ligands in their immediate neighborhood.

The model presented in this paper and most ABMs of biomedical systems display a close relationship between the characteristics of the agents and biomedical phenomena. The technology should be considered in the development of ABMs for biomedical systems:

1. Recognition of the independent biomedical entities such as distinct cell types in the biomedical system that perform the most basic actions required to fulfill a goal, as agent classes. It is desirable to design the ABM with minimum number of agent classes.
2. Determination of the main behaviors of these entities that are essential for fulfilling their goals, and abstracting them into a number of functions. These functions must be designed in such a way to grant the agent independence in deciding its actions based only on what it perceives from its environment and its own state.
3. Design of the internal logic that will enable agents to perform their actions. This will be a rule-base for each agent type and it may include quantitative relations. In biomedical applications, this logic can include only simple if-then-else rules, or may incorporate sophisticated rules including sets of dynamic partial differential equations that model the gene-protein networks governing a cell's behavior.
4. Building an environment in which the agents function and selecting the environmental variables that are important for performing the actions of each agent type. The environment must be designed to include these variables, and the agents must have functions that enable them to survey their environment, and to identify their neighbors and their states. The environment in

biomedical applications usually includes tissue or extracellular matrix, and the variables can include the concentration of soluble or insoluble biological factors, or the matrix properties. It is possible to represent elements in the environment by dedicated agent types, but this approach requires more memory compared to using the built-in data structures of the programming environment used.

5. Definition of communication protocols between agents and with the environment.

The level of detail included in the model in each of these steps, will affect the accuracy and applicability of the model. However, there would be a compromise between the level of detail (and hence model accuracy) and the computational and memory costs. One challenge involved in translating from a biological process to an ABM is to decide the level of detail that results in useful simulation results, while it is feasible to run the model on available hardware.

3.3 Computational Challenges

One difficulty in developing ABMs for simulating biological systems is the computational limitations encountered, specially when the number of agents increase exponentially over time. In 3D the number of grid points may increase by 100 times depending on the size of third dimension, and hence memory becomes a limiting factor. One solution for this problem was proposed [2] by designing the 3D pores in an external software, such as 3ds Max [5], identifying the pore surfaces using a triangular mesh and providing the information of all the vertices and their normals. In this work, we overcame the memory limitation by using the 64-bit versions of Java jdk, Eclipse, and Java 3D on a 64 bit Windows 7 Professional edition that enables addressing as high as 192GB to memory. For 3D simulation runs, we used a work-station with 24GB RAM, on an Intel Pentium i7 processor that enabled a maximum grid size of $400\times400\times400$ μm.

In addition, we have made changes to the agent structure in our model to address the challenges regarding high memory cost of the model. As an example, we used the built in ValueLayer structures of Repast to represent scaffold and pore parts of the hydrogel, instead of using agents which required more memory. Using parallel computing techniques is a desirable way to overcome more extensive computational requirements encountered in modeling larger scale MAS systems. A single computer with multiple-core CPUs and large amounts of memory can also be considered for parallel computing. If the computational load distribution is performed based on the scaffold section in which the agents act, then the issue of agents that are crossing boundaries should be addressed since they are leaving one processing core and go into the control of another core. Da-Jun et al. have developed one sample platform for simulating a large multicellular ensemble of bacterial populations [11]. Their framework is aimed at running an ABM on a cluster-based hardware, using Message Passing Interface.

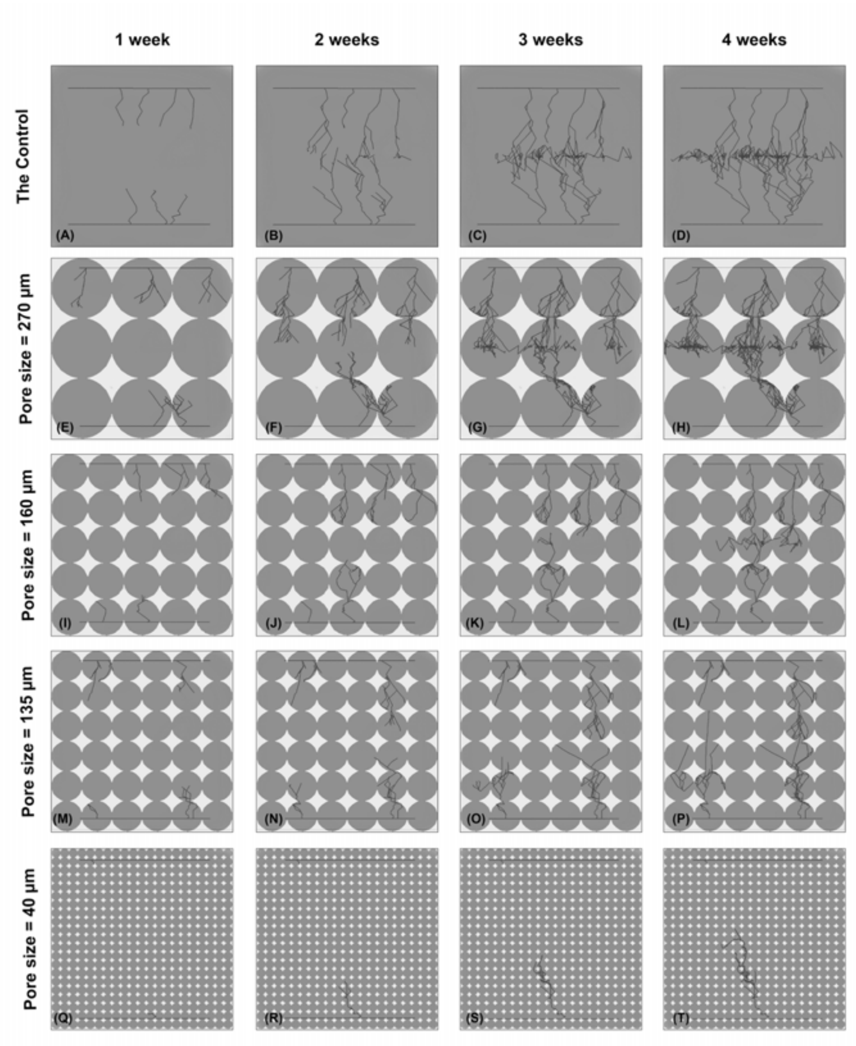


Fig. 4. Sample 2D simulation results for different cases after 1, 2, 3, and 4 weeks [3]. (A-D) The control, (E-H) Pore size = 270 μm, (I-L) Pore size = 160 μm, (M-P) Pore size = 135 μm, and (Q-T) Pore size = 40 μm.

4 Simulation Results

We have used the angiogenesis simulator developed to run case studies with different pore sizes in 2D. To adjust the speed of capillary growth, we first compared a number of simulation results with experimental results presented in [6], and we used this comparison to convert the simulation time ticks to real time in all subsequent case studies [3].

The parameters of interest are the time required for new capillaries to reach the center of scaffold and the number of sprouts after specific time (15 days in this case). Figure 4 depicts snapshots of 2D simulation runs for different cases at four consecutive times. A detailed description and evaluation of simulation results, along with comparison with published experimental data, can be found in [3]. Figure 5 shows a time series of the results of running the simulation in 3D for the control case, in which entire scaffold is available to ECs.

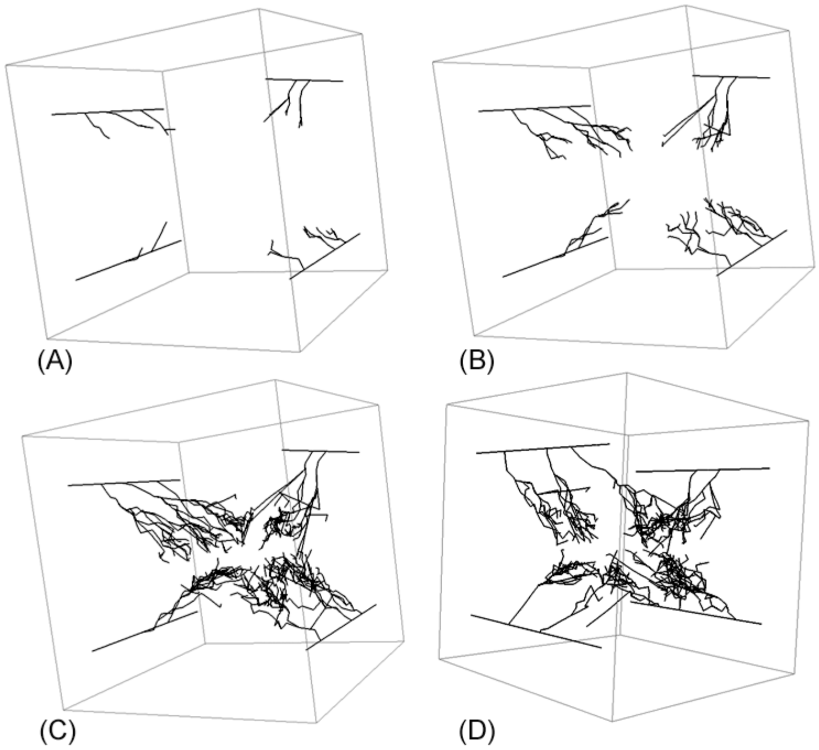


Fig. 5. Sample 3D simulation results for control case after 10 (A), 20 (B), and 30 (C, D) time steps. Picture (D) shows the same time step as picture (C) but from an opposite angle to illustrate the effect of 3D.

2D simulation results indicate formation of branching vascular networks from the host vessels. These branched blood vessels resemble 2D snapshots of blood vessel networks formed in in vivo angiogenesis in 2D assays [17]. 3D simulation results show how blood vessels fill the scaffold vascularizing it. The average time for blood vessels to reach the center of the scaffold is less than 10 days for the control case(Fig. 5).

The results of the simulation runs follow the same trends as experiments, indicating that the model is able to predict the basic behaviors expected from a developing blood vessel network. The rate of angiogenesis and the number of generated sprouts increase with increasing pore size, which supports the idea that larger pore sizes result in improved and faster angiogenesis in porous scaffolds. The simulation framework can also be used to predict the time for effective vascularization of the scaffold.

5 Conclusions

In this paper, we have shown how MAS can aid in addressing complex problems in biomedical engineering by offering computational tools required for the realistic simulation of biological systems comprised of a large number (approximately 100,000) of interacting biological cells. We have illustrated the advantages of using MAS concepts in simulating complex biomedical processes. The simulation framework developed is aimed at both understanding real world phenomena, and optimizing the way these phenomena take place. These models and simulations prove to be valuable specially when conducting experiments is costly and time consuming, which is the case with most biomedical experiments. The combination of computational and experimental results enables designing optimized biomaterials.

Acknowledgements. This paper is based upon work supported in part by the National Science Foundation (CBET-0731201) and the Veterans Administration.

References

1. An, G.: In silico experiments of existing and hypothetical cytokine-directed clinical trials using agent-based modeling. Crit. Care Med. 32, 2050–2060 (2004)
2. Artel, A.: Agent-based techniques in local, autonomous and adaptive decision-making. PhD dissertation: Illinois Institute of Technology (2010)
3. Artel, A., Mehdizadeh, H., Chiu, Y.-C., Brey, E.M., Cinar, A.: An Agent-Based Model for the Investigation of Neovascularization within Porous Scaffolds. Tissue Eng. Part A 17, 2133–2141 (2011)
4. Ausk, B.J., Gross, T.S., Srinivasan, S.: An agent based model for real-time signaling induced in osteocytic networks by mechanical stimuli. J. Biomech. 39 (2006)
5. Autodesk 3ds Max Products: 3D modeling, animation and rendering software, `http://usa.autodesk.com/3ds-max/`
6. Brey, E.M., McIntire, L.V., Johnston, C.M., Reece, G.P., Patrick, C.W.: Three-Dimensional, Quantitative Analysis of Desmin and Smooth Muscle Alpha Actin Expression During Angiogenesis. Ann. Biomed. Eng. 32, 1100–1107 (2004)
7. Brey, E.M., Uriel, S., Greisler, H.P., Patrick Jr., C.W., McIntire, L.V.: Therapeutic Neovascularization: Contributions from Bioengineering. Tissue Engineering 11, 567–584 (2005)
8. Brown, B.N., Price, I.M., Toapanta, F.R., Dealmeida, D.R., Wiley, C.A., Ross, T.M., Oury, T.D., Vodovotz, Y.: An agent-based model of inflammation and fibrosis following particulate exposure in the lung. Math. Biosci. 231, 186–196 (2011)

9. Carmeliet, P., Jain, R.K.: Angiogenesis in cancer and other diseases. Nature 14, 249–257 (2000)
10. Chiu, Y.C., Cheng, M.H., Uriel, S., Brey, E.M.: Materials for Engineering Vascularized Adipose Tissue. J. Tissue Viability 20, 37–48 (2011)
11. Da-Jun, T., Tang, F., Lee, T., Sarda, D., Krishnan, A., Goryachev, A.B.: Parallel Computing Platform for the Agent-Based Modeling of Multicellular Biological Systems. In: Liew, K.-M., Shen, H., See, S., Cai, W. (eds.) PDCAT 2004. LNCS, vol. 3320, pp. 5–8. Springer, Heidelberg (2004)
12. Dastani, M., van Riemsdijk, M.B., Winikoff, M.: Rich Goal Types in Agent Programming. In: Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011), pp. 405–412 (2011)
13. Grant, M.R., Mostov, K.E., Tlsty, T.D., Hunt, C.A.: Simulating properties of in vitro epithelial cell morphogenesis. PLoS Comput. Biol. 2, e129, Epub. (2006)
14. Hart, P.E., Nilsson, N.J., Raphael, B.: A Formal Basis for the Heuristic Determination of Minimum Cost Paths. IEEE Transactions on Systems Science and Cybernetics 4, 100–107 (1968)
15. Levine, H.A., Nilsen-Hamilton, M.: Angiogenesis - A biochemical/mathematical perspective. In: Tutorials in Mathematical Biosciences III; Cell Cycle, Proliferation and Cancer, pp. 23–76 (2006)
16. Liu, G., Qutub, A.A., Vempati, P., Mac Gabhann, F., Popel, A.S.: Module-based multiscale simulation of angiogenesis in skeletal muscle. Theor. Biol. Med. Model. 8 (2011)
17. Muthukkaruppan, V.R., Kubai, L., Auerbach, R.: Tumor-induced neovascularization in the mouse eye. J. Natl. Cancer Inst. 69, 699–708 (1982)
18. North, M.J., Collier, N.T., Vos, J.R.: Experiences Creating Three Implementations of the Repast Agent Modeling Toolkit. ACM Transactions on Modeling and Computer Simulation 16, 1–25 (2006)
19. O'Neil, C.A., Sattenspiel, L.: Agent-based modeling of the spread of the 1918-1919 flu in three Canadian fur trading communities. Am. J. Hum. Biol. 22, 757–767 (2010)
20. Papavasiliou, G., Cheng, M.H., Brey, E.M.: Strategies for Vascularization of Polymer Scaffolds. J. Investig. Med. 58, 834–844 (2010)
21. Railsback, S.F., Lytinen, S.L., Jackson, S.K.: Agent-based Simulation Platforms. Review and Development Recommendations Simulation 82, 609–623 (2006)
22. Repast Home Page: Repast Organization for Architecture and Design, Chicago, IL, http://repast.sourceforge.net/
23. Rouwkema, J., Rivron, N.C., van Blitterswijk, C.A.: Vascularization in tissue engineering. Trends Biotechnol. 26, 434–441 (2008)
24. Thorne, B.C., Bailey, A.M., DeSimone, D.W., Peirce, S.M.: Agent-based modeling of multicell morphogenic processes during development. Birth Defects Res. C. Embryo. Today. 81, 344–353 (2007)
25. Thorne, B.C., Bailey, A.M., Peirce, S.M.: Combining experiments with multi-cell agent-based modeling to study biological tissue patterning. Brief. Bioinform. 8, 245–257 (2007)
26. van Riemsdijk, M.B., Dastani, M., Winikoff, M.: Goals in agent systems: A unifying framework. In: Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008), pp. 713–720 (2008)
27. Zhang, L., Chen, L.L., Deisboeck, T.S.: Multi-scale, multi-resolution brain cancer modeling. Math. Comput. Simul. 79, 2021–2035 (2009)

Group Abstraction for Large-Scale Agent-Based Social Diffusion Models with Unaffected Agents

Alexei Sharpanskykh and Jan Treur

Vrije Universiteit Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
{sharp,treur}@few.vu.nl
http://www.few.vu.nl/~{sharp,treur}

Abstract. In this paper an approach is proposed to handle complex dynamics of large-scale multi-agents systems modelling social diffusion processes. A particular type of systems is considered, in which some agents (e.g., leaders) are not open to influence by the other agents. Based on local properties characterising the dynamics of individual agents and their interactions, groups and properties of the dynamics of these groups are identified. To determine such dynamic group properties two abstraction methods are proposed: determining group equilibrium states and approximation of group processes by weighted averaging of the interactions within the group. This enables simulation of the group dynamics at a more abstract level by considering groups as single entities substituting a large number of interacting agents. In this way the scalability of large-scale simulation can be improved significantly. Computational properties of the developed approach are addressed in the paper. The approach is illustrated for a collective decision making model with different types of topology, which may occur in social systems.

Keywords: group dynamics, model abstraction, social diffusion, large-scale agent-based simulation.

1 Introduction

Social diffusion models describe spread and changes of states or attitudes in a group or community of agents under the impact of social interaction. Such models have been extensively used to study diverse social processes, such as dynamics of social power [5], polarization of opinions of individuals in a group [11, 3], and spread of innovation [13].

In many existing social diffusion models it is assumed that each agent changes its state (e.g., an opinion) continuously, under influence of other agents. However, in real systems actors may exist, which for some reason are not open to change, for example, because they are not willing or able to change their state. Specific examples of such actors are autocratic leaders or persons with no control over a given state (e.g., related to ethnicity or gender) [11]. Such agents may affect other agents, by continuously propagating their state to them, and as they are not affected themselves in the end have much effect on the group's state, as happens, for example, for strong leader figures. In this paper social diffusion in large-scale social systems with such unaffected agents is considered from an agent-based perspective.

Although the local behaviors of each agent may be simple, the global patterns that emerge from interaction between the agents in a large-scale social diffusion system are far from trivial. Such patterns are difficult to infer directly from the local dynamic properties of the agents. A high computational complexity of such large-scale multi-agent systems hinders automated analysis of such systems by systematic simulation and verification.

In this paper an approach is proposed to handle complex dynamics of large-scale agent-based social diffusion models by using abstraction methods. The approach is based on identifying groups of interacting agents with similar states (e.g., opinions on an issue) in a society of agents. The idea is that an approximate form of simulation is obtained by using such groups as single entities representing abstractions of large numbers of interacting agents. In such a way the scalability of a large-scale multi-agent simulation can be improved. The obtained abstracted process provides an approximation with a behavioural error that can be estimated.

To determine global emerging properties of groups based on local properties of the group members, two group abstraction methods are proposed. In the first method relative degrees of importance of the agents in a group are determined. The degree of importance of an agent is an estimation of the strength of the agent's influence on the group. The aggregated state of the group is determined as the weighted average of the states of the group members with the weights defined by the relative degrees of importance of the members. In the paper this method is called *abstraction by weighted averaging*. The second abstraction method used is based on identifying an equilibrium state of a group by a standard procedure. In the paper this method is called *equilibrium-based abstraction*.

The proposed group abstraction approaches are illustrated by a case of a collective decision making model for a number of scenarios with two different topologies, which may exist in real social systems. The approximation errors and time complexity of the proposed abstraction methods applied for this case are discussed.

The paper is organized as follows. An agent-based collective decision making model used as a case is described in Section 2. The proposed methods for group abstraction are explained in Section 3. Some simulation results are discussed in Section 4. In Section 5 the proposed abstraction methods applied to the collective decision making model are evaluated. Related literature is discussed in Section 6. Section 7 concludes the paper.

2 A Collective Decision Making Model

In the model collective decision making is specified as the process of social diffusion of opinions of agents on decision options. The agents are assumed to consider two different decision options *s1* and *s2* for one issue (e.g., two exits of a burning building).

In most existing social diffusion models (e.g., [11, 12, 6]), opinions of agents are represented by binary variables, which reflect the opposite attitudes of agents towards an issue. The choice for binary variables is well motivated for models, which focus on attitudes of agents towards highly salient events, for which strong opinions are common (e.g., in voting). However, continuous variables are suited better than binary variables for representing doubts of agents, e.g., when they are situated in uncertain environments with scarce information. Furthermore, the change of the agent's opinion to the opposite one occurs gradually, through a number of phases [10]. This can be captured better by a continuous variable than by a binary variable. Similarly to [7], the opinions of an agent in the model used here are described by continuous variables within the range $[0,1]$. These variables reflect the degrees of support of an agent for the decision options $s1$ and $s2$.

The initial values of the opinions of the agents on both options are uniformly distributed in the interval $[0,1]$. By interaction the agents start to influence each other's opinions. The strength of social influence of an agent i on another agent j is determined by parameter $\gamma_{i,j}$ within the range $[0, 1]$. This parameter may be refined, e.g., by distinguishing expressiveness of the sender (agent i), openness of the receiver (agent j), the strength of the channel between i and j [8]. This parameter may also refer to a distance between i and j in 'social' space. For simplicity $\gamma_{i,j}$ will be used without refinement.

It is assumed that agents interact synchronously with each other: all states of the agents are updated in parallel. As stated in [12] the dynamics of group interaction is captured more accurately when many individuals are allowed to interact simultaneously than by pairwise interaction.

The strength of the social influence on agent i with respect to decision option s at time t is determined by:

$$\delta_{s,i}(t) = \sum_{j\neq i} \gamma_{j,i}(q_{s,j}(t) - q_{s,i}(t))/\sum_{j\neq i} \gamma_{j,I} \qquad \text{when } \sum_{k\neq i} \gamma_{k,i} \neq 0$$

$$\delta_{s,i}(t) = 0 \qquad \text{when } \sum_{k\neq i} \gamma_{k,i} = 0$$

Here $q_{s,j}(t)$ the strength of support of agent j of decision option s. The update of the strength of support of agent i for s is determined by:

$$q_{s,i}(t+\Delta t) = q_{s,i}(t) + \eta_i\, \delta_{s,i}(t)\Delta t$$

Here η_i is an agent-dependent parameter within the range $[0,1]$, which determines how fast the agent adjusts to the opinion of other agents.

First an initial consolidation phase takes place during the interval $[0, t_{end\_init}]$, in which the agents exchange opinions on the options. After this phase the whole population of agents is divided into two groups $G1$ and $G2$ depending on which from two options $s1$ or $s2$ is preferred:

$$G1 = \{i \mid q_{s1,i}(t_{end\_init}) \geq q_{s2,i}(t_{end\_init})\}$$
$$G2 = \{i \mid q_{s2,i}(t_{end\_init}) > q_{s1,i}(t_{end\_init})\}.$$

Each group can be viewed as a connected directed graph $G=<V, E>$ with a set of vertices V representing agents and a set of directed edges E representing influence relations between the agents. It is assumed that there are less interactions between members of different groups than between members within a group. This assumption is partially supported by social studies [3].

Definition
A subset S of G is called *isolated from impact by others* if it is nonempty and not equal to G and for all agents $i \in S$ and $j \notin S$ it holds $\gamma_{j,i} = 0$.

In the paper scenarios will be addressed based on the following topologies of groups:

(a) There is exactly one subset isolated from impact by others, and this is a singleton $\{i\}$- an agent, which is not willing or able to change its state (for an example see Fig. 1, left).
(b) There are exactly two subsets isolated from impact by others, and these are singletons $\{i\}$ and $\{j\}$ – two agents with different states, which are not willing or able to change their states; e.g., two conflicting dogmatic leaders (for an example see Fig. 1, right).

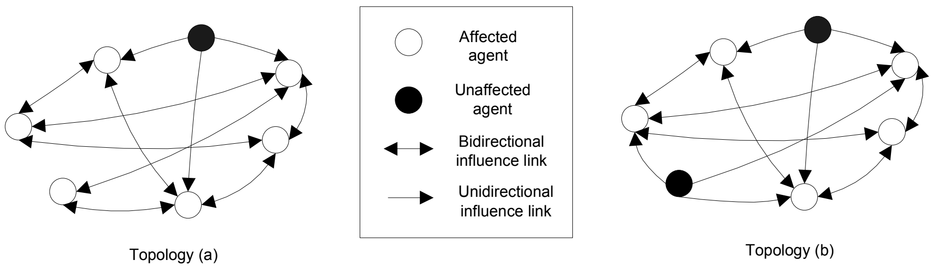


Fig. 1. Examples of topologies of groups considered in the paper

The topology of the network considered in the paper is random and dense.

Every now and then members of a group receive information from diverse external sources via peer-to-peer communication. External sources comprise agents from other groups, connected according to the network topology, and environmental information sources, connected dynamically and randomly to the agents from the network at each time point. The degree of influence of external source k (e.g., an agent from another group) on a group member i is represented by parameter $\gamma_{k,i.}$ Based on the states of k and i concerning option s, agent i updates its state as follows:

$$q_{s,i}(t+\Delta t) = q_{s,i}(t) + \eta_i \gamma_{j,i}(q_{s,k}(t) - q_{s,i}(t))\Delta t$$

If after interaction with an external source, agent i from group *G1* supporting option *s1* changes its preference from *s1* to *s2*, and $q_{s2,i}(t) - q_{s1,i}(t) > threshold$, then an agent is considered to leave *G1* and become a member of *G2* supporting *s2*. In the scenarios considered in the paper *threshold=0.3*.

3 Two Methods for Group Abstraction

To model the dynamics of abstracted states of a group two group abstraction methods are proposed in this section.

3.1 Abstraction by Weighted Averaging

The first method introduced is based on an estimation of an aggregated group state by determining the contributions of each group member to the group state. It is assumed that the contribution of an agent is in proportion to the strength of influence of the agent on the other group members. An agent may influence another agent directly or indirectly through other agents. In this direct case the strength of this (first-order) influence of i on j can be estimated by $\gamma_{i,j.}$ If i influences k through j, the strength of (second-order) indirect influence of i on k via j is estimated as $\gamma_{i,j}\ \gamma_{j,k.}$, and in the total second-order influence is estimated as $\sum_{j\neq i, j\neq k} \gamma_{i,j}\ \gamma_{j,k.}$. In the general case, the higher order strengths of influence of an agent on any other agent can be calculated recursively.

Thus, for each agent a network of influence can be identified, through which an agent exerts influence and is influenced by other agents. In such a network the degree of importance of an agent i (doi_i) on the group is calculated as follows:

$$doi_i = \sum_{i\neq j} \gamma_{i,j}(1 + \sum_{k\neq i,\, k\neq j} \gamma_{j,k}(1+\ldots))/(1+\sum_{i\neq j} \gamma_{j,i}(1 + \sum_{k\neq i,\, k\neq j} \gamma_{k,j}(1+\ldots)))$$

The denominator contains the term 1 to ensure that it is not equal to 0 for the agents isolated from impact by others, e.g., as in topologies (a) and (b). The precision of estimation of the group state depends on the number of hops in a network of influence for which indirect influences are calculated. However, the more hops are taken the more intensive computation is required for abstraction by this method. In this paper two hops in a network of influence are used. In the single-hop variant of the method (called *first-order weighted averaging*) doi_i is calculated as:

$$doi_i = \sum_{i\neq j} \gamma_{i,j} /(1+\sum_{i\neq j} \gamma_{j,i})$$

The two-hop variant (called *second-order weighted averaging*) doi_i is:

$$doi_i = \sum_{i\neq j} \gamma_{i,j}(1 + \sum_{k\neq i, k\neq j} \gamma_{j,k})/(1+\sum_{i\neq j} \gamma_{j,i}(1 + \sum_{k\neq i,\, k\neq j} \gamma_{k,j}))$$

Initially and after each interaction of an agent from group G with an external agent, the aggregated state of group G for option s is estimated by the following weighted average:

$$q_{s,G}(t+\Delta t) = \sum_{i\in G} doi_i\ q_{s,i}(t)/ \sum_{i\in G} doi_i$$

This state represents a common opinion of all agents in the group for decision option s. It persists until a new interaction with an external agent occurs. Then, the formula for $q_{s,G}(t+\Delta t)$ is applied again. The weighted averaging method can be used for abstraction of groups with both types of topologies (a) and (b) described in Section 2.

3.2 Abstraction by Determining Equilibria

For the cases where one or more of the agents i are not affected (agents i with $\gamma_{j,i} = 0$ for all j), the equilibria for other agents do not depend on their initial values and the standard approach (solving the equilibrium equations) using the differential equations can be used. Note that if $\gamma_{j,i} = 0$ for all j then the value for i will be in an equilibrium right from the start, since no impact of other group members occurs. Therefore a group equilibrium with common value can only concern the initial value $q_{s,i}(0)$ for such an agent *i*. For one such an agent in the group this indeed takes place. Consider an example of a group with topology (a) comprising 50 agents, under the condition that no external messages are provided to the group (see Fig.2). The decision states of the agents are initialized randomly. Over time the decision states of the agents converge gradually to the decision state of the unaffected agent (Fig.2, right).

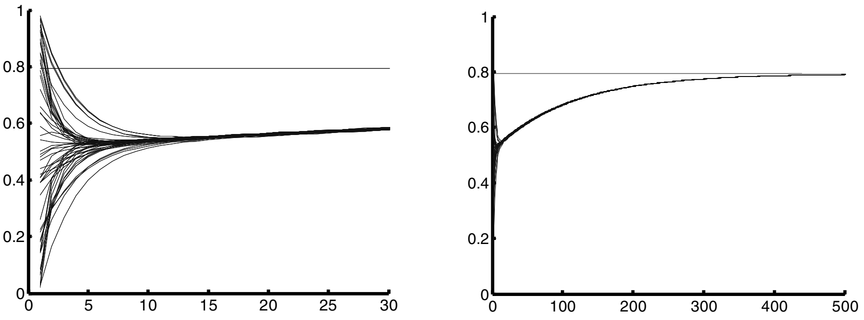


Fig. 2. Convergence of the decision states of 50 agents in a group with topology (a) for the time period [0, 30] (left) and the time period [0, 500]; no external messages are provided to the group

When two or more of such agents occur, with different initial states, then apparently a common equilibrium value is not possible, and the group will end up in a divided equilibrium situation. Consider an example of a group comprising 50 agents with two unaffected agents with different decision states (see Fig. 3). The decision states of the agents are initialized randomly. Over time the affected agents move towards an equilibrium state between the decision states of the two unaffected agents (see Fig.3, right). In the simulation for which the results are in Figures 2 and 3 the strength of social influence of both unaffected and affected agents on other affected agents was taken from the uniform distribution in the interval]*0,1*].

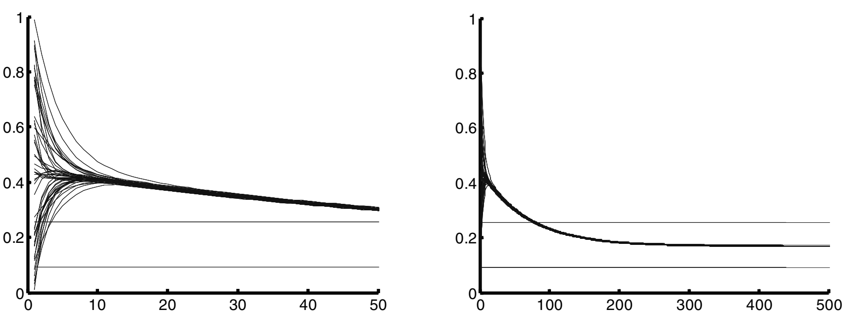


Fig. 3. Convergence of the decision states of 50 agents in a group with topology (b) for the time period [0, 50] (left) and the time period [0, 500]; no external messages are provided to the group

These cases can be analyzed in a direct, standard manner using the differential equations as follows. Take the set of agents that are not affected:

$$S_0 = \{ i \mid \sum_{k\neq i} \gamma_{k,i} = 0\}$$

Then the differential equations are

$$q_{s,i}(t+\Delta t) = q_{s,i}(t) + \eta_i \; [\, \sum_{j\neq i} \gamma_{j,i}(q_{s,j}(t) - q_{s,i}(t))/\sum_{k\neq i} \gamma_{k,i} \,]\, \Delta t \qquad \text{for } i \notin S_0$$

$$q_{s,i}(t+\Delta t) = q_{s,i}(t) \qquad \text{for } i \in S_0$$

For all agents $i\in S_0$ the equilibrium value $q_{s,i}$ is the initial value $q_{s,i}(0)$. This value can be used in the equations for the agents $i \notin S_0$, thus obtaining for all $i \notin S_0$:

$$\sum_{j\notin S_0,\, j\neq i} \gamma_{j,i}(q_{s,j} - q_{s,i})/\sum_{k\neq i} \gamma_{k,i} + \sum_{j\in S_0} \gamma_{j,i}(q_{s,j}(0) - q_{s,i})/\sum_{k\neq i} \gamma_{k,i} = 0$$

This can be rewritten into the following system of linear equations in $q_{s,k}$ for $k \notin S_0$:

$$q_{s,i} - \sum_{j\notin S_0,\, j\neq i} (\gamma_{j,i}/\sum_{k\neq i} \gamma_{k,i})\, q_{s,j} = \sum_{j\in S_0} (\gamma_{j,i}/\sum_{k\neq i} \gamma_{k,i})\, q_{s,j}(0)$$

This can be expressed in matrix form as $\mathbf{B}\boldsymbol{q} = \boldsymbol{c}$, with $\boldsymbol{q}$ the vector $(q_{s,i})_{\, i\notin S_0}$, and $\boldsymbol{c}$ the vector $(\sum_{j\in S_0} (\gamma_{j,i}/\sum_{k\neq i} \gamma_{k,i})\, q_{s,j}(0))_{\, i\notin S_0}$. The matrix **B** has only 1 as diagonal entries, and negative values elsewhere. Taking into account determinant $\mathbf{det(B)} \neq 0$, this system is solvable in a unique manner.

4 Simulation

The methods described in Section 3 were implemented in Matlab. Simulation time was 1030 with the initial stabilization interval [0, 30]. For each simulation setting 50 iterations were executed. The number of agents was varied across simulation runs: 50, 100, 200 and 500. The initial states of each agent for the strengths of support for the two decision options *s1* and *s2* were uniformly distributed in the interval [*0,1*].

In addition to the agents external sources were used, which number was 10 times less than the number of agents. The average time between two subsequent messages

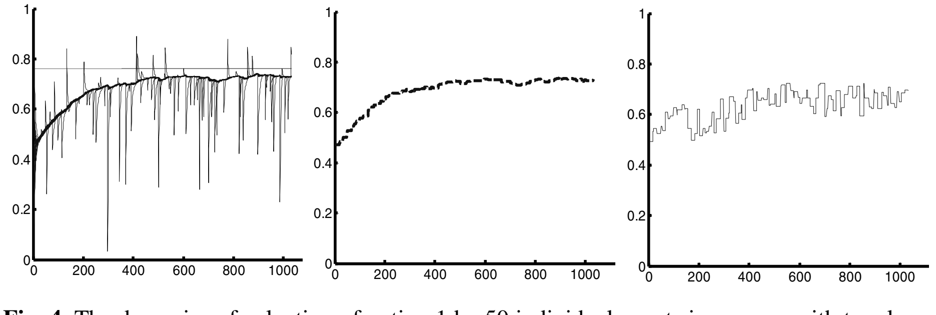


Fig. 4. The dynamics of valuation of option 1 by 50 individual agents in a group with topology (a) (left) and the abstraction of the group dynamics obtained by the equilibrium-based method (center) and by the weighted averaging 2 (right); the average time between messages is 10.

provided by each external source to a randomly chosen agent was varied across simulation runs: 1, 2, 5, and 10. Each average time value can also be interpreted as a ratio of the time scale of the group's internal dynamics to the time scale of the external dynamics. The impact of these ratios on approximation errors was investigated.

The simulation was performed for both types of topology described in Section 2. The parameters γ and η of the agents were taken from the uniform distribution in the interval $(0,1]$. For topology (a) all values $\gamma_{j,i}$ for a randomly chosen agent *i* were set to *0*. For topology (b) all values of $\gamma_{j,i}$ and $\gamma_{m,k}$ for two randomly chosen agents *i* and *k* were set to *0*.

In the simulation the first and second-order weighted averaging methods and the equilibrium-based method were used for abstraction of the model with both types of topologies.

Some of the simulation results for topologies (a) and (b) are presented respectively in Figures 4 and 5. The peaks in the graphs indicate incoming messages from external sources. As can be seen from the both figures, after receiving each message the group quickly reaches a new stable state.

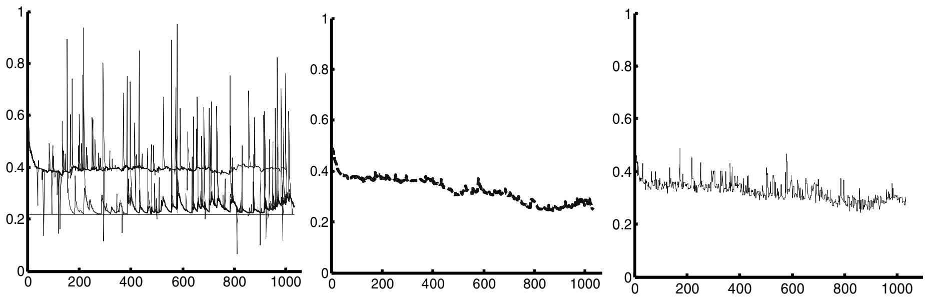


Fig. 5. The dynamics of valuation of option 1 by 50 individual agents in a group with topology (b) (left) and the abstraction of the group dynamics obtained by the equilibrium-based method (center) and by the weighted averaging 2 (right); the average time between messages is 2.

Since a divided equilibrium situation occurs in the group with topology (b) (Fig.5), the abstraction of the group dynamics for this topology is less precise than for topology (a) (Fig.4), in which the group is driven towards a single equilibrium state. A detailed evaluation of efficiency and quality of the proposed abstraction methods is considered in the following Section 5.

5 Evaluation of the Two Abstraction Methods

In this section the time complexity and approximation errors are considered.

5.1 Time Complexity Results

The mean time complexity for the original model from Section 2 and for the proposed abstraction methods is provided in Tables 1 and 2.

Table 1. Mean simulation time in seconds for the original and abstracted models for 50 and 100 agents agents

# of agents	**50**				**100**			
Average time between messages	**1**	**2**	**5**	**10**	**1**	**2**	**5**	**10**
Original model	5.87	5.46	4.98	4.72	23.67	22.7	20.5	19.25
Equilibrium-based abstraction (a)	0.27	0.24	0.22	0.21	0.9	0.83	0.78	0.76
Equilibrium-based abstraction (b)	0.29	0.25	0.22	0.21	1.02	0.90	0.81	0.78
Abstraction by weighted averaging 1	0.25	0.22	0.20	0.20	0.88	0.83	0.78	0.76
Abstraction by weighted averaging 2	0.27	0.24	0.21	0.20	0.96	0.88	0.80	0.78

Table 2. Mean simulation time in seconds for the original and abstracted models for 200 and 500 agents

# of agents	**200**				**500**			
Average time between messages	**1**	**2**	**5**	**10**	**1**	**2**	**5**	**10**
Original model	96.4	93.5	87.9	82.7	383.7	383.2	365.3	350.8
Equilibrium-based abstraction (b)	3.60	3.24	3.03	2.96	13.8	13.1	11.7	11.5
Equilibrium-based abstraction (c)	4.08	3.58	3.16	3.05	15.7	14.0	12.1	11.8
Abstraction by weighted averaging 1	3.32	3.18	3.05	3.01	12.9	12.3	11.9	11.1
Abstraction by weighted averaging 2	3.71	3.39	3.14	3.06	14.5	13.4	12.4	12.2

The variances of these results are very low (of the order of 10^{-5}). Since the same mechanisms of abstraction by weighted averaging are applied for both topologies (a) and (b), the mean simulation time of the abstracted models based on the weighted averaging methods is the same for all these topologies. The developed abstraction methods increase the computational efficiency of the simulation significantly. The acceleration factor grows with the number of agents: for smaller numbers (around *50*) it is of the order *20* to *25*, for larger numbers (around *500*) it grows to the order of *25* to *33*.

The fastest simulation models are obtained by the abstraction by first-order weighted averaging. The slowest are the models obtained by the equilibrium-based abstraction. However, as one can see from Tables 1 and 2, the ratio of the simulation time of the slowest to the fastest abstraction method is less than *1.3* for all cases. The impact of the number of messages on the simulation time is stronger for the equilibrium-based method than for the weighted averaging methods. This is because (large) systems of linear equations need to be solved in the former methods every

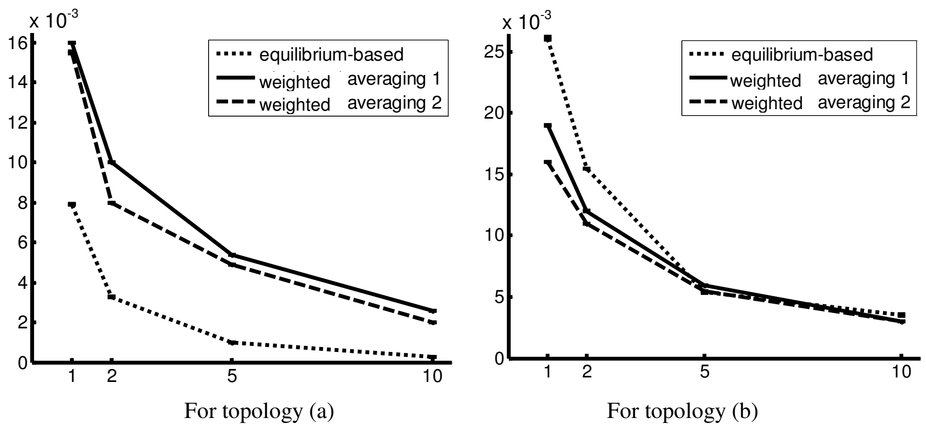


Fig. 6. Mean approximation errors for the proposed abstraction methods for 50 agents; the horizontal axis is the average time between messages.

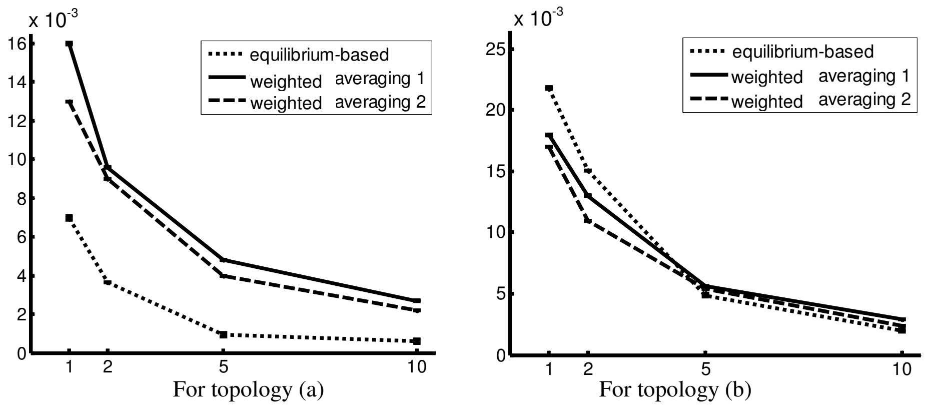


Fig. 7. Mean approximation errors for the proposed abstraction methods for 100 agents; the horizontal axis is the average time between messages.

time when the structure of a group changes. The greatest decrease of the acceleration rate for the equilibrium-based method for the settings considered in the paper is of the order of *1.4*.

5.2 Approximation Errors

The error of approximation of the original model by a group abstraction method is defined as

$$\sum_{t\in[31,\ 1031]} (|G1^{o,t} \cup G1^{a,t}| - |G1^{o,t} \cap G1^{a,t}|)/1000,$$

where $G1^{o,t}$ is the group comprising the agents supporting decision option *s1* at time point *t* according to the original model, and $G1^{a,t}$ is the group of the agents supporting *s1* at time point *t* according to the abstracted model.

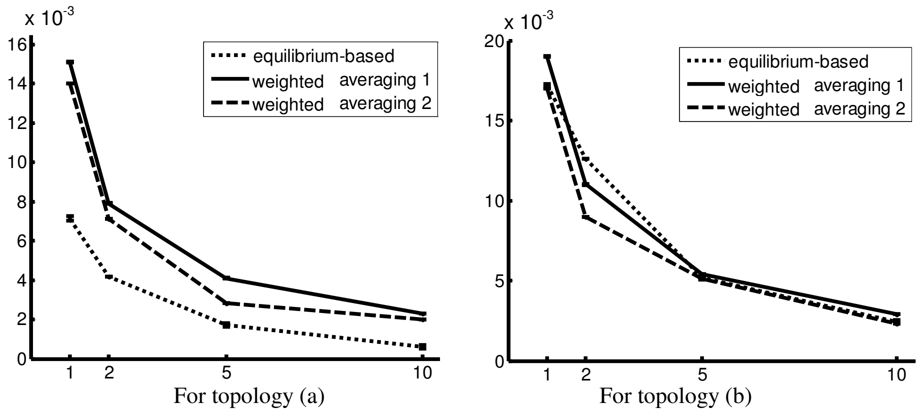


Fig. 8. Mean approximation errors for the proposed abstraction methods for 200 agents; the horizontal axis is the average time between messages.

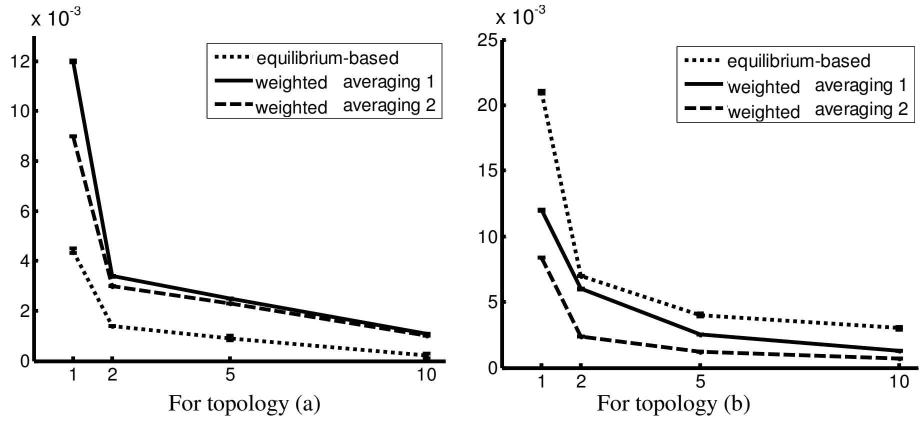


Fig. 9. Mean approximation errors for the proposed abstraction methods for 500 agents; the horizontal axis is the average time between messages.

A comparison of the mean approximation errors for the proposed abstraction methods is provided in Figures 6-9. The variances of the errors are low (of the order 10^{-6}); they are depicted by small error bars in the figures.

As can be seen from Figures 6-9, both the equilibrium-based and weighted averaging methods are sensitive to the average time between messages from external sources. In particular, for topology (a), when the average time is high (10) the error of the equilibrium-based abstraction is very low: in average less than one agent is placed in a wrong group for 1000 time points. However, when the external world interacts with each group every time point, the error of the equilibrium-based abstraction grows significantly: 11 times in the worst case for topology (a). Nevertheless, the maximal error of the equilibrium-based abstraction is still rather low: $7.9*10^{-3}$ (meaning that for topology (a) less than *8* agents are placed in a wrong group for 1000 time points). As can be seen from the results, the abstraction methods by weighted averaging perform significantly worse than the equilibrium-based abstraction for the topology (a).

For topology (b) the rate of stabilization of the system is generally slower than for topology (a). Furthermore, the agents of the system do not converge to the same state. For these reasons the approximation errors for (b) are higher than for (a). Surprisingly, although the exact equilibrium state of the system can be determined by the equilibrium-based abstraction method, this method performs comparably to or even worse than the weighted averaging methods for topology (b). However, the greater the average time between the messages, the better the equilibrium-based method performs. This can be explained by the dynamics of convergence of the group to a stable state: the greater the time between the messages, the more closely the group approaches an equilibrium state, thus the smaller the approximation error of the equilibrium-based method. In particular, for the average time 10 the equilibrium-based method performs in average better than the weighted averaging methods.

The weighted averaging methods are less sensitive to the rate of convergence, but also less precise, as they are based on an approximation of the group's emergent property. The approximation error grows with frequency of external messages, since every message results into a decision re-evaluation by the agents, and thus the error accumulates. For topology (b) the group approaches its equilibrium states slowly, thus the equilibrium-based method is less suitable.

The rate of convergence is also the reason why the approximation error of the equilibrium-based method is less when the ratio of the time scale of the external world dynamics to the time scale of the group's internal dynamics is higher. The greater the difference in the scales, the closer a group approaches an equilibrium, which can be calculated precisely using the equilibrium-based method. The error depends on how often an equilibrium state of a group is disturbed by external messages and on how quickly the group reaches a new equilibrium.

As expected, the abstraction by second-order weighted averaging is more precise than the abstraction by first-order weighted averaging. The difference in precision between first- and second-order weighted averaging depends on the density of connections in the topology of a group: in general, the higher the density, the less the error difference between both variants. This is because the density determines how many direct neighbors an agent has, and thus, how many agents are influenced directly by one-hop message propagation of new information. The more densely a graph is connected, the more agents in a group new information reaches by one-hop propagation, and the more fully the new group's state can be captured by first-order weighted averaging. The less the graph's density, the more information about the group dynamics each additional hop provides. In a sparsely connected graph, one-hope propagation reaches only a small number of agents, thus, only partial information about the group dynamics can be extracted by first-order weighted averaging. In this case the difference between first- and second-order weighted averaging may be significant. For the experiments considered in this paper densely connected groups were used. Furthermore, as can be seen from the results, the abstraction by weighted averaging becomes more precise with the increase of the number of agents.

6 Discussion

Social diffusion models have been studied extensively [3, 4, 9, 11, 12]. A common research question of these studies is about the existence of equilibrium states of a model for different topologies. In contrast to the continuous model considered in the paper, most of other studies consider binary, threshold-based models. Among a few exceptions are the studies described in [7] and [4], which focus on the behavioral abstraction of continuous social diffusion models.

Currently several techniques for abstraction of models based on hybrid automata [1] and differential equations [2] exist. However, such approaches can be applied efficiently for systems described by sparse matrixes. Social diffusion models represent tightly connected systems, which do not allow a significant reduction of the state space using such techniques. In particular, a previous study showed that common model reduction techniques such as balanced truncation [2] do not allow decreasing the rank of the matrix describing the model from Section 2.

7 Conclusions

In the paper an approach is proposed to handle complex dynamics of large-scale agent-based social diffusion models. On the one hand this approach allows identifying global, emergent properties of groups of agents. On the other hand, it enables a significant increase of the computational efficiency of automated analysis of large-scale social diffusion models (up to 33 times for larger numbers of agents).

The approach comprises two methods dedicated for abstraction of a variety of topologies of social groups. In particular, the equilibrium-based method is well suited for models with topologies with one unaffected agent. The higher the ratio of the time scale of the external world dynamics to the time scale of the group's internal dynamics, the less the approximation error of the equilibrium-based method. The high ratio of the scales is also required to reduce the approximation error of the equilibrium-based abstraction of models with topologies with two isolated agents. For low ratios, especially for large groups of agents, the second-order weighting averaging approach is the most suitable.

Note that in many applications the sizes of dynamic groups, which could be numerous, are (much) smaller than the total number of agents. The developed abstraction techniques were applied in a large-scale crowd evacuation study (~10000 agents) [14]. Although the number of agents was significant, the maximal size of emergent dynamic groups was 174.

In the future it will be investigated whether the developed approach can be applied for abstracting more complex cognitive multi-agent systems, involving interaction between cognitive and affective processes (e.g., collective decision making with emotions and trust).

References

1. Alur, R., Henzinger, T.A., Lafferriere, G., Pappas, G.J.: Discrete abstraction of hybrid systems. Proceedings of IEEE 88(7), 971–984 (2000)
2. Antoulas, A.C., Sorensen, D.C.: Approximation of large-scale dynamical systems: An overview. International Journal of Appl. Math. Comp. Sci. 11, 1093–1121 (2001)
3. Axelrod, R.: The Dissemination of Culture: A Model with Local Convergence and Global Polarization. Journal of Conflict Resolution 41, 203–226 (1997)
4. Deffuant, G., Neau, D., Amblard, F., Weisbuch, G.: Mixing beliefs among interacting agents. Advances in Complex Systems 3, 87–98 (2001)
5. French, J.R.: A Formal Theory of Social Power. Irvington Publishers (1993)
6. Granovetter, M.: Threshold Models of Collective Behavior. American Journal of Sociology 83(6), 1420–1443 (1978)
7. Hegselmann, R., Krause, U.: Opinion Dynamics and Bounded Confidence Models, Analysis and Simulation. Journal of Artificial Societies and Social Simulation 5(3) (2002)
8. Hoogendoorn, M., Treur, J., van der Wal, C.N., van Wissen, A.: An Agent-Based Model for the Interplay of Information and Emotion in Social Diffusion. In: Proceedings of IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2010, pp. 439–444. IEEE Computer Society Press (2010)
9. Jiang, Y.: Concurrent collective strategy diffusion of multiagents: the spatial model and case study. IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews 39(4), 448–458 (2009)
10. Lewin, K.: Group Decision and Social Change. Holt, Rinehart and Winston, New York (1958)
11. Macy, M., Kitts, J.A., Flache, A.: Polarization in Dynamic Networks: A Hopfield Model of Emergent Structure. In: Dynamic Social Network Modeling and Analysis, pp. 162–173. National Academies Press, Washington, DC (2003)
12. Parunak, H.V.D., Belding, T.C., Hilscher, R., Brueckner, S.: Modeling and Managing Collective Cognitive Convergence. In: Proceedings of AAMAS 2008, pp. 1505–1508. ACM Press (2008)
13. Rogers, E.M.: Diffusion of innovations. Free Press, New York (2003)
14. Sharpanskykh, A., Zia, K.: Grouping behaviour in AmI-enabled crowd evacuation. In: Proceedings of the 2nd International Symposium on Ambient Intelligence, ISAMI 2011, pp. 233–240. Springer, Heidelberg (2011)

Towards a Quantitative Concession-Based Classification Method of Negotiation Strategies

Tim Baarslag, Koen Hindriks, and Catholijn Jonker

Man Machine Interaction Group
Delft University of Technology
{T.Baarslag,K.V.Hindriks,C.M.Jonker}@tudelft.nl

Abstract. In order to successfully reach an agreement in a negotiation, both parties rely on each other to make concessions. The willingness to concede also depends in large part on the opponent. A concession by the opponent may be reciprocated, but the negotiation process may also be frustrated if the opponent does not concede at all.

This process of concession making is a central theme in many of the classic and current automated negotiation strategies. In this paper, we present a quantitative classification method of negotiation strategies that measures the willingness of an agent to concede against different types of opponents. The method is then applied to classify some well-known negotiating strategies, including the agents of ANAC 2010. It is shown that the technique makes it easy to identify the main characteristics of negotiation agents, and can be used to group negotiation strategies into categories with common negotiation characteristics. We also observe, among other things, that different kinds of opponents call for a different approach in making concessions.

Keywords: Automated bilateral negotiation, Classification, Concession, Cooperation, Competition, Negotiation strategy.

1 Introduction

In bilateral negotiation, an opening offer is usually met with a counteroffer, and this then defines the initial bargaining range [12] of the negotiation. Sometimes the other party will immediately accept the offer, or will state that the set of demands is unacceptable, breaking off the negotiation. But usually, after the first round of offers, the question is: what concession is to be made next? One can choose to signal a position of firmness and stick to the original offer. Or one can take a more cooperative stance, and choose to make a concession. If one side is not prepared to make concessions, the other side must capitulate, or more commonly, the negotiation will end up in a break off.

The making of concessions is therefore central to a successful negotiation. Without them, negotiations would not exist [12]. Negotiation can even be defined in terms of making concessions: Pruitt [16] defines it as a process by which a joint decision is made by two or more parties that first verbalise contradictory demands and then move towards agreement by a process of concession making or search for new alternatives.

Many of the classic negotiation strategies are characterized by this process of concession making throughout the negotiation. For example, the time dependent tactics such as Boulware and Conceder [5] are characterised by the fact that they steadily concede throughout the negotiation process. Other strategies, like behaviour dependent tactics (such as Tit for Tat) [1,4] base their concessions on the concessions of the other negotiating party.

The choice for what concessions to make depends in large part on the opponent. A concession by the opponent may be reciprocated by another concession, leading to a whole progression of concessions. On the other hand, the negotiation process can easily be frustrated if the opponent adopts a take-it-or-leave-it approach. Against this background, this paper studies negotiation strategies according to the way they concede towards different types of opponents.

This work advances the state-of-the-art in automated negotiation in the following ways. We present a new classification method for negotiation strategies, based on their pattern of concession making against different kinds of opponents. We introduce a definition of Concession Rate (CR) which measures the cooperativeness of an agent. We present a technique to quantitatively measure the CR against two extreme types of strategies: a take-it-or-leave-it strategy, and a conceding tactic. We then apply this technique to classify some well-known negotiating strategies, including the agents of ANAC 2010. This gives, for the first time, insight into the negotiation strategy space of the ANAC 2010 agents. It also aids our understanding of what concession making strategies are effective in settings such as ANAC.

In the discussion of our experimental results, we conclude that our technique has the desirable property of grouping negotiation strategies into categories with common negotiation characteristics. Among other things, we observe that different kinds of opponents call for a different approach in making concessions. For instance, a successful negotiating agent should behave competitively, especially against very cooperative strategies.

The remainder of this paper is organized as follows. Section 2 provides an overview of concession making in negotiation, including our adopted model of negotiation, and the definition of concession rate. In Section 3 we outline a method to compute the concession rate, followed by Section 4 that presents our experimental results. In Section 5 we discuss our findings, and finally, Section 7 presents our conclusions and our plans for future work.

2 Concession Making in Negotiation

In earlier work on conflict management through negotiation, the negotiation stance was characterized by two orientations: cooperative and competitive [3]. The theory relates to two basic types of goal interdependence that negotiators might have. It is either positive, where the negotiators' goals are linked in such a way that their goal attainments are positively correlated ('sink or swim together'), or the interdependence is negative, namely when the goal attainments are negatively correlated ('when one swims, one sinks').

However, a negotiator's stance is usually not limited to one of the two orientations, because negotiation is a dynamic process and the position of the negotiators can change in response to the other party's information or behavior [12]. In this paper, we take the stance that negotiators can exhibit a mixture of the two orientations, mainly depending on the type of opponent (see Figure 1). For example, a negotiator may cooperate with a cooperative opponent, but the same negotiator may be very competitive when facing competition. That is, in this case it *matches* the behavior of the opponent.

Conversely, a negotiator can be cooperative towards a competitive opponent and at the same time exploit cooperative opponents by playing competitive against them. In that case, it *inverts* the opponent's behavior.

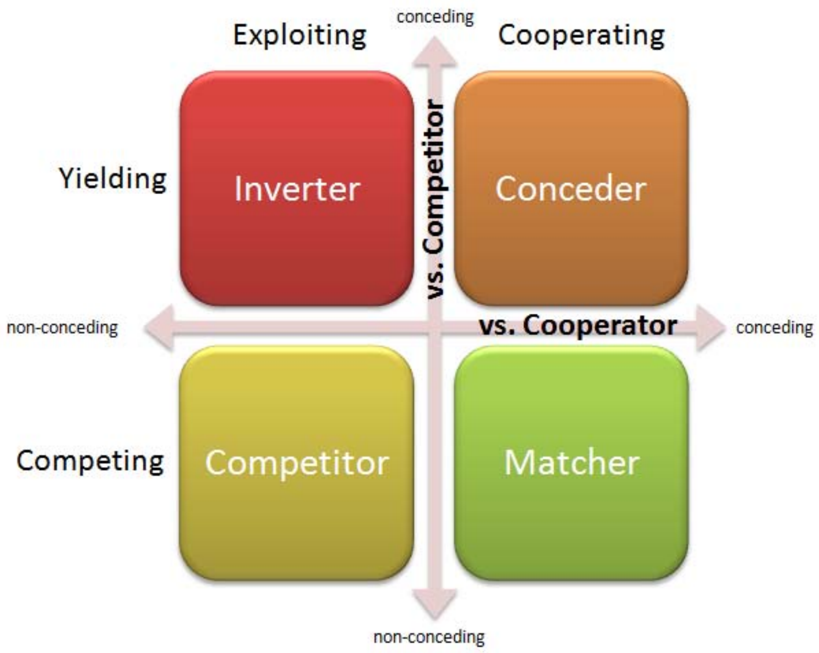


Fig. 1. The diagram of conceding behavior against both cooperative and competitive opponents

This way, we distinguish four types of negotiation orientations depending on the behavior against the opponent (see Table 1): *Inverter*, *Conceder*, *Competitor*, and *Matcher*. Every negotiation orientation corresponds to a different stance towards either of the two types of opponents. The main contribution of this paper is to define a formal, mathematical procedure for classifying agents into one of the four categories.

Table 1. Four types of negotiation orientations

Orientation	**vs. Conceder**	**vs. Hardliner**
Inverter	Exploiting	Yielding
Conceder	Cooperating	Yielding
Competitor	Exploiting	Competing
Matcher	Cooperating	Competing

2.1 Negotiation Model

We consider *bilateral* negotiations, i.e. a negotiation between two parties or agents A and B. The agents negotiate over *issues* that are part of a negotiation *domain*, and every issue has an associated range of alternatives or *values.* A negotiation outcome consists of a mapping of every issue to a value, and the set Ω of all possible outcomes is called the *outcome space.* The outcome space is common knowledge to the negotiating parties and stays fixed during a single negotiation session.

We further assume that both parties have certain preferences prescribed by a *preference profile* over Ω. These preferences can be modeled by means of a normalized utility function U, which maps a possible outcome $\omega \in \Omega$ to a real-valued number in the range $[0, 1]$. In contrast to the outcome space, the preference profile of the agents is private information.

Finally, the interaction between negotiating parties is regulated by a *negotiation protocol* that defines the rules of how and when proposals can be exchanged. We use the alternating-offers protocol [17] for bilateral negotiation, in which the negotiating parties exchange offers in turns.

As in [18], we assume a common global time, represented here by $\mathcal{T} = [0, D]$. The alternating-offers protocol is supplemented with a deadline D at the end of the time line, so for any $t \in \mathcal{T}$, we stipulate that the deadline has been reached when $t = D$, at which moment both agents receive utility 0. This is the same setup as [6], with the exception that issues are not necessarily real-valued and both agents have the same deadline. We represent by $x^t_{A\to B}$ the negotiation outcome proposed by agent A to agent B at time t. A *negotiation thread* (cf. [5,18]) between two agents A and B at time $t \in \mathcal{T}$ is defined as a finite sequence

$$H^t_{A\leftrightarrow B} := \left(x^{t_1}_{p_1\to p_2}, x^{t_2}_{p_2\to p_3}, x^{t_3}_{p_3\to p_4}, \ldots, x^{t_n}_{p_n\to p_{n+1}}\right),$$

where

1. $t_k \leq t_l$ for $k \leq l$, the offers are ordered over time $\mathcal{T}$,
2. $p_k = p_{k+2} \in \{A, B\}$ for all k, the offers are alternating between the agents,
3. All t_i represent instances of time $\mathcal{T}$, with $t_n \leq t$,
4. $x^{t_k}_{p_k\to p_{k+1}} \in \Omega$ for $k \in \{1, \ldots, n\}$, the agents exchange complete offers.

Additionally, the last element of $H^t_{A\leftrightarrow B}$ may be equal to one of the particles $\{Accept, End\}$. We will say a negotiation thread is *active* if this is not the case.

When agent A receives an offer $x^t_{B\to A}$ from agent B sent at time t, it has to decide at a later time $t' > t$ whether to accept the offer, or to send a counter-offer $x^{t'}_{A\to B}$. Given a negotiation thread $H^t_{A\leftrightarrow B}$ between agents A and B, we can express the action performed by A with an *decision function* [6,18]. The resulting action is used to extend the current negotiation thread between the two agents. If the agent does not accept the current offer, and the deadline has not been reached, it will prepare a counter-offer by using a bidding strategy or *tactic* to generate new values for the negotiable issues.

Tactics can take many forms, e.g. time-dependent, resource dependent, imitative, and so on [18]. In our setup we will consider the tactics as given and try to categorize them according to their willingness to concede.

2.2 Concession Rate

In this section we introduce the notion of concession rate which quantifies the amount an agent has conceded towards the opponent during a negotiation. It is generally not enough to simply consider the utility of the agreement as a measure for the concession rate. For instance, a negotiator may not get an agreement before the deadline. In that case, both parties receive zero utility, but this gives no information about the concessions that were made. Therefore, we define the concession rate in terms of the minimum utility a negotiator has demanded during the negotiation.

Suppose a player A has a utility function U_A, mapping any outcome in Ω into the range $[0, 1]$. As we have assumed that the utility function is normalized in our setting, there will exist an *optimal outcome* $\omega^{\text{opt}}_A \in \Omega$ for which $U_A(\omega^{\text{opt}}_A) = 1$. In typical negotiation domains, the corresponding utility $U_B(\omega^{\text{opt}}_A)$ of this outcome is far from optimal for player B, because the best outcome for A is typically not the best outcome for B. Player B should be able to always obtain at least this outcome in a negotiation, as A will always be inclined to accept it. We shall refer to this utility as the *full yield utility* (FYU_B) of player B (see Fig. 2). Intuitively, it is equal to his bottom line utility.

Note that an optimal outcome ω^{opt}_A is not necessarily unique, but typical domains (including those considered in ANAC and hence, in this paper) all have unique optimal outcomes for both players, so that the full yield utility is well-defined.

For any $t \in \mathcal{T}$, let

$$H^t_{A\to B} = \left\{x^s_{A\to B} \in H^t_{A\leftrightarrow B} \mid s \leq t\right\}$$

denote all bids offered by A to B until time t in an active negotiation thread. We can now formulate the minimum utility that agent A demanded during the negotiation thread $H^t_{A\to B}$. That is to say, we consider the largest concession the player has made so far:

$$\text{MIN}^t_A = \min\{\, U_A(x) \mid x \in H^t_{A\to B}\}$$

Informally, MIN^t_A denotes the lowest that A is willing to bid up until time t. The inverse of this is called the *yield* of player A. The lower player B is willing to go,

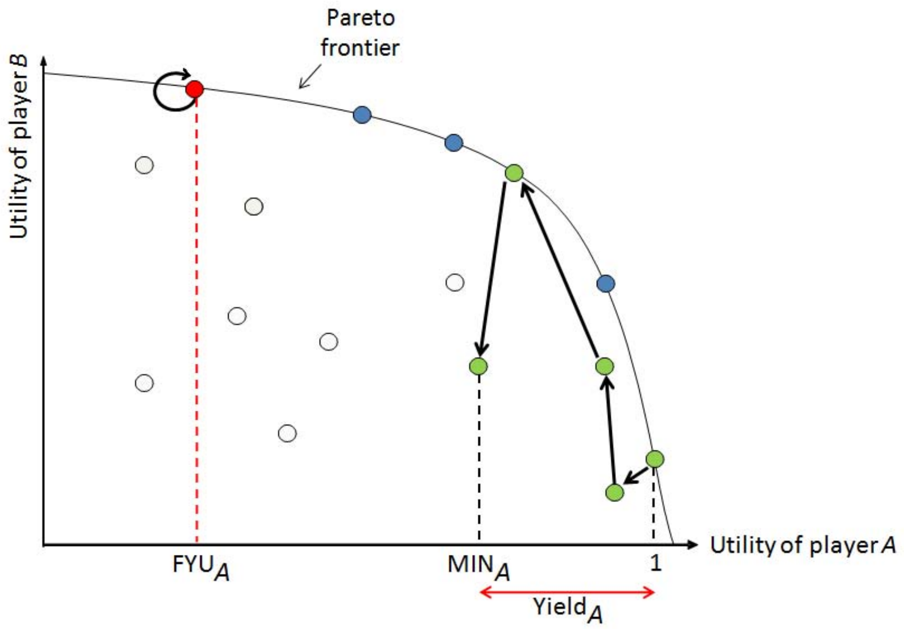


Fig. 2. The yield of player A is determined by MIN_A^t

the larger the yield. A yield of zero means the player has made no concession whatsoever (and therefore his demanded utility remains equal to one); A yield of $1 - \text{FYU}$ means the player has yielded fully (see Fig. 2). That is, it is defined as:

$$\text{Yield}_A^t = 1 - \max\left(\text{MIN}_A^t, \text{FYU}_A\right).$$

The Concession Rate $\text{CR}_A^t \in [0, 1]$ of player A up until time t is then simply the normalized yield:

$$\text{CR}_A^t = \frac{\text{Yield}_A^t}{1 - \text{FYU}_A}.$$

By normalizing, it is guaranteed that if $\text{CR}_A^t = 0$, then A has not conceded at all, while for $\text{CR}_A^t = 1$, player A has conceded fully (i.e., up to its full yield utility). Normalizing has the added benefit of reducing domain bias: in a typical competitive domain such as Itex–Cypress (defined in the experimental section below) players may obtain utilities anywhere between 0.2 and 1, while in very cooperative domains utilities may vary between 0.9 and 1. Normalization ensures that the concession rate can be compared over such different domains.

This paper only deals with the concession rate CR_A^D of a player A during the entire negotiation thread. We shall denote this simply by CR_A. We also omit the subscript A when it is clear from the context.

3 Method

In order to classify agents according to their concession rate, we considered a negotiation setup with the following characteristics. We selected a set of agents (introduced later) and let them negotiate against both a very cooperative and a very competitive opponent. The opponent tactics that we use to measure concession rates are simple, non-adaptive negotiation tactics. This ensures that the results depend as much as possible on the agent's own negotiating tactic. To be more precise, we aim for three opponent characteristics:

1. **Simplicity**
   If the opponent negotiation tactic is simple and easy to understand, then the results depend on the agent's own negotiating tactic, which makes them easier to interpret.
2. **Regularity**
   We want to give the agent enough time to show its bidding behavior; therefore, the opponent should not end the negotiation prematurely by either reaching an agreement too fast or breaking off the negotiation.
3. **Deterministic behavior**
   In order to reduce variance in experimental results, we prefer deterministic agents to agents that demonstrate random bidding behavior.

For the *competitive opponent*, we chose Hardliner (also known as take-it-or-leave-it or Hardball [12]). This strategy simply makes a bid of maximum utility for itself and never concedes. This is the most simple competitive strategy that can be implemented and it fits the other two criteria as well: it is deterministic, and it gives the agent the full negotiation time to make concessions.

For the *cooperative opponent*, we selected Conceder Linear, i.e. the Time Dependent Tactic adapted from [6,5] with parameter $e = 1$. Depending on the current time $t \in [0, 1]$, this strategy makes a bid with utility closest to

$$P_{min} + (P_{max} - P_{min}) \cdot (1 - F(t)), \tag{1}$$

with

$$F(t) = k + (1 - k) \cdot t^{1/e}.$$

In this experiment, we selected the standard values $k = 0$, and P_{max}, P_{min} are respectively set to the maximum and minimum utility that can be obtained in the domain. With these values, and setting $e = 1$, we obtain a very simple conceding tactic. It reduces equation (1) to

$$P_{min} + (P_{max} - P_{min}) \cdot (1 - t),$$

so that it linearly reduces its demanded utility (from maximum P_{max} to minimum P$_{min}$) as time passes.

There exist even simpler conceding tactics such as Random Walker (which generates random bids), or an agent that accepts immediately. However, both opponent strategies are not *regular* in the sense that they do not give the agent enough time to show its bidding behavior. Random Walker has the added disadvantage of not being deterministic. Therefore, we believe Random Walker can serve as a useful base line strategy, but not as a useful opponent to measure an agent's willingness to concede. Consequently, we selected Conceder Linear as the cooperative opponent, as it fulfills the three requirements listed above.

We measured the concession rate of an agent A playing against the two agents in the following way. Suppose agent A negotiates with either Conceder Linear, or Hardliner. The two parties may attain a certain outcome, or reach the deadline. In both cases, at the end of the negotiation, A has reached a certain concession rate as defined in Section 2.2. The concession rate is then averaged over all trials on various domains (see Section 4.1), alternating between the two preference profiles defined on that domain. E.g., on the negotiation scenario between England and Zimbabwe, A will play both as England and as Zimbabwe.

4 Experiments

For our experimental setup we employed Genius (General Environment for Negotiation with Intelligent multi-purpose Usage Simulation) [13]. This environment, which is also used in ANAC, facilitates the design and evaluation of automated negotiators' strategies. It can be used to simulate tournaments between negotiating agents in various negotiation scenarios, such as the setup described in this section. It supports the alternating offer protocol with a real-time deadline as outlined in our negotiation model. The default negotiation time in Genius and in the setup of ANAC is 3 minutes per negotiation session; therefore, we use the same value in our experiments.

4.1 Detailed Experimental Setup

Agents

In our experimental setup we included all the negotiation tactics that were submitted to The Automated Negotiating Agents Competition (ANAC 2010) [2]. ANAC is a negotiation competition aiming to facilitate and coordinate the research into proficient negotiation strategies for bilateral multi-issue negotiation, similar to what the Trading Agent Competition (TAC) has achieved for the trading agent problem [20]. The seven agents that participated in ANAC 2010 have been implemented by various international research groups of negotiation experts. We used these strategies in our experiments as they are representative of the current state-of-the-art in automated negotiation. In order of final ranking the strategies are: *Agent K*, *Yushu*, *Nozomi*, *IAMhaggler*, *FSEGA*, *IAMcrazyHaggler*, and *Agent Smith*.

Table 2 gives a short overview of the variety of agent strategies used in our experiments. The "Time dependent strategy" column shows whether the strategies keep track of the time that is left and change their proposals accordingly. The next column specifies what kind of learning method the agents use to generate the next offer (more details are provided in [2]).

Table 2. Short overview of the strategies employed by the ANAC 2010 agents

	Time dependent strategy	Learning method	Deterministic
Agent K	Yes	All proposals	No
Yushu	Yes	Best proposals	No
Nozomi	No	Match compromises	No
IAMhaggler	Yes	Bayesian learning	No
FSEGA	Yes	Bayesian learning	Yes
IAMcrazyHaggler	No	None	No
Agent Smith	Yes	Learning weights	Yes

In addition to the ANAC agents, we included some well-known agents to explore some extreme cases. First, we included the Hardball strategy described in Section 3, which consistently makes the maximum bid for itself.

We also studied three members of the Time Dependent Tactics family [5] as defined above, namely: Boulware ($e = 0.2$), Conceder Linear ($e = 1$), and Conceder ($e = 2$). We included a variant of the Relative Tit-for-Tat agent from the same paper. This strategy, called Simple Nice Tit for Tat, tries to reproduce the behaviour that its opponent performed in the previous step.

Finally, we included the Random Walker strategy, also known as Zero Intelligence strategy [7], which randomly jumps through the negotiation space. It does not employ any information about its own preferences to make an offer.

Domains

The specifics of a negotiation domain can be of great influence on the negotiation outcome [8]; therefore, negotiation characteristics such as concession rate have to be assessed on negotiation domains of different size and competitiveness (or *opposition* [9]). With this in mind, we aimed for two domains (with two preference profiles each) with a good spread of negotiation characteristics. We picked two our of the three domains that were used in ANAC 2010 [2]. We omitted the third domain (Travel) as some of the ANAC agents did not scale well and had too many difficulties with it to make it a reliable testing domain.

Our first scenario is taken from [10], which describes a buyer–seller business negotiation. It involves representatives of two companies: Itex Manufacturing, a producer of bicycle components and Cypress Cycles, a builder of bicycles. There are four issues that both sides have to discuss, including the price of the components, delivery times, etc. The opposition between the parties is strong in this domain, as the manufacturer and consumer have naturally opposing requirements. Altogether, the Itex–Cypress domain contains 180 potential offers.

The second domain taken from [13,14] involves a case where England and Zimbabwe negotiate an agreement on tobacco control. The leaders of both countries must reach an agreement on five issues. The England–Zimbabwe is of medium opposition, because the parties have contradictory preferences for some issues, but other issues have options that are jointly preferred by both sides. The domain has a total of 576 possible agreements.

Table 3. The four preference profiles used in experiments

	Itex–Cyp	**Eng–Zim**
Size	180	576
Opposition	Strong	Medium

4.2 Experimental Results

We present the results of the experiments in Table 4 and its graphical representation is depicted in Fig. 3.

Table 4. An overview of the concession rate of every agent in the experiments

Agent	**CR vs. Conceder**	**CR vs. Hardliner**
Agent K	0.12	0.18
Agent Smith	0.46	1.00
Boulware	0.14	1.00
Conceder Linear	0.43	1.00
Conceder	0.63	1.00
FSEGA	0.33	0.76
Hardliner	0.00	0.00
IAMcrazyHaggler	0.05	0.05
IAMhaggler	0.02	0.27
Nozomi	0.20	0.22
Random Walker	0.97	1.00
Simple Nice Tit for Tat Agent	0.42	0.01
Yushu	0.11	0.95

Extreme Cases

The Hardliner strategy and the Random Walker strategy are at the opposite sides of the spectrum. Hardliner will not concede to any type of strategy, so by definition it has $CR = 0$ against both Hardliner and Conceder. Consequently, Hardliner defines the most competitive strategy possible.

On the other hand, Random Walker will make arbitrary concessions given enough time. This makes Random Walker one of the most cooperative strategies possible. Against Hardliner, there is plenty of time for Random Walker to randomly produce a bid with which it fully concedes, so it has $CR = 1$ against Hardliner. Against Conceder, it may not have time to fully concede, but it generally will produce offers of very low utility in this case as well, resulting in a CR of 0.97.

We considered three members of the Time Dependent Tactics family: Boulware ($e = 0.2$), Conceder Linear ($e = 1$), and Conceder ($e = 2$) who are all in the top of the chart because they have a CR equal to 1 versus the Hardliner. In addition to the time dependent tactics, at the top of the chart we see two more strategies: *Agent Smith* and Random Walker. This means all these strategies give in fully to Hardliner and are thus fully exploited by a strategy that does

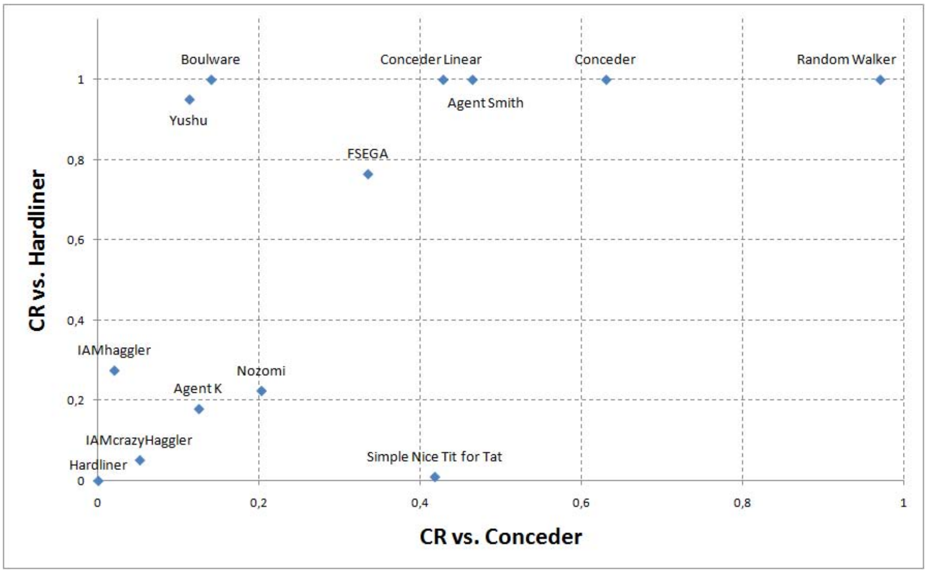


Fig. 3. A graphical overview of the concession rates of the agents

not give in at all. All of these five strategies have a very simple bidding strategy and are apparently not optimized to deal with very uncooperative opponents.

5 Discussion

This section makes observations regarding the clustering of different strategies in Fig. 3, and then classifies them into the four negotiation orientations we have discussed previously.

5.1 Clustering

Agent Smith and Conceder Linear are very close in the chart and this is no coincidence: *Agent Smith* uses essentially the same strategy as the linear Conceder, by first making a proposal of maximum utility and subsequently conceding linearly towards the opponent.

The same holds for *Yushu* and Boulware: the strategies are very similar, as is indicated by their close vicinity in the chart. Like Boulware, *Yushu* adopts a very competitive time dependent strategy, making larger concessions when the negotiation deadline approaches. Both adopt a conservative concession making strategy and are not willing to make large concession at the beginning, but prefer to wait for their opponent to make concessions.

These two examples show that this chart can be useful to cluster strategy types, as similar strategies have similar concession characteristics.

Clustering also occurs on lines in the chart. For example, the three Time Dependent Tactics all share the same behavior against Hardliner: all three ultimately give in to it. This is to be expected, as any agent from the time dependent family will offer the reservation value when the deadline is being reached [6], resulting in full concession to the opponent. In general, all Time Dependent Tactics will lie on the line $CR = 1$ against Hardliner.

Against a more cooperative strategy like Conceder, the results are also intuitively clear: concessions get bigger when the parameter $e \in (0, \infty)$ gets bigger, so Boulware concedes the least, while Conceder concedes the most. More generally, when $e \to 0$, then $CR \to 0$. Conversely, when $e \to \infty$, then $CR \to 1$.

Finally, there is a big cluster of strategies in the left part of the chart, which is populated by the top four strategies in ANAC: *Agent K*, *IAMhaggler*, *Nozomi* and *Yushu*. The better performing strategies of ANAC have different approaches towards the Hardliner, but they seem to have one trait in common: they all concede very little to the Conceder. In other words: they exploit the Conceder by waiting for even bigger concessions. The fact that these strategies did very well in ANAC seems to indicate that in order to be successful in an automated negotiation competition, an agent should behave competitively, especially against very cooperative strategies.

5.2 Four Negotiation Orientations

We classify the different agent strategies of Fig. 3 into the four negotiation orientations of Fig. 1. This procedure is necessarily arbitrary; nevertheless, we propose the following grouping.

The top left agents in the diagram can be considered to be Inverters: *Yushu*, Boulware, and *FSEGA*. The remaining agents in the top right are then Conceders, namely: Conceder Linear, *Agent Smith*, Conceder, and Random Walker.

The Simple Nice Tit for Tat strategy is the only strategy that can be considered a Matcher, i.e.: it does not concede to a Hardliner, but it does concede to the Conceder. Clearly, this is to be expected from a Tit for Tat strategy, as it is based on cooperation through reciprocity: it matches whatever the other player did on the preceding move. The fact that this type of strategy does not occur naturally in ANAC can be explained by our previous comments on clustering: following a Tit for Tat strategy is not as successful in negotiation, because it does not exploit the conceding strategies.

All of the remaining strategies are Competitors, i.e. they do not concede much, whether it is against a cooperative or a competitive agent. The majority of strategies that performed well during ANAC are located in this region. Again, we observe that the successful strategies are very competitive.

6 Related Work

This paper is inspired by ideas presented in [11] (of which parts originally appeared in unpublished work by Kersten in 2005). In [11], four dual negotiation orientations are distinguished, depending on the negotiator's own orientation and that of the negotiating partner. Both orientations can be either competitive or cooperative, leading to four different labels: Competitor, Yielder, Exploiter, and Cooperator. We re-use these labels to name the stance of a negotiator against different kinds of opponents (see Fig. 1). However in our work, the negotiators are assumed to have different responses to different observed behavior by the other party. Therefore, instead of the negotiator having one particular stance during the negotiation, the position of the negotiators can change in response to the competitiveness of the opponent. For example, a negotiator may be *both* an Exploiter (against a Cooperator), and a Yielder (against a Competitor). The negotiator would then be called an Inverter, as he takes on the reverse role of his opponent.

In [15], a classification scheme is given for electronic commerce negotiation, including characteristics of the negotiating agents. It is argued that agents can act in a self-interested way, or altruistically, or strike a balance in between. This choice is then seen as a component of the bidding strategy of the agent, which ultimately decides how and when to place offers, or when to withdraw, etc. Although the paper makes this distinction in bidding characteristics, it does not provide a definition or a way to quantify them.

Thomas [19] redefines five conflict–handling modes that can be applied to negotiation: competing, collaborating, compromising, avoiding and accommodating. Similar to our work, the classification method uses two underlying dimensions. However, the underlying dimension are different, namely: assertiveness (attempting to satisfy one's own concerns), and cooperativeness (attempting to satisfy other's concerns). This classification method is phrased in qualitative, intentional terms of the conflict-handler. Similarly, [22] distinguishes negotiation strategies into two strategy types: distributive and integrative. This description also focuses on the approach used by the negotiators. Our paper has a different focus from both papers, centering around quantitative negotiation characteristics *in response* to agents having either high and low concession rates. Furthermore, we do not classify negotiation strategies in a binary way (either cooperative or non-cooperative), but we employ a continuous spectrum in our approach.

Currently, there are two papers that analyze the results of ANAC 2010. Baarslag et al. [2] give a short overview of all negotiation tactics and their rankings in the tournament, but they do not provide an in-depth analysis of the bidding behavior of the ANAC participants. Williams et al. [21] consider self-play scores of the agents and also perform an empirical game theoretic analysis of the tournament results. This work focuses on stability of a strategy in a tournament with different mixes of opponent strategies, but unlike our work, it does not discuss or aim to classify the characteristics of the agent's negotiation strategies.

7 Conclusion and Future Work

Making concessions during a negotiation is vital for getting an agreement. Successful negotiations are only possible when both parties employ an effective conceding strategy. Designing a good strategy of when and how much to concede is challenging, and that is why there are many current negotiation implementations that concede in very different ways.

In this paper, we aimed to classify a selection of current automated negotiation strategies according to their pattern of concession making in order to gain insight into their negotiation characteristics. We first formally defined the notion of concession rate which gives a normalized measure of the largest concession that was made during the negotiation. This formalizes the concept of an agent's willingness to concede against different opponents.

We then presented an empirical method to effectively compute the concession rate of agents, and then applied our approach to a selection of well-known agents (including all participants of ANAC 2010) in an experimental setting. For the first time, this gives insight into the strategy space of negotiation tactics employed in ANAC 2010. We subsequently used our method to classify the agents into four categories types of concession behavior.

In addition to classifying agent strategies, we have drawn various conclusions based on charting the experimental results. We have seen that there is indeed a wide spread in concession rates of current agents. We established that the chart can be useful to cluster strategy types, as similar strategies have similar concession characteristics. Secondly, it makes it easy to understand the agent's main negotiation characteristics at a glance.

Some extreme agents are located in the extreme regions of the chart, while the stronger agents form a cluster in the competitive corner. The results indicate that in order to be successful in an automated negotiation competition, an agent should not concede much, especially not to very cooperative strategies.

While making a number of contributions, this paper also opens up some lines of future work. We plan to conduct a deeper investigation of the impact that concession rates have on tournament results. The focus of this paper is on the tournament setting of ANAC 2010, but it would be interesting to extend the ideas presented in this paper to the results of ANAC 2011, especially because the 2011 competition contains negotiation domains that have discount factors. As discount factors devaluate utility with the passing of time, they require the negotiating agents to give even more consideration to effective concession-making.

Secondly, the focus of this paper has been on bidding behavior, and not on acceptance strategy. In general, this is an important part of a negotiator's strategy that also highly influences the outcome of a negotiation. We believe the same interactions between cooperation and competition play a role when agents decide when and whether to accept. This could provide an interesting addition to our work, which we plan to examine in future research.

Acknowledgements. This research is supported by the Dutch Technology Foundation STW, applied science division of NWO and the Technology Program of the Ministry of Economic Affairs. It is part of the Pocket Negotiator project with grant number VICI-project 08075.

References

1. Axelrod, R.: The Evolution of Cooperation. Basic Books (1984)
2. Baarslag, T., Hindriks, K., Jonker, C.M., Kraus, S., Lin, R.: The first automated negotiating agents competition (ANAC 2010). In: Ito, T., Zhang, M., Robu, V., Fatima, S., Matsuo, T., Yamaki, H. (eds.) Innovations in Agent-Based Complex Automated Negotiations. SCI, vol. 319. Springer, Heidelberg (2010) (to appear)
3. Deutsch, M., Coleman, P.T., Marcus, E.C.: The Handbook of Conflict Resolution: Theory and Practice, 1st edn. Jossey-Bass (April 2000)
4. Faratin, P., Sierra, C., Jennings, N., Buckle, P.: Designing flexible automated negotiators: Concessions, trade-offs and issue changes. Tech. rep. (1999)
5. Faratin, P., Sierra, C., Jennings, N.R.: Negotiation decision functions for autonomous agents. Int. Journal of Robotics and Autonomous Systems 24(3-4), 159–182 (1998)
6. Fatima, S.S., Wooldridge, M.J., Jennings, N.R.: Optimal Negotiation Strategies for Agents with Incomplete Information. In: Meyer, J.-J.C., Tambe, M. (eds.) ATAL 2001. LNCS (LNAI), vol. 2333, pp. 377–392. Springer, Heidelberg (2002)
7. Gode, D.K., Sunder, S.: Allocative efficiency in markets with zero intelligence (zi) traders: Market as a partial substitute for individual rationality. Journal of Political Economy 101(1), 119–137 (1993)
8. Hindriks, K.V., Tykhonov, D.: Towards a Quality Assessment Method for Learning Preference Profiles in Negotiation. In: Ketter, W., La Poutré, H., Sadeh, N., Shehory, O., Walsh, W. (eds.) AMEC 2008. LNBIP, vol. 44, pp. 46–59. Springer, Heidelberg (2010)
9. Kersten, G., Noronha, S.: Rational agents, contract curves, and inefficient compromises report. Working papers, International Institute for Applied Systems Analysis (1997), http://econpapers.repec.org/RePEc:wop:iasawp:ir97050
10. Kersten, G.E., Zhang, G.: Mining inspire data for the determinants of successful internet negotiations. InterNeg Research Papers INR 04/01 Central European Journal of Operational Research (2003)
11. Lai, H., Doong, H.S., Kao, C.C., Kersten, G.: Negotiators' communication, perception of their counterparts, and performance in dyadic e-negotiations. Group Decision and Negotiation 15, 429–447 (2006)
12. Lewicki, R.J., Saunders, D.M., Minton, J.W.: Essentials of Negotiation. McGraw-Hill, Boston (2003)
13. Lin, R., Kraus, S., Tykhonov, D., Hindriks, K., Jonker, C.M.: Supporting the design of general automated negotiators. In: Proceedings of the Second International Workshop on Agent-based Complex Automated Negotiations, ACAN 2009 (2009)
14. Lin, R., Kraus, S., Wilkenfeld, J., Barry, J.: Negotiating with bounded rational agents in environments with incomplete information using an automated agent. Artificial Intelligence 172(6-7), 823–851 (2008)
15. Lomuscio, A., Wooldridge, M., Jennings, N.: A Classification Scheme for Negotiation in Electronic Commerce. In: Dignum, F., Sierra, C. (eds.) AgentLink 2000. LNCS (LNAI), vol. 1991, pp. 19–33. Springer, Heidelberg (2001)
16. Pruitt, D.G.: Negotiation Behavior. Academic Press (1981)
17. Rubinstein, A.: Perfect equilibrium in a bargaining model. Econometrica 50(1), 97–109 (1982), http://www.jstor.org/stable/1912531
18. Sierra, C., Faratin, P., Jennings, N.: A service-oriented negotiation model between autonomous agents. In: Boman, M., Van de Velde, W. (eds.) MAAMAW 1997. LNCS (LNAI), vol. 1237, pp. 17–35. Springer, Heidelberg (1997)

19. Thomas, K.W.: Conflict and conflict management: Reflections and update. Journal of Organizational Behavior 13(3), 265–274 (1992)
20. Wellman, M.P., Wurman, P.R., O'Malley, K., Bangera, R., de Lin, S., Reeves, D., Walsh, W.E.: Designing the market game for a trading agent competition. IEEE Internet Computing 5(2), 43–51 (2001)
21. Williams, C.R., Robu, V., Gerding, E.H., Jennings, N.R.: Using gaussian processes to optimise concession in complex negotiations against unknown opponents. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence. AAAI Press (January 2011)
22. Zachariassen, F.: Negotiation strategies in supply chain management. International Journal of Physical Distribution and Logistics Management 38, 764–781 (2008)

Consensus Policy Based Multi-agent Negotiation

Enrique de la Hoz[1], Miguel A. Lopez-Carmona[1],
Mark Klein[2], and Ivan Marsa-Maestre[1]

[1] Computer Engineering Department, Universidad de Alcala
Escuela Politecnica, 28871, Alcala de Henares (Madrid), Spain
{enrique.delahoz,miguelangel.lopez,ivan.marsa}@uah.es
[2] Center for Collective Intelligence, MIT Sloan School of Management,
Massachusetts Institute of Technology
m_klein@mit.edu

Abstract. Multiagent negotiation may be understood as a consensus based group decision-making which ideally should seek the agreement of all the participants. However, there exist situations where an unanimous agreement is not possible or simply the rules imposed by the system do not seek such unanimous agreement. In this paper we propose to use a consensus policy based mediation framework (CPMF) to perform multiagent negotiations. This proposal fills a gap in the literature where protocols are in most cases indirectly biased to search for a quorum. The mechanisms proposed to perform the exploration of the negotiation space are derived from the Generalized Pattern Search non-linear optimization technique (GPS). The mediation mechanisms are guided by the aggregation of the agent preferences on the set of alternatives the mediator proposes in each negotiation round. Considerable interest is focused on the implementation of the mediation rules where we allow for a linguistic description of the type of agreements needed. We show empirically that CPMF efficiently manages negotiations following predefined consensus policies and solves situations where unanimous agreements are not viable.

1 Introduction

Most research in multiparty automated negotiation has focused on building efficient mechanisms and protocols to reach agreements among multiple participants, being an objective to optimize some type of social welfare measurement [4,12,3,5,9,10]. Examples of such measurements would be the *sum or product of utilities*, the *min* utility, etc. However, social welfare has not been usually placed itself as an integral part of the negotiation process.

There are remarkable works which incorporate a social welfare criterion within the search process [1,2,7]. In these works, the authors build mechanisms to obtain fair agreements by using fair direction improvements in the joint exploration of the negotiation space. Put simply, first a mediator proposes a solution and agents provide their utility gradients in the solution, and finally the mediator proposes a new contract in the bisector or in an arbitrary direction which is considered fair enough. These proposals present, however, several limitations. Firstly, they work only when utility functions are derivable and quasi-concave. Secondly, the absolute value of the gradient is not considered, and so, the marginal utility obtained by the agents in each negotiation round may not be fair. Finally, even considering that the agents reveal also the gradient magnitude, the protocol is prone to untruthful revelations of information to bias the direction generated by the mediator.

We argue that the type of consensus by which an agreement meets in some specific manner the concerns of all the negotiators should be considered as an integral part within the multiparty negotiation protocols. To study this hypothesis this paper proposes CPMF, a *Consensus Policy Based Mediation Framework for multi-agent negotiation*. CPMF relies on a novel distributed agreement exploration protocol based on the *Generalized Pattern Search* optimization technique (GPS) [6], and on the use of *Ordered Weighted Averaging* (OWA) operators [14]. This framework allows to search for agreements following predefined consensus policies, which may take the form of linguistic expressions in order to satisfy system requirements or to circumvent situations where unanimous agreements are not possible.

The rest of the paper is organized as follows. Next section presents first the GPS algorithm for unconstrained optimization and then the basic operation of the negotiation protocol. Section 3 focuses on the mechanisms used by the mediator to aggregate agents' preferences and Section 4 presents the agreement search process. The last section summarizes our conclusions and sheds lights on some future research.

2 The Mediation Protocol

We shall assume a set of n agents $A = \{A_i | i = 1, \ldots, n\}$ and a finite set of issues $X = \{x_l | l = 1, \ldots, s\}$, where each issue x_l can be normalized to a continuous or discrete range $d_l = [x_l^{min}, x_l^{max}]$. Accordingly, a *contract* is a vector $x' = \{x'_l = 1, \ldots, s\}$ defined by the issues values. Furthermore, we assume that each agent A_i has a real or virtual mapping $V_i : X \rightarrow \mathbb{R}$ function that associates with each contract x a value $V_i(x)$ that gives the payoff the agent assigns to a contract. The exact nature of this mapping needs not be known. All that we want to assume is that each agent has some means to formulate a preference function over a set of alternatives. Thus, the preference function can be described as a mapping function between the negotiation space contracts and the set of real numbers. We make a general assumption that the preference of each agent can be non-monotonic and non-differentiable. We only require the preferences to be rational:

Definition 1. *The ordinal preference $\precsim_i$ of agent A_i in the negotiation domain is rational if it satisfies the following conditions:*
1. Strict preference is asymmetric: There is no pair of x and x' in X such that $x \prec_i x'$ and $x' \prec_i x$;
2. Transitivity: For all x, x', and x'' in X, if $x \precsim_i x'$ and $x' \precsim_i x''$, then $x \precsim_i x''$;

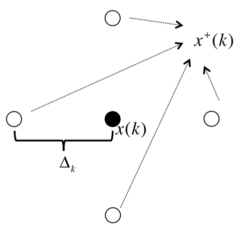


Fig. 1. An illustration of Generalized Pattern Search for unconstrained optimization

3. Completeness: For all x and x' in X, either $x \lesssim_i x'$ or $x' \lesssim_i x$;
where $x \lesssim_i x'$ (or $x \prec x'$) indicates that the offer x' is at least as good as (or better than) x for agent i.

The aim of the agents will be to reach an agreement on a contract x', maximizing their individual payoff and minimizing the revelation of private information.

Next, we describe in detail the GPS for unconstrained optimization, which is used in the construction of the negotiation protocol. GPS belongs to the family of *Direct Search Based* optimization algorithms [6]. Note, however, that our negotiation protocol is not a single-objetive or multi-objective centralized optimization process.

2.1 Generalized Pattern Search Algorithm for Unconstrained Optimization

The optimization problem can be defined as $\max f(x)$, where $f : \mathbb{R}^m \to \mathbb{R}$, $x \in \mathbb{R}^m$. At an iteration k of the protocol, we have an iterate $x(k) \in \mathbb{R}^m$ and a step-length parameter $\triangle_k > 0$. We successively look at the points in the *mesh* $x^+(k) = x(k) \pm \triangle_k e_j$, $j \in \{1, \ldots, m\}$, where e_j is the jth standard basis vector, to search for a contract $x'(k)$ in $x^+(k)$ for which $f(x'(k)) > f(x(k))$. We will use the notation $x^{+o}(k)$ to designate the mesh at round k including the current point $x(k)$. Fig. 1 illustrates the set of points among which we search for $m = 2$. This set of points or mesh is an instance of what we call a *pattern*, from which pattern search takes its name. If we find no $x'(k)$ such that $f(x'(k)) > f(x(k))$, then we reduce $\triangle_k$ by half and continue; otherwise, we leave the step-length parameter alone, setting $\triangle_{k+1} = \triangle_k$ and $x(k+1) = x'(k)$. In the latter case we can also increase the step-length parameter, say, by a factor of 2, if we feel a longer step might be justified. We repeat the iteration just described until $\triangle_k$ is deemed sufficiently small. One important feature of pattern search that plays a significant role in a global convergence analysis is that we do not need to have an estimate of the derivative of f at $x(k)$ so long as the search includes a sufficient set of directions to form a positive spanning set for the cone of feasible directions, which in the unconstrained case is all of $\mathbb{R}^m$. In the unconstrained case the set $\{\pm e_j | j = 1, \ldots, m\}$ satisfies this condition, the purpose of which is to ensure that if the current iterate is not a stationary point of the problem, so that we have at least one ascendent direction.

The set e_j is defined by the number of independent variables in the objective function m and the positive standard basis set. Two commonly used positive basis sets in pattern search algorithms are the maximal basis, with $2m$ vectors, and the minimal basis, with $m+1$ vectors. For example, if there are two independent variables in the optimization problem, the default for a $2m$ positive basis consists of the following pattern vectors: $e_1 = \{1, 0\}$, $e_2 = \{0, 1\}$ and $-e_1 = \{-1, 0\}$, $-e_2 = \{0, -1\}$. An $m + 1$ positive basis consists of the following standard basis set: $e_1 = \{1, 0\}$, $e_2 = \{0, 1\}$ and only a negative vector $-e_1 = \{-1, -1\}$. In our approach we will take the $2m$ positive basis. We will use the notation $x^{e_j}(k)|j = 1, \ldots, 2m$ to describe each point in a mesh, and $x(k)$ or $x^{e_0}(k)$ to designate the current point. For example, $x^{e_1}(k)$ specifies the contract generated by the current contract $x(k)$ and the vector e_1 for the current step-length $\triangle_k$, while $x^{e_{m+1}}(k)$ points to the negative version of $x^{e_1}(k)$.

2.2 Basic Operation of the Negotiation Protocol

The basic protocol of the proposed negotiation process is the following:

1. The mediator proposes a mesh from an initial contract $x^{ini}(1)$ for a step-length parameter $\triangle_1$. The point $x^{ini}(1)$ is randomly chosen by the mediator.
2. Each agent provides the mediator their preferences for the contracts in the current mesh x^{+o}, in terms of a mapping $S_i : X \rightarrow [0, 1]$ such that for example $S_i(x^{e_j}(k))$ indicates agent i's support for the alternative $x^{e_j}(k)$. An agent does not know the other agents' support for the contracts. Though agents are free to provide support values which are coincident or not with the corresponding private valuation function $V_i(x^{e_j}(k))$, in this work we will assume a perfect correspondence between both values.
3. The individual agent preferences for each contract are aggregated by the mediator to obtain the corresponding group preferences for each of the contracts in the mesh. We shall refer to this as the **aggregation of preferences** step.
4. The mediator decides which is the **preferred contract** in the mesh according to the group preferences for the different contracts.
5. Based on the **the preferred contract**, the mediator decides to **expand** or **contract** the mesh. Should a contraction make $\triangle_k$ small enough, the negotiation ends, otherwise go to step 2.

We assume that the negotiation process is such that a solution from X is always obtained. Negotiation may end when $\triangle_k$ is below a predefined threshold value or when a deadline expires. Essentially, the multi-agent negotiation is a dynamic process where at each stage of the process an agent provides a support measure determined by its underlying payoff function and any information available about the previous stages of the negotiation. The process of choosing the specific support for the different alternatives in a mesh at each round of the negotiations then constitutes a participating agent's strategy. An important consideration in an agent's determination of their strategy are the rules and procedures used in the negotiation process. In the following we shall describe the implementation of the negotiation process steps outlined above.

3 The Aggregation of Preferences

Here we look at the process where the mediator aggregates the individual support for the contracts in the mesh at round k. Our point of departure here is a collection of n agents and a set $x^{+o}(k)$ of contracts (mesh) given a current contract $x(k)$ at round k. We assume each agent has provided at round k her preference $S_i(x^{+o}(k))$ over the set $x^{+o}(k)$ such that it indicates the degree to which each agent A_i supports each contract. The mediator objective in this mediation step is to obtain a group preference function $G : x^{+o} \rightarrow [0, 1]$ which associates with each alternative $x^{e_j}(k) \in x^{+o}(k)$ a value $G(x^{e_j}(k)) = M(S_1(x^{e_j}(k)), \ldots, S_n(x^{e_j}(k)))$.

M is called the *mediation rule*, which describes the process of combining the individual preferences. The form of M can be used to reflect a desired mediation imperative or *consensus policy* for aggregating the preferences of the individual agents to get the mesh group preferences. M will guide the mediator in the expansion-contraction decisions in order to meet the desired type of agreements for the negotiation process.

The most widespread consensus policy found in the automated negotiation literature suggests using as an aggregation imperative a desire to satisfy *all* the agents. However, the policy of requiring that all the agents be satisfied by a solution may not be suitable for multi-agent preference aggregation, or simply the system may need to implement more sophisticated forms of aggregation.

We propose to use other mediation rules to improve the negotiation processes where either a quorum is not necessary or simply such quorum is not possible. For example, a solution may be acceptable if *most* of the agents support it. To incorporate these notions into our negotiation framework we will use a more general class of aggregation rules. The idea is to use a *quantifier guided aggregation*, which allows a natural language expression of the quantity of agents that need to agree on an acceptable solution. As we shall see the *Ordered Weighted Averaging* (OWA) operator [13] will provide a tool to model this kind of softer mediation rule.

3.1 OWA Operators

An aggregation operator $M : S^n \rightarrow G, (S, G \in [0, 1])$ is called an OWA operator of dimension n if it has an associated weighting vector $W = [w_1 w_2 \ldots w_n]$ such that $w_t \in [0, 1]$ and $\sum_{t=1}^{n} w_t = 1$ and where $M(S_1, \ldots, S_n) = \sum_{t=1}^{n} w_t b_t$ where b_t is the tth largest element of the aggregates $\{S_1, \ldots, S_n\}$.

Note that in the definition of OWA we have used the notation M to identify the aggregation operator with the mediation rule, S^n to make reference to the preferences of the agents, and G to define the group preference. In the OWA aggregation the weights are not directly associated with a particular argument but with the ordered position of the arguments. If *ind* is an index function such that $ind(t)$ is the index of the tth largest argument, then we can express $M(S_1, \ldots, S_n) = \sum_{t=1}^{n} w_t S_{ind(t)}$.

It can be shown the OWA operator is a mean operator. The form of the aggregation is dependent upon the associated weighting vector. We have a number of special cases of weighting vector which are worth noting. The vector W^* defined such that $w_1 = 1$ and $w_t = 0$ for all $t \neq 1$ gives us the aggregation $Max_i[S_i]$. Thus, it provides the largest possible aggregation. The vector W_* defined such that $w_n = 1$ and $w_t = 0$ for all $t \neq 1$ gives the aggregation $Min_i[S_i]$. The weighting vector W_{ave} defined such that $w_t = 1/n$ gives us the average $\frac{1}{n}\sum_{i=1}^{n} S_i$. Finally, an interesting family of OWA operators are the E-Z OWA operators. There are two families. In the first family we have $w_t = 1/q$ for $t = 1$ to q, and $w_t = 0$ for $t = q + 1$ to n. Here we are taking the average of the q largest arguments. The other family defines $w_t = 0$ for $t = 1$ to q, and $w_t = \frac{1}{n-q}$ for $t = q + 1$ to n. We can see that this operator can provide a softening of the original *min* and *max* mediation rules by modifying q.

3.2 Quantifier Guided Aggregation

In the preceding, we have seen how the OWA operators can be used to compute the group preference for different alternatives, in our case, the different contracts in the current mesh $x^{+o}(k)$. However, our aim is to define consensus policies in the form of a linguistic agenda for our mediation mechanisms. For example, the mediator should make decisions regarding the exploration of the negotiation space, i.e. expansion and contraction of the mesh, following mediation rules like *"Most* agents must be satisfied by the contract", *"at least* α agents must be satisfied by the contract","*many* agents must be satisfied", ...

The above statements are examples of *quantifier guided aggregations.* Zadeh [15] suggested a formal representation of these linguistic quantifiers using fuzzy sets. He suggested that any relative linguistic quantifier can be expressed as a fuzzy subset Q of the unit interval $I = [0, 1]$. In this representation for any proportion $y \in I$, $Q(y)$ indicates the degree to which y satisfies the concept expressed by the term Q. In most applications of the quantifier guided aggregation we use a special case class of these linguistic quantifiers, called *Regular Increasing Monotone* (RIM) quantifiers. These types of quantifiers have the property that as more agents are satisfied our overall satisfaction can't decrease. Formally, these quantifiers are characterized in the following way: 1) $Q(0) = 0$, 2) $Q(1) = 1$ and 3) $Q(x) \geq Q(y)$ if $x > y$. Examples of this kind of quantifier are *all*, *most*, *many*, *at least* α. Two examples of RIM quantifiers are *all,* which is represented by Q_* where $Q_*(1) = 1$ and $Q_*(x) = 0$ for all $x \neq 1$, and *any,* which is defined as $Q^*(0) = 0$ and $Q^*(x) = 1$ for all $x \neq 0$.

The question now is how to obtain the OWA operator to satisfy a quantifier guided aggregation. Again assume we have a collection of n agents. These agents have their preferences represented as fuzzy subsets over the set of alternatives in the mesh $\{S_1(x^{+o}(k)), \ldots, S_n(x^{+o}(k))\}$. Under the quantifier guided mediation approach a group mediation protocol is expressed in terms of a linguistic quantifier Q indicating the proportion of agents whose agreement if necessary for a

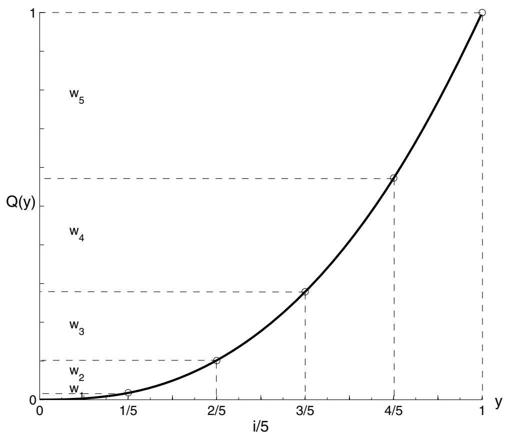


Fig. 2. Example of how to obtain the weights from a quantifier for $n = 5$ agents

solution to be acceptable. The basic form of the mediation rule in this approach is that Q agents must be satisfied by the contract, where Q is a quantifier.

The formal procedure used to implement this mediation rule is described in the following. The quantifier Q is used to generate an OWA weighting vector W of dimension n. This weighting vector is then used in an OWA aggregation to determine the group support for the contract. For each contract in the mesh the argument of this OWA aggregation is the degree of support for that contract by each of the agents, $S_i(x^{e_j}(k))$, $i = 1, \ldots, n$. Thus, the process used in the quantifier guided aggregation is as follows:

1. Use Q to generate a set of OWA weights, $w_1, \ldots, w_n$.
2. For each contract $x^{e_j}(k)$ in $x^{+o}(k)$ calculate the overall group support:

$$G(x^{e_j}(k)) = M(S_1(x^{e_j}(k)), \ldots, S_n(x^{e_j}(k))).$$

The procedure used for generating the weights from the quantifier is to divide the unit interval into n equally spaced intervals and then to compute the length of the mapped intervals using Q

$$w_t = Q(\frac{t}{n}) - Q(\frac{t-1}{n}) \, \text{for} \, t = 1, \ldots, n \, .$$

Because of the nondecreasing nature of Q it follows that $w_t \geq 0$. Furthermore from the regularity of Q, $Q(1) = 1$ and $Q(0) = 0$, it follows that $\sum_t w_t = 1$. Thus we can see that the weights generated are an acceptable class of OWA weights.

In Fig. 2 we show an example of a RIM linguistic quantifier and illustrate the process of determining the weights from the quantifier. We see that the weights depend on the number of agents as well as the form of Q. In Fig. 3 we show the functional form for the quantifiers *all*, *any*, Q_*, Q^*, *at least* α

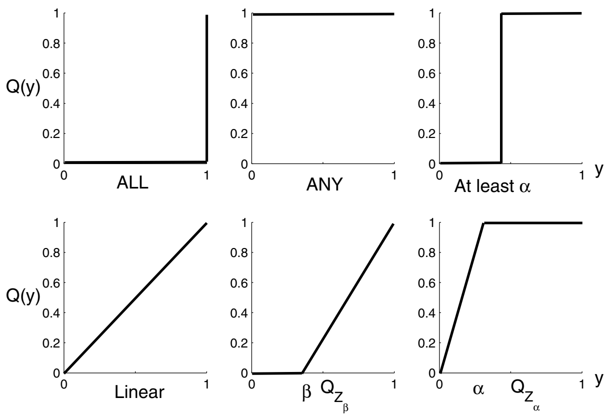


Fig. 3. Functional form of typical quantifiers: all, any, at least, linear, piecewise linear Q_{Z_β} and piecewise linear Q_{Z_α}

percent, *linear quantifier*, *piecewise* Q_{Z_β} and *piecewise* Q_{Z_α}. The quantifiers *all*, *any* and *at least* α describe the consensus policy using a natural language verbal description. However, more generally any function $Q : [0,1] \rightarrow [0,1]$ such that $Q(x) \geq Q(y)$ for $x \geq y$, $Q(1) = 1$ and $Q(0) = 0$ can be seen to be an appropriate form for generating mediation rules or consensus policies. Thus there are two techniques to generating these quantifier based mediation rules. One possibility is to start with a linguistic expression and then obtain Q. The second approach is to allow the mediation rule to be directly expressed in terms of a function Q. One important characteristic of this second method is that we can easily introduce into our mediation a number of formal properties that are not very easily expressed using a verbal description of the quantifier. The linear quantifier $Q(y) = y$ for instance generates $w_t = 1/n$, and thus, all the agents get the same weight. The Q_{Z_β} quantifier it is required that at least β agents are satisfied to initiate a Q linear improvement. Q_{Z_α} initiates the Q linear improvement with the first satisfied agent, and once there are α agents satisfied there is no improvement in Q if more agents are satisfied.

One feature which distinguishes the different types of mediation rules is the power of an individual agent to eliminate an alternative. For example, in the case of *all* this power is complete. In order to capture this idea the *Value Of Individual Disapproval* (VOID)

$$VOID(Q) = 1 - \int_0^1 Q(y)dy$$

measures this power. For the *all*, *any, at least* α and *linear* quantifiers the VOID measures are respectively 1, 0, α and 0.5. For the Q_{Z_β} quantifier, $VOID(Q_{Z_\beta})$ $= \frac{1}{2} + \frac{\beta}{2})$ and therefore $VOID(Q_{Z_\beta}) \in [0.5, 1]$. Similarly, the Q_{Z_α} quantifier gets $VOID(Q_{Z_\alpha}) = \frac{\alpha}{2}$ and $VOID(Q_{Z_\alpha}) \in [0, 0.5]$.

Another family of quantifiers are those defined by $Q_p(y) = y^p$ for $p > 0$. In this case $VOID(Q_p) = 1 - \int_0^1 r^p dr = \frac{p}{p+1}$. For this quantifier we can easily obtain the OWA weights with

$$w_t = \left(\frac{t}{n}\right)^p - \left(\frac{t-1}{n}\right)^p .$$

For Q_p we see that as p increases we get closer to the *min* and that as p gets closer to zero we get the *max*.

4 The Search Process

The search process is based on a mechanism whereby the mediator decides whether to generate a new mesh in order to continue with a new negotiation round, or to finish the negotiation process. This process starts just after any aggregation of preferences process, when the mediator has determined the group preferred contract $x^{e_*}(k)$. The relevant information available to the mediator at this point is at least the group preference $G(x^{+o}(k))$, the preferred contract $x^{e_*}(k)$, the current step-length $\triangle_k$, and the current round number k. With this information, the mediator has to select among three possible alternatives:

1. Move to the group preferred contract $x(k+1) = x^{e_*}(k)$ in $x^+(k)$ and expand the mesh by a factor of two $\triangle_{k+1} = 2 \cdot \triangle_k$.
2. Keep the current contract $x(k+1) = x(k)$ and reduce by half the mesh step-length $\triangle_{k+1} = \triangle_k/2$.
3. Finish the negotiation process.

For this paper we will assume what we call the *Standard Search Process* which selects among the mentioned alternatives as follows. The mediator selects alternative 1 if the preferred contract is in $x^+(k)$, i.e., $x^{e_*}(k) \in x^+(k)$. If the preferred contract is $x(k)$ then the mediator selects alternative 2. Finally, we define two stopping rules, one which bounds the maximum number of rounds k_{max}, and a second one which stops negotiation when the step-length $\triangle_k$ is below a predefined threshold γ. We assume that in both cases the agreement reached is the preferred group contract in the last negotiation round.

4.1 Preferred Contract Selection in the Search Process

Here the mechanisms used to select the preferred contract are described in detail. The point of departure is the set of final group preferences for the contracts in $x^{+o}(k)$ at round k. We propose a probabilistic selection process to select the winner contract in the mesh at a round k. We associate with each contract $x^{e_j}(k) \in x^{+o}(k)$ a probability

$$P(x^{e_j}(k)) = \frac{G(x^{e_j}(k))^\sigma}{\sum_j G(x^{e_j}(k))^\sigma} .$$

The process selects the winner contract using a biased random experiment with these probabilities. The parameter $\sigma > 0$ works as an indication of the significance we give to the final group preferences. If $\sigma \to \infty$ we select the contract with the maximum support, which means that the mediator is given the higher significance to the group preferences. If $\sigma = 1$ then the probability of selecting $x^{e_j}(x)$ would be proportional to its group support.

The rationale behind using this probabilistic process is to introduce randomness and avoid local optima in the following way. With G the mediator is able to select a contract within the mesh. However, this selection is based on a relative measurement and it is not considering how good is the selection made. The mediator must consider both the G value and the relative values to make the decision of expansion and contraction. Thus, we make σ vary as a function of G and the number of rounds k. If G is high, σ must be high, favoring a deterministic mesh movement, i.e. with a high probability the contract with a higher G is selected. Otherwise, if G is low, σ must be low to induce randomness and avoid local optima. More specifically, for $\sigma = 0$ the selection of contracts is equiprobable, making such selection independent of G. For $\sigma = 1$ the selection probability is proportional to G. Higher values for σ increases the probability of choosing the contract with a higher G. To control σ we define

$$\sigma(k, G) = \sigma_{min} + (\sigma_{max} - \sigma_{min}) \cdot G^{(1-\frac{k}{k_{max}})\cdot\alpha} ,$$

where σ depends on the negotiation round k, the maximum number of rounds k_{max} and G. The function is bounded by σ_{max} and σ_{min} given $G = 0$ and $G = 1$ respectively. The parameter $\alpha > 0$ determines the curvature of $\sigma(k, G)$. As the number of rounds k increases, the function increases its concaveness, which means that G induces higher values for σ, favoring convergence. The principle of this approach is analogous to the simulated annealing technique [4] without reannealing. We can also introduce reannealing for $k_r < k_{max}$ such that k/k_{max} converts into $\frac{k-k_r}{k_{max}-k_r}$.

5 Experimental Evaluation

In this section, we test our negotiation framework and show that the mechanisms proposed provide the mediator the tools to efficiently conduct multiagent negotiations by considering different consensus policies.

In the experimental setup, without loosing generality, we have considered 7 agents, 2 issues and 2 different types of negotiation spaces: a negotiation space where agents' utility functions are strategically built to define a *proof of concept negotiation scenario*, and a *complex negotiation scenario* where utility functions exhibit a more complex structure. In both cases utility functions are built using an aggregation of *Bell functions*. This type of utility functions capture the intuition that agents' utilities for a contract usually decline gradually with distance from their ideal contract. Bell functions are ideally suited to model, for instance, spatial and temporal preferences [11,8]. In addition, they allow to configure different negotiation scenarios in terms of different complexity degrees.

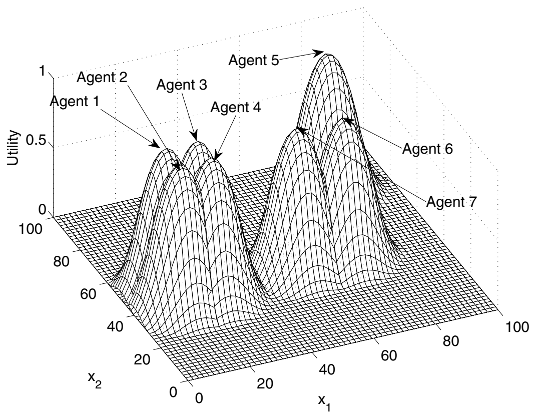


Fig. 4. Utility functions for the *proof of concept negotiation scenario*

Definition 2. *A* Bell *is defined by a center c, height h, and a radius r. Let* $\| s - c \|$ *be the euclidean distance from the center c to a contract s, then the* Bell function *is defined as*

$$fbell(s,c,h,r) = \begin{cases} h - 2h\frac{\|s-c\|^2}{r^2} & if \ \| s - c \| < r/2, \\ \frac{2h}{r^2}(\| s - c \| - r)^2 & if \, r > \| s - c \| \geq r/2 \, , \\ 0 & \| s - c \| \geq r \end{cases}$$

and the Bell utility function *as*

$$U_{b,s}(s) = \sum_{i}^{nb} fbell(s, c_i, h_i, r_i) \, ,$$

where nb is the number of generated bells. The complexity of the negotiation space can be modulated by varying c_i*,* h_i*,* r_i *and nb.*

In the *proof of concept negotiation scenario* each agent has a utility function with a single optimum. Fig. 4 shows in the same graph the agents' utility functions in the bidimensional negotiation space $[0, 100]^2$. In this scenario four agents (Agent 1, 2, 3, 4) are in weak opposition (i.e. their preferences are quite similar), Agents 6 and 7 are in weak opposition and in very strong opposition with respect the other agents, and Agent 5 is in very strong opposition with respect the rest of the agents. In the *complex negotiation scenario* each agent's utility function is generated using two randomly located bells. The radius and height of each bell are randomly distributed within the ranges $r_i \in [20, 35]$ and $h_i = [0.1, 1]$. Fig. 5 shows the utility functions generated for each agent in this second case.

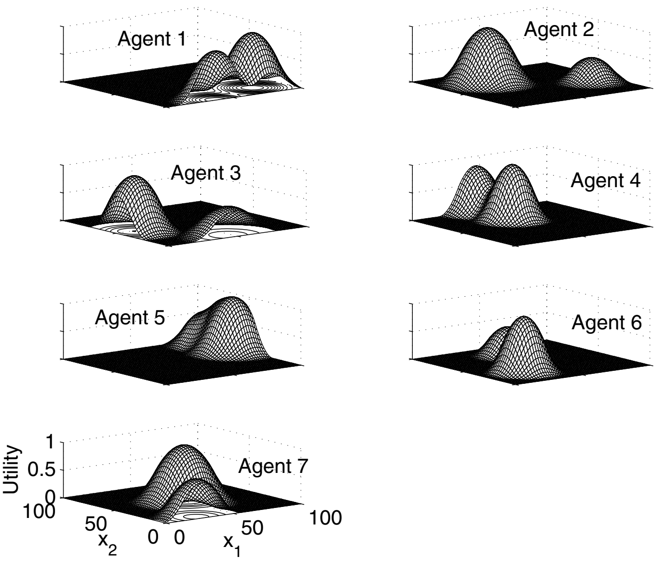


Fig. 5. Utility functions for the *complex negotiation scenario*

The configuration of parameters in the mediator is: $k_{max} = 50$ rounds, mesh tolerance $1e - 6$, and $\alpha = 2$, $\sigma_{min} = 1$, $\sigma_{max} = 200$ for the preferred contract selection process. Previous experiments have confirmed that these parameter values perform well under most negotiation scenarios.

We tested the performance of the protocol under the proof of concept and complex negotiation scenarios for 5 different consensus policies defined by the corresponding VOID degrees: 0, 0.25, 0.5, 0.75 and 0.95, using the quantifier $Q_p(y) = y^p$. We also define a contrast experiment where the consensus policy based mediation process is deactivated, such that the mediator uses the pattern search based process but there is no randomness and the group preference evaluation is limited to compute the sum of agents' valuations for a given contract (i.e. the winner contract is that with the highest sum of valuations).This experiment uses also 50 rounds and a mesh tolerance $1e - 6$.

Each experiment consists of 100 negotiations where we capture the utilities achieved by each agent. To analyze the results we first build a 7 agents $\times$ 100 negotiations utility matrix where each row provides each agent's utilities and each column is a negotiation. The matrix is then reorganized such that each column is individually sorted from higher to lower utility values. Note that after this transformation the association row/particular-agent disappears. Given the matrix, we form 7 different utility groups: a first group named *group level 1* where we take the highest utility from each negotiation (i.e. the first row), a second group named *group level 2* with the two first rows and so on. In order to show the performance of the protocol we have used the Kaplan-Meier estimate

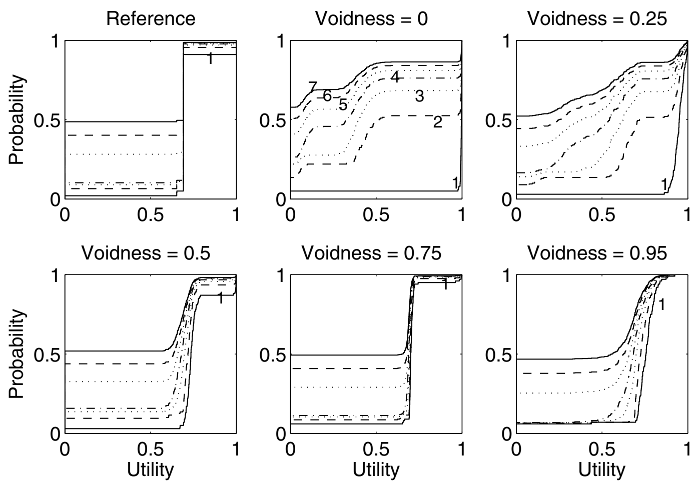


Fig. 6. Cumulative distributions of utilities for the *proof of concept scenario*

of the cumulative distribution function (*cdf*) of agents' utilities for each group. Thus, we compute the *cdf* for the highest utilities, for the two highest utilities and so on. The *cdf* estimates the probability of finding agent's utilities below a certain value. The rationale behind using grouping in the analysis is to evaluate the ability of the protocol to find solutions which satisfy groups of agents.

In the proof of concept scenario (see Fig. 4) it can be seen that when a quorum is needed, the best alternative is to get satisfied agents 1, 2, 3 and 4. If it is enough to have one agent satisfied, any of the utility peaks would be a good solution. In Fig. 6 we show the results for the proof of concept scenario. Each line shows the *cdf* for a group level and the number above each line identifies the corresponding level. For instance, for the reference experiment and the group level 1 there is approximately a 98% probability of having agents with a utility 0.7, and a 2% probability of having agents with utility 0. In the group level 7 case, there is a 50% probability of having agents with utility 0.7, and a 50% probability of having agents with utility 0. For a VOID=0 and group level 1, however, the probability of having agents with a utility 1 is around 98%, which means that the mediator is applying efficiently the consensus policy which states that it is good enough to have one agent satisfied. As VOID increases (i.e. as it is necessary to have more agents satisfied) the *cdf* for group level 1 performs worse, though better than in the reference scenario, and for higher group levels the performance increases.

In Fig. 7 are shown the results for the complex negotiation scenario. Here we can also see how as VOID increases, the mediator biases the search for agreements where more agents are satisfied at the expense of not having individual agents highly satisfied. Globally, the results show that the proposed mechanisms are able to focus the negotiation process in terms of consensus policies and to obtain better results than when using a classical welfare maximization approach.

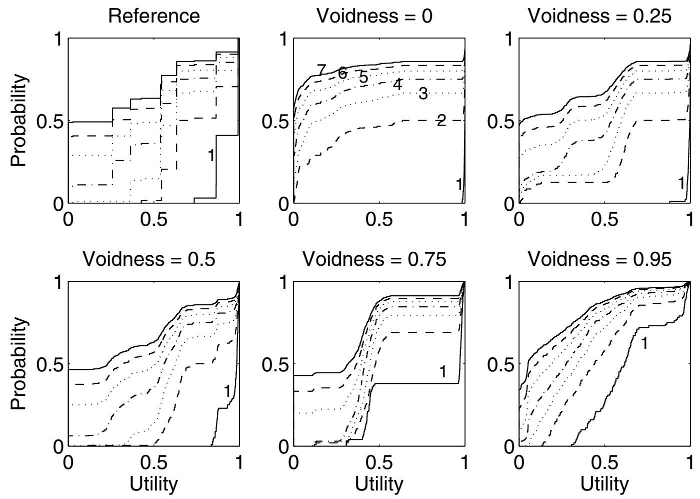


Fig. 7. Cumulative distributions of utilities for the *complex negotiation scenario*

6 Conclusion

The main hypothesis of our work is that the consensus type by which an agreement meets in some specific manner the concerns of all the negotiators should be considered in the construction of multiparty negotiation protocols. We argue that there exist situations where an unanimous agreement is not possible or simply the rules imposed by the system may not seek such unanimous agreement. Thus, we develop a consensus policy based mediation framework to perform multiparty negotiations. The mediation mechanisms proposed to perform the exploration of negotiation space in the multiparty negotiation setting are derived from the Generalized Pattern Search non-linear optimization technique. The exploration performed in the mediator is guided by the aggregation of the agent preferences on the set of alternatives the mediator proposes in each negotiation round. The mediation rules at the mediator may take the form of a linguistic description of the type of agreements needed. We showed empirically that CPMF efficiently manages negotiations following predefined consensus policies and solves situations where unanimous agreements are not viable.

We believe that the negotiation framework presented opens the door to a new set of negotiation algorithms where consensus criteria will play an important role. However, strategical issues remains open. We have assumed that agents reveal their true valuations. It is expected that the performance of the protocol deviates from the optimal if agents act strategically. Thus, strategy issues need to be evaluated, and mechanisms need to be implemented to avoid or mitigate incentive compatibility problems.

Acknowledgments. This work has been supported by the Spanish Ministry of Education and Science grants: TIN2008-06739-C04-04 and the mobility grant JC2010-003.

References

1. Ehtamo, H., Hamalainen, R.P., Heiskanen, P., Teich, J., Verkama, M., Zionts, S.: Generating pareto solutions in a two-party setting: constraint proposal methods. Management Science 45(12), 1697–1709 (1999)
2. Heiskanen, P., Ehtamo, H., Hamalainen, R.P.: Constraint proposal method for computing pareto solutions in multi-party negotiations. European Journal of Operational Research 133(1), 44–61 (2001)
3. Ito, T., Klein, M., Hattori, H.: A multi-issue negotiation protocol among agents with nonlinear utility functions. Journal of Multiagent and Grid Systems 4(1), 67–83 (2008)
4. Klein, M., Faratin, P., Sayama, H., Bar-Yam, Y.: Protocols for negotiating complex contracts. IEEE Intelligent Systems 18(6), 32–38 (2003)
5. Lai, G., Sycara, K.: A generic framework for automated multi-attribute negotiation. Group Decision and Negotiation 18, 169–187 (2009)
6. Lewis, R.M., Torczon, V., Trosset, M.W.: Direct search methods: then and now. Journal of Computational and Applied Mathematics 124, 191–207 (2000)
7. Li, M., Vo, Q.B., Kowalczyk, R.: Searching for fair joint gains in agent-based negotiation. In: Decker, Sichman, Sierra, Castelfranchi (eds.) Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009), Budapest, Hungary, May, 10-15, pp. 1049–1056 (2009)
8. Lopez-Carmona, M.A., Marsa-Maestre, I., De La Hoz, E., Velasco, J.R.: A region-based multi-issue negotiation protocol for nonmonotonic utility spaces. Computational Intelligence 27(2), 166–217 (2011)
9. Lopez-Carmona, M.A., Marsa-Maestre, I., Ibanez, G., Carral, J.A., Velasco, J.R.: Improving trade-offs in automated bilateral negotiations for expressive and inexpressive scenarios. Journal of Intelligent & Fuzzy Systems 21, 165–174 (2010)
10. Lopez-Carmona, M.A., Marsa-Maestre, I., Klein, M., Ito, T.: Addressing stability issues in mediated complex contract negotiations for constraint-based, nonmonotonic utility spaces. Journal of Autonomous Agents and Multiagent Systems, 1–51 (2010)
11. Marsa-Maestre, I., Lopez-Carmona, M.A., Velasco, J.R., Ito, T., Klein, M., Fujita, K.: Balancing utility and deal probability for auction-based negotiations in highly nonlinear utility spaces. In: 21st International Joint Conference on Artificial Intelligence (IJCAI 2009), Pasadena, California, USA, pp. 214–219 (July 2009)
12. Vo, Q.B., Padgham, L., Cavedon, L.: Negotiating flexible agreements by combining distributive and integrative negotiation. Intelligent Decision Technologies 1(1-2), 33–47 (2007)
13. Yager, R.: Quantifier guided aggregation using owa operators. International Journal of Intelligent Systems 11, 49–73 (1996)
14. Yager, R., Kacprzyk, J.: The Ordered Weighted Averaging Operators: Theory and Applications. Kluwer (1997)
15. Zadeh, L.: A computational approach to fuzzy quantifiers in natural languages. Computing and Mathematics with Applications 9, 149–184 (1983)

Distributed Lagrangian Relaxation Protocol for the Over-constrained Generalized Mutual Assignment Problem

Kenta Hanada and Katsutoshi Hirayama

Kobe University
5-1-1 Fukaeminami-machi, Higashinada-ku, Kobe 658-0022, Japan
`kenta_hanada@stu.kobe-u.ac.jp, hirayama@maritime.kobe-u.ac.jp`

Abstract. *The Generalized Mutual Assignment Problem* (GMAP) is a distributed combinatorial optimization problem in which, with no centralized control, multiple agents search for an optimal assignment of goods that satisfies their individual knapsack constraints. Previously, in the GMAP protocol, problem instances were assumed to be feasible, meaning that the total capacities of the agents were large enough to assign the goods. However, this assumption may not be realistic in some situations. In this paper, we present two methods for dealing with such "over-constrained" GMAP instances. First, we introduce a *disposal agent* who has an unlimited capacity and is in charge of the unassigned goods. With this method, we can use any off-the-shelf GMAP protocol since the disposal agent can make the instances feasible. Second, we formulate the GMAP instances as an Integer Programming (IP) problem, in which the assignment constraints are described with inequalities. With this method, we need to devise a new protocol for such a formulation. We experimentally compared these two methods on the variants of *Generalized Assignment Problem* (GAP) benchmark instances. Our results indicate that the first method finds a solution faster for fewer over-constrained instances, and the second finds a better solution faster for more over-constrained instances.

Keywords: generalized mutual assignment problem, distributed optimization, Lagrangian relaxation.

1 Introduction

Obviously, in distributed AI, the distributed assignment, whose task is to assign something in a distributed context, has been a fundamental problem for decades. The *contract net protocol* [13] may be the oldest example that performs this task. More recently, *multi-robot task allocation* [3] and *distributed target tracking* [2] have attracted much attention as applications of this technology. For more formal treatment, *distributed constraint optimization* [10] and *distributed facility location* [4] seem popular as a basis for the reasoning or the optimization of distributed assignments.

To formally deal with complex assignment problems, Hirayama et al. proposed the *Generalized Mutual Assignment Problem* (GMAP) and the *Distributed Lagrangian Relaxation Protocols* (DisLRP) [6][7] [8]. GMAP is the distributed version of the *Generalized Assignment Problem* (GAP), whose goal is to find the most profitable assignment of goods to agents. In GMAP, the agents themselves cooperatively search for such an assignment. We regard this problem as a set partitioning problem by agents, each of whom has a resource constraint.

Here is an outline of agent behavior in DisLRP. First, the agents solve their individual 0-1 knapsack problems and announce their assignments of goods to their respective neighbors. Second, for all goods, the agents raise their *price* (Lagrange multiplier) if it is chosen by two or more agents, and they reduce their price if it is not chosen by any agent. Third, under the new prices, the agents solve their individual new 0-1 knapsack problems again. The agents repeat this procedure until all of the goods are chosen by exactly one agent, which means we get a proper set partition for the entire set of goods.

Previously, we assumed that, in GMAP, the total capacity of agents is large enough to assign the goods. However, we can easily imagine "over-constrained" situations, where the agents don't have enough resource capacities for the entire set of goods. We develop two methods to deal with such an over-constrained situation and experimentally compare them using the variants of GAP benchmark instances.

The basic idea of the first method is the introduction of an additional agent without a knapsack constraint, or equivalently, one with infinite capacity. We call this agent the *disposal agent*. The disposal agent assigns the goods that have not been chosen by any agent. By this method, since we do not need to change the existing GMAP formulation, we have an advantage because the off-the-shelf DisLRP can be used without any modifications. On the other hand, a basic idea of the second method is that we first formulate GMAP as an Integer Programming (IP) problem in which the assignment constraints are described with inequalities and relax them to decompose the problem. By this method, we do not need to introduce a special agent like the disposal agent, but we do need to adapt DisLRP to this new formulation.

The remainder of this paper is organized as follows. First, we define GMAP in Section 2. In Section 3, we present our two methods for dealing with over-constrained GMAP instances, each of which consists of the formulation of a problem and a solution. Next, we show the results of experiments on the variants of the GAP benchmark instances in Section 4 and conclude in Section 5.

2 Generalized Mutual Assignment Problem

GAP has been studied for many years in operations research. Since it is a NP-hard problem, many exact algorithms [11] and heuristic approaches [14] have been proposed in centralized contexts. GMAP is a distributed version of GAP. In the entire system, agents on GMAP solve the following IP problem, denoted as $\mathcal{GAP}$:

$$
\begin{aligned}
\mathcal{GAP} \quad & (\text{decide } x_{kj},\ \forall k \in A,\ \forall j \in J): \\
\text{max.} \quad & \sum_{k \in A} \sum_{j \in J} p_{kj} x_{kj} \\
\text{s. t.} \quad & \sum_{k \in A} x_{kj} = 1,\ \ \forall j \in J, & (1) \\
& \sum_{j \in J} w_{kj} x_{kj} \le c_k,\ \ \forall k \in A, & (2) \\
& x_{kj} \in \{0,1\},\ \forall k \in A,\ \forall j \in J,
\end{aligned}
$$

where $A = \{1, ..., m\}$ is a set of agents, $J = \{1, ..., n\}$ is a set of goods, and p_{kj} and w_{kj} are the profit and the amount of resources required when agent k selects goods j. c_k is the capacity, i.e., the amount of available resources, of agent k. x_{kj} is a decision variable whose value is set to 1 when agent k selects goods j and 0 otherwise. The goal of the problem is to maximize the summation of profits under the assignment constraints (1), which means each good is assigned to exactly one agent and the knapsack constraints (2), which means no agent can use more resources than its capacity.

To solve this problem by using a distributed method, we have to divide the problem while keeping its structure. The *Lagrangian decomposition* [5] provides such decomposition of the problem. The Lagrangian relaxation problem is obtained by dualizing the assignment constraints (1) of $\mathcal{GAP}$ as follows:

$$
\begin{aligned}
L(\mu) = \text{max.} \quad & \sum_{k \in A} \sum_{j \in J} p_{kj} x_{kj} \\
& + \sum_{j \in J} \mu_j \left(1 - \sum_{k \in A} x_{kj} \right) \\
\text{s. t.} \quad & \sum_{j \in J} w_{kj} x_{kj} \le c_k,\ \forall k \in A, \\
& x_{kj} \in \{0,1\},\ \forall k \in A,\ \forall j \in J,
\end{aligned}
$$

where μ_j is a real-valued parameter called a *Lagrange multiplier* for goods j and vector $\mu = (\mu_1, \mu_2, ..., \mu_n)$ is called a *Lagrange multiplier vector*. For any value of μ, $L(\mu)$ provides an upper bound on the optimal value of $\mathcal{GAP}$[1].

Since an upper bound should be the lowest, we have another minimization problem on μ:

$$\text{min.} \ \ L(\mu).$$

We usually call this the *Lagrangian dual problem*. In $L(\mu)$, the objective function is additive over the agents and the constraints are separable over them; this maximization can be achieved by solving the following subproblems: for each agent k,

$$L_k(\mu) = \text{max.} \ \sum_{j \in J} (p_{kj} - \mu_j) x_{kj}$$

$$\text{s. t. } \sum_{j \in J} w_{kj} x_{kj} \le c_k,$$
$$x_{kj} \in \{0,1\},\ \forall j \in J,$$

and for the remaining terms,

$$L_{const}(\mu) = \sum_{j \in J} \mu_j,$$

We can thus describe the Lagrangian dual problem as follows:

$$\text{min. } \sum_{k \in A} L_k(\mu) + L_{const}(\mu),$$

Our distributed solution method solves this problem using only local communications among agents.

3 Solutions for the Over-constrained Problem

Basically, GAP and GMAP are supposed to be feasible, meaning a proper set partition of the goods exists that does not violate the knapsack constraints. However, in reality, we may face over-constrained situations, where the agents do not have enough capacity for the entire set of goods. In this paper, we present two methods for dealing with such over-constrained situations.

3.1 DisLRP with a Disposal Agent

The first method introduces an additional agent, called a disposal agent, who has no knapsack constraint or is equivalently equipped with infinite capacity. The disposal agent does not get any profit even if he has some goods. Among the regular agents and the disposal agent, all goods must be assigned to exactly one agent.

Formulation. We can formulate GMAP including the disposal agent, denoted by $d \notin A$, as follows:

$$\begin{aligned}
\mathcal{GAP}' \quad & (\text{decide } x_{kj},\ \forall k \in A \cup \{d\},\ \forall j \in J): \\
\text{max.} \quad & \sum_{k \in A} \sum_{j \in J} p_{kj} x_{kj} \\
\text{s. t.} \quad & \sum_{k \in A \cup \{d\}} x_{kj} = 1,\ \ \forall j \in J, \\
& \sum_{j \in J} w_{kj} x_{kj} \le c_k,\ \ \forall k \in A, \\
& x_{kj} \in \{0,1\},\ \forall k \in A \cup \{d\},\ \forall j \in J.
\end{aligned}$$

For $\mathcal{GAP}'$, the Lagrangian relaxation problem that dualizes the assignment constraints is described as follows:

$$\begin{aligned} L'(\mu) = \max.\ & \sum_{k \in A} \sum_{j \in J} p_{kj} x_{kj} \\ & + \sum_{j \in J} \mu_j \left(1 - \sum_{k \in A \cup \{d\}} x_{kj} \right) \\ \text{s. t. } & \sum_{j \in J} w_{kj} x_{kj} \le c_k,\ \forall k \in A, \\ & x_{kj} \in \{0,1\},\ \forall k \in A \cup \{d\},\ \forall j \in J, \end{aligned} \tag{3}$$

For any value of μ, $L'(\mu)$ provides an upper bound on the optimal value of $\mathcal{GAP}'$.

Since an upper bound should be the lowest, we have another minimization problem on μ:

$$\min.\ \ L'(\mu).$$

We usually call this the *Lagrangian dual problem.* Since, in $L'(\mu)$, the objective function is additive over the agents and the constraints (3) are separable over the agents, this maximization can be achieved by solving the following subproblems: for each regular agent k,

$$\begin{aligned} L'_k(\mu) = \max.\ & \sum_{j \in J} (p_{kj} - \mu_j) x_{kj} \\ \text{s. t. } & \sum_{j \in J} w_{kj} x_{kj} \le c_k, \\ & x_{kj} \in \{0,1\},\ \forall j \in J, \end{aligned}$$

for disposal agent d,

$$\begin{aligned} L'_d(\mu) = \max.\ & \sum_{j \in J} (-\mu_j) x_{dj} \\ \text{s. t. } & x_{dj} \in \{0,1\},\ \forall j \in J, \end{aligned}$$

and for the remaining terms,

$$L'_{const}(\mu) = \sum_{j \in J} \mu_j.$$

We can thus describe the Lagrangian dual problem as follows:

$$\min.\ \sum_{k \in A} L'_k(\mu) + L'_d(\mu) + L'_{const}(\mu).$$

Our distributed solution method solves this problem using only local communications among agents including the disposal agent.

Virtualization of the Disposal Agent. Since we added the disposal agent in this method, communication costs would increase because the total number of agents increases by one. To avoid this scenario, we can virtualize the disposal agent in our real implementation. Looking at the subproblem of the disposal agent $L'_d(\mu)$, we realize that the disposal agent assigns goods j if μ_j becomes lower than zero. Therefore, the regular agents should know what goods are assigned to the disposal agent by using Lagrange multiplier μ. Consequently, we need not to add the *real* disposal agent. Instead, the regular agents can simulate the behavior of the disposal agent. We will describe this method more concretely below.

Solution. The basic procedure of this protocol is as follows:

(Step 1) All agents initialize their Lagrange multiplier vectors as 0.
(Step 2) Under a current value of μ, each agent k solves his knapsack problem to compute $L'_k(\mu)$. Then the agents send those results to their respective neighbors. At the same time, each agent treats every good j whose μ_j is lower than zero as if it is assigned to the disposal agent.
(Step 3) If all assignment constraints of the original problem are satisfied, the agents can quit the procedure to provide an optimal solution.
(Step 4) Each agent k finds an upper bound and a lower bound. Agent k also finds the smallest upper bound, *BestUB*, and the largest lower bound, *BestLB*, among those found so far. If both *BestUB* and *BestLB* have the same value, the agents can quit the procedure to provide an optimal solution.
(Step 5) Each agent k updates the Lagrange multiplier vector from $\mu^{(t)}$ to $\mu^{(t+1)}$ by the subgradient method [1] and returns to Step 2.

After initialization at Step 1, this procedure iterates Steps 2 through 5. We refer to one iteration as a *round*.

Note that the global information of the entire system is required for computing in Steps 3, 4, and 5. In this work, we use a *spanning tree* to collect this global information, as proposed in [8].

In Step 4, we need to compute both an upper and a lower bound. The upper bound can be computed, at each round, as the total sum of the optimal values of agents (including the disposal agent) plus the total sum of the elements of μ. On the other hand, the lower bound can be computed, at each round, by building a feasible solution for $\mathcal{GAP}'$ out of a solution for $L'(\mu)$. More specifically, a feasible solution is built as follows:

– If a good is assigned to exactly one agent, it will be assigned to the agent.
– If a good is assigned to two or more agents, it will be assigned to the agent among those agents having the largest profit.
– If a good is assigned to no agent, it will be assigned to the disposal agent.

In Step 5, we use the subgradient method to update Lagrange multiplier μ_j. In this method, agent k computes subgradient $g_j^{(t)}$ for all goods j by

$$g_j^{(t)} \leftarrow 1 - \sum_{i \in A \cup \{d\}} x_{ij}$$

and updates Lagrange multiplier μ_j as follows:

$$\mu_j^{(t+1)} \leftarrow \mu_j^{(t)} - \frac{\pi^{(t)}(BestUB^{(t)} - BestLB^{(t)})g_j^{(t)}}{\sum_{j \in J}(g_j^{(t)})^2}.$$

In this rule, agent k needs to know *BestUB* and *BestLB* at this point, but they are obviously global information. As we mentioned before, we use the same *spanning tree* as [8] to collect such global information. π is a control parameter, whose initial value is two, that halves itself if neither *BestUB* nor *BestLB* is updated through 30 consecutive rounds.

3.2 DisLRP with Inequality-Based Formulation

The second method describes assignment constraints with inequalities instead of equalities.

Formulation. We can formulate the entire problem as the following IP problem:

$$\begin{aligned}
\mathcal{GAP}'' \quad & (\text{decide } x_{kj},\ \forall k \in A,\ \forall j \in J): \\
\text{max.} \quad & \sum_{k \in A} \sum_{j \in J} p_{kj} x_{kj} \\
\text{s. t.} \quad & \sum_{k \in A} x_{kj} \leq 1, \ \ \forall j \in J, \qquad (4) \\
& \sum_{j \in J} w_{kj} x_{kj} \leq c_k, \ \ \forall k \in A, \\
& x_{kj} \in \{0, 1\},\ \forall k \in A,\ \forall j \in J.
\end{aligned}$$

The difference between $\mathcal{GAP}$ and $\mathcal{GAP}''$ is that assignment constraints are described with equalities in $\mathcal{GAP}$ and inequalities in $\mathcal{GAP}''$. Clearly, with this "relaxation," we allow all goods to be assigned to no more than one agent.

The Lagrangian relaxation problem is obtained by dualizing the assignment constraints (4) of $\mathcal{GAP}''$ as follows:

$$\begin{aligned}
L''(\mu) = \text{max.} & \sum_{k \in A} \sum_{j \in J} p_{kj} x_{kj} \\
& + \sum_{j \in J} \mu_j \left(1 - \sum_{k \in A} x_{kj} \right)
\end{aligned}$$

$$\begin{aligned} \text{s. t. } & \sum_{j \in J} w_{kj} x_{kj} \le c_k, \ \forall k \in A, \\ & x_{kj} \in \{0,1\}, \ \forall k \in A, \ \forall j \in J, \\ & \mu_j \ge 0, \ \forall j \in J. \end{aligned}$$

Note that in the above Lagrange multiplier vector μ_j has a nonnegative constraint, since the assignment constraint on all goods j is described with inequality. Similar to the previous section, the Lagrangian dual problem is the following minimization problem on non-negative real vector space and gives an upper bound on the optimal value of $\mathcal{GAP}''$,

$$\text{min. } L''(\mu) \qquad \text{s. t. } \quad \mu_j \ge 0, \ \forall j \in J.$$

In $L''(\mu)$, the objective function is additive over the agents and the constraints are separable over them; this maximization can be achieved by solving the following subproblems: for each agent k,

$$\begin{aligned} L''_k(\mu) = \text{max. } & \sum_{j \in J} (p_{kj} - \mu_j) x_{kj} \\ \text{s. t. } & \sum_{j \in J} w_{kj} x_{kj} \le c_k, \\ & x_{kj} \in \{0,1\}, \ \forall j \in J, \end{aligned}$$

and for the remaining terms,

$$L''_{const}(\mu) = \sum_{j \in J} \mu_j.$$

We can thus describe the Lagrangian dual problem as follows:

$$\begin{aligned} \text{min. } & \sum_{k \in A} L''_k(\mu) + L''_{const}(\mu) \\ \text{s. t. } & \mu_j \ge 0, \ \forall j \in J. \end{aligned}$$

Our distributed solution method solves this problem by using only local communications among agents.

Solution. The basic procedure of this protocol is as follows:

(Step 1) All agents initialize their Lagrange multiplier vectors as 0.

(Step 2) Under a current value of μ, each agent k solves his knapsack problem to compute $L''_k(\mu)$. Then the agents send these results to their respective neighbors.

(Step 3) If all assignment constraints of the original problem are satisfied, and on any good j,

$$\mu_j (1 - \sum_{k \in A} x_{kj}) = 0 \tag{5}$$

holds, then the agents can quit the procedure to provide an optimal solution.

(Step 4) Each agent k finds an upper bound and a lower bound. Agent k also finds the smallest upper bound, *BestUB*, and the largest lower bound, *BestLB*, among those found so far. If both *BestUB* and *BestLB* have the same value, the agents can quit the procedure to provide an optimal solution.

(Step 5) Each agent k updates the Lagrange multiplier vector from $\mu^{(t)}$ to $\mu^{(t+1)}$ by the subgradient method [1] while keeping $\mu_j \geq 0$ and goes back to Step 2.

In this procedure, note that the following differences are derived from relaxing inequalities instead of equalities.

First, in Step 3, the termination condition of this protocol becomes more difficult than that of the previous one. Second, in Step 4, to build a feasible solution, we can simply ignore the goods that have not been chosen by any agent. Third, in Step 5, we need to replace the updating rule for a Lagrange multiplier by the following rule:

$$\begin{aligned} g_j^{(t)} &\leftarrow 1 - \sum_{i \in A} x_{ij} \\ temp &\leftarrow \mu_j^{(t)} - \frac{\pi^{(t)}(BestUB^{(t)} - BestLB^{(t)})g_j^{(t)}}{\sum_{j \in J}(g_j^{(t)})^2}, \\ \mu_j^{(t+1)} &\leftarrow \max\{temp, 0\}. \end{aligned}$$

Clearly, by this rule, a Lagrange multiplier never gets lower than zero.

Obviously, both $\mathcal{GAP}'$ and $\mathcal{GAP}''$ provide the same optimal value for the over- constrained GMAP because $\mathcal{GAP}''$ turns into $\mathcal{GAP}'$ by introducing slack variables for the inequality constraints. It must be pointed out that the difference between $\mathcal{GAP}'$ and $\mathcal{GAP}''$ may be slight, but their Lagrangian duals, which our methods try to solve, differs significantly in that

- The Lagrangian dual of $\mathcal{GAP}'$ has the problem of the disposal agent while that of $\mathcal{GAP}''$ does not.
- The Lagrangian dual of $\mathcal{GAP}''$ has non-negativity constraints on variables while that of $\mathcal{GAP}'$ does not.

Furthermore, as mentioned before, the method for solving the Lagrangian dual of $\mathcal{GAP}''$ has a more complex termination condition.

4 Experiments

We experimentally compared the performance of the two methods on the variants of GAP benchmark instances from the OR-Library[12]. Clearly, the instances in our experiments should be over-constrained, but none of the GAP benchmark instances have that property; in other words, in each benchmark instance, the total capacities of the agents are large enough for the goods to be assigned.

Table 1. DisLRP with disposal agent on c1040-1

x	*Round*	*BestUB*	*BestLB*	*Quality*
0.1	1	0.0000	0	-
0.2	190	244.0000	244	1.0000
0.3	2115	456.0000	455	1.0000
0.4	1002	601.0000	601	1.0000
0.5	10000	705.7143	692	0.9806
0.6	819	828.0000	828	1.0000
0.7	10000	900.4581	887	0.9851
0.8	10000	934.9447	933	0.9979
0.9	10000	950.0000	948	0.9979

Table 2. DisLRP with inequality-based formulation on c1040-1

x	*Round*	*BestUB*	*BestLB*	*Quality*
0.1	1	0.0000	0	-
0.2	2	244.0000	244	1.0000
0.3	1189	455.0000	455	1.0000
0.4	1264	601.0000	601	1.0000
0.5	10000	705.7146	690	0.9777
0.6	491	828.0000	828	1.0000
0.7	10000	900.4357	887	0.9851
0.8	10000	935.0000	933	0.9979
0.9	10000	950.0000	931	0.9800

Thus, to make an instance over-constrained, we simply reduce the capacity of each agent by a constant factor. More specifically, we multiplied capacity c_k of each agent k by $x \in \{0.1, 0.2, \dots, 0.9\}$. Consequently, we generated 540 instances in total, most of which can be expected to be over-constrained. We call this x a *capacity coefficient.*

We conducted our experiments on a simulator written in JAVA and used lp_solve 5.5.2.0[9] for each agent to solve a local knapsack problem. Lp_solve is a freely available Linear/Integer programming solver with many easy-to-use application program interfaces (APIs).

Tables 1 and 2 show the results for the DisLRP with the disposal agent and the DisLRP with the inequality-based formulation, respectively, on a benchmark instance called c1040-1 that consists of 40 goods to be assigned to 10 agents. Tables 3 and 4 also show the results on a benchmark instance called c1060-1 that consists of 60 goods to be assigned to 10 agents. In these tables, x is the capacity coefficient, *Round* is the number of rounds spent until a procedure is terminated, *BestUB* is the lowest upper bound, and *BestLB* is the highest lower bound. Also, *Quality*, which means the ratio of *BestLB* on *BestUB*, denotes the quality lower bound of the obtained feasible solutions. Obviously, the closer this quality lower bound is to one, the better the performance. In our experiments we stopped a run at 10000 rounds if the procedure failed to find an optimal solution. In that case, *BestUB* and *BestLB* do not reach the same value.

To see whether the differences on *Round* and *Quality* are statistically significant, we applied the Wilcoxon signed-rank test to series of data obtained by our two methods. The results are summarized in Table 5. The null hypothesis on *Quality* is not rejected, which means that our two methods may not be different in terms of *Quality*. On the other hand, the null hypothesis on *Round* is rejected, which suggests that our two methods are different in terms of *Round*.

We further analyzed by dividing the instances into two groups: one was comprised of instances whose capacity coefficients ranged from 0.2 to 0.5 and the other was comprised of instances whose capacity coefficients ranged from 0.6

Table 3. DisLRP with disposal agent on c1060-1

x	*Round*	*BestUB*	*BestLB*	*Quality*
0.1	1	239.0000	239	1.0000
0.2	23	474.0000	474	1.0000
0.3	301	784.0000	784	1.0000
0.4	181	1010.0000	1010	1.0000
0.5	10000	1167.4033	1159	0.9928
0.6	10000	1316.7786	1295	0.9835
0.7	10000	1397.3609	1373	0.9826
0.8	10000	1426.0116	1406	0.9860
0.9	10000	1442.0958	1419	0.9840

Table 4. DisLRP with inequality-based formulation on 1060-1

x	*Round*	*BestUB*	*BestLB*	*Quality*
0.1	1	239.0000	239	1.0000
0.2	13	474.0000	474	1.0000
0.3	53	784.0000	784	1.0000
0.4	198	1010.0000	1010	1.0000
0.5	10000	1167.4036	1159	0.9928
0.6	10000	1316.7501	1310	0.9949
0.7	10000	1397.3601	1378	0.9861
0.8	10000	1426.0126	1406	0.9860
0.9	10000	1442.0978	1419	0.9840

Table 5. Wilcoxon signed-rank test for all instances

	Quality	*Round*
Hypothesis	No difference between two data series.	
Test statistic T	67704.5	15658.5
$\|Z\|$-value	0.3087	3.2182
Conclusion	Not rejected at significance level 5%	Rejected at significance level 1%
Median		
(disposal)	0.9998	285
(inequality)	0.9990	174

to 0.9[1]. For each of these two groups, we also applied the Wilcoxon signed-rank test again to see whether the differences on *Round* and *Quality* are statistically significant. The results are summarized in Tables 6 and 7. According to Table 6, both null hypotheses on *Round* and *Quality* are rejected, which means that our two methods have different performances. Looking at their medians, DisLRP with inequality-based formulation seems to find a better solution faster for more over-constrained instances. On the other hand, according to Table 7, only the null hypothesis on *Round* is rejected. Looking at its median, DisLRP with the disposal agent can find a solution faster for fewer over-constrained instances.

To summarize, we found that

1. DisLRP with an inequality-based formulation finds a better solution faster for more over-constrained instances,
2. DisLRP with the use of a disposal agent finds a solution faster for fewer over-constrained instances.

We are a bit surprised by the first finding in our experiments. For more over-constrained instances, we sometimes observed that the agents in the DisLRP with inequality-based formulation are likely to select the sets of goods that are

[1] We ignored the instances whose capacity coefficients are 0.1 since most have zero as their optimal values.

Table 6. Wilcoxon signed-rank test for instances whose capacity coefficients range from 0.2 to 0.5

	Quality	*Round*
Hypothesis	No difference between two data series.	
Test statistic T	250	3122
$\|Z\|$-value	3.17479	7.3127
Conclusion	Rejected at significance level 1%	Rejected at significance level 1%
Median		
(disposal)	0.9980	171.5
(inequality)	1.0000	64

Table 7. Wilcoxon signed-rank test for instances whose capacity coefficients range from 0.6 to 0.9

	Quality	*Round*
Hypothesis	No difference between two data series.	
Test statistic T	2937	980
$\|Z\|$-value	1.9490	2.6479
Conclusion	Not rejected at significance level 5%	Rejected at significance level 1%
Median		
(disposal)	0.9859	455
(inequality)	0.9831	1220

mutually exclusive in the very early rounds and, as a result, seem to have many chances to satisfy the termination condition (5). This suggests that such instances may be solved (nearly) optimally even with local knapsack optimization. On the other hand, in the DisLRP with the disposal agent, who has poor knowledge about the system, especially in the very early rounds, the states of the system may be disturbed in those early rounds. We expect this explains our first finding. For less over-constrained instances, both methods require much effort to coordinate the selection of goods among the agents. The DisLRP with inequality-based formulation requires more rounds to coordinate since its Lagrangian dual problem is more restrictive.

5 Conclusion

We presented two methods for over-constrained GMAP instances. The first is DisLRP with a disposal agent, which performs better for fewer over-constrained instances. The second is DisLRP with inequality-based formulation, which performs better for more over-constrained instances.

It is important to keep in mind that these two methods are designed for over-constrained GMAP instances, where the total capacities of agents are not sufficient for the goods. On the other hand, for an under-constrained instance with sufficient overall capacities, these methods may yield the optimal value that is different from the one obtained by the conventional DisLRP with equality-based formulation. This is obvious because, by relaxing assignment constraints for an under-constrained instance, a feasible region gets larger and as a result the optimal value may change. Currently, it seems reasonable to suggest that the proposed methods be used for over-constrained instances while the previous DisLRP with equality-based formulation be used for under-constrained instances. In our future work, we would like to develop an unified framework for handling both under- and over-constrained GMAP instances.

References

1. Bertsekas, D.P.: Nonlinear Programming, 2nd edn. Athena Scientific (1999)
2. Bhatti, S., Xu, J.: Survey of target tracking protocols using wireless sensor network. In: Proceedings of the 5th International Conference on Wireless and Mobile Communications, pp. 110–115 (2009)
3. Dias, M.B., Zlot, R., Kalra, N., Stentz, A.: Market-based multirobot coordination: a survey and analysis. Proceedings of the IEEE 94(7), 1257–1270 (2006)
4. Frank, C., Römer, K.: Distributed Facility Location Algorithms for Flexible Configuration of Wireless Sensor Networks. In: Aspnes, J., Scheideler, C., Arora, A., Madden, S. (eds.) DCOSS 2007. LNCS, vol. 4549, pp. 124–141. Springer, Heidelberg (2007)
5. Guignard, M., Kim, S.: Lagrangean decomposition: A model yielding stronger Lagrangean bounds. Mathematical Programming 39, 215–228 (1987)
6. Hirayama, K.: A new approach to distributed task assignment using Lagrangian decomposition and distributed constraint satisfaction. In: Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 2006), pp. 660–665 (2006)
7. Hirayama, K.: An α-approximation protocol for the generalized mutual assignment problem. In: Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI 2007), pp. 744–749 (2007)
8. Hirayama, K., Matsui, T., Yokoo, M.: Adaptive price update in distributed Lagrangian relaxation protocol. In: Proceedings of the 8th International Joint Conference on Autonomous Agents Multi-Agent Systems (AAMAS 2009), pp. 1033–1040 (2009)
9. lpsolve, http://sourceforge.net/projects/lpsolve/
10. Modi, P.J., Shen, W.-M., Tambe, M., Yokoo, M.: An asynchronous complete method for distributed constraint optimization. In: Proceedings of the Second International Joint Conference on Autonomous Agents Multi-Agent Systems (AAMAS 2003), pp. 161–168 (2003)
11. Nauss, R.M.: The generalized assignment problem. In: Karlof, J.K. (ed.) Integer Programming: Theory and Practice, pp. 39–55. CRC Press (2006)
12. OR-Library, http://people.brunel.ac.uk/~mastjjb/jeb/info.html
13. Smith, R.G.: The contract net protocol: high-level communication and control in a distributed problem solver. IEEE Transactions on Computers 29(2), 1104–1113 (1990)
14. Yagiura, M., Ibaraki, T.: Generalized assignment problem. In: Gonzalez, T.F. (ed.) Handbook of Approximation Algorithms and Metaheuristics. Computer Information Science Series. Chapman Hall/CRC (2006)

The Effect of Congestion Frequency and Saturation on Coordinated Traffic Routing

Melanie Smith and Roger Mailler

Computational Neuroscience & Adaptive Systems Lab
Department of Computer Science
University of Tulsa, Oklahoma
{melanie,mailler}@utulsa.edu
http://www.cnas.utulsa.edu

Abstract. Traffic congestion is a widespread epidemic that continually wreaks havoc in urban areas. Traffic jams, car wrecks, construction delays, and other causes of congestion, can turn even the biggest highways into a parking lot. Several congestion mitigation strategies are being studied, many focusing on micro-simulation of traffic to determine how modifying road structures will affect the flow of traffic and the networking perspective of vehicle-to-vehicle communication. Vehicle routing on a network of roads and intersections can be modeled as a distributed constraint optimization problem and solved using a range of centralized to decentralized techniques. In this paper, we present a constraint optimization model of a traffic routing problem. We produce congestion data using a sinusoidal wave pattern and vary its amplitude (saturation) and frequency (vehicle waves through a given intersection). Through empirical evaluation, we show how a centralized and decentralized solution each react to unknown congestion information that occurs after the initial route planning period.

Keywords: routing, coordination, constraint optimization.

1 Introduction

Traffic congestion is a familiar and frustrating occurrence for anyone who drives on the road. In fact, congestion is continually getting worse, affecting more of the road system (secondary, tertiary roads), during more times of the day, and with rising penalties for extra travel time [5]. Except for consistent and repetitive physical bottlenecks (causing 40% of congestion), congestion occurs with maddening irregularity where nothing is the same from one day to the next. Studies show that the unpredictability of traffic incidents or inclement weather cause the majority of congestion because no one can accurately determine how different factors such as time-of-day, weather conditions, number of vehicles on the road, etc, will affect traffic [5]. A domino effect also occurs due to road congestion [5].

Because congestion causes are all related, significant payoffs can be expected by treating them. By decreasing the effects of congestion, the total delay experienced by travellers is reduced, time and fuel are saved, vehicles cause less emissions, safety for travelers is increased, the benefits for just-in-time manufacturing are increased, overall transportation costs are decreased, and benefits are provided to the national economy [5].

An obvious approach to mitigating congestion is changing road structures and governing rules to increase traffic flow and/or lighten the demand for a particular road. The problems are the physical, financial, and political limits that cap the effectiveness of these kinds of solutions. Instead, we ask the question: how can we better utilize what is already in place, making our roadways more efficient? This is the impending problem as our society continues to grow and physical expansions of roads become less and less feasible [3].

Some Global Positioning System (GPS) devices have the ability to warn drivers of upcoming congestion and automatically suggest alternative routes. However, GPS units typically use similar path planning algorithms to reroute users, meaning that drivers will often be advised to take the same detour. Thus, detouring does not always help the individual driver as others receive the same advice, moving the jam to secondary/tertiary roads where drivers will again be faced with the same congestion, but on a smaller road.

A separate, but substantial, issue related to route coordination is not only how the vehicles coordinate with one another, but also how they mitigate congestion due to vehicles on the road who do not use any routing or coordinating to get from one place to another. Essentially, this type of congestion travels in waves across the road network. The "green phase" of a traffic light is analogous to the frequency of the wave and the amount of traffic flowing determines the amplitude of the wave (or saturation level).

In most cases, approaches to traffic management only deal with how to distribute and use global information in order to make local (per vehicle) routing decisions. To our knowledge, very few address issues on how to best use all the information that is shared in the system and how to make routing decisions to satisfy both local and global objectives (e.g. individual driver route preferences and minimizing overall network congestion) while also reacting to external congestion.

In this paper, we discuss the traffic domain as a constraint optimization problem and use our coordination algorithm for vehicle routing decisions. We then generate congestion in the network over varying frequency (waves per hour) and saturation levels to compare a centralized and decentralized technique.

2 Background

Traffic modelling takes place on three distinct levels: microscopic, mesoscopic, and macroscopic [1]. Microscopic models consider each vehicle individually and model each according to its own position, headway gap, and velocity (e.g. [11]). Essentially, vehicle trajectories are produced as the vehicles move through the system using rules that dictate how they move and interact, including acceleration, deceleration, lane changing, and passing maneuvers [1]. Macroscopic models aggregate the data depicted by the micro models and use average velocity and density numbers to determine the flow over a particular roadway. Mesoscopic models are considered to have an intermediate level of detail, modelling individual vehicles, but describing their interactions and actions from a macroscopic perspective.

Balbo [2] also uses an agent-based approach to design a Transportation Regulation Support System called SATIR. SATIR integrates a decision support system with an automatic vehicle monitoring system. This is similar to our work in that it combines different architectural units in the simulation environment. However, SATIR focuses on public transit bus system management and routing them in real-time in order to minimize the delay of the bus schedule.

Another similar work investigates the effect of route information sharing over lattice, radial-ring, and a real road network [13]. Their results show that as the number of vehicles that share route information increases, that congestion within the system decreases. However, they only consider a highly centralized approach and vary the percentage of vehicles that participate in the information sharing.

Wunderlich, et al [12] investigate link travel time for decentralized route guidance and find that predictive route guidance can be supported within the constraints of a decentralized architecture. Both the SAVaNT and SAVaNT-CNV approaches are studied, but no comparison is shown between different degrees of centralization as they focus more on the percentage of guided versus unguided vehicles, or how congestion changes as more vehicles adopt their guidance technology. The DIVERT simulation system deals with large-scale environments and vehicle-to-vehicle communication [7]. DIVERT focuses on learning the road network and on the ability and feasibility of networked communication between vehicles.

Centralized systems are generally infrastructure-based and require roadside equipment to upload information into the centralized server [10]. Because all information is aggregated in one place, there is a complete global view that allows the entire system to be evaluated, and given infinite time and computational power, could garner an optimal routing solution. However, optimality is prohibitively expensive. Otto found that having a global perspective allows a greater time savings in travel times than the distributed, local perspective, but that savings comes at an additional cost in route length to drivers [10].

Distributed systems totally rely on V2V communications instead of V2I (vehicle-to-infrastructure). They take advantage of direct exchanges between participating vehicles to achieve higher scalability, but at the risk of losing data consistency. By using V2V communication, deployment and maintenance costs are less. However we are then faced with questions of user adoption and how much effect imprecise information will have on the vehicles. Taken from a local perspective, Otto shows an improvement in reliability, scalability, and cost without negatively impacting travel time and distance relative to a centralized version [10].

Additionally, Otto compares centralized and distributed methods using several routing algorithms and several different neighborhood sizes for the distributed environment: information is gathered from either a two block radius around the vehicle's current location, from the full route within two intersections, or from a quarter of the route within four intersections. The study finds that looking at the congestion information for the entire route was pointless because congestion levels may change before the vehicle reaches that point in the route, which wastes significant bandwidth communication what becomes useless information. Even though the centralized algorithm assigns routes to vehicles upon request, it does not use the route request data to help predict how congestion will be in the future, it only factors in the current congestion levels being experienced.

3 Modeling Traffic as a Constraint Optimization Problem

To model the traffic domain within a constraint optimization problem, we define the road map to be a directed graph with a set of nodes representing intersections and a set of edges that represent road segments, or links. A route is a path, or a sequence of connected intersections, that is traversed in order and does not contain cycles.

Let $M(P, Q)$ be a directed graph representing a road map with a set of nodes, P, representing intersections and a set of edges, Q, representing road segments, or links. Each $q \in Q$ connects two intersections, $p_i \in P$ and $p_j \in P$, and is denoted as $(p_i, p_j) \in Q$. A route is a path, or a sequence of connected intersections, that is traversed in order and does not contain cycles. For any route, $r =< p_1, p_2, \ldots, p_n >, \forall i \mid 1 \leq i < n, p_i \in P \bigwedge p_{i+1} \in P \bigwedge (p_i, p_{i+1}) \in Q$.

The weight of the elements in Q is the degree of saturation ($saturation = flow/maxflow$) on that element. This is defined as $w(x) = x.saturation \mid x \in Q$. For $p \in P$, the weight is the queue length, or the number of vehicles waiting to pass through the intersection.

When cast as a constraint optimization problem, following from Yokoo's original definition [14], each vehicle acts as an agent with start, destination, and departure time variables. There exists a set of possible routes for each vehicle, and each are evaluated using a cost function chosen by that vehicle. We define the traffic routing component as:

- a set of n vehicles $V = \{v_1, \ldots, v_n\}$.
- a set of start places $S = \{s_1, \ldots, s_n\}$, where each s_i is the start place for v_i.
- a set of goal places $G = \{g_1, \ldots, g_n\}$, where each g_i is the goal place for v_i.
- a set of start times $T^0 = \{t^0_1, \ldots, t^0_n\}$, where each t^0_i is the start time for v_i.
- a finite, discrete set of possible routes for each vehicle $R = \{R_1, \ldots, R_n\}$ where each R_i contains the set of routes to which the associated vehicle, v_i, may be assigned. Each $r \in R_i$ is a path from s_i to g_i, and are ranked according to the anticipated cost that the route will entail.
- a set of cost functions $f = \{f_1, \ldots, f_n\}$ where each $f_i(r) \mid r \in R_i \bigwedge r = v_i.currentRoute$ is function

$$f_i(r) : \sum_{j=1}^{r.length-1} travelTime(r_j, r_{j+1}) \tag{1}$$

For this work, we use the same cost function for all vehicles that minimizes total travel time. Each vehicle is then assigned to a coordinator agent, who is responsible for coordinating routes among the vehicles it controls. Thus, there is a set of k coordinator agents, $C = \{c_1, \ldots, c_k\}$ and a mapping function $\alpha : C \rightarrow V$, implying that $\alpha(c_i) \rightarrow \bar{V}$ ($\bar{V} \subset V$) states that it is coordinator c_i's responsibility to assign routes to vehicles in the set $\bar{V}$. In the centralized approach, all vehicles are assigned to one coordinator, so $C = \{c_1\}$ and $\bar{V} \equiv V$. Distributed approaches have one coordinator per vehicle, so in that case, $k = n$.

The problem is to find an assignment $R^* = \{r_1, \ldots, r_n \mid r_i \in R_i\}$ such that the global cost, called F, is minimized. We define F as follows:

$$F(R^*) = \sum_{i=1}^{n} f_i(r_i) \tag{2}$$

4 Coordinated Routing

The A* algorithm is a best-first graph search algorithm that is used for finding the optimal (least-cost) path from a start point to an end point in a directed, acyclic graph [8]. A* uses a combination of the predicted distance to the goal and the known cost to the current position instead of just a cost heuristic: $f(x) = g(x) + h(x)$, where $g(x)$ is the known cost from the start point to the current position and $h(x)$ is the estimated distance to the end point.

We set $h(x)$ as the euclidian distance from the current point to the destination divided by a constant speed limit that ensured admissibility to give us an estimated cost in time units. The straightline distance proves to be much less than the actual distance travelled between two points because typically taking roads means traveling in a grid-like manner. The generic equation for $g(q)$ is as follows.

$$g(q) = g(p) + runningTime(p, q) + queueingTime(q) \tag{3}$$

Given that q is a successor to p, meaning there exists a road segment that flows out of p and into q, the actual cost of traveling from p to q is the time it takes for a vehicle to drive from p to q. The time spent waiting at an intersection is also added in.

By combining congestion information with the routing specifics of each vehicle, we can coordinate routes between vehicles, giving us a more efficient use of the existing road network. Routing is by itself an expensive operation; coordinating efforts between several entities that are searching for a route together only magnifies the cost.

The other modeling construct facilitates data dissemination. Studies have looked at centralized versus distributed data dissemination [10], but have not paired that completely with vehicular control. Others have also looked at how quickly information can be accumulated as vehicles drive through a network [6]. We incorporate congestion data using the concept of neighborhoods, where information can be shared among neighbors. Two vehicles can be considered neighbors if they are assigned to the same coordinator (siblings) or if they are physically located on the same intersection or link (local), and in some cases we consider all vehicles to be neighbors (global). Each vehicle can share its own route plan and destination, but does not pass along other information it may have learned previously from other vehicles. Each coordinator also has congestion information knowledge for the any reporting entities on the network, for example, this information would include saturation information that summarizes vehicles that are not being simulated.

4.1 Centralized

The centralized algorithm takes an initial snapshot of the environment's congestion and then begins the routing process. Then it uses an A* search that takes into account the known congestion and the anticipated congestion of all the previously routed vehicles. In order to do this, we must make the assumption that order is preserved, meaning vehicles plan their routes in the same order they are moved forward during the simulation. For example, if we know v_1 is the first vehicle, we can accurately predict the amount of congestion it will encounter. Order preservation is restrictive, however to solve without that restriction, there is need to run a solver before the simulation can begin and again every time rerouting is desired. It is conceivable to prioritize vehicles in order of submitting their information to the coordinator, which is what we have simulated. To consider different orderings and find an optimal solution, the A* search grows to be on the order of $(3^n)^h)$ where 3 is the typical number of outgoing links, n is the number of vehicles, and h is the maximum number of route hops over all vehicles. For a set of 25 vehicles with an average of 8 hops, it is about $O(10^95)$ each time routes are calculated. In algorithms where rerouting is allowed, the ordering is different each time routes are recalculated, so over time, order preservation becomes less restrictive.

Algorithm 1. Centralized Coordination Algorithm

```
$cong \leftarrow$ initial congestion knowledge;
**foreach** $vehicle\ v_i \in \bar{V} | \alpha(c) \rightarrow \bar{V}$ **do**
    $routes \leftarrow$ route($v_i$, $cong$);
    update_congestion_info($routes$, $cong$);
**end**
```

4.2 Decentralized

While the Centralized Algorithm plans routes for all vehicles before the simulation begins, the Decentralized Algorithm updates routes as the vehicles are traveling in much the same fashion as the Distributed Stochastic Algorithm (DSA) [15]. As previously mentioned, the decentralized approach assigns a separate coordinator to each vehicle in the simulation. This means that instead of having information on all vehicles in the simulation, this algorithm only considers information from a vehicle's neighbors. A neighbor is considered any vehicle that is co-located at the same intersection or link. At each time step where the vehicle is at an intersection, this algorithm (see Algorithm 2) determines who its neighbors are, which we define as any other vehicle that is stopped at the same intersection at the same time. At each time click, the neighborhood can contain a different set of vehicles and thus a different set of knowledge.

The *update_congestion_info* method is the same as with the Centralized algorithm except that instead of updating it from the beginning of all the vehicle's routes, it picks up at the current position of the vehicles in order to keep from calculating congestion information that is in the past. It also only updates known congestion info that has been collected from its neighbors. If stale information is passed, it is ignored.

Algorithm 2. Decentralized Coordination Algorithm

```
$neighbors$ ← current congestion info from neighbors;
update_congestion_info($neighbors$, $cong$);
foreach $vehicle\ v_i \in \bar{V} | \alpha(c) \to \bar{V}$ do
    $neighbors$ ← route($v_i$, $cong$);
    update_congestion_info($neighbors$, $cong$);
end
foreach $cycle$ do
    foreach $vehicle\ v_i \in \bar{V} | \alpha(c) \to \bar{V}$ do
        if $random < p$ then
            $neighbors$ ← route($v_i$, $cong$);
            update_congestion_info($neighbors$, $cong$);
        end
    end
end
```

5 Simulator and Experimentation Setup

Our vehicles operate in a simulated queue based traffic model (see Figure 1) that runs inside the FARM simulator [9], meaning that each road segment, or link, is split into two parts, one where traffic is flowing normally and the other where traffic is queueing at the intersection. Each link has a maximum number of vehicles that are allowed to flow over the link simultaneously, and as the number of vehicles approaches the maximum, congestion is experienced. The effective length of a link is shortened by the length of the queue, meaning that if the queue contains the maximum number of vehicles, there is no movement. Each intersection has one queue per incoming link. Each queue is given a "green light" that allows a certain number of vehicles to pass through the intersection. Within each single queue, vehicles leave in the same order they arrived, however across all queues flowing into one intersection, that may or may not be the case.

To determine the velocity of the vehicles traveling over the link, we use a Weibull distribution (see Equation 5) for any saturation level that is over the free flow density (k_{free} = 60% saturation) and less than the jam density (k_{jam} = 95%

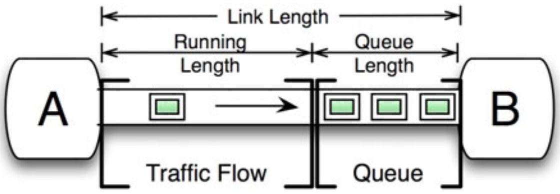


Fig. 1. Queue Based Traffic Model

saturation). $Maxflow$ is the capacity on the link, which was 10 vehicles for these simulations The Weibull distribution choice allows us to take advantage of a multistage model, which is very similar to the DynaMIT model [4].

$$\begin{aligned} 1 + (avgvel(k_{free} * maxflow) - 1) * \\ 1 - ((saturation - k_{free})/(k_{jam} - k_{free}))^2)^2 \end{aligned} \quad (4)$$

This equation is also contingent upon the average velocity that a vehicle travels while on a link. The distance a vehicle travels during each cycle of the simulator is calculated based on the flow experienced on that link at that time. The average velocity is calculated as the current running length (total length (len) minus the queue length (ql)) of the link divided by the time taken to accelerate (ta), decelerate (td), and cruise (tc) from the time the vehicle began its journey (ti).

$$avgvel = \frac{len - ql}{ta + td + tc - ti} \quad (5)$$

Congestion data is generated in a wave pattern that is routed between the start and destination regions. The wave travels from one intersection to the next, and if the congestion arrives from several incoming links, it is combined on the outgoing link. The phases are related, and an example can be found in Figure 2. The thickness of the edges indicates the saturation levels and the thickness is proportional to the amount of congestion on that link. The sinusoidal wave propagation formula is as follows:

$$saturation * (0.5 + 0.5 * sin(phase - \frac{\pi * time * wph}{1800})) \quad (6)$$

When a new vehicle is routed, it can calculate what its average velocity and wait times will be as it progresses through its route, based on the congestion data and the knowledge of the routes other vehicles are planning to take.

By including the knowledge of other vehicles and any other existing congestion on the system in the evaluation function, each subsequent vehicle to be rerouted can make the best decision based on that information. The method $update\_congestion\_info(routes, cong)$ creates a table containing traffic flow information for each link and queue length for each intersection, keeping counts for each entity over time. When a new vehicle is routed, it can calculate what its average velocity and wait times will be as it progresses through its route. Generating these information tables is a similar process to building the optimal search tree: all vehicles are initially scheduled to move at their start time, and as the table is filled out for each time click, the counts are updated when vehicles are expected to leave or arrive at a new intersection.

Rerouting occurs when the encountered congestion is greater than the expected congestion level that was identified during the route planning phase. The expected congestion is not applicable for the cases where no congestion data is used, and is therefore set to 0. The rerouting is triggered by encountering congestion along the route, but congestion data is not known for any other intersection besides the current location of the vehicle in this case, simulating the visual perception of congestion that a driver can see ahead of him.

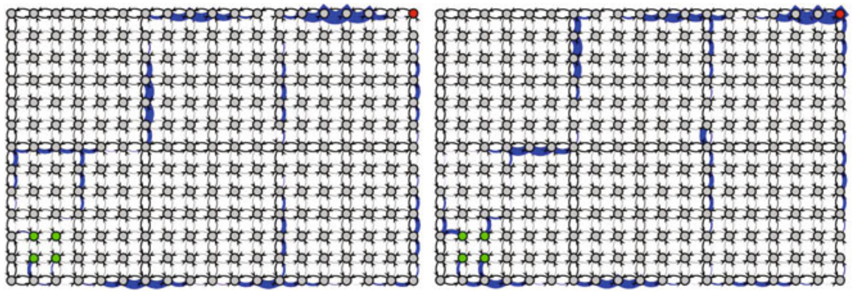


Fig. 2. Wave Propagation as Time Changes from $t = t_i$ (left) to $t = t_{i+k}$ (right)

The simulations presented in this paper are based on a 25 to 10 density ratio of vehicles to the maximum number of vehicles that can be on a link at a given time, and is evaluated over a small-area grid map (∼6 square miles). Vehicles start from a four-intersection cluster and travel to the same destination intersection. This would be analogous of an evacuation scenario where everyone in a particular area gets sent to the same shelter location. We use the map shown in Figure 2 that has arterial roads each mile with speed limit of 45mph, secondary streets each half mile with speed limits of 35mph, and tertiary roads with speed limits of 25mph.

All vehicles attempt to start at the same time, however this is staggered by the saturation and frequency of the external congestion and the queueing of the vehicle at the initial intersection. Thus, actual start times are dependent on whether another vehicle can fit on the road given the external congestion and whether the intersection will let the vehicle through at a given time step. Time steps of the simulation roughly represent an actual second. However, it is also the case that when rerouting occurs, it occurs within one time step and generally causes the time step to take much longer than an actual second. Simulation continues until all vehicles have arrived at their destination.

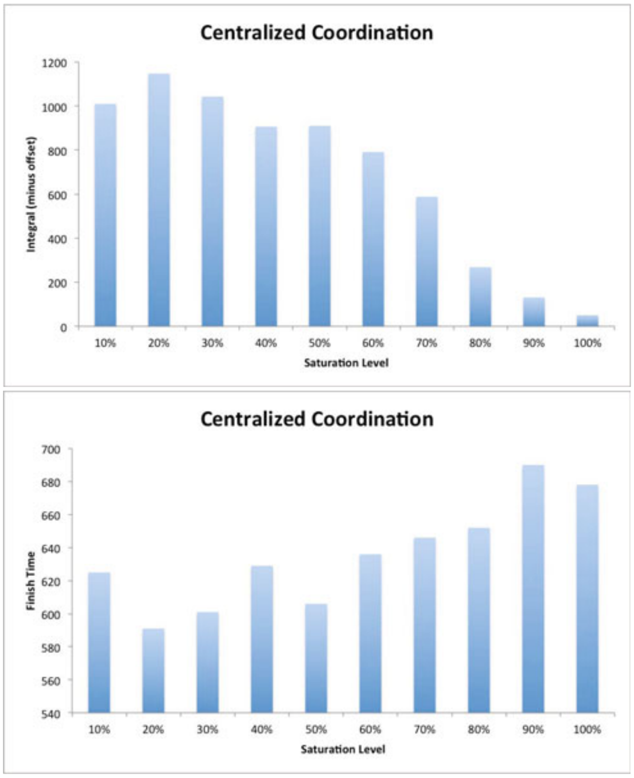


Fig. 3. Averaged Centralized results for Saturation Variance. Top: Integral of the Number of Vehicles that have reached their destination over time. Bottom: Finish time for the last vehicle.

6 Results

We show results for varying the saturation level from 10% to 100% in 10% increments at 20 waves per hour and varying the waves per hour from 5 to 60 in increments of 5 at 60% saturation for a centralized approach and a decentralized approach.

To measure the effects that varying the saturation and wave frequency, we keep track of the number of vehicles that have arrived at their destination at each time click. As the finished vehicle count approaches the number of vehicles in our simulation, we can compare two different aspects of time spent traveling. First, there is the obvious comparison of which approach got ALL the vehicles to their destination in the least amount of time. This is equal to the finish time for the last vehicle to arrive at its destination.

However, that metric does not consider the rate at which the vehicles finish, or whether more vehicles finish faster regardless of when the last vehicle arrives. For instance, which is superior, a route scheme that takes everyone 10 minutes to reach their destination or a route scheme where it takes some vehicles 5 minutes, another set 7 minutes, and the last vehicle takes 11 minutes? This requires a different metric, and is the one we emphasize as showing a more realistic view of how well the approach solved the problem. To calculate this, we take the integral under the curve of vehicles finished over time, so the higher the number, the more vehicles finish faster. In the graphs below, we do offset the numbers for ease of graphing because the difference between them is the important part, not the actual value.

Figure 3 shows the centralized results for the variations on saturation level. On the top is the last vehicle's finish time, which increases (predictably) as saturation increases in a somewhat linear fashion. However, the story left untold by that graph is that up until 70% saturation, the algorithm doesn't notice a dramatic drop in effectiveness. The integral (bottom) results show that there's an increase in the rate of decline as saturation rises towards the fully saturated end. This means that the vehicles are ALL taking longer as a whole to complete their route and having some trouble dealing with congestion. For example, compare

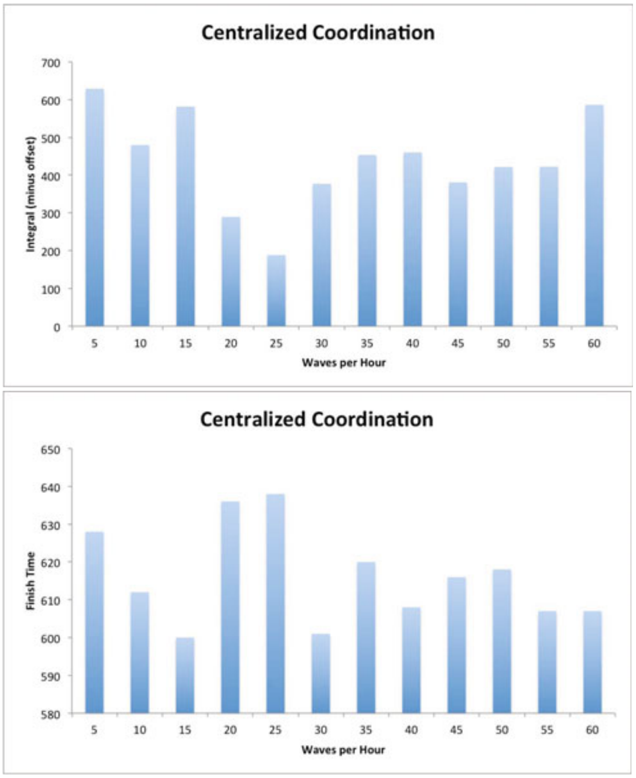


Fig. 4. Averaged Centralized results for Frequency Variance. Top: Integral of the Number of Vehicles that have reached their destination over time. Bottom: Finish time for the last vehicle.

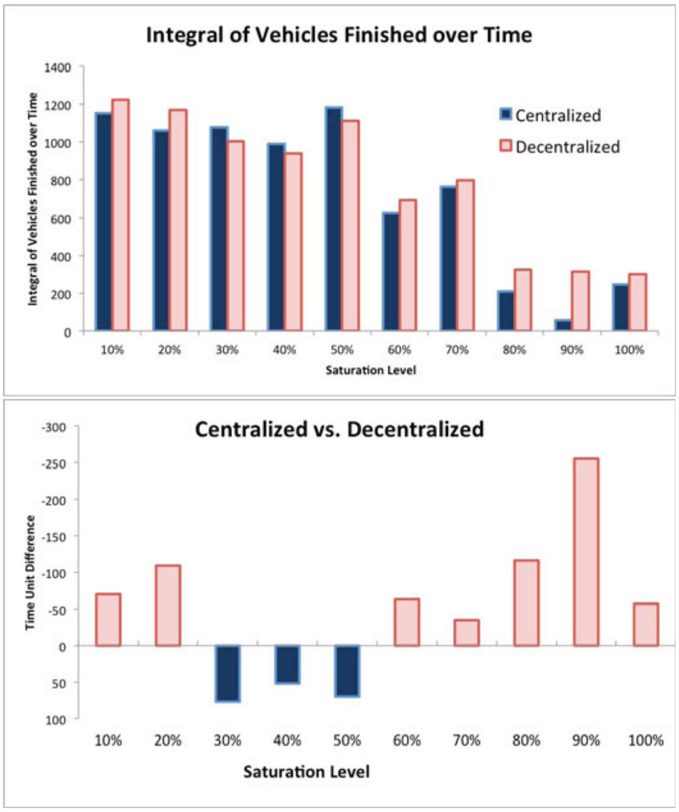


Fig. 5. Comparison of Centralized and Decentralized Solutions: Integral of Vehicles Finished over time for varying the Saturation Level (top) and the difference between centralized and decentralized integrals (bottom). The bars indicate the difference between each solution. The x-axis is the "break-even" point, and for instances where the centralized solution is better, the bar goes downward and where the decentralized solution is better, the bar goes upward.

the Integral results of 60%, 70%, and 80%. The finish time of the last vehicle does not change significantly, but the integral numbers show that a significantly larger number of vehicles finish sooner in the 60% scenario versus 70% and in the 70% versus 80% scenario. This result shows us that even a tiny drop in congestion can make a substantial impact on many drivers.

Figure 4 shows centralized results for the variations on frequency, or waves per hour. This is like saying a wave of congestion comes through the network every 1 minute when there are 60 waves per hour. The top shows the integral results and the bottom contains the last vehicle's finish time. The graphs visually break into three sections: 5-15 wph, 20-25wph, and 30-60wph.

Likely the first batch (5-15wph) are not going to be seen very often in real life, as it is rare to see a 12 minute long green light. The next batch deals with the 2.4-3 minute wave frequency, which for large intersections is common. Our results show that having congestion waves at this frequency has a largely negative impact on getting vehicles to their destination, emphasized by the fact that the integral values are lower than the rest and the finish times are much higher. The last batch (1-2 minute frequency range) shows relatively similar results except for the 60wph case. In this case, the graphs show a high integral value (the highest disregarding the first batch) and one of the lowest overall finishing times. This shows a nice balance of the two, and a marked improvement over the 55wph case that boasts the same low finish time but a significant drop with the integral.

Another reason for seeing improvement with increased frequencies is that the centralized algorithm's initial congestion snapshot used for initial route formulation and because of the higher frequency, it is never too far off from reality. However, with lower frequencies the consequences of inaccurate or outdated congestion information rise. Additionally, the higher the frequency, the more often there is a lower-congested gap for our coordinated vehicles to enter into.

Figure 5 compares the saturation variance between the centralized and decentralized approaches for one run (the previous graph in Figure 3 were averaged over all runs). The top graph shows the raw data and the bottom shows the difference between the two. What we are highlighting here is the shift from how the decentralized approach is better for the extreme ends and the centralized approach works better for the 30%-50% saturation levels. Although the differences are relatively small, the trend appears in the bottom graph where a 0 value indicates both approaches are equivalent, centralized outperforms decentralized when the bars are below the axis, and decentralized does better when the bars are above the axis. We graphed these numbers like this to show how as saturation increases in the 70% - 100% range, the benefits of using a decentralized approach become more pronounced.

7 Conclusion

In this paper, we discuss how coordinating vehicle routes improves the ability for vehicles to maneuver through the system. We discover that the amount of dynamic congestion information impacts how quickly a set of vehicles traverses through the system due to their ability to communicate and dynamically route around congestion.

In any case where a road network is close to fully saturated with vehicles, there does not exist a good solution when you consider congestion that is outside the route sharing and coordination environment. No matter how good the algorithm is, it is limited by the participation of the vehicles in the system (i.e. how much congestion is due to vehicles that are not being coordinated and sharing route information) and by the physical capacity of the road. However, by using a decentralized approach, we find that vehicles are more equipped to react to encountered congestion. We also find that small changes in congestion saturation can positively impact the system as a whole, and that having 60 waves per hour at intersections allow vehicles to move more effectively throughout the road network.

Adopting a technology like coordinated routing that involves a human taking direction from a computer device in order to help solve a problem also creates another problem. By coordinating a driver's route with the known routes of other drivers so that everyone gets where they are going at a better time, we also run the risk of the system being frustrating to use. For instance, if a route keeps changing every 3 minutes, the driver will get frustrated and turn it off or disregard its instructions. Thus, as this work continues, an important consideration is how well the technology will be received by its users. This means that no significant amount of time should pass causing the user to wait for a route – the user should be able to get in the car, find a route quickly, and get on their way. Second, there should be some stability in the route over time so as not to confuse the user or cause unnecessary lane changes and maneuvers in order to quickly get in the correct turn lane. Lastly, the system should balance the emotional effects that encountering congestion has on drivers with the ability to mitigate that congestion. If it is more frustrating to put up with the technology than it is to deal with the congestion, there is no purpose to deploying this type of technology. However, if a system can guarantee users to reliably get them to their destination in the least possible time, then this has the possibility of opening up a new market for GPS manufacturers to those people who may not need directions, but would benefit from the intelligence built into the system.

References

1. Administration, F.H.: Traffic analysis toolbox volume iii: Guidelines for applying traffic microsimulation modeling software. Tech. Rep. FHWA-HRT-04-040, USDoT (2004)
2. Balbo, F., Pinson, S.: An agent oriented approach to transportation regulation support systems. In: Proceedings of the 5th Workshop in Agent in Traffic and Transport (2008), http://www.lamsade.dauphine.fr/FILES/publi931.pdf
3. Bazzan, A.L.C.: Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. AAMAS 18(3), 342–375 (2009), http://www.inf.ufrgs.br/~bazzan/%23pub
4. Ben-Akiva, M.: Development of a deployable real-time dynamic traffic assignment system, task d interim report: analytical developments for dta system. Tech. rep., MIT ITS Program, Cambridge, MA (1996)
5. Cambridge Systematics Inc.: Traffic congestion and reliability: Trends and advanced strategies for congestion mitigation pp. 1–140 (September 2005)
6. Conceição, H., Damas, L., Ferreira, M., Barros, J.: The divert project: Development of inter-vehicular reliable telematics. OSGeo Journal 3, 51–56 (2007), http://www.dcc.fc.up.pt/~michel/homepage/publications/2007-OSGeo-2.html
7. Conceição, H., Ferreira, M., Barros, J.: On the Urban Connectivity of Vehicular Sensor Networks. In: Nikoletseas, S.E., Chlebus, B.S., Johnson, D.B., Krishnamachari, B. (eds.) DCOSS 2008. LNCS, vol. 5067, pp. 112–125. Springer, Heidelberg (2008), http://www.dcc.fc.up.pt/~michel/homepage/publications/2008-DCOSS.html
8. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions of Systems Science and Cybernetics 4(2), 100–108 (1968)

9. Horling, B., Mailler, R., Lesser, V.: Farm: A Scalable Environment for Multi-Agent Development and Evaluation. In: Lucena, C., Garcia, A., Romanovsky, A., Castro, J., Alencar, P.S.C. (eds.) SELMAS 2003. LNCS, vol. 2940, pp. 225–242. Springer, Heidelberg (2004)
10. Otto, J.S., Bustamante, F.E.: Distributed or centralized traffic advisory systems - the application's take. In: SECON, pp. 1–10 (September 2009)
11. Wahle, J., Schreckenberg, M.: A multi-agent system for online simulations based on real-world traffic data. In: 34th Hawaii Int'l Conf. on System Sciences (2001), http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=926332
12. Wunderlich, K.E., Kaufman, D.E., Smith, R.L.: Link travel time prediction for decentralized route guidance architectures. IEEE Transactions on Intelligent Transportation Systems, 4–14 (March 2000)
13. Yamashita, T., Kurumatani, K.: New approach to smooth traffic flow with route information sharing. In: Bazzan, A., Klugl, F. (eds.) Multi-Agent Systems for Traffic and Transportation Engineering, pp. 291–306. IGI Global (2009)
14. Yokoo, M., Durfee, E.H.: Distributed constraint optimization as a formal model of partially adversarial cooperation. Tech. Rep. CSE-TR-101-91, University of Michigan, Ann Arbor, MI 48109 (1991)
15. Zhang, W., Wang, G., Wittenburg, L.: Distributed stochastic search for constraint satisfaction and optimization: Parallelism, phase transitions and performance. In: Proceedings of the AAAI Workshop on Probabilistic Approaches in Search (2002)

Coordination, Conventions and the Self-organisation of Sustainable Institutions

Jeremy Pitt[1], Julia Schaumeier[1], and Alexander Artikis[1,2]

[1] Department of Electrical & Electronic Engineering, Imperial College London, SW7 2BT, UK
[2] National Centre for Scientific Research "Demokritos" Athens 15310, Greece

Abstract. Applications where autonomous and heterogeneous agents form opportunistic alliances, which require them to share collective resources to achieve individual objectives, are increasingly common. We model such applications in terms of self-governing institutions for shared resource management. Socio-economic principles for enduring institutions are formalised in a logical framework for dynamic specification of norm-governed systems. The framework is implemented in an experimental testbed to investigate the interplay of coordination in a social dilemma with mutable conventions of an institution. Experimental results show that the presence of conventions enables the norm-governed system to approximate the performance of a theoretically ideal system. We conclude that this approach to self-organisation can provide the foundations for implementing sustainable electronic institutions.

1 Introduction

Applications in which autonomous and heterogeneous agents form opportunistic alliances, which require them to share collective resources in order to achieve individual objectives, are increasingly common. Examples include vehicular networks [14], service-oriented systems such as cloud computing [1], and demand-side infrastructure management for water [6], energy [16], and so on. These examples are all open, distributed and resource-constrained. However, we are unable to 'privatise' the system, otherwise it would no longer be open, nor to 'centralise' the system, otherwise it would no longer be distributed.

Instead, we address the issue of resource constraint from the perspective of self-governing institutions for common pool resource (CPR) management [12]. By definition, an institution embodies the rules which specify the conditions concerning the provision and appropriation of resources. These rules should be mutable by other rules, and so can be adapted to suit the environment in which the system is embedded. This might itself be changed by exogenous events.

The institution then has to satisfy three performance criteria. Firstly, the coordination mechanisms and conventions should encourage compliance pervasion, defined as behaviour in accordance with the rules or norms, amongst members of the institution. Secondly, the selection, modification and adaptation of the rules should not only suit the environment but also result in a 'fair' outcome. Thirdly, even a fair distribution has to be sustainable in the long term; in other words, the rules also have to ensure that the institution itself is somehow enduring.

In the investigation of these criteria, this paper is organised as follows. Section 2 reviews the background to this work. Using a methodology for engineering socio-technical systems [9], Sections 3 and 4 develop a formal characterisation of self-governing institutions as dynamic norm-governed systems. In Section 5 describes experiments to evaluate the interplay of coordination in an iterated n-player game with mutable conventions of an institution. Results show that using conventions enables the institution to approximate the performance of a theoretically ideal system. Related and further work is discussed in Section 6, and we conclude in Section 7 that this approach to self-organisation provides the foundations for implementing electronic institutions whose properties of compliance pervasion, fairness and endurance support sustainability for CPR management.

2 Background

This section reviews the background to the current work, including the work on CPR management of Ostrom [12], institutionalised power [10], the dynamic specification of norm-governed systems [2], and the linear public good game [7].

2.1 Self-governing Institutions

Ostrom [12] observed that common pool resource (CPR) management problems have often been resolved in human societies through the 'evolution' of institutions. Ostrom defined an institution as a "set of working rules that are used to determine who is eligible to make decisions in some arena, what actions are allowed or constrained, ... [and] contain prescriptions that forbid, permit or require some action or outcome" [12, p. 51]. She also maintained that the rule-sets were conventionally agreed (ideally by those affected by them), mutually understood, monitored and enforced; that they were nested; and that they were mutable.

On the issue of nesting, Ostrom [12, p. 52] distinguished three levels of rules. These were, at the lowest level, *operational choice* rules, which were concerned with the processes of resource appropriation, provision, monitoring and enforcement. In the middle level, *collective choice* rules were concerned with selecting the operational rules, as well as processes of policy-making, role assignment and dispute resolution. At the highest level, *constitutional choice* rules indirectly affected the operational rules by determining who is eligible to, and what specific rules are to be used to, define the set of collective choice rules.

The nesting of rules was important for the process of *institutional change* for two reasons. Firstly, the changes which constrain action at a lower level occur in the context of a 'fixed' set of rules at a higher level. Secondly, lower level rules were easier and less 'costly' to change than the higher level rules, thus increasing the stability of strategies and expectations of those individuals having to interact with others in the context of the institutional setting.

Ostrom also observed that there were occasions when the institutions were *enduring*, and others where they were not. Accordingly, eight principles of institutions were identified for *self*-management of common pool resources (CPR) to endure. Of these, three were:

1. Clearly defined boundaries: those who have rights or entitlement to appropriate resources from the CPR are clearly defined, as are its boundaries.
2. Congruence between appropriation and provision rules and the state of the prevailing local environment. For example, an appropriate rule that allows everyone an unrestricted claim on resources is not congruent with the environmental condition where those same resources are scarce.
3. Collective choice arrangements: in particular, those affected by the operational rules participate in the selection and modification of those rules.

It is necessary to identify who is a *member* of the institution, and who is not, as it is precisely the members of the institution who are those affected by modification of the rules. We also need to distinguish specific members who are *empowered* to enact, announce and enforce these modifications.

2.2 Institutionalised Power and Roles

Following the third principle, if the set of working rules defining an institution contains "prescriptions that forbid, permit or require some action or outcome", and specifies formally "who is eligible to make decisions", it is generally not a specific agent that is eligible to make decisions, but instead it is agent that occupies a designated role, that is *empowered* to make those decisions.

Therefore, we need to represent the concepts of role, role assignment [15], and *institutionalised power* [10]. The term institutionalised power refers to that characteristic feature of institutions, whereby designated agents, often acting in specific roles, are empowered to create or modify facts of special significance in that institution (*institutional facts*), through the performance of a designated action, e.g. a speech act.

This necessitates defining a role-assignment protocol that appoints a specific agent to a role. It must also be possible to change which agent occupies that role, for example if the appointed agent leaves the system, performs badly or incorrectly, or is unable to execute the duties associated with the role. To deal with assignment and change, we need dynamic norm-governed specifications.

2.3 Dynamic Specifications

Artikis [2] defined a framework that allowed agents to modify the rules or protocols of a norm-governed system at runtime. This framework defined three components: a specification of a norm-governed system, a protocol-stack for defining how to change the specification, and a topological space for expressing the 'distance' between one specification instance and another.

A specification of a norm-governed system can be (partially) given by defining the permissions, prohibitions and obligations of the agents in the system, and the sanctions and enforcement policies that deal with the performance of prohibited actions and non-compliance with obligations [3].

The protocol stack allowed agents to modify the rules or protocols of a norm-governed system at runtime. This framework defined a set of object level protocols, and assumed that during the execution of an object protocol the participants could start a meta-protocol to (try to) modify the object-level protocol. The participants of the meta-protocol could initiate a meta-meta protocol to modify the rules of the meta-protocol, and so on. In addition to object- and meta protocols, there are also ‘transition’ protocols. These protocols define the conditions in which an agent may initiate a meta-protocol, who occupies which role in the meta-protocol, and what elements (the *degrees of freedom*: DoF) of an object protocol can be modified as a result of the meta-protocol execution.

For example, we need to define who is, and who is not, a *member* of an institution, where agents can join an institution if they satisfy certain criteria, and can be excluded if they do not comply to the rules. We specify two types of method, one for access control and another for exclusion. The type of access control method is *acMethod*, which can be *attribute-based*, whereby if the applicant satisfies certain qualification criteria then it is automatically admitted, or *discretionary*, i.e. an applicant must satisfy another agents’s criteria, who is acting on behalf of the institution in its appointed role. The type of exclusion method is *exMethod*, which can be either by *jury*, in which case the institution members vote on whether or not to exclude a non-complying agent, or again *discretionary*, i.e. some specific agent decides whether or not to exclude a agent.

Each type of method is a DoF, and with two values for each method, this gives four possible *specification instances*. This the basis for defining a *specification space* $\langle T, d\rangle$, where T is the set of all possible specification instances and d is a function which defines a ‘distance’ between any pair of elements in T.

2.4 Linear Public Good (LPG) Game

CPR management by an institution requires that each agent provides to and appropriates resources from the common pool. The agents must comply with the rules concerning provision and appropriation, but in an open system, this includes dealing with intentional violations as well as unintentional ones.

Analysing the problem of individual resource contribution in a CPR is considered as a linear public good (LPG) game [7]. This problem has proved useful for examining the free rider hypothesis, and the incentives for voluntary contributions, in both laboratory-based simulations and agent-based modelling. In a typical LPG game, n people or agents form a group or *cluster*. All cluster members individually possess a quantity of resource. Each cluster member $i, i \in \{1, \dots, n\}$ decides independently to contribute resources $r_i \in [0, 1]$ to the public good. The contributions from the whole cluster are summed and the payoff u_i for each player i is given by:

$$u_i = \frac{a}{n}\sum_{j=1}^{n} r_j + b(1 - r_i), \quad \text{where} \quad a > b \quad \text{and} \quad \frac{a}{n} < b$$

The first term represents the payoff from the public good (the 'public payoff'), distributed equally among the n cluster members. The second term represents the payoff from the resources withheld from the public good (the 'private payoff') irrespective of how much was contributed individually and collectively. The coefficients a and b represent the relative value of the public/private payoffs respectively. If the conditions on a and b hold, a rational but selfish agent has the incentive to contribute 0 to the public good, i.e. free riding, so that:

– The dominant strategy is defect: the individual allocation is greatest when a member contributes 0 and every other cluster member contributes 1;
– The collective payoff is least when every cluster member contributes 0, but increases as contributions increase;
– The collective payoff is greatest when all cluster members contribute fully.

3 Formal Characterisation

In this section, we describe a methodology for sociologically-inspired computing [9], apply it to cast Ostrom's definition of an institution (Section 2.1) as a dynamic norm-governed specification (Section 2.3), and derive a formal model of a multi-agent system to play the n-player iterated linear public good game.

3.1 Methodology

A methodology for sociologically-inspired computing is illustrated in Figure 1.

We start from an observed phenomenon, for example a human social, legal or organisational system. The process of *theory construction* creates a pre-formal 'theory', usually specified in a natural language. Ostrom [12] comes into this category, as it is an evidence-based theory of enduring institutions but without formalism. The process of *formal characterisation* represents such theories in a calculus of some kind, where by calculus we mean any system of calculation or computation that is based on symbolic representation and manipulation. This representation can be at different levels of abstraction depending on the intended

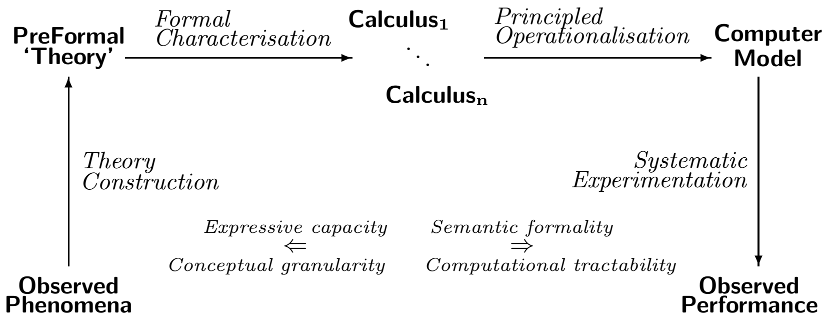

PreFormal 'Theory'
Formal Characterisation
Calculus₁
Calculusₙ
Principled Operationalisation
Computer Model
Theory Construction
Systematic Experimentation
Expressive capacity ⇐ Conceptual granularity
Semantic formality ⇒ Computational tractability
Observed Phenomena
Observed Performance

Fig. 1. Sociologically-Inspired Computing

role of the calculus: expressive capacity or conceptual granularity with regard to 'theory'; computational tractability or semantics with regard to implementation. The step of *principled operationalisation* embeds such formal representations in simulations which include detailed implementation of individual agents.

3.2 Institutions as Dynamic Specifications

The three elements of Artikis' framework [2] were a norm-governed specification, a protocol stack, and a specification space.

Firstly, the institutional rules of Ostrom are characterised as a norm-governed specification. As such, the specification will define the following aspects of institutional action: the physical capabilities, institutionalised powers, permissions, prohibitions and obligations of the agents; the sanctions and enforcement policies that deal with the performance of prohibited actions and non-compliance with obligations; and the designated roles of empowered agents. The Event Calculus (EC) [11] is used as the calculus for formal characterisation.

Secondly, the nesting of operational-choice rules within collective-choice rules within constitutional-choice rules is treated by the object, meta- and meta-meta-protocols, and we handle *institutional change* within the framework of dynamic specifications. This proposal is illustrated in Figure 2. We show the type of rule in Ostrom's framework on the left, and the protocol we will specify in the Artikis framework on the right. For example, the appropriation and provision operational choice rules of Ostrom are implemented by actions in an object-level protocol for the LPG game; similarly the monitoring and enforcement rules are implemented by protocols for access control and exclusion. At the meta-level, there are protocols which change object level rules, i.e. through role assignment and choosing the DoF values for the access control and exclusion methods.

The Artikis framework originally defined the specification space as a metric space. In practice, we find this too restrictive and instead of a metric space we represent the set of specification instances T as nodes on a graph with a constant 'distance' k between any two nodes, i.e. $\forall l_1, l_2 \in T, d(l_1, l_2) = k$. The

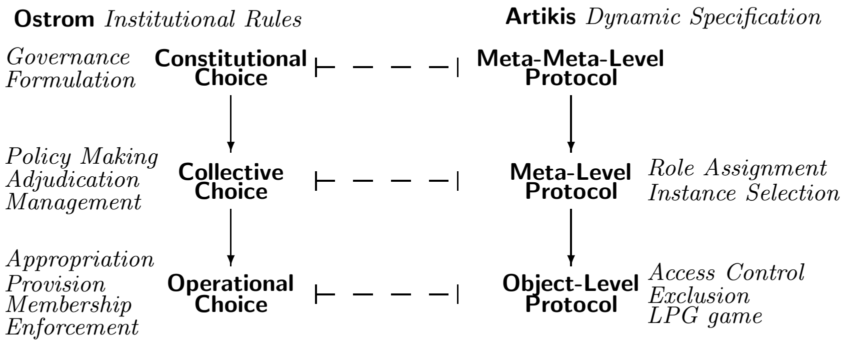

Ostrom Institutional Rules
Artikis Dynamic Specification
Governance Formulation
Constitutional Choice
Meta-Meta-Level Protocol
Policy Making Adjudication Management
Collective Choice
Meta-Level Protocol
Role Assignment Instance Selection
Appropriation Provision Membership Enforcement
Operational Choice
Object-Level Protocol
Access Control Exclusion LPG game

Fig. 2. Institutional rules as Protocol Stack

specification space used here is given below in Section 4.4. (Note that we will not use d further in these experiments; however, we find it convenient to retain it for further work in representing the 'cost' of modifying operational, collective and constitutional choice rules.)

3.3 Formal Model

We will now instantiate a formal model of an institution for the LPG game. Let $\mathcal{IC}_t$ be a multi-agent system at time t defined by:

$$\mathcal{IC}_t = \langle \mathcal{A}, \mathcal{I}, \mathcal{L}, G, d \rangle_t$$

where (omitting the subscript t if clear from context):

- $\mathcal{A}$ is the set of all agents;
- $\mathcal{I}$ is the set of institutional clusters;
- $\mathcal{L}$ is a norm-governed system specification (defining a specification space T);
- G is the LPG game;
- d is a distance function defined on specification instances of T.

Each institutional cluster $I_t \in \mathcal{I}_t$ is given by:

$$I_t = \langle \mathcal{M}, l, \epsilon \rangle_t$$

where (again omitting the subscript t if clear from context):

- $\mathcal{M}$ is the set of member agents, such that $\mathcal{M} \subseteq \mathcal{A}$
- l is a specification instance of T; and
- ϵ is the cluster's local environment, a pair $\langle \mathit{Bf}, \mathit{If} \rangle$ with Bf the set of 'brute' facts whose values are determined by the physical state, including the average contribution made by members to the cluster; and If the set of 'institutional' facts, whose values are determined by the conventional state, including the roles assigned to members of $\mathcal{M}$.

The intuitive idea is that at each time-point t, the agents in $\mathcal{A}$ will form into clusters $\mathcal{I}$ using the access control method. Each cluster plays a linear public good game, where the members either comply or defect by contributing more or less resources than the cluster average. After the game, non-compliance may be punished by exclusion according to the operational rules.

A specific type of institutional fact recorded in ϵ is which agent is empowered to perform a certain role in each cluster. We identify four roles: *member*, which is the standard role for membership of a cluster in order to participate in G; *gatekeeper*, which is empowered to assign the role of *member*; *monitor*, which is empowered to remove the role of *member*, and *head*, which is empowered to assign to the *gatekeeper* and *monitor* roles.

Therefore the set $\mathcal{L}$ contains the following two rules for role assignment, with (in parenthesis) the role responsible for its enactment and enforcement:

$$\begin{aligned} (\mathit{gatekeeper}) \;\; & \mathit{ocr}_1 : \mathcal{M}^c \times \mathit{acMethod} \rightarrow \mathit{Bool} \\ (\mathit{monitor}) \;\; & \mathit{ocr}_2 : \mathcal{M} \times V(\cdot)_{a \in I} \times \mathit{exMethod} \rightarrow \mathit{Bool} \end{aligned}$$

where $V(\cdot)_{a \in I}$ is a set of expressed preferences on an issue by each *member* agent in cluster I, where $I \in \mathcal{I}$.

The operational choice rule ocr_1 is applied by the *gatekeeper* to map an application to join from an agent not in $\mathcal{M}$ (i.e. the set complement $\mathcal{M}^c$) to a boolean outcome depending on the access control method. A true result means the applicant can be assigned the role of *member*. Similarly, the rule ocr_2 is applied by the *monitor* to map an agent in $\mathcal{M}$ that did not comply with the rules of the LPG game to a boolean outcome using the exclusion method.

4 Action Language Specification

In this section, we illustrate the axiomatisation of the rules in $\mathcal{L}$ using the Event Calculus (EC) [11], and define the graph for the 'specification space'. A summary of the EC is given in [2], and a full EC specification of six of Ostrom's principles in [13]. For space reasons, we do not review EC or reproduce those axioms.

4.1 Fluents (Institutional Facts)

Some of the institutional facts, represented as fluents F of the EC, are as follows. The multi-valued fluent *role_of* has the value *head*, *gatekeeper*, *monitor* or *member* if the agent occupies the associated role, and so participates in the LPG game G for a cluster I, and has the value *none* otherwise.

The multi-valued fluent *acMethod* determines which access control method the gatekeeper must use in determining *member* role assignment. Its value is either *attribute* or *discretionary*. The multi-valued fluent *exMethod* determines which exclusion method the monitor must use in determining *member* exclusion (note that the the jury method requires a winner-determination method. Technically this is a mutable DoF but we will assume it is fixed at plurality here). Three other fluents record the (institutionalised) powers, permissions and obligations of each agent. These are all institutional facts in If. The real-valued fluent *cluster_average*, a physical fact in Bf, records the average contribution of the agents to the public good.

In this specification, we stipulate that agents can occupy only one role in a cluster and that agents can be members of only one cluster. It is straightforward to modify the specification (and the testbed used for the experiments, described in the next section), so that agents can occupy more than one role and be members of more than one cluster, but neither choice fundamentally affects the issue being investigated, i.e. self-regulating sustainable institutions.

4.2 Member Role Assignment (ocr_1)

The dynamic specification of the operational choice rule ocr_1 is given by a role-assignment protocol for membership. An agent can apply for membership to a cluster I if it does not occupy a role in any other cluster:

$$\begin{array}{l} \textit{apply}(A, I) \quad \mathsf{initiates} \quad \textit{applied}(A, I) = \textit{true} \quad \mathsf{at} \quad T \quad \leftarrow \\ \qquad \textbf{not} \quad \textit{role\_of}(A, \_) = \textit{member} \quad \mathsf{holdsAt} \quad T \quad [\text{etc.}] \end{array}$$

The gatekeeper agent is empowered to admit the agent, to the cluster, by an *assign* action, depending on the access control method.

$$\begin{array}{l}
\mathit{assign}(G, A, \mathit{member}, I) \;\; \mathsf{initiates} \;\; \mathit{role\_of}(A, I) = \mathit{member} \;\; \mathsf{at} \;\; T \;\; \leftarrow \\
\qquad \mathbf{pow}(G, \mathit{assign}(G, A, \mathit{member}, I)) = \mathit{true} \;\; \mathsf{holdsAt} \;\; T \\
\mathbf{pow}(G, \mathit{assign}(G, A, \mathit{member}, I)) = \mathit{true} \;\; \mathsf{holdsAt} \;\; T \;\; \leftarrow \\
\qquad \mathit{applied}(A, I) = \mathit{true} \;\; \mathsf{holdsAt} \;\; T \;\; \wedge \\
\qquad \mathit{acMethod}(I) = \mathit{attribute} \;\; \mathsf{holdsAt} \;\; T \;\; \wedge \\
\qquad \mathit{role\_of}(G, I) = \mathit{gatekeeper} \;\; \mathsf{holdsAt} \;\; T \;\; \wedge \\
\qquad \mathit{role\_conditions}(\mathit{member}, A, I) = \mathit{true} \;\; \mathsf{holdsAt} \;\; T \\
\mathbf{pow}(G, \mathit{assign}(G, A, \mathit{member}, I)) = \mathit{true} \;\; \mathsf{holdsAt} \;\; T \;\; \leftarrow \\
\qquad \mathit{applied}(A, I) = \mathit{true} \;\; \mathsf{holdsAt} \;\; T \;\; \wedge \\
\qquad \mathit{acMethod}(I) = \mathit{discretionary} \;\; \mathsf{holdsAt} \;\; T \;\; \wedge \\
\qquad \mathit{role\_of}(G, I) = \mathit{gatekeeper} \;\; \mathsf{holdsAt} \;\; T
\end{array}$$

If the *acMethod* is *attribute*, then the gatekeeper is empowered to assign the role *member* provided the applicant satisfies certain (external) role conditions. The conditions could include, for example, not exceeding a fixed number of non-compliant actions, a duration since the last non-compliant action, and so on.

If the *acMethod* is *discretionary*, then the *gatekeeper* is empowered to assign the role without conditions, according to its (internal) decision-making, which could yet make reference to the external conditions.

4.3 Member Exclusion (ocr_2)

The *monitor* is empowered to exclude a *member* that does not comply with the rules of the game G. For each iteration of G, agents should contribute resources in the interval $[ave_I, 1]$ to comply, where ave_I is the average contribution of resources from the previous iteration (the value of the fluent $\mathit{cluster\_average}(I)$). For each iteration, an agent's default provision is 0, and if it does not *provide* an average (or greater) provision then it is sanctioned:

$$\begin{array}{l}
\mathit{provide}(A, R, I) \;\; \mathsf{initiates} \;\; \mathit{provision}(A, I) = R \;\; \mathsf{at} \;\; T \;\; \leftarrow \\
\qquad \mathbf{pow}(A, \mathit{provide}(A, R, I)) = \mathit{true} \;\; \mathsf{holdsAt} \;\; T \\
\mathit{provide}(A, R, I) \;\; \mathsf{initiates} \;\; \mathit{sanctioned}(A, T, I) = \mathit{true} \;\; \mathsf{at} \;\; T \;\; \leftarrow \\
\qquad \mathbf{pow}(A, \mathit{provide}(A, R, I)) = \mathit{true} \;\; \mathsf{holdsAt} \;\; T \;\; \wedge \\
\qquad \mathit{provision}(A, I) = R \;\; \mathsf{holdsAt} \;\; T \;\; \wedge \\
\qquad \mathit{cluster\_average}(I) = \mathit{Ave} \;\; \mathsf{holdsAt} \;\; T \;\; \wedge \;\; R < \mathit{Ave} \\
\mathbf{pow}(A, \mathit{provide}(A, R, I)) = \mathit{true} \;\; \mathsf{holdsAt} \;\; T \;\; \leftarrow \\
\qquad \mathit{role\_of}(A, I) = \mathit{member} \;\; \mathsf{holdsAt} \;\; T
\end{array}$$

If the *exMethod* is discretionary, then the *monitor* agent G can exclude the applicant A (or not) as it decides. If the *exMethod* is jury, then the *monitor* must have called for a vote on the issue of the exclusion of A:

$$
\begin{array}{l}
\mathit{exclude}(G, A, \mathit{member}, I) \quad \mathsf{initiates} \quad \mathit{role\_of}(A, I) = \mathit{none} \quad \mathsf{at} \quad T \quad \leftarrow \\
\qquad \mathbf{pow}(G, \mathit{exclude}(G, A, \mathit{member}, I)) = \mathit{true} \quad \mathsf{holdsAt} \quad T \\
\mathbf{pow}(G, \mathit{exclude}(G, A, \mathit{member}, I)) = \mathit{true} \quad \mathsf{holdsAt} \quad T \quad \leftarrow \\
\qquad \mathit{role\_of}(G, I) = \mathit{monitor} \quad \mathsf{holdsAt} \quad T \quad \wedge \\
\qquad \mathit{exMethod}(I) = \mathit{discretionary} \quad \mathsf{holdsAt} \quad T \\
\mathbf{pow}(G, \mathit{exclude}(G, A, \mathit{member}, I)) = \mathit{true} \quad \mathsf{holdsAt} \quad T \quad \leftarrow \\
\qquad \mathit{role\_of}(G, I) = \mathit{monitor} \quad \mathsf{holdsAt} \quad T \quad \wedge \\
\qquad \mathit{exMethod}(I) = (\mathit{jury}, \mathit{WDM}) \quad \mathsf{holdsAt} \quad T \quad \wedge \\
\qquad \mathit{ballot}(\mathit{exclude}(A), I) = V \quad \mathsf{holdsAt} \quad T \quad \wedge \\
\qquad \mathit{winner\_determination}(\mathit{WDM}, V, \mathit{true}) \\
\mathbf{per}(G, \mathit{exclude}(G, A, \mathit{member}, I)) = \mathit{true} \quad \mathsf{holdsAt} \quad T \quad \leftarrow \\
\qquad \mathit{role\_of}(G, I) = \mathit{monitor} \quad \mathsf{holdsAt} \quad T \quad \wedge \\
\qquad \mathit{sanctioned}(A, T', I) = \mathit{true} \quad \mathsf{holdsAt} \quad T \quad \wedge \quad T' < T
\end{array}
$$

Note that the *monitor* is empowered to exclude any *member*, but it is only *permitted* to exercise that power when that member has been sanctioned (and, when, the exclusion method is *jury*, only when the vote is in favour of exclusion). This means that when the monitor excludes an agent, that agent really is excluded and it has no role in the institution. However, an excluded agent can appeal against an invalid use of the power, the *monitor* could be removed from the role, and so on. This is a higher-order effect which is beyond the scope of the current paper. Furthermore, voting and winner determination has been studied in this context in [13], but is not considered further here.

4.4 Specification Space

These rules in $\mathcal{L}$ effectively define four DoF (degrees of freedom): the selection of the *acMethod* and the selection of the *exMethod*, and the assignment to the *gatekeeper* role and the assignment to the *monitor* role. Meta-level protocols for role assignment and instance selection can be specified in the EC, as above, but for space constraints are omitted here.

Since no agent can occupy both roles *monitor* and *gatekeeper* in a cluster I with n members, it follows that there are $4n^2 - 4n$ possible specification instances. Rather than dynamically computing the entire space for each cluster and trying to determine the 'optimal' configuration, we separate the 'specification instance' selection function into two dimensions.

For the first dimension, the decision of which agent to assign to the role of *monitor* or *gatekeeper*, we define a family of preference functions, some based on relevant properties of the agent (e.g. compliance probability, time already spent in the role, etc.) and some not (e.g. random, nominative proximity, etc.). Each agent is associated with a subset of these functions, and applies them when

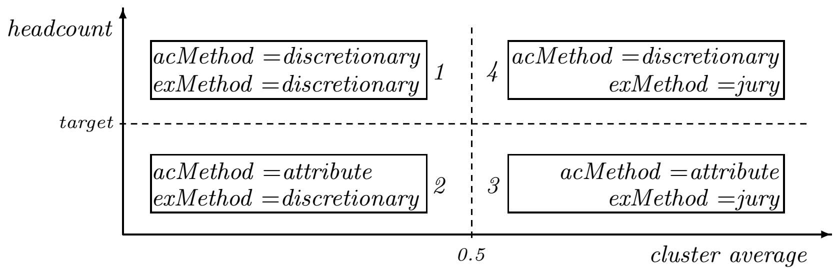


Fig. 3. Specification Space

voting for either *monitor* or *gatekeeper*. The *head* is empowered to assign the role to the agent with the most votes according to a winner determination method.

For the second dimension, the selection of *acMethod* and *exMethod*, we define two criteria. The first criteria is a *target* membership: this value is a trade-off between total cost of ownership (which is too high if the *headcount* is less than the target) and the quality of service (which it too low if the *headcount* is more than the target). The second criteria is the average probability of compliance in the LPG game. For each agent, we define a probability distribution for voting for a change in the specification according to these criteria. As a result, the selected specification instance falls into one of the quadrants *1–4* as shown in Figure 3.

5 Experimental Results

This section describes the implementation of a testbed and experimental results, evaluating the performance of an institutional approach to the LPG game.

5.1 Testbed Implementation

The control loop for the testbed is shown in Algorithm 1. A run starts with the random generation of a population of M agents and N clusters, and sets the time-point t to 0. One agent is assigned to the *head* role in each cluster.

To introduce an element of volatility, each agent (except the *head*) is associated with a random cycle of length x time-points, of which it is 'present' for y (time-points) and 'absent' for z, such that $x = y + z$. In step 5, those agents that transition from absent to present are given the status *present*, but have no role in any cluster; those that transition from present to absent leave their cluster and whatever roles they occupy, and their status is *absent*.

In step 6 and 7, the member agents vote for a monitor or gatekeeper, if one or both of those roles have been vacated. Then, *present* agents which are not members of a cluster (either because they became present from step 5, they failed to get into a cluster or they were excluded from a cluster in the previous time-point) apply to a cluster. The *gatekeeper* applies the *acMethod* as presented in Section 4.2 and assigns the applicant to the role of *member* or it is rejected. In step 9, member agents play the LPG game within each cluster, and the cluster average is updated. Non-complying agents may then be excluded as per Section 4.3 using the operational *exMethod*. Afterwards, each cluster's *acMethod* and *exMethod* are updated by a vote of its members, the time-point is incremented and the cycle repeats.

Algorithm 1. Control Loop for CPR testbed.

```
1.  $generate\_agents(M, A)$
2.  $generate\_clusters(N, C)$                          %assign (designate) head
3.  $t \leftarrow 0$
4.  repeat
5.    $update\_present(A)$
6.    $gatekeeper\_role\_assignment(A, C)$              %role assignment by vote
7.    $monitor\_role\_assignment(A, C)$                 %role assignment by vote
8.    $member\_role\_assignment(A, C)$                  %use acMethod, Section 4.2
9.    $public\_good\_game(A, C)$                        %play LPG game, Section 2.4
10.   $member\_exclusion(A, C)$                         % %use exMethod, Section 4.3
11.   $update\_clusters(C)$                             %update acMethod, exMethod by vote
12.   $t \leftarrow t + 1$
13. until $t = \ldots$
```

5.2 Agent Strategies

When a population of agents is generated, a bundle of information is associated with each agent. This includes its name, up-time, down-time, initial cluster and role assignment (may be none), and its strategy for the LPG game. This strategy is given by a probability of complying with the rules of the game. Therefore the contribution $r_{i,t}$ that an agent i makes at time t is given by:

$$\begin{aligned} r_{i,t} &= Ave_{(t-1)} + rnd(1) \cdot (1 - Ave_{(t-1)}), \quad \text{if} \quad rnd(1) \geqslant pc_{i,t} \\ &= rnd(1) \cdot Ave_{(t-1)}, \quad \text{otherwise} \end{aligned}$$

where $pc_{i,t}$ is the probability of i's compliance at t and $Ave_{(t-1)}$ the average cluster contribution from the last time-point. As a result the contribution is in the interval $[Ave, 1]$ if a random number generated in the interval $[0, 1]$ is greater than the probability of compliance, and is in the interval $[0, Ave]$ otherwise.

The agents update their probability of compliance in the next time-point according to a form of social influence. Letting $|I|$ denote the *headcount* for the number of agents in cluster I and $|I|^+$ denote the number of agents in I which complied in the current round, then:

$$\begin{aligned} pc_i(t+1) &= pc_i(t) + \alpha \cdot (1 - pc_i(t)), \quad \text{if} \quad (|I|^+ \,/\, |I|) \geqslant 0.5 \\ &= pc_i(t) - \beta \cdot pc_i(t), \quad \text{otherwise} \end{aligned}$$

where α and β are globally defined coefficients in $[0, 1]$ determining the rate of positive and negative *reinforcement* respectively. If a majority of agents complies in the current round, then the likelihood of each one complying in the next round is increased, and vice versa.

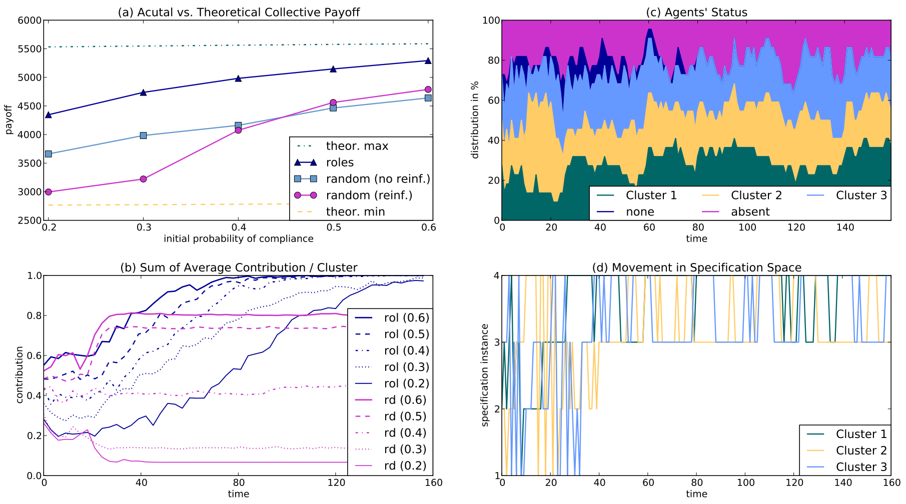


Fig. 4. Experimental Results

5.3 Evaluation

The experiments were run with 22 agents and 3 clusters, the same population distribution was used for 10 runs each of 160 time-points, and the results averaged. The independent variable was the initial probability of compliance for each agent, for the entire population. The dependent variables were the combined payoffs and the combined cluster averages. The compliance reinforcements rates were $\alpha = \beta = 0.05$ and the LPG game coefficients were $a = 2$ and $b = 1$.

Figure 4(a) shows the results for 5 populations starting with a probability of non-compliance 0.2, through to 0.6. There is a theoretical maximum collective reward at each time-point, which is the number of agents 'present', independent of whether they were in a cluster or not, multiplied by 2 (the coefficient a), and a theoretical minimum, i.e. the number of present agents (since the coefficient b is 1). Three sets of 10 runs are shown: for a population with role assignment and with updating the probability of compliance (reinforcement), without role assignment (i.e. random) but with reinforcement, and without either role assignment or reinforcement (i.e. random with a fixed probability of compliance). The graph shows that with roles and reinforcement, even at low levels of initial compliance, the cluster performs better than random, with or without reinforcement. As the initial compliance increases, the overall payoff increases and starts to approximate the payoff of the theoretical ideal. Reinforcement without rules is about as effective as without either reinforcement or rules at higher levels of initial compliance, but actually worse at lower levels of initial compliance.

Figure 4(b) displays the cumulative moving cluster averages for the same agent populations, totalled over the 10 runs each, for clusters with role assignment and reinforcement and for clusters without role assignment (random) but with reinforcement. The graph shows that, even at low levels of initial compliance, the agents using role assignment and reinforcement were able to form stable and sustainable clusters, with full compliance pervasion as each agent is contributing 1.0. Without role assignment, each run tends either to full non-compliance or full compliance, and as expected this happens around 50% of the time with initial compliance probability of 0.5/0.6, and much less often at lower levels of initial compliance. Thus the sum of the average cluster contributions, taken over 10 runs, is then less than full compliance.

Figure 4(c) shows a typical distribution of agents to clusters. Initial probability of non-compliance is 0.5, coefficients α, β, a, b as before. Each cluster has a target membership of 6, or approximately one-third of the expected total number of agents expected to be present at any one time-point. The total number of present but non-member agents (as a result of exclusions or rejections) reduces to zero by about time-point 100; after that, the distribution of present agents to clusters is more or less even.

Finally, Figure 4(d) shows the change of access control and exclusion methods in the specification space. About the time agents stop being excluded, the access control and exclusion methods oscillate between specification instances 3 and 4, i.e. the cluster average is high so the access control method sticks at discretionary, and the exclusion method varies according to the number of agents present. This confirms that the cluster average was so high it was only the headcount that was affecting which specification instance the agents were using.

The robustness of the institutions and the distribution of rewards suggests that the institution is, in some sense, both ‘fair’ and ‘enduring’. On that basis, we contend that the management of the shared contributed resources, which is the responsibility of the institutions, is sustainable.

6 Related and Further Work

The linear public good game used here was studied in [17], and took an evolutionary algorithms approach to solving the social dilemma it presented, in contrast to the explicit representation of rules and institutions in this work.

Axtell outlines a dynamic model for team formation based on evolutionary game theory [4], in which a set of agents attempt to form a stable coalition. This work analysed the conditions under which agents cooperate, and demonstrated that groups become unstable beyond a certain size due to free riding.

This argument was extended in [5] to volatile populations of agents, who leave and join teams based on a local view of utility rather than through a cyclical up/down time. There is scope to include such behaviour in our system. This work also argues that the mathematical complexity of such volatile systems precludes any analytic results. Our axiomatisation in the Event Calculus supports off-line tasks like proving properties, and supports direct computational implementation for experimental investigation, when the randomness in the system makes the system behaviour inherently unpredictable.

In service-oriented computing, there is increasing attention being paid to applications of cloud computing for enterprise management and business delivery, in particular the real-time on-demand provisioning of Software-as-a-Service

(SaaS), Infrastructure-as-a-Service (IaaS), and so on. It is interesting to cast this problem as a non-cooperative game of group formation and resource management problem [1]. In that work, a game theoretic approach is proposed, based on competing SaaS providers managing IaaS provider capacity. It would be worthwhile to investigate the effects of the clusters themselves as competing entities, each offering flat-rate, on-demand and spot-market resource access, and to model this from an institutional perspective.

Our aim here has been to leverage Ostrom's work for *agent-based software engineering*, but there is related research from the perspective of *agent-based modelling*. This reveals many additional parameters to consider in developing experiments to test the emergent property of endurance. For example, [8] investigates whether or not people are prepared to invest their own resources in endogenous rule change, e.g. from open access to private property. There is much scope for investigating richer agent strategies in this context.

On sustainability, we draw attention to the MAELIA project [6], which is building a multi-agent platform to model the interaction, from a network perspective, of agents, actions and norms on renewable resources. Their emphasis is understanding how to analyse and optimise policy with respect to sustainability, and their representation of norms is not grounded in an action language. Scaling up the system described here and deploying it for demand-side management of physical resources (where the resource consumers are also resource providers) is a substantial and significant challenge for further work (cf. [16]).

Finally, the testbed offers several directions for further research. Currently, the testbed implements the EC axioms and generates the actions, but does not put the narrative through an EC engine. We are investigating use of the cached EC for this purpose. This also introduces scope for partial observation and unintentional error, and examining how the costs of monitoring and dispute resolution affect compliance. A representation of cost could also be used to explore the specification space and strategies to use of the distance function d to determine a preferred specification instance. This also relates to Ostrom's comments about the cost of changing operational- and collective-choice rules.

7 Summary and Conclusions

In summary, we applied a methodology for sociologically-inspired computing to a (pre-formal) theory of socio-economics. Our aim was to cast self-governing institutions for common pool resource management in the framework of dynamic (norm-governed) specifications. The resulting formal model was given a complete axiomatisation in the Event Calculus and an experimental testbed was designed and implemented to investigate the dynamic behaviour of the system. The results showed that the distributed self-organising system was robust even to initially non-compliant populations, that its behaviour approximated the theoretically ideal centralised solution with 'perfect' agents, and that the distribution of rewards indicated that the institution was, in some sense, 'fair' and sustainable.

Our technical conclusion is that the application of institutional rules and institutional change for CPR management has a beneficial impact on autonomous and autonomic multi-agent systems. The challenge is to apply these ideas in socio-technical multi-agent systems as part of a sustainable infrastructure for common pool resource management, such as water and energy.

References

1. Ardagna, D., Panicucci, B., Passacantando, M.: A game theoretic formulation of the service provisioning problem in cloud systems. In: WWW 2011, pp. 177–186 (2011)
2. Artikis, A.: Dynamic protocols for open agent systems. In: Proc. AAMAS 2009, pp. 97–104. IFAAMAS (2009)
3. Artikis, A., Sergot, M., Pitt, J.: Specifying norm-governed computational societies. ACM Transactions on Computational Logic 10(1), 1–42 (2009)
4. Axtell, R.: Non-cooperative dynamics of multi-agent teams. In: Castelfranchi, C., Johnson, W. (eds.) Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems, pp. 1082–1089 (2002)
5. Axtell, R.: What economic agents do: How cognition and interaction lead to emergence and complexity. The Review of Austrian Economics 20(2), 105–122 (2007)
6. Boulet, R., Mazzega, P., Jouve, B.: Environmental, social and normative networks in the maelia platform. In: Poblet, M., Schild, U., Zeleznikow, J. (eds.) Proc. ICAIL Worshop Legal & Decision Support Systems, pp. 83–93 (2009)
7. Gaechter, S.: Conditional cooperation: Behavioral regularities from the lab and the field and their policy implications. Discussion Papers 2006-03, The Centre for Decision Research and Experimental Economics, School of Economics, University of Nottingham (2006)
8. Janssen, M., Goldstone, R., Menczer, F., Ostrom, E.: Effect of rule choice in dynamic interactive spatial commons. International Journal of the Commons 2(2), 288–311 (2008)
9. Jones, A., Pitt, J., Artikis, A.: On the analysis and implementation of normative systems: Towards a methodology: In: Cranefield, S., Noriega, P. (eds.) PreProceedings COIN@AAMAS 2011, pp. 47–56 (2011)
10. Jones, A., Sergot, M.: A formal characterisation of institutionalised power. Journal of the IGPL 4(3), 427–443 (1996)
11. Kowalski, R., Sergot, M.: A logic-based calculus of events. New Generation Computing 4, 67–95 (1986)
12. Ostrom, E.: Governing the Commons. CUP (1990)
13. Pitt, J., Schaumeier, J., Artikis, A.: The axiomatisation of socio-economic principles for self-organising systems. In: Proceedings SASO 2011 (2011)
14. Raya, M., Hubaux, J.P.: Securing vehicular ad hoc networks. Journal of Computer Security 15(1), 39–68 (2007)
15. Sandhu, R., Ferraiolo, D., Kuhn, R.: The NIST model for role-based access control: Toward a unified standard. In: 5th ACM Workshop Role-Based Access Control, RBAC 2000, pp. 47–63 (2000)
16. Strbac, G.: Demand side management: Benefits and challenges. Energy Policy 36(12), 4419–4426 (2008)
17. Yamashita, T., Axtell, R., Kurumatani, K., Ohuchi, A.: Investigation of mutual choice metanorm in group dynamics for solving social dilemmas. In: Kurumatani, K., Chen, S.-H., Ohuchi, A. (eds.) IJCAI-WS 2003 and MAMUS 2003. LNCS (LNAI), vol. 3012, pp. 137–153. Springer, Heidelberg (2004)

An Agent-Based Extensible Climate Control System for Sustainable Greenhouse Production

Jan Corfixen Sørensen[1], Bo Nørregaard Jørgensen[1],
Mark Klein[2], and Yves Demazeau[3]

[1] The Maersk Mc-Kinney Moller Institute, University of Southern Denmark,
Campusvej 55, 5230 Odense M, Denmark
{jcs,bnj}@mmmi.sdu.dk
HTTP://www.mmmi.sdu.dk

[2] Center for Collective Intelligence, MIT Sloan School of Management
Five Cambridge Center NE25-754, Cambridge MA 02139, US
m_klein@mit.edu
HTTP://cci.mit.edu/klein

[3] Laboratoire d'Informatique de Grenoble
CNRS, BP 53 38041 Grenoble, France
Yves.Demazeau@imag.fr
HTTP://membres-lig.imag.fr/demazeau

Abstract. The slow adoption pace of new control strategies for sustainable greenhouse climate control by industrial growers, is mainly due to the complexity of identifying and resolving potentially conflicting climate control requirements. In this paper, we present a multi-agent-based climate control system that allows new control strategies to be adopted without any need to identify or resolve conflicts beforehand. This is achieved by representing the climate control requirements as separate agents. Identifying and solving conflicts then becomes a negotiation problem among agents sharing the same controlled environment. Negotiation is done using a novel multi-objective negotiation protocol that uses a generic algorithm to find an optimized solution within the search space. The multi-agent-based control system has been empirically evaluated in an ornamental floriculture research facility in Denmark. The evaluation showed that it is realistic to implement the climate control requirements as individual agents, thereby opening greenhouse climate control systems for integration of independently produced control strategies.

Keywords: Feature interaction, Negotiation, Resource contention.

1 Introduction

In Northern Europe, the production of ornamental pot plants depends on greenhouses equipped with artificial heating and lighting systems, as heat and light are here restricting climatic factors for growth. To make this production ecologically and economically sustainable there is a critical need for energy-efficient climate control strategies that do not compromise product quality. Due to its urgency this issue has attracted the attention of an increasing number of researchers during the past decade and several research projects have produced promising control strategies [1,2,3,4,5]. Contrary to all expectations the industrial adoption pace of those control strategies has been very slow. The main reason seems to be the intrinsic complexity of combining climate control requirements of control strategies originating from independent research projects, as the optimal greenhouse climate prescribed by the climate control requirements of one control strategy may differ from or even conflict with the climates prescribed by others. Basically, independence of work implies that control strategies may have different requirements for the same climatic growth factors at the same time. If the span between requirements for a shared growth factor is narrow, it may be possible to combine the corresponding control strategies. However, if the span is broad, the requirements of the control strategies are most likely conflicting, making their combination infeasible.

Generally, when combining independently-procured control strategies they become implicitly interrelated through sharing of resources in their environment, which is recognized as a typical cause of conflicts among requirements of individual program features [6,7,8]. This is referred to as the feature interaction problem [9]. *Feature interactions* occur whenever the modification or addition of a system feature interferes with the correctness of other system features. In the worst-case scenario, such feature interactions can compromise the correctness of the overall system behavior and cause unexpected runtime faults that may lead to system failure. Hence, creation of an independently extensible greenhouse control system, in which feature interactions do not happen, requires an implementation approach that is capable of coordinating the effects of independent control strategies on shared growth factors in such a way that the requirements of all control strategies are satisfied. Multi-agent systems provide an implementation approach that can support independent extensibility by modeling the units of composition as autonomous agents. Using multi-agent systems allow us to achieve separation of specifications by implementing each climate control requirement as an agent. Hence, prevention of feature interactions becomes a question of coordinating the agents' actions on the controlled environment such that the goal of each agent is satisfied without compromising the goals of others. As the desired effects of the agents' goals for the shared growth factors are described by non-linear multi-variable functions, the coordination of the agents' actions on the environment constitutes an open-ended multi-objective negotiation problem with non-linear utility functions. In this paper, we propose a multi-agent system that allows independent extensibility of greenhouse climate control systems by using a novel multi-objective negotiation protocol to find an optimized greenhouse climate which satisfies the requirements of all control strategies.

The paper is organized as follows. Section 2 describes the typical setup for greenhouse climate control. Section 3 presents requirements for an energy-efficient production. Section 4 presents our multi-agent system. Section 5 describes the implementation of our negotiation protocol. Experimental validation of our approach is presented in Section 6 using a data set obtained from a production greenhouse. Section 7 discusses related work. Section 8 highlights future work. Finally, we conclude our work in Section 9.

2 Greenhouse Climate Control Setup

In general, a greenhouse climate control system is a computer system that controls the climate-related factors for growth by sensing and manipulating the greenhouse climate through the use of sensors and actuators. Sensors are used to measure the actual levels for each of the growth factors while actuators are used to change them. Measured growth factors include temperature, light, CO_2, and humidity. Actuators include the lighting, heating, CO_2, window, curtain and irrigation subsystems of the greenhouse. The lighting subsystem adds supplemental light when the present natural light level is insufficient for sustaining the required plant growth. The heating subsystem is used to maintain the right temperature during cold periods. The CO_2 subsystem doses CO_2 to increase the photosynthesis efficiency of the plants. The window subsystem lowers the temperature and the humidity by opening the windows. Similarly, the curtain subsystem can lower the temperature by shading the plants. Furthermore, curtains are used at night during the winter season to isolate the greenhouse. The irrigation system is responsible for watering the plants. The control of these subsystems is linked to sensor readings through the combined control strategy of the greenhouse's climate control system. The combined control strategy prescribes what is considered to be the optimal greenhouse climate in terms of *temperature*, *light*, *CO_2* and *humidity levels*. The combined control strategy must also coordinate the subsystems of the climate control system, such that no unwanted interactions emerges. For instance, the CO_2 subsystem should not dose CO_2 while the windows are open, as the CO_2 will simply diffuse before it is absorbed by the plants' photosynthesis process. Hence, coordination of the subsystems is vital to ensure the correctness of the climate control. Figure 1 depicts the actual setup of our climate greenhouse control system.

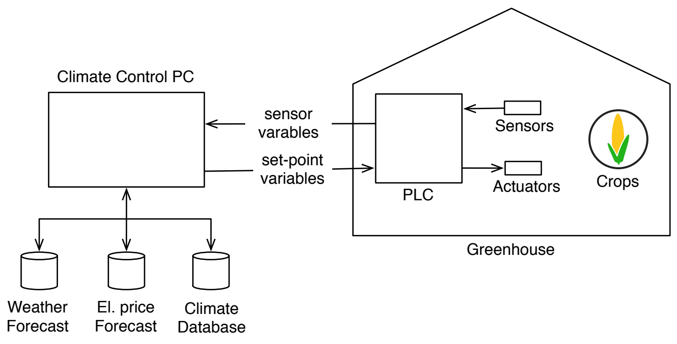


Fig. 1. Climate Control System Overview

Our climate control system consists of a climate control PC that is connected to three database servers for accessing weather forecasts, electricity prices and historical climate data. Additionally, the climate control PC is connected to a Programmable Logic Controller (PLC) that is physically connected to the sensors and actuators. The role of the PLC is to convert signals from the sensors into sensor variables that can be read by the control strategy running on the climate control PC. Furthermore, the PLC converts set-point variables from the climate control PC into signals that can be sent to the physical actuators.

3 Requirements in Greenhouse Climate Control

Requirements for energy-efficient greenhouse production are closely related to the use of artificial heating and lighting systems, as heat and light constitute the restricting climatic growth factors from late autumn to early spring. The IntelliGrow climate control system introduced in [1] provides a control strategy for efficiently reducing the amount of energy required for heating. The system uses a mathematical model of the plants' photosynthesis process to optimize the temperature and CO_2 levels according to the actual light level in the greenhouse. Compared to traditional climate control strategies, which use constant temperature settings for predefined time periods, IntelliGrow does not waste energy on unnecessary heating under low light conditions. A similar approach was demonstrated to reduce energy consumption for use of supplemental light in [10]. Here the same photosynthesis model is used together with weather forecasts and electricity prices to compute a light plan that optimizes the photosynthesis gain with respect to energy consumption and electricity costs. The result is a growth and energy-related control strategy that does not compromise product quality. Other issues may also influence the use of artificial lighting. For instance, during winter working light is required in the early morning and late afternoon. By inspecting the proposed approaches and work-related restrictions we can now identify the following requirements:

$[r_{LightHour}]$ is a requirement to ensure light to be switched on at specific hours. For example, artificial lighting could be used as working light during dark winter mornings.

$[r_{DarkHour}]$ is a requirement to ensure a fixed number of dark hours. For example, light can be forced off during specific dark hours. Most cultivars require at least two hours of total darkness during the night.

$[r_{OptimalPhotosynthesis}]$ is a requirement to ensure that the temperature and CO_2 levels are optimal with respect to the actual light level in the greenhouse.

$[r_{GrowthGoal}]$ is a requirement to ensure that a specific growth goal expressed as an accumulated photosynthesis gain is achieved.

$[r_{MinLightPrice}]$ is a requirement that minimizes the price of the light plan for a given period based on forecasted electricity prices.

4 Multi-Agent-Based Control System

The premise for our approach to the feature interaction problem is that the feature interaction problem can be perceived as a search problem. In the search problem, each climate control requirement is represented by an agent and the feature interactions are represented as conflicts between the agents' goals. The objectives are defined in terms of actuator set points that each agent needs to agree upon to fulfil the climate control requirements of the combined control strategies. The resolution of the feature interactions is accomplished by a *trusted negotiator* through negotiation with the agents over the set of objectives.

An agent has beliefs about the environment in terms of sensor inputs, desires in terms of goals that the agent attempts to fulfil to accomplish the climate control requirement it represents, and intentions to accept or reject the proposed options from the negotiator based on its internal goals. Agent goals are specified in terms of the growth factors that is indirectly influenced by the sensors and actuators through the environment. Therefore, an agent can only be added if the corresponding sensors and actuators are present in the control system. For example, if an agent's goals is specified in terms of light levels, then a light sensor needs to be represented in the control system. Likewise, sensors and actuators can only be removed from the control system, if none of the agents' goals indirectly depends on them. Typically, the set of sensors and actuators stay constant over time and are often added/removed only when the physical greenhouse configuration is changed.

The trusted negotiator manages the negotiation protocol and creates options for a consensus solution that has potential to fulfil the climate control requirements represented by the agents. The negotiator starts a negotiation for each control cycle by suggesting a number of options that have potentials to become a solution. An option is the negotiator's assignment of values for actuator set points based on a given set of sensor-input values. The negotiation process is governed by a protocol which specifies the interaction between the negotiator and the agents, ensuring that only allowed messages are sent and that the messages are sent in the right order. The protocol constitutes four different messages: *option*, *accept*, *satisfy* and *alert* messages. The negotiator asks each agent if the proposed option is acceptable. Each agent evaluates each proposed option against its internal goals and responds back to the negotiator with an accept message if the option is within the boundary conditions of its goals. Agents that cannot accept a proposed option responds with a *reject* message. Additionally, if an agent accepts the proposed option, the negotiator asks the agent how well its goals are met. The agent answers back with a satisfy message that specifies a value for how well the goals of the agent are satisfied based on evaluation of the proposed option. A satisfy message is expressed in terms of an objective criteria value in the interval $[0; 1]$. A value of zero means that the goals are fully satisfied by the option while a value of one means the goals are not well satisfied by the proposed option. The negotiator computes the overall *satisfaction degree* for the proposed option given the accept and satisfy messages from the agents. The trusted negotiator repeatedly generates and proposes options to the agents until the negotiation process terminates. The negotiation process terminates after a specified amount of time depending on the time interval between control cycles. At termination of the negotiation process, the negotiator selects the option with the best satisfaction degree as the solution that can be effectuated by the control system.

In cases where no acceptable solution can be found, the system will continue to run, but the negotiator will send an *alert* message to the user of the system that explains which agents are in conflict with each other. Based on information about the conflicting agents and information about which set points they share, the domain expert user of the system can make an informed adjustment of the conflicting goals to resolve the conflicts. *Explanation* of conflicts is an important feature of our approach as it may be impossible to find solutions in situations where conflicts emerge as a consequence of conflicting requirements. In such situations the goals of the agents need to be relaxed by an informed decision made by the user of the system.

5 Negotiation-Based Coordination

Most approaches to multi-objective problems are based on an optimization process. Focus on optimization requires definition of a global optimization criterion that the solution should get close to. In our open-ended multi-objective control domain, it is difficult to establish such an optimization criterion because information is limited. By definition, an extensible system is never complete and as a consequence optimization approaches that require complete information cannot be applied to independently extensible systems [11]. In other words, the agents can only know about their shared environment and has no information about the other agents. For that reason, agents will act with bounded rationality rather than with economical rationality in terms of utility. In open-ended environments with limited information, it is not clear what defines the best solution. The main limitation of information in our domain is how the plants will be affected by control parameters, since there are no complete integrated plant models available. That is, models only represent parts of the environment. Even if an optimization criterion can be defined, it is not certain that a solution can be found because of limited information. For example, the size of search space for an artificial light-control system with a 24-hour light-plan output would be 2^{24}, assuming light is turned on/off once per hour. Thus, the size of search space explodes as the number of control set points increases. Last but not least, if there is an optimal solution, it is uncertain whether it can be found within reasonable time with current computing power [12].

Our approach, therefore, focuses on satisficing rather than optimization to overcome the problem of limited information [13,14]. Instead of defining a global optimization criterion, we define a *satisficing criterion* that a given solution should fulfil. In other words, a good enough solution is a solution that satisfies the boundary constraints of the agents' goals. A satisficing solution may or may not be an optimal economic solution. The negotiation process is implemented as a genetic algorithm that is executed for every control cycle of the control system. The process is described in Pseudocode 1 and includes four phases: 1) create initial options, 2) evaluate each option against all requirements, 3) create new options based on evaluation, and 4) select best option as a solution when all negotiation rounds have been executed.

First, *phase 1* of the negotiation process is started by the negotiator that creates a random set of options *optionSet* that are open for negotiation, see line 2 of Pseudocode 1. For example, the negotiator creates a number of different random light plans. The size of the option set is determined by population size in the genetic algorithm that is found by experimental fine-tuning.

Pseudocode 1. Negotiation process

```
1:  {Phase 1}
2:  $optionSet = createInitialOptions(system)$
3:  for all $round = 0 \rightarrow maxRounds$ do
4:     {Phase 2}
5:     for all $option \in optionSet$ do
6:        $option.satisfySum = 0$
7:        $option.accepted = 0$
8:        for all $agent \in agentSet$ do
9:           if $agent.accept(option)$ then
10:             $option.satisfySum = option.satisfySum + agent.satisfy(option)$
11:          else
12:             $option.accepted = 1$
13:          end if
14:       end for
15:       $option.fitness = option.accepted + option.satisfySum/size(agentSet)$
16:    end for
17:    {Phase 3}
18:    $sort(optionSet)$
19:    $optionSet = createNewOptions(optionSet)$
20: end for
21: {Phase 4}
22: $solution = selectBestSolution(optionSet)$
23: if $\neg solution.accepted$ then
24:    $sendAlert(solution)$
25: end if
```

Phase 2 starts the first negotiation round given the initial option set as a start condition. The negotiator asks each participating agent if it can accept the option from the option set. Each agent evaluates the proposed option based on its beliefs and goals and responds back with an accept message if it can accept. Formally, each requirement is represented by an agent. For example, the requirement $r_{DarkHour}$ is represented by the agent $a_{DarkHour}$ with its goal to ensure that light is turned off for a specified number of hours. The goal $goal_{DarkHour}$ is expressed as an inequality goal constraint over the agent's dark-hour plan and

Pseudocode 2. $a_{DarkHour}.accept(option)$

```
for all hour ∈ option.lightPlan do
    if option.fixedPlan[hour] ≠ option.lightPlan[hour] then
        return false
    else
        return true
    end if
end for
```

Pseudocode 3. $agent_{MinLightPrice}.satisfy(option)$

```
for all hour ∈ option.lightPlan do
    if isOn(hour, option.lightPlan) then
        $consumedEnergy = option.totalLampLoad \times seconds(hour)$;
        $energyCost = option.priceForecast[hour] \times consumedEnergy$;
        $energyCostSum = energyCostSum + energyCost$;
    end if
end for
return $objectiveCriteria(energyCostSum)$;
```

the proposed light plan from the negotiator. The intention of agent $a_{DarkHour}$ is to respond with an accept message if the goal constraint is not broken for all hours in the light plan. If an agent accepts an option, the negotiator asks the agent for a satisfy message. The satisfiability value returned by the agent expresses how well the proposed options satisfy the agent's goals. For example, the requirement $r_{MinLightPrice}$ is represented by agent $a_{MinLightPrice}$ with the goal $g_{MinLightPrice}$. The goal $g_{MinLightPrice}$ is to minimize the cost of the proposed light plan as much as possible, see Pseudocode 3. The intention of the agent is to accept a proposed light plan and to send a satisfy message that expresses how well the cost of the light plan was minimized. The two different agents represent two different kinds of requirements. Agent $a_{DarkHour}$ represents a requirement that is fully satisfied if the agent's goal constraint is not broken by the proposed option. Oppositely, agent $a_{MinLightPrice}$ represents an optimization requirement to the proposed option. For example, using accept and satisfy messages we can represent constraint-based requirements that are expressed as goal constraints and optimization requirements that are expressed as an utility value. Based on the received satisfy messages, the negotiator calculates a satisfy sum $options.satisfySum$ for the option, see line 10 of Pseudocode 1. If the agent responds with a reject message, the negotiator marks the options as being not accepted (value 1), see line 12 of Pseudocode 1. When all agents have responded on the proposed options, the negotiator marks the option with a fitness value calculated based on option acceptance and the option's satisfy sum, see line 15 of Pseudocode 1. The lower the fitness value is, the better the option is. The marking of options continues until all options in the option set *optionSet* have been marked with a fitness value.

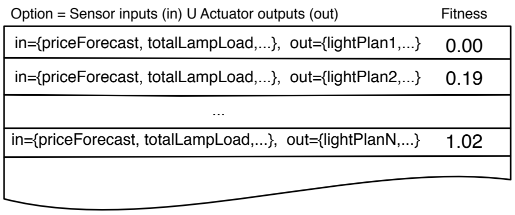

Option = Sensor inputs (in) U Actuator outputs (out)
Fitness
in={priceForecast, totalLampLoad,...}, out={lightPlan1,...} 0.00
in={priceForecast, totalLampLoad,...}, out={lightPlan2,...} 0.19
...
in={priceForecast, totalLampLoad,...}, out={lightPlanN,...} 1.02

Fig. 2. An option set sorted according to option fitness

In *phase 3*, the set of options are now sorted according to their fitness. The result is an ordered set of options. Options that are accepted by most agents and have the best average satisfiability appear first in the set, see Figure 2. The negotiator examines the ordered option set and replaces the worst half of the set with new options. The new options are generated by either mutation or crossover of options, selected randomly across the entire option set. The $isMutation()$ is a function that randomly returns true or false for whether or not an option i should be mutated or replaced by a crossover result, see line 2 of Pseudocode 4. Each *mutation operator* is domain-specific and allows the developer to specify the allowed variability of the set-point values. For example, the mutation function for a light plan is specified in Pseudocode 5. Domain-specific mutation functions improve the performance of the genetic algorithm as they make the search more directed and avoid mutating options into options that are unrealistic; for example, mutating a temperature set point into a value that is outside the allowed range of temperatures. A domain-specific mutation function for a temperature avoids that problem by specifying mutation within a specific temperature range. The *crossover operator* is generic and is randomly applied to half of the outputs for the selected option to be crossed, see Pseudocode 6. The random selection together with elimination of the worst individuals reduce the chance of ending up with a suboptimal solution since all suitable individuals have a chance to mutate and reproduce. The updated options become the new input for the next negotiation round (generation). The negotiation process continues for a fixed number of negotiation rounds.

Pseudocode 4. $createNewOptions(optionSet)$, see line 19 of Pseudocode 1

```
Require: optionSet exists
Ensure: new optionSet
1: for $i = size(optionSet)/2 \rightarrow size(optionSet)$ do
2:    if $isMutation()$ then
3:       $optionSet[i] = mutate(random(optionSet))$
4:    else
5:       $optionSet[i] = crossover(random(optionSet), random(optionSet))$
6:    end if
7: end for
```

Pseudocode 5. *mutate*(*option*), see line 3 of Pseudocode 4

```
for $i = 0 \rightarrow size(option.lightPlan)/4$ do
    $idx = randomIndex(option.lightPlan)$
    $option.lightPlan[idx] = \neg option.lightPlan[idx]$
end for
return $option.lightPlan$;
```

Pseudocode 6. *crossover*(*optionA*, *optionB*), see line 5 of Pseudocode 4

```
for $i = 0 \rightarrow size(optionA.out)/2$ do
    $j = removeRandom(optionA.out)$
    $optionA.out[j] = optionB.out[j]$
end for
return $optionA$
```

The end of the negotiation rounds marks the start of phase 4, where the solution of the negotiation rounds is selected as the option with the best fitness. If the selected solution is marked as not accepted, it means that not all agents accepted the solution and that there are conflicts between the goals of the agents. The negotiator generates an informed alert message that explains which agents are in conflict with each other and which sensor input and actuator set points they share. The alert message is sent to the user of the system that can then make an informed decision to relax one or more of the goals of the conflicting agents. The relaxation of goals will be taken into consideration in the next control cycle when a new negotiation process is triggered. Finally, the set points from the selected solution are sent to the PLC and effectuated by the subsystems' actuators.

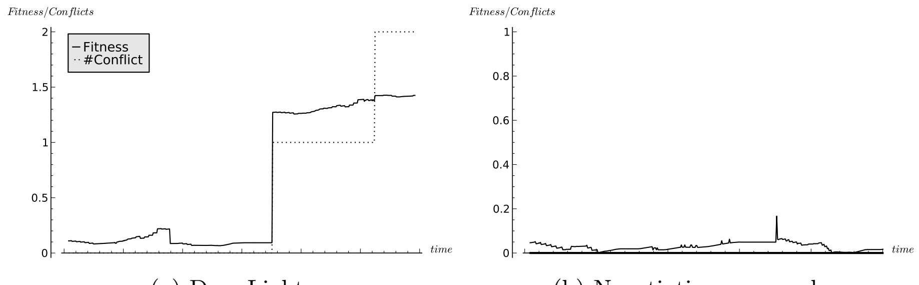
6 Experimental Validation

The evaluation of our solution is based on real data from an experiment conducted in the period from 12th October to 9th November 2009 with a total of 300 cuttings of *Chrysanthemum Morifolium*. The motivation of the experiment was to identify how irregular light periods affect the plant growth of *Chrysanthemum Morifolium*. The results confirm that climate-control strategies based on hourly changes in electricity prices can have an economical value for the production of pot plants [10]. The light strategy to generate irregular light periods was provided by a system (DynaLight earlier also known as Climate Monitor) that implemented the requirements $r_{GrowthGoal}$, $r_{MinLightPrice}$, $r_{LightHour}$, $r_{DarkHour}$ and $r_{OptimalPhotosynthesis}$ described in Section 3.

DynaLight represents a centralized system with all requirements implemented in one specification without any negotiation mechanism to coordinate and explain possible conflicts between the requirements [15]. Contrary, our approach represents a system with all requirements implemented as independent separate agents with a negotiation mechanism to manage and explain conflicts between the agents. Since the requirements are similar in the two systems, they can be represented by the same set of agents and it is possible to compare the fitness of the solutions and the number of conflicts generated. For that reason, the data generated by DynaLight is ideal for an experimental evaluation of our multi-agent-based control system. The purpose of the experimental evaluation is to compare the fitness of the solutions from DynaLight with the fitness of the solutions from our approach. Additionally, the experiment validates the ability of our approach to manage and explain conflicts compared to DynaLight. The experimental setting includes two experiments. The first experiment evaluates the fitness and the number of conflicts of the solutions found by DynaLight. That is, the agents only represent the requirements from DynaLight and are used to evaluate fitness and conflicts of the set-point values that were generated by DynaLight. The second experiment evaluates the fitness and number of conflicts for the same set of agents but using sensor inputs from the earlier experiment. That is, the set points are determined by the negotiation process between the agents and the negotiator. Both experiments share the same configuration from the experiment (60% NB) described in [10]. In summary, the photosynthesis optimization is 60%, the lamp intensity is 60 $\mu mol/m^2s$, the photosynthesis growth goal is 268 $mmol/m^2s$[1], the dark-hour period is 5-8pm and the control-cycle interval is 10 minutes. The time zone for the experiment is GMT+1 at wintertime. Furthermore, the total connected lamp load is 7 KW, i.e., the installed lamp effect per square meter is 70 W and the size of the greenhouse is 100 m^2. In the second experiment, the size of the option set is configured to be 500 and the number of negotiation rounds were set to 1000.

Figure 3a illustrates the fitness and the number of conflicts for the DynaLight experiment. The first conflicts start emerging midnight October 21 because artificial light is proposed to be turned on at 8 pm causing violation of the requirement $r_{DarkHour}$ because the hour at 8 pm is specified as a dark hour. The requirement $r_{DarkHour}$ persists to be compromised from midnight until the end of the experiment due to the lack of a mechanism for handling the conflict automatically. One explanation for the conflict is that the climate computer, which DynaLight interfaces, was reconfigured by accident. The second conflict occur early morning October 22 as a consequence of a proposed light plan that does not fulfil the specified photosynthesis growth goal. The proposed light plan causing the second conflict, has a total photosynthesis contribution of 110 $mmol/m^2s$. The achieved photosynthesis contribution from natural light and the light plan, at the time were the conflict emerges, is 11 $mmol/m^2s$. The estimated photosynthesis contribution from natural light for the rest of the day is 142 $mmol/m^2s$. Summed together, the proposed light plan, the achieved photosynthesis contribution from natural and artificial light, and the expected photosynthesis contribution from natural light for the rest of the day is not enough to achieve the target of a photosynthesis growth goal of 268 $mmol/m^2s$. That is, the photosynthesis sum balance calculated is $(110 + 11 + 142) - 268 = -6$ $mmol/m^2s$ which means that

[1] Photosynthesis is given as flux of Photosynthetically active radiation photons per unit area (PPFD).



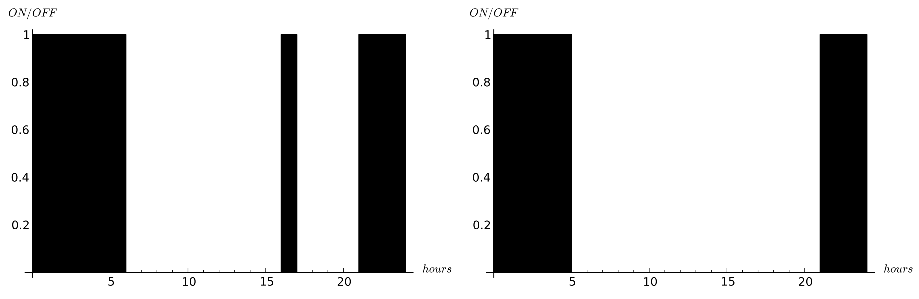
(a) DynaLight. (b) Negotiation approach.

Fig. 3. Fitness and conflicts of solutions found with and without a negotiator over a period (19-22 Oct 2009)

the agent presenting requirement $r_{GrowthGoal}$ lacked 6 $mmol/m^2s$ to reach its growth goal. Figure 3b depicts the result of our negotiation mechanism where no conflicts emerge as the negotiator finds solutions that satisfy the requirements.

The average fitness of the result from the DynaLight experiment, in the period till the first conflict occurrence, is 0.1044 and the average fitness for the entire period is 0.4488. In comparison, the fitness of the negotiated result for the same period before any conflicts is 0.0298 and the average fitness for the entire period is 0.0295. To explain the relationship between fitness and the solutions found it is relevant to investigate light plans from the two experiments before the occurrence of any conflicts. Figure 4a and 4b depict the light plans effectuated October 19 by DynaLight and our approach. The cost of the DynaLight light plan (Figure 4a) is calculated to €2.44 based on the forecasted electricity prices and the connected lamp load. Moreover, the photosynthesis contribution from the DynaLight light plan is 115 $mmol/m^2s$. Conversely, the cost of our negotiated light plan for the same period is €1.94 and the photosynthesis contribution is 92 $mmol/m^2s$. In summary, that means there is a saving of €0.50 for the day for the negotiated solution but that the photosynthesis contribution from the negotiated light plan is 23 $mmol/m^2s$ less than is gained from the DynaLight light plan.

The poor fitness of the light plan from DynaLight can be explained by understanding the way DynaLight plans ahead of time. DynaLight generates its light plan based on a daily analysis of the forecasted natural light. If the forecasted natural light is pessimistically estimated at the time DynaLight generates its light plan, the result will be a too large photosynthesis contribution from the light plan that together with the actual achieved photosynthesis contribution from natural light will exceed the specified growth goal. Exceeding the growth goal leads to a poor fitness of requirement $r_{GrowthGoal}$. Additionally, a pessimistic forecast leads to a higher cost (bad fitness of requirement $r_{MinLightPrice}$) as the photosynthesis growth goal can be achieved with less artificial light because of the higher photosynthesis contribution from the actual natural light. In contrast, our approach negotiates the light plan for each control cycle and continuously adapts the light plan according to the actual achieved photosynthesis contribution from natural light. The result is a dynamic light plan that gets very



(a) DynaLight. (b) Negotiation approach.

Fig. 4. Light plans for 24-hours found October 19 2009 with and without a negotiator

close to the specified growth goal, resulting in a better fitness of both requirement $r_{GrowthGoal}$ and $r_{MinLightPrice}$. In fact the 92 $mmol/m^2s$ is just enough to achieve the growth goal given the forecasted natural light. Last, the light plans for 19th October look very similar which could be expected considering both approaches represent the same set of requirements.

7 Related Work

Our work is inspired by [7,8,16,17] which differ from other prevalent approaches by perceiving the feature interaction problem as a resource-sharing problem rather than a feature-behavior problem. The idea behind the approaches is based on the assumption that feature interactions emerge as a consequence of features sharing resources. The argument for focusing on resources instead of feature behaviors is that resources are simpler to model and understand than feature behaviors. To manage feature interactions, the approach requires a specification based solely on knowledge about the resources.

Bisbal and Cheng contribute with a resource-oriented approach to detect and handle feature interactions in component-based software at runtime [7]. Their resource-aware specification declares the resource goals of a component and the relationship between the component and its resources. The specification is declared at design time and used by runtime techniques to address feature interactions in resource-aware systems.

Liu and Meier contribute with resource-aware contracts and address resource-based feature interactions in dynamic adaptable systems [17]. The resource specification proposed by Bisbal and Cheng is based only on fixed-capacity and varying-capacity resources and can only be applied at component level. In contrast, resource-aware contracts support both fixed-capacity, varying-capacity, exclusive and shared resources and can be applied at both component and component-assembly level. In addition, resource-aware contracts also specify the global resource constraints at system level.

Zambrano et al. focus on aspect interactions and use metadata annotations to specify the resource requirements of each aspect [16]. Zambrano et al. provide a detection and a resolution strategy that can detect and avoid feature interaction in resource-aware systems, without compromising obliviousness of the aspects. The resource specification is declared as metadata on the aspects at design time in the form of semantic annotations. The interactions between aspects are avoided at runtime by a coordinator aspect. The coordinator aspect is augmented with a list of user-defined conflict situations declared at design time as conflict rules. A conflict rule expresses the resource conditions that should be avoided by corresponding corrective actions. Affected resources are asserted against a coordinator rule engine to detect conflict conditions. To avoid interactions, the coordinator aspect can deactivate the conflicting aspects as declared in the action part of the triggered conflict rule.

The approaches suggested by Bisbal, Cheng, Liu, Meier and Zambrano et al. are based on formal specifications/analysis of the resources shared between the features. The analysis of the resources can be accomplished at design time and applied to runtime approaches to enhance detection and resolution of feature interactions. Feature interactions are perceived as violations of the feature requirements as a consequence of wrong assumptions about the shared resources. The resource specifications are described independently of the features and support development of features by third-party vendors. In case no solutions can be found to resolve the feature interaction, the mentioned approaches do not support that the user can redefine the requirements of the conflicting features to find alternative solutions. Redefinition of conflicting feature requirements to resolve interactions requires explanation of what caused the feature interaction. To our knowledge none of the approaches support such detailed *explanation* of the cause of feature interactions.

8 Future Work

Explanation of runtime feature interactions requires that the agents are kept separate and represent requirements as perceived by the user of the system. That is, if the requirements are not understood by the user then it is difficult to explain the conflicts as they are expressed in terms of the requirements. For that reason, future work should incorporate a description language that can support the creation of more user friendly alert messages. The explanation of feature interactions should be improved by supporting visual views that explain the interactions as a resource sharing problem among agents. One suggestion for a visual view, would be to visualize the shared outputs over time using color scales corresponding to the number of conflicting agents. For example, if one agent can't accept the negotiated solution because the output is negatively influenced by other agents, then that shared output should be colored red. The conflicting resource should be colored more dark red when the number of conflicting agents increases. Additionally, the color-view of the resources can be combined with a color-view of the agents showing which agents are causing the resource conflicts. Conflicting agents should be colored red if they didn't accept any of the proposed options, otherwise the agents should be colored according to their satisfaction degree. Additionally, the overall performance of the system should be visualized using a colored graph-view illustrating the satisfaction degree of the found solutions compared to the satisfaction of each of the agents. Finally, the approach needs to be more thoroughly tested. It is obvious to compare the approach with our earlier proposed counter-proposal negotiation approach to evaluate how well the approach performs [18]. An other aspect that should be evaluated, is how well the approach supports explanation compared to other negotiation approaches.

9 Conclusion

In this paper, we presented a novel multi-agent-based approach to control system engineering that frees developers from identifying and resolving interactions among control strategies before they can be deployed as part of the same system. Our approach achieves this by implementing each control requirement as a separate agent which then participates in the negotiation of acceptable values for the control set points. Through empirical evaluation in an ornamental floriculture research facility, we have shown that it is realistic to implement independent climate control requirements as individual agents, thereby opening greenhouse climate control systems for integration of independently produced control strategies.

References

1. Aaslyng, J., Lund, J., Ehler, N., Rosenqvist, E.: IntelliGrow: a greenhouse component-based climate control system. Environmental Modelling & Software 18(7), 657–666 (2003)
2. Markvart, J., Kalita, S., Jørgensen, B.N., Aaslyng, J.M., Ottosen, C.O.: IntelliGrow 2.0 - a greenhouse component-based climate control system. In: Proceedings of the International Symposium on High Technology for Greenhouse System Management (2007)
3. Körner, O., Challa, H.: Temperature integration and process-based humidity control in chrysanthemum. Computers and Electronics in Agriculture 43, 1–21 (2004)
4. Körner, O., Andreassen, A.U., Aaslyng, J.M.: Simulating dynamic control of supplementary lighting. Acta Horticulturae 711, 151–156 (2006)
5. Körner, O., Aaslyng, J.M., Andreassen, A.U., Holst, N.: Microclimate prediction for dynamic greenhouse climate control. HortScience 42(2), 272–279 (2007)
6. Armstrong, N., Robin, L., Bashar, N.: Feature Interaction as a Context Sharing Problem. In: Feature Interactions in Software and Communication Systems X (2009)
7. Bisbal, J., Cheng, B.H.C.: Resource-based approach to feature interaction in adaptive software. In: WOSS 2004: Proceedings of the 1st ACM SIGSOFT Workshop on Self-Managed Systems, pp. 23–27. ACM, New York (2004)
8. Metzger, A.: Feature interactions in embedded control systems. Computer Networks 45(5), 625–644 (2004)

9. Calder, M., Kolberg, M., Magill, E.H., Marganiec, S.R.: Feature interaction: a critical review and considered forecast. Comput. Netw. 41(1), 115–141 (2003)
10. Kjaer, K.H., Ottosen, C.O.: Growth of Chrysanthemum in Response to Supplemental Light Provided by IrregularLight Breaks during the Night. Journal of the American Society for Horticultural Science 136, 3–9 (2011)
11. Szyperski, C.: Independently Extensible Systems - Software Engineering Potential and Challenges. In: Proceedings of the 19th Australasian Computer Science Conference (1996)
12. Goodrich, M., Stirling, W., Frost, R.: A satisficing approach to intelligent control of nonlinear systems. In: 1996 IEEE International Symposium on Intelligent Control, pp. 248–252. IEEE (September 1996)
13. Simon, H.A.: A Behavioral Model of Rational Choice. The Quarterly Journal of Economics 69(1), 99–118 (1955)
14. Simon, H.: Optimal problem-solving search: All-or-none solutions. Artificial Intelligence 6(3), 235–247 (1975)
15. Mærsk-Møller, H.M., Jørgensen, B.N.: A Software Product Line for Energy-Efficient Control of Supplementary Lighting in Greenhouses. In: The International Conference on Green Computing (2011)
16. Zambrano, A., Vera, T., Gordillo, S.E.: Solving Aspectual Semantic Conflicts in Resource Aware Systems. In: RAM-SE, pp. 79–88 (2006)
17. Liu, Y., Meier, R.: Resource-Aware Contracts for Addressing Feature Interaction in Dynamic Adaptive Systems. In: 2009 Fifth International Conference on Autonomic and Autonomous Systems. IEEE, Los Alamitos (2009)
18. Sørensen, J.C., Jørgensen, B.N.: Counter-proposal: A Multi-Agent Negotiation Protocol for Resolving Resource Contention in Open Control Systems. In: The 9th International Conference on Autonomous Agents and Multiagent Systems - Workshop 1 Agent Communication (2009)

ACTraversal: Ranking Crowdsourced Commonsense Assertions and Certifications

Tao-Hsuan Chang, Yen-Ling Kuo, and Jane Yung-jen Hsu

Department of Computer Science and Information Engineering
National Taiwan University
{doudi.tw,a33kuo}@gmail.com, yjhsu@csie.ntu.edu.tw

Abstract. Building commonsense knowledge bases is a challenging undertaking. While we have witnessed the successful collection of large amounts of commonsense knowledge by either automatic text mining or *games with a purpose* (GWAP), such data are of limited precision. Verifying data is typically done with repetition, which works better for very large data sets. Our research proposes a novel approach to data verification by coupling multiple data collection methods. This paper presents *ACTraversal*, a graph traversal algorithm for ranking data collected from GWAP and text mining. Experiments on aggregating data from two GWAPs, i.e. Virtual Pets and Top10, with two text mining tools, i.e. SEAL and Google Distance, showed significant improvements.

1 Introduction

Codifying several million pieces of commonsense knowledge into machine usable forms has proved to be time-consuming and expensive. In 1984, a team of knowledge engineers [6] started to craft the Cyc knowledge base using CycL, a logic-based language. Extreme care was taken to ensure its correctness. In contrast, Open Mind Common Sense (OMCS) [8] took the Web 2.0 approach by collecting voluntary contributions of commonsense sentences from online users. In recent years, with the advances in human computation, large amounts of data can be crowdsourced or mined from the web efficiently. Despite improvements in the efficiency of knowledge acquisition, those methods are limited in their precision.

Several methods have been developed to guarantee the precision of crowdsouced knowledge. Frequency is the most used approach to filtering out noisy data in crowdsourcing, which takes advantage of the huge amount of user input for data verification. Mechanism design is another approach to eliciting correct answers from players of GWAPs. For example, Verbosity [1] uses sequential verification to verify answers, and Virtual Pets [5] improves the quality of answers by punishinig malicious behaviours in community interaction.

Unfortunately, these mechanisms may not work in practice. The data collected are subject to pollution due to tricks by players. Speer et al. [9] found that half of the data collected by Verbosity should be rejected by OMCS since they are only sound-alike or look-alike clues used for guessing the answers. In their experiment, half of the data were filtered, yet the remaining data were still had lower quality than the assertions in OMCS. In a single GWAP, players may learn tricks from experiences, using the tricks to get game points but contribute little to the knowledge base. Even the mechanism is well designed, the quantity and quality may be limited due to those tricks.

In order to improve the precision without modifying the original GWAP, one can develop a separate game for data verification [3]. Meanwhile, text mining methods has been proved it is useful to learning sentences from the web[2]. By coupling GWAP or text mining methods, the precision may also be improved. Nevertheless, the existing text mining depends on properly defined seeds as training examples and templates, while GWAP tend to produce redundant data over time. As a result, aggregating both GWAP and text mining are better than coupling only one type of methods for building large commonsense knowledge bases.

This paper proposes an approach, *ACTraversal*, to improving the precision of commonsense knowledge collected by aggregating GWAP with text mining components. The proposed *ACTraversal* is a universal graph traversal aggregation for ranking common sense assertions and certifications.The assertions and certifications of commonsense knowledge are defined as below:

- Assertions: Sentences composited by subject-relation-object triples.
- Certifications: Evidences indicating partial-order of associated assertions confidence level. Assertion with higher confidence level is associated with higher order certification. Every assertion is associated with at least one certification.

In the following sections, we first compare the pros and cons of existing verification methods of GWAP and text mining techniques, and introduce the proposed ACTraversal. We then present our prototype implementation, followed by the experiments designed to show that

- ACTraversal can improve quality of the partial order ranking produced by a single component,
- ACTraversal can aggregate data from multiple components, and
- ACTraversal is efficient on large dataset.

2 Approaches to Knowledge Verification

This section presents an overview of approaches to knowledge verification and discusses their pros and cons to show the advantages in coupling the approaches.

Games with a Purpose. Games With A Purpose (GWAP) [10] utilizes computer games to gather players and guides them to perform tasks. The quality of collected sentences is guaranteed via the mechanism of the game. It has demonstrated its efficiency in commonsense knowledge collection. For example, Verbosity [1] and Virtual Pets [5] have collected over a million English and Chinese commonsense sentences respectively within a year with acceptable precision.

Verbosity is a two-player game in which the Narrator gives clues to help the Guesser figure out a secret word. The clues are collected as commonsense knowledge about the corresponding secret word. Virtual Pets utilizes the interactions in communities to collect knowledge via question-answering between online users. Players in Virtual Pets answer questions such as "Spoon is used for ___?" in exchange for food for their pets. Both games use frequency as filters to verify the assertions. This approach succeeded in building useful knowledge bases within a relatively short time and with extremely low costs.

However, previous research found that the precision of answer is not perfect. The precision can be further improved by coupling with a verification game [3]. For example, Top10, a FamilyFeud-like verification game, can be used for verifying the concepts of an assertion collected by Virtual Pets. The precision of assertions collected by Top10 and Virtual Pets is significantly higher than the assertions collected by only a single game. This approach shows the opportunity to aggregate GWAP to gain precision.

Text Mining. Some types of general knowledge, such as categorical relations, can be extracted automatically [7,4,2] from a corpus. The data quality is ensured via text mining or machine learning techniques. Previous work has demonstrated that pattern-based and list-based extraction methods can be combined to achieve significant improvements in recall [4]. Never-Ending Language Learner (NELL) [2] also showed that it is possible to collect sentences with high precision and improve the learning process itself by using coupled machine learning components. This approach shows the opportunity to couple text mining to gain precision.

However, these coupled methods are limited to their constraints. While text mining is restricted on certain knowledge domains, GWAP produces fewer assertions. By coupling GWAP and text mining methods, the data can be further verified. In addition, aggregation of the non-overlapping data can also improve the coverage of knowledge base. With larger coverage, the aggregation approach can verify more assertions.

3 ACTraversal

To get the best of both worlds, we propose ACTraversal (ACT); a universal ranking aggregation by graph traversal on assertions and certifications. By utilizing partial-order of certifications, ACT ranks assertions from multiple components, constructing a total-order of all assertions. This section introduces the data structure, graph, assumptions, and the algorithms used in ACTraversal.

3.1 Data Structure

The proposed ACT uses assertions and their certifications from multiple components.

Assertions. Commonsense knowledge is comprised of assertions, which are defined as "subject-relation-object" triples in this paper. For example, the corresponding assertion for "dog is an animal" is "dog-`IsA`-animal". Both subject and object of an assertion are concepts, which are represented in plain text. For the sake of simplicity, we assume that all the texts have only one sense since the concepts in the same text but different senses can be stored as different instance in this system. The relation of an assertion comes from a predefined relation set which can be incrementally enlarged. Also, the relations are directed edges connecting two concepts. For example, "Subject-`IsA`-Object" and "Object-`IsA`-Subject" are regard as different assertions. If every corresponding element in two triples are the same, the system treats them as the same assertions.

Certifications. Assertion with higher confidence level is associated with higher order certification. Every assertion is associated with at least one certification. Certifications are the evidences indicating partial-order of associated assertions confidence level. Each assertion may associate with multiple certifications from different sources (i.e. GWAP and text mining components.) For example, both "2 players answered dog-`IsA`-animal in Verbosity" and "NELL learned dog-`IsA`-animal with 100.0% confidence" are valid certifications for "dog-`IsA`-animal". Since each component returns assertions with their certifications, every assertion must associate with at least one certification. In addition, an assertion can also associate with multiple certifications from a single source. For example, "UserA vote for dog-`IsA`-animal on OMCS website" and "UserB vote for dog-`IsA`-animal on OMCS website" are all valid certifications. Multiple certifications in a single component can be aggregated into a single certification. For example, the above certification can be aggregated as "2 users vote for dog-`IsA`-animal on OMCS website'. By doing so, it reduces the computational complexity and hide personal information.

AC Graph. The graph of assertions and certifications (i.e. the AC graph) in ACT has two types of nodes and two types of edges. The nodes are *Assertion* and *Certification*. Each distinct assertion and certification corresponds to an *Assertion* node and *Certification* node, respectively. The edge type between *Assertion* and *Certification* is *Cross*. The edge type between *Certifications* is *Order*. The *Cross* edge is bidirectional, and the *Order* edge is unidirectional. There is no edge between *Assertion* nodes. Each node has a ranking score, and each edge has a weight. Weight of *Cross* edge is the confidence score of its source component. For component with higher confidence score, its associated certifications are more trustworthy. Weight of *Order* edge is set to a uniform score since it only indicates the direction of traversal. Figure 1 shows a sample graph containing 3 assertions and 3 certifications from 2 components.

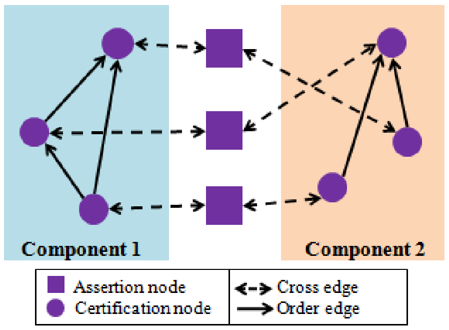


Fig. 1. A sample AC graph

Assumptions. Two assumptions are made in using the assertions and certifications:

– The ranking scores of assertions are positively correlated to ranking score of associated certifications.

– The certifications of a single component are in a partial-order indicating the confidence level of associated assertions.

The first assumption means a good assertion has a strong certification, and the strength of certification depends on the assertion it attests to. The second assumption implies that the confidence scores of assertions are comparable through certifications. For example, “2 users vote for dog-`IsA`-animal on OMCS website’ has higher confidence score than “1 users vote for dog-`IsA`-pet on OMCS website’. However, “2 users vote for and 1 user votes against bird-`CapableOf`-fly on OMCS website’ is not comparable with “1 users vote for bird-`IsA`-creature on OMCS website.’ These assumptions hold for components that use frequency as filter to verify data. Therefore, GWAP that use frequency of answers/votes and text mining components that output assertions with probabilities/rankings can be formalized as components of ACTraversal.

3.2 ACTraversal Algorithm

ACTraversal (ACT) is a graph traversal algorithm to rank assertions and certifications on a weighted directed graph. It takes the output assertions and certification of GWAP or text mining components as input. The confidence score of each component is predefined according to the result of previous study. The output of ACT is a ranking list of assertions and certifications with the converged scores.

AC Graph Construction. ACT builds the graph according to the input pairs of assertions and certifications. For each pair, the assertion and certification are connected by *Cross* edges. After reading all the pairs from a component, ACT compares the certifications and adds a *Order* edge from lower order (weak certification) to higher order (strong certification). For example, if every element of a 2-ary certification is greater than another 2-ary certification, its order must be higher. So, ACT adds an edge from (2,1) to (2,3), but it does not build edge between (2,1) and (1,2). The time complexity of building AC graph for a single component is $O(|Cross| + |Certification|^2)$.

Algorithm 1. $ACGraph(Components)$

```
Require: $Components$ with output pairs of assertion and certification
    $\{ACG$ is a weighted directed graph$\}$
 1: for all $C \in Components$ do
 2:     for all $assert, cert \in C.ACpairs()$ do
 3:         $ACG.addNode(assert, cert)$
 4:         $ACG.addBidirectEdge(assert, cert, C.confScore)$
 5:     end for
 6:     for all $cert1, cert2 \in ACG.Certs$,
        s.t. $cert1 < cert2$ do
 7:         $ACG.addUnidirectEdge(cert1, cert2)$
 8:     end for
 9: end for
10: return $ACG$
```

Once constructing the AC graph, ACT assigns the weights to nodes and edges. All nodes are assigned weight evenly; with sum of weight of all nodes is equal to 1. The weight of edges are assigned as defined in AC graph. Algorithm 1 describes the details of building AC graph.

AC Graph Traversal. The traversal algorithm proceed iteratively until the ranking score of node converges. In each iteration, scores of nodes are updated while the weight of edges remains the same. For each *Assertion* node, ACT assigns the weighted average of the linked *Certification* nodes to it as the new score. The weight used here is weight of *Cross* edge. For each *Certification* node, ACT collects two sources of scores, and then assign the node by averaging scores from the two sources. One source is the score of its associated *Assertion* nodes. ACT computes the weighted average scores of *Assertion* nodes as the first source. The other source is the score of lower order *Certification* nodes. ACT passes score of lower order *Certification* evenly to its linked higher order neighbors through the *Order* edges. After updates of the scores, ACT redistributes a small portion of scores to every node as random restart. In the end of each iteration, ACT normalizes the scores to make their sum equal to 1. Algorithm 2 demonstrates the details of ACTraveral. The time complexity of this algorithm is $O(iteration \times (N+E))$.

Since the *Assertion* and *Certification* are positively correlated, they can be viewed as the attributes of each other. Also, the partial order of *Certification* nodes are used for estimating authority. The ACTraversal can be considered as a process to 1) average the attributes and 2) compute the authority. With the two steps, we can estimate the rank of *Assertion* and *Certification* at the same time. Figure 2 shows the interaction result of *Certification* and *Assertion* in ACT.

Algorithm 2. $ACTraversal(Components, \lambda)$

```
Require: $Components$ with output pairs of assertion and certification
    {$ACG$ is a weighted directed graph}
1:  $ACG \leftarrow ACGraph(Components)$
2:  repeat
3:      for all $Node \in ACG$ do
4:          if $Node$ is $Assertion$ then
5:              $score \leftarrow weightedAvg(neighborCert(Node))$
6:          else if $Node$ is $Certification$ then
7:              $score \leftarrow weightedAvg(neighborAssert(Node))$
8:              for all $cert \in predecessorCert(Node)$ do
9:                  $score \leftarrow score + (\frac{cert.score}{cert.outDegree})$
10:             end for
11:             $score \leftarrow score/2$
12:         end if
13:         $Node.score \leftarrow score$
14:     end for
15:     $ACG.RandomRestart(\lambda)$
16:     $ACG.NormalizeScores()$
17: until score is converged
18: return $ACG.rankingScore$
```

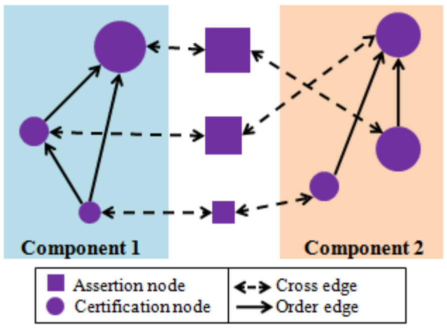


Fig. 2. ACT on sample AC graph. The size is the result weight.

4 Implementation

We evaluated ACT in terms of its quality of aggregation results and efficiency. In the following section, we will first describe the components in our implementation. Second, we will show that ACT can rank assertions in components with partial order certifications to achieve higher precision. Then we will demonstrate the quality of ranked assertions after aggregating the data from multiple components. Finally, we will compare the running time of ACT in dataset of different sizes.

4.1 Components

In this experiment, we chose 2 GWAP and 2 AutoExtraction components that can output pairs of assertion and certification. The components are VirtualPets (VP) [5], Top10 [3], SEAL [11], GoogleCount (GC)[1]. The components are chosen based on previous studies[5,3,11]. The input, output, and the formalization of certification of each component are described in the following paragraphs.

Virtual Pets. Virtual Pets (VP) is a game in which players teach common sense to their own virtual pets. It outputs assertions with play logs, which are the frequency, good rank, and bad rank. We formalized these logs as certifications in 3-ary tuple, (+frequency, +good rank, -bad rank). The partial order of certifications is set if every element in one certification is greater than or equal to another.

Top10. Top10 is a family-feud-like game in which players guess the top 10 concepts of a given question to get high scores. The input of game is a list of assertions ranked by their frequency. It outputs assertions with play logs, which are the frequency, good votes, and bad votes. We formalized these logs as certification in 3-ary tuple, (+frequency, +good votes, -bad votes). The partial order of certifications is set if every element in one certification is greater than or equal to another.

[1] Google search engine, http://www.google.com

SEAL. Set Expander for Any Language(SEAL) is an algorithm that uses common wrappers of a given set of seeds to auto extract instances from the web. For example, given "tea" and "milk", both are liquid, as input seeds of SEAL, it can extract "water". It extracts the wrapper around seeds from web pages, and then use wrapper the extract new instances. A possible extracted wrapper of the example above is "drink X.</li>", where X is the extraction. It outputs the top 100 instances retrieved using a set of seeds belonging to a given category. We formalized the assertions as "instance-IsA-category" triples. Also, we defined the mutually exclusive relations between categories. If an instance belongs to two mutually exclusive categiries, we add a flag on the assertion. The certifications can then be formalized in 2-ary tuples, (-rank, -flag), which is in partial order relationship.

GoogleCount. GoogleCount is an algorithm that computes the number of web pages Google search engine returned for a given sentence. The input assertions are represented as natural language sentences. The output of GoogleCount is the assertions with page counts. We formalized the counts as certifications. One certification is in higher order only if its associated count is strictly greater than others.

In our experiment, we evaluated the precision of top 800 instances ranked by 1) each sigle component, 2) ACT on each single component, or 3) ACT on the four components, with λ equals to 0.01. We recruited 46 graduate students to label the top 800 assertions returned by each method as ground truth. Each label is contributed by at least 3 students. A label is true only if over half of annotators vote true.

4.2 Single Component Ranking

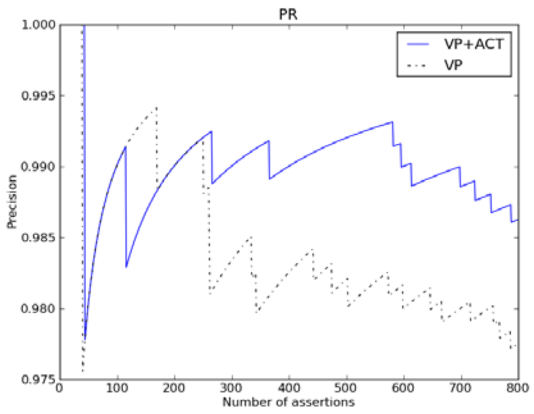
In the single component ranking, we compare ACTraversal with the serializing heuristic which sort results by frequency. This heuristic is previously used by the existing component, so we apply ACTraversal to examine the improvement. Table 1 lists the number of distinct assertions each component outputs. Figure 3 shows the precision of top 800 instances ranked by ACT on single component with single component precision as baseline. The improvement of precision in VP, Top10, and SEAL demonstrates the re-ranked lists produced by ACT are better than their original ranking lists. Therefore, ACT can be viewed as a ranking algorithm that turns partial order certifications into a better total order ranking of assertions. The result of GC is omitted in figure 3 because ACT does not produce new ranking list for a total order ranking.

Table 1. Number of assertions output by components

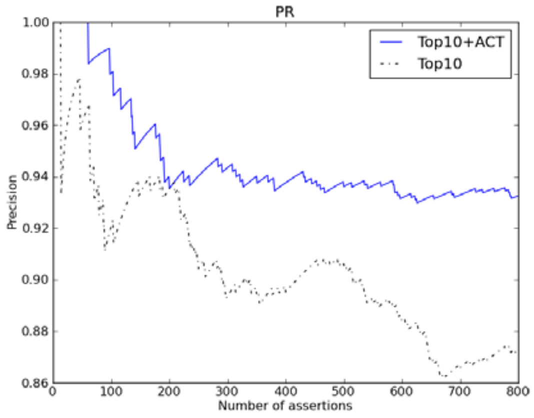
Component	VP	Top10	SEAL	GC
Number	378k	378k	2011	49989

4.3 Multiple Components Aggregation

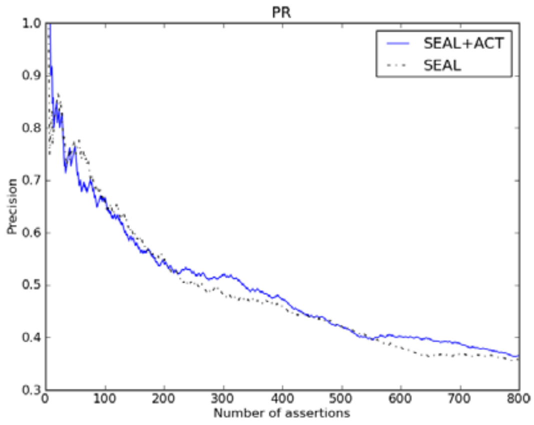
We first experimented on different confidence scores used by ACT to verify the impact of parameters. Then, we tested on dataset of different sizes to evaluate the efficiency of ACT.



(a) Precision of ACT on VP



(b) Precision of ACT on Top10



(c) Precision of ACT on SEAL

Fig. 3. Precision improvement by ranking single component with ACT

Confidence Score. We examined three possible settings of aggregation weighting parameters. The first setting uses uniform weight for all components. The other two settings are based on the precision of each component. The precision used for calculating the confidence score is the precision of top 200 assertions returned by ACT on each single component. One parameter setting uses the precision as the weight. The other parameter setting comes from the following formula, which is inverted logistic function:

$$confScore = -\ln(\frac{1}{precision} - 1) \quad (1)$$

The weights are then scaled to [0,1] by dividing the maximum weight. Table 2 lists the precision and weights used in this experiment. Figure 4 shows the precision of the three parameter settings for multiple components aggregation and ACT on each component. Compared to the uniform and precision weighted setting, inverted logistic weight has comparable high precision in aggregating both noisy and high quality components.

Table 2. Confidence Score in experiment

Component	VP	Top10	SEAL	GC
Uniform	1	1	1	1
Precision	0.99	0.935	0.525	0.745
Inverted Logistic	1.000	0.580	0.022	0.233

Table 3. Experiment on each size setting

Questions	Assertions	Certifications	Precision	Building Time	Traverse Time	Iterations
1250	138398	2007	0.94	34s	2m26s	56
2500	183628	2103	0.9525	1m2s	4m6s	58
5000	258997	2204	0.96375	1m54s	8m54s	62
10000	395612	2294	0.9725	3m10s	11m18s	64
20000	626077	2328	0.98125	6m39s	23m43s	64

Size of Dataset. We used different number of questions (i.e. the concept-relation pair used in VP and Top10) to generate dataset of different sizes. Table 3 lists the number of questions, number of the corresponding assertions returned by the four components, and the precision of top 800 assertions returned by ACT on dataset of different sizes. We observed that the precision was increased as the size of total assertions. Table 3 also gives the time in building graph and time in traversing graph of different sizes. Using linear regression to predict the execution time, the traversal time is $y = 4E - 05x - 2.8577, R^2 = 0.9739$ each iteration, and the graph construction time is $y = 0.0007x - 78.865, R^2 = 0.9894$. Both of them are nearly linear time.

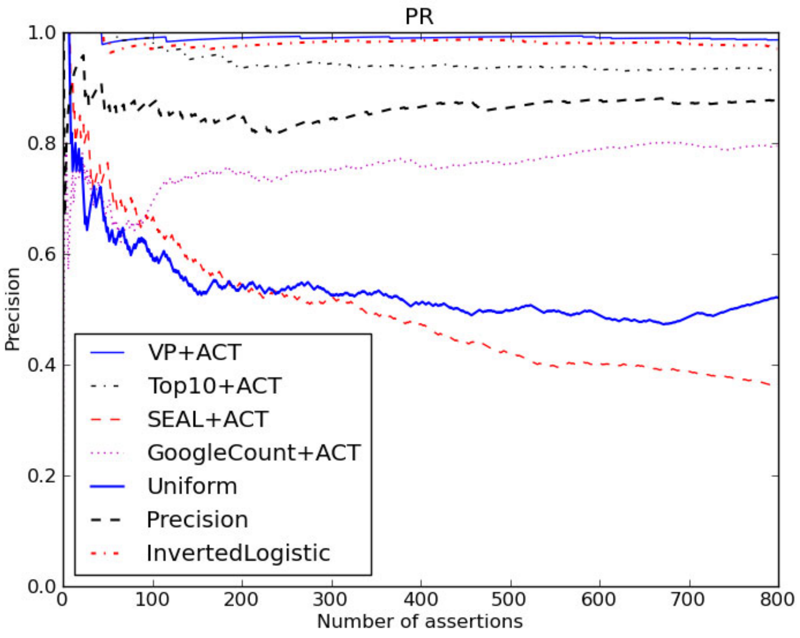


Fig. 4. Precision of ACT on each component and aggregation

5 Discussion

– *GWAP v.s. Text Mining* In previous study, text mining components are more productive in quantity[3,11]. While GWAP like Virtual Pets produces one thousand assertion-certification pairs a day, text mining components can produce the same quantity in few minutes. On the other hand, GWAP have higher precision than text mining components in our experiments. Meanwhile, text mining component is limited to certain domains, such as IsA relation, but GWAP can be used to collect variant assertions. From these observations, we know that the two types of components are good complement to each other. By aggregating the data from both components, we can take advantage of the two approaches.
– *Non-overlapping Data* The ACT can rank data from multiple sources without requirement of overlaps. ACTraversal ranks non-overlapping data by utilizing partial order from overlapping data. This property implies that we can infer a total ranking of confidence level of assertions. We believe the ranking result can be a guidance to bootstrap text mining component to collect more data. We can feed the assertions with high ranking score as the input of text mining components to collect more data. For the assertions with middle ranking scores, we can feed them as input of GWAP for verification. The assertions with low ranking score can then be filtered out. With this kind of coupling after aggregation of ACT, it is possible to get the data with higher quality and quantity.

– *Quantity of Assertions* In the experiment, we observed the quantity of assertions is positively correlated with the quality of top assertions. It implies that our two assumptions hold in our crowdsourced data. If good assertions are collected, they would be ranked in higher order. However, we also noticed the aggregation result is slightly less precise than the component with the best precision, i.e. ACT on VP. This slightly lower precision shows that ACT is not noise-proof. If stronger certification is not associated with better assertion, it would promote the bad assertion to the top. In our experiment, some noisy assertions were reported with high rank by GC component. Due to Google is a keyword based search engine, it cannot detect incorrect template filling. In this case, the count of Google retrieved data was high, but many of the results were not in the same meaning of our assertion. This errors can disturb the ranking order and bring up noises into results. Therefore, we suggest that components put in ACT should be verified to ensure that it 1) holds the assumptions and 2) has acceptable precision.

6 Conclusion

This paper presents the ACTraversal (ACT); a graph traversal algorithm to rank crowdsourced assertions and certifications. The partial order certifications are used for reconstructing the total order ranking of assertions. Our experiments of ACT on Virtual Pets, Top10, SEAL and GoogleCount shows that ACT successfully improve the precision of single component and construct total ranking of assertions from multiple components in nearly linear time.

References

1. von Ahn, L., Kedia, M., Blum, M.: Verbosity: A game for collecting common-sense knowledge. In: ACM Conference on Human Factors in Computing Systems (CHI Notes), pp. 75–78 (2006)
2. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka Jr., E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: Twenty-Fourth Conference on Artificial Intelligence, AAAI 2010 (2010)
3. Chang, T.h., Chan, C.w., Hsu, J.Y.j.: Human computation game for commonsense data verification. In: Proceedings of the 2010 AAAI Fall Symposium Series on Commonsense Knowledge (2010)
4. Etzioni, O., Cafarella, M., Downey, D., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Methods for domain-independent information extraction from the web: an experimental comparison. In: Proceedings of the 19th National Conference on Artifical Intelligence, AAAI 2004, pp. 391–398 (2004)
5. Kuo, Y.L., Lee, J.C., Chiang, K.Y., Wang, R., Shen, E., Chan, C.W., Hsu, J.Y.j.: Community-based game design: experiments on social games for commonsense data collection. In: Proceedings of the ACM SIGKDD Workshop on Human Computation (2009)
6. Lenat, D.B.: Cyc: A large-scale investment in knowledge infrastructure. Communications of the ACM 38(11), 33–38 (1995)
7. Schubert, L., Tong, M.: Extracting and evaluating general world knowledge from the brown corpus. In: Proceedings of the HLT-NAACL Workshop on Text Meaning (2003)

8. Singh, P., Lin, T., Mueller, E.T., Lim, G., Perkins, T., Zhu, W.L.: Open mind common sense: Knowledge acquisition from the general public. In: On the Move to Meaningful Internet Systems, 2002 - DOA/Coop IS/ODBASE 2002 Confederated International Conferences DOA, CoopIS and ODBASE 2002 (2002)
9. Speer, R., Havasi, C., Surana, H.: Using verbosity: Common sense data from games with a purpose. In: The Florida AI Research Society Conference (2010)
10. Von Ahn, L.: Games with a purpose. Computer 39(6), 92–94 (2006)
11. Wang, R., Cohen, W.: Language-independent set expansion of named entities using the web. In: Seventh IEEE International Conference on Data Mining, ICDM 2007, pp. 342–350. IEEE (2008)

Automated Adaptation of Strategic Guidance in Multiagent Coordination*

Rajiv T. Maheswaran[1], Pedro Szekely[1], and Romeo Sanchez[2]

[1] University of Southern California
Information Sciences Institute and Department of Computer Science
{maheswar,pszekely}@isi.edu
[2] Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica
nigenda.romeosn@uanl.edu.mx

Abstract. We address multi-agent planning problems in dynamic environments motivated by assisting human teams in disaster emergency response. It is challenging because most goals are revealed during execution, where uncertainty in the duration and outcome of actions plays a significant role, and where unexpected events can cause large disruptions to existing plans. The key to our approach is giving human planners a rich strategy language to constrain the assignment of agents to goals and allow the system to instantiate the strategy during execution, tuning the assignment to the evolving execution state. Our approach outperformed an extensively-trained team coordinating with radios and a traditional command-center organization, and an agent-assisted team using a different approach.

Keywords: Multi-Agent Systems, Real-Time Coordination, Human-Agent Collaboration, Mixed-Initiative Approaches, Disaster / Emergency Response.

1 Introduction

Recent disasters in Haiti, Chile, Christchurch and Japan caution us that emergency response is a significant practical global issue. They draw attention to the need for technologies that assist human responders to perform more effectively. We address a fundamental challenge in these domains: most important tasks (e.g., rescuing injured, restoring services) are only revealed during execution, i.e., the severity, type and locations of injured people or the damage done are revealed *while* the rescue operation is being performed. Humans form high-level strategies that do not fully specify actions

* The work presented here is funded by the DARPA COORDINATORS Program under contract FA8750-05-C-0032. The U.S. Government is authorized to reproduce and distribute reports for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of any of the above organizations or any person connected with them. Approved for Public Release, Distribution Unlimited.

and dynamically implement them with information revealed during execution. Additional complexities include managing teams with heterogeneous capabilities, tasks that require coordination of multiple skills and resources, task failure, loss of agent capabilities and uncertain durations.

There are several relevant multi-agent system approaches. However, most require a specification of the problem that is infeasible under conditions where most tasks are revealed during execution. Considering all evolutions (e.g., all possibilities of injured and damage) scales the problem beyond the capabilities of current technologies. It is prudent and often necessary to allow humans to outline a high-level strategy. This necessitates having a formalization that captures how humans describe strategies, while maintaining sufficient richness to enable computational assistance for execution. The challenge is devising a strategy specification language amenable to human strategic thinking and algorithms to execute such strategies flexibly and efficiently.

Our contribution is an approach, STaC, composed of a strategy specification language that captures human-generated high-level strategies and corresponding algorithms that execute them in dynamic and uncertain settings. This partitions the problem into *strategy generation*, designed by humans and understood by the system, and *tactics*, orchestrated by the system with information to and from responders on the ground. STaC gives the ability to create changing *subteams* with *task* threads under *constraints* (e.g., focus on injured). The connection between a STaC strategy and the STaC execution algorithm is the notion of *capabilities*: agents have capabilities; tasks require capabilities. STaC dynamically updates the *total capability requirements* (TCRs) for the tasks in the strategy and assigns agents to tasks during execution following the human guidance.

STaC was evaluated in three high-fidelity independently-conducted real-world field exercises. We outperformed a traditional human approach by a significant margin in all three exercises, where a third team using software-assisted agents did not succeed. Additional analyses show STaC robustness with respect to strategy and automated planners that manage goals.

2 Motivation

We were challenged to create a multi-agent system that improved performance in simulated disaster-rescue field exercises. Each exercise was conducted in a park. Different parts were designated as sites with an unknown number of injured in *serious* or *critical* condition as well as *gas*, *power* and *water* substations which may have been damaged. Park pathways were roads that agents walked to travel between sites (see Fig. 1). Teams earned points by rescuing injured to hospitals or operational clinics (before a deadline associated with each injured person) and restoring damaged substations. The goal was to maximize points in 90 minutes.

The points per rescue and repair differed by site, injury type and service type. The number of injured or type of damage at each site was unknown ahead of time. To discover this information, agents with capabilities to *survey for injured* or *survey for damage* needed to visit the sites. Before performing any survey, the site first had to be *isolated* to ensure safety for those conducting the surveys. Isolation, survey, repair and rescue tasks all required different combinations of capabilities.

Fig. 1. Agent activities in the field exercises (left) and map of park and pathways (right)

Each team had 8 field agents and 2 commander agents. Each field agent had different capabilities, which included performing major or minor repairs for gas, power or water and saving certain types of injured. Commander agents stayed at a base and helped to coordinate activities. The baseline *Radio Team* operated under a traditional command structure and communicated only with radios. Our *STaC Team* and a third team with a different approach had radios and ruggedized tablet computers with cell modem and GPS capabilities running agents.

Consider an agent who completed surveys for damage and injured at a site. They can (1) start a low-probability repair alone, (2) wait for another agent to start a medium-probability repair, (3) get a repair kit from the warehouse for a high-probability repair, (4) rescue seriously injured, (5) wait for another agent to rescue critically injured, or (6) go to another isolated site and perform surveys. The right choice depends on the availability of other agents. Making good decisions is difficult without situational awareness.

The Radio Team realized that commanders could not make detailed decisions for agents in the field. Instead, they employed a strategy that formed subteams, defined itineraries for subteams and restricted actions to specific tasks (e.g., isolation, power repairs, critical injured rescue). These structural restrictions enabled commanders to delegate detailed decision-making to agents in the field. One commander assembled up-to-date situational awareness from multiple simultaneous field reports. The other focused on problem solving to execute the strategy by directing agents to locations, where agents would decide how to accomplish tasks. Consider the following exchange:

Agent 1:	Found power problem at Site 2, need power specialist.
Agent 1:	Found 3 seriously injured at Site 2.
Commander:	Power Specialist, go to Site 2.

In the meantime, Agent 1 succeeds with a low probability power repair.

Agent 1:	Power repaired at Site 2.
Commander:	Agent 1 go to Site 3 and do surveys.
Commander:	Power Specialist, skip Site 2 and go to Site 4.

Agent 1 and the Power Specialist are a subteam responsible only for power repairs and surveys for injured. Agent 1 arrives at Site 2 first, surveys for power damage, reports a problem, and, after looking at repair tasks, requests the Power Specialist. Agent 1 then surveys for injured and because the Power Specialist has not arrived, attempts and completes a low-probability repair. Consequently, the Commander redirects the Power Specialist to Site 4. Agent 1 makes detailed field decisions such as doing the power survey first to discover the needed capabilities and attempting the low-probability repair. The Commander, who has better situational awareness, executes the high-level strategy, i.e., assigning agents to goals. This simple scenario illustrates the management commanders perform. The complexity increases as all 8 agents frequently and often simultaneously report status, and task failures, delays, agent injuries, vehicle breakdowns and road blocks occur. The structural restrictions of the strategy let the Radio Team partition responsibility for decisions in a manner that allowed them to perform well.

Two insights yielded avenues for improvement: (1) humans do not execute their strategies efficiently (e.g., it may take a human commander a few minutes to gather situational awareness, calculate the desired actions and communicate them to the team), and (2) humans sometimes forget to task agents or communicate actions (e.g., forgetting to tell the Power Specialist to skip Site 2) causing waste. If a system could generate the appropriate assignments, it could calculate and communicate them in seconds and would not forget to enact them. The cost is that the formalization that systems require do not have the expressivity to capture the full space of strategies and adaptations that humans can generate. Thus, we must find a scheme and associated algorithms where loss in expressivity and flexibility is overcome by gain in efficiency of execution.

3 Approach

Figure 2 shows inputs and outputs of a system that addresses our problem. The domain model gives utilities for potential goals (e.g., restore power at Site 2) and tasks that achieve them, agents and capabilities, geography of sites and roads, etc., before execution. Field reports, received during execution, include discovered goals (e.g., 5 critically injured at Site 3), task success/failure, agent locations and availability, etc. User guidance is information provided by a human planner to influence system behavior. The objective is generating actions for human agents, i.e., where to go, what to do.

An "ideal" system could select assignments to maximize utility without user guidance. Current planning technology does not support this. We encoded simplified versions of the field exercise scenarios in PDDL [7]. We made all goals known *a priori* (i.e., gave prescience on what would be discovered), fixed travel durations, and removed failure, deadlines on injured and all dynamic events (e.g., road blocks, agent injuries,

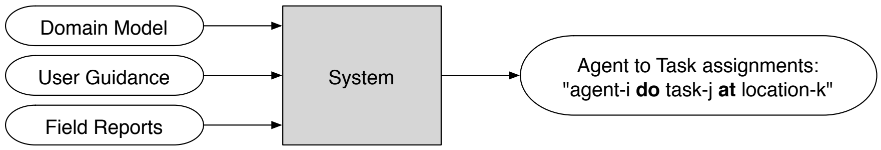


Fig. 2. The problem to be solved

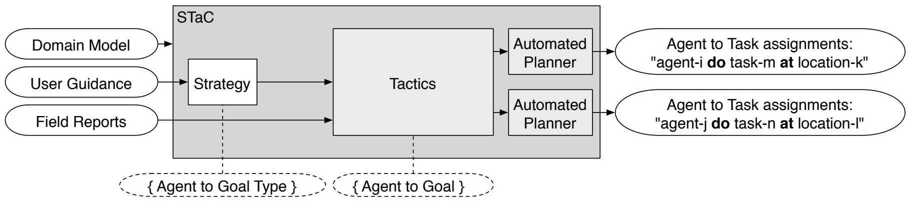

STaC
Domain Model
User Guidance
Field Reports
Strategy
Tactics
Automated Planner
Automated Planner
Agent to Task assignments: "agent-i **do** task-m **at** location-k"
Agent to Task assignments: "agent-j **do** task-n **at** location-l"
{ Agent to Goal Type }
{ Agent to Goal }

Fig. 3. STaC problem reformulation

vehicle breakdowns). From the deterministic planners we tried, only LPG-TD [8], and SGPLAN [6] could scale up to 5 sites under such restrictions[1]. This verified the need for significant user guidance in this domain.

We reformulated the problem as shown in Figure 3. The human planner gives a *strategy* that specifies how agents are assigned to goal types (e.g., rescue, restore, isolate, survey) under various constraints (location, timing, ordering). STaC provides the *tactics*, i.e., assigning agents to discovered goals by fusing the strategy with field reports (e.g., discovered goals, agent locations, availability) during execution. Once agents are assigned to goals, automated planners (one for each goal) handle the agent-to-task assignment. The decomposition allows the automated planners to independently solve significantly smaller problems, i.e., single goals instead of the entire mission.

For this approach to work, we must solve two related problems: (1) We need a strategy specification language that enables human planners to express agent-to-goal-type assignments, and relevant constraints. (2) We need algorithms to efficiently execute the strategy, i.e., make appropriate assignments of agents to discovered goals.

3.1 Problem Formulation

We do not present our full formalization of the general problem (described in Section 2) due to space. However, we discuss a few key ideas that will aid in understanding our approach. Before the exercise begins, there are certain goals (e.g., isolation, surveys) that we know we can achieve if we so choose, because the associated tasks to complete them exist (e.g., turning off power and gas for isolation). But, we cannot know whether there are people to rescue at a site or whether a particular substation needs restoration until we survey that site during execution. Thus, we introduce a concept, *goal type*, to discuss goals we might discover during execution. Utilities are associated with achieving particular goal types (rescues and restorations) at particular sites, if they are discovered during execution. Rescuing critically injured as Site 1 may be twice as valuable as restoring power at Site 7 which in turn may be twice as valuable as restoring gas at Site 7.

If we discover these goals in the exercise, task hierarchies capture the set and sequence of actions needed to achieve them. The leaves represent actions that agents with appropriate capabilities must perform. Each goal may have multiple task hierarchies that could achieve the goal but failure of any task within a hierarchy means that it no

[1] SGPLAN was the winner of suboptimal tracks in the 4th and 5th ICAPS International Planning Competition, while LPG-TD got a second prize in the 4th competition.

longer can achieve the goal. For example, a broken transformer at a power substation has multiple repair options. Each repair option is a task hierarchy that may have enablement or synchronization constraints. Similarly, there are several ways to rescue an injured person. Our task representation is a variant of CTAEMS [3], extended so that the capabilities needed to perform tasks are explicitly represented.

3.2 Strategies

STaC strategies enable human planners to (1) flexibly assign goals or goal types to appropriate subteams, (2) re-form subteams as needed, (3) order goals, while allowing parallelism, and (4) address special situations that require clever use of valuable resources and remaining time. We define a STaC strategy using a grammar:

```
1.          <strategy> := <thread>+
2.            <thread> := <agent>+ <goal-constraint>+
3. <goal-constraint> := <goal-spec> <constraints>
4.        <goal-spec> := <goal> | <goal-type> <location>
5.      <constraints> := [<timing-constraints>] (<capability> <usage-limit>)*
```

Strategies are composed of *threads* that can operate in parallel during execution based on agent availability. Each thread is composed of a subteam (<agent>+) and a sequence of goal or goal types they should pursue under some constraints (<goal-constraint>+). Threads and goal-constraint tuples are ordered and agents move sequentially through those where they are part of the subteam. Timing constraints prevent agents from attempting goals when there is insufficient time, redirecting them to quicker goals when time is running out. Capability constraints limit how often a capability can be exercised while pursuing a goal. Using these constraints, a human planner can control the amount of work done (e.g., rescue only 4 injured). In a later thread, a subteam can revisit the same location and achieve more goals of the same type. Goals and goal types can appear in multiple threads and multiple times within a single thread. Thus, human planners can create strategies that specify multiple attempts to meet goals at different times, with different agents and different constraints.

The table below shows an example strategy. For compactness, we show it in tabular format rather than the grammar, and encode goal types and locations as $g@l$ where g is a goal or goal type and l is a location:

Thread	**Subteam**	**Goals & Constraints**
1	A_1	*survey@(a, b, c, d)*
2	A_2, A_3	*gas@(a, b)*
3	A_4, A_5	*power@(c, d)*
4	A_1, A_4	*water@(a, b)*
5	A_2, A_3, A_5	*injured@(a, b, c)[drop-off 6]*
6	$A_1, \ldots, A_5$	*critically-injured@(d, e)*

In thread 1, A_1 is to survey locations a, b, c and d. The others are split into subteams optimized for gas (thread 2) and power repairs (thread 3). Subteams are reshuffled after these threads complete. The strategy limits each subteam to specific goals with the expectation that threads 1,2 and 3 complete at roughly the same time.

While the strategy sends both A_2 and A_3 to locations a and b, STaC algorithms will optimize the execution by determining when agents can skip goals, locations and threads. For example, the surveys A_1 performs may reveal no damage in b or reveal goals with tasks requiring a single agent. In the former case, both agents can advance to their next threads without visiting location b. In the latter case, one agent will be sent to location b while the other proceeds to its next thread.

In threads 4 and 5, subteams are reshuffled to optimize for different goal types. A_1 and A_4 focus on water repairs at locations a and b, which by this time should have been surveyed. The other agents focus on rescuing injured at locations a, b and c. Thread 5 limits the agents to rescuing 6 injured people by limiting the *drop-off* capability. This forces agents to move to other locations where they are also needed. In thread 6, all agents come together to rescue critically injured at two locations until time runs out. The sequence in which agents arrive at thread 6 is determined by events and outcomes during execution. The example illustrates how a strategy can exert control while also providing flexibility to accommodate variance during execution. A strategy constrains the space of execution paths for a team. The instantiated path is determined at run-time.

An important issue is to help human planners develop good strategies. We built simulation and visualization tools to help human planners understand the evolution of strategies [10]. Using these visualizations, planners can observe the behavior of subteams and refine the strategies as appropriate.

3.3 Executing Strategies

A strategy puts structure on who can be assigned to what, when and where. Once goals are discovered during execution, the exact who, what, when and where must be decided based on agent availability and location along with the strategy. We must continuously assign and re-assign agents to goals. The choice of which agents remain and which are passed to the next goal depends on the needs of the particular goal being considered.

Our strategy execution algorithms calculate the capability requirements needed to achieve each goal and ensure that the collective capabilities of the agents assigned to the goal are sufficient to meet the capability requirements of the goals. The capability requirements are updated dynamically when agents complete tasks. As these requirements change, unneeded agents are re-assigned to the next goals in the strategy.

We assign agents to goals based on *total capability requirements* (TCR). The TCR of a task characterizes the set of capabilities required to complete it. TCR calculations aggregate capabilities required by atomic actions up the task hierarchy, across enablements and synchronizations that may need to be performed at different locations. TCRs are defined as follows:

```
1.         <tcr> := <req>+
2.         <req> := <req-num> <req-type>
3.  <req-type> := <req-elem>+
4. <req-elem> := <capability> <location> <amount>
```

A TCR structure is composed of *requirement elements* (line 4) which denotes that a certain *amount* of a particular *capability* is required at a particular *location*, e.g., two instances of minor power at Site 3. A *requirement type* (line 3) is a collection of these elements. Restoring a power at Site 1 could require simultaneously turning on power at the substation, and a power main located in a different city, Site 2.. The requirement type for this task would be composed of two requirement elements: one instance of minor power at Site 1 and one instance of major power at Site 2. A *requirement* (line 2) states how many instances of a particular requirement type are needed. If there are 5 critically injured at a site, the requirement for rescuing all of them would be 5 instances of the requirement type to rescue one critically injured. Finally, the *TCR* for a task is a collection of requirements. This could represent the requirements for all tasks at a site.

The TCR for a task is computed bottom-up, starting with the actions at the leaves of task structures. The *local* TCR for leaf nodes is a single requirement type, composed of one requirement element for one instance of the capability associated with the action at the location to be performed. The TCRs for all nodes incorporate TCRs from all enabling nodes and children nodes, if they exist. We first discuss how TCRs from children are aggregated and then present the general algorithm for updating TCRs (Fig. 4).

AsyncMerge (Alg. 1) takes in a set of TCRs and outputs a single TCR. This is applied only to tasks that do not require any synchronization of their children. We decompose and organize all the TCRs by requirement types, and then aggregate the associated requirement numbers based on how the planner views execution of that requirement type. A key definition is the function Operator[reqType] which determines how the aggregation occurs. For a requirement type where the planner desires serial execution, the operator is max. This may be used for repairs where a single skill can be used to satisfy multiple requirements. Alternatively, if the planner desires parallel execution, the operator is sum. This may be used to increase resources allocated to rescuing injured.

SyncMerge (Alg. 2) also takes in a set of TCRs and outputs a single TCR. This aggregates the TCRs of children of synchronization tasks which require capabilities to be devoted in a manner such that each child node executes simultaneously. A subtlety of synchronization tasks is that their children may be achieved in multiple ways. The TCR for a synchronization task must consider all ways that the children can be executed simultaneously. Each requirement in the TCR of a child node captures the capabilities that may be needed for it to succeed. A combination of requirements is a set that takes one requirement from the TCR of each child. We gather all such combinations to characterize what may be needed to successfully execute the synchronized task.

Consider a synchronized task with two children having TCR$_1 = \{1 \cdot (gas, a, 1), 1 \cdot (water, a, 1)\}$, TCR$_2 = \{1 \cdot (power, b, 1)\}$. Here, the first child has two requirements at location a, the second has one requirement at location b. All requirement types have a single element, representative of primitive actions. The TCR of the synchronized task will have TCR$_0 = \{1 \cdot [(gas, a, 1)(power, b, 1)], 1 \cdot [(water, a, 2)(power, b, 1)]\}$. This states that the task could require a *gas* capability at a with a *power* capability at b or a *water* capability at a with a *power* capability at b. At least two agents are required.

The *UpdateTCR* procedure (Alg. 3) dynamically updates the TCR of any task, based on the TCRs of both subtasks and enabling tasks. If the task is a leaf node (i.e., action), we use local TCR. Otherwise, we collect the TCRs of the children subtasks and apply the appropriate merge operation based on the task type. We then add the TCRs of the enabling tasks and merge them using *AsyncMerge*. As actions get completed or the deadline associated with any node passes, the local TCR becomes the empty set. These

Alg 1: AsyncMerge(tcrSet)

```
reqSet = ∅; reqTypeSet = ∅
for all tcr ∈ tcrSet do
  for all req ∈ tcr do
    for all reqSet ← reqSet ∪ req do
      reqSet ← reqSet ∪ req
      for all reqType ∈ req do
        reqTypeSet ← reqTypeSet ∪ reqType
for all reqType ∈ reqTypeSet do
  num = 0
  for all req ∈ reqSet do
    if GetType(req) = reqType then
      num =
        Operator[reqType](num, GetNum(req))
  newTcr ← newTcr ∪ NewReq(num, reqType)
return newTcr
```

Alg 2: SyncMerge(tcrSet)

```
reqTypeSet = ∅
for all tcr ∈ tcrSet do
  for all req ∈ tcr do
    for all reqType ∈ reqTypeSet do
    newReqType = reqType
      for all reqElem ∈ GetType(req) do
        newReqType
          ← newReqType ∪ reqElem
      newSet ← newSet ∪ newReqType
  reqTypeSet = newSet
for all reqType ∈ reqTypeSet do
  newTcr ← newTcr ∪ NewReq(1,reqType)
return newTcr
```

Alg 3: UpdateTCR(t)

```
tcrSet = ∅
if isAction($t$) then
  tcrSet ← tcrSet ∪ localTCR($t$)
else
  for all subTask ∈ ChildrenOf($t$)
    tcrSet ← tcrSet ∪ TCR(subTask)
  if TaskType($t$) = Synchronization then
    tcrSet = SyncMerge(tcrSet)
  else
    tcrSet = AsyncMerge(tcrSet)
for all enablerTask ∈ EnablersOf($t$) do
  tcrSet ← tcrSet ∪ TCR(enablerTask)
return newTcr = AsyncMerge(tcrSet)
```

Alg 4: AgentSelection(agents,tcr,constraints)

```
newTcr = removeDisallowed(tcr,constraints)
remainingSubsets = PowerSet(agents)
for all subset ∈ PowerSet(agents) do
  for all cap ∈ capabilities do
    if NumAgents(subset,cap) <
      Needed(cap,newTcr) then
        remainingSubsets ←
          remainingSubsets \ subset
minSet = GetSmallest(feasibleSet)
return bestSet = GetLowestCost(minSet)
```

Fig. 4. Strategy execution algorithms

modifications get propagated up to parent nodes and forwarded through enablement relationships so TCRs are an up-to-date picture of capability requirement.

Task TCRs are used to determine when an agent can be released from a goal-constraint tuple in a thread. *AgentSelection* (Alg 4.) takes in the current TCR for a task, the relevant constraints according the the human strategy for the current execution of this task and the available agents. First, we trim the TCR according to the capability constraints from the strategy, e.g., the strategy might disallow performing any repairs at the current time, so we remove requirements related to repair. Then, for each capability, we calculate the needed amount by the maximum <amount> in any <req-elem> with the given <capability> in the modified TCR. We consider all possible subsets of available agents who can meet all the needed amounts of required capability.

We choose the set with the fewest number of agents and break ties using a cost function. Each agent is weighted by the value of their capabilities so that the most valuable agents are not in the selected set. The weights can be customized for each domain. For the field exercises, the value of an agent was the sum of the value of its capabilities where the value of its capability was one over the number of agents possessing the capability. Agents outside the selected set are released from that goal-constraint tuple in the thread. TCRs enable STaC to dynamically manage the execution of the strategy.

Once agents are assigned to goals, the next step is to compute the plans for achieving the goals. These plans specify which tasks should be performed, and which agent should perform them. STaC is agnostic to the methods used to solve these planning problems, so we do not discuss them in detail. We used an MDP planner to construct policies for performing repairs and a custom BDI planner for rescues with simple heuristics for critically injured, which required pairs of agents to load patients into vehicles.

4 Evaluation

Our contribution is a strategy specification language and execution algorithms that enable humans to state a high-level strategy that our system can execute effectively in dynamic and uncertain domains. We evaluate (1) the approach as a whole, (2) the strategy language and (3) the execution algorithms individually.

Effectiveness of the STaC approach. STaC was evaluated in 3 field exercises with 3 teams: a baseline team coordinating using radios, a team assisted by STaC agents, and a third team also assisted by software agents built by a different team using a different approach. Each team received one week of training covering all aspects of the field exercise: the rules, the physical procedures to simulate rescues and perform repairs, etc. They toured the grounds to get familiar with the map and terrain, and conducted three days of full practice scenarios. The teams using agents were trained on the software and used it during the practice scenarios. The radio team was coached by a retired U.S. Marines commander on effective techniques for managing the command center and coordinating using radios.

Each scenario has 15 sites with over 100 repair and injury rescues that required over 400 actions: more than was achievable in the 90 minutes allotted. Disruptions such as road blocks, vehicle breakdowns and agent injuries forced re-planning during execution. The scenarios were significantly more complex than we have space to describe here. The formal description of the field exercise, the scenario specifications and the strategies that the STaC team used for each scenario are provided online[2]. Teams were given scenario specifications only 24 hours before the start of each exercise.

In Figure 5, vertical bars labeled Radio and STaC show scores from the field exercises. The horizontal bars show simulation results discussed later. The results for the third software-agent team were not published but were lower than those of the radio team. We outperformed the radio team in all three scenarios (by 26%, 26%, and 24%). It is difficult to assess whether the scores are good in an absolute sense as calculating an optimal solution, even if centralized, is infeasible for scenarios of such complexity.

The field exercise results provide compelling evidence for the effectiveness of the STaC approach because (1) the radio team represents a reasonable baseline as they were

[2] http://www.isi.edu/∼maheswar/prima-2011/

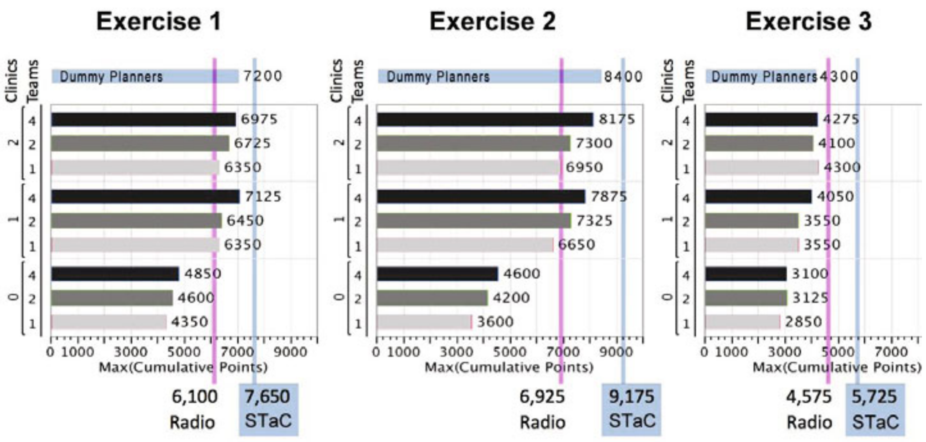


Fig. 5. Field Exercise Evaluation Results

extensively trained and operated with technology often in use today in such scenarios, and (2) the other agent-based team was unable to outperform the radio team in any scenario.

Effectiveness of the STaC strategy specification language. The complexity of the scenarios for the field exercise required sophisticated strategies to optimize performance. A strategy we employed for one exercise first created three subteams for gas, power and water to restore the "mains" for each service. This is a prerequisite for restoring substations and making strategically important clinics operational[3]. One of the remaining agents traveled to perform only isolations and the other began early deadline rescues. The agents then reformed into one large team to make the clinics operational. They then repartitioned into gas, power, water teams that only performed high-value repairs and a medical team that only rescued high-value critically injured. When the deadline was approaching, they became two large teams that performed only injured rescue. This required intricate orchestration to ensure that isolation, surveys, repairs, rescues, and repair kit management which all needed different combinations of skills were satisfied in the right sequence without causing excessive delays.

The goal of the first set of simulations was to assess the importance of the quality of the strategies. In the simulations, we used the logs of the real exercises to compute average values for the duration of all activities, and used the same outcomes and disruptive events that the evaluators had used in the field exercises. Our simulations of the STaC field exercise strategies were within 4% of the real results, so we posit that it is reasonable to expect a similar margin of error in the simulations of the other strategies.

We designed a collection of simple strategies organized along two dimensions: the number of clinics that would be made operational (0, 1 and 2), and the number of

[3] In each scenario, it was crucial to determine whether to restore any of two damaged clinics. Each required commitment of many agents and capabilities, but once operational, injured could be moved there rather than to the hospital which may be much further away.

independent subteams (1, 2 and 4). Each subteam had enough capabilities to achieve the enabling isolation and survey goals. All repairs for clinics were done first, as both the radio team and the STaC team did in the field exercises. Sites were ordered by total expected utility from rescuing all injured and repairing all substations and assigned to subteams in a round-robin manner from most to least valuable. Thus, each subteam had a single thread. When a subteam traveled to a site, it would attempt all the goals for the site, irrespective of utility, i.e., no constraints.

These simpler strategies performed worse than those used by the STaC team on the field (vertical bars labelled STaC). The greater the use of STaC features, e.g., subteaming and reformation (used after making clinics operational), the better the performance. These simulations suggest that exploiting richer strategy structures is valuable.

Effectiveness of the STaC Execution Algorithms. Several of the simpler, simulated strategies obtained scores that were better or comparable to the scores of the radio team. We attribute these results to the effectiveness of the execution algorithms. The radio team was not required to write down their strategies, however, informal conversations with their team members revealed that their strategies were more sophisticated than our simpler strategies. For example, in one scenario, they designated their least capable agent as a warehouse runner who got repair parts and delivered them wherever they were needed. The cases where the simpler strategies outperformed the radio team suggest that the radio team was not able to effectively carry out their strategies. Their commanders often had to handle multiple radio calls simultaneously and put agents on hold, wasting valuable time, and sometimes made incorrect decisions sending agents to sites where they were not needed. In the third scenario, which emphasized injury rescues, none of the simple strategies would have outperformed the radio team. The simple strategies had agents rescuing all injured at a site before going to a different site, and thus missed the deadlines for many injured. It was important to round-robin multiple rescue teams over sites to ensure saving the injured with urgent deadlines.

To evaluate the effect of the TCR agent-to-goal assignment algorithm, we also ran simulations where we replaced the planners for repairs and rescues with dummy planners. In these simulations, the agent-to-goal assignments were done using the TCR algorithms, but task assignments were done using simple algorithms. For repairs, we randomly selected repair tasks. For injured, we rescued the injured with the most urgent deadline and never considered waiting to have two agents simultaneously ready to load the more valuable critically injured. The results show that in the first two exercises the effect of the automated planner was relatively minor: the STaC strategies executed using the TCR algorithms (and dummy planners) would still have outperformed the radio team. In the third exercise that emphasized rescues, the greedy rescue algorithm did have a very detrimental effect.

5 Related Work

Real-world applications of the type addressed here are challenging because they must work in dynamic and uncertain environments [11,14,18]. The requirements for a richer model for actions and time are generally problematic for current planning frameworks based on MDPs, POMDPs, SAT, CSP, planning graphs or state-space encodings [4]. The main obstacle is scale as it is currently infeasible for a fully automated system to effectively reason about all the possible futures that may arise during execution. STaC is an approach that enables partitioning these large problems into smaller subproblems where these approaches can be used effectively.

Mixed-initiative approaches, where humans and systems collaborate in the development and management of solutions, are viable alternatives to tackle complex real world problems [12,1,9,18]. While mixed-initiative planning [5] is an established field, the type of problems we address with STaC bring new challenges. These include agent teams with heterogeneous and dynamic capabilities, tasks with uncertain durations and outcomes, and most importantly, planning over incomplete reward structures. In these problems, the goals that give reward to teams are not revealed *a priori*, but have to be discovered during execution.

Several mixed-initiative planning systems have been developed where users and software interact to construct and refine plans by adding and removing activities during execution while minimizing changes to a reference plan or schedule [1,9,12]. In the MAPGEN framework [1], humans control the construction of plans offline, while automated planning and reasoning capabilities are used to actively enforce constraints to generate safe plans for the Mars rovers. The output of the collaboration is a concrete plan. In the STaC approach, the human planner provides a high-level plan (strategy) that the system fleshes out during execution to produce concrete plans to satisfy the particular goals that are revealed during execution.

O-Plan is a knowledge-based mixed-initiative framework, which uses an agenda of outstanding (goal) issues to incrementally refine and repair plans by tightening constraints [16]. In O-Plan, a task assigner or commander specifies the set of tasks to consider; and a planning agent, using multiple classes of constraints, restricts the range of plans generated for the tasks specified [17]. An O-Plan-style planner could be used for dynamic problems by allowing the human planner to select goals as they are revealed and work with the automated planner to construct plans to achieve them. This approach involves human planners during execution, giving them more control than human planners have in the STaC approach. This approach could be effective in less dynamic situations where a human-in-the-loop planner does not become a bottleneck and where human input into the agent to task assignment is worthwhile. In our field exercises, a human planning during execution would have become a bottleneck. In the STaC approach, the role of human planners during execution is to oversee the execution of the strategy, assess its effectiveness and adjust the strategy accordingly. Our current system enables human planners to oversee the execution of the strategy giving commanders full situational awareness. In future work we plan to enable dynamically adjusting the strategy and distributing it to the agents during execution.

STaC can be seen as a particular form of control-knowledge that leverages the structure of domains. Work on using domain control-knowledge to solve complex planning problems has been extensively documented [13,2,16], and has been shown to convert intractable planning problems into a tractable ones [2,13]. Two planning systems that consider human intuition for problem solving are SHOP2 [13] and TLPLAN [2]. While SHOP2's search space consists only of those nodes that are reachable, TLPLAN uses its formulas to tell which part of the search space should be avoided. Similarly, STaC strategies can be viewed as a type of control knowledge that specifies constraints on the goals that should be attempted, the order for attempting them and the resources that can be used to fulfill them. One key difference is that STaC strategies allow expressing this control knowledge over goals that have not yet been discovered.

Systems like DEFACTO [15] study the role of adjustable autonomy to improve human/agent collaboration for disaster management. While adjustable autonomy is not the focus of our work, STaC enables human planners to delegate significant control to the system by imposing fewer constraints on strategies, or to exert more control by imposing additional constraints. It would be interesting to explore a DEFACTO-style approach to adjustable autonomy in STaC whereby humans would be consulted on agent-to-goal assignments in specific contexts (e.g., when the decision is critical and human planners have time to intervene).

6 Conclusions and Future Work

STaC is an approach to address multi-agent planning problems in dynamic environments where most goals are revealed during execution, where uncertainty in the duration and outcome of actions plays a significant role, and where unexpected events can cause large disruptions to existing plans. The key insight of our approach is to (1) give human planners a rich language to control the assignment of agents to goals (strategy language), (2) allow the system to assign agents to goals during execution, tuning the assignment to the evolving execution state (TCR algorithms), and (3) use off-the-shelf planners to select and plan the tasks to achieve these goals. Evaluations in 3 realistic field exercises showed that STaC is a promising approach. The STaC strategy language enabled us to encode sophisticated strategies. Additional evaluations using simulations of the field exercises showed that the TCR algorithms provided significant benefit. These simulations showed how in 2 of the exercises, simple strategies executed using the TCR algorithms produced better scores than more sophisticated strategies used by the radio team, but executed without the help of our algorithms. The evaluations also showed how the STaC approach effectively partitioned the large and complex planning problems to create small subproblems that traditional planning technology (MDP and BDI) could address effectively. The STaC system outperformed an extensively-trained team coordinating with radios and a traditional command-center organization, and an agent-assisted team using a different approach.

The evaluations suggest that the following enhancements would be valuable. First, our agent-to-goal assignment algorithms are based on the capability requirements of the current goals without consideration of capability requirements of future goals in the strategy. This local optimization may lead to situations where agents with important capabilities for future goals are inappropriately assigned to the current goals. Fast algorithms to reason more globally are desirable. Second, strategies are easy to understand from a local point of view, i.e., each strategy thread is easy to understand, but the evolution of a strategy is hard to predict. In the field exercises, we used detailed simulations of strategies and visualizations to understand their evolution and to judge their effectiveness. Each simulation ran for 20 minutes, so it was only possible to run a small number of simulations before an operation. It is infeasible to run simulations of this sort during execution to evaluate potential revisions to a strategy. Less detailed and significantly faster simulations coupled with strategy analysis algorithms would enable rapid evaluation of alternative strategy adjustments under different dynamic evolutions. This would help human planners evaluate and enact strategy adjustments during execution.

References

1. Ai-Chang, M., Bresina, J., Charest, L., Chase, A., Hsu, J.C.-j., Jonsson, A., Kanefsky, B., Morris, P., Rajan, K., Yglesias, J., Chafin, B.G., Dias, W.C., Maldague, P.F.: Mapgen: Mixed-initiative planning and scheduling for the mars exploration rover mission. IEEE Intelligent Systems 19(1), 8–12 (2004)
2. Bacchus, F., Kabanza, F., De Sherbrooke, U.: Using temporal logics to express search control knowledge for planning. Artificial Intelligence 116, 2000 (1999)
3. Boddy, M., Horling, B., Phelps, J., Goldman, R.P., Vincent, R., Long, A.C., Kohout, B., Maheswaran, R.: CTAEMS language specification: Version 2.04 (2007)
4. Bresina, J., Meuleauy, N., Ramakrishnan, S., Smith, D., Washingtonx, R.: Planning under continuous time and resource uncertainty: A challenge for ai. In: Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence, pp. 77–84. Morgan Kaufmann (2002)
5. Burstein, M.H., McDermott, D.V.: Issues in the development of human-computer mixed-initiative planning systems. In: Cognitive Technology (1996)
6. Chen, Y., Wah, B., Hsu, C.W.: Temporal planning using subgoal partitioning and resolution in sgplan. Journal of Artificial Intelligence Research (JAIR) 26(1), 323–369 (2006)
7. Fox, M., Long, D.: Modelling mixed discrete-continuous domains for planning. Journal of Artificial Intelligence Research (JAIR) 27 (2006)
8. Gerevini, A., Saetti, A., Serina, I., Toninelli, P.: Fast planning in domains with derived predicates: an approach based on rule-action graphs and local search. In: Proceedings of the 20th National Conference on Artificial Intelligence (AAAI), pp. 1157–1162. AAAI Press (2005)
9. Hayes, C.C., Larson, A.D., Ravinder, U.: Weasel: A mipas system to assist in military planning. In: ICAPS 2005 MIPAS Workshop, WS3 (2005)
10. Jin, J., Sanchez, R., Maheswaran, R.T., Szekely, P.A.: Vizscript: on the creation of efficient visualizations for understanding complex multi-agent systems. In: Intelligent User Interfaces, pp. 40–49 (2008)
11. Kleiner, A., Freiburg, U.: Wearable computing meets multiagent systems: A real-world interface for the robocupsrescue simulation platform. In: First International Workshop on Agent Technology for Disaster Management at AAMAS 2006 (2006)
12. Myers, K.L., Jarvis, P.A., Mabry, W., Michael, T., Wolverton, J.: A mixed-initiative framework for robust plan sketching. In: Proceedings of the 13th International Conference on Automated Planning and Scheduling (2003)
13. Nau, D., Ilghami, O., Kuter, U., Murdock, J.W., Wu, D., Yaman, F.: Shop2: An htn planning system. Journal of Artificial Intelligence Research 20, 379–404 (2003)
14. Nourbakhsh, I.R., Sycara, K., Koes, M., Yong, M., Lewis, M., Burion, S.: Human-robot teaming for search and rescue. IEEE Pervasive Computing 4(1), 72–78 (2005)
15. Schurr, N., Marecki, J., Scerri, P., Lewis, J., Tambe, M.: The DEFACTO System: Coordinating human-agent teams for the future of disaster response. In: Programming Multi-Agent Systems: Third International Workshop. Springer, Heidelberg (2005)

16. Tate, A., Drabble, B., Dalton, J.: O-plan: A knowledge-based planner and its application to logistics. In: Advanced Planning Technology, pp. 259–266. AAAI Press (1996)
17. Tate, A.: Multi-agent planning via mutually constraining the space of behaviour. Tech. rep. In: Constraints and Agents: Papers from the 1997 AAAI Workshop (1997)
18. Veloso, M.M., Mulvehill, A.M., Cox, M.T.: Rationale-supported mixed-initiative case-based planning. In: Proceedings of the Ninth Annual Conference on Innovative Applications of Artificial Intelligence. Menlo., pp. 1072–1077. AAAI Press (1997)

Weaving a Fabric of Socially Aware Agents

Mark d'Inverno[1], Michael Luck[2], Pablo Noriega[3],
Juan A. Rodriguez-Aguilar[3], and Carles Sierra[3]

[1] Department of Computing, Goldsmiths, University of London,
London SE14 6NW, United Kingdom
dinverno@gold.ac.uk

[2] Department of Informatics, King's College London, Strand,
London WC2R 2LS, United Kingdom
michael.luck@kcl.ac.uk

[3] IIIA, Artificial Intelligence Research Institute,
CSIC, Spanish National Research Council
{pablo,jar,sierra}@iiia.csic.es

Abstract. The expansion of web-enabled social interaction has shed light on social aspects of intelligence that have not been typically studied within the AI paradigm so far. In this context, our aim is to understand what constitutes intelligent social behaviour and to build computational systems that support it. We argue that social intelligence involves socially aware, autonomous individuals that agree on how to accomplish a common endeavour, and then enact such agreements. In particular, we provide a framework with the essential elements for such agreements to be achieved and executed by individuals that meet in an open environment. Such framework sets the foundations to build a computational infrastructure that enables socially aware autonomy.

1 Introduction

The current expansion of social networks (and tools to support them) provides some valuable opportunities to examine new models of *social intelligence*, in particular in situations in which multiple rational entities engage in a common endeavour. In this sense, social intelligence is an emergent property of the interactions between individuals, whether human, machine, or a combination of both. Key to achieving it is the requirement for *social awareness*: the ability of such individuals (henceforth agents) to understand their context in order to act in such a way that interactions with others can be meaningful and effective. In this context, and in seeking to contribute to the development of systems manifesting social intelligence, this paper addresses the underlying infrastructure needed to support social interaction processes.

These processes are usually *open*, in the sense that they involve autonomous individuals that may participate in (or leave) social interactions at will. Such social processes cannot in general be prescribed *a priori* but need to be adaptive, in view both of the autonomy of individual agents, and the openness of systems in which agents might not know each other in advance. In consequence, agents must themselves come to agreements and then put them in practice. Only by manipulating (creating, modifying, exchanging) such agreements in meaningful ways can agents create and execute complex social processes. It is this ability of agents to operate in environments in which they can understand, participate in, create and modify agreements that we refer to as *social awareness*. We do not discuss how to create, modify or argue about these agreements, but instead provide a framework with the minimal set of elements for such agreements to be achieved and executed.

In doing so, we provide the conceptual and technological foundation for a wide variety of applications. For instance, to build *ad hoc* coordination support environments beyond what, say, Facebook can currently provide, consider how a group of friends might organise their weekend outings. They might join a virtual group to decide what movie to watch, where and when to meet; then agree on what type of food to have, what restaurant to go to and if a reservation is needed choose someone to make it, and have them actually do so, etc. Similarly, we envisage systems that provide an environment in which companies come together as a supply network where a call for tenders results in contracts for production, purchase and delivery of goods or services that are then enacted, and where incidents and failures are dynamically resolved by the interested parties. In a similar vein, a game designer could design and run multi-party distributed role playing games. In more abstract terms, we seek to develop functionalities for agreed-upon regulated social interaction in a flexible, principled way.

While the notion of intelligence has been considered (and indeed formalised) from many perspectives, there has been little work on the *formalisation* of *interaction* of intelligent entities that facilitates social awareness. The structuring of interactions as a static blueprint similar to classical human institutions has been studied in the area of *electronic institutions* [2], and modelling software as an organisation of components with clear interactions among them was pioneered by Gasser [8]. However, the fact that the most important aspect of any interaction is to represent *joint* activities of agents, makes recent developments in process algebras (e.g. [4]) and workflow languages (e.g. [3]) not well suited, as their emphasis is on the programming of hard-wired processes constituting individual agents along with their interactions. In other words, agents are not allowed to autonomously agree on how and when to interact.

Software environments that enable social intelligence need to include the explicit generation and representation of *social interaction process agreements* (for example, jointly starting, jointly finishing or jointly entering an activity), on top of the simpler capacity to work together once the agreements are set. In response, we seek to contribute to this broad goal by providing a formal framework for the management of such interactions. Our framework includes those aspects that we claim are necessary to support the processes of how an activity may be organised and put into action. In particular, we make explicit (i) the requirements for meaningful communication among agents; (ii) the requirements for the set up of coordination or interaction processes; and (iii) the required operations that the environment needs to support social interactions and those that agents need in order to interoperate.

To ensure that our ideas have a clear and unambiguous semantics, but can also be described at the appropriate level of abstraction, we use formal specification techniques developed in software engineering, and give a formal account of the concepts underpinning interoperation in open systems, including the essential data structures and operations. While the description omits some aspects due to space limitations, a complete version can be found in [1].

The remainder of the paper is organised as follows. Section 2 presents the main data structures supporting the static definition of activities or interaction processes, Section 3 describes the data structures supporting the execution of interaction processes, and the most basic operation that agents may perform. Section 4 concludes.

2 Social Intelligence

Social intelligence, as considered in this paper, essentially arises from *activities* that are carried out by groups of agents. Hence, in our approach to open systems we need to include the basic notion of group meetings, or *scenes*, in which agents interact. We therefore postulate a framework for open systems in which the *activities* of agents are *social*, *decomposable*, *scalable*, *local* and *dialogical*. We consider each of these in turn below in order to motivate our approach.

For any activities to be performed together by groups of agents, those agents must coordinate their individual activities with each other. Such coordination underpins *social intelligence* and, in this sense, activities are *social*, because agents that interact need to be aware, first of others and their roles, and second that certain capacities (or roles) may be required to achieve a particular goal within a common activity. Activities are *decomposable* in the sense that the goals of an agent, and the overall goals of a group of agents, can be decomposed into simple activities whose performance achieves the individual and collective goals. This composition requires the interconnection of *atomic activities* into a *graph* in which the achievement of individual and social goals can thus be associated with particular paths (or to subgraphs) that represent particular combinations of goals.

Openness means that agents may enter and leave a system at any time, potentially causing a large number of agents to interact. To cope with the *scalability* that is required as a result of this, not only is problem decomposition needed, but so is the possibility of *replicating* the enactment of simple activities so that different groups of agents can be allowed to perform the same activity, concurrently, over an enlarging infrastructure. Openness and dynamism cause knowledge about others necessarily to be limited. In consequence, interactions are naturally *local* within subgroups of agents. This locality of interaction supports scalability by establishing bounds on interactions of agents, and also supports security and privacy.

Finally, activities are *dialogical* as they are achieved via agent interactions composed of non-divisible units that occur at discrete instants of time. These units can be modelled as *point-to-point messages* within a *communication language*, and physical actions can be considered as wrapped by appropriate messages of this form.

Now, in order to provide a computational model capturing this view of social intelligence, we have developed a complete, type-checked specification using the Z specification language. Z has been used to specify many different agent and multi-agent systems in the last 15 years (e.g. [5]); due to its computational, functional-programming style semantics [9], Z can provide an unambiguous formal account of a system and its operation as a basis for implementation. In what follows, we present a type-checked subset of our formal model for social intelligence in Z, detailing the key concepts, and justifying in text the need or those whose formal description is not given. We begin by identifying the languages and primitives required, before building up to the concept of a society by means of interactions achieved through dialogue.

2.1 Languages

Our concern is with specifying open systems in which agent *interactions* are *meaningful*, *contextual*, *consequential* and *regulated*. We handle these properties through different languages and constructs as follows.

Interactions are *meaningful* in the sense that the meaning of the interaction is the same for all participants. When interacting within a common environment, agents need to communicate about their problem domain, and thus an *ontology* that describes this domain is required. We call this the *domain ontology*, specified in terms of a *domain language* in which to express the *purpose and means* of the interaction. Although this shared language might be the result of some process of semantic alignment, for the interaction to be successful it must be meaningful to all participants.

Interactions between agents are *contextual* in the sense that their effects depend on the specific circumstances under which they take place. In fact, interactions generate a history of information exchanges — a sequence of messages between agents playing various roles — that determines the particular context in which new interactions are to be interpreted. The representation of such evolving circumstances requires a *stateful* architecture in which context is explicitly represented. Changes to the context of an interaction occur as actions are executed when agents speak. Since interactions have *consequences* for the context of the participants, an *action language* determines how to update the state of contexts after illocutions occur. For example, if *John* wins an auction for a box of fish, his credit (a variable associated with any buyer) is decreased by his winning bid amount.

Building on the domain language types, the *property language* provides the means to represent attributes of the components of an interaction model. Such attributes might indicate that buyers have credit, or that there can be only one auctioneer in an auction room. Interactions are further *regulated* in being constrained by context. For this reason, a *constraint language* is needed to guarantee that every atomic interaction occurs in the right context. For example, any bid in an English auction must be higher in value than a preceding bid.

Now, to bootstrap our specification, we need a set of basic types that specify agents, roles and time, variables, terms, the formula of the languages already mentioned above (domain, property, constraint and action), whose syntactic features we abstract out in this paper and simply declare as given sets.

$$[Agent, Role, Time]$$
$$[AgentVar, RoleVar, TimeVar]$$
$$[AgentTerm, RoleTerm, TimeTerm]$$
$$[DLFormula, CLFormula, PLFormula, ALFormula]$$

2.2 Roles and Relationships

Our aim is to specify patterns of interaction to enable meaningful interoperation of multiple socially intelligent individuals. To interact successfully, these individuals need to be *socially aware*: they need to know the actions that they and others may take. Of course it is not possible to define interactions at the level of individual agents in advance, so we need an abstraction of an agent in the context of a social setting for which we introduce the notion of *roles*. Roles enable us to describe, design and understand interactions in an abstract and re-usable sense. First, roles define concrete patterns of behaviour; that is, what can be said and to whom. Second, roles have relationships of different sorts that either further restrict behaviour (for example, a committee member cannot perform the actions of the chair) or determine subsumption policies (for example, the chair may take on the responsibilities of any member). Thus, our framework contains roles and any relationships that we may wish to specify between them. Formally, we define these simply in the schema Roles.

Roles

$roles : \mathbb{P}\, Role$

$socialrels : \mathbb{P}(Role \leftrightarrow Role)$

2.3 Interaction and Communication

Humans communicate through a variety of forms (visual, phonetic, linguistic) but if we are designing open systems to include mixed societies with software agents, then basing it on anything other than the linguistic would seem impossible. So agents in roles must interact by means of communication or, more precisely, by exchanging speech acts. Given this, the template for any such exchange must include a formula from the domain language (to identify the content), and an illocutionary particle (the basic illocutionary force of the communication, such as promising, commanding or asserting). Along with this, we must also provide a minimal context: a time term (to record when the utterance took place), and a sender agent and a receiver agent, both with their associated roles. For example, we might have an *inform* illocutionary particle, with a domain language formula *view(movie, Jaws)*, at time *36487*, from a *proposer* role to a *friend* role (instantiated at some later point with the agents *Alice* and *Bob* as sender and receiver). The syntax is thus similar to that of agent communication languages, FIPA or KQML [7,6], but we only identify the *core* components that are required for our model of social intelligence.

We call this combination of data items an *IllocutionType*, represented formally in the schema below, which first requires introduction of a set for *IllocutionaryParticle*. The schema includes a predicate that asserts that the sender and receiver agents must be distinct, ruling out agents talking to themselves as social interaction.

$[IllocutionaryParticle]$

$$\begin{array}{|l}
\textit{IllocutionType} \\
\hline
\textit{dlformula} : \textit{DLFormula}; \\
\textit{illocutionaryparticle} : \textit{IllocutionaryParticle} \\
\textit{time} : \textit{TimeTerm};\ \textit{sender}, \textit{receiver} : \textit{AgentTerm} \\
\textit{sendrole}, \textit{receiverole} : \textit{RoleTerm} \\
\hline
\textit{sender} \neq \textit{receiver} \\
\hline
\end{array}$$

When considering the kinds of interactions that may take place, some elements must be abstracted away. In particular, it is not known which concrete formulae will be exchanged, who will talk to whom or when. However, since interactions are regarded as dialogue patterns, it is known which roles the unknown agents will incarnate in a particular communicative act and the illocutionary force (or particle) that will be used. Thus, we introduce the notion of *Scheme* as a subschema of *IllocutionType*, in which the agents and the time, at least, are abstracted away as variables, where the time term must always be a free variable.

$$\begin{array}{|l}
\textit{isavar}\ \_ : \mathbb{P}\,\textit{AgentTerm};\ \ \textit{istvar}\ \_ : \mathbb{P}\,\textit{TimeTerm}
\end{array}$$

$$\begin{array}{|l}
\textit{Scheme} \\
\hline
\textit{IllocutionType} \\
\hline
\textit{isavar sender} \wedge \textit{isavar receiver} \wedge \textit{istvar time} \\
\hline
\end{array}$$

An *Illocution* defines what occurs when an agent acts through an *IllocutionType* for which all variables are bound. (In the definition below, we omit the predicate part that would simply state that all variables are bound.)

$$\textit{Illocution} == \textit{IllocutionType}$$

2.4 Conversations and Scenes

We have stated that agents interact by exchanging illocutions within group meetings that contextualise those exchanges. We now make more precise the fact that speech acts are always uttered in a conversation in a particular environment that involves the goals of the participating agents, the roles they are playing, and a particular shared set of variables modelling the properties of the conversation. If we do not group utterances into these conversations (which we refer to as *scenes* to emphasise the fact that agents are taking on specific roles within a conversation), then we could never be able to interpret an utterance. In our view it is the combination of roles, illocutions, and scenes together that provide agents with the necessary social awareness required to interpret messages.

Our example of agents deciding which movie to go see, is just one of several situations that agents may encounter in the larger set of activities involved in actually going to see a movie, including setting a time and place to meet, making travel arrangements, and actually going to the movie. Each of these separate situations can be regarded as a *scene* where agents exchange utterances. When an agent makes a speech act, the state of the conversation changes, we thus say it moves to a new *conversation place*. At any time during its life, a scene is in one of these conversation places, transitions between which are achieved either by agents uttering illocutions, or eventually by agents *not* uttering anything at all after some time. Thus, a scene is a directed graph of *conversation places*, where the arcs are *labeled* with speech acts, constraints, and actions. The speech act takes the form of an unbound illocution scheme, the set constraints (or pre-conditions) determine what must be satisfied for the speech act to take place, and the sequence of actions (or post-conditions) to update relevant information if and only if the speech act is successful.

First, we provide the formal definitions required, and then explain them.

$$[\mathit{ConvPlace}, \mathit{SceneName}]$$

$$\begin{array}{|l}
\hline
\textit{Label} \\
\mathit{scheme} : \mathit{Scheme} \\
\mathit{constraints} : \mathbb{P}\,\mathit{CLFormula} \\
\mathit{actions} : \mathrm{seq}\,\mathit{ALFormula} \\
\hline
\end{array}$$

$$\begin{array}{|l}
\hline
\textit{Scene} \\
\mathit{sname} : \mathit{SceneName} \\
\mathit{sceneroles} : \mathbb{P}\,\mathit{Role} \\
\mathit{limits} : \mathit{Role} \mathrel{+\!\!\!\rightarrow} \mathbb{P}_1(\mathbb{N}) \\
\mathit{places} : \mathbb{P}\,\mathit{ConvPlace} \\
\mathit{moves} : \mathit{ConvPlace} \leftrightarrow \mathit{ConvPlace} \\
\mathit{label} : (\mathit{ConvPlace} \times \mathit{ConvPlace}) \mathrel{+\!\!\!\rightarrow} \mathit{Label} \\
\mathit{startingpoint} : \mathit{ConvPlace} \\
\mathit{closing} : \mathbb{P}\,\mathit{ConvPlace} \\
\mathit{access}, \mathit{leaving} : \mathit{Role} \mathrel{+\!\!\!\rightarrow} (\mathbb{P}\,\mathit{ConvPlace}) \\
\hline
\forall\, \mathit{cs} : \mathit{ConvPlace} \mid \mathit{cs} \neq \mathit{startingpoint} \bullet \\
\qquad \mathit{cs} \in (\mathrm{ran}(\{\mathit{startingpoint}\} \lhd \mathit{moves})\star) \\
\forall\, r : \mathit{Role};\ \mathit{cs}_1 : \mathit{ConvPlace} \mid \mathit{cs}_1 \in (\mathit{access}\ r) \bullet \\
\qquad \exists\, \mathit{cs}_2 : \mathit{closing} \bullet (\mathit{cs}_1, \mathit{cs}_2) \in \mathit{moves}\star \\
\mathit{closing} \cap (\mathrm{dom}\,\mathit{moves}) = \{\} \\
\hline
\end{array}$$

The *Scene* schema above contains the following elements. First, it requires a name so that we can identify it, the set of roles that can act within it, and the limits on the number of agents that play those roles (e.g., one meeting coordinator to determine the movie of a cinema visit). Second, we must define the set of conversation places and those moves between them that may occur because of an utterance. Each such move must be labelled with the illocution scheme that needs to take place, a (possibly empty) set of constraints (e.g., to stop someone changing the cinema venue at the last minute when some people may already have left home), and a set of action formulae (e.g., if we buy tickets, then our balance goes down and the cinema's goes up). Third, every scene has a unique starting point but there are several places where the conversation may legitimately close; conversations are finished at a *closing* place, so no other place can be reached by a move from them; and, in a scene graph, all places must be reachable from the *starting* place, and there must be at least one *closing* place that is reachable from any place. Fourth, in a truly open interoperational framework, agents playing various roles may either join or leave an ongoing conversation at some socially accepted places. Some simple integrity constraints must be satisfied by the specification of a conversation, but they are omitted here for space reasons.

2.5 Weaving the Society

Because agents are concurrent and autonomous, a society amounts to a set of connected conversations. Any social system that respects the changing goals and evolution of behaviours of autonomous agents must thus provide mechanisms for moving freely between scenes, for weaving together these agents and their actions in the various scenes in which they are involved. Thus, now that we have defined scenes, we need some way to connect them so that agents autonomously move between them once a social decision is reached. In our model, the means of doing this is via transitions and arcs: arcs give routes out of scenes and into scenes, and transitions provide synchronisation and choice points for agents as they leave and enter scenes. The resulting network of scenes allows groups of agents to jointly decide whether to start a new scene, join a scene, leave a scene, or close a scene.

As indicated, arcs link scenes to transitions and transitions to scenes, with each arc associated with a set of actions from the action language and constraints from the constraint language, corresponding to preconditions that govern the ability of an agent playing a role to traverse an arc. For example, an agent is seeking to go to the group movie event, must express a preference for a particular movie, and must pay part of a group rate fee in advance. In such a case, there may be an arc to the movie decision scene with the constraint that the agent has voted for a movie, and an action that decreases the agent's balance by the movie fee. Networking scenes is necessary to capture the causal dependencies between scenes including order, synchronisation, parallelism, choice points, creation, change of roles between scenes and so on.

Formally, this is captured in the *Society* schema, which contains the set of all scenes, of which there must be one entry scene and one exit scene, which are distinct. It also contains arcs linking scenes. Then, there is a labelling function (*disjnorm*), which maps each arc to a disjunctive normal form of agent variables and role identifiers, defining the possible (non-empty) set of agents and associated (non-empty) roles that agree on simultaneously traversing the arc. Similarly, a second labelling function (*constraints*) maps arcs to constraints (in our constraint language), which individual agents must fulfill in order to traverse the arc. A third labelling function (*actions*) maps arcs to actions (in our action language), which are triggered when individual agents traverse the arc. In this way, each arc is labelled with (possibly empty) sets of constraints and actions. Finally, to enforce the earlier restrictions on entry and exit scenes, it is not possible to return to the entry scene, nor is it possible to leave the exit scene and return to another scene in the network of scenes.

$$Arc == (Scene \times Scene)$$

Society

$$\begin{array}{|l}
\mathit{allscenes} : \mathbb{P}_1\, \mathit{Scene} \\
\mathit{entryscene}, \mathit{exitscene} : \mathit{Scene} \\
\mathit{arcs} : \mathbb{P}_1\, \mathit{Arc} \\
\mathit{disjnorm} : \mathit{Arc} \pfun \mathbb{P}_1(\mathbb{P}_1(\mathit{AgentVar} \times \mathit{Role})) \\
\mathit{constraints} : \mathit{Arc} \pfun (\mathbb{P}\, \mathit{CLFormula}) \\
\mathit{actions} : \mathit{Arc} \pfun (\mathbb{P}\, \mathit{ALFormula}) \\
\hline
\mathit{entryscene} \neq \mathit{exitscene} \\
\neg\, (\exists\, a : \mathit{Arc} \bullet \mathit{second}(a) = \mathit{entryscene}) \\
\neg\, (\exists\, a : \mathit{Arc} \bullet \mathit{first}(a) = \mathit{exitscene})
\end{array}$$

3 State of an Open System

3.1 Agent Processes

At run time, agents can concurrently take part in multiple scenes; that is, they may perform several activities at the same time. For example, a human agent may participate in a skype call, in a meeting, and may also send and receive text messages at the same time. Similarly, a software agent in an auction house may simultaneously bid in several auctions that run in parallel. Certainly, these various activities can be interrelated in the sense that a text message may influence the agent's behaviour in the meeting and skype call, and the selling price in one auction may influence what an agent chooses to bid in another. In this respect, any model of social intelligence should be able to account, for each agent, for a set of different *processes* that share memory so that performance in one activity may influence performance in others. Each such process represents an agent playing a single role within an instance of a scene, and the number of such processes may change over time as different instances of scenes are created and terminated at different points in time. Thus, we introduce the notion of *agent processes*, which effectively capture the representation of state. In the *AgProc* schema, we define these processes, which have the owning agent, an identifier, the role of the agent, and the set of roles that the agent is allowed to take on at some later time.

$[\mathit{Id}]$

AgProc

$$\begin{array}{|l}
\mathit{name} : \mathit{Agent} \\
\mathit{place} : \mathit{Id} \\
\mathit{role} : \mathit{Role} \\
\mathit{allowedroles} : \mathbb{P}\, \mathit{Role} \\
\hline
\mathit{role} \in \mathit{allowedroles}
\end{array}$$

Given this, we can give a more precise account of the contextualisation of actions we had mentioned before. In order to define the state of a scene during its execution, we introduce the notion of a *scene instance*, which requires a record of the history of moves and the binding contexts in which utterances (speech acts) were made. A *FrameContext* represents the instantiation of variables in schemes, a *MoveContext* defines the move and the frame context in which a speech act occurred, and a generic *Context* is the sequence (or history) of such move contexts that have occurred since the beginning of the instantiation of the scene.

$$[\mathit{FrameContext}]$$
$$\mathit{MoveContext} == \mathit{ConvPlace} \times \mathit{FrameContext} \times \mathit{ConvPlace}$$
$$\mathit{Context} == \operatorname{seq} \mathit{MoveContext}$$

In addition to an identifier and the conversation context, any scene instance includes the *scene* of which it is an instance, the current conversation place and the set of *agent processes* involved in the scene instance. The predicates in the *SceneInst* schema guarantee that the scene instance is consistent with the data structures of its scene. In particular, we require that the current conversation place to be well-defined (within the conversation places of the scene), and that the number of agents in the scene instance is within the allowable range of the scene.

SceneInst

$$\mathit{scene} : \mathit{Scene}$$
$$\mathit{position} : \mathit{ConvPlace}$$
$$\mathit{agents} : \mathbb{P}\,\mathit{AgProc}$$
$$\mathit{context} : \mathit{Context}$$
$$\mathit{sid} : \mathit{Id}$$
$$\mathit{exitingAgents} : \mathbb{P}\,\mathit{AgProc}$$

$$\mathit{position} \in \mathit{scene.places}$$
$$\forall\, r : \mathit{Role} \bullet \#(\{a : \mathit{agents} \bullet (\mathit{a.name}, \mathit{a.role})\} \rhd \{r\}) \in \mathit{scene.limits}\ r$$

3.2 Making a Move in Group Interaction

The last element of our proposal is to identify the operations that a framework enabling the interoperability of socially aware agents needs to support. We have identified many such operations but, due to space constraints, here we only discuss the key operation *Speak*, which processes an atomic action of an agent: a speech act. Another twelve operations — dealing with the environment in which interactions take place (such as initialising an environment or creating a scene instance), or corresponding to voluntary acts of agents (such as joining a scene instance or allowing an agent to change roles) — are dealt with in a similar fashion; details can be found elsewhere [1].

Thus, using the definitions above we can specify how to process a speech act through the *Speak* schema below. In particular, we specify what happens when a speech act *i* is uttered successfully. Line by line, the *Speak* schema states the following. First, this is an operation that changes the state of the scene instance. We define the variable *nextmoves*, which is the set of potential next conversation places and the associated schemes that label them. The variable *test* is the potential new history that would be defined if a move took place. It is needed to see whether constraints are satisfied. The first predicate states that we generate the set of pairs of possible next moves from the current position and their associated schemes. The second predicate makes use of the function *proceed*, which takes a set of pairs described in *nextmoves* and an illocution, and returns a frame context and next position if one can be found that matches the illocution. In other words, this ensures that the illocution defines a specific move. The next predicate generates a test context using the generated frame context and destination place. Next, we determine whether the constraints of the scheme are satisfied within this test context (by making use of the predicate *sat*.) The position is updated as is the context, and the actions that label the move to the next position are queued for action.

$$
\begin{aligned}
&\mathit{proceed} : \mathbb{P}(\mathit{Scheme} \times \mathit{ConvPlace}) \rightarrow \mathit{Illocution} \rightarrow \\
&\qquad\qquad\qquad\qquad (\mathit{FrameContext} \times \mathit{ConvPlace}) \\
&\mathit{sat}\_ : \mathbb{P}((\mathbb{P}\,\mathit{CLFormula}) \times \mathit{Context})
\end{aligned}
$$

Speak

$$
\begin{aligned}
&\Delta \mathit{SceneInst} \\
&i? : \mathit{Illocution} \\
&\mathit{nextmoves} : \mathbb{P}(\mathit{Scheme} \times \mathit{ConvPlace}) \\
&\mathit{test} : \mathrm{seq}(\mathit{ConvPlace} \times \mathit{FrameContext} \times \mathit{ConvPlace}) \\
&\mathit{actionstodo}! : \mathrm{seq}\,\mathit{ALFormula}
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{nextmoves} = \{s : \mathit{Scheme};\ c : \mathit{ConvPlace} \mid \\
&\qquad c \in (\mathrm{ran}(\{\mathit{position}\} \lhd \mathit{scene.moves})) \bullet \\
&\qquad\qquad ((\mathit{scene.label}(\mathit{position}, c)).\mathit{scheme}, c)\} \\
&\mathit{nextmoves} \in (\mathrm{dom}\,\mathit{proceed}) \\
&\mathit{test} = \mathit{context} \frown \langle(\mathit{position}, \mathit{first}\ (\mathit{proceed\ nextmoves}\ i?), \\
&\qquad\qquad\qquad \mathit{second}\ (\mathit{proceed\ nextmoves}\ i?))\rangle \\
&\mathit{sat}((\mathit{scene.label}\ (\mathit{position}, \mathit{position}')).\mathit{constraints}, \mathit{test}) \\
&\mathit{position}' = \mathit{second}\ (\mathit{proceed\ nextmoves}\ i?) \\
&\mathit{context}' = \mathit{test} \\
&\mathit{actionstodo}! = (\mathit{scene.label}\ (\mathit{position}, \mathit{position}')).\mathit{actions}
\end{aligned}
$$

4 Conclusion

In this paper we have explored fundamental intuitions underpinning the notion of social intelligence, and on those grounds have outlined a framework to construct computational environments that support open interoperability among autonomous entities. We discussed the conceptual assumptions, data structures, constructs and functionalities — for joint activity definition and execution, and for agent flow — of the framework and illustrated their formalisation.

In particular, our framework allows us to incorporate new functionality into mixed societies of human and agents. For instance, such functionality might be used by a group of friends in Facebook to coordinate their weekly outings; by an NGO to set-up a fund raising campaign, by a group of companies and services to run a supply network, or by game designers to create and manage web-supported participatory role-playing games. Although in this paper we have only provided a formalisation of conceptual aspects, the framework may be supported by an associated computational architecture and corresponding development tools and methodologies. By using the Z specification language, which lends itself well to supporting implementations that directly implement specifications, we address these concerns, and provide a foundational basis for different implementations.

This paper thus provides an outline of those core elements that we believe are necessary in any framework that enables such open interoperability. It is an *outline* only due to space constraints, but the complete specification (in which this draws) [1] does provide full detail. As an outline, however, the specification is especially valuable, since it does not overcommit, but allows different reifications of the framework to suit different purposes, and at the same time provides a *minimal* set of core elements. Indeed, we believe that to contend with open, concurrent, regulated and socially aware interoperability, all those features that we have included are necessary. We have thus been careful not to include any accessory materials and have preferred to err on the side of abstraction, rather than commit to features that are dispensable. Finding the right balance here is a challenge, but one we believe we have met in this paper.

Acknoledgements. This work has been supported by the projects EVE(TIN2009-14702-C02-01) and AT (CSD2007-0022). We are all immensely grateful to Carme Roig for her tolerance, patience and support. Finally, thanks to Marc Esteva for discussions on his work on AMELI and electronic institutions in general.

References

1. A framework for communication in open systems. Technical report, `http://www.mediafire.com/?mfpq42e0194eqa8`
2. Arcos, J.L., Esteva, M., Noriega, P., Rodríguez-Aguilar, J.A., Sierra, C.: Engineering open environments with electronic institutions. Eng. Appl. of AI 18(2), 191–204 (2005)
3. Boile, J., Cardella, M., Blanyalet, S., Juric, M., Carey, S., Chandran, P., Coene, Y., Geminiuc, K., Zirn, M.: BPEL Cookbook. Packt Publishing (2006)
4. Cardelli, L., Gordon, A.: Mobile Ambients. In: Nivat, M. (ed.) FOSSACS 1998. LNCS, vol. 1378, pp. 140–155. Springer, Heidelberg (1998)
5. D'inverno, M., Luck, M., Georgeff, M.P., Kinny, D., Wooldridge, M.: The dMARS architechure: A specification of the distributed multi-agent reasoning system. Autonomous Agents and Multi-Agent Systems 9(1–2), 5–53 (2004)
6. Finin, T., Labrou, Y., Mayfield, J.: Kqml as an agent communication language. In: Bradshaw, J. (ed.) Software Agents. MIT Press (1995)
7. FIPA: Fipa 97 specification version 2.0 part 2. Tech. rep., FIPA - Foundation for Intelligent Physical Agents (1998)
8. Gasser, L., Braganza, C., Herman, N.: MACE: A flexible test-bed for distributed AI research, pp. 119–152. Pitman (1987)
9. Spivey, M.: The Z Notation, 2nd edn. Prentice Hall (1992)

Dynamic Ad Hoc Coordination of Distributed Tasks Using Micro-Agents

Christopher Frantz, Mariusz Nowostawski, and Martin K. Purvis

Department of Information Science, University of Otago, New Zealand

Abstract. The notion of μ-agents to develop complex software applications has been under active research interest for some time. Through improved organisational modelling μ-agents provide stronger support for decomposition and abstraction in decentralized applications. With the advent of the mobile application platform Android – which exhibits strong analogies to multi-agent system principles – we strongly believe that μ-agent-based modelling has become an increasingly attractive alternative. It can combine decentralized application development with the wide-ranging set of sensors and communication channels to foster both context-sensitivity and flexibility of applications. By integrating Android with the μ-agent concept mobile applications can put stronger emphasis on coordination of task-oriented agent organisations. As an example how this can facilitate the development of distributed applications, we describe an application for the field of "Unconferences" to dynamically schedule informal talks in an ad hoc manner. We model the central aspects of the application and show the advantages of our μ-agent-based approach. Finally, we contrast our approach to existing work in this field and suggest the consideration of μ-agents as an alternative to conventional object-oriented software development.

Keywords: Multi-Agent Systems, Micro-Agents, μ-Agents, Unconferences, Android, Agent-Based Modelling, Distributed Information Systems.

1 Introduction

The increasing penetration of smartphones into consumer markets has made ubiquitous computing a reality. Beyond the traditional communication role, smartphone capabilities are increasingly used for context-sensitive coordination of tasks and location-based services, supported by a number of sensors (e.g. GPS, accelerometer, gyroscope) and a wide range of communication channels (e.g. NFC, WiFi, Bluetooth).

As a result, many of the applications running on those devices are inherently complex. They must integrate multiple sensors and communication channels with different characteristics and behaviour (e.g. blocking/non-blocking communication) and often rely on concurrent and asynchronous communication patterns within the application.

The modern mobile application platform Android, reflects this understanding and allows flexible modelling of mobile applications from application components, structuring applications from asynchronously interacting front-end, back-end, content-related and context-related elements. Although the Android architecture simplifies complex application development, the application components do not provide enough flexibility to model more fine-grained functionality beyond the high-level application layout enforced by the Android architecture.

To improve the modelling capabilities and also allow a flexible concurrent layout of applications, we present our integrated approach of agent-based technology and the Android platform, which we call 'μ-agents on Android' (MOA). We see μ-agents as a lightweight approach to model concurrent and potentially distributed applications from autonomous entities. The micro-agent concepts that we introduce also provide a comprehensive organisational meta-model to allow the strong level of decomposition that Android application components cannot offer "out of the box".

We begin by introducing the notion of *application components* in Android and suggest some improvements to establish the development of a more flexible Android application framework. Following this, we introduce the concept of μ-agents and elaborate some of their key modelling features. After this, we describe how both technologies have been integrated as MOA.

To clarify the potential of our approach for dynamic information systems, we describe an application which benefits from those capabilities, and demonstrates well how to address coordination concerns for ad hoc talks at "Unconferences" [10]. Unconferences represents a comparatively young phenomenon and rely on a strong degree of decentralized organisation.

2 Android and μ-Agents

2.1 Android Application Components

The application platform Android [4], developed by the Open Handset Alliance under the lead of Google, is a Linux-based software stack providing a uniform approach towards writing smartphone applications. The Android application design principles make even elementary applications and their components interchangeable by the phone user. Android's software stack includes the Dalvik virtual runtime environment, which allows for application development using the widely known Java syntax. On higher levels of the software stack, capabilities are organized in a task-oriented manner using the notion of managers. For example the *LocationManager* provides support for location awareness and location-based event handling.

To model applications, Android supports the notion of *application components*, which can be used to model a basic template of an application by its functional characteristics. Android comes with four different application component types: *Activities* run in the foreground, directly interact with the user, and are of rather short-running nature. *Services* run in the background, are long-running and often maintain the core application state, as their functionality is least likely to be interrupted. *Broadcast receivers* represent an event subscription mechanism used both for system events (such as receiving text messages), and *Content providers* act as an abstraction layer for any persistent data source.

All these components (apart from Content providers) are linked by so-called Android *intents*, which represent a message structure encapsulating abstract request specification plus extra key-value pairs (extras). Intents are asynchronously executed and allow dynamic binding of application components at runtime. As a consequence, applications are constituted by the combination of application components and the intents registered with particular components, which are documented as an application manifest. The application manifest is a XML file describing application components and permissions, among other project-specific details.

2.2 Modelling Constraints of Android Application Components

Although Android provides a number of building blocks for modular application composition, we feel that the framework still constrains application development, in particular, the structural decomposition: Android applications often result in a rather static structure consisting of the four provided coarse application component types. From a modelling perspective, the provided modelling artefacts are considerably limited and only allow simple application templates, structuring applications by foreground and background activity, as well as context (e.g. events) and content storage. Applications thus show uniform structure and are typically modelled using conventional object-oriented programming.

This is not a fundamental flaw, but Android does not take the full step to ease the task-oriented modelling aspects such as a more fine-grained decomposition of functionality into more primitive and specialized application components while dealing with low-level aspects such as concurrency. To provide further decomposition of application functionality in Android, application developers need to fall back to Object-orientation. Along with the weaker abstractions developers also need to deal with aspects such as thread handling and loose the built-in capability for asynchronous message passing on this level.

In order to provide a better and also more uniform modelling support, we provide more fine-grained application-oriented modelling mechanisms based on software agents [11]. This way, application architects and developers can use the same conceptual and modelling tools on higher and lower levels of abstraction.

2.3 μ-Agents

The idea of using small-size agents to compose application functionality goes back to the late 1990's (e.g. [3], [8]) and is motivated by the suggestion to engineer comprehensive applications with numerous entities having narrowly specialised functionality in a cascading, hierarchical structure. The use of μ-agents, in the approach taken here, is similar to the one of object-oriented programming in that μ-agents can be recursively constructed from μ-agents. This allows effective decomposition in fine-grained entities and also allows abstraction from lower level functionality while constantly thinking in an agent metaphor. What makes μ-agents different from objects is that they pursue their own objectives (e.g. composing their functionality from other agents) and they do not only act in a reactive manner. Agents communicate asynchronously instead of blocking message calls. Internally however, the application developer is not tied to a particular interaction mechanism. μ-agents can thus largely differ in their internals but commit to common communication mechanisms. As a result, applications developed with this μ-agent concept in mind effectively exploit concurrency without the need to deal with thread-related aspects (such as in many object-oriented languages). At the same time, they deliver a performance which is comparable to existing object-oriented systems. With this motivation in mind, we implemented a μ-agent platform we call μ^2.

The μ-agent meta-model applied here is shown in Figure 1 and discussed in the following paragraphs.

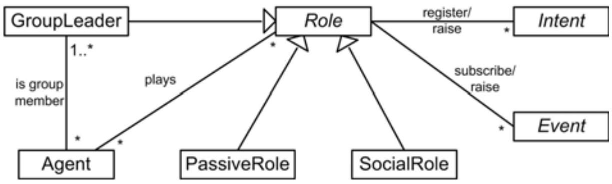


Fig. 1. Core relationships in μ^2

μ-agents are lightweight autonomous persistent entities which show reactive – and, potentially, proactive – behaviour and can efficiently interact in synchronous or asynchronous manner. Organisational aspects come into play when considering that μ-agents are designed to play one or more user-defined roles which are dedicated to fulfil *applicable intents*. Roles represent a set of potentially interrelated behaviours and may require necessary capabilities by a μ-agent in order to be played.

μ-agent *intents* (as opposed to *Android intents*) represent the notion of intentions, i.e. they are abstract request specifications. Roles registering *applicable intents* need to provide mechanisms to satisfy those *intents*. μ-agent *intents* can be raised by any agent on the platform and are automatically bound to a satisfying agent (if any agent can fulfil those).

Apart from the interaction mechanisms, organisational modelling capabilities are realized using the special '*Group Leader Role*'. Group leaders control an arbitrary number of sub-agents and serve the purpose to propagate control messages from the agent platform, as well as to structure the agent society. This allows flexible partitioning by functionality aspects. Agents playing the group leader role typically use sub-agents to compose their own functionality. Sub-agents playing this role can themselves have sub-agents which allows an agent organisation of arbitrary depth. The advantages of this organisational modelling mechanism not only include the strong degree of decomposition down to a very fine-grained level, but the organisational modelling also allows the definition of abstraction levels to hide details from the application developer.

Apart from the *group leader role*, other specializations to be mentioned are: *social roles* which support asynchronous communication between agents; and *passive roles* which only rely on synchronous communication. The purpose of the latter role is to provide the most fine-grained functionality without the performance penalty introduced by asynchronous message passing. The notion of events is used to provide event subscription capabilities, which are of particular concern when coordinating state in decentralised systems.

At this point we want to draw the reader's attention to one aspect, which is the similar role and denomination of intents. In both, Android and μ^2, *intent* instances represent request specifications for particular tasks. However, the intent structures of Android and μ-agents are not compatible. The dynamic resolution mechanism is similar, as in both cases they allow a loose coupling between application components (in the shape of *IntentFilters* in Android and, respectively, intent-based dynamic binding (via *applicable intents*) with μ-agents).

Some of the core principles of μ-agents such as loose coupling and asynchronous communication are essential part of mobile application platforms such as Android. μ-agents offer a more unified yet consistent architecture and extend those modelling mechanisms with an explicit task-oriented organisational perspective that is not part of Android's concept.

2.4 Integrating μ-Agents with Android

To make our vision of μ-agents a reality, we have ported our μ-agent framework μ^2 to the Android platform. A key to this operation is the open communication principle of Android that enables the integration of μ-agents in Android by translating the differing intent data structures at runtime.

Android intents have a static class structure but can hold dynamically-typed content. The structure of μ-agent intents is not predefined and leaves the developer with a wide range of options; the μ-agents intent structure also includes the capability to integrate methods. The core of μ-agents on Android (MOA) – which is schematically visualized on Figure 2 – is thus the conversion of the different intent types at runtime in order to allow direct interaction between Android components and μ-agents.

From an architectural perspective this integration is achieved by encapsulating the μ-agent organisation into an Android service which is addressable by other legacy Android application components. The μ-agents wrapped in the Android service can directly address arbitrary Android application components, pass data, and receive responses from addressable application components. Application components themselves can address μ-agents by using a predefined intent type that allows the specification of an agent name. Intents sent from Android not using this particular type are raised as events that μ-agents may or may not have subscribed to (and eventually react to).

To allow the direct access to Android functionality that is not addressable via intents (such as the GPS functionality of the LocationManager, or the

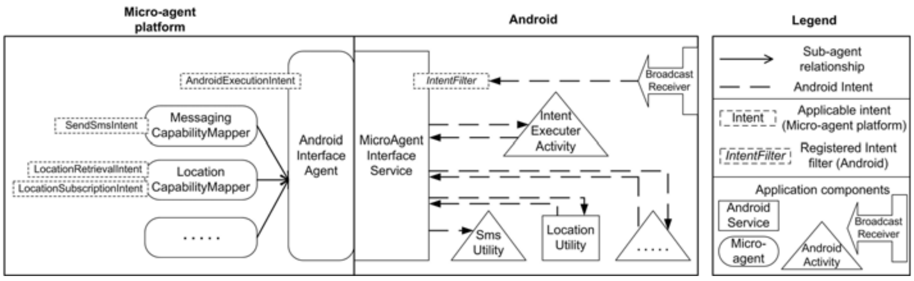


Fig. 2. μ-agents on Android Architecture

SmsManager), this functionality is remodelled with dedicated μ-agents and offered to other agents in the μ-agent organisation. Through flexible dynamic binding, this model allows agents to use Android functionality in a consistent manner without concern about where the actual functionality is executed. This gives MOA – compared to the desktop version of the μ-agent platform (μ^2) – an expanded functionality set, as μ-agents can thereby make use of the wide range of sensors, sensory information and context (e.g. retrieve location information) and communication channels (e.g. write SMS text messages).

3 Mobile App. for Ad Hoc Meetings at Unconferences

In order to show the potential of μ-agents to complement mobile application development, we describe a practical application that exploits some of the characteristics offered by μ-agents.

3.1 Application Context

Unconferences represent a new phenomenon that has emerged as a counterpart to conventional conferences that require intensive and expensive quality assurance mechanisms, organisation, committees, peer review, and publication of presented papers. The idea of an unconference is based on the perception that the collective knowledge of the audience is likely to be more extensive than that of the scheduled speakers; apart from typically time-constrained question sessions this knowledge is hardly used at conventional conferences. In short, many productive and inspiring talks actually seem to happen in hallways, by spontaneous interaction between a few conference participants instead of well-prepared and time-constrained presenters.

Apart from the contrary philosophy of this grass-roots approach, Unconferences also heavily rely on the widely adopted social media of the Web 2.0, such as blogs and social networking sites. Depending on the particular nature of the unconference, topics of concern can be suggested and rated via Wiki-based websites before the event, or are negotiated on the spot, at the venue. During the actual unconference, constant multi-channel interactions take place; apart from the presentations – which can eventually transform to discussions – participants constantly share ideas and inspiration via social media[1]. Input can also come from people without physical presence – for example Twitter, Facebook or the like. As a general rule, every person going to an unconference should expect to at least give his opinion on a topic of concern, or communicate his or her own idea.

Talks can spin off into smaller groups of people discussing either niche topics or specialist aspects. Those groups typically vary in size, as well as in composition with regards to personal attributes (such as rhetorical abilities and confidence) and skills (e.g. skills to use social media). Persons providing bright and innovative ideas may not necessarily have the soft skills of comparable quality. This makes the presence of a person with moderation skills beneficial, in order to convene a session and encourage participation of less dominant attendees, rather than leaving this entirely to the uncertain and unguided dynamics of the group.

Although general purpose social media can sometimes be effective to support the organisation of those dynamic sub-events, we think that dedicated software can outperform and improve the basis for productive and balanced interactions while capitalising on experiences from previous meetings. Another seemingly simpler but important logistic aspect is the coordination of concurrent meetings with differing numbers of participants and a limited number of rooms at a venue. Depending on the number of participants and the available time slots, the system can effectively coordinate spontaneous scheduling of rooms.

We use this scenario to show the potential of the agent-based modelling approach for mobile applications.

3.2 Mobile App. for Ad Hoc Organisation of Spontaneous Talks

The use of μ-agents as a modelling paradigm on mobile devices supports a fairly broad spectrum of modelling needs. Agents represent the interacting entities, intents are used to express particular requests, and events are used to inform an unspecified number of recipients. Apart from the support in shaping group structure, group size is also of concern. To facilitate this, the application demands more static information about the location of the venue in order to function productively.

The overall application is thus divided into an agent platform holding an agent that deals with room assignments and the management of topics, while an arbitrary number of further platforms – running the Android-based client part of the application – can actively use the system.

The following Figure 3 visualizes the static structure of this application. The notation used shows the organisational decomposition of applications into agents and emphasizes their hierarchical relationships. Agents are annotated with intents they are able to process (i.e. fulfill) and events they want to subscribe to.

[1] An exemplified overview on collaborative tools involved in Unconferences is provided by Crossett et al. [2]

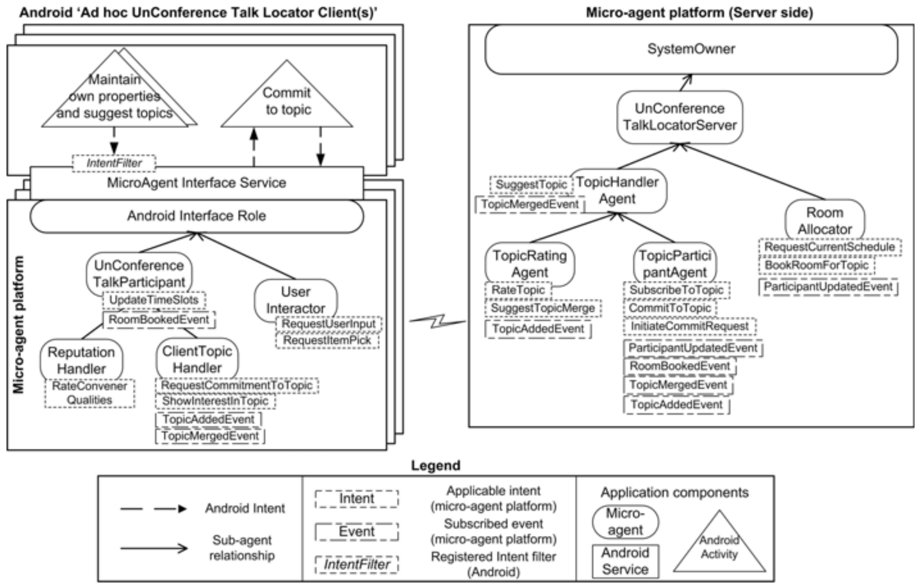


Fig. 3. Static Structure of application

For the description of the agent internals, we use message-centric Coloured Petri Nets (CPN) [7]. CPNs are a good way to capture the dynamic aspects within the different agents – and thus implicitly the dynamic structure of the entire application. With 'message-centric' we refer to the fact that μ-agents generally react on incoming messages (be it intents or events delivered in messages – or even custom messages without intents or events), which modify their state (visualized by places representing state repositories), and eventually produce outgoing messages. On this level, internals of μ-agents are thus fully represented in terms of message flow.

Instead of describing all application details in the form of diagrams, we provide below an overview of the overall application in a narrative manner. For a selected μ-agent we describe the internal message flow to show the intent-based loose coupling as well as to emphasize the decentralized character of application composition using μ-agents.

The application allows clients connected to the server part of the application to see, rate, and suggest new topics (by invoking the *SuggestTopic* intent resolving to the *TopicHandlerAgent* on the server side). New topics are immediately available to all clients and potential participants can suggest their merging (if topics seem related) – using the *SuggestTopicMerge* intent – and, most importantly, subscribe to topics of interest (using the *SubscribeToTopic* intent). Subscribing to topics does not automatically imply participation, but it ensures that the subscriber is notified about all modifications (e.g. via the *TopicMergedEvent* registered to the *ClientTopicHandler*) and contacted once another subscriber requests commitment to the topic. All subscribers then need to accept or decline this request, and in consequence a suitable room (according to the number of committing participants) and time is scheduled, with preference for topics of high rating.

This base set of functionality gives rise to strong dynamics that can unfold, given a high number of users and a wide range of different interests.

Participants can not only rate topics but also rate other attendees with regards to their mediation qualities, which is of help to suggest potential moderators for particular talks. The merging of topics can be suggested by any client connected the server, but demands for confirmation by the ones who originally suggested the affected topics.

To give some insight on the internals of μ-agents, we take a representative diagram for a server side agent entity.[2]

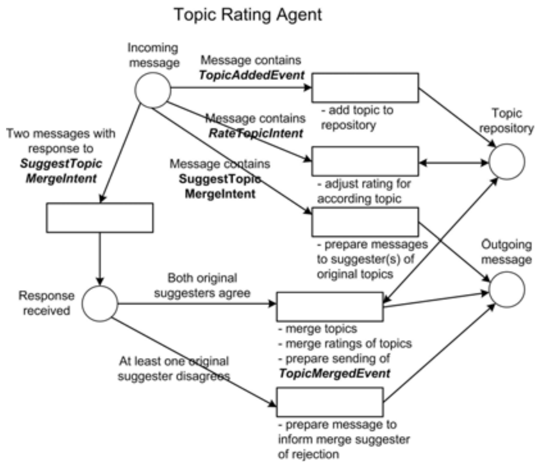

Topic Rating Agent
Incoming message
Message contains TopicAddedEvent
- add topic to repository
Topic repository
Message contains RateTopicIntent
- adjust rating for according topic
Two messages with response to SuggestTopic MergeIntent
Message contains SuggestTopic MergeIntent
- prepare messages to suggester(s) of original topics
Outgoing message
Response received
Both original suggesters agree
- merge topics
- merge ratings of topics
- prepare sending of TopicMergedEvent
At least one original suggester disagrees
- prepare message to inform merge suggester of rejection

Fig. 4. TopicRatingAgent on the server side

The *TopicRatingAgent*, whose operation is depicted by the CPN in Figure 4, resides on the server side of the application, and handles the aspects of adding a topic, rating it, and merging topics. Upon receiving a message, the agent checks it for contained intents or events. It reacts to messages containing a *TopicAddedEvent*; the topic is saved to the local topic repository (indicated as a Petri-net place in the diagram). Messages containing a *RateTopicIntent* carry information about a rating done by a client, and results in an update of the topic rating on the server side. Upon receipt of a *SuggestTopicMergeIntent*, the request is forwarded to the original suggesters of the topics concerned. Upon their response – and the case that both original suggesters approve the merging

[2] In contrast to the static structure shown in Figure 3, where intents and events can be disambiguated by the graphical notation, intents and event names shown in the CPN diagram in Figure 4 are written in bold italics and additionally carry the type name as suffix to facilitate the interpretation.

– the topics are merged and an according event is sent out in order to notify all subscribers (which includes the clients but also two other μ-agents on the server side as seen from the static application diagram in Figure 3). If they disagree, the client who suggested the merge is notified about the rejection of his request.

Note that Figure 4 shows the message-centrism of μ-agents which enables a streamlined modelling of core functionality with a low threshold between design and implementation. The only aspects not captured by the diagrams are the intent internals. In fact, the internals of intents are only known to the μ-agents that either raise or process those. This principle hides unnecessary information from non-affected agents and thus avoids the unintented or accidental interpretation by unrelated agents. As briefly mentioned in section 2.3, the dynamic binding (matching of raised intents against registered *applicable intents*) between different agents is realized by the underlying agent platform in an asynchronous manner and is transparent to the application level. Still, if of interest, agents can formulate custom messages addressed by agent name or using various addressing patterns (e.g. broadcast or rolecast).

Although only briefly shown through the above example, the MOA-based interaction mechanism allows a strong embedding of μ-agents with contextual information from sensor information (e.g. GPS, accelerometer, gyroscope) and use of communication channels (e.g. WiFi, Bluetooth, SMS). This opens the MOA approach for a wide range of smart context-embedded applications and eases extension of those. An example would be the suggestion of topics via SMS or the suggestion of informal meeting points for a smaller number of participants based on the proximity of all attendees. On the server side the obvious potential is to allow remote users to suggest topics (e.g. via the Web) and keep those informed about the current schedule of events.

4 Related Work

Looking at related work in the field of Unconferences is rather difficult, as both the process of organisation (i.e. often decentralized organisation, at most a central wiki) as well as the output (i.e. no formal proceedings) is informal. To the authors' knowledge, the only comprehensive description of a collaborative toolset for the operation of Unconferences is provided by Crossett et al. [2][3]. From the available literature, tools used to organise such venues are only weakly integrated (e.g. wikis, blogs, twitter) and rather heterogeneous in their nature. Given this context, a more unified but still decentralized and extensible μ-agent-based infrastructure seems helpful to foster a more efficient ad hoc organisation of events and management of limited resources (e.g. rooms and time) during Unconferences.

Taking a look at Android-based agent platforms, a number of those are derived from desktop platforms. An Android derivate of the prominent multi-agent platform JADE, JADE-Android [5], is one of the most prominent examples, and essentially is a subset of the desktop version of JADE. It allows the execution

[3] A further resource of information is the Unconference blog of Kaliya Hamlin [6].

of one agent. For the distributed operation it relies on at least one connected desktop version of the platform.

Another, quite elaborated, approach is JaCa-Android [9] which identifies agents as first order entities and interprets all handled objects (e.g. GPS coordinate) as so-called artifacts. The internal architecture of the platform relies on an implementation of the belief-desire-intention (BDI) model.

Agüero et al. [1] are among the earliest adopters of agent-based software development for Android. They implemented an abstract agent platform model (Agent Platform Independent Model (APIM)) on Android, which largely focuses on agent internals; organisational modelling aspects are not covered. The implementation extends the Android application components directly, thus building an agent platform 'on top of Android', in contrast to the interfacing approach taken by MOA (see section 2.4).

None of the above mentioned approaches encourages the use on many light-weight μ-agents, but instead, requires rather a limited number of more intelligent agents. This is a useful approach to offer smart applications, but hinders strong degrees of decomposition and flexible reconfiguration in cases involving newly introduced agents and changing agent capabilities at runtime.

5 Conclusion

The use of μ-agents, as a general application modelling tool, provides a strong degree of decentralization, effective (hierarchical) decomposition and transparent interoperability between application components (i.e. external dependencies of agents are captured using message-centric diagrams as shown in section 3.2).

The possibility of employing more primitive (bare-boned) μ-agents facilitates the decomposition to a fine-grained level. μ-agents can be used where the instantiation of yet another Android application component would be inefficient. MOA-based applications are open to extension and allow developers to capitalise on implementation efforts by reusing existing intents exploiting the dynamic binding mechanism. In principle, this mechanism is available across various applications and thus allows a better contextualization not only of isolated applications (by integrated sensors and communication channels) but over the entire application landscape. Applications can not only rely on other applications (such as those offered with Android's own development approach), but also on elements of other applications (i.e. μ-agents of MOA applications, or individual application components of traditional Android applications).

Using the example of Unconferences – an inherently dynamic and decentralized phenomenon in itself – μ-agents seem suitable to cope with the fluid and heterogeneous nature of those events, and offer useful mechanisms of coordination while maintaining the necessary flexibility. Relating it to the given application scenario, μ-agents allow the handling of unexpected events, such as the unexpected occupation of a scheduled location, or the 'no show' of participants. Also, bearing in mind the wide set of different tools used to run Unconferences, the agent principles are adequate to handle the coordination of many heterogeneous information sources. As such the application described in this context is only a starting point but provides enough understanding about the mechanics of μ-agents to inspire further, more context-dependent functionality.

We hope that the μ-agent modelling principles find adoption as an extension to the wide-spread use of object-oriented modelling, and we think that the lightweight framework presented in this article provides a low threshold to achieve this. The development around Android has shown how open software and open communication principles foster a diverse application ecosystem. μ-agents can extend those principles, not only to ease communication between devices and applications, but promote the ad hoc organisation of more elementary application elements, treating the mobile device as their natural open environment.

References

1. Agüero, J., Rebollo, M., Carrascosa, C., Julián, V.: Does Android Dream with Intelligent Agents? In: Corchado, J., Rodríguez, S., Llinas, J., Molina, J. (eds.) International Symposium on Distributed Computing and Artificial Intelligence 2008 (DCAI 2008). Advances in Soft Computing, vol. 50, pp. 194–204. Springer, Heidelberg (2009)
2. Crossett, L., Kraus, J., Lawson, S.: Collaborative tools used to organize a library camp unconference (March 2009), http://eprints.rclis.org/bitstream/10760/12831/1/Preprint-CollaborativeToolsUsedtoOrganizeaLibraryCampUnconference.pdf (accessed on: August 24, 2011)
3. Ferber, J., Gutknecht, O.: A meta-model for the analysis and design of organizations in multi-agent systems. In: Third International Conference on Multi-Agent Systems (ICMAS 1998), pp. 128–135. IEEE Computer Society (1998)
4. Google: What is Android? http://developer.android.com/guide/basics/what-is-android.html (accessed on: August 24, 2011)
5. Gotta, D., Trucco, T., Ughetti, M., Semeria, S., Cucé, C., Porcino, A.M.: JADE Android Add-on Guide, http://jade.tilab.com/doc/tutorials/JADE_ANDROID_Guide.pdf, (accessed on: August 24, 2011)
6. Hamlin, K.: Unconference blog, http://www.unconference.net (accessed on: August 24, 2011)
7. Jensen, K.: Coloured Petri-Nets – Basic concepts, Analysis Models and Practical Use, vol. 1. Springer, Heidelberg (1992)
8. Nowostawski, M., Purvis, M., Cranefield, S.: KEA - Multi-Level Agent Architecture. In: Dunin-Keplicz, B., Nawarecki, E. (eds.) CEEMAS 2001. LNCS (LNAI), vol. 2296, pp. 355–362. Springer, Heidelberg (2002)
9. Santi, A., Guidi, M., Ricci, A.: JaCa-Android: An Agent-Based Platform for Building Smart Mobile Applications. In: Dastani, M., El Fallah Seghrouchni, A., Hübner, J., Leite, J. (eds.) LADS 2010. LNCS, vol. 6822, pp. 95–114. Springer, Heidelberg (2011)
10. Wikipedia: Unconference, http://en.wikipedia.org/wiki/Unconference (accessed on: August 24, 2011)
11. Wooldridge, M., Jennings, N.R.: Intelligent agents: Theory and practice. The Knowledge Engineering Review 10(2), 115–152 (1995)

Programming Dynamics of Multi-Agent Systems

Cuiyun Hu, Xinjun Mao, and Huiping Zhou

School of Computer, National University of Defense Technology, Changsha,
Hunan Province, China 410073
hcy56316@163.com, mao.xinjun@gmail.com, icent@qq.com

Abstract. Dynamics are one of the most important properties of multi-agent systems (MAS), which often operate in open environment and with dynamically changing requirements. This paper firstly gives a comprehensive view of the dynamics in MAS based on "where" and "what" aspects of change and discusses the software engineering issues of engineering such dynamics. To solve related issues, we propose an organization-based programming approach that provides programming abstraction and mechanisms to describe and manage dynamics of MAS. An organization-based language for programming dynamics (OBLPD) of MAS is defined. The syntax of OBLPD is defined and its semantics are informally explained with a case study.

Keywords: MAS programming, dynamic MAS, reorganization, programming dynamics.

1 Introduction

Multi-Agent Systems (MAS) are often considered as an appropriate and powerful approach to develop complex systems [1]. Most of MAS and their environments are not static. Agents can enter or leave dynamically, agents' interactions and users' objectives may change at run-time and the situated environments are dynamic and uncertain [2]. Therefore, dynamics are one of the most important properties of MAS with respect to the dynamically changing environments and requirements. On the one hand, agents need to change their behaviors dynamically to adapt to changing environment and users' objectives. On the other hand, MAS should be able to reconfigurate and regulate its structure dynamically for the viability with its changing members, partial failures and its changing environment. Though great progresses of agent-oriented software engineering (AOSE) have been made in the past decade such as methodology, modeling language, architecture and pattern, programming language, etc. [2][3][4][5], the potentials of MAS paradigm has not convinced for such systems. To develop dynamic MAS is still a great challenge of AOSE.

A recent trend in the literature of AOSE is to study the dynamics and adaptation of MAS with organization theory [6]. On the one hand, organization concepts provide a high abstraction level to construct MAS, and several models, languages and methodologies based on organization abstractions have been proposed, such as ARG[4], Moise+[9]. On the other hand, Organization theory is also used as a foundation to investigate the dynamics of MAS, like self-adaptation [3]. However, current works usually concentrate on model and framework to support dynamics of MAS [3][5][8], few works being concerned on the programming technology. Most agent-oriented programming languages usually focus on the internal of agents (e.g. goals, beliefs and plans) and somewhat neglect social and organization aspects [10]. Therefore, there are usually two classes of approaches for implementing dynamic MAS: (1) agents are endowed with the ability to reason with the organization and manage its dynamics (e.g. OMACS[3]); (2) an organization middleware is developed to manage dynamics as a separation from the agents (e.g. MACODO[8]). However, the first approach usually makes the agents' internals complex and hard to implement, and the second approach affects the readability and expressiveness of the program. [11] proposes a third approach – extending current agent programming language with programming constructs for organization concepts. Since this work mainly focuses on the normative MAS, the mechanism for dynamics is inadequate currently.

This paper aims to propose an organization-based programming approach to implementing and managing the dynamics of MAS. The rest of the paper is organized as follows. Section 2 analyzes the dynamics in MAS with an example of cooperative robotic for airport sanitary maintenance, and discusses software engineering issues to engineer such MAS. Section 3 presents the core dynamic organization programming model and the programming mechanisms for dynamics, defines an organization-based language for programming dynamics (OBLPD) of MAS. A case is studied in section 4 and related works are discussed in section 5. Finally, conclusions are made and future work is discussed in section 6.

2 Dynamics in MAS and Challenges to Construct

2.1 Categories of Dynamics in MAS

Dynamics are considered as changes of behaviors and structure of agents and MAS and are cornerstones of the adaptation and robustness of MAS. Organization theory not only provides method to organize and manage the behavior and interaction of agents, but also provides an efficient partition of MAS [4]. According to modularity and encapsulation principles, this section studies dynamics in MAS distinguishing three different levels: agent level, intra-organization and inter-organization. Fig.1 shows the categories of dynamics in MAS, where the dynamics are usually related.

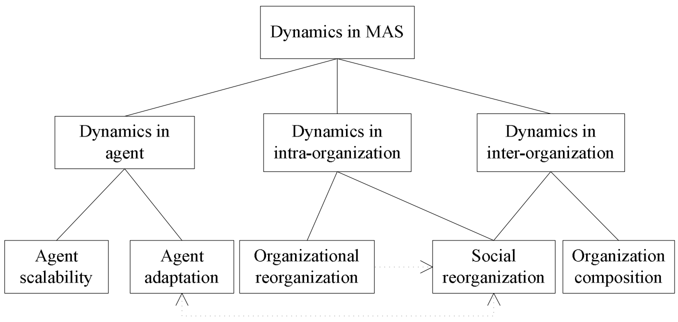


Fig. 1. Various dynamics in MAS

For example, an agent's adaptation behavior may cause the social reorganization and the social reorganization may need agents to adaptation. The detailed descriptions of the dynamics are given in following.

- Dynamics in Agent Level. As well known, a main kind of dynamics in agents is adaptation, which means that agents dynamically change their behaviors, rights and responsibilities according to the changing environments or goals. Besides, to operate in a new environment or achieve new requirements, agents may need to add new behaviors and responsibilities, where such dynamics exhibit as scalability.
- Dynamics in Intra-Organization. Organizations can be seen from two different views: organization structure and social structure [4]. So dynamics in organization structure can be considered as organization reorganization, which is mainly about the modification of the organization's structural elements (e.g., roles, norms and etc.). And with respect to social structure, social reorganization is named to represent changes of the membership and interactions in the organization, e.g., a new agent joining or leaving the MAS, interaction pattern instantiation and so on.
- Dynamics in Inter-Organizations. A complex MAS usually involves several organizations, and the relationships among each other are changing. On the one hand, dynamics such as creation, merging, splitting and disbanding organizations are considered as a kind of social reorganization, as they are usually including dynamically grouping agents at run-time. On the other hand, the cooperative organizations of each organization can change from time to time and such dynamics are called organization dynamical composition.

2.2 An Example: Cooperative Robotic for Airport Sanitary Maintenance

Throughout this paper, an example of Cooperative Robotic for Airport Sanitary Maintenance (CRASM) is studied to illustrate our approach. A software agent is deployed on each robot, which can play either explorer role to detect garbage or cleaner role to clean the explored garbage. Suppose there are two groups of robots: *groupA* and *groupB*, each of which is responsible to clean a waiting room. There are two robots in *groupA*: *robotA1* plays explorer and *robotA2* plays cleaner, and they cooperate each other to maintain the sanitary of the room. Moreover, there are three robots in *groupB*: *robotB1* and *robotB2* play explorer and cleaner respectively, and *robotB3* plays both explorer and cleaner. However, as the environment changes and failure may occur at any times, the CRASM system exhibits dynamic characteristics.

- Agent level. With respect to the agent *robotB3*, it will explore the room by playing explorer role at initialization, and if garbage is found, it will play cleaner role autonomously to clean the garbage.
- Intra-organization. Considering the scenario of *groupB*, when *robotB1* has explored garbage, he can send the garbage information to *robotB2* or *robotB3*. The interaction is dynamically initialized according to the state of the *robotB2* and *robotB3*, i.e., the *robotB2* is free or the activated role of *robotB3* is explorer.
- Inter-organizations. Given the failure that *robotA2* can not clean garbage anymore, and as there is no more agents can play cleaner in *groupA*, it is necessary for *groupA* and *groupB* to be emerged as one group for cleaning the two rooms.

2.3 Software Engineering Issues to Develop Dynamic MAS

Obviously, developing such dynamic MAS challenges current software engineering technology. We divide the requirements for developing such dynamic MAS into two parts: at design-time and at run-time.

- At design-time. Firstly, with respect to modularity and encapsulation principles, organizational concepts are needed to be explicit and first-class entities to decompose the system [4]. Secondly, with respect to agent adaptation, a separation of an agent's context-dependent behavior and its intrinsic behavior is necessary, so that an agent can bind different behaviors dynamically at run-time. Thirdly, the dynamics in intra-organization and inter-organizations desire a flexible structure of the MAS. As a result, abstraction (e.g., protocol) for the organization structure are needed besides agents and mechanisms are needed to support the dynamically interaction between agents and composition between organizations.
- At run-rime. On the one hand, platform and infrastructure are needed to support the execution of organizational concepts, such as organizations' creation/ destruction, which should be consistent with the management of agents. On the other hand, mechanisms supporting the dynamics should be employed, including adaptation mechanisms for dynamically composition of agents' behaviors at runtime according to its context and communication mechanisms for the dynamical interaction between agents, agents and organizations, even among organizations.

3 An Organization-Based Approach to Programming Dynamic of MAS

This section introduces an organization-based approach to programming dynamics in MAS, focusing on the issues discussed previously. Firstly, a core dynamical programming model is described at conceptual level, which bridges the gap between implementation and design with organization abstraction. Then, the programming mechanisms for dynamics are discussed. At last, an organization-based language is defined to implement the model.

3.1 A Core Dynamic Organization Programming Model

This section proposes a core dynamic organization programming model for dynamic MAS (see Fig. 2). In such model, agents' context-dependent behaviors are defined with roles while agents encapsulate the adaptation logic that describes how to enact different roles according to the context. Interactions among agents are programmed explicitly with protocols which are independently from agents, so that interactions can be easily added, removed and modified. A detailed description of these concepts is given in the following.

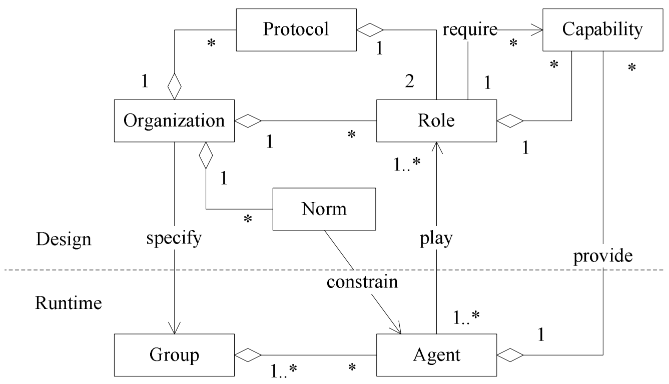


Fig. 2. A core dynamic organization programming model

- Agent. An agent is an autonomous and adaptive entity that can play multiple roles in multiple groups. An agent can own intrinsic property and capabilities that may be needed to play a special role, and adaptation behavior to change its roles according to the changing environment. The observable properties of agents are their identities, intrinsic behavior, groups belonging to and enacting roles. Whereas identity and intrinsic capabilities are immutable characteristics, groups and roles may vary over time and constitute states of an agent that are related with dynamics.
- Group. A group can be seen as a set of agents with aims to collaborate or share information. A group has an identity and a set of agents. Groups are said to be open, since new agents can join or leave at run-time. What's more, the duration of groups can be either transient or persistent. In the former case, a group is created for a specific task and is disbanded once that task is performed. The latter case allows groups provide a focus point for a certain capability that can be re-used by others [12]. For example, a bank can provide payment service in several trading.
- Organization. Organizations are introduced to specify organizational structure what persists when agents enter or leave a group [4]. So an organization is an abstraction of a class of groups each of which is one possible instantiation of an organization. An organization defines interaction patterns for its members and norms to constrain their behavior. Therefore, the cornerstones elements of every organization are roles that agents can play, protocols that specify the interaction between agents based on roles, and norms that constrain the agents' behavior.
- Roles. On the one hand, roles are the cornerstones in defining organizations, which define positions in a group with expected behaviors of agents to achieve the global goals. On the other hand, from the agent's perspective, a role defines the agents' context-dependent behavior that empowers agents to access the state of the group, access resources in the group and interact with other roles in the organization. Thus, a role can be defined as a set of requirements capabilities that agents must provide to play it, and a set of powers that can extend agents' capabilities.
- Protocol. A notion of protocol is introduced to solve uncertainty and dynamics of interactions, which is a binary association entity that expresses the cooperation

among agents and separates business logic and interaction logic to support dynamic interaction. All protocols need to specify roles bound to the protocol and how the messages are transferred.

- Capability. Capabilities as the encapsulations of the agents' behavior are the key to determining exactly which agents can play which roles in the group. That is, to play a role, an agent must provide the required capabilities in the role and the agent gains the power capabilities when playing it.

3.2 Programming Mechanisms for Dynamics

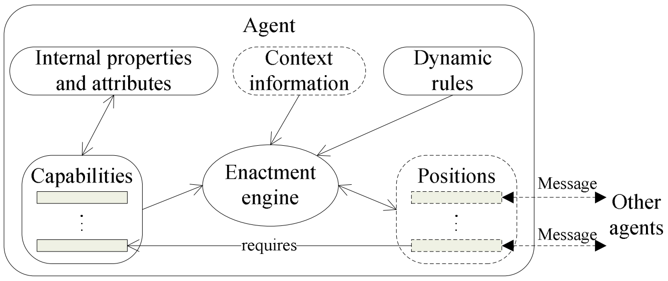


Fig. 3. Internal of an agent with enactment mechanism

Role enactment. Role enactment is provided as a cornerstones mechanism for the agent adaptation and social reorganization. As roles enhance the modularity of the MAS program, role enactment makes an agent composes its behavior dynamically in run-time. Inspired by [11], we take roles as execution entities named positions. Enacting a role means the agent has gained a position in the group. As an agent can play multiple roles, it can own multiple positions and is responsible for the coordination and management of the execution of each position.

The internal of an agent with enactment mechanism is represented in Fig.3, where dynamical constructs, changed at run-time, are outlined with a dashed border. The role enactment is supported by an enactment engine within the agent. The engine takes as input the context information, the events generated by its capabilities, together with the programmer-defined dynamic rules that describe how to change roles for the current context or generated event.

Delegation mechanism. A delegation mechanism is provided to support the intra-organizational reorganization for partial failure. That is, if an agent fails to complete a task, it will notify it to its group, and the group will search its members with the same role, and further delegate the task to the appropriate member, as Fig.4 shown.

3.3 An Organization-Based Language for Programming Dynamics

As previous description, a MAS can be programmed as a set of organizations that describe the interaction patterns and context-dependent behavior in terms of protocols and roles. This section proposes an Organization-Based Language for Programming

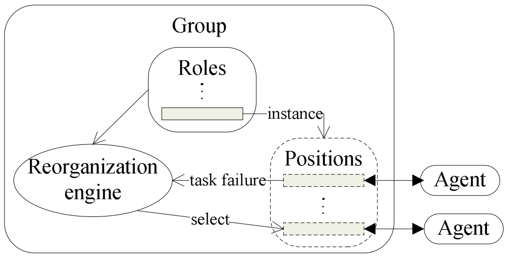


Fig. 4. Delegation mechanism in the group

Dynamics (OBLPD) based on the previous model and mechanisms. In following, we describe the syntax of the language facilities and informally explain their meaning.

The EBNF syntax definition of OBLPD is specified in Fig. 5. The initialization of a MAS is specified by a set of environments where its agents may situated and a set of groups each of which is described with a name, its organization type and its situated environment. The declaration of an environment consists the <environment_name> , its <environment_type> and the <environment_initialization> that initialize the initial state of the environment. The <organization_intialzation> describes the agents that are created when the group is initialized. In following, the core constructs, i.e. organizations, protocol, roles and agents are explained in detail.

The specification of an organization starts with the keyword "persistent" or "transient" to denote the type of the organization. Generally, persistent organizations are usually created by designer, while transient organizations are usually created dynamically at run-time. Thus, in a transient organization <initiation> and <elimination> must be defined to specify the triggering events and actions that execute when its groups are created and disbanded, respectively. The implementation of the <initiation> usually consists a list of <agent_creation> statements, each of which starts with the keyword "agent:" followed by the <agent_name> with the keyword "enact" and a list of <role_name>. And <agent_initialization> is used to describe the initial state of the agent. The roles of an organization are declared with the keyword "roles:" followed by a list of <role_name>.

Protocols define the interaction patterns among agents with their enacted roles. The keyword "repulsive" is used to declare a protocol whose partner can not be the same agent. For example, an agent can not play both seller and buyer roles in the same trading protocol. However, in CRASM a robot agent can play the explorer and carrier roles in the same protocol. A protocol is created when an agent joins the group and the role it enacted is initiator of the protocol, and its execution is based on the initiator's behavior. Thus, the <protocol_body> specifies the time for execution with the keyword "before", "after" or "on" followed by a <role_capability> of a <role_name>, which means before or after the execution of a special capability or triggered by some event during the execution of a special capability, respectively. Moreover, the statement in the protocol is a sequence of messages and defined with <performative_action> such as "request", "inform", "query", "accept" and "reject".

```
<MAS> = {<environment_name> ":" <environment_type> "{" <enviornment_initialization> "}"}
            {<group_name> ":" <organization_name> {"@" <environment_name>} "{"
            <organization_initialization > "}"}+;
<orgranization> = ("transient "|"persistent") "organizaiton" <organization_name> "{"
                    "roles: " <role_name> {"," <role_name>)} ";"
                    {["repulsive"] "protocol" <protocol_name> "(" <role_name> "," <role_name>")" "{"
                    <protocol_body>"}"}
                    [<initialization>][<elimination>]
                     "norm" <norm_name> "{" <constraint_statement> "}";
<initialization> = "initialize() when" <event_name> "{" {<agent_creation>} "}";
<elimination> = "eliminate() when" <event_name> "{" {<clear_action>} "}";
<agent_creation> = "agent :" <agent_name> "enact" <role_name> {"," <role_name>} "{"<agent_initialization> "}";
<protocol_body> = "run(){" ("before"|"after"|"on") <role_name> "." <role_capability> "{"
                     {<role_name> "." <performative_action> "(" <role_name> ")"
                     ("after"|"before") <role_name> "." <role_capability>}+;
<performative_action>="request"|"inform"|"reqery"|"accept"|"reject";
<role> = <role_kind> "role" <role_name> ["roleof" <role_name> {"," <role_name>}] "{"
          {["agent"] <property_type> <property_name>}
          {"agent" "capability" <capability_name>}
          {<capability>}+"}";
<role_kind> = ""|"abstract"|"singel"|"mono";
<capability> = "capability" <capability_name> "{"
                         <capability_body> "}" [<execution_result> "{" {<generate_event_statement>}+ "}"]
<execution_result> = "done"|"failed";
<agent> = "agent" <agent_name> "play" <role_name> {"," <role_name>}    "{"
                     {<property_type> <property_name>}
                     {<capability>}
                     "strategy" <strategy_name> "{" {<dynamic_rule>}+"}" "}";
<dynamic_rule> = (("when" <event_name> ) | (("after"|"before") <role_name> "." <role_capability>)) "{"
                    {<enactment_action>}+ "}";
<enactment_action> = ("enact"|"deact"|"activate"|"deactivate" |"enactOrActivate") "(" <role_name> ")"
```

Fig. 5. EBNF grammar of OBLPD

Besides normal roles discussed previously, three special kinds of role are defined: abstract roles, single roles and mono roles. Firstly, abstract roles are used for reuse of execution entities. For example, a student role can be defined as an abstract role and undergraduate and graduate roles can be defined for student. When an agent deacts undergraduate and enacts graduate, the property and behavior about student can be preserved. The relationship is defined with the keyword "roleof". For the student and undergraduate roles, the code can be programmed as following. It is noted that an agent can not play an abstract role directly, but only one of its reified roles. Secondly, a single role means that a maximum of one player for the role in a group, i.e. there is only one position for the role. For example, there is only one president in one university. A mono role means that a maximum of one position of the role per agent.

The internal of a role defines the requirements from its players and powers to its players. The keyword “agent” is used to declare the required property or capabilities which must be provided by its player to succeed in enacting it. The powers which its player has gained when playing it are defined as capabilities. Since the execution of one role’s capabilities may affect the behavior of its player and the structure of its group, the keywords of “done” and “failed” are introduced to declare that some events may be posted in the corresponding situation. And these events are mainly used to trigger agents’ adaptation, interaction starting and social reorganization.

For the specification of the syntax of agents, strategies are used to describe the dynamic logic as a set of rules (see Fig.5). The dynamic rules can be triggered in three ways: some event generated, before or after an execution of a role’s capability.

4 Case Study

Now, the CRASM described in section 2.2 will be programmed with our programming approach, shown in Fig.6. Explorer and cleaner roles require their players have *Moving* capability, and provide *Explore* and *Clean* capability respectively. When an

```
role Explorer{
  agent capability Moving(Location   loc);
  capability Explore {
     ......
     if(senseGarbage()){
         GarbageExploredEvent garbage_event =
            generateGarbageExploredEvent(myLocation);
       }
     }
}
role Cleaner{
  agent capability Moving(Location loc);
  capability Clean {
     garbageEvent = receiveMessage();
     myPlayer.Moving(garbageEvent.garbageLocation);
     cleanGarbage();
  }failed{
     generateCleanerFaildEvent(myLocation);
  }
  ......
}
organization CleaningGroup{
  roles Explorer, Cleaner;
  protocol CleaningProtocol(Explorer ex, Cleaner cn){
     run() on GarbageExploredEvent{
             ex.inform(cn);
       }
     }
}
agent RobotAgent play Explorer, Cleaner{
  capability Moving(Location loc){...}
  strategy CleanAdaptation{
     when(GarbageExploredEvent){
        deactivate(Explorer); enactOrActivate(Cleaner);
     }
     after Cleaner.Cleaning(){
           deactivate(Cleaner); activate(Explorer);
     }
     ......
  }
}
mas{
  groupA : CleaningGroup{.....}
  groupB : CleaningGroup{
     robotB1 : RobotAgent enact Explorer;
     robotB2 : RobotAgent enact Cleaner;
     robotB3 : RobotAgent enact Explorer, Cleaner;
     createPotocol(CleaningProtocol, robotA, robotB);
  }
}
```

Fig. 6. Code fragments of CRASM

explorer has sensed garbage at current location, it generates event as *GarbageExploredEvent* which will be sent to its players, the protocol which it participates and the group it belongs to.

An organization *CleaningGroup* is defined to specify the relationship between Explorer and Cleaner roles as *CleaningProtocol*, which is triggered by the *GarbageExploredEvent* from Explorer. A type of RobotAgent is specified to play both Explorer and Cleaner roles with *Moving* capability. And *CleaningAdaptation* strategy is defined to specify how the agent to change its roles. That is, when *GarbageExploredEvent* is generated, the agent will deactivate *Explorer* and *enactOrAcitivate Cleaner* to clean the garbage; when *CleanerFailedEvent* is generated the agent will deact *Cleaner* as it loses the clean capability and activate *Explorer* to continue exploring; and when the garbage is successfully cleaned, the agent will deactivate *Cleaner* and activate *Explorer*. Note that this norm is triggered only when the agent play more than one roles.

In the initialization file of the CRASM, two *CleaningGroup* are created by the programmer: groupA and groupB. The protocol of the agents can be created either by the programmer or by the system. If there is not explicit initialization of the protocol, the group will initialize a protocol when the agent join it, and when the trigger is happened the group will search a suitable agent (with the special role) as a partner. Now, let's consider the scenario in *groupB* when *robotB2* fails to clean the garbage. The group manager (implemented with the infrastructure) will dynamically change the partner of *robotB1* from *robotB2* to *robotB3* which can also play *Cleaner* role.

5 Related Works

Recently, in agent-oriented software engineering field, the research on organization dynamics has received increasing attentions. Several approaches have been proposed to support such dynamics at different stages in the life-cycle of MAS, ranging from design and specification to analysis and runtime [8]. There are three ways to implement dynamic MAS.

Firstly, adaptation or reorganization algorithms are designed for agents, with which they can change their behaviors and reorganize the structure (e.g. OMACS [3]). In this approach, agents have to achieve its personal goals, contribute the organization goals and manage organization dynamics, which makes the agents' internals complex and hard to implement.

Secondly, organization middleware or infrastructure developed to manage and reorganize agents (e.g. MACODO [8], Moise+[9], powerJade [13], Janus [14]). The organization middleware or infrastructure can separate the management of dynamics from the agents, and simplify the comprehension and specification of the dynamics. However, there is usually a lock of unify syntax and semantics for agents and organizations, which affects the readability and expressiveness and limit the potential of organization abstraction.

Lastly, programming languages designed to support the dynamics with organization concepts and mechanisms. Recently organization concepts started to appear in some programming languages (e.g. 2OPL[11]), the dynamics are not widely and thoroughly considered yet. However, current approaches mainly focus on the construction of the MAS based on organization abstraction, but mechanisms to deal with the dynamics of the systems are lacked or inadequate. While our approach concentrate on dynamics in MAS, with aim to support programming construct and mechanisms for the agent adaptation and structure flexibility.

6 Conclusions and Future Researches

As the dynamics are considered as an intrinsic property of MAS, it is important to provide abstraction and mechanisms to support the development of such properties. This paper analyzes different dynamics in MAS from behavior and structure perspectives at various levels: agent level, intra-organization and inter-organizations. We propose an organization-based programming approach for implementing dynamic MAS in terms of a core programming concept model. Role enactment and delegation mechanisms are introduced to implement the dynamics focusing on agent adaptation and social reorganization. A programming language OBLPD is given based on such model and mechanisms. The syntax of language facilities organizations, roles and agents is defined and its semantics is informal discussed.

Different from existing researches, our approach focuses on the programming approach for MAS based on organization theory, especially for the dynamic issues. One of advantages with our programming approach is that with high level organization concepts as first-class abstraction, the gap between models and codes are bridged. Moreover, the role enactment and delegation mechanisms facilitate the implementation of agent adaptation and social reorganization. At last, it makes the program more readability and comprehensibility with the unify syntax and semantics for organizations and agents in one programming language.

Our ongoing researches include: (1) the formal semantics of OBLPD as a basic for the implementation of an interpreter; (2) execution model and an interpreter is developed for OBLPD. Moreover, as the current vision of OBLPD only support the social reorganization within intra-organization, the following work to provide language constructs and infrastructure for the inter-organization reorganization.

Acknowledgment. The authors gratefully acknowledge financial support from Natural Science Foundation of China under granted number 61070034 and 90818028, Program for New Century Excellent Talents in University, and 973 Program of China under granted number 2011CB302601.

References

1. Jennings, N.R.: An agent-based approach for building complex software systems. Communication of ACM 44(4), 35–41 (2001)
2. Dignum, V.: The Role of Organization in Agent Systems. In: Dignum, V. (ed.) Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models, pp. 1–16. IGI Global (2009)
3. DeLoach, S.A., Oyenan, W.H., Matson, E.: A capabilities-based model for adaptive organizations. Autonomous Agents and Multi-Agent Systems 16(1), 13–56 (2008)

4. Ferber, J., Gutknecht, O., Michel, F.: From Agents to Organizations: an Organizational View of Multi-Agent Systems. In: Giorgini, P., Müller, J.P., Odell, J.J. (eds.) AOSE 2003. LNCS, vol. 2935, pp. 214–230. Springer, Heidelberg (2004)
5. Costa, A.C.R., Dimuro, G.P.: A Minimal Dynamical MAS Organization Model. In: Dignum, V. (ed.) Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models, pp. 419–445. IGI Global (2009)
6. Dignum, V.: Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models (2009)
7. Bordini, R.H., Dastani, M., Dix, J., El Fallah Seghrouchni, A. (eds.): Multi-Agent Programming: Languages, Platforms and Applications. Springer, Heidelberg (2005)
8. Weyns, D., Heasevoets, R., Helleboogh, A.: The MACODO Organization Model for Context-driven Dynamic Agent Organizations. ACM Transactions on Autonomous and Adaptive Systems 5(4), 1–29 (2010)
9. Hübner, J.F., Sichman, J.S., Böissier, O.: Developing organised multiagent systems using the MOISE+ model: programming issues at the system and agent levels. International Journal of Agent-Oriented Software Engineering 1(3/4), 370–395 (2007)
10. Garcia, E., Argente, E., Giret, A., Botti, V.: Issues for Organizational Multiagent Systems Development. In: Jung, Michel, Ricci, Petta (eds.) 6th Int. Workshop on AT2AI-6 Working Notes, From Agent Theory to Agent Implementation, AAMAS 2008, Estoril, Portugal, EU, May 13 (2008)
11. Tinnemeier, N.A.M.: Organizing Agent Organizations: Syntax and Operational Semantics of an Organization-Oriented Programming Language. SIKS Dissertation Series 2011(2), Utrecht University (2011)
12. Ghidini, C., Hirsh, B., Fisher, M.: Programming group computations. In: Proceedings of the First European Workshop on Multi-Agent System, EUMAS (2003)
13. Baldoni, M., Boella, G., Genovese, V., Grenna, R., van der Torre, L.: How to Program Organizations and Roles in the JADE Framework. In: Bergmann, R., Lindemann, G., Kirn, S., Pěchouček, M. (eds.) MATES 2008. LNCS (LNAI), vol. 5244, pp. 25–36. Springer, Heidelberg (2008)
14. Gaud, N., Galland, S., Hilaire, V., Koukam, A.: An Organisational Platform for Holonic and Multiagent Systems. In: Hindriks, K.V., Pokahr, A., Sardina, S. (eds.) ProMAS 2008. LNCS, vol. 5442, pp. 104–119. Springer, Heidelberg (2009)

Capability Modeling of Knowledge-Based Agents for Commonsense Knowledge Integration[⋆]

Yen-Ling Kuo and Jane Yung-jen Hsu

Department of Computer Science and Information Engineering
National Taiwan University
yjhsu@csie.ntu.edu.tw

Abstract. Robust intelligent systems require commonsense knowledge. While significant progress has been made in building large commonsense knowledge bases, they are intrinsically incomplete. It is difficult to combine multiple knowledge bases due to their different choices of representation and inference mechanisms, thereby limiting users to one knowledge base and its reasonable methods for any specific task. This paper presents a multi-agent framework for commonsense knowledge integration, and proposes an approach to capability modeling of knowledge bases without a common ontology. The proposed capability model provides a general description of large heterogeneous knowledge bases, such that contents accessible by the knowledge-based agents may be matched up against specific requests. The concept correlation matrix of a knowledge base is transformed into a k-dimensional vector space using low-rank approximation for dimensionality reduction. Experiments are performed with the matchmaking mechanism for commonsense knowledge integration framework using the capability models of ConceptNet, WordNet, and Wikipedia. In the user study, the matchmaking results are compared with the ranked lists produced by online users to show that over 85% of them are accurate and have positive correlation with the user-produced ranked lists.

Keywords: multi-agent system, common sense, commonsense knowledge integration, capability model, agent description.

1 Introduction

Commonsense knowledge is an essential element for building intelligent systems. It enables computers to infer new facts or to perform actions with commonsense about the world so that applications can interact with humans intelligently. Also, it helps break the software brittleness bottleneck by taking place whenever the domain-specific knowledge fails.

For thirty years, many projects [5,14] have been devoted to the collection of common sense knowledge. While significant progress has been made in building large commonsense knowledge bases (KBs), e.g. Cyc [5] and ConceptNet [3], such KBs are still

[⋆] This research was supported by a grant from the National Science Council of Taiwan, NSC 99-2221-E-002-139-MY3.

intrinsically incomplete or even inconsistent. Therefore, it may be necessary for an intelligent system to reason with multiple commonsense KBs at the same time in order to meet the specific goals of an applications. Example 1 demonstrates a sample scenario from an application developer's viewpoint.

Example 1. Goal-oriented search engine [6]
The goal-oriented search engine is a search engine interface that uses commonsense reasoning to turn search goals specified by the user in some natural language description into effective query terms. When processing an input like "my golden retriever has a cough", it should identify that the user's search goal is to find a veterinarian/remedy for his/her dog. Unfortunately, systems developed using ConceptNet alone will be unable to find the answers, as it does not contain any knowledge about golden retrievers. Alternatively, an intelligent system can first consult the lexicon database WordNet to find a generalization of the concept, e.g. "golden retriever is a kind of dog". It is then possible to find "veterinarian" by reasoning from "dog" and "cough" in the ConceptNet semantic network.

In summary, enabling applications to reason across multiple knowledge bases improves their goal-achieving behaviors. In the dynamic world today, it is especially important that applications should be equipped with up-to-date knowledge to interact with their users. Multi-agent systems (MAS) provide a powerful paradigm to facilitate application building when multiple heterogeneous knowledge representations and reasoning are required [8]. In this paper, we present a multi-agent framework for commonsense knowledge integration, which is especially effective in supporting the reasoning with knowledge on different commonsense domains from heterogeneous KBs. It can reflect upon and improve its behavior when KBs are expanded.

One major challenge in such a multi-agent system is to describe the capability of each knowledge-based agent, given that multiple representations are used, and without common ontology. This paper proposes a distributed capability model which is built by transforming the concepts contained in the KBs into a k-dimensional space using low-rank approximation. Requests from applications are evaluated based on vector similarity to decide which KB to match up with.

This paper starts with a brief overview of commonsense knowledge representations, collection, and reasoning methods. A specific multi-agent system for common sense knowledge integration is then proposed, along with the introduction of our capability model and capability evaluation procedure. We then present our experimental setup of the capability models in ConceptNet, WordNet, and Wikipedia. The matching results are evaluated by comparing them with matchmakings made by online users to verify their correctness and relevance.

2 Commonsense Knowledge Bases

Due to their different design decisions, the current commonsense KBs are heterogeneous and inconsistent in representations, quantity, quality, and means of access. Therefore, a system to integrate different commonsense knowledge bases are needed for the benefit of application building. We will review these elements and illustrate their heterogeneity in this section.

2.1 Knowledge Representation

When building applications, developers may choose commonsense KBs with different knowledge representations to serve their specific requirements. The two most prominent representations for common sense are formal logical framework and semantic network, used by Cyc [5] and ConceptNet [3] respectively.

The formal logical framework is appropriate for representing precise and unambiguous facts, which facilitates the automation of commonsense reasoning. On the other hand, the semantic network is more flexible in incorporating new knowledge and contextual reasoning. It represents all sentences in the corpus as a directed graph. The nodes of this graph are *concepts*, and its labeled edges are *relations* between two concepts. For example,

- `UsedFor(a, b)`, e.g. [Spoon] is used for [eating].
- `IsA(a, b)`, e.g. [Dog] is an [animal].

2.2 Commonsense Knowledge Collection

Codifying millions of pieces of human knowledge into machine usable forms has proved to be time-consuming and expensive. While techniques for mining knowledge from corpus or web pages have been developed [12,2], it is difficult for computers to discover the commonsense knowledge underlying a text. Therefore, sources of commonsense knowledge are still majorly reliant on experts or the general public.

Expert-Developed Knowledge Bases. A team of knowledge engineers encode common sense into the KBs. This approach ensures the highest quality of data. However, it is expensive, time-consuming, and difficult to scale up.

WordNet [7] is a highly structured database of words, which are carefully crafted by expert linguists. Synonyms are grouped into *synsets* and are conneted with each other by relations. It has been successfully used in a variety of applications to measure the proximity of words.

Started in 1984, the Cyc project [5] carefully crafted knowledge into CycL, a rigorous logic-based language to ensure its correctness. Now, the OpenCyc 2.0 ontology contains hundreds of thousands of terms with millions of assertions relating the terms to each other.

Collaboratively-Built Knowledge Bases. The success of crowd-sourcing approaches led many research groups to start to use websites or games to appeal to online users for contribution. However, the knowledge collected from these sources is highly dependent on the performance of users, which also makes the KBs incomplete and inconsistent.

The Open Mind Common Sense (OMCS) project at MIT [15] has collected over a million sentences in multiple language. The English and Portuguese corpora were collected from over 15,000 contributors at the OMCS website[1] within the span of 10 years. With innovations in community-based social games, the up-to-date knowledge in

[1] http://openmind.media.mit.edu/

the Chinese ConceptNet was successfully collected and verified via question-answering between players [4].

Wikipedia[2] is one of the world's largest KBs of both encyclopaedic knowledge and commonsense knowledge. The knowledge is stored in documents connected with page links. It also provides a taxonomy by its categories, where articles can be assigned to one or more categories. The unstructured documents are thus put into a network of categories.

2.3 Commonsense Reasoning

It is straighforward to equip a variety of applications with common sense by querying the KBs using APIs. For example, one may ask if a specific assertion is present in the corpus. Furthermore, KBs with different representations may call for different reasoning methods. The semantic network is suitable for finding related and similar concepts. Measures of similarity/relatedness quantify how much two concepts are alike/related. Both relatedness and similarity measures are developed for WordNet [10], ConceptNet [17], and Wikipedia [18] so that it is possible to reason in large and noisy semantic networks. The logic framework, on the other hand, uses deduction and theorem prover to reason new facts. Heuristics are often applied to logic-based reasoning for better efficiency. OpenCyc[3] also released its planner for reasoning out actions and events with its rules and assertions.

3 Commonsense Knowledge Integration

In response to the emergence of heterogeneous commonsense knowledge sources and the different reasoning methods using them, a system for commonsense knowledge integration should utilize the reasoning abilities of KBs while maitaining their own autonomies.

3.1 Multi-agent Framework

A multi-agent system is fitting for this open and dynamic environment to achieve the interoperation of commonsense KBs. We devised a common sense integration framework (see figure 1) to provide integrated reasoning service for application to use.

Instead of integrating knowledge sources into a single ontology, the key idea of this framework is to treat knowledge as resources that different reasoning methods can access. The integration of knowledge is achieved via matchmaking and composition of different reasoning methods. Following are the detailed descriptions of the agents in figure 1.

– **KB agent:** A KB agent is responsible for a commonsense KB. It monitors knowledge in the KB and is equipped with behaviors that make the KB complete. If a matchmaker asks for a KB agent's capability in handling a request, the KB agent will answer the query based on its belief (i.e. the capability model) of the KB.

[2] http://www.wikipedia.org
[3] http://www.opencyc.org/

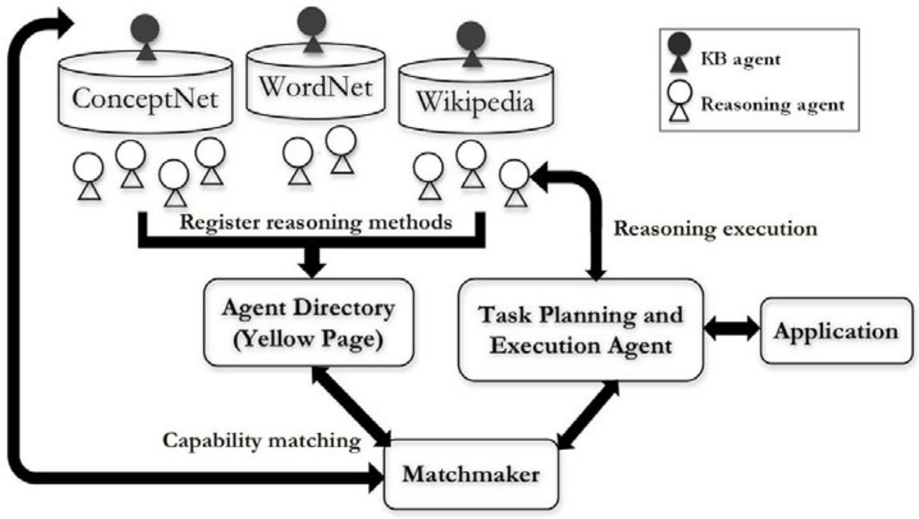


Fig. 1. The multi-agent framework for commonsense knowledge integration

- **Reasoning agent:** There are multiple reasoning agents for a KB. Each agent performs an atomic reasoning task of the KB. Once the KB agent updates the KB, the quality of reasoning results is also improved.
- **Agent directory:** An agent directory records the types of reasoning agents in the system and the KBs they can access.
- **Matchmaker:** A matchmaker forms sample queries to KB agents to check which knowledge bases are able to handle its request. Then, it returns a list of candidate reasoning agents which are sorted by their capability to the task planning agent.
- **User agent:** A user agent represents a specific application. It states the needs of an application to the task planning agent and communicates results with users.
- **Task planning and execution agent:** Based on the requirements of a user agent, a task planning and execution agent decides whether to form a composite task or to send the request directly to the matched reasoning agents.

3.2 Challenges

There are three challenges in realizing such a framework. The major challenge is the capability matching of large and heterogeneous KBs. Unlike service matchmaking in service-oriented computing, there is no common ontology for applications to identify the capability of commonsense KBs. The second one is to maintain good reasoning quality for reasoning agents. It requires the complete KBs. The third challenge is the composition of reasoning methods. The dependency of reasoning agents should be specified to facilitate composite reasoning.

The second and third challenges have been studied in knowledge acquisition and MAS community respectively and are not in the scope of our discussion. In this paper, we focus on the first challenge, the capability model of KB agents. Instead of querying every KB at the same time (which causes a very high overhead), we use a vector space to model the belief of KB agents.

4 Related Work

Many approaches to integrating knowledge and describing capability of agents has been proposed in MAS. The differences between these systems and commonsense knowledge integration are discussed in this section.

Knowledge Integration Systems. MAS has been proposed to integrate and reuse information in web environment for many years. Most of them opt to combine loosely coupled sources into integrated wholes, such as KRAFT [11] and KSNet [16]. InfoSleuth[9] is another system for integrating heterogeneous information sources. An application-specific ontology is used as a basic ontology to locate different queries. However, these approaches are not feasible for commonsense knowledge integration, because there is no common ontology and exists conflicts in different commonsense KBs such that it may results in errors if we combine them into a single one.

Capability Descriptions in Matchmaking. One of the most important issues in matchmaking is to describe agents' capability. Several languages are defined to represent agent capabilities, e.g. LARKS (Language for Advertisement and Request for Knowledge Sharing) [19], WSDL (Web Service Description Language)[4], etc. With the representations of capabilities, attributes can be matched using constraint satisfaction or semantic similarity [13]. However, this process usually requires ontologies to facilitate generalizing service needs and calculating semantic similarities. Also, the description languages cannot reflect the differences resulted from the knowledge the agents can access. For example, even if the agents in WordNet and ConceptNet provide the same reasoning method, e.g. finding related concepts, they may return different results because of different knowledge coverage.

5 Capability Model

The major challenge in finding a reasoning agents to handle a query is the variety of domains contained in the KBs. Since the capability of a reasoning agent is reflect on the coverage of knowledge base, each KB agent should be equipped with a compact model of its KB, and be able to inform the capable reasoning agents to provide the required knowledge quickly. In this section, we introduce the proposed capability model for KB agents and an algorithm to evaluate requests.

5.1 Representation of Capability Model

In most KBs, we can argue that they can be determined by or associated with a small set of eigenconcepts. Therefore, we can use the eigenconcepts as the capability model of KB agents.

[4] http://www.w3.org/TR/wsdl20/

Correlation Matrix. Consider a commonsense knowledge base $\mathcal{K}$, a sentence is always represented as a triple, $(c_i, \textit{relation}, c_j)$, where c_i and c_j are concepts. Each concept can form a vector of related concepts $\boldsymbol{v}$, where the j_{th} component of the vector, v_j, is the number of triples containing c_i and c_j in $\mathcal{K}$. Thus, we can construct an $n \times n$ *correlation matrix*, where n is the number of concepts in $\mathcal{K}$. The *correlation matrix* reflects the capabilities of a KB agent by describing the relatedness of concepts in the KB it can access.

Knowledge Represented by Eigenconcepts. In order to evaluate the query quickly, we need to reduce the dimension of a correlation matrix without losing its capability. We re-formulate the high-dimensional concept correlation matrix into a k-dimensional eigenspace, where $k \ll n$. A concept in $\mathcal{K}$ is then represented as a vector in a k-dimensional space spanned by *eigenconcepts*. It is the responsibility of a KB agent to identify the dimensions that are useful in summarizing the KB while truncating dimensions that are less relevant.

This process is also referred to as "dimensionality reduction." *Low-rank approximation* is an approach to achieving dimensionality reduction. Given $m \times n$ matrix A, we aim to approximate it by a matrix of rank k, which is much smaller than m and n. In this paper, we use singular value decomposition (SVD) to achieve low-rank approximation.

5.2 Notations

In what follows, let A be the correlation matrix constructed from a commonsense KB. We use $U\Sigma U^T$ to denote the SVD of A since A is a symmetric matrix. The diagonal entries of Σ are singular values of A and denoted as σ_i. Similarly, we use $A_k = U_k \Sigma_k U_k^T$ to denote the best rank k approximation to A and use $A^{(c)}$ to refer to the column of concept c in A. The projection of the concept c in A onto the first k column of U is denoted as a_c. Finally, we denote the 2-norm of a vector a_c by $\|a_c\|$ and the 2-norm of a matrix A by $\|A\|_2$.

5.3 Capability Modeling: Choosing the Best *k*

In order to create a capability model for describing a KB, we apply SVD on the correlation matrix: $A \approx A_k = U_k \Sigma_k U_k^T$, where

- U_k: a $n \times k$ matrix that relates concepts to eigenconcepts
- Σ_k: a $k \times k$ diagonal matrix of singular values σ_i that assigns weights to each eigenconcept.

The best rank k is chosen in algorithm 1 so that the resulted space is the best approximation to describe the correlation of concepts in a KB.

The confidence of choosing k by $\sigma_k - \sigma_{k+1} \leq \theta$ is motivated by the Eckart-Young theorem.

Theorem 1. *Eckart-Young theorem [1]*
Let $A = U\Sigma V^T = U\textit{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0)V^T$. *For any* k *with* $0 \leq k \leq r$,

$$\|A - A_k\|_2 = \sigma_{k+1} \tag{1}$$

Algorithm 1. Capability Model $(\mathcal{K})$

```
Require: A commonsense knowledge base $\mathcal{K}$
Ensure: A set of eigenvectors that span the capability model $U_k$ and the projection $A_k$ of concepts in $\mathcal{K}$
1: Build a correlation matrix $A$ from triples in $\mathcal{K}$
2: Apply SVD on the correlation matrix: $A = U\Sigma U^T$
3: Choose the largest $k$ such that $\sigma_k - \sigma_{k+1} \leq \theta$
   where $\theta$ is a small constant
4: Represent concepts in the $k$-dimensional eigenspace:
   $A_k = U_k^T A$
5: return $U_k$ and $A_k$
```

Theorem 1 implies that $\sigma_k - \sigma_{k+1}$ is the key factor of incorporating the k_{th} column of U into the compact representation of capability model. If we set θ small enough, we are returned with the representative eigenconcepts so that the error between real and modeled capability of KB is within σ_{k+1}.

5.4 Capability Evaluation for Matchmaking

With the capability model, the KB agent is now able to answer whether it can handle a request from applications. Here, we introduce two kinds of concept vectors.

- **Knowledge-based concept vector**, a_c: The vector of concept c is $a_c = U_k^T A^{(c)}$. It represents the supply of a KB.
- **Application-based concept vector**, v: For any application that would like to incorporate commonsense knowledge, we can find a corpus to describe the application, e.g. using google snippets to describe the application that requires up-to-date news. A concept can be represented by a vector v constructed from the co-occurred concepts in the corpus. This vector states the need of an application.

We can also project v onto the capability model, i.e. $v_k = U_k^T v$. The capability of KB to handle the request is correspondent to the similarity between a_c and v_k. The algorithm to evaluate the capability of a KB is illustrated in algorithm 2. The evaluation score returned by algorithm 2 is defined by the cosine similarity of a_c and v_k.

Algorithm 2. Evaluate Capability (U_k, A_k, c)

```
Require: A capability model $U_k$, a projection $A_k$ of concepts in $\mathcal{K}$, and a testing concept $c$
Ensure: An evaluation score $sim$ that indicates the capability of $\mathcal{K}$ to handle a request
1: Construct term-frequency vector $v$ of concept $c$
   from the corpus of application, e.g. google snippets
2: Project $v$ onto the capability model: $v_k = U_k^T v$
3: Get vector of $c$ from $A_k$: $a_c$ = column of $c$ in $A_k$
4: Calculate similarity of $a_c$ and $v_k$: $sim = \frac{a_c \cdot v_k}{\|a_c\|\|v_k\|}$
5: return $sim$
```

Once the matchmaker gets the request from the task planning agent, algorithm 3 is used for matching suitable KBs. The matchmaker asks every KB agent to get the evaluation score of each KB respectively, and then sort the KBs based on their evaluation scores in descending order. Finally, the KBs with evaluation scores > 0.5 are returned to the task planning agent. The task planning agent will send requests to the reasoning agents of the matched KBs to get answers.

Algorithm 3. Matchmaking $(\mathcal{K}, c)$

```
Require: A set of available KBs $\mathcal{K}$ and a query concept $c$
Ensure: A sorted list of matched KBs $L$
 1: Construct application-based vector $v$
 2: for $K_i$ in $\mathcal{K}$ do
 3:     $sim_i$ = Evaluate Capability $(U_{ik}, A_{ik}, c, v)$
 4: end for
 5: for $K_i$ in $\mathcal{K}$ do
 6:     if $sim_i > 0.5$ then
 7:         Add KB $K_i$ into $L$
 8:     end if
 9: end for
10: Sort $L$ in descending order based on the evaluation scores of KBs
11: return $L$
```

6 Evaluation

In order to evaluate the proposed capability model, we incorporated the model into the matchmaking in the commonsense knowledge integration framework. The matchmaking results produced by our approach are compared with the matches made by online users.

6.1 Experimental Setup

The commonsense knowledge integration framework is implemented using JADE (Java Agent DEvelopment Framework). ConceptNet, WordNet, and Wikipedia are chosen to be our experimental KBs. The number of concepts in each KB are shown in table 1. Despite the large number of concepts in these KBs, they are quite inconsistent. For example, the overlap of ConceptNet and WordNet is only 4.79%.

Table 1. Statistics of knowledge bases

Knowledge base	Number of concepts
ConceptNet	274,477
WordNet	128,391
Wikipedia	3,440,143

Three reasoning agents are designed for each KB.

- **Topic agent:** Find the topics of a given concept, e.g. “computer” and “computer science” are topics of “cpu”.
- **Related agent:** Find related concepts of a given concept, e.g. “teacher”, “student”, “blackboard” are related concepts of “school”.
- **Similarity agent:** Calculate the similarity score of two concepts, e.g. the similarity of “cow” and “horse” is 0.889 in ConceptNet.

In this experiment, snippets from google’s search results are selected as our corpus to create the application-based concept vector v in the matchmaking. Related concepts of a specific query term are returned by “related agent” of the matched KB.

Collecting Matchmaking Lists from Online Users. In order to evaluate the matchmaking results, we incorporated the proposed capability model into the system. The matchmaking results produced by our approach are compared with the matches made by online users. In order to evaluate the matchmaking results, we collected a user-produeced matching list as the ground truth. The list was collected from workers on Amazon Mechanical Turk[5], the largest crowd-sourcing market in the world. First, we uniformly sampled 100 concepts from each KB as input queries from applications. About half of these concepts were found in at least two KBs. Every worker in our task was given a concept and three web pages containing related concepts of that concept. The web pages are generated from the three KBs. What the worker required to do was to browse the web pages and rate the relatedness of the web page and the given concept. Every concept was rated by 3 workers to increase its validity. In this process, we created a ranking of the KBs for each concept according to their relatedness with the query concept. The KB ranked first was considered as the matched KB for finding related concepts.

Building a Correlation Matrix. Since the correlations of concepts are in different forms for different KBs, we had to use a different method to create the triples required by KB agents. Triples in ConceptNet are its assertions; triples in WordNet are its words and the relations between words; triples in Wikipedia are the pages and their links with other pages. The θ in algorithm 1 is set to 0.1 for every KB agent.

6.2 Experimental Result

In this experiment, our matchmaking mechanism used the evaluation scores of the query concept in each KB to create a ranking of KBs for each query concept. Accuracy and rank correlation are two measures we used for evaluating matchmaking correctness and relevance against the user-produced ranking.

Correctness. Intuitively, the correctness of a matchmaking mechanism is whether it can find user’s desired result. If the result ranked first in the matchmaking is also the

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

KB ranked first by online users, we marked it as a "match". The accuracy is thus defined as the proportion of matches to query terms:

$$accuracy = \frac{\#\ of\ matches}{\#\ of\ query\ concepts}$$

For all sampled concepts, the accuracy is 93.32%. If we only consider the concepts that can be found in at least two knowledge bases, the accuracy is 87.67%, which is slightly lower than the former case. This drop of accuracy appears in the concepts with polysemy. For such concepts, it is likely that the application-based vector may not be aligned with the user's goal, therefore causing errors in matching KBs. It indicates that with application-based vectors corretly representing an applications' goal, the matchmaker can produce an accurate match using the proposed capability model.

Relevance. We also compare the rank correlation of the matchmaking produced by the matchmaker and online users. The relevance of our matchmaking results and ranked list produced by online users were measured by Kendall's τ rank correlation coefficient τ_B. If the two ranked lists are in perfect agreement, τ_B is 1; if they are perfect disagreement, τ_B is -1. In our experiment, the average τ_B are 0.818 and 0.695 respectively for all query concepts and query concepts in at least two knowledge bases respectively. A positive correlation was found between the two lists. This result suggests that the capability model corresponds to our understanding of the knowledge bases. With our capability model, we can represent knowledge bases well and make judgment that is relevant to human intuition.

7 Conclusion

This paper proposed a distributed capability model for large and heterogeneous KBs. Instead of integrating KBs into a single one, the model has the following features:

- it can express multiple KBs without a common ontology.
- it can update the belief of agents in a dynamic environment with frequently updated knowledge.

Using the capability model created from concept correlation matrix, we are able to identify differences, e.g. different senses of concepts, for KBs with no common ontology. The capabilities of agents are described based on the queries they can answer. Experiments have been conducted to match KB for finding related concepts. Compared to the user-produced ranking lists, our proposed method shows a high accuracy of 87.6% and a positive rank coefficient. With the capability model of KBs, many complex reasoning tasks can then be built on top of the commonsense knowledge integration system that matches relevant reasoning agents to applications.

References

1. Eckart, C., Young, G.: The approximation of one matrix by another of lower rank. Psychometrika 1(3), 211–218 (1936)
2. Etzioni, O., Cafarella, M., Downey, D., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Methods for domain-independent information extraction from the web: an experimental comparison. In: Proceedings of AAAI 2004 (2004)

3. Havasi, C., Speer, R., Alonso, J.: ConceptNet 3: A flexible, multilingual semantic network for common sense knowledge. In: Recent Advances in Natural Language Processing, Borovets, Bulgaria (September 2007)
4. Kuo, Y.L., Lee, J.C., Chiang, K.Y., Wang, R., Shen, E., Chan, C.W., Hsu, J.Y.j.: Community-based game design: experiments on social games for commonsense data collection. In: Proceedings of the ACM SIGKDD Workshop on Human Computation (2009)
5. Lenat, D.B.: CYC: A large-scale investment in knowledge infrastructure. Communications of the ACM 38(11), 33–38 (1995)
6. Liu, H., Lieberman, H., Selker, T.: GOOSE: A Goal-Oriented Search Engine with Commonsense. In: De Bra, P., Brusilovsky, P., Conejo, R. (eds.) AH 2002. LNCS, vol. 2347, pp. 253–263. Springer, Heidelberg (2002)
7. Miller, G.A.: WordNet: A lexical database for english. Communications of the ACM 38(11), 39–41 (1995)
8. Minsky, M.: The Society of Mind. Simon and Schuster (1988)
9. Nodine, M., Fowler, J., Ksiezyk, T., Perry, B., Taylor, M., Unruh, A.: Active information gathering in InfoSleuth. International Journal of Cooperative Information Systems 9(1/2), 3–28 (2000)
10. Pedersen, T., Patwardhan, S., Michelizzi, J.: WordNet: Similarity - measuring the relatedness of concepts. In: Proceedings of AAAI 2004 (2004)
11. Preece, A., Hui, K., Gray, A., Bench-capon, T., Joes, D., Cui, Z.: The KRAFT architecture for knowledge fusion and transformation. In: Proceedings of the 19th SGES International Conference on Knowledge-Based Systems and Applied Artificial Intelligence (1999)
12. Schubert, L., Tong, M.: Extracting and evaluating general world knowledge from the brown corpus. In: Proceedings of the HLT-NAACL Workshop on Text Meaning (2003)
13. Singh, M.P., Huhns, M.N.: Service-Oriented Computing: Semantics, Processes, Agents. Wiley (2005)
14. Singh, P.: The public acquisition of commonsense knowledge. In: Proceedings of AAAI Spring Symposium (2002)
15. Singh, P., Lin, T., Mueller, E.T., Lim, G., Perkins, T., Zhu, W.L.: Open Mind Common Sense: Knowledge Acquisition from the General Public. In: Meersman, R., Tari, Z. (eds.) CoopIS 2002, DOA 2002, and ODBASE 2002. LNCS, vol. 2519, pp. 1223–1237. Springer, Heidelberg (2002)
16. Smirnov, A.V., Pashkin, M., Chilov, N., Levashova, T.: Multi-Agent Architecture for Knowledge Fusion from Distributed Sources. In: Dunin-Keplicz, B., Nawarecki, E. (eds.) CEEMAS 2001. LNCS (LNAI), vol. 2296, pp. 293–302. Springer, Heidelberg (2002)
17. Speer, R., Havasi, C., Lieberman, H.: AnalogySpace: Reducing the dimensionality of common sense knowledge. In: Proceedings of AAAI 2008 (2008)
18. Strube, M., Ponzetto, S.P.: WikiRelate! computing semantic relatedness using wikipedia. In: Proceedings of the 21st National Conference on Artificial Intelligence, AAAI 2006 (2006)
19. Sycara, K., Widoff, S., Klusch, M., Lu, J.: Larks: Dynamic matchmaking among heterogeneous software agents in cyberspace. Autonomous Agents and Multi-Agent Systems 5, 173–203 (2002)

Producing Enactable Protocols in Artificial Agent Societies

George K. Lekeas[1], Christos Kloukinas[1], and Kostas Stathis[2]

[1] City University London, Northampton Square, London EC1V 0HB
{g.k.lekeas,c.kloukinas}@soi.city.ac.uk
[2] Royal Holloway, University of London, Egham Surrey TW20 0EX
kostas.stathis@cs.rhul.ac.uk

Abstract. This paper draws upon our previous work [7, 16] in which we proposed the organisation of services around the concept of *artificial agent societies* and presented a framework for representing roles and protocols using LTSs. The agent would apply for a role in the society, which would result in its participation in a number of protocols. We advocated the use of the *games-based metaphor* for describing the protocols and presented a framework for assessing the admission of the agent to the society on the basis of its *competence*. In this work we look at the subsequent question: *what information should the agent receive upon entry?*. We can not provide it with the full protocol because of security and overload issues. Therefore, we choose to only provide the actions pertinent to the protocols that the role the agent applied for participates in the society. We employ *branching bisimulation* for producing a protocol equivalent to the original one with all actions not involving the role translated into silent (τ) actions. However, this approach sometimes results in *non-enactable* protocols. In this case, we need to *repair* the protocol by adding the role in question as a recipient to certain protocol messages that were causing the problems. We present three different approaches for repairing protocols, depending on the number of messages from the original protocol they modify. The modified protocol is adopted as the final one and the agent is given the role automaton that is derived from the *branching bisimulation* process.

1 Introduction

Ubiquitous computing envisages objects with information processing and communication capabilities that will assist users in their daily tasks [18]. An example of such an setting could be a "user's personal assistant" (UPA) running on a Personal Digital Assistant (PDA). It will have knowledge of the user's timetable and assist him on the task(s) he has to carry out. The UPA could, for example, determine the location of the user and if he has to be at the airport in a short period of time order him a taxi.

In this example, the context and/or location in which the UPA is deployed play a significant role, as it will have to use a local taxi service or identify the products that are of interest to the user it represents. Furthermore, such an application will need to have a number of properties such as *autonomy* and *pro-activity*. This is the case as some of the taxi services the UPA is using might temporarily be down or not accepting a certain method of payment. On the other hand, it will need to be pro-active and be able to make decisions on what services to contact and what resources to use.

A paradigm that fits these requirements is this of a single *agent* or *Multi-Agent System* [12]. Such systems exhibit autonomy, reactivity and pro-activeness within the social context they operate (social ability). Moreover, in a *Multi-Agent System* no agent has complete ability to solve the problem, data is spread across the system, no agent can control the whole system and the computation is asynchronous; UPA will need the collaboration of other agents providing the required *services*.

In [16] we proposed the organisation of services around *artificial agent societies*. These would be *semi-open* in the sense of [3] (i.e. new members are accepted only on completion of a successful application). We should note here that *openness* is considered from a *membership* viewpoint and not, for example, from an *agent communication language* or *agent architecture* perspective.

The agent will choose the society to apply for membership on the basis of its *service needs* and apply for a *role* $\mathcal{R}$ in it. It will have to submit its *communication abilities* (i.e. the set of messages that it can utter/understand). These will be judged against the requirements of the protocols that $\mathcal{R}$ is participating in. The requirement is that the agent should be able to understand the messages that it can receive, as well as be able to utter the messages that $\mathcal{R}$ can send.

The representation of protocols is done using the games-based metaphor [15], which we extended to include different representations for the state of the game. This metaphor is not related to game theory; we are simply using the notion of game to represent the evolution of a protocol and not to quantify the agent strategies (which are, in general, unknown). The representation of the game as a protocol should allow for the representation of protocol states. These are described by the values of a number of properties we are interested in; e.g. who was the last player, who is the next one and what is the last move made in the game. In [15] *destructive assignment* has been used for this purpose, i.e., every time there is a change in the value of a property the old value is deleted and the new one is inserted. However, this is not the only option. *Situation Calculus* [9] can be used to represent the state as a sequence of actions forming a *situation*. On the other hand, if we are interested in a game that has concurrent moves, *Event Calculus* [8] could be used to describe the game state as events happening at specific time points. Finally, *commitments* [17] could be used.

Assuming that the agent in question is accepted into the society, there is a new issue of *what part of the protocol it should receive*. It could, of course, be provided by the full protocol but that might not always be easy e.g. for security or information overload problems. A procedure is, thus, needed for providing the agent only with the protocol information needed. This procedure should discard (hide from the agent) any parts of the protocol there is no need to know about as it is not involved in those. It should, also, ensure that there are no *structural* problems with the protocol that the agent receives, i.e., it is enactable. This means that any time the agent needs to take a decision as to what action to perform next, all information needed for making the decision is available to it.

The rest of the paper is structured as follows: Section 2 provides a quick overview of bisimulation, whereas Section 3 describes NetBill, our working example. We present our approach for creating the role automata in Section 4. In Section 4.1 we present three approaches for repairing non-enactable protocols . Finally, Section 5 discusses related work and we conclude the paper in Section 6.

2 Bisimulation

Bisimulation [10] is a way of minimising LTS on abstract (silent) actions while preserving the properties of the original model. It can be computed automatically without any human involvement.

Formally, it can be defined as [13]: A binary relation $\mathcal{R}$ on the states of a *Labelled Transition System* is *bisimulation* if whenever $s_1 \mathcal{R} s_2$:

$$\begin{aligned} &\text{for all } s_1' \text{with } s_1 \xrightarrow{\mu} s_1', \text{ there is } s_2' \text{ such that } s_2 \xrightarrow{\mu} s_2' \text{ and } s_1' \mathcal{R} s_2' \\ &\qquad (\forall s_1'.s_1 \xrightarrow{\mu} s_1' \Rightarrow \exists s_2' \colon s_2 \xrightarrow{\mu} s_2', s_1' \mathcal{R} s_2'); \\ &\text{the converse, on the transitions emanating from } s_2 \\ &\qquad (\forall s_2'.s_2 \xrightarrow{\mu} s_2' \Rightarrow \exists s_1' : s_1 \xrightarrow{\mu} s_1', s_2' \mathcal{R} s_1'). \end{aligned} \tag{1}$$

As the state of the protocol can be determined at any stage by the actions that have been already executed and the choice of what action to execute next, two equivalent (*bisimilar*) systems should represent the same evolution. This means that for any evolution of the first system (the original protocol), the second system (bisimulated model) should be able to evolve in the same way and any choice of actions in the first system should exist in the second system as well.

Any action the role in question is not involved in, either as a sender or amongst the recipients, is replaced by a silent (τ) action. Depending on how silent actions are treated, we distinguish between different types of bisimulation. The first option is to merge all silent actions with the first non-silent one, i.e. $\tau^\star\alpha \equiv \alpha$. This is a quick and easy way of dealing with τ actions, but it does not respect the structure of the protocol.

Branching bisimulation rectifies this by considering the structure of the LTS as well. Two LTS *P* and *Q* are branching bisimilar via a relationship *R* if: (i) their initial states are related via *R* and (ii) if r and s are related by *R* and $r \xrightarrow{\alpha} r'$, then either $\alpha = \tau$ or there exists a path $s \Rightarrow s_1 \xrightarrow{\alpha} s_2 \Rightarrow s'$ such that r and s_1, r' and s_2 as well as r' and s' are related by R.

The difference between the two types of bisimulation can be seen in Fig. 1.

Fig. 1a shows the original protocol with three roles, *A*, *B* and *C*. The protocol starts with role *A* sending message α to role *B* and afterwards role *C* has the option of sending *B* either *b* or *c*. Finally, role *A* can send role *B* either *d* or *e* but this choice is not independent of the previous steps. It depends on what message role *B* received. Fig. 1b shows the role automaton for role *A* by replacing any non-observable actions (i.e. actions that the role is not involved in as sender or recipient) with τ. The result of $\tau^\star\alpha$ bisimulation is shown in Fig. 1c. According to this, role *A* sends message α to role *B* and then it can send *B* either *d* or *e*.

However, this is not accurate. The choice of the second message is not with *A*, but depends on the choice that *C* made on the previous step. This is knowledge that role *A*

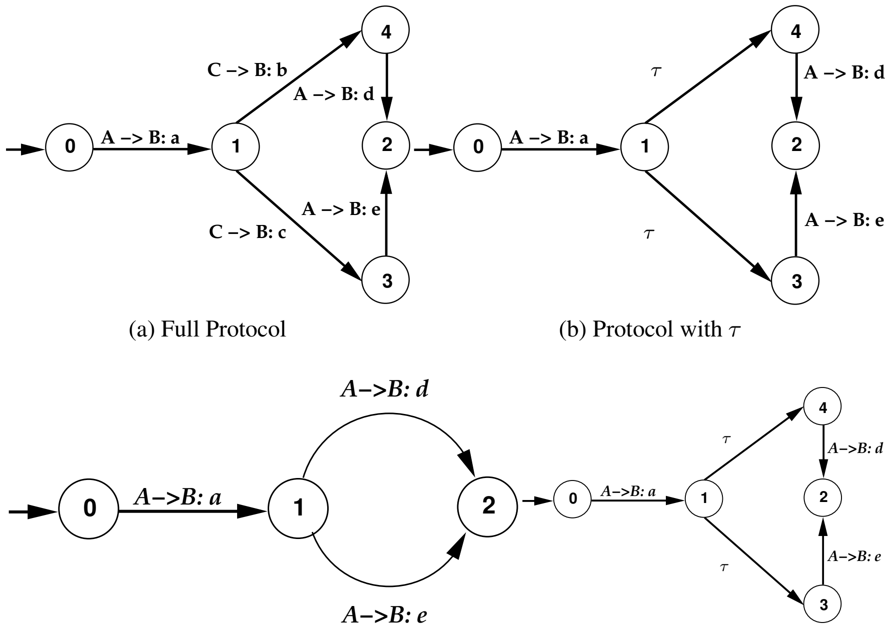


Fig. 1. Non-implementable protocol due to incomplete knowledge

does not have (in effect, it does not know neither whether role *C* acted nor what message it chose to send). Branching bisimulation in Fig. 1d takes this into consideration by keeping the two branches with τ actions and not discarding them, even if they do not represent role *A*'s knowledge. This means that for role *A* the protocol that it should receive should be the same as the original one with the τ actions, even if they do not represent role's knowledge.

3 The Netbill Protocol

In this section, we introduce a variation of the e-commerce protocol NetBill [6]. This can be used by a society that aims at allowing *merchants* to sell goods to *customers* and make use of *payment gateways* in order to collect payment. An agent wishing to enter a society where *Netbill* is available will have to apply for the role of customer, merchant or gateway depending on the goal it wishes to achieve when entering the society. In the original protocol, there are three roles - *customer* (c), *merchant* (m) and *gateway* (g)- and eight overall steps for a customer to purchase goods from a merchant and the merchant to process payment for the order through NetBill's gateway. These are depicted in Fig. 2 and are as follows:

– The customer requests a quote for some digital goods from a merchant - see the transition $(s_0, (c, rq, \{m\}), s_8)$, i.e., from state 0 to state 8, labelled as $(c, rq, \{m\})$.

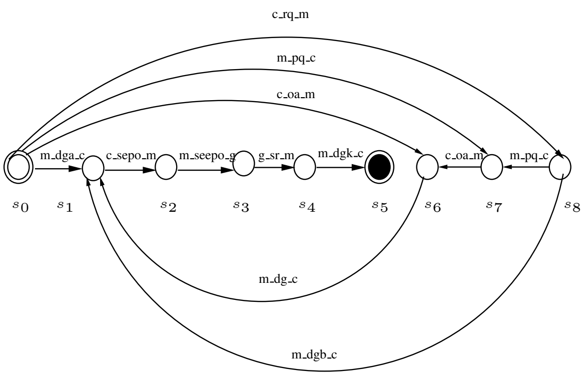


Fig. 2. A variant of the NetBill protocol

- The merchant provides a quote to the customer - (s_8, $(m, pq, \{c\})$, s_7).
- The customer accepts the quote made by the merchant - (s_7, $(c, oa, \{m\})$, s_6).
- The merchant proceeds to deliver the ordered goods encrypted with a key K - (s_6,$(m, dg, \{c\})$, s_1).
- The customer signs an *Electronic Purchase Order* (EPO) with the merchant - (s_1, $(c, sepo, \{m\})$, s_2).
- The merchant signs in its turn the EPO and sends it to the NetBill gateway - (s_2, $(m, ssepo, \{g\})$, s_3).
- The NetBill gateway internally checks the information on the EPO, transfers the money and ends by sending the merchant a receipt - (s_3,$(g, sr, \{m\})$,s_4).
- Finally, the merchant sends the customer the key needed to decrypt the goods it purchased - (s_4, $(m, dgk, \{c\})$, s_5).

We made the following additions to the original NetBill protocol to create one with branching structure so that we can illustrate problems when the agent has to make a decision but does not have all the information required:

- The merchant can now make a price quote directly - (s_0, $(m, pq, \{c\})$, s_7); e.g., in the case of a promotional offer.
- The merchant could select to deliver the goods as its first move - (s_0, $(m, dga, \{c\})$, s_1); e.g., when the customer has good credit and solid reputation with that merchant. In this case, the encryption method used in the delivery can be more relaxed than the normal one as the process involves a trusted customer.
- The customer might accept the merchant's quote directly - (s_0,$(c, oa,$ $\{m\})$, s_6); e.g., when the merchant is trusted or this is a recurring order.
- On reception of a quote request, the merchant can make the quote and ship the goods directly without waiting for a formal acceptance of the quote - (s_8, $(m, dgb, \{c\})$, s_1); e.g. when dealing with a trusted customer or a recurring order. The delivery and encryption method will have to be different again, as if it is a recurring order it will mean that the customer is low on stock for this particular item.

4 Producing the Final Enactable Protocol

In order to derive the role automata for each individual role involved in the protocol, the followed process is applied:

1. prepare the initial role automaton, i.e. the automaton we get from the original protocol automaton by replacing actions for which the role is neither the sender nor amongst the recipients by τ;
2. run *branching bisimulation* on the resulting automaton;
3. examine the resulting automaton for the presence of τ actions;
   (a) if τ actions exist but not make the protocol non-enactable, this is the protocol that the role receives;
   (b) if τ actions exist and they make the protocol non-enactable, then the protocol is repaired using one of the approaches in Section 4.1 and we start over with the *updated* protocol automaton.

By following this process, the protocol for the gateway role of the NetBill protocol is reduced to two transitions and three states, as shown in Fig. 3.

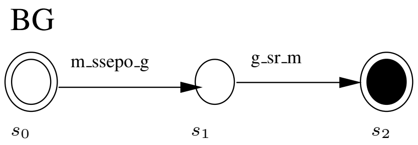


Fig. 3. The gateway role of NetBill after running branching bisimulation

The resulting protocol for the merchant agent would be the whole protocol, as the merchant is involved in all communications, while for the customer agent it would be the whole protocol except for the messages involving the gateway agent. The customer needs have no knowledge of these.

4.1 Protocol Repair

The NetBill protocol has been decomposed into role automata with no silent actions in them, as it is a well designed protocol. However, the breakdown of a protocol into its constituent roles need not always produce enactable specifications. If the resulting role automaton contains silent (τ) actions, then repair might be required. The repair process takes place at step 3b of Section 4 and consists of adding the role in question to the recipients of certain moves from the original protocol. The choice of the moves will depend on the algorithm we choose for the repair; the following sections describe three such algorithms starting with the one that will make most repairs to the protocol and finishing with the one making the least.

Updating All τ Actions. One approach is to find the equivalent states in the original protocol of the problematic states in the bisimulated one and add the role as a recipient to any messages originating from these states in the protocol. The algorithm is described in Listing 1.1.

```
// Game Protocol ⇒ GP, Role Protocol ⇒ RP
repair(GP, RP_badState, GP_role) {
  GP_class= equivalence_class(RP_badstate, GP);
  // Add role to the recipients of the moves of these states
  foreach (GP_state in GP_class)
    foreach (GP_tran from GP_state.transitions)
      GP_tran.move.receivers = GP_tran_move.receivers ∪ GP_role;
  }
```

Listing 1.1. Updating all silent transitions

This algorithm repairs the protocol by adding the extra information that was missing and was causing the occurrence of the τ move, i.e., adds the role in question to the recipients of the communication act. At the beginning, we calculate all states from the original protocol that are in the equivalence class of the originating state of the transition with the silent move in the bisimulated protocol. Once these are found, for every transition that starts from these states in the original protocol, the set of receivers is updated with the inclusion of the role whose automaton we are calculating.

Updating Frontier τ Actions. Another approach would be to repair a few transitions of the original protocol, those that start from any state in the original protocol that belongs to the same equivalence class as the original state of the silent action in the bisimulated protocol and finish in any of the states belonging to the same equivalence class as the end state of the same transition. The intuition here is that τ transitions within states of the same equivalent class will not be present in the resulting role automaton, so no repair is needed.

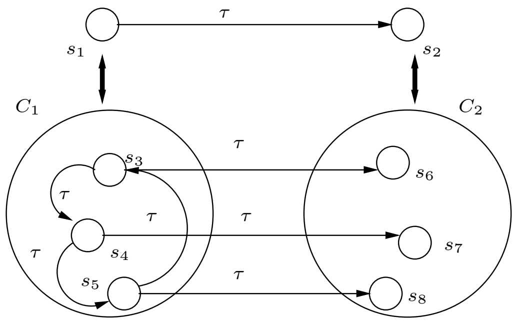


Fig. 4. Branching Bisimulation Equivalence Classes

In Fig. 4 after running branching bisimulation we have states s_1 and s_2 linked with a τ transition. However, as branching bisimulation is an equivalence relation placing states into equivalence classes, each of these two states would belong to an equivalence class of states from the original automaton. In this case, we have two equivalence classes $C_1 = \{s_3, s_4, s_5\}$ (represented by s_1) and $C_2 = \{s_6, s_7, s_8\}$ (represented by s_2). By looking at the transitions, we can see that the transitions from states belonging to class C_1 to states belonging to class C_2 are all τ transitions that need to be repaired. The benefit, however, in comparison with the approach described in Section 4.1 is that we do not repair any silent transitions *internal* to the class , i.e., the transitions from s_3 to s_4, s_4 to s_5 and s_5 to s_3.

The algorithm that performs the repair is described in Listing 1.2:

```
// Game Protocol ⇒ GP, Role Protocol ⇒ RP
repair(GP, RP_Transition, GP_role) {
  RP_initial_state = RP_Transition.initial_state;
  RP_end_state = RP_Transition.final_state;
  GP_equiv_initial_states = equivalence_class(initial_state,
GP);
 GP_equiv_end_states = equivalence_class(RP_end_state,GP);
 //Add role in the recipients of the moves
 //of those transitions that start in
 //GP_equiv_initial_states, end in GP_equiv_end_states
 //and is a silent transition in the original protocol
 foreach (GP_tran from GP_state.transitions) {
  GP_initial_state = GP_tran.initial_state;
  GP_final_state = GP_tran.final_state;
  GP_m = GP_tran.move;
  GP_recipients = GP_tran.recipients;
  if (GP_initial_state ∈ GP_equiv_initial_states ∧
   GP_final_state ∈ GP_equiv_end_states ∧
    GP_role ∉ GP_recipients)
   GP_tran.move.recipients = GP_tran.move.recipients ∪
                                  GP_role;
  }
}
```

Listing 1.2. Updating silent actions by looking at equivalence groups

Updating Selected τ Actions. Our approaches to protocol repair so far, have considered silent actions as something that needs to be removed from the role's final automaton - their presence would imply lack of knowledge and failure in implementation.

However, this is not always true. A role will need to have a silent action repaired only if it is causing problems in the role's action selection process. Assuming a branch where the first move in both leaves is τ, the following combinations exist for the follow-ups:

– the two actions following the silent ones are both receive actions for the role - in that case, we do not need to repair the transition as the role has no decision to make and just waits to receive a message;
– the two actions following the silent ones are both send actions for the role and they are different in terms of either the move or the recipients of the move (or both); in this case repair is needed so that the role will have the required information to decide on which move to pursue;

– one of the following moves is a send, while the second one is a receive; we need to repair the protocol in this case too, as the role in question will need the extra information to decide whether it will wait to receive the prescribed message or go ahead and send a message.

If such moves are found in a role's LTS, then they need to be repaired. This presents the overhead of having to examine a much larger section of the protocol every time we come across a silent move, but gives smaller final protocol sizes.

The algorithm for repairing a protocol in this way is shown in Listing 1.3 (this time we have to include the role LTS as well).

```
// Game Protocol ⇒ GP, Role Protocol ⇒ RP
repair(GP, RP_Transition, GP_role, RP) {
  RP_initial_state = RP_Transition.initial_state;
  RP_end_state = RP_Transition.final_state;
  // check if the transition needs to be repaired
  RP_outgoing_transitions = find_outgoing(RP_initial_state);
  forall ( t ∈ RP_outgoing_transitions,k ∈ RP_outgoing_transitions, $k \neq t$){
        if ( t.Move == "tau" ∧ k.Move == "tau"){
            final_state_t = t.FinalState;
            final_state_k = k.FinalState;
            outgoing_transitions_newt = find_outgoing(final_state_t);
            outgoing_transitions_newk = find_outgoing(final_state_k);
            forall (r ∈ outgoing_transitions_newt ∧ s ∈
                 outgoing_transitions_newk){
               $Move_1$ = r.Move;     $Move_2$ = s.Move;
               $sender_1$ = r.Sender; $sender_2$ = s.Sender;
               $Recipients_1$ = r.Recipients; $Recipients_2$ = s.Recipients;
                if (($sender_1$ == $sender_2$ == GP_Role) ∧ (($Move_1 \neq Move_2$) ∨ (
                     $Recipient_1 \neq Recipient_2$))  ∨
                ($Sender_1$ == GP_Role ∧$Sender_2 \neq$ GP_Role ∧ GP_Role ∈ $Recipient_2$)){
                   // repair process
                  initial_equiv =equivalence_class(RP_initial_state,GP);
                  end_equiv = equivalence_class(RP_end_state,GP);
                  forall (v ∈ GP.Transitions) {
                    initial_state = v.InitialState;
                    final_state = v.FinalState;
                    if (initial_state ∈ initialequiv ∧
                        final_state ∈ endequiv)
                          v.Recipients = v.Recipients ∪ GP_Role;
                   }
                }
       }
    }
}
```

Listing 1.3. Updating selected silent transitions for role R

Example of Protocol Repair. As an example of protocols requiring repair, we can look at the example in Fig. 1d. According to $\tau^{\star}\alpha$ bisimulation there is no need for repair as no silent actions are present in the resulting automaton. However, when running *branching bisimulation* two silent actions remain. The issue here is that role *A* arrives at a point where it has to make a decision as to which message to send to role *B*, but this decision will depend on the previous decision of role *C* for which *A* has no information about.

In this case, because of the size and the structure of the protocol, all repair algorithms will require the addition of role *A* to the recipients of messages starting from state one and emanating to states three and four. Thus, role *A* should receive all messages of the protocol and receives the protocol in Fig. 5.

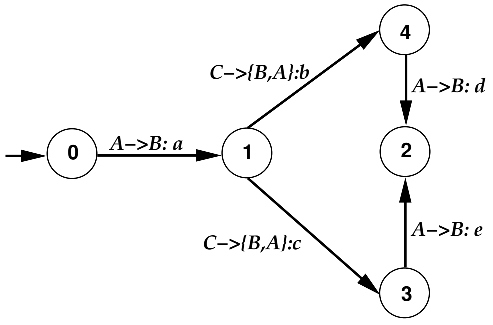


Fig. 5. Final Protocol for role A

5 Related Work

The concept of breaking down (& repairing) a protocol into constituent roles has been studied using a variety of approaches and protocol representations. In [4], Desai et al. identify the dangers of moving from the global view of a choreography (or protocol) to a local view of a single role (or agent) in either web service or multi-agent systems applications. This is important as the shift of viewpoint and the respective limitations on what the web service (or agent) can observe might mean that in the isolated agent view, there might be not enough information to implement their role specification in the choreography (or protocol).

Their description of the protocol is in a form of rules of the type $\alpha \Rightarrow \beta$. They demand that the description of the protocol always allows any proposition that is part of a rule's consequent to be part of another rule's antecedent and reachable from the beginning of the protocol. As a result of these rules, all protocols are *enactable*.

Furthermore, since they look at protocols as distributed entities and as a composition of roles, they provide an algorithm for deriving a *role skeleton*, i.e., the local view of the interaction that a role will have of the protocol including its own message exchanges. The role will need to know the messages it can send and receive, as well as any facts that enable them and lead to the creation (or discharge) of commitments (obligations of the role to bring about certain properties). The main idea in the algorithm for working out the role skeleton for a certain role is that if the role does not have knowledge of the immediate proposition needed to make a decision as to how to proceed, it should be possible to backtrack and find another one that leads with certainty to the one been examined. If the role needs to know α but it does not, then the role should go back in history and find β so that the role knows it and $\beta \rightarrow \alpha$. This algorithm works on the assumption that protocols are *enactable*. However, if they are not, there is no proposed action to rectify the problem.

Bouaziz [2] uses XML and XSD schema to describe a protocol ontology and views role as a component that can be fully specified by the *Role Profile* and *Role Behaviour* elements, as specified in [1]. In order to provide a full description of a role in the form of an XML document, all actions involving role *R* are been identified. Then, for every action *a* found a new node is added to the role XML document and all protocol actions succeeding a are added to it. As a result, the role schema will contain actions that the role in question is not directly involved in as we are just selecting everything succeeding action a from the protocol ontology, rather than the set of actions that the role is involved in. In our approach, only if the protocol is not enactable, additional knowledge will have to be inserted.

Blanc and Haumerlain [14] raise the issue of the agent been overloaded with big protocols if all the information is provided, and suggest the separation of knowledge in two different aspects. These would be the *strategic* aspect which is generated by the agent itself and consists of generating a strategy for the protocol (e.g. in an auction how should the agent bid) and the *participation* aspect that is about the agent actually participating in the protocol. The *participation* aspect will, effectively, realise the strategy plotted by the agent's *strategic* aspect. The protocol rules are defined as a *Petri Net* [11]. In order to retrieve the rules pertinent to the role, we replace any actions in which the role is not involved with ϵ. The idea is that every state in the Petri net will be characterised by a marking, i.e., the number of tokens on each place of the Petri net. The initial markings will make up the initial state and the transition relation is an empty set ($\emptyset$); afterwards, the *Graphe* [14] algorithm is applied. Their definition of a protocol can easily be accommodated by the games-based representation in [16]. Moreover, as we are interested in assessing the agent's competence and return to the agent the part of the protocol that it will be assuming in the society, we need the actual content of the messages rather than the Petri-net markings.

Giordano et al. [5] consider the representation of a local view (or role skeleton), as they look at the alphabet of each agent (Σ_i) separately. They are specifically interested in the actions that agent i can understand (send or receive). Any other action taken in the protocol will have a local equivalent that will be the empty action (ϵ) if the agent in question is not involved in it, either as a sender or a receiver. Also, the way that the local view of the agent is constructed is essentially by the use of $\tau^\star\alpha$ bisimulation, as any actions not relevant to the agent are discarded. This leads to problems, especially for protocols with a branching structure as it is not taken at all into consideration.

6 Conclusion

In this work, we looked at how a protocol specified as an LTS can be broken down into individual role automata with the use of *branching bisimulation*. However, no assumption can be made about the *enactability* of the resulting protocol. In some cases repair will be needed. We presented three approaches for repairing the protocol differentiating on the actions that need to get repaired.

This work can be expanded along with the work on the representation of the protocols in [16]. We are aiming for a representation with a higher level of abstraction, including the notion of *compound games* (i.e., describe the initial game as a composition of smaller games). If the resulting role automata can be composed in the same way that the original protocol was, it will allow for a much higher level of granularity. Furthermore, we aim to look closer into the effectiveness of the repair algorithms. We plan to perform all different repair algorithms on a number of protocols and assess the number of repairs that they will be making.

References

1. Bouaziz, W.: Une Ontologie de Protocoles pour la Coordination de Systèmes Distribués. In: Journées Francophones sur les Ontologies (JFO), Sousse, Tunisie, 18/10/07-20/10/07. pp. 231–246. Centre de Publication Universitaire (October 2007)
2. Bouaziz, W., Andonoff, E.: Dynamic Execution of Coordination Protocols in Open and Distributed Multi-Agent Systems. In: Håkansson, A., Nguyen, N.T., Hartung, R.L., Howlett, R.J., Jain, L.C. (eds.) KES-AMSTA 2009. LNCS, vol. 5559, pp. 609–618. Springer, Heidelberg (2009)
3. Davidsson, P., Johansson, S.: On the potential of norm-governed behavior in different categories of artificial societies. Comput. Math. Organ. Theory 12(2-3), 169–180 (2006), http://dx.doi.org/10.1007/s10588-006-9542-x
4. Desai, N., Mallya, A.U., Chopra, A.K., Singh, M.P.: Interaction protocols as design abstractions for business processes. IEEE Transactions on Software Engineering 31(12), 1015–1027 (2005)
5. Giordano, L., Martelli, A.: Verifying Agents' Conformance with Multiparty Protocols. In: Fisher, M., Sadri, F., Thielscher, M. (eds.) CLIMA IX. LNCS, vol. 5405, pp. 17–36. Springer, Heidelberg (2009)
6. Goradia, V., Mowry, B., Kang, P., Panjwani, M., Lowe, D., Somogyi, A., Magruder, P., Wagner, T., McNeil, D., Yang, C., Arms, W., Sirbu, M., Tygar, D.: Netbill 1994 prototype. TR 1994-11, Information Networking Institute, Carnegie Mellon University (1994)
7. Kloukinas, C., Lekeas, G., Stathis, K.: From agent game protocols to implementable roles. In: EUMAS 2008, Sixth European Workshop on Multi-Agent Systems, Bath, UK, pp. 1–15 (December 2008)
8. Kowalski, R., Sergot, M.: A logic-based calculus of events. New Generation Computing 4(1), 67–95 (1986)
9. McCarthy, J., Hayes, P.J.: Some philosophical problems from the standpoint of artificial intelligence, pp. 26–45 (1987)
10. Milner, R.: A Calculus of Communicating Systems. Springer-Verlag New York, Inc., Secaucus (1982)
11. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE 77(4), 541–580 (1989)
12. Nwana, H.S.: Software agents: An overview. Knowledge Engineering Review 11(3), 205–244 (1996)
13. Sangiorgi, D.: On the origins of bisimulation and coinduction. ACM Trans. Program. Lang. Syst. 31, 15:1–15:41 (2009), http://doi.acm.org/10.1145/1516507.1516510
14. Sibertin-Blanc, C., Hameurlain, N.: Participation Components for Holding Roles in Multiagent Systems Protocols. In: Gleizes, M.-P., Omicini, A., Zambonelli, F. (eds.) ESAW 2004. LNCS (LNAI), vol. 3451, pp. 60–73. Springer, Heidelberg (2005)
15. Stathis, K.: Game–based development of interactive systems. Ph.D. thesis, Department of Computing, Imperial College London (November 1996)
16. Stathis, K., Lekeas, G., Kloukinas, C.: Competence Checking for the Global E-Service Society Using Games. In: O'Hare, G.M.P., Ricci, A., O'Grady, M.J., Dikenelli, O. (eds.) ESAW 2006. LNCS (LNAI), vol. 4457, pp. 384–400. Springer, Heidelberg (2007)
17. Venkatraman, M., Singh, M.P.: Verifying compliance with commitment protocols. Autonomous Agents and Multi-Agent Systems 2(3), 217–236 (1999)
18. Weiser, M.: The world is not a desktop. ACM Interactions 1(1), 7–8 (1994), http://doi.acm.org/10.1145/174800.174801

Argumentation Schemes for Collaborative Planning

Alice Toniolo[1], Timothy J. Norman[1], and Katia Sycara[2]

[1] Department of Computing Science, University of Aberdeen, Scotland, UK
{a.toniolo,t.j.norman}@abdn.ac.uk
[2] Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, US
katia@cs.cmu.edu

Abstract. We address the collaborative planning problem among agents where they have different objectives and norms. In this context, agreeing on the best course of action to adopt represents a significant challenge. Concurrent actions and causal plan-constraints may lead to conflicts of opinion on what to do. Moreover, individual norms can constrain agent behaviour. We propose an argumentation-based model for deliberative dialogues based on argumentation schemes. This model facilitates agreements about joint plans by enriching the quality of the dialogue through the exchange of relevant information about plan commitments and norms.

Keywords: Argumentation schemes, Practical reasoning, Planning.

1 Introduction

In collaborative planning, intelligent agents work in cooperation to create an agreed plan of activities to fulfil requirements that are unachievable by individuals. For effective teamwork, mechanisms that enable agreements to be reached regarding a shared plan are essential. Identifying the best course of action is a complex task when agents of the team represent independent organisations with their own objectives, activities to perform and regulations to follow. Recent work has shown that the use of argumentation models for deliberative dialogues is a promising approach for generating consistent collaborative plans [1,8]. Argumentation-based dialogues provide mechanisms to facilitate agreements through the exchange of information about collaborative tasks.

In this paper, we present a model, based on argumentation schemes, that can be used for deliberative dialogues among a team of agents in preparing a collaborative plan where agents have different but interacting objectives. This model deals with issues involved in planning, considering a wide set of conflicts among actions and norm constraints. Previous work has considered norms and plan-constraints in separate contexts for solving conflicts in practical reasoning [1,8] and for norm adoption [6], but in this paper we bring together these issues within a single coherent model. Existing research on argumentation for practical reasoning has mainly focussed on collaboration among agents in the creation of a common plan when agents have different beliefs or preferences. For example, in the model presented in [1] the arguments allow agents to explore what is possible and justified at the level of a single plan among a group of agents.

In contrast, our model explores what is possible when agents elaborate individual plans for achieving different objectives where only some activities require cooperation. When an agent requires collaboration for performing an action, our model allows agents to argue about possible conflicts with their individual plans such as concurrency, causality and legality of actions. Agents can also justify the need to adopt certain actions according to their plan rules and norms. Our model allows agents to exploit these conflicts, understand the reasons that have caused them and facilitate the establishment of agreements.

This paper is organised as follows. Section 2 introduces the deliberative dialogue. Section 3 describes a language for plans. In Section 4 we describe the structure of the argumentation framework followed by the different kinds of arguments. In Section 5 we discuss related work and our conclusions.

2 Deliberative Dialogue

In our scenario agents are heterogeneous and may have different objectives that are not necessarily known to others. Agents prepare individual plans and, then, engage in deliberative dialogue regarding collaborative actions. The debate commences when one agent informs the team about its intention to perform an action or requests an action to be performed by others. The proponent may seek collaboration for different reasons; for example, the agent needs to obtain a permission from others to perform an action. The proponent engages in a discussion with other agents describing the action with preconditions, effects and the goal that this action will help to achieve. The opponents, receiving more information about the action, can select new arguments according to conflicts with their commitments. The agents involved will then exchange arguments attacking others' opinions. The discussion ends when the parties agree on which course of action to perform, or when there are no other new arguments to exchange. If agents agree, the action is included in the agents' individual plans and in the shared plan. If agents disagree, the proponent withdraws the proposal and re-plans the action with an alternative that would not conflict with the new information gathered. The protocol of a dispute between two agents x, y about a proposal φ for an action is proposed in Fig.1 following similar protocols presented in the literature (e.g. [5]). In this paper we propose three schemes for arguments that agents can use during the discussion according to issues of practical reasoning such as concurrent actions, causality among actions and norms.

3 A Model of Plans

The language for plans that underpins our model is based on *situation calculus* [9]. In this section we introduce the planning domain using the foundational axioms [9] extended for temporal applications [7] and norms [4].

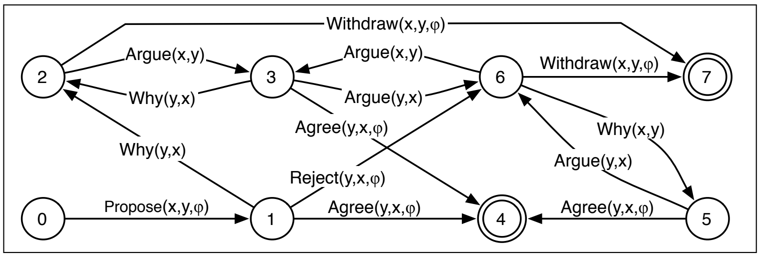


Fig. 1. Deliberative protocol for proponent x and opponent y about a proposal φ

Planning Domain. The language includes the following sorts: $\mathcal{A}$ for actions, $\mathcal{S}$ for situations, $\mathcal{O}$ for domain objects and $\mathcal{T}$ for time stamps ranging over the integers. Lower case letters refer to variables, $a \in \mathcal{A}, s \in \mathcal{S}, v \in \mathcal{O}, t \in \mathcal{T}$ and upper case letters to constants. The operators are $\wedge, \vee, \supset, \equiv, \forall, \exists, =, \neq$ and where not specified free variables are universally quantified. We refer to the set of agents as $Agt \subset \mathcal{O}$ where $Agt = \{x, y, z, \ldots\}$. A fluent $r(v_1, \ldots, v_n, s)$ is a predicate that represents the feature of the world in situation s; $R(s)$ refers to the ground predicate r in s. $S_0 \in \mathcal{S}$ is the initial situation. Other elements are: a predicate $do(a, s)$ to indicate a situation resulting from performing action a in s; a relation $s < s'$ to order situations, where s occurs before s'; and a predicate $Poss(a, s)$ to indicate that action a can be performed in s. Intuitively, through instantiations of $Poss(a, s)$, situations are structured as a tree where the root is S_0, nodes are situations representing the world while arcs are possible actions that modify the state of the world. Each situation s is the result of performing a sequence of actions from root S_0, represented as $s = do(a_n, do(a_{n-1}, \ldots do(a_1, S_0)))$ and abbreviated $s = do(\langle a_1; \ldots; a_n\rangle, S_0)$. This tree represents the planning domain. A Basic Action Theory $\mathcal{D}$ [9] defines the axioms for formalising the domain. Our model is based on an extended version, $\mathcal{D}_{ext}$, for norms and durative actions.

Definition 1. *An Extended Action Theory is* $\mathcal{D}_{ext} = \mathcal{D} \cup \Omega_d \cup \Omega_n$ *where:*

– $\mathcal{D} = \Sigma \cup \mathcal{D}_{S_0} \cup \mathcal{D}_{una} \cup \mathcal{D}_{ss} \cup \mathcal{D}_{ap}$ *is a Basic Action Theory where:*
  - Σ*: set of domain independent axioms for situations.*
  - $\mathcal{D}_{S_0}$*: set of sentences representing the initial state of the world.*
  - $\mathcal{D}_{una}$*: set of unique name axioms for actions.*
  - $\mathcal{D}_{ap}$*: set of action precondition axioms in the form* $\Pi_A(s) \equiv Poss(A, s)$.
  - $\mathcal{D}_{ss}$*: set of successor state axioms in the form:*

$$Poss(a, s) \supset [R(do(a, s)) \equiv \gamma_R^+(a, s) \vee (R(s) \wedge \neg\gamma_R^-(a, s))]$$

where γ_R^+, γ_R^- *represent the add and delete conditions for fluent* R*.*

– Ω_d*: set of axioms that handle actions with duration.*
– Ω_n*: set of axioms that handle norms.*

The set Ω_d includes axioms that ensure consistency for actions with duration $a_d \in \mathcal{A}$, following the approach of Pinto et al. [7]. Intuitively, action a_d is formed by: an instantaneous action $begin(a_d)$ for the beginning of a_d; an action $end(a_d)$ for the end of a_d; and a fluent $exec(a_d, s)$ for the execution of a_d during s. In this approach different actions can be concurrent as long as their beginning and ending points do not coincide. The extension Ω_n (based on Demolombe et al. [4]) includes axioms that define what an agent is obliged, permitted or forbidden to do in terms of actions and features of the world under certain conditions. We assume that everything is permitted when not explicitly prohibited and that all individual norms are logically consistent. Active norms branch nodes or arcs of the possible situation tree. Two operators, O and F, combined to a fluent express obligations and prohibitions. A special fluent $occ(a, s)$, denoting an action which occurs at the end of s, is used to represent norms that regulate actions.

Definition 2. *The following fluents define norm constraints:* $\boldsymbol{O_R(s)}$ *is an obligation fluent which holds in situation s iff a norm asserts that fluent R must hold in s.* $\boldsymbol{F_R(s)}$ *is a prohibition fluent which holds in s iff a norm asserts that R must not hold in s.* $F_R(s) \wedge R(s)$ *and* $O_R(s) \wedge \neg R(s)$ *are violations for these norms.* $\boldsymbol{O_{occ(A)}(s)}$ *indicates that an action A must be the next action occurring after s.* $\boldsymbol{F_{occ(A)}(s)}$ *indicates that action A is forbidden to occur as the next action. Violations are* $Poss(A, s) \wedge [(\exists a).(a \neq A) \wedge O_{occ(a)}(s)]$ *and* $F_{occ(A)}(s) \wedge Poss(A, s)$.

When an action a_d is executed in an interval s_1 to s_2, if a_d is forbidden the beginning, end and execution of a_d are forbidden for the whole interval, $F_{occ(A_d)}(s_1, s_2)$. If a_d is obliged, the condition enforces a_d to start in situation s_1 and end in s_2, $O_{occ(A_d)}(s_1, s_2)$. Each norm fluent is accompanied by a successor state axiom in the form of Def. 1; e.g. $O_R(do(a, s)) \equiv [\xi^+_{O_R}(a, s) \vee (O_R(s) \wedge \neg\xi^-_{O_R}(a, s))]$, where ξ^+, ξ^- are conditions of activation and expiration of norms. A situation s is legal when all the actions in the history of s are possible, defined by the predicate $legal(s) \equiv [s = S_0 \vee (\forall a, s').do(a, s') \leq s \supset Poss(a, s') \wedge \neg\eta(a, s')]$ and compliant with the norms ensured by the predicate $\eta(a, s)$ defined as: $\eta(a, s) \equiv [(\exists r).r(s) \wedge F_r(s) \vee \neg r(s) \wedge O_r(s)] \vee [F_{occ(a)}(s) \vee (\exists a' \neq a) \wedge O_{occ(a')}(s)]$.

Agent Plans. An agent x maintains a description of the planning domain as a subset of the extended action theory, $\mathcal{D}^x_{ext} \subseteq \mathcal{D}_{ext}$. Agent x's individual plan P^x is a sequence of actions that identifies a path of possible and legal situations in the situation tree that goes from the root to a situation where the overall goal is satisfied. The goals of an agent x are a set of sentences ψ_k. The overall goal Ψ is defined by their conjunction through a sentence $\Psi \equiv \psi_1 \wedge \cdots \wedge \psi_m$. Since actions of an agent may not be known by other agents, situations represent only internal states and they have no meaning outside that agent's subjective frame of reference. We denote S^x as a situation on the path identified by x's plan.

Definition 3. *An individual plan* P^x *is a solution to a planning problem identified by a domain* $\mathcal{D}^x_{ext}$ *and a goal* Ψ*.* $P^x = \langle A_1; \ldots; A_n\rangle$ *is a sequence of actions that identifies a path of legal situations from* S^x_0 *to a situation* S^x_n *where* $\Psi(S^x_n)$ *is satisfied.* $\mathcal{D}^x_{ext} \models legal(S^x_n) \wedge \Psi(S^x_n)$ *and* $S^x_n = do(\langle A_1; \ldots; A_n\rangle, S^x_0)$.

Table 1. Elements of $\vec{P}^x$

- $\boldsymbol{P^x(\psi_k)}$: sequence of actions in P^x to achieve a goal ψ_k in S^x_k where $S^x_k \leq S^x_n$.
- $\boldsymbol{\mathcal{P}_{A_k} = \{R_1, \ldots, R_n\}}$: set of preconditions of action A_k. If A_k is in P^x, Π_{A_k} must hold in a situation S^x_{k-1} since $\Pi_{A_k}(S^x_{k-1}) \equiv Poss(A_k, S^x_{k-1})$. $\mathcal{P}_{A_k}$ is the minimal set of R_i holding in S^x_{k-1} that satisfies the formula $\Pi_{A_k}(S^x_{k-1})$.
- $\boldsymbol{\mathcal{E}_{A_k} = \{R_1, \ldots, R_m\}}$: set of effects of action A_k. A fluent $R_i(S^x_k)$ is an effect of A_k if the transition causes a change of its truth value $R_i(S^x_{k-1}) \equiv \neg R_i(S^x_k)$.
- $\langle \boldsymbol{\mathcal{P}_{A_k}, \mathcal{E}_{A_k}} \rangle$: tuple representing the execution of an action. When A_{dk} has duration, $\mathcal{P}_{A_{dk}} = \mathcal{P}_{begin(A_{dk})}$ and $\mathcal{E}_{A_{dk}} = \mathcal{E}_{end(A_{dk})}$ are related to the global action.
- $\boldsymbol{cLink(R, A_h, A_k)}$: causal link which represents a relation between A_h and A_k in P^x, with $S^x_k = do(A_k, S^x_{k-1})$ and $S^x_h = do(A_h, S^x_{h-1})$, where a precondition $R(S^x_{k-1}) \in \mathcal{P}_{A_k}$ not true in S_0, is provided by an effect $R(S^x_h) \in \mathcal{E}_{A_h}$ of a previous action A_h and $S^x_h \leq S^x_{k-1}$. $R(s^x)$ holds in all the situations $S^x_h \leq s^x \leq S^x_{k-1}$.
- $\boldsymbol{occBet(a, s^x_1, s^x_2)}$: an action a occurs in the path from s^x_1 to s^x_2 defined as $occBet(a, s^x_1, s^x_2) \equiv [(\exists s^x_p, s^x_q).s^x_q = do(a, s^x_p) \wedge s^x_1 \leq s^x_p < s^x_q \leq s^x_2]$.
- $\boldsymbol{occBet_{\mathcal{T}}(a, t_1, t_2)}$:] represents the previous relation in terms of time obtained from $occBet_{\mathcal{T}}(a, t_1, t_2) \equiv [occBet(a, s^x_1, s^x_2) \wedge start(s^x_1) = t_1 \wedge start(s^x_2) = t_2]$.
- $\boldsymbol{occOver(a_d, s^x_1, s^x_2)}$: the execution of a_d overlaps the path s^x_1 to s^x_2, where $occOver(a_d, s^x_1, s^x_2) \equiv [occBet(begin(a_d), s^x_1, s^x_2) \vee occBet(end(a_d), s^x_1, s^x_2)]$.
- $\boldsymbol{occOver_{\mathcal{T}}(a_d, t_1, t_2)}$: the corresponding time relation where $occOver_{\mathcal{T}}(a_d, t_1, t_2) \equiv [occBet_{\mathcal{T}}(begin(a_d), t_1, t_2) \vee occBet_{\mathcal{T}}(end(a_d), t_1, t_2)]$.
- **Norm Premises**: A norm is activated if the formula $\xi^+_{F/O_R}(A_p, S^x_p)$ holds in $S^x_p = do(A_p, S^x_{p-1})$. The norm premises are identified by $\boldsymbol{\mathcal{N}_{Prem} = \{R_1, \ldots, R_n\}}$ as the minimal set of $R_i(S^x_{p-1})$ which satisfy ξ^+ and $\boldsymbol{A_{dp}}$ causing the transition.

In order for agents to engage in dialogue about their plans, they rely on a common time line where the path of situations identified by the plan is grounded to obtain a temporal plan, $\vec{P}^x$. Two functions are used for an agent x to identify the plan temporally grounded. A function $start(s^x) = t$ indicates the time t when the situation s^x begins ($start(S_0) = 0$). A function $sit(x, t)$ returns the ongoing situation in x's plan at time t. In fact, s^x is returned if $(\exists s^x, a).start(s^x) \leq t < start(do(a, s^x)) \supset sit(x, t) = s^x$. When action A_k in P^x causes the transition from S^x_{k-1} to $S^x_k = do(A_k, S^x_{k-1})$ and the temporal axioms are $start(S^x_{k-1}) = T_{k-1}$ and $start(S^x_k) = T_k$, we refer to the temporal action A_k as $[T_k]A_k$. Hence, the plan $\vec{P}^x$ is composed by $\vec{P}^x = \langle [T_1]A_1; \ldots; [T_n]A_n \rangle$. Moreover, agents engaging in dialogue need to deal with new knowledge introduced by other team members. We use the K-accessibility relation in [9] where $K(s', s)$ indicates that in s an agent believes that the world is in situation s'. This relation associates situations that the agent considers indistinguishable. An agent x knows φ in s^x, $Know(\varphi, s^x) \overset{\text{def}}{=} \forall s^x_q(K(s^x_q, s^x) \supset \varphi(s^x_q))$, if φ holds in all situations K-accessible from s^x. Table 1 summarises further elements of plan that are used in the paper.

4 Argumentation

The arguments that agents use during the deliberative dialogue are structured as argumentation schemes [10]. These schemes are composed by premises and conclusions (delimited by "⇒"). Each scheme also includes a set of critical questions *CQ*s that provide a structured way of challenging an argument. Thus, argumentation schemes are defeasible rules of inference in the norm-governed practical reasoning context that provide heuristics to guide the deliberative dialogue. We introduce general critical questions which identify the type of argument that can be formulated and then we specify what can be captured by each argument with more specific attacks (*ATK*s). The following locutions are used in the arguments:

Definition 4. *Given plan $\vec{P}^x$ of agent x:* $\boldsymbol{Perform(x, A_{dk}, T_{k1}, T_{k2})}$ *indicates that x intends to perform action A_{dk} through $[T_{k1}]begin(A_{dk})$, $[T_{k2}]end(A_{dk})$;* $\boldsymbol{Hold(x, R, T_k)}$ *indicates that for x a feature R holds at T_k, $Hold(x,R,T_k) \stackrel{def}{=} [R(S_k^x) \wedge start(S_k^x) = T_k]$ and $\neg Hold(x,R,T_k) \stackrel{def}{=} [\neg R(S_k^x) \wedge start(S_k^x) = T_k]$; $Hold(x, \{R_1, \dots, R_n\}, T_k) \stackrel{def}{=} [Hold(x, R_1, T_k) \wedge \dots \wedge Hold(x, R_n, T_k)]$ is used for a set of features;* $\boldsymbol{achieve(x, \psi_k)}$ *indicates that x intends to achieve goal ψ_k.*

In our description the debate is between agent x and agent y where x is informing y about the intention of performing action A_{dk}. Hence, the claim φ is $Perform(x, A_{dk}, T_{k1}, T_{k2})$. The individual plans are $\vec{P}^x$ and $\vec{P}^y$ where A_{dk} is in $\vec{P}^x$ and if agents agree, A_{dk} will be included in the set of agreed actions $\vec{P}^{xy}$.

We take as starting point an adaptation of Atkinson's argumentation scheme for practical reasoning [1]. This scheme involves a single action warranted by preconditions, where the effects allow the agent to achieve a goal which promotes or demotes a value. An argument Arg_I initiates the debate about the action, i.e. $Perform(x, A_{dk}, T_{k1}, T_{k2})$. The proponent specifies $\mathcal{P}_{A_{dk}}$, $\mathcal{E}_{A_{dk}}$ for A_{dk} in $\vec{P}^x$ and a partial goal ψ_k where $A_{dk} \in P^x(\psi_k)$. The initial argument Arg_I results:

$\boldsymbol{Arg_I}$ **:** - Given preconditions $Hold(x, \mathcal{P}_{A_{dk}}, T_{k1})$
- $Perform(x, A_{dk}, T_{k1}, T_{k2})$
- brings about $Hold(x, \mathcal{E}_{A_{dk}}, T_{k2})$
- that will contribute to $achieve(x, \psi_k)$ [where $A_{dk} \in P^x(\psi_k)$],

⇒ therefore $Perform(x, A_{dk}, T_{k1}, T_{k2})$

The opponent y receiving Arg_I can select new arguments according to possible conflicts with commitments and norms following the critical questions:

- **CQ1:** Is the action possible according to concurrent actions in the plan?
- **CQ2:** Is the action possible according to causal plan constraints?
- **CQ3:** Is there any norm which regulates actions or states of the world?

A description of the argument structures is presented in the following sections.

Arguments for Concurrent Actions. Here, we explore the critical question *CQ1* concerned with concurrent actions. The argumentation scheme Arg_c is an extension of the initial argument where action A_{dk} is considered in the context of other actions already scheduled in an individual plan. Although in an individual plan actions can be concurrent only if starting or ending points do not coincide, actions from different plans can entirely overlap. Two concurrent actions A_k and A_h are executable if their preconditions hold and their effects are consistent.

Furthermore, the effects of A_k should not contradict the preconditions of A_h and vice-versa. For a durative action A_{dk} there should not be direct interference on its preconditions and effects with other actions A_{dh} throughout its execution.

When agent x discusses with y about A_{dk} with specifications $\langle \mathcal{P}_{A_{dk}}, \mathcal{E}_{A_{dk}} \rangle$ (see Table 1), y needs to identify its internal situations corresponding to the execution of A_{dk}. Action A_{dk} is represented by $[T_{k1}]begin(A_{dk})$ and $[T_{k2}]end(A_{dk})$ in $\vec{P}^x$. The situations that delimit A_{dk} from y's point of view are $sit(y, T_{k1})$ and $sit(y, T_{k2})$. Suppose that in y's plan there is an action A_{dh} overlapping the execution of A_{dk}, $(\exists s^y).sit(y, T_{k1}) \leq s^y \leq sit(y, T_{k2})$ where $exec(A_h, s^y)$ holds. Action A_{dh} is formed by $[T_{h1}]begin(A_{dh}), [T_{h2}]end(A_{dh})$ in $\vec{P}^y$. The concurrency for y is $occOver(A_{dk}, S^y_{h1}, S^y_{h2})$ where $S^y_{h1-1} < S^y_{h1} \leq S^y_{h2-1} < S^y_{h2}$ and $S^y_{h1} = do(begin(A_{dh}), S^y_{h1-1}) \wedge S^y_{h2} = do(end(A_{dh}), S^y_{h2-1}) \wedge start(S^y_{h1}) = T_{h1} \wedge start(S^y_{h2}) = T_{h2}$. Action A_{dk} is possible for agent y only if there is no interference on preconditions and effects between A_{dk} and A_{dh} throughout their execution; i.e. $(\forall s^y, r).r \in \{\mathcal{P}_{A_{dk}} \cup \mathcal{E}_{A_{dk}}\} \wedge S^y_{h1} \leq s^y \leq S^y_{h2} \wedge occOver(A_{dk}, S^y_{h1}, S^y_{h2}) \supset Know(r, s^y) \vee \neg Know(\neg r, s^y) \vee \neg Know(r, s^y)$. The conflicting condition is $Know(\neg R, s^y)$. Action A_{dk} cannot be adopted by y when in its plan there is a concurrent action A_{dh} and: *i)* A_{dk} has an effect $R \in \mathcal{E}_{A_{dk}}$ that contradicts an effect $\neg R \in \mathcal{E}_{A_{dh}}$ of A_{dh}; *ii)* A_{dk} has an effect $R \in \mathcal{E}_{A_{dk}}$ that negates a precondition $\neg R \in \mathcal{P}_{A_{dh}}$ of A_{dh}; *iii)* A_{dk} has a precondition $R \in \mathcal{P}_{A_{dk}}$ negated by an effect $\neg R \in \mathcal{E}_{A_{dh}}$ of A_{dh}; or *iv)* A_{dk} has a precondition $R \in \mathcal{P}_{A_{dk}}$ that contradicts precondition $\neg R \in \mathcal{P}_{A_{dh}}$.

Argument Arg_c involves the two conflicting actions A_{dk}, A_{dh} with their specifications. The conflict is caused by A_{dk}, which $occOver_{\mathcal{T}}(A_{dk}, T_{h1}, T_{h2})$, since $R \in \{\mathcal{P}_{A_{dk}} \cup \mathcal{E}_{A_{dk}}\}$ and $\neg R \in \{\mathcal{P}_{A_{dh}} \cup \mathcal{E}_{A_{dh}}\}$. Arg_c has the structure:

Arg_c:
- Given preconditions $Hold(x, \mathcal{P}_{A_{dk}}, T_{k1})$,
- $Perform(x, A_{dk}, T_{k1}, T_{k2})$,
- brings about $Hold(x, \mathcal{E}_{A_{dk}}, T_{k2})$ [and $R \in \{\mathcal{P}_{A_{dk}} \cup \mathcal{E}_{A_{dk}}\}$],
- and given preconditions $Hold(y, \mathcal{P}_{A_{dh}}, T_{h1})$,
- $\neg Perform(y, A_{dh}, T_{h1}, T_{h2})$,
- brings about $\neg Hold(y, \mathcal{E}_{A_{dh}}, T_{h2})$ [and $\neg R \in \{\mathcal{P}_{A_{dh}} \cup \mathcal{E}_{A_{dh}}\}$]
- that will contribute $\neg achieve(y, \psi_h)$, [and $A_{dh} \in P^y(\psi_h)$]
- and $occOver_{\mathcal{T}}(A_{dk}, T_{h1}, T_{h2})$

$\Rightarrow$ therefore $\neg Perform(x, A_{dk}, T_{k1}, T_{k2}) \wedge Perform(y, A_{dh}, T_{h1}, T_{h2})$.

The conditions for conflicts between two concurrent actions lead an agent to formulate the following attacks against an action A_{dk} (formalised in Table 2):

– **ATK1.1:** A_{dk} has an effect R that contradicts an effect $\neg R$ of action A_{dh} in my plan and their execution overlaps, therefore A_{dk} should not be performed.
– **ATK1.2:** A_{dk} has an effect R that negates a precondition $\neg R$ of A_{dh} in my plan and their execution overlaps, therefore A_{dk} should not be performed.
– **ATK1.3:** A_{dk} has a precondition R that is negated by an effect $\neg R$ of A_{dh} in my plan and their execution overlaps, therefore A_{dk} should not be performed.
– **ATK1.4:** A_{dk} has a precondition R that contradicts a precondition $\neg R$ of action A_{dh} in my plan and their execution overlaps, hence A_{dk} should not be performed.

Arguments for Plan Constraints. In this section we define the argument for causal plan constraints Arg_p following *CQ2*. The preconditions allowing an action to be scheduled in the plan must be satisfied by performing other actions when not true at the outset. The causal relations among these actions must be protected when new actions are included in the plan for ensuring consistency. When an agent y receives a proposal $Perform(x, A_{dk}, T_{k1}, T_{k2})$ from agent x with $\langle \mathcal{P}_{A_{dk}}, \mathcal{E}_{A_{dk}} \rangle$, it should verify that A_{dk} does not threat any causal relation in $\vec{P}^y$. Assume that A_{dk} has a precondition or an effect R, but $\neg R$ is a causal link between two actions A_{da} and A_{db} in $\vec{P}^y$. A conflict occurs when action A_{dk} is concurrent to the execution of the sequence of the two actions A_{da} and A_{db}. The link is represented as $cLink(\neg R, A_{da}, A_{db})$ where $\neg R \in \mathcal{E}_{A_{da}}$ and $\neg R \in \mathcal{P}_{A_{db}}$ (see Table 1). Agent y believes that $Know(\neg R, s^y)$ in the path $(\forall s^y) S^y_{a1} \leq s^y \leq S^y_{b2}$ from the starting of A_{da} to the end of A_{db}, where $S^y_{a1} = do(begin(A_{da}), S^y_{a1-1})$ and $S^y_{b2} = do(end(A_{db}), S^y_{b2-1})$. Action A_{dk} where $R \in \{\mathcal{P}_{A_{dk}} \cup \mathcal{E}_{A_{dk}}\}$ is a threat to the causal link if $occOver(A_{dk}, S^y_{a1}, S^y_{b2})$ holds. Hence, argument Arg_p results:

$\boldsymbol{Arg_p}$: - Given preconditions $Hold(x, \mathcal{P}_{A_{dk}}, T_{k1})$,
- $Perform(x, A_{dk}, T_{k1}, T_{k2})$,
- brings about $Hold(x, \mathcal{E}_{A_{dk}}, T_{k2})$ [and $R \in \{\mathcal{P}_{A_{dk}} \cup \mathcal{E}_{A_{dk}}\}$],
- Given $cLink(\neg R, A_{da}, A_{db})$
- $\neg Perform(y, A_{da}, T_{a1}, T_{a2})$ and $\neg Perform(y, A_{db}, T_{b1}, T_{b2})$,
- and $occOver_{\mathcal{T}}(A_{dk}, T_{a1}, T_{b2})$

$\Rightarrow$ therefore $\neg Perform(x, A_{dk}, T_{k1}, T_{k2}) \wedge Perform(y, A_{da}, T_{a1}, T_{a2}) \wedge Perform(y, A_{db}, T_{b1}, T_{b2})$

An opponent agent can attack A_{dk} with Arg_p if its preconditions or effects are threats to a causal link. Agent x can also justify the adoption of an action with a similar argument. If x's plan contains a causal link from A_{dk} to $A_{dj} \in \vec{P}^x$, $cLink(Q, A_{dk}, A_{dj})$ such that $Q \in \mathcal{E}_{A_{dk}}, Q \in \mathcal{P}_{A_{dj}}$, then A_{dk} is necessary to perform A_{dj}. Therefore, agent x can formulate the arguments (see Table 2):

– **ATK2.1:** A precondition of action A_{dk} is a threat to a causal link between two action A_{da}, A_{db} in my plan, therefore A_{dk} should not be performed.
– **ATK2.2:** An effect of action A_{dk} is a threat to a causal link between two action A_{da}, A_{db} in my plan, therefore A_{dk} should not be performed.
– **ATK2.3:** The effects of action A_{dk} are fundamental preconditions for executing action A_{dj} in my plan, therefore A_{dk} should be performed.

Arguments for Norms. Here, we expand upon the critical question *CQ3* for norms. We consider norms as external regulations about what the agent is forbidden or obliged to do in terms of actions and states of the world. Agents' plans are internally norm-consistent enforced by the legality of situation S_n where the overall individual goal is achieved. Conflicts may arise when new information about states of the world or actions introduced by others cause inconsistencies. An agent x can assert that a feature R_k holds, $Hold(x, R_k, T_k)$. From agent y's point of view this is possible only if R_k is not forbidden to hold, $\neg Know(F_{R_k}, sit(y, T_k)) \wedge \neg Know(O_{\neg R_k}, sit(y, T_k))$. When an agent x claims to $Perform(x, A_{dk}, T_{k1}, T_{k2})$, y's norms allow x to perform A_{dk} only if the interval of the execution of A_{dk} does not overlap an interval where a norm forbids the performance of the action. If $F_{occ(A_{dk})}(S_1^y, S_2^y)$ and $OccOver(A_{dk}, S_1^y, S_2^y)$, where the interval of situations $sit(y, T_{k1})$ to $sit(y, T_{k2})$ overlaps S_1^y to S_2^y, then performing A_{dk} is forbidden. The argument for norms is based on the normative reasoning scheme proposed by Oren *et al.* [6]. The structure involves the active norm with conclusions about actions and features of the world and its premises as $Hold(y, \mathcal{N}_{Prem}, T_p)$ and $Perform(y, A_{dp}, T_{p1}, T_{p2})$ (see Table 1). Hence, the argument for norms for a prohibition Arg_n has the following structure:

$\boldsymbol{Arg_n}$: - Given premises $Hold(y, \mathcal{N}_{Prem}, T_p)$ and $Perform(y, A_{dp}, T_{p1}, T_{p2})$,
- a norm $Forbids$ to
- $Perform(x, A_{dk}, T_{k1}, T_{k2})/Hold(x, R, T_k)$
$\Rightarrow$ therefore $\neg Perform(x, A_{dk}, T_{k1}, T_{k2})/\neg Hold(x, R, T_k)$

An opponent y can argue against a proponent x according to the instantiated norms. Furthermore, those parts in x's plan committed to maintain norm consistency can be justified with a similar argument. Agent x can claim $Hold(x, R_k, T_k)$ because of a norm that enforces $O_{R_k}(S_k^x)$ or $F_{\neg R_k}(S_k^x)$ where $T_k = start(S_k^x)$. A norm can enforce an agent x to $Perform(x, A_{dk}, T_{k1}, T_{k2})$. If action A_{dk} is obliged to occur between S_1^x and S_2^x where $start(S_1^x) = T_{k1}$ and $start(S_2^x) = T_{k2}$ then, $O_{occ(A_{dk})}(S_1^x, S_2^x)$, action A_{dk} must be performed between time T_{k1} and T_{k2}. The arguments for norms are identified as follows (formal structure in Table 2):

- **ATK3.1:** An active norm in my plan forbids an action A_{dk} to be performed, therefore A_{dk} should not be performed.
- **ATK3.2:** An active norm in my plan forbids a feature of the world R to hold, therefore R should not hold.
- **ATK3.3:** An active norm in my plan obliges an action A_{dk} to be performed, therefore A_{dk} should be performed.
- **ATK3.4:** An active norm in my plan obliges a feature of the world R to hold, therefore R should hold.

4.1 Example

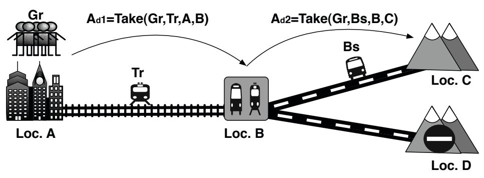
In this section we illustrate some characteristics of our model using an example. Agent x represents a travel agency that arranges sustainable tourism trips in collaboration with public transport companies; the use of public transport being motivated by the obligation to minimise the carbon footprint of the trip. Agent x schedules a package to the mountains considering two destinations, C and D. C is preferred because D is dangerous and an internal policy forbids to schedule trips to unsafe locations. The group of customers Gr leaves from location A by train Tr to a station B and, then, they will take bus Bs to reach C (See Fig.2). Travel agency x engages in a discussion with train company y to obtain information about Tr from A to B. The initial proposal is $Perform(x, A_{d1}, 2, 8)$. Agent y has planned at the same time to do maintenance on the railway from A to B and it is obliged to divert the train Tr for B towards a location E for safety reasons. Hence, the train company rejects the proposal. The agents will exchange arguments following the protocol in Fig.1. At the end, since x cannot formulate

Table 2. Formalisation of Arguments Arg_c, Arg_p, Arg_n

CQ1-ATK1.1: $\langle Hold(x,\mathcal{P}_{A_{dk}},T_{k1}), Perform(x,A_{dk},T_{k1},T_{k2}), Hold(x,\mathcal{E}_{A_{dk}},T_{k2}) \wedge \boldsymbol{R} \in \boldsymbol{\mathcal{E}_{A_{dk}}}, Hold(y,\mathcal{P}_{A_{dh}},T_{h1}), \neg Perform(y,A_{dh},T_{h1},T_{h2}), \neg Hold(y,\mathcal{E}_{A_{dh}},T_{h2}) \wedge \neg \boldsymbol{R} \in \boldsymbol{\mathcal{E}_{A_{dh}}}, \neg achieve(y,\psi_h), occOver_{\mathcal{T}}(A_{dk},T_{h1},T_{h2}) \Rightarrow \neg Perform(x,A_{dk},T_{k1},T_{k2}) \wedge Perform(y,A_{dh},T_{h1},T_{h2})\rangle$
CQ1-ATK1.2:$\langle Hold(x,\mathcal{P}_{A_{dk}},T_{k1}), Perform(x,A_{dk},T_{k1},T_{k2}), Hold(x,\mathcal{E}_{A_{dk}},T_{k2}) \wedge \boldsymbol{R} \in \boldsymbol{\mathcal{E}_{A_{dk}}}, \neg Hold(y,\mathcal{P}_{A_{dh}},T_{h1}) \wedge \boldsymbol{\neg R} \in \boldsymbol{\mathcal{P}_{A_{dh}}}, \neg Perform(y,A_{dh},T_{h1},T_{h2}), \neg Hold(y,\mathcal{E}_{A_{dh}},T_{h2}), \neg achieve(y,\psi_h), occOver_{\mathcal{T}}(A_{dk},T_{h1},T_{h2}) \Rightarrow \neg Perform(x,A_{dk},T_{k1},T_{k2}) \wedge Perform(y,A_{dh},T_{h1},T_{h2})\rangle$
CQ1-ATK1.3: $\langle Hold(x,\mathcal{P}_{A_{dk}},T_{k1}) \wedge \boldsymbol{R} \in \boldsymbol{\mathcal{P}_{A_{dk}}}, Perform(x,A_{dk},T_{k1},T_{k2}), Hold(x,\mathcal{E}_{A_{dk}},T_{k2}), Hold(y,\mathcal{P}_{A_{dh}},T_{h1}), \neg Perform(y,A_{dh},T_{h1},T_{h2}), \neg Hold(y,\mathcal{E}_{A_{dh}},T_{h2}) \wedge \boldsymbol{\neg R} \in \boldsymbol{\mathcal{E}_{A_{dh}}}, \neg achieve(y,\psi_h), occOver_{\mathcal{T}}(A_{dk},T_{h1},T_{h2}) \Rightarrow \neg Perform(x,A_{dk},T_{k1},T_{k2}) \wedge Perform(y,A_{dh},T_{h1},T_{h2})\rangle$
CQ1-ATK1.4: $\langle Hold(x,\mathcal{P}_{A_{dk}},T_{k1}) \wedge \boldsymbol{R} \in \boldsymbol{\mathcal{P}_{A_{dk}}}, Perform(x,A_{dk},T_{k1},T_{k2}), Hold(x,\mathcal{E}_{A_{dk}},T_{k2}), \neg Hold(y,\mathcal{P}_{A_{dh}},T_{h1}) \wedge \boldsymbol{\neg R} \in \boldsymbol{\mathcal{P}_{A_{dh}}}, \neg Perform(y,A_{dh},T_{h1},T_{h2}), \neg Hold(y,\mathcal{E}_{A_{dh}},T_{h2}), \neg achieve(y,\psi_h), occOver_{\mathcal{T}}(A_{dk},T_{h1},T_{h2}) \Rightarrow \neg Perform(x,A_{dk},T_{k1},T_{k2}) \wedge Perform(y,A_{dh},T_{h1},T_{h2})\rangle$
CQ2-ATK2.1: $\langle Hold(x,\mathcal{P}_{A_{dk}},T_{k1}) \wedge \boldsymbol{R} \in \boldsymbol{\mathcal{P}_{A_{dk}}}, Perform(x,A_{dk},T_{k1},T_{k2}), Hold(x,\mathcal{E}_{A_{dk}},T_{k2}), cLink(\boldsymbol{\neg R},A_{da},A_{db}), \neg Perform(y,A_{da},T_{a1},T_{a2}), \neg Perform(y,A_{db},T_{b1},T_{b2}), occOver_{\mathcal{T}}(A_{dk},T_{a1},T_{b2}) \Rightarrow \neg Perform(x,A_{dk},T_{k1},T_{k2}) \wedge Perform(y,A_{da},T_{a1},T_{a2}) \wedge Perform(y,A_{db},T_{b1},T_{b2})\rangle$
CQ2-ATK2.2: $\langle Hold(x,\mathcal{P}_{A_{dk}},T_{k1}), Perform(x,A_{dk},T_{k1},T_{k2}), Hold(x,\mathcal{E}_{A_{dk}},T_{k2}) \wedge \boldsymbol{R} \in \boldsymbol{\mathcal{E}_{A_{dk}}}, cLink(\boldsymbol{\neg R},A_{da},A_{db}), \neg Perform(y,A_{da},T_{a1},T_{a2}), \neg Perform(y,A_{db},T_{b1},T_{b2}), occOver_{\mathcal{T}}(A_{dk},T_{a1},T_{b2}) \Rightarrow \neg Perform(x,A_{dk},T_{k1},T_{k2}) \wedge Perform(y,A_{da},T_{a1},T_{a2}) \wedge Perform(y,A_{db},T_{b1},T_{b2})\rangle$
CQ2-ATK2.3: $\langle Hold(x,\mathcal{P}_{A_{dk}},T_{k1}), Perform(x,A_{dk},T_{k1},T_{k2}), Hold(x,\mathcal{E}_{A_{dk}},T_{k2}) \wedge \boldsymbol{R} \in \boldsymbol{\mathcal{E}_{A_{dk}}}, cLink(\boldsymbol{R},A_{dk},A_{dj}), Perform(x,A_{dk},T_{k1},T_{k2}), Perform(x,A_{dj},T_{j1},T_{j2}), occOver_{\mathcal{T}}(A_{dk},T_{k1},T_{j2}) \Rightarrow Perform(x,A_{dk},T_{k1},T_{k2}) \wedge Perform(x,A_{dj},T_{j1},T_{j2})\rangle$
CQ3-ATK3.1: $\langle Hold(y,\mathcal{N}_{Prem},T_p) \wedge Perform(y,A_{dp},T_{p1},T_{p2}), Forbids : Perform(x,A_{dk},T_{k1},T_{k2}) \Rightarrow \neg Perform(x,A_{dk},T_{k1},T_{k2})\rangle$
CQ3-ATK3.2: $\langle Hold(y,\mathcal{N}_{Prem},T_p) \wedge Perform(y,A_{dp},T_{p1},T_{p2}), Forbids : Hold(x,R,T_k) \Rightarrow \neg Hold(x,R,T_k)\rangle$
CQ3-ATK3.3: $\langle Hold(x,\mathcal{N}_{Prem},T_q) \wedge Perform(x,A_{dq},T_{q1},T_{q2}), Obliges : Perform(x,A_{dk},T_{k1},T_{k2}) \Rightarrow Perform(x,A_{dk},T_{k1},T_{k2})\rangle$
CQ3-ATK3.4: $\langle Hold(x,\mathcal{N}_{Prem},T_q) \wedge Perform(x,A_{dq},T_{q1},T_{q2}), Obliges : Hold(x,R,T_k) \Rightarrow Hold(x,R,T_k)\rangle$

any other argument it withdraws the action and re-plans avoiding the train from A to B. The arguments presented are formalised in Tab.3. The argument Arg_I explains the requirements of action A_{d1}. Using ATK2.3, x explains that the customers have to reach B for catching the bus from B to C. Action A_{d1} is justified with A_{d2} following the causal link $cLink(R_2, A_{d1}, A_{d2})$. Using ATK1.1 y explains that Tr leaving from A is diverted towards location E and it will not reach B at the scheduled time. Diverting Tr, A_{d4}, is necessary for the train company to secure the part of railway under maintenance, $\psi_2(S_4^y)$. The conflict occurs because A_{d1} is concurrent to A_{d4} where $occOver_{\mathcal{T}}(A_{d1}, 2, 7)$. Agent y justifies A_{d4} with

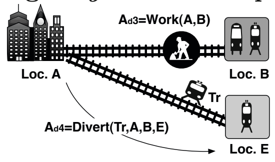
Agent x: Travel Agency



Def. $R_1 = in(Gr, A)$
$R_2 = in(Gr, B)$
$R_3 = in(Gr, C)$
$R_4 = in(Gr, D)$
$R_5 = in(Tr, A)$
$R_6 = in(Tr, B)$
$R_7 = in(Bs, B)$
$R_8 = in(Bs, C)$
$R_9 = dest(Tr, B)$
$R_{10} = dest(Bs, C)$

Initial situation: $R_1(S_0^x), R_5(S_0^x), R_7(S_0^x), R_9(S_0^x), R_{10}(S_0^x)$
Norms: $F_{R_4}(S_0^x)$ **Goals**: $\Psi(s) = \psi_1(s)$ where $\psi_1 = R_3 \vee R_4$
Plan: $\vec{P}^x = \langle [2]begin(A_{d1}), [8]end(A_{d1}), [9]begin(A_{d2}), [12]end(A_{d2})\rangle$
Situations: $S_0^x; S_1^x = do(begin(A_{d1}), S_0^x); S_2^x = do(end(A_{d1}), S_1^x);$
$S_3^x = do(begin(A_{d2}), S_2^x); S_4^x = do(end(A_{d2}), S_3^x)$
Time: $start(S_0^x) = 0, start(S_1^x) = 2, start(S_2^x) = 8, start(S_3^x) = 9, start(S_4^x) = 12$

Agent y: Train Company



Def.
$R_{11} = altDest(Tr, E)$
$R_{12} = in(Tr, E)$
$R_{13} = singleTrack(A, B)$
$R_{14} = safeRail(A, B)$
$R_{15} = problems(A, B)$

Initial situation: $R_5(S_0^x), R_9(S_0^x), R_{11}(S_0^x), R_{13}(S_0^x), R_{15}(S_0^x)$
Norms: $O_{occ(A_{d4})}(S_2^y, S_3^y)$ **Goals**: $\Psi(s) = \psi_2(s)$ where $\psi_2 = R_{14}$
Plan: $\vec{P}^y = \langle [1]begin(A_{d3}), [2]begin(A_{d4}), [7]end(A_{d4}), [22]end(A_{d3})\rangle$
Situations: $S_0^y; S_1^y = do(begin(A_{d3}), S_0^y); S_2^y = do(begin(A_{d4}), S_1^y);$
$S_3^y = do(end(A_{d4}), S_2^y); S_4^y = do(end(A_{d3}), S_3^y)$
Time: $start(S_0^y) = 0, start(S_1^y) = 1, start(S_2^y) = 2, start(S_3^y) = 7; start(S_4^y) = 22$

Fig. 2. Scenario of Example

ATK3.3 claiming that a norm obliges y to divert the train when railway works are scheduled, A_{d3}, and the path between A and B is a single track railway R_{13}.

5 Discussion and Conclusions

In this paper we have presented a model for arguments that contributes in deliberative dialogues based on argumentation schemes for arguing about norms and actions in a multi-agent system. Argumentation for deliberative dialogues has been the topic of numerous research efforts over the last few years [1,8]. Atkinson et al. [1] proposed an approach for practical reasoning based on argumentation schemes, accompanied by a set of critical questions that allow agents to evaluate the outcomes on the basis of the social values highlighted by the arguments. More recent work [3] considers temporal aspects regarding the consequences of performing an action on a subsequent action. Belesiotis et al. [2] have explored the use of situation calculus as a language to present arguments about a common plan in a multi-agent system. Existing research, however, does not adequately address the requirements of applications where agents are concerned with agreeing joint plans. It focuses on the choice of the best of a set of mutually exclusive actions to perform. In contrast we consider a team dialogue focussed upon what is the best course of action to adopt based on the integration of individual agent plans with different objectives. Furthermore, team members might have internal norms which guide the choice of the plan and they should be considered as part of the discussion about a course of action. Our aim is to construct a coherent model for referring to multi-agent plans that considers norm and plan-constraints.

In future research, the aim is to evaluate the strength of this model against the quality of the plan. We will also consider the integration of the model on a rigours formalised dialogue. In this paper, however, we have demonstrated through examples, how this model allows agents to clarify conflicts in different courses of action and facilitate the exchange of information about joint plans.

Table 3. Example of Argumentation-Based Dialogue

1. $Propose(x, y, Perform(x, A_{d1}, 2, 8))$ **2.** $Reject(y, x, Perform(x, A_{d1}, 2, 8))$
3. $Argue(x, y,$ ArgI$)$ - **ArgI**: $\langle Hold(x, \{R_1, R_5, R_9\}, 2), Perform(x, A_{d1}, 2, 8),$
$Hold(x, \{\neg R_1, \neg R_5, R_2, R_6\}, 8), achieve(x, \psi_1) \Rightarrow Perform(x, A_{d1}, 2, 8)\rangle$
4. $Why(y, x, Perform(x, A_{d1}, 2, 8))$ **5.** $Argue(x, y,$ ATK2.3$)$ - **ATK2.3**:
$\langle Hold(x, \{R_1, R_5, R_9\}, 2) Perform(x, A_{d1}, 2, 6), Hold(x, \{\neg R_1, \neg R_5, R_2, R_6\}, 8),$
$cLink(R_2, A_{d1}, A_{d2}), Perform(y, A_{d1}, 2, 8), Perform(y, A_{d2}, 9, 12),$
$occOver_{\mathcal{T}}(A_{d1}, 2, 12) \Rightarrow Perform(y, A_{d1}, 2, 8) \wedge Perform(y, A_{d2}, 9, 12)\rangle$
6. $Argue(y, x,$ ATK1.1$)$ - **ATK1.1**: $\langle Hold(x, \{R_1, R_5, R_9\}, 2), Perform(x, A_{d1}, 2, 8),$
$Hold(x, \{\neg R_1, \neg R_5, R_2, R_6\}, 8), Hold(y, \{R_5, R_9, R_{11}\}, 2), \neg Perform(y, A_{d4}, 2, 7),$
$\neg Hold(y, \{\neg R_5, \neg R_6, R_{12}\}, 7), \neg achieve(y, \psi_2), occOver_{\mathcal{T}}(A_{d1}, 2, 7) \Rightarrow$
$\neg Perform(x, A_{d1}, 2, 8) \wedge Perform(y, A_{d4}, 2, 7)\rangle$
7. $Why(y, x, Perform(y, A_{d4}, 2, 7))$ **8.** $Argue(y, x,$ ATK3.3$)$ - **ATK3.3**:
$\langle Hold(y, R_{13}, 1), Perform(y, A_{d3}, 1, 22), Obliges : Perform(x, A_{d4}, 2, 7) \Rightarrow$
$Perform(x, A_{d4}, 2, 7)\rangle$ **9.** $Withdraw(x, y, Perform(x, A_{d1}, 2, 8))$

Acknowledgments. This research is supported by the award made by the RCUK Digital Economy programme to the dot.rural Digital Economy Hub; award reference: EP/G066051/1.

References

1. Atkinson, K., Bench-Capon, T.: Practical reasoning as presumptive argumentation using action based alternating transition systems. Artificial Intelligence 171(10-15), 855–874 (2007)
2. Belesiotis, A., Rovatsos, M., Rahwan, I.: A Generative Dialogue System for Arguing about Plans in Situation Calculus. In: McBurney, P., Rahwan, I., Parsons, S., Maudet, N. (eds.) ArgMAS 2009. LNCS, vol. 6057, pp. 23–41. Springer, Heidelberg (2010)
3. Bench-Capon, T., Atkinson, K.: Action-state semantics for practical reasoning. In: Proceeding of the Fall Symposium on the Uses of Computational Argument (2009)
4. Demolombe, R., Pozos-Parra, P.: The Chisholm Paradox and the Situation Calculus. In: Hacid, M.-S., Murray, N.V., Raś, Z.W., Tsumoto, S. (eds.) ISMIS 2005. LNCS (LNAI), vol. 3488, pp. 425–434. Springer, Heidelberg (2005)

5. McBurney, P., Hitchcock, D., Parsons, S.: The eightfold way of deliberation dialogue. International Journal of Intelligent Systems 22(1), 95–132 (2007)
6. Oren, N., Luck, M., Miles, S., Norman, T.J.: An argumentation inspired heuristic for resolving normative conflict. In: Proceedings of the Fifth Workshop on Coordination, Organizations, Institutions and Norms in Agent Systems (2008)
7. Pinto, J.A., Reiter, R.: Reasoning about time in the situation calculus. Annals of Mathematics and Artificial Intelligence 14, 251–268 (1995)
8. Rahwan, I., Amgoud, L.: An argumentation-based approach for practical reasoning. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 347–354 (2007)
9. Reiter, R.: Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems. MIT Press (2001)
10. Walton, D.N.: Argumentation schemes for presumptive reasoning. Lawrence Erlbaum Associates (1996)

Preference-Based Argumentation Handling Dynamic Preferences Built on Prioritized Logic Programming

Toshiko Wakaki

Shibaura Institute of Technology
307 Fukasaku, Minuma-ku, Saitama-city, Saitama, 337–8570 Japan
twakaki@sic.shibaura-it.ac.jp

Abstract. To treat dynamic preferences correctly is crucially required in the fields of argumentation as well as nonmonotonic reasoning. To meet such requirements, first, we propose a *hierarchical* Prioritized Logic Program (or a *hierarchical* PLP, for short), which enhances the formalism of Sakama and Inoue's PLP so that it can represent and reason about dynamic preferences. Second, using such a hierarchical PLP as the underlying language, the proposed method defines the preference-based argumentation framework (called the dynamic PAF) built from it. This enables us to argue and reason about dynamic preferences in argumentation. Finally we show the interesting relationship between semantics of a hierarchical PLP given by *preferred* answer sets and semantics of the dynamic PAF given by $\mathcal{P}$-extensions.

1 Introduction

In the field of argumentation, Dung's theory of abstract argumentation [10] has gained wide acceptance and become the basis for the implementation of concrete formalisms. Dung [10] showed that argumentation can be viewed as a special form of logic programming with negation as failure and gives a series of theorems that relate semantics of logic programs and semantics of argumentation.

Recently, several approaches to generalize Dung's theory so as to handle preferences have been proposed because preferences are useful in solving conflicts between arguments or conflicts between rules. It is well-known that there are two kinds of preferences, that is, static preferences and dynamic ones [8]. Preferences are static if it is fixed at the time the argumentation theory (or the logic program) is specified, whereas they are dynamic if preferences themselves are derived as conclusions within the arguing system (or reasoning system).

So far a significant number of studies [8] have been done w.r.t. handling preferences in logic programming based on answer set semantics [11,12]. For example, Brewka and Eiter's approach [4], Eiter *et al.*'s approach [5] and Sakama and Inoue's prioritized logic programming (or a PLP, for short) [18] handle static preferences, whereas Delgrande and Schaub's ordered logic programs [7] can reason about dynamic preferences. On the other hand, in the field of argumentation, for example, Amgoud and Cayrol's approach [1] handles static preferences on arguments in a preference-based argumentation framework, whereas Prakken's approach [16], Modgil's approach [14] and Modgil and Prakken's approach [15] handle dynamic preferences in their respective formalisms of argumentation.

Recently Amgoud and Vesic [2] pointed out that, there is a critical problem such that extensions are not ensured to be conflict-free w.r.t. the attack relation for existing preference-based argumentation frameworks such as Amgoud and Cayrol's approach [1] handling static preferences, Modgil's approach [14] handling dynamic preferences and so on.

Hence to handle static preferences correctly in the theory of argumentation, recently in our previous work [19], we proposed a new approach of an abstract preference-based argumentation framework (an abstract PAF, for short) whose semantics ensures requirements of *conflict-freeness* along with *generalization* to recover Dung's acceptability semantics in the case that preferences are not available. Moreover, we presented the non-abstract PAF along with its semantics whose underlining language is a PLP [18] that handles static preferences.

On the other hand, it is well-known especially in legal reasoning, preferences may be based on some basic principles (e.g. Lex Posterior, Lex Superior and so on as addressed in [16,3]) regulating how conflicts among rules are to be resolved, and there often exists a meta-preference between preferences based on basic principles. In these cases, the assessment of such preferences, i.e. handling of dynamic preferences becomes a crucial part of the reasoning or arguing.

Therefore, the aim of this paper is to propose a new approach for a preference-based argumentation framework capable of handling dynamic preferences while enhancing the expressive power of a PLP as its underlying language. To this end, in this paper, first, we propose a *hierarchical* Prioritized Logic Program (or a *hierarchical* PLP, for short), which extends the formalism of a PLP so that it can represent and reason about dynamic preferences between not only rules at the object-level but also between meta rules at any meta-level based on the newly introduced general inference rules together with the technique of *meta-programming*. Second, using such a hierarchical PLP as the underlying language, the proposed method defines the non-abstract preference-based argumentation framework capable of arguing and reasoning about dynamic preferences in argumentation (called the dynamic PAF) built from it. Finally we show the interesting relationship between semantics of a hierarchical PLP given *preferred* answer sets and semantics of the dynamic PAF given by $\mathcal{P}$-extensions.

This paper is organized as follows: Section 2 gives the preliminaries. Section 3 presents a hierarchical PLP handling dynamic preferences and shows its semantics. Section 4 presents the dynamic PAF translated from a hierarchical PLP along with the semantics. Section 5 is devoted to discussion and the conclusion.

2 Preliminaries

2.1 Prioritized Logic Programs and Preferred Answer Sets

Definition 1. *An extended logic program (ELP) is a set of rules of the form:*

$$L \leftarrow L_1, \ldots, L_m, notL_{m+1}, \ldots, notL_n, \tag{1}$$

where L and L_i's $(0 \leq i \leq n,\ n \geq m)$ are literals, i.e. either atoms or atoms preceded by the classical negation sign $\neg$. The symbol "not" denotes the negation as failure (NAF) operator. For a rule r of the form (1), we call L the head of the rule, head(r), and $\{L_1, \ldots, L_m, notL_{m+1}, \ldots, notL_n\}$ the body of the rule, body(r). Especially, $body^+(r)$, $body^-(r)$ denote $\{L_1, \ldots, L_m\}$, $\{L_{m+1}, \ldots, L_n\}$ respectively. The head is possibly empty. We often write $L \leftarrow body(r)$ instead of (1) by using $body(r)$. For a rule with an empty body, we may write L instead of $L \leftarrow$. As usual, a rule with variables stands for the set of its ground instances.

The semantics of ELPs is given by the *answer sets semantics* [12,11]. Let P be an ELP and Lit_P be the set of ground literals in the language of P. Then an answer set S of P is the subset of Lit_P whose detailed definition is omitted due to space limitation. An answer set is *consistent* if it is not Lit_P. A program P is *consistent* if it has a consistent answer set; otherwise, P is *inconsistent*. An ELP P (*skeptically*) *entails* a literal L, written as $P \models L$, if a literal L is included in every answer set of P.

A prioritized logic program (PLP) [18] is defined as follows.

Definition 2. *(**Prioritized Logic Programs, PLPs**) Let P be an ELP. A reflexive and transitive relation $\preceq$ is defined on Lit_P. For any element e_1 and e_2 from Lit_P, $e_1 \preceq e_2$ is called a priority, and we say e_2 has a higher priority than e_1. We write $e_1 \prec e_2$ if $e_1 \preceq e_2$ and $e_2 \npreceq e_1$.*

A prioritized logic program (PLP, for short) is defined as a pair (P, Φ), where P is an ELP [1] *and Φ is a set of priorities on Lit_P.*

The declarative semantics of a PLP (P, Φ) is given by *preferred answer sets*. In what follows, Φ^* denotes the reflexive and transitive closure of Φ.

Definition 3. [18] *(**Preferred answer sets**) Given a PLP (P, Φ), let $\mathcal{AS}$ be the set of answer sets of P. Then the preference relation $\sqsubseteq_{as}$ on $\mathcal{AS}$ is defined as follows: For two answer sets S_1, S_2 of P, S_2 is preferable to S_1, written as $S_1 \sqsubseteq_{as} S_2$, if for some literal $e_2 \in S_2 \setminus S_1$,*

(i) there is a literal $e_1 \in S_1 \setminus S_2$ such that $e_1 \preceq e_2 \in \Phi^*$, and

(ii) there is no literal $e_3 \in S_1 \setminus S_2$ such that $e_2 \prec e_3 \in \Phi^*$.

Here, the relation $\sqsubseteq_{as}$ is also defined as the preorder, i.e. reflexive and transitive.

Then, an answer set S of P is called a preferred answer set (or p-answer set, for short) of (P, Φ) if $S \sqsubseteq_{as} S'$ implies $S' \sqsubseteq_{as} S$ for any answer set S' of P.

[1] In this paper, for a PLP (P, Φ), P is restrictedly given as an ELP though such P is originally allowed to be a GEDP, i.e. a member of the superclass of an ELP [18].

2.2 Argumentation Frameworks and Acceptability Semantics

An abstract argumentation framework (an abstract AF, for short)[10] is a pair $(\mathcal{A}, \mathcal{R})$, where $\mathcal{A}$ is a set of arguments and $\mathcal{R}$ is a binary relation on $\mathcal{A}$, representing attacks among arguments. A set $S \subseteq \mathcal{A}$ is *conflict-free* iff $\nexists a, b \in S$ s.t. $a\mathcal{R}b$. An argument $a \in \mathcal{A}$ is acceptable w.r.t. a set $S \subseteq \mathcal{A}$ iff $\forall b \in \mathcal{A}$ if $b\mathcal{R}a$, then $\exists c \in S$ s.t. $c\mathcal{R}b$. Based on the monotone function $F : 2^{\mathcal{A}} \rightarrow 2^{\mathcal{A}}$ such that $F(S) = \{a \mid \text{a is acceptable w.r.t. } S\}$ and conflict-freeness for a set $S \subseteq \mathcal{A}$, Dung's *Acceptability Semantics* such as `complete` (resp. `stable`, `preferred`, `grounded`) semantics is defined by the respective extensions.(See details in [10].)

Non-abstract argumentation formalisms for ELPs are briefly shown as follows.

Definition 4. (Non-Abstract Argumentation Frameworks) *Let P be an extended logic program whose rules have the form (1). An argument associated with P [17] is a finite sequence $Ag = [r_1; \ldots; r_n]$ of ground instances of rules $r_i \in P$ s.t. for every $1 \leq i \leq n$, $L_j \in body^+(r_i)$ implies the existence of a rule $r_k \in Ag$ s.t. $i < k$ and $head(r_k) = L_j$. We call the head of the first rule r_1 the claim of an argument Ag as written $claim(Ag)$. An argument Ag is minimal if it is minimal for its claim, i.e. $claim(Ag)$. The set of minimal arguments associated with P is denoted by $Args_P$. Let $attacks_P$ be the binary relation over $Args_P$ constructed according to some notion of attack such as "rebut", "undercut", "attack"[16,17]. Dung's acceptability semantics is also given as the set of extensions w.r.t. the non-abstract $AF_P = (Args_P, attacks_P)$ instantiating AF using an ELP P.*

2.3 Preference-Based Argumentation Capturing Prioritized LP

Definition 5. *(***Preference-based Argumentation Framework, PAF***)* [2] *For any element a_1 and a_2 from a set $\mathcal{A}$ of arguments, $a_1 \leq a_2$, or equivalently $(a_1, a_2) \in \leq$, is called a priority, and we say a_2 has a higher priority than a_1. We write $a_1 < a_2$ if $a_1 \leq a_2$ and $a_2 \not\leq a_1$. An abstract preference-based argumentation framework (an abstract PAF, for short) is a tuple PAF=($\mathcal{A}$,$\mathcal{R}$,$\leq$), where $\mathcal{A}$ is a set of arguments, $\mathcal{R}$ is an attack relation on $\mathcal{A}$, and $\leq$ is a preorder (i.e., a reflexive and transitive relation) on $\mathcal{A}$, called a priority relation.*

The semantics of PAF is given by $\mathcal{P}$-extensions [19] as follows.

Definition 6. *(***$\mathcal{P}$-extensions of PAF***)* [19] *Given PAF=($\mathcal{A}$, $\mathcal{R}$, $\leq$) and $Sname \in \{$`complete`, `stable`, `preferred`, `grounded`$\}$, let $\mathcal{E}$ be the set of extensions for $AF = (\mathcal{A}, \mathcal{R})$ under $Sname$ semantics. Then the preference relation $\sqsubseteq_{ex}$ over $\mathcal{E}$ (i.e., $\sqsubseteq_{ex} \subseteq \mathcal{E} \times \mathcal{E}$) is defined as follows. For two $Sname$ extensions E_1, E_2 from $\mathcal{E}$, E_2 is preferable to E_1, i.e. $E_1 \sqsubseteq_{ex} E_2$, if for some argument $a_2 \in E_2 \setminus E_1$,*

(i) there is an argument $a_1 \in E_1 \setminus E_2$ such that $a_1 \leq a_2$ w.r.t. $\leq$, and

(ii) there is no argument $a_3 \in E_1 \setminus E_2$ such that $a_2 < a_3$ w.r.t. $\leq$.

$\sqsubseteq_{ex}$ is also defined as reflexive and transitive. Then a $Sname$ extension $E \in \mathcal{E}$ (e.g. a preferred extension) is called a $Sname$ $\mathcal{P}$-extension (e.g. a preferred $\mathcal{P}$-extension) of PAF if $E \sqsubseteq_{ex} E'$ implies $E' \sqsubseteq_{ex} E$ for any $E' \in \mathcal{E}$.

Definition 7. *(***Non-abstract PAFs translated from PLPs** *)* [19]
Given a PLP (P, Φ) and Sname $\in \{\texttt{complete}, \texttt{stable}, \texttt{preferred}, \texttt{grounded}\}$, the non-abstract preference-based argumentation framework (i.e. non-abstract PAF) translated from the PLP is defined as $PAF(P, \Phi) = (Args_P, attacks_P, \leq)$, where $\leq$ is a priority relation defined as a preorder on $Args_P$ such that, $Ag_1 \leq Ag_2$ iff $e_1 \preceq e_2 \in \Phi^*$ for $claim(Ag_1) = e_1$ and $claim(Ag_2) = e_2$.   where Φ^* is the set of priorities which are reflexively or transitively derived using priorities in Φ. The semantics of our $PAF(P, \Phi) = (Args_P, attacks_P, \leq)$ is also given by the instantiated Sname $\mathcal{P}$-extensions.

Theorem 1. [19] *Let P be an ELP without integrity constraints, $PAF(P, \Phi) = (Args_P,\ attacks_P, \leq)$ be the non-abstract PAF associated with a PLP (P, Φ), where $attacks_P$ is the binary relation over $Args_P$ defined according to undercut. Then S is a preferred answer set (p-answer set) of a PLP (P, Φ) iff there is a stable $\mathcal{P}$-extension E of $PAF(P, \Phi)$ such that $S = claims(E)$, where $claims(E) = \{L | L = claim(Ag)\ for\ Ag \in E\}$.*

3 Prioritized LP for Handling Dynamic Preferences

To meet the requirements of handling dynamic preferences as addressed in the Introduction, the formalism of a PLP (P, Φ) is extended so that it can express preferences on rules as well as on meta rules along with preferences on literals. To this end,

1. We use a set N of rule names together with a naming function *name* to be able to refer to particular rules. To simplify our notation, for a particular rule r, we may write n_r instead of $name(r)$.
2. We use a 3-ary predicate symbol $\preceq$, whose atom, $\preceq (n_{r_1}, n_{r_2}, n_\delta)$ called a *a priority atom* denotes that a rule r_2 with the name $n_{r_2} \in N$ has the higher priority than a rule r_1 with the name $n_{r_1} \in N$ due to a rule δ with the name $n_\delta \in N$. The symbol $\preceq$ has its intended meaning, i.e. represents a preorder relation which is reflexive and transitive w.r.t. first and second arguments as follows,
$$\preceq (n, n, \eta) \leftarrow, \qquad \text{for } n, \eta \in N$$
$$\preceq (X, Z, t(U, V)) \leftarrow \preceq (X, Y, U), \preceq (Y, Z, V).$$
where $t(U, V)$ denotes the name of the rule transitively derived from two rules named U and V with using the mapping function t.

In our approach, we consider a *hierarchical* PLP (P, Ψ, Φ) such that $\Psi = \bigcup_{i \geq 1} \Psi^{(i)}$, where P and $\Psi^{(i)}$ $(i \geq 1)$ are ELPs and i denotes the level number of the hierarchy for $\Psi^{(i)}$. P consists of rules at level 0 expressing the object level knowledge without expressing preferences or priorities, whereas $\Psi^{(i)}$ consists of rules at level i $(i \geq 1)$ called *priority rules*. Each rule in $\Psi^{(i)}$ is named by some name $\delta \in N$ and has *a priority atom* expressed by $\preceq (n_{r_1}, n_{r_2}, n_\delta)$ as its head, where $r_1, r_2 \in \Psi^{(i-1)}$ and $\delta \in \Psi^{(i)}$. That is, the rule δ named n_δ whose head is

$\preceq (n_{r_1}, n_{r_2}, n_\delta)$ is the one level higher meta rule w.r.t. r_1, r_2 with the respective names n_1, n_2, and gives a precedence for them.

Besides it holds that, for rules $r, r' \in P$ with names $n_r, n_{r'} \in N$,

$$r \preceq r' \quad \text{iff} \quad head(r) \preceq head(r') \quad \text{iff} \quad \preceq (n_r, n_{r'}, n_\delta), \quad \text{for } n_\delta \in N$$

where $r \preceq r'$ denotes that a rule r' has the higher priority than a rule r. Then in order not to use redundant symbols, we use $n_r \in N$ (resp. $n_{r'} \in N$) instead of $head(r) \in Lit_P$ (resp. $head(r') \in Lit_P$) w.r.t. a named rule from P for expressing priorities between literals contained in Φ.

In the following, we extend a PLP(P, Φ) into a PLP (P, Ψ, Φ) with introducing a set Ψ of rules for treating dynamic preferences.

Definition 8. *(***PLPs for treating dynamic preferences***) A prioritized logic program is defined as a triple (P, Ψ, Φ), where P is an ELP whose rule has the form (1) with no name or (2) with a name $n_r \in N$ as follows:*

$$L \leftarrow L_1, \dots, L_m, notL_{m+1}, \dots, notL_n, \tag{1}$$

$$n_r: \quad L \leftarrow L_1, \dots, L_m, notL_{m+1}, \dots, notL_n, \tag{2}$$

expressing that $name(L \leftarrow L_1, \dots, L_m, notL_{m+1}, \dots, notL_n) = n_r$, Ψ is an ELP whose rule named $n_\delta \in N$ has the form (2) with a priority atom $\preceq (n_1, n_2, n_\delta)$ as its head as follows:

$$n_\delta: \quad \preceq (n_1, n_2, n_\delta) \leftarrow L_1, \dots, L_m, notL_{m+1}, \dots, notL_n,$$

where L, L_is are literals not expressing priorities or preferences, and Φ is a set of priorities to define a preorder relation $\preceq$ on the set $\mathcal{L}$ of literals as follows:

$$\mathcal{L} \overset{def}{=} Lit_P \cup N_0 \setminus \{L | L = head(r) \ for \ a \ rule \ r \in P \ with \ the \ name \ n_r\},$$

where $N_0 \subseteq N$ is the set of names used for naming a rule from P. Note that, for $e \preceq e' \in \Phi$, one of e and e' may be $e \in N_0$ or $e' \in N_0$, but not both. That is, for $n_r, n_{r'} \in N_0$, $n_r \preceq n_{r'}$ is not allowed as a member of Φ but it should be written as $\preceq (n_r, n_{r'}, n_\alpha) \in \Psi$ for some $n_\alpha \in N$. When Φ is empty, we may write a PLP (P, Ψ) instead of a PLP $(P, \Psi, \emptyset)$.

The declarative semantics of a PLP (P, Ψ, Φ) is given by *preferred answer sets.* Roughly speaking, the basic idea of our approach is that, incorporating the set Π of general inference rules for reasoning about dynamic preferences, the *strict* preference relation $\prec$ over a set of rules can be *skeptically* inferred from Π along with domain-specific knowledge P as well as Ψ expressing *preorder* preferences on rules based on priority atoms. Hereafter, we write

$$S|_X \overset{def}{=} S \cap X \qquad \text{for an answer set } S \text{ and a set } X,$$

(called X-*projection of* S) for notational convenience.

Definition 9. *Given a PLP (P, Ψ, Φ), let Π be the set of domain-independent rules defined as follows,*

$$\begin{aligned}\Pi: \ & \prec (X, Y) \leftarrow \preceq (X, Y, Z), \preceq (Y, X, U), X \neq Y, not \prec (Y, X), not \prec (Z, U),\\ & \prec (X, Y) \leftarrow \preceq (X, Y, Z), not\ ng(Y, X),\\ & ng(Y, X) \leftarrow \preceq (Y, X, U),\\ & \preceq (X, Z, t(U, V)) \leftarrow \preceq (X, Y, U), \preceq (Y, Z, V).\end{aligned}$$

where $\prec$, ng are newly introduced predicate symbols, and Δ [2] *be the set of rules constructed from $P \cup \Psi$ as follows:*

$$\Delta\text{: } n_r \leftarrow body(r). \quad \text{for a rule } r \in P \cup \Psi \text{ with a name } n_r \in N.$$

Then, Φ_d is defined as the set of strict priorities skeptically entailed by $P \cup \Psi \cup \Pi \cup \Delta$ as follows:

$$\Phi_d \stackrel{def}{=} \{(n, n') \mid P \cup \Psi \cup \Pi \cup \Delta \models \prec (n, n')\}.$$

$(n, n') \in \Phi_d$ denotes a strict priority $n \prec n'$ meaning the rule with the name n' has a strictly higher priority than the rule with the name n. Besides Φ_s is defined as the set of strict priorities derived from Φ, where Φ^* is the transitive closure of Φ.

$$\Phi_s \stackrel{def}{=} \{(e, e') \mid (e, e') \in \Phi^* \wedge (e', e) \notin \Phi^*\}.$$

Note that the head, n_r of a rule r from Δ is an atom though it is a term, i.e. an individual constant since $n_r \in N$. Thus a *meta-programming* technique is used in our approach, which enables us to establish the meta-reasoning about dynamic preferences. In what follows, $\prec$ is defined as $(\Phi_d \cup \Phi_s)^*$. which is the transitive closure of $\Phi_d \cup \Phi_s$ ensuring a strict partial order, i.e. an irreflexive, transitive and anti-symmetric relation. As usual, we write $e \prec e'$ iff $(e, e') \in \prec$.

Definition 10. *(***Preferences between answer sets***) Given a PLP (P, Ψ, Φ), let $\mathcal{AS}$ be the set of answer sets of $P \cup \Psi \cup \Pi \cup \Delta$, and $\prec$ be $(\Phi_d \cup \Phi_s)^*$. Then the preference relation $\sqsubseteq_{as}$ over $\mathcal{AS}$ is defined as follows. For any answer sets M_1, M_2, and M_3 from $\mathcal{AS}$,*

1. *$M_1 \sqsubseteq_{as} M_1$.*
2. *$M_1 \sqsubseteq_{as} M_2$ if for some literal $e_2 \in M_2 \setminus M_1$,*
   (i) there is a literal $e_1 \in M_1 \setminus M_2$ such that $e_1 \prec e_2$ [3]*, and*
   (ii) there is no literal $e_3 \in M_1 \setminus M_2$ such that $e_2 \prec e_3$.
3. *if $M_1 \sqsubseteq_{as} M_2$ and $M_2 \sqsubseteq_{as} M_3$, then $M_1 \sqsubseteq_{as} M_3$.*

$\sqsubseteq_{as}$ is the preorder relation since it is reflexive and transitive according to items no.1 and no.3. We write $M_1 \sqsubset_{as} M_2$ if $M_1 \sqsubseteq_{as} M_2$ and $M_2 \not\sqsubseteq_{as} M_1$.

Definition 11. *(***Preferred answer sets***) Let (P, Ψ, Φ) be a PLP and $\mathcal{AS}$ be the set of answer sets of $P \cup \Psi \cup \Pi \cup \Delta$. Then, an answer set S of P such that $S = M|_{Lit_P}$ for $M \in \mathcal{AS}$ is called a preferred answer set (or p-answer set, for short) of (P, Ψ, Φ) if $M \not\sqsubset_{as} M'$ (with respect to Φ and Ψ) for any answer set $M' \in \mathcal{AS}$.*

Example 1. Let us consider the PLP (P, Ψ_i) $(1 \leq i \leq 3)$, where P and Ψ_i are ELPs shown as follows: [4]

[2] The form of a rule from Δ is introduced to express *priorities between rules* in [18].

[3] Semantics of a PLP$(P, \emptyset, \Phi)$ is slightly different from a PLP (P, Φ) since $e_1 \prec e_2$ is used instead of $e_1 \preceq e_2$ in Definition 10.

[4] This example is slightly modified one from Example 8 shown in [14].

$$\begin{array}{llll} P: & n_1: \ a \leftarrow not\ \neg a, & & n_2: \ \neg a \leftarrow not\ a, \\ & n_3: \ b \leftarrow a, not\ \neg b, & & n_4: \ \neg b \leftarrow \neg a, not\ b. \end{array}$$

$$\Psi_1 = \{n_5 : \preceq (n_2, n_1, n_5),\ n_6 : \preceq (n_1, n_2, n_6),\ \ n_7 : \preceq (n_6, n_5, n_7)\},$$
$$\Psi_2 = \Psi_1 \cup \{n_8 : \preceq (n_3, n_4, n_8)\}, \qquad \Psi_3 = \Psi_2 \cup \{n_9 : \preceq (n_4, n_3, n_9)\}.$$

Note that P has two answer sets, $S_1 = \{a, b\}$ and $S_2 = \{\neg a, \neg b\}$.
Firstly w.r.t. the PLP (P, Ψ_1), Δ_1 for the PLP is obtained as follows.

$$\begin{array}{llllll} \Delta_1: & n_1 \leftarrow not\ \neg a, & n_2 \leftarrow not\ a, & & & \\ & n_3 \leftarrow a, not\ \neg b, & n_4 \leftarrow \neg a, not\ b, & n_5, & n_6, & n_7. \end{array}$$

Then $P \cup \Psi_1 \cup \Pi \cup \Delta_1$ has two answer sets, M_1, M_2 as follows.

$$M_1 = \{a, b, n_1, n_3\} \cup H_1, \qquad M_2 = \{\neg a, \neg b, n_2, n_4\} \cup H_1$$

where $H_1 = \{\prec (n_6, n_5), \prec (n_2, n_1), \preceq (n_2, n_1, n_5), \preceq (n_1, n_2, n_6), \preceq (n_6, n_5, n_7), n_5, n_6, n_7, ng(n_1, n_2), ng(n_2, n_1), ng(n_6, n_5)\}$.

Therefore, $P \cup \Psi_1 \cup \Pi \cup \Delta_1 \models \prec (n_2, n_1)$, $\quad P \cup \Psi_1 \cup \Pi \cup \Delta_1 \models \prec (n_6, n_5)$.
Thus based on $\Phi_d = \{(n_2, n_1), (n_6, n_5)\}$, $M_2 \sqsubseteq_{as} M_1$, $M_1 \not\sqsubseteq_{as} M_2$ are derived.
Hence, $S_1 = M_1|_{Lit_P} = \{a, b\}$ is obtained as the unique p-answer set of (P, Ψ_1).

Secondly, w.r.t. the PLP (P, Ψ_2), $P \cup \Psi_2 \cup \Pi \cup \Delta_2$ with $\Delta_2 = \Delta_1 \cup \{n_8\}$ has two answer sets, M'_1, M'_2 such that $M'_1 = M_1 \cup H_2$ and $M'_2 = M_2 \cup H_2$ where $H_2 = \{\prec (n_3, n_4), \preceq (n_3, n_4, n_8), n_8, ng(n_3, n_4)\}$. Thus $\Phi_d = \{(n_2, n_1), (n_6, n_5), (n_3, n_4)\}$ is inferred similarly, which leads to $M'_2 \sqsubseteq_{as} M'_1$ and $M'_1 \sqsubseteq_{as} M'_2$. Hence, we obtain both $S_1 = M'_1|_{Lit_P} = \{a, b\}$ and $S_2 = M'_2|_{Lit_P} = \{\neg a, \neg b\}$ as p-answer sets of (P, Ψ_2). Finally w.r.t. (P, Ψ_3), $P \cup \Psi_3 \cup \Pi \cup \Delta_3$ with $\Delta_3 = \Delta_2 \cup \{n_9\}$ has four answer sets, N_1, N_2, N_3, N_4 as follows.

$$N_1 = M_1 \cup \{\prec (n_4, n_3)\} \cup H_3, \qquad N_2 = M_1 \cup \{\prec (n_3, n_4)\} \cup H_3,$$
$$N_3 = M_2 \cup \{\prec (n_4, n_3)\} \cup H_3, \qquad N_4 = M_2 \cup \{\prec (n_3, n_4)\} \cup H_3.$$

where $H_3 = \{\preceq (n_3, n_4, n_8), \preceq (n_4, n_3, n_9), n_8, n_9, ng(n_3, n_4), ng(n_4, n_3)\}$. Since Φ_d for this PLP coincides with Φ_d for the PLP (P, Ψ_1), $S_1 = N_1|_{Lit_P} = N_2|_{Lit_P} = \{a, b\}$ is similarly inferred as the unique p-answer set of this (P, Ψ_3).

Example 2. (Ex. 1 Cont.) Consider the PLP (P_1, Ψ_1, Φ) with $P_1 = P \cup \{c \leftarrow not\ a\}$ and $\Phi = \{n_1 \preceq c\}$, where $n_1 \preceq c$ is used instead of $a \preceq c$. Then $M|_{Lit_{P_1}} = \{\neg a, \neg b, c\}$ is obtained as the unique p-answer set of this (P_1, Ψ_1, Φ), where $M = \{\neg\ a, \neg b,\ c, n_2, n_4\} \cup H_1$ is one of answer sets of $P_1 \cup \Psi_1 \cup \Pi \cup \Delta_1$.

Example 3. (Gordon's Perfected Shipping Problem)
Let us consider the famous legal reasoning example from Gordon [13]. [5] The domain knowledge about possession of the ship is presented by the following P_1:

$$\begin{array}{lll} P_1: & \text{ucc:} & perf \leftarrow poss, not\ \neg perf, \\ & \text{sma:} & \neg perf \leftarrow ship, \neg file, not\ perf, \\ & & poss, \quad ship, \quad \neg file. \end{array}$$

[5] Using this example, Brewka [3] illustrates how his approach for handling dynamic preferences can resolve conflicts between laws based on well-founded semantics.

Here P_1 has two answer sets S_1 and S_2 as follows since two laws, i.e. UCC and SMA are in conflict with one another:

$$S_1 = \{perf, poss, ship, \neg file\}, \qquad S_2 = \{\neg perf, poss, ship, \neg file\}.$$

Now, there are two legal principles such as Lex Posterior (LS) and Lex Superior (LS) for resolving conflict between them along with preference information such that Lex Superior has a higher priority than Lex Posterior. Hence the problem is described as the PLP $(P_1 \cup P_2, \Psi)$, where Ψ and P_2 are expressed as follows.

$$\begin{array}{lll} \Psi\colon & lp1: & \preceq(sma, ucc, lp1) \leftarrow mR(ucc, sma), \\ & ls1: & \preceq(ucc, sma, ls1) \leftarrow fed(sma), state(ucc), \\ & lex: & \preceq(lp1, ls1, lex). \\ P_2\colon & \multicolumn{2}{l}{mR(ucc, sma), \quad fed(sma), \quad state(ucc).} \end{array}$$

In this case, Δ consists of five rules according to Definition 9 as follows:

$$\begin{array}{llll} \Delta\colon & ucc \leftarrow poss, not\ \neg perf, & sma \leftarrow ship, \neg file, not\ perf, & \\ & lp1 \leftarrow mR(ucc, sma), & ls1 \leftarrow fed(sma), state(ucc), & lex. \end{array}$$

Now, $P \cup \Psi \cup \Pi \cup \Delta$ has two answer sets, M_1, M_2 as follows.

$$M_1 = \{perf, ucc\} \cup H, \qquad M_2 = \{\neg perf, sma\} \cup H,$$

where $H = \bigcap_{i=1,2} M_i = \{\prec(ucc, sma), \prec(lp1, ls1), lp1, ls1, lex, poss, ship, \neg file, mR(ucc, sma), fed(sma), state(ucc), \preceq(sma, ucc, lp1), \preceq(ucc, sma, ls1), \preceq(lp1, ls1, lex), ng(sma, ucc), ng(ucc, sma), ng(lp1, ls1)\}$.

Therefore, $P \cup \Psi \cup \Pi \cup \Delta \models \prec(ucc, sma)$, $\quad P \cup \Psi \cup \Pi \cup \Delta \models \prec(lp1, ls1)$.
Based on $\Phi_d = \{(ucc, sma), (lp1, ls1)\}$, $M_1 \sqsubseteq_{as} M_2$ and $M_2 \not\sqsubseteq_{as} M_1$ are derived. Thus $M_2|_{Lit_P} = \{\neg perf, poss, ship, \neg file, mR(ucc, sma), fed(sma), state(ucc)\}$ is the preferred answer set of this PLP. As a result, $\neg perf$ is decided.

4 Preference-Based AF for Handling Dynamic Preferences

We show the preference-based argumentation framework handling dynamic preferences whose underlying language is a *hierarchical* PLP (P, Ψ, Φ).

Definition 12. *Given a PLP (P, Ψ, Φ) and Sname $\in$ {*`complete`*,* `stable`*,* `preferred`*,* `grounded`*}, let π be the set of rules as follows:*

$$\pi \overset{def}{=} P \cup \Psi \cup \Pi \cup \Delta.$$

$\mathcal{E}_\pi$ be the set of Sname extensions for the non-abstract argumentation framework $AF_\pi = (Args_\pi, attacks_\pi)$ associated with π under Sname semantics, and $\prec$ be the strict partial order defined as $(\Phi_s \cup \Phi_e)^*$, where Φ_s and Φ_e are sets of strict priorities defined as follows, and * denotes the transitive closure of the set.

$$\Phi_s \overset{def}{=} \{(e, e') \mid (e, e') \in \Phi^* \wedge (e', e) \notin \Phi^*\}.$$

$$\Phi_e \overset{def}{=} \{(n, n') \mid \prec(n, n') \in claims(E) \ for\ any\ extension\ E \in \mathcal{E}_\pi\} = \bigcap_{E \in \mathcal{E}_\pi} claims(E)|_\Theta.$$

where $claims(E) = \{L \mid L = claim(Ag)\ for\ Ag \in E\}$, $\Theta = \{\prec(n, n') \mid n, n' \in N\}$.

Then the non-abstract preference-based argumentation framework built from the PLP (P, Ψ, Φ) is defined as $PAF(P, \Psi, \Phi)$ called a dynamic PAF as follows:

$$PAF(P, \Psi, \Phi) \stackrel{def}{=} (Args_\pi, attacks_\pi, <)$$

where $<$ is a priority relation defined as a strict partial order over $Args_\pi$, which is derived from $\prec$ as follows. For any Ag_1, Ag_2 from $Args_\pi$,
$Ag_1 < Ag_2$ iff $e_1 \prec e_2$ w.r.t. $\prec$ for $claim(Ag_1) = e_1$ and $claim(Ag_2) = e_2$.

Given a PLP (P, Ψ, Φ), preferences between extensions are defined as follows.

Definition 13. *(***Preferences between extensions***)*
For a PLP (P, Ψ, Φ) and Sname $\in \{\texttt{complete}, \texttt{preferred}, \texttt{stable}, \texttt{grounded}\}$, let $PAF(P, \Psi, \Phi) = (Args_\pi, attacks_\pi, <)$ be the non-abstract preference-based argumentation framework built from the PLP, where $\pi = P \cup \Psi \cup \Pi \cup \Delta$, and $\mathcal{E}_\pi$ be the set of Sname extensions for AF_π associated with π under Sname semantics. Then the preference relation $\sqsubseteq_{ex}$ over $\mathcal{E}_\pi$ (i.e., $\sqsubseteq_{ex} \subseteq \mathcal{E}_\pi \times \mathcal{E}_\pi$) is defined as follows. For any extensions, E_1, E_2 and E_3 from $\mathcal{E}_\pi$,

1. *$E_1 \sqsubseteq_{ex} E_1$.*
2. *$E_1 \sqsubseteq_{ex} E_2$ if for some argument $Ag_2 \in E_2 \setminus E_1$,*
   (i) there is an argument $Ag_1 \in E_1 \setminus E_2$ such that $Ag_1 < Ag_2$, and
   (ii) there is no argument $Ag_3 \in E_1 \setminus E_2$ such that $Ag_2 < Ag_3$.
3. *if $E_1 \sqsubseteq_{ex} E_2$ and $E_2 \sqsubseteq_{ex} E_3$, then $E_1 \sqsubseteq_{ex} E_3$.*

$\sqsubseteq_{ex}$ is reflexive and transitive according to items no.1 and no.3. We say that E_2 is preferable to E_1 with respect to $<$ if $E_1 \sqsubseteq_{ex} E_2$ holds.

The semantics of $PAF(P, \Psi, \Phi)$ is given by $\mathcal{P}$-extensions as follows.

Definition 14. *(***$\mathcal{P}$-extensions***) For a PLP (P, Ψ, Φ) and Sname $\in \{\texttt{complete}, \texttt{preferred}, \texttt{stable}, \texttt{grounded}\}$, let $\mathcal{E}_\pi$ be the set of the Sname extensions for AF_π associated with $\pi = P \cup \Psi \cup \Pi \cup \Delta$ under Sname semantics. Then an extension $E \in \mathcal{E}_\pi$ is called a Sname $\mathcal{P}$-extension of $PAF(P, \Psi, \Phi) = (Args_\pi, attacks_\pi, <)$ if $E \sqsubseteq_{ex} E'$ implies $E' \sqsubseteq_{ex} E$ (with respect to $<$) for any $E' \in \mathcal{E}_\pi$. In other words, $E \in \mathcal{E}_\pi$ is a Sname $\mathcal{P}$-extension of $PAF(P, \Psi, \Phi)$ iff $E \not\sqsubset_{ex} E'$ (with respect to $<$) for any $E' \in \mathcal{E}_\pi$.*

The following theorem shows that stable $\mathcal{P}$-extensions of $PAF(P, \Psi, \Phi)$ capture preferred answer sets of a PLP (P, Ψ, Φ).

Theorem 2. *Given a PLP (P, Ψ, Φ), let $PAF(P, \Psi, \Phi) = (Args_\pi, attacks_\pi, <)$ be the non-abstract preference-based argumentation framework built from the PLP, where $\pi = P \cup \Psi \cup \Pi \cup \Delta$, and $attacks_\pi$ is the binary relation over $Args_\pi$ defined according to undercut as the notion of attack. Then S is a preferred answer set (or p-answer set) of a PLP (P, Ψ, Φ) iff there is a stable $\mathcal{P}$-extension E of $PAF(P, \Psi, \Phi)$ such that $S = claims(E)|_{Lit_P}$.* [5]

[5] This theorem is reduced to Dung's Theorem 5 shown in [9] in the case of $\Psi = \Phi = \emptyset$.

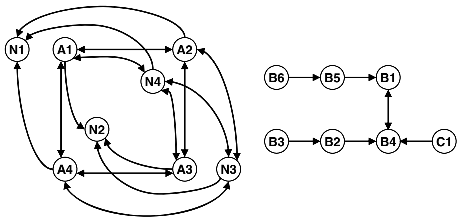


Fig. 1. The Argumentation Framework (AF) of Example 4

Proof: (sketch) This is proved in a similar way to the proof of Theorem 3 in [19].

Example 4. Consider the PLP (P, Ψ_1) given in Example 1. AF_{π_1} associated with $\pi_1 = P \cup \Psi_1 \cup \Pi \cup \Delta$ is constructed as $(Args_{\pi_1}, attacks_{\pi_1})$, where $Args_{\pi_1} = \{A_1, \ldots, A_7, B_1, \ldots, B_6, C_1, N_1 \ldots N_7\}$, whose arguments are shown as follows:

$$
\begin{aligned}
&A_1 = [a \leftarrow not\ \neg a], \quad A_2 = [\neg a \leftarrow not\ a],\\
&A_3 = [b \leftarrow a, not\ \neg b; a \leftarrow not\ \neg a], \quad A_4 = [\neg b \leftarrow \neg a, not\ b; \neg a \leftarrow not\ a],\\
&A_5 = [\preceq (n_2, n_1, n_5)], \quad A_6 = [\preceq (n_1, n_2, n_6)], \quad A_7 = [\preceq (n_6, n_5, n_7)],\\
&B_1 = [\prec (n_2, n_1) \leftarrow \preceq (n_2, n_1, n_5), \preceq (n_1, n_2, n_6), not \prec (n_1, n_2,),\\
&\qquad not \prec (n_5, n_6); \preceq (n_2, n_1, n_5); \preceq (n_1, n_2, n_6)],\\
&B_2 = [\prec (n_2, n_1) \leftarrow \preceq (n_2, n_1, n_5), not\ ng(n_1, n_2); \preceq (n_2, n_1, n_5)],\\
&B_3 = [ng(n_1, n_2) \leftarrow \preceq (n_1, n_2, n_6); \preceq (n_1, n_2, n_6)],\\
&B_4 = [\prec (n_1, n_2) \leftarrow \preceq (n_1, n_2, n_6), \preceq (n_2, n_1, n_5), not \prec (n_2, n_1),\\
&\qquad not \prec (n_6, n_5); \preceq (n_1, n_2, n_6); \preceq (n_2, n_1, n_5)],\\
&B_5 = [\prec (n_1, n_2) \leftarrow \preceq (n_1, n_2, n_6), not\ ng(n_2, n_1); \preceq (n_1, n_2, n_6)],\\
&B_6 = [ng(n_2, n_1) \leftarrow \preceq (n_2, n_1, n_5); \preceq (n_2, n_1, n_5)],\\
&C_1 = [\prec (n_6, n_5) \leftarrow \preceq (n_6, n_5, n_7), not\ ng(n_5, n_6); \preceq (n_6, n_5, n_7)],\\
&N_1 = [n_1 \leftarrow not\ \neg a], \quad N_2 = [n_2 \leftarrow not\ a], \quad N_3 = [n_3 \leftarrow a, not\ \neg b; a \leftarrow not\ \neg a],\\
&N_4 = [n_4 \leftarrow \neg a, not\ b; \neg a \leftarrow not\ a], \quad N_5 = [n_5 \leftarrow \preceq (n_2, n_1, n_5); \preceq (n_2, n_1, n_5)],\\
&N_6 = [n_6 \leftarrow \preceq (n_1, n_2, n_6); \preceq (n_1, n_2, n_6)],\\
&N_7 = [n_7 \leftarrow \preceq (n_4, n_3, n_7); \preceq (n_4, n_3, n_7)].
\end{aligned}
$$

and
$$
\begin{aligned}
attacks_{\pi_1} = \{&(A_1,A_2), (A_2,A_1), (A_1,A_4), (A_4,A_1), (A_2,A_3), (A_3,A_2), (A_3,A_4),\\
&(A_4,A_3), (A_1,N_4), (N_4,A_1), (A_3,N_4), (N_4,A_3), (A_2,N_3), (N_3,A_2), (A_4,N_3),\\
&(N_3,A_4), (N_4,N_3), (N_3,N_4), (A_2,N_1), (A_4,N_1), (N_4,N_1), (A_1,N_2), (A_3,N_2),\\
&(N_3,N_2), (B_1,B_4), (B_4,B_1), (C_1,B_4), (B_6,B_5), (B_5,B_1), (B_3,B_2), (B_2,B_4)\},
\end{aligned}
$$
where *undercut* is used as the notion of attack. Figure 1 shows the graph of AF_{π_1}, where arguments A_i, N_i $(5 \le i \le 7)$ are not depicted. Then there are two preferred as well as stable extensions, E_1, E_2 for this AF_{π_1} as follows:

$$E_1 = \{A_1, A_3, N_1, N_3\} \cup E', \qquad E_2 = \{A_2, A_4, N_2, N_4\} \cup E',$$

where $E' = E_1 \cap E_2 = \{A_5, A_6, A_7, B_1, B_3, B_6, C_1, N_5, N_6, N_7, \}$ with $claims(E') = \{\preceq (n_2, n_1, n_5), \preceq (n_1, n_2, n_6), \preceq (n_6, n_5, n_7), \prec (n_2, n_1),$
$ng(n_1, n_2), ng(n_2, n_1), \prec (n_6, n_5), n_5, n_6, n_7\}.$

For this case, $<= \{(N_2, N_1), (N_6, N_5)\}$ is derived from $\Phi_e = \{(n_2, n_1), (n_6, n_5)\}$ since $\prec (n_2, n_1), \prec (n_6, n_5) \in claims(E')$. Hence, $E_2 \sqsubseteq_{ex} E_1, \quad E_1 \not\sqsubseteq_{ex} E_2$.

As a result, E_1 is the unique preferred and stable $\mathcal{P}$-extension for this $PAF(P_1, \Psi_1)$. Note that $claims(E_1)|_{Lit_P}$=$\{a, b\}$ coincides with $M_1|_{Lit_P} = \{a, b\}$ for the p-answer set M_1 of the PLP(P_1, Ψ_1) in Example 1 as addressed by Theorem 2.

5 Discussion and Conclusions

In this paper, first, we propose a hierarchical PLP for handling dynamic preferences along with static ones in logic programming. Second, in order to treat dynamic preferences in argumentation, we present a new approach of the non-abstract PAF called a dynamic PAF built from such a hierarchical PLP. Thus the dynamic PAF built from a hierarchical PLP is regarded as an extended one from our previous PAF [19] handling static preferences built from a PLP [18].

Semantics of the proposed PAF ensures requirements of *conflict-freeness* w.r.t. the attack relation for an extension along with *generalization* to recover Dung's acceptability semantics in the case where preferences are not available [2]. Moreover, when *Direct Consistency* [6] is required as rationality postulates such that the set of conclusions of all arguments in an extension should be consistent, a $\mathcal{P}$-extension for the dynamic PAF as well as the *static* PAF [19] proposed in our approach can ensure its consistency by means of incorporating our *constrained argumentation frameworks* (CAFs) [19] into it with introducing integrity constraints whose form is $\leftarrow L, \neg L.$ for a literal $L \in Lit_P$.

Modgil's approach [14] as well as Modgil and Prakken's approach[15] treating dynamic preferences alter the respective argumentation framework by removing attacks based on preference information, which causes Modgil's approach [14] to have the critical problem such as the violation of conflict-freeness for an extension as addressed in [2]. Instead in our approach, static as well as dynamic preferences are taken into account for selecting extensions of the *unaltered* argumentation framework, which ensures conflict-freeness for a $\mathcal{P}$-extension of our PAF.

To our best knowledge, except our approach, there have been no proposed studies which relate semantics of an argumentation framework capable of handling dynamic preferences to semantics of logic programming capable of handling dynamic preferences based on answer set semantics.

References

1. Amgoud, L., Cayrol, C.: A reasoning model based on the production of acceptable arguments. Annals of Mathematics and Artificial Intelligence 34(1-3), 197–215 (2002)
2. Amgoud, L., Vesic, S.: Repairing preference-based argumentation frameworks. In: Proceedings of IJCAI 2009, pp. 665–670 (2009)
3. Brewka, G.: Well-founded Semantics for Extended Logic Programs with Dynamic Preferences. Journal of Artificial Intelligence Research 4, 19–36 (1996)
4. Brewka, G., Eiter, T.: Preferred answer sets for extended logic programs. Artificial Intelligence 109, 297–356 (1999)

5. Eiter, T., Faber, W., Leone, N., Pfeifer, G.: Computing preferred answer sets by meta-interpretation in answer set programming. Theory and Practice of Logic Programming 3(4-5), 463–498 (2003)
6. Caminada, M., Amgoud, L.: On the evaluation of argumentation formalisms. Artificial Intelligence 171(5-6), 286–310 (2007)
7. Delgrande, J.P., Schaub, T., Tompits, H.: A framework for compiling preferences in logic programs. Theory and Practice of Logic Programming 3(2), 129–187 (2003)
8. Delgrande, J.P., Schaub, T., Tompits, H., Wang, K.: A Classification and survey of preference handling approaches in nonmonotonic reasoning. Computational Intelligence 20(2), 308–334 (2004)
9. Dung, P.M.: An argumentation semantics for logic programming with explicit negation. In: Proceedings of the Tenth International Conference on Logic programming (ICLP 1993), pp. 616–630. MIT press (1993)
10. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming, and n-person games. Artificial Intelligence 77, 321–357 (1995)
11. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Proceedings of the Fifth International Conference and Symposium on Logic Programming (ICLP/SLP-1988), pp. 1070–1080. MIT Press (1988)
12. Gelfond, M., Lifschitz, V.: Classical negation in logic programs and disjunctive databases. New Generation Computing 9, 365–385 (1991)
13. Gordon, T.F.: The pleadings game: An Artificial Intelligence Model of Procedural Justice, Ph.D. thesis, TU Darmstadt (1993)
14. Modgil, S.: Reasoning about preferences in argumentation frameworks. Artificial Intelligence 173, 901–934 (2009)
15. Modgil, S., Prakken, H.: Reasoning about preferences in Structured Extended Argumentation Frameworks. In: Proceedings of COMMA 2010, pp. 347–358. IOS (2010)
16. Prakken, H., Sartor, G.: Argument-based extended logic programming with defeasible priorities. Journal of Applied Non-Classical Logics 7(1), 25–75 (1997)
17. Schweimeier, R., Schroeder, M.: A Parameterized hierarchy of argumentation semantics for extended logic programming and its application to the well-founded semantics. Theory and Practice of Logic Programming 5(1,2), 207–242 (2005)
18. Sakama, C., Inoue, K.: Prioritized logic programming and its application to commonsense reasoning. Artificial Intelligence 123, 185–222 (2000)
19. Wakaki, T.: Preference-Based Argumentation Capturing Prioritized Logic Programming. In: McBurney, P. (ed.) ArgMAS 2010. LNCS (LNAI), vol. 6614, pp. 306–325. Springer, Heidelberg (2011)

Adaption of Stepsize Parameter Using Newton's Method

Itsuki Noda

AIST, Tsukuba Univ. and Tokyo Inst. of Tech.
Tsukuba, Japan
`i.noda@aist.go.jp`

Abstract. A method to optimize stepsize parameters in exponential moving average (EMA) based on Newton's method to minimize square errors is proposed. The stepsize parameters used in reinforcement learning methods should be selected and adjusted carefully for dynamic and non-stationary environments. To find the suitable values for the stepsize parameters through learning, a framework to acquire higher-order derivatives of learning values by the stepsize parameters has been proposed. Based on this framework, the authors extend a method to determine the best stepsize using Newton's method to minimize EMA of square error of learning. The method is confirmed by mathematical theories and by results of experiments.

1 Introduction

Exponential moving averages (EMA) as shown in the following equation are widely used in a part of various learning methods, for example, to estimate values or utilities of actions and states from given examples in the several methods of reinforcement learning:

$$\tilde{x}_{t+1} = \tilde{x}_t + \alpha(x_t - \tilde{x}_t) = (1-\alpha)\tilde{x}_t + \alpha x_t, \tag{1}$$

where x_t and $\tilde{x}_t$ are given example and its estimation at time t, respectively, and α is a learning parameter called *stepsize*. For example, Q-learning [8] generally uses the following update schema of state-action values:

$$\begin{aligned} Q_{t+1}(s_t, a_t) = &(1-\alpha)Q_t(s_t, a_t) \\ &+\alpha(r_t + \gamma \max_{a'} Q_t(s_{t+1}, a')), \end{aligned} \tag{2}$$

where $Q_t(s_t, a_t)$ is an expected utility of state action a_t at s_t, and r_t is a given reward from the environment at time t. γ is a discount parameter. In this schema, $Q_t(s_t, a_t)$ corresponds to the estimated value $\tilde{x}_t$ in Eq. (1), and $r_t + \gamma \max_{a'} Q_t(s_{t+1}, a')$ corresponds to the given value x_t.

In the most cases of these schema, the stepsize parameter α is set to be a small positive number in $(0, 1)$ and decreased to be zero through learning according to equation $\alpha(t) = \frac{1}{(t+1)^\omega}$ for $\omega \in (\frac{1}{2}, 1]$ [3]. This is because EMA with a smaller stepsize estimates long-term moving average that can reduct noisy factors included in given values x_t. Such setups are reasonable for stationary environments in which target values or utilities of the estimation by Eq. (1) or 2 is fixed over time. On the other hand, in many applications of reinforcement learning, environments are non-stationary so that the target values may change over time. For example, in a resource sharing problem with multiple resources and multiple agents, an expected utility of a certain resource may change by total allocation of resources, the number of agents, and the choice policies of other agents. In such cases, the authors can not simply decrease the stepsize parameter to be zero, but should adjust it to be a suitable value according to environments.

Several works have tried to modify and adjust the stepsize parameter to be suitable for a given environment. George and Powell[4] proposed a method, called optimal stepsize algorithm (OSA), to control stepsize parameters in order to minimize noise factors on the basis of the relationships among the stepsize parameter, noise variance, and changes in learning values. Sato et. al.[7] also proposed a framework to accumulate error variance to find out the suitable learning parameters. Bonarini et. al.[1] proposed a method to switch two stepsize parameters according to the process of learning based on the similar concept of WoLF (Win or Learn First) proposed by Bowling and Veloso [2].

In order to find the suitable stepsize parameter, Noda have proposed a method called Gradient Descent Adaptation of Stepsize by Recurrent Exponential Moving Average (GDASS-REMA), in which square of difference between an estimated and given values, $(\tilde{x}_t - x_t)^2$, is minimized in each time-step by a gradient descent manner using derivatives $\frac{\partial^k \tilde{x}_t}{\partial \alpha^k}$ that is calculated by recursive exponential moving average (REMA) [5,6] . Because this method uses the gradient descent procedure, it may take a long time to converge and follow to changes of the ideal stepsize value when the environment changes drastically. In this article, the authors try to improve this method using Newton's method to find the suitable stepsize value quickly with the higher order derivatives acquired by REMA.

2 Recursive Exponential Moving Average

The Recursive Exponential Moving Average (REMA) $\xi_t^{\langle k \rangle}$ is defined as follows:

$$\begin{aligned}
\xi_t^{\langle 0 \rangle} &= x_t \\
\xi_{t+1}^{\langle 1 \rangle} &= (1-\alpha)\xi_t^{\langle 1 \rangle} + \alpha \xi_t^{\langle 0 \rangle} \\
\xi_{t+1}^{\langle k \rangle} &= (1-\alpha)\xi_t^{\langle k \rangle} + \alpha \xi_t^{\langle k-1 \rangle} .
\end{aligned} \tag{3}$$

In other words, the sequence $\xi_t^{\langle 0 \rangle}$ is identical to the given sequence x_t, and the $(k+1)$-th sequence $\xi_t^{\langle k+1 \rangle}$ is the EMA of k-th sequence $\xi_t^{\langle k \rangle}$. Therefore, $\xi_t^{\langle 1 \rangle}$ is identical to $\tilde{x}_t$.

With respect to the REMA, the following lemma and theorem about partial differentials of estimated values $\tilde{x}_t$ by the stepsize parameter α can be derived. [5,6]

Lemma 1
The first partial derivative of REMA $\xi_t^{\langle k \rangle}$ by α is given by the following equation:

$$\frac{\partial \xi_t^{\langle k \rangle}}{\partial \alpha} = \frac{k}{\alpha}(\xi_t^{\langle k \rangle} - \xi_t^{\langle k+1 \rangle}) \tag{4}$$

□

Theorem 1
The k-th partial derivative of EMA $\tilde{x}_t$ ($= \xi_t^{\langle 1 \rangle}$) is given by the following equation:

$$\frac{\partial^k \tilde{x}_t}{\partial \alpha^k} = (-\alpha)^{-k} k! (\xi_t^{\langle k+1 \rangle} - \xi_t^{\langle k \rangle}) \tag{5}$$

□

In our previous work, GDASS-REMA updates the stepsize α to the direction to decrease the following squared error between the given (x_t) and the estimated ($\tilde{x}_t$) values in each time t gradually [5,6] :

$$\mathcal{E}_t = (1/2)(\tilde{x}_t - x_t)^2. \tag{6}$$

Therefore, the actual update schema in GDASS is:

$$\alpha \leftarrow \alpha - \eta \cdot \text{sign}(\frac{\partial \mathcal{E}_t}{\partial \alpha}) = \alpha - \eta \cdot \text{sign}((\tilde{x}_t - x_t) \cdot \frac{\partial \tilde{x}_t}{\partial \alpha}).$$

3 Exponential Moving Average of Squared Error

Because Theorem 1 provides a way to calculate higher order derivatives of $\tilde{x}_t$ by α, we can get a higher order Taylor expansion of $\mathcal{E}_t$ by α as follows:

$$\mathcal{E}_t(\Delta\alpha) = \mathcal{E}_t(0) + \frac{\partial \mathcal{E}_t}{\partial \alpha}\Delta\alpha + \frac{1}{2}\frac{\partial^2 \mathcal{E}_t}{\partial \alpha^2}\Delta\alpha^2 + \frac{1}{6}\frac{\partial^3 \mathcal{E}_t}{\partial \alpha^3}\Delta\alpha^3 + \cdots.$$

Therefore, if we focus on the expansion of the first and second order terms, we will determine the optimum change of the stepsize, $\Delta\alpha^{*\langle t \rangle}$, which minimize the error at time t, using the Newton's method as follow:

$$\Delta\alpha^{*\langle t \rangle} = \frac{\left(\frac{\partial \mathcal{E}_t}{\partial \alpha}\right)}{\left(\frac{\partial^2 \mathcal{E}_t}{\partial \alpha^2}\right)}. \tag{7}$$

However, updating α using the above equation directly does not work well, because the given value x_t includes noise that should be eliminated in calculation of $\tilde{x}_t$, so that α tends to be adjusted to estimate the noise factor instead of the true value of x_t.

So, in the following sections, we focus on EMA of squared error and construct a method to minimize the averaged error by the Newton's method.

3.1 Squared Error and Derivatives

Here, we re-define the squared error shown in Eq. (6) using an error δ_t of given and estimated values, x_t and $\tilde{x}_t$, as follows:

$$\begin{aligned}\delta_t &= \tilde{x}_t - x_t \\ \mathcal{E}_t &= (1/2)\delta_t^2.\end{aligned}$$

Then, we get the following theorem.

Theorem 2
The k-th partial derivative of the squared error $\mathcal{E}_t$ by α is calculated by the following equations:

$$\frac{\partial^k \mathcal{E}_t}{\partial \alpha^k} = \sum_{i=0}^{k-1} \frac{(k-1)!}{(k-1-i)!i!} \frac{\partial^i \delta_t}{\partial \alpha^i} \frac{\partial^{k-i} \delta_t}{\partial \alpha^{k-i}}, \tag{8}$$

where,

$$\begin{aligned}\frac{\partial^0 \delta_t}{\partial \alpha^0} &= \delta_t \\ \frac{\partial^k \delta_t}{\partial \alpha^k} &= \frac{\partial^k \tilde{x}_t}{\partial \alpha^k} \quad (k > 0).\end{aligned} \tag{9}$$

□

(See Appendix for the proof.)

3.2 EMA of Squared Error and Partial Derivatives

As discussed above, our target is a method to determine the stepsize parameter α that minimizes the EMA of the squared error $\mathcal{E}_t$. Here, we define the EMA $\tilde{\mathcal{E}}_t$ as follows:

$$\tilde{\mathcal{E}}_{t+1} = (1-\beta)\tilde{\mathcal{E}}_t + \beta \mathcal{E}_t, \tag{10}$$

where β is another stepsize parameter for EMA of the squared error, and $\tilde{\mathcal{E}}_0 = 0$. This $\tilde{\mathcal{E}}_t$ is equal to the estimated variance of the expected reward value introduced in [7].

As same as the case of the squared error $\mathcal{E}_t$ shown in Eq. (7), we can estimate the optimal stepsize value $\alpha^*$to minimize $\tilde{\mathcal{E}}_t$ using the Newton's method and Taylor expansion as follows:

$$\Delta\alpha^* = \frac{\left(\frac{\partial \tilde{\mathcal{E}}_t}{\partial \alpha}\right)}{\left(\frac{\partial^2 \tilde{\mathcal{E}}_t}{\partial \alpha^2}\right)} \tag{11}$$

$$\alpha^* = \alpha - \Delta\alpha^* \tag{12}$$

On the other hand, we can accumulate higher order partial derivatives of $\tilde{\mathcal{E}}_t$ by α from Eq. (10) by the following equations:

$$\frac{\partial \tilde{\mathcal{E}}_{t+1}}{\partial \alpha} = (1-\beta)\frac{\partial \tilde{\mathcal{E}}_t}{\partial \alpha} + \beta \frac{\partial \mathcal{E}_t}{\partial \alpha} \tag{13}$$

$$\frac{\partial^2 \tilde{\mathcal{E}}_{t+1}}{\partial \alpha^2} = (1-\beta)\frac{\partial^2 \tilde{\mathcal{E}}_t}{\partial \alpha^2} + \beta \frac{\partial^2 \mathcal{E}_t}{\partial \alpha^2} \tag{14}$$

$$\frac{\partial^k \tilde{\mathcal{E}}_{t+1}}{\partial \alpha^k} = (1-\beta)\frac{\partial^k \tilde{\mathcal{E}}_t}{\partial \alpha^k} + \beta \frac{\partial^k \mathcal{E}_t}{\partial \alpha^k} \tag{15}$$

This means that these partial derivatives can be calculated by the same manner of EMA using the derivatives of the squared error, $\frac{\partial^k \mathcal{E}_t}{\partial \alpha^k}$. As shown in Eq. (8) and Eq. (9), these values can be determined systematically using REMA $\xi_t^{\langle K \rangle}$.

Finally, we get the following procedure to obtain the optimal stepsize α^* to minimize EMA of the squared error. We call it as Rapid Recursive Adaption of Stepsize Parameter by Newton's method (RRASP-N).

```
Initialize: $\forall k \in \{0 \dots k_{\max}-1\} : \xi^{\langle k \rangle} \leftarrow x_0$
            $\forall k \in \{0 \dots k_{\max}-2\} : \frac{\partial^k \tilde{\mathcal{E}}}{\partial \alpha^k} \leftarrow 0$
while forever do
  Let $x$ be an observation.
  for $k = k_{\max}-1$ to 1 do
    $\xi^{\langle k \rangle} \leftarrow (1-\alpha)\xi^{\langle k \rangle} + \alpha \xi^{\langle k-1 \rangle}$
  end for
  $\xi^{\langle 0 \rangle} \leftarrow x$
  $\delta \leftarrow \xi^{\langle 1 \rangle} - x$
  for $k = 1$ to $k_{\max}-2$ do
    Calculate $\frac{\partial^k \mathcal{E}}{\partial \alpha^k}$ by Eq. (8), Eq. (9) and Eq. (5).
    Update $\frac{\partial^k \tilde{\mathcal{E}}}{\partial \alpha^k}$ by Eq. (13)~ Eq. (15).
  end for
  if $\frac{\partial^2 \tilde{\mathcal{E}}}{\partial \alpha^2} > 0$ then
    Calculate $\Delta\alpha^*$ by Eq. (11).
    if $|\Delta\alpha^*| > \alpha$ then
      $\Delta\alpha^* \leftarrow \mathrm{sign}(\Delta\alpha^*)\alpha$
    end if
    $\alpha \leftarrow \alpha + \frac{\Delta\alpha^*}{2}$.
    if $\alpha$ is not in $[\alpha_{min}, \alpha_{max}]$ then
      let $\alpha$ be $\alpha_{min}$ or $\alpha_{max}$.
    end if
    for $k = 1$ to $k_{\max}-1$ do
      Update $\xi^{\langle k \rangle}$ according to changes of $\alpha$ using $\frac{\partial \xi^{\langle k \rangle}}{\partial \alpha}$ determined by
      Eq. (4).
    end for
  end if
end while
```

In this procedure, α is updated only when $\frac{\partial^2 \tilde{\mathcal{E}}}{\partial \alpha^2}$ is positive, because the changes of $\tilde{\mathcal{E}}$ by α is concave down in the case of $\frac{\partial^2 \tilde{\mathcal{E}}}{\partial \alpha^2} < 0$. We also cut-off $\Delta\alpha^*$ because of the following reason: The Taylor expansion of $\xi^{\langle k \rangle}$ using Eq. (5) includes the term $\left(\frac{\Delta\alpha}{\alpha}\right)^n$, which becomes huge when α is small. Therefore, truncation errors of the Taylor expansion may be large and affects other calculations in the procedure. In order to avoid such effects, we limit the absolute value of $\Delta\alpha^*$ within the value of α.

4 Experiments

In order to show the performance of RRASP-N, the authors carried out several experiments.

4.1 Exp.1: Finding Optimal Stepsize

In order to show that RRASP-N can determine the optimal stepsize α^*, the authors conducted an experiment using the following noisy random-walk as a given value x_t:

$$x_t = v_t + \epsilon_t, \tag{16}$$

where ϵ_t is a random noise whose average and standard deviation are 0 and σ_ϵ, respectively. The true value v_t is a random walk defined by the following equations:

$$v_{t+1} = v_t + \Delta v_t,$$

where Δv_t is a random step whose average and standard deviation are 0 and σ_v, respectively. Figure 1 shows an example of the noisy random-walk. In this graph, band-like spikes are the given sequence x_t, and curves at the center of the band are true value v_t and its learning result $\tilde{x}_t$.

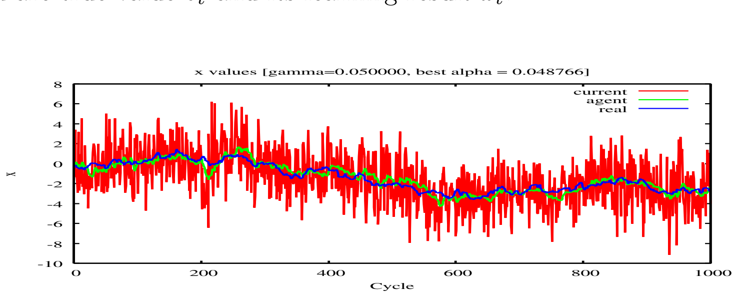
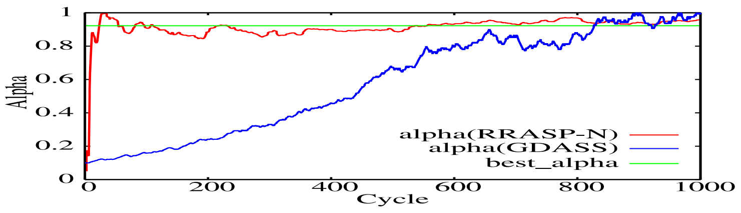
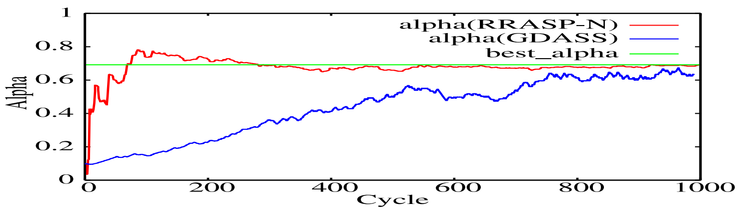


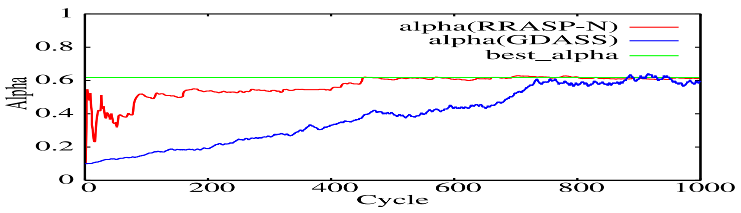
Fig. 1. Exp.1: Changes of Learned Expected Value $\tilde{x}_t$ using Acquired Stepsize α by RRASP-N



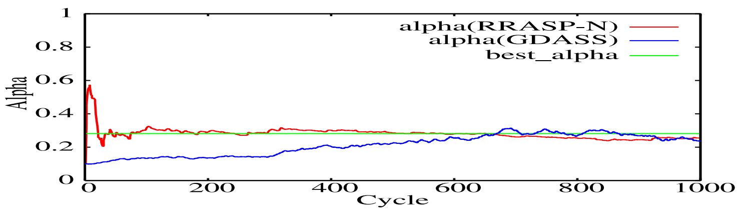
(a) $\gamma = 3.33, \alpha^* = 0.923$



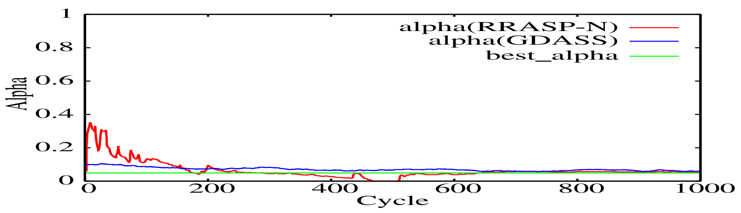
(b) $\gamma = 1.25, \alpha^* = 0.693$



(c) $\gamma = 1.00, \alpha^* = 0.618$



(d) $\gamma = 0.33, \alpha^* = 0.282$



(e) $\gamma = 0.05, \alpha^* = 0.049$

Fig. 2. Exp.1: Adjustment of Stepsize Parameter by RRASP-N (red) and GDASS (blue) for Various Ratio of Standard Deviations of Random Walk and Noise

If the standard deviations (σ_v and σ_ϵ) of the random step and noise are known, we can calculate the optimal stepsize to minimize average of square error ($|\mathcal{E}_t|$) for the noisy random walks by the following equation:

$$\alpha^* = \frac{-\gamma^2 + \sqrt{\gamma^4 + 4\gamma^2}}{2}, \tag{17}$$

where, $\gamma = \frac{\sigma_v}{\sigma_\epsilon}$ [5]. However, the standard deviations are not given generally for learning agents, so that, they must acquire it through learning like RRASP-N.

Figure 2 shows results of adaptation of α by RRASP-N for the given values x_t. Each graph of this figure indicates changes of α through learning with the optimal value of α, which indicated by a horizontal line, for different setting of the noisy random-walk. As shown in these graphs, acquired α quickly converges to the optimal value of α.

Moreover, the speed of convergence is drastically improved compared with GDASS proposed in the previous work. The red lines in figure 2 shows results of adaptation by GDASS for the same settings of the RRASP-N. While adaptation of GDASS converge to the optimal stepsize gradually as a nature of gradient decent methods, RRASP-N can adapt it to the optimal one so quickly by jumping stepsize to the optimal directly using Newton's method.

4.2 Exp.2: Repeated Multi-agent Resource Sharing

Finally, the authors conducted a learning experiment using repeated multi-agent resource sharing. In the experiment, we suppose that multiple agents share four resources (resource-0 ... resource-3). The agents are grouped into three types, *fixed users* who never change their choice from a certain resource, *random hoppers* who choose one of resources randomly every cycle, *big players* who usually stay on a certain resource but sometime change their choice, and a *learning agent* who try to estimate the average of utility for each resource. The population and weight of each group is as follows:

type	population	weight
fixed users	1	7
random hoppers	17	1
big players	2	10
learning agent	1	1

where the weight of an agent means a degree of consuming resources compared with a random hopper. Therefore, a resource that is used by big players, who has a big weight, will have a poor utility. Each resource also has its own capacity, which indicates the size of resource. In this experiment, the actual utility of a resource k at time t is calculated by the following equation:

$$\mathrm{utility}_k(t) = \frac{1}{1 + \mathrm{totalWeight}_k(t)/\mathrm{capacity}_k},$$

where, $\mathrm{totalWeight}_k(t)$ is the summation of weights of agents who choose the resource k at time t.

The purpose of the learning agent is to acquire estimation of an utility of each resource by reducing noisy factor caused by the random hoppers. In the same time, the learning agent must adapt drastic changes brought by big players' change of choice. Therefore, the agent must adapt its stepsize parameter according to changes of the environment. Figure 3 shows the result of the learning. In each graph in the right of this figure indicates given and expected utilities for each time step, while graphs in the left shows changes of stepsize parameters for each resource. Note that an independent stepsize parameter is assigned for each resource. Therefore, the parameters changes independently with each other. From these graphs, we can find that the agent can estimate suitable expected utilities by reducing noise factors of the random hoppers. Also, they can adapt drastic changes by big players. Changes of the stepsize parameters in figure 3 shows how the agent adapts to the changes of the environment.

5 Concluding Remarks

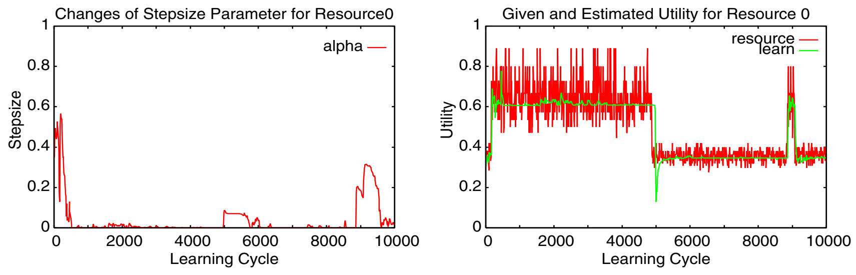
In this article, the authors proposed a method to adapt stepsize parameter, called rapid recursive adaptation of stepsize parameter by Newton's method (RRASP-N), in which EMA of the squared error of estimated value is minimized by changing the stepsize parameter. RRASP-N utilizes higher order partial derivatives of the estimated value and EMA of the squared error by the stepsize parameter, which can be calculated from recursive exponential moving average systematically.

Experimental results shows that RRASP-N responds changes of environments so quickly that it adapts the stepsize parameter to be suitable. In the same time, RRASP-N' behavior is stable because it use statistical value, EMA of the squared error. We also can apply RRASP-N to various noise model. While we use only Gaussian noise for input values, the formalization only suppose to minimize squared error between expected and given values. Actually, the situation used in Exp.2 is a non-Gaussian noise case. In this experiment, *random hoppers* provides noisy effects to the environment. As shown in the results of experiments, RRASP-N perform reasonably in such an environment.

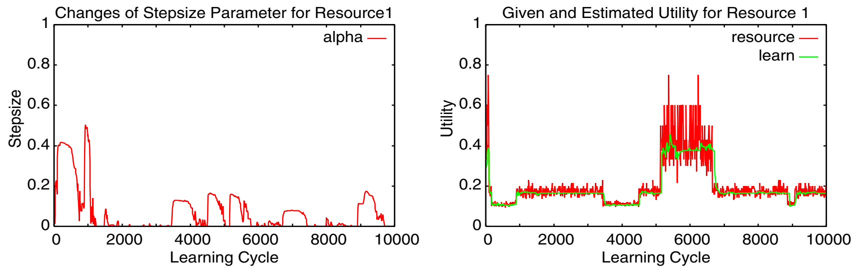
While the experiments shown in this article used only the case of single-state environmentsare, we can apply it learning in general multiple-state environments. EMA is widely used in various reinforcement learning, and the Q-learning.

There also remain several open issues that include:

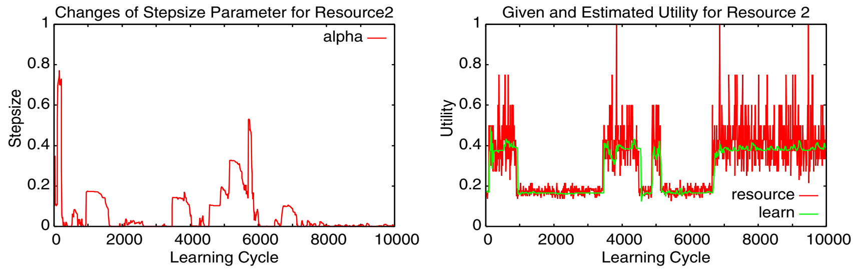
– effects of different stepsize parameters for states and actions in Q-learning.
– utilization of more higher order derivatives ($k > 2$) to analyze structures of errors function ($\tilde{\mathcal{E}}_t$) with respect to α.
– tuning of β, another stepsize parameter for the EMA of the squared error.



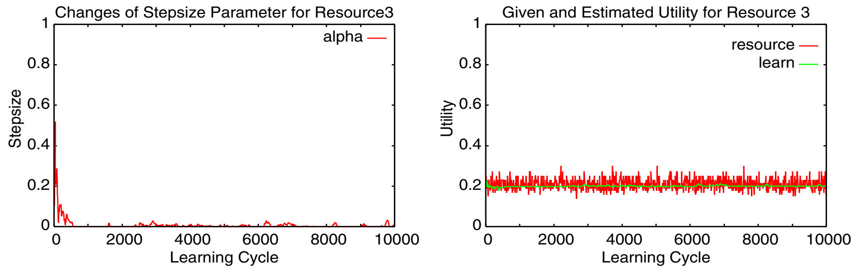
(a) Resource 0



(b) Resource 1



(c) Resource 2



(d) Resource 3

Fig. 3. Exp.2: Learning of Expected Utility of Each Resources. (left: changes of stepsize parameter α. right: given and estimated utilities for each resource.)

Acknowledgment. This work was supported by JSPS KAKENHI 21500153.

References

1. Bonarini, A., Lazaric, A., de Cote, E.M., Restelli, M.: Improving cooperation among self-interested reinforcement learning agents. In: Proc. of Workshop on Reinforcement Learning in Non-Stationary Environments. ECML-PKDD 2005 (October 2005)
2. Bowling, M., Veloso., M.: Multiagent learning using a variable learning rate. Artificial Intelligence 136, 215–250 (2002)
3. Even-dar, E., Mansour, Y.: Learning rates for q-learning. Journal of Machine Learning Research 5, 2003 (December 2003)
4. George, A.P., Powell, W.B.: Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming. Machine learning 65(1), 167–198 (2006)
5. Noda, I.: Adaptation of stepsize parameter for non-stationary environments by recursive exponential moving average. In: Prof. of ECML 2009 LNIID Workshop, ECML, pp. 24–31 (September 2009)
6. Noda, I.: Recursive Adaptation of Stepsize Parameter for Non-stationary Environments. In: Taylor, M.E., Tuyls, K. (eds.) ALA 2009. LNCS, vol. 5924, pp. 74–90. Springer, Heidelberg (2010)
7. Sato, M., Kimura, H., Kobayashi, S.: TD algorithm for the variance of return and mean-variance reinforcement learning (in japanese). Transactions of the Japanese Society for Artificial Intelligence 16(3F), 353–362 (2001)
8. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)

A Proof of Theorem 2

First of all, we show the following lemma:

Lemma 2
The partial derivative of δ_t by α is equal to the derivatives of $\tilde{x}_t$ by α:

$$\frac{\partial \delta_t}{\partial \alpha} = \frac{\partial \tilde{x}_t}{\partial \alpha} \tag{18}$$

In addition, generally, we can get the following equations for the k-th partial derivatives:

$$\frac{\partial^k \delta_t}{\partial \alpha^k} = \frac{\partial^k \tilde{x}_t}{\partial \alpha^k} \tag{19}$$

□

On the other hand, we can calculate $\frac{\partial^k \tilde{x}_t}{\partial \alpha^k}$ from REMA $\xi_t^{\langle k \rangle}$ using Theorem 1.

Therefore, we can calculate $\frac{\partial^k \delta_t}{\partial \alpha^k}$ by REMA.

Here, let's focus on the partial derivative of $\mathcal{E}_t$.

Suppose that the i-th partial derivative of $\mathcal{E}_t$ by α satisfies Eq. (8) for all $j \leq k$ as follow:

$$\frac{\partial^j \mathcal{E}_t}{\partial \alpha^j} = \sum_{i=0}^{j-1} \frac{(j-1)!}{(j-1-i)!i!} \frac{\partial^i \delta}{\partial \alpha^i} \frac{\partial^{j-i} \delta}{\partial \alpha^{j-i}}.$$

In this case, we have the $k+1$-th partial derivative as follows:

$$\frac{\partial^{k+1} \mathcal{E}_t}{\partial \alpha^{k+1}} = \sum_{i=0}^{k-1} \frac{(k-1)!}{(k-1-i)!i!} \cdot \left[\frac{\partial^i \delta}{\partial \alpha^i} \frac{\partial^{j-i+1} \delta}{\partial \alpha^{j-i+1}} + \frac{\partial^{i+1} \delta}{\partial \alpha^{i+1}} \frac{\partial^{j-i} \delta}{\partial \alpha^{j-i}} \right].$$

Here, we reform the equation by terms $\frac{\partial^i \delta}{\partial \alpha^i} \frac{\partial^{k-i+1} \delta}{\partial \alpha^{k-i+1}}$. Then its factor c_{ki} can be calcuated as follows: In the case of $i = 0$ or $i = k$,

$$c_{ki} = 1 = \frac{((k+1)-1)!}{((k+1)-1-i)!i!}.$$

In the case of $0 < i < k$,

$$\begin{aligned} c_{ki} &= \frac{(k-1)!}{(k-1-(i-1))!(i-1)!} + \frac{(k-1)!}{(k-1-i))!i!} \\ &= \frac{((k+1)-1)!}{((k+1)-1-i)!i!}. \end{aligned}$$

Therefore, Eq. (8) is satisfied in the case of the $k+1$-th partial derivative.

As the result, Eq. (8) is satisfied for all $k > 0$. ■

A Health Social Network Recommender System

Insu Song[1], Denise Dillon[2], Tze Jui Goh[3], and Min Sung[3]

[1,2] School of Business Information Technology, and Psychology
James Cook University Australia
{insu.song,denise.dillon}@jcu.edu.au
[3] Institute of Mental Health (IMH), Singapore
{Tze_Jui_Goh,Min_SUNG}@imh.com.sg

Abstract. People with chronic health conditions require support beyond normal health care systems. Social networking has shown great potential to provide the needed support. Because of the privacy and security issues of health information systems, it is often difficult to find patients who can support each other in the community. We propose a social-networking framework for patient care, in particular for parents of children with Autism Spectrum Disorders (ASD). In the framework, health service providers facilitate social links between parents using similarities of assessment reports without revealing sensitive information. A machine learning approach was developed to generate explanations of ASD assessments in order to assist clinicians in their assessment. The generated explanations are then used to measure similarities between assessments in order to recommend a community of related parents. For the first time, we report on the accuracy of social linking using an explanation-based similarity measure.

Keywords: Social networking, health social network, health informatics, recommender system.

1 Introduction

1.1 Motivation

Recently, social networking for health care has shown great potential to empower patient self-care. Examples include PatientsLikeMe[1] and the IBM Patient Empowerment System. These newly emerging patient-driven health care services facilitate information exchange and collaboration between patients and between patients and doctors. The services provided by health social networks include (a) emotional support and information sharing, (b) physician Q&As, and (c) self-tracking of a condition, its symptoms, treatment options and other biological information [16].

In this paper, we propose a social networking framework for parents of autistic children. Autism is characterized by a triad of impairments [18] in the areas of reciprocal social interaction, communication and repetitive and stereotyped behaviors. Asperger's Disorder is similar to Autism, but involves no deviance or delay in language development. Studies have demonstrated that early diagnosis can lead to better prognosis

[1] (http://www.patientslikeme.com/all/patients)

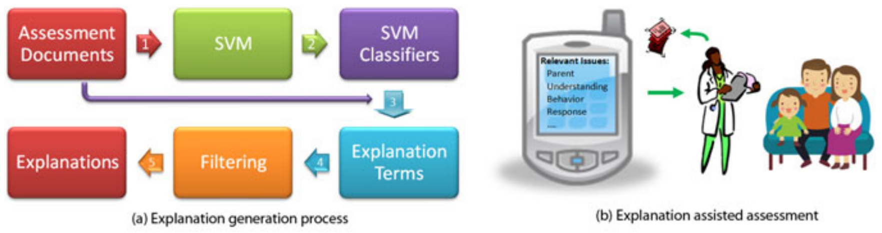


Fig. 1. (a) illustrates the overall process of generating textual explanations to classification results. (b) shows an example use of the explanation method, where a clinician make use of textual explanations to previous or current assessments, such as Autism. Mobile devices such as smart phones can be used to record interview questions and provide on the spot classification and explanations to provide more objective mental health assessments.

for children with Autism Spectrum Disorders (ASD) [1]. However, most children get diagnosed only upon entering the school environment when the behavioral difficulties become more prominent. Hence, the implementation of early surveillance and screening is crucial to identify children at risk for ASD at an earlier stage [8,7,11].

Recently, machine learning techniques such as support vector machines (SVMs) have shown significant potential for supporting the practice of medicine and psychiatric classification [5]. The application of machine learning techniques in ASD diagnosis has significant merits because of the potential to provide early diagnosis and more standardized objective diagnosis. Conventionally, expert psychiatrists consciously and unconsciously analyze the language of their patients to make a clinical diagnosis using diagnostic classification schemas, such as the DSM IV [6] and ICD 10 [9]. Although the DSM-IV and ICD-10 guidelines are helpful to clinicians in the diagnostic process, the effectiveness of their utilization depends on the experience of the clinician [12].

ASD is usually a lifelong condition such that long-term treatment planning and supports from family and communities are essential. Social networking for health care may empower parents of autistic children to share information with other parents and more easily collaborate with doctors. In this paper we develop a method of facilitating social linking of parents by similarities of assessment reports of their children. Explanations of classification results of assessment reports are used to measure similarities of the assessment reports. The experiments describe a first attempt to generate explanations of why practicing psychiatrists would have diagnosed autism cases using a decompositional approach: learned SVM model parameters are analyzed to select informative features, and then sensitivity filtering is used to select some subsets of more relevant features.

Figure 1 (a) illustrates an overview of our approach to generating explanations for psychological assessments using Support Vector Machines (SVMs). Explanation terms are extracted from assessment documents using both SVM models and classification results. Figure 1 (b) shows how our method can be used to provide explanation assisted assessment of autism and other mental health issues. For example, the explanations can highlight the main issues that were used to differentiate the particular autism case from normal cases.

1.2 Background

This work is based on social networking, text mining, text classification and, in particular, recommendation systems. The following section provides a brief overview of the core techniques, focusing on social networking, recommendation systems, support vector machines (SVMs), and the significance of generating human-comprehensible explanations from SVMs.

Social Network and Recommendation System. A health social network is an online information service which facilitates information sharing between closely related members of a community. Also known as social media on the Internet, or Health 2.0, a health social network empowers patients and health service providers by promoting collaboration between patients, their caregivers, and clinicians [14]. At its basic level, a health social network provides emotional support by allowing patients to find others in similar health situations. They can also share information about conditions, symptoms and treatments [16]. Other services include physician Q&A, and self-tracking of condition, symptom, treatment and other biological information [16]. The self-supporting community is particularly important for lifelong conditions like autism.

The main means of finding patients with similar health conditions are based on labor-intensive methods such as searching the Internet, keywords in community titles and descriptions of other members in communities [15]. Over the years, many recommender systems and similarity measurement methods have been developed [3]. The approaches can be broadly classified into two categories: content matching based on available semantic information and a collaborative filtering approach based on overlapping membership of pairs of communities [15]. Our novel approach is based on semantic information of autism assessment reports.

Support Vector Machines. Cortes and Vapnik [2] introduced support vector machines (SVMs) which are a novel approach to machine learning. SVMs are based on the structural risk minimization principle in order to overcome the overfitting problems. Support vector machines find the hypotheses out of the hypothesis space H of a learning system which approximately minimizes the bound on the actual error by controlling the empirical error using training samples and the complexity of the model using the VC-dimension of H. SVMs are very universal learning systems [10]. In their basic form, SVMs learn maximal margin hyperplanes (linear threshold functions). A hyperplane can be defined by a weight vector $\mathbf{w}$ and a bias b:

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

The corresponding threshold function for an input vector $\mathbf{x}$ is then given by:

$$f(\mathbf{x}) = sign(\mathbf{w} \cdot \mathbf{x} + b)$$

However, it is possible learn polynomial classifiers, radial basis function (RBF) networks and three or more layered neural networks by mapping input data $\mathbf{x}$ to some other (possibly infinite dimensional) feature space $\phi(\mathbf{x})$ and using kernel functions $K(\mathbf{x}_i, \mathbf{x}_j)$ to obtain dot products, $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$, of feature data.

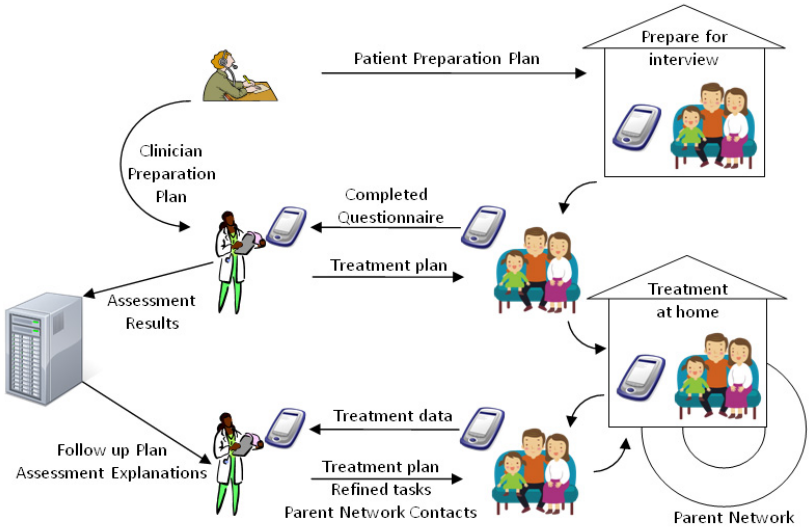


Fig. 2. Parents of autistic children collaborate with hospital and the community to share experiences and learning to address their needs. The hospital engages in the community by providing links between parents having children with similar assessment results. Families are matched based on similarity of their explanation terms.

Generating Explanations from SVMs. Much of the work that aims at providing an explanation capability to SVMs has focused on rule extraction techniques [4] following the footsteps of the earlier effort to obtain human-comprehensible rules from artificial neural networks (ANNs). One approach to classifying rule extraction methods is the translucency dimension which includes decompositional and pedagogical (or learning based) techniques as extremes [13]. The decompositional approach analyzes the internal representation of the ANN. In general, decompositional rule extraction techniques start with analyzing each individual neuron and their weight vectors to generate localized rules. Initially, the inputs and outputs of the neurons form antecedents and consequents of the rules, respectively. On the other hand, the strategy of the pedagogical approaches considers the trained ANN as a "black box" and aims at finding rules that map the ANN inputs directly to outputs [17]. For example, a decision tree can be generated from pairs of input and output values of the trained ANN.

1.3 Overview

In the next section, we propose a social networking framework for parents of autistic children that utilizes mobile phone-based ubiquitous computing. The remainder of the paper summarizes experiments and their results: text classification, explanation generation for classification results, statistical analysis on the model parameters that are generated for autism diagnosis reports, and social linking using explanation similarities.

2 Parent Network Framework

We propose a parent social-network framework, where health service providers can actively engage in facilitating information sharing and social links between parents. Figure 2 shows how hospitals can interact with communities of parents. Parents who are concerned about their children can obtain preliminary assessment tools from hospitals via their mobile phones. They can fill in a standard assessment questionnaire to get a preliminary diagnosis and to obtain information on how to get help. Upon the first consultation with a clinician, the mobile agent on the parent can provide the clinician's agent completed questionnaires and other preliminary diagnosis results improving both the effectiveness and the efficiency of the clinician. The clinician can then, via the mobile agent of the parent, provide a treatment plan and tasks that parents can follow. The assessment report is then stored in the data mining server to generate explanations and parent-link information about other parents of children with similar diagnoses. Subsequent visits to the hospital will provide more refined treatment plans and information about communities who can share their experiences and should facilitate learning in order to meet the parents' needs, such as emotional supports and clinical knowledge.

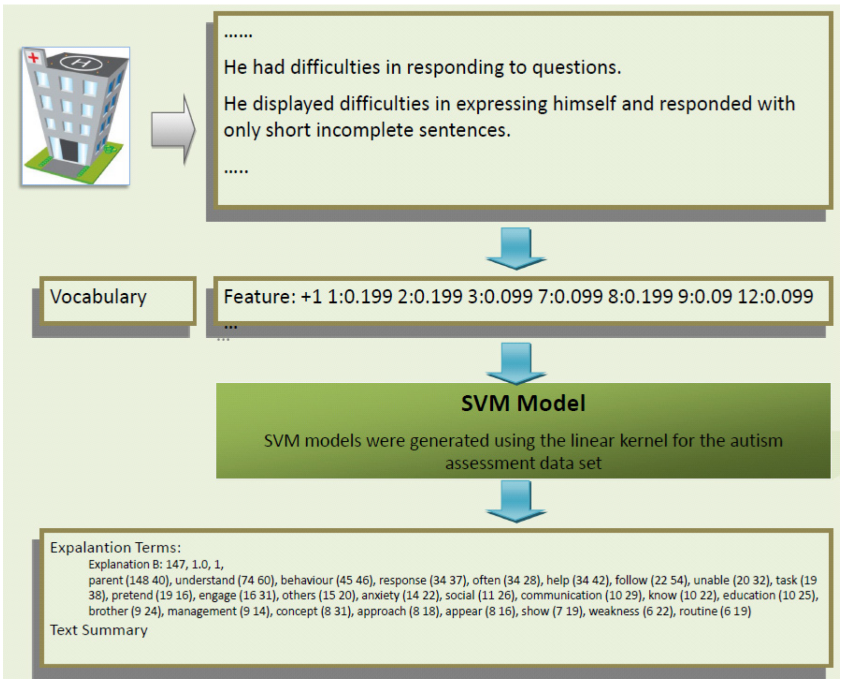
3 Experimental Evaluation

3.1 Methodology

A preliminary study has been undertaken to generate explanations of autism diagnosis reports obtained from IMH (Institute of Mental Health, Singapore). Figure 3 illustrates our method of generating explanations. The autism diagnosis reports were obtained from mental health clinics and comprise of a total of 236 reports: 217 positive cases and 19 negative cases. A small part of the observation section of an autism assessment report is shown below (the sentences are paraphrased to protect the identity of patients):

– He had difficulties in responding to questions.
– He displayed difficulties in expressing himself and responded with only short incomplete sentences.
– He would respond with body gestures or single words when asked to elaborate on his responses.
– Often, he responded very slowly taking time to think before responding to questions.

The autism text documents are represented as attribute-value vectors ("bag of words" representation) where each distinct word corresponds to a *feature* whose value is the frequency of the word in the text sample. A text document is represented as a feature vector $\mathbf{x} = (x_1, ., x_j, .., x_L)$ where x_j is the j-th feature. Values were transformed with regard to the length of the sample. Function words were removed and stemming was performed on each extracted text. In summary, input vectors for machine learning consist of attributes (the words used in the sample) and values (the transformed frequency of the words). Outputs are autism versus normal, that is, binary decision tasks were learned. Clearly, the expressive power of the resulting explanations is limited by this "bag of words" representation.



……

He had difficulties in responding to questions.

He displayed difficulties in expressing himself and responded with only short incomplete sentences.

…..

Feature: +1 1:0.199 2:0.199 3:0.099 7:0.099 8:0.199 9:0.09 12:0.099 …

SVM models were generated using the linear kernel for the autism assessment data set

Expalantion Terms:

Explanation B: 147, 1.0, 1,

parent (148 40), understand (74 60), behaviour (45 46), response (34 37), often (34 28), help (34 42), follow (22 54), unable (20 32), task (19 38), pretend (19 16), engage (16 31), others (15 20), anxiety (14 22), social (11 26), communication (10 29), know (10 22), education (10 25), brother (9 24), management (9 14), concept (8 31), approach (8 18), appear (8 16), show (7 19), weakness (6 22), routine (6 19)

Fig. 3. Overview of the methodology and experiment of generating explanations to autism diagnosis result

For LOO (leave-one-out) cross validation, 236 SVM models were generated using the linear kernel for the autism assessment data sets. Thus, each model is used to classify one document. An SVM model is defined by support vectors $\mathbf{x}_i$ and associated parameters. The decision value of a text sample (represented as a feature vector $\mathbf{x}$) is then obtained as follows:

$$d(\mathbf{x}) = \sum_{i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

where $\mathbf{x}_i$ are support vectors and $\mathbf{x}$ is the feature vector, α_i are Lagrangian multipliers, and b is the offset. The antecedent of the rule of inference is then this:

$$\sum_{i \in SV} \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b \geq 0$$

That is, if $d(\mathbf{x}) \geq 0$, the feature vector $\mathbf{x}$ is positive or else negative. We use this insight into the SVM models to define three types of explanations:

1. Explanation A comprising all the features contributing to the decision value $d(\mathbf{x})$;
2. Explanation B comprising top-N contributing features that are sufficient to classify the features;
3. Explanation C comprising top-N contributing features that also have their sensitivity values $\partial d / \partial x_j$ greater than a set threshold value τ.

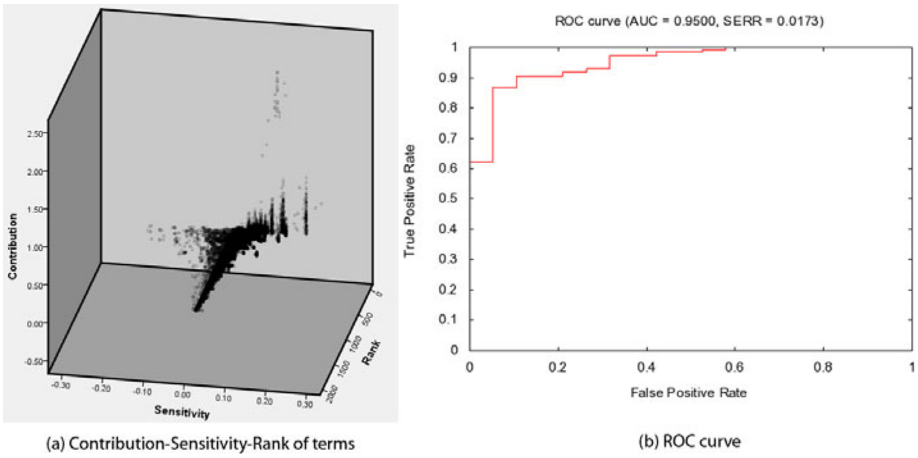


Fig. 4. (a) Relationship between contribution (deviation), sensitivity, and word ranks. Each point is a feature component that contributes to the decision of a feature. If a feature is a positive (negative) case, only the feature components having positive (negative) contributions are plotted. Rank 1 represents the most frequent term. (b) True-positive rate vs. false-positive rate of a linear support vector machine.

Technical details on generating each explanation type are described in Section 4. This approach is clearly a decompositional approach: analysis on the model parameters to select informative components and selecting subsets of more relevant components. Figure 4a summarizes the significance of each type of explanations. It plots contribution, sensitivity, and word rank of all features of the autism data set. It shows that sample features with higher ranking orders (more frequent words) and higher sensitivity values tend to have larger contribution values. This suggests that features having higher sensitivity values and higher ranking order provide greater information in decision making than other features. It also shows that most of the large contributions are made by more frequent words (high rank words).

3.2 Results

Support vector machines trained on the autism assessment data set achieved an accuracy of 90% and AUC (Area Under the Curve) of 0.95. The corresponding ROC curve is shown in Figure 4 (b). The sensitivity values are adjusted manually to obtain a reasonable amount of terms for Explanation type C. Sample explanations of a positive autism diagnosis case are provided below (Explanation A samples too big to show here):

1. Explanation B: social (94 46), **mother** (53 18), **brother** (50 23), old (44 58), interest (39 27), **game** (28 24), **describe** (27 24), share (21 41), computer (18 31), family (18 33), resource (17 42), limit (16 36), information (16 62), **create** (13 28), Strategies (11 36),.., (omitted the rest).

2. Explanation C: social (94 46), old (44 58), interest (39 27), share (21 41), computer (18 31), family (18 33), resource (17 42), limit (16 36), inform (16 62), create (13 28), strategies (11 36),.., (omitted the rest).

The numbers ($d(\mathbf{x})_i$, $\partial d/\partial x_i$) indicate relative contribution values $d(\mathbf{x})_i$ to the decision value $d(\mathbf{x})$ and sensitivity $\partial d/\partial x_i$ of the i-th term, respectively. Sensitivity-filtering (Explanation C) eliminates some of less sensitive terms (bold-faced terms) from Explanation B.

Sample explanations of a negative autism diagnosis case are provided below:

1. Explanation B: average (-190 -41), appropriate (-126 -71), **his** (-84 -18), attention (-62 -84), during (-41 -29), children (-32 -40), indicated (-20 -51), good (-19 -57), age (-19 -29), **reason** (-18 -20), attempt (-16 -26), **regular** (-14 -12), in (-12 -10), apparently (-11 -26), **mental** (-10 -22), well (-9 -35),.., (omitted the rest).
2. Explanation C: average (-190 -41), appropriate (-126 -71), attention (-62 -84), during (-41 -29), children (-32 -40), indicated (-20 -51), good (-19 -57), age (-19 -29), attempt (-16 -26), apparently (-11 -26), well (-9 -35),.., (omitted the rest).

Negative cases have negative contribution and sensitivity values.

4 Generating Explanations from SVM Models

In order to calculate the contribution values of each feature of a feature vector x, we use the centroid $\mathbf{C}$ of the population, which is estimated using the centroid $\mathbf{C}_{sv}$ of the support vectors:

$$\mathbf{C}_{sv} = \frac{1}{Nsv} \sum_{i \in SV} \phi(\mathbf{x}_i)$$

where N_{sv} is the number of support vectors. We can then calculate the deviation of a feature vector $\mathbf{x}$ from the estimate population centroid:

$$\mathbf{D}(\mathbf{x}) = \phi(\mathbf{x}) - \mathbf{C}_{sv}$$

Suppose $\mathbf{C}_{sv}$ is on the hyperplane: $\mathbf{w} \cdot \phi(\mathbf{C}_{sv}) \approx -b$. Then, we can obtain the decision value $d(\mathbf{x})$ using the deviation $\mathbf{D}(\mathbf{x})$:

$$d(\mathbf{x}) \approx (\mathbf{w} \cdot (\phi(\mathbf{x}) - \mathbf{C}_{sv})) = \sum_{i \in SV} \alpha_i y_i [K(\mathbf{x}_i, \mathbf{x}) - K(\mathbf{x}_i, \mathbf{C}_{sv})]$$

If K is the linear kernel, we can estimate the contribution of each j-th feature x_j as follows:

$$C_{sv,j} = \frac{1}{Nsv} \sum_{i \in SV} x_{i,j}$$

$$d(\mathbf{x})_j = \sum_{i \in SV} \alpha_i y_i x_{i,j} (x_j - C_{sv,j})$$

Now, for a feature vector $\mathbf{x}$, we can explain why a sample is positive (negative) by listing the feature elements that contribute to the decision value. That is, we can rank the features of a feature vector according to the amount of contributions made by the features. This is used as the basis of the explanation type A. We can also calculate the sum of all negative (positive) contributions and choose the top N positive (negative) contributions that are sufficient to push the decision value to positive (negative). This is used as the basis of the explanation type B.

It can be shown that Explanation A, B, and C are consistent: the same features are not used to explain an opposite class. Consistency is one of the criteria for evaluating rule quality (Andrew et al. [13]). Other important criteria for evaluating rule quality are accuracy and fidelity. It can be shown that the accuracy of an SVM model is bounded by the accuracy of explanation terms. Furthermore, we can achieve a similar performance using only the explanation-terms as the vocabulary: explanation-terms can mimic the behavior of the SVM model from which the explanation terms are extracted. That is, the explanations display a high level of fidelity.

This method can easily be extended to non-linear SVM models with convex decision boundaries. Applying the K-NN algorithm, N number of support vectors can be selected as an explanation reference point forming a centroid and a hyperplane in the input space. This new hyperplane is now a linear SVM model that can be used to generate explanations with regard to the selected support vectors.

4.1 Filtering Explanations with Sensitivity

Training a support vector machine for a data set of interest generates a hyperplane, which can be used to obtain the distance of a feature vector to the hyperplane to classify. The distance is normal to the hyperplane and thus the importance of a feature can be measured as the rate of change of the distance with respect to the feature. This can be easily obtained for a linear classifier as follows:

$$\frac{\partial d(\mathbf{x})}{\partial x_j} = \sum_{i \in SV} \alpha_i y_i x_{i,j}$$

where $d(\mathbf{x})$ is the distance of feature $\mathbf{x}$ to the hyperplane, x_j is the j-th component of the feature $\mathbf{x}$, and $x_{i,j}$ is the j-th component of a support vector $\mathbf{x}_i$. As we can see from the above equation, the importance of the j-th component for the hyperplane is the sum of j-th component of the support vectors multiplied by the class label and the Lagrange multipliers.

5 Social Linking of Parents by Explanation Similarities

The explanations generated provide relevancy of each feature to the particular classes. We can use this information to measure the relevancy of each part of the explanations to measure similarities between assessments. We use a semantic similarity measure between two terms based on a common sense database called ConceptNet[2]. This measure then can be used to link parents having children with similar assessment results.

[2] Used ConceptNet v2.1 from the Common Sense Computing Initiative at the MIT Media Lab (`http://csc.media.mit.edu`).

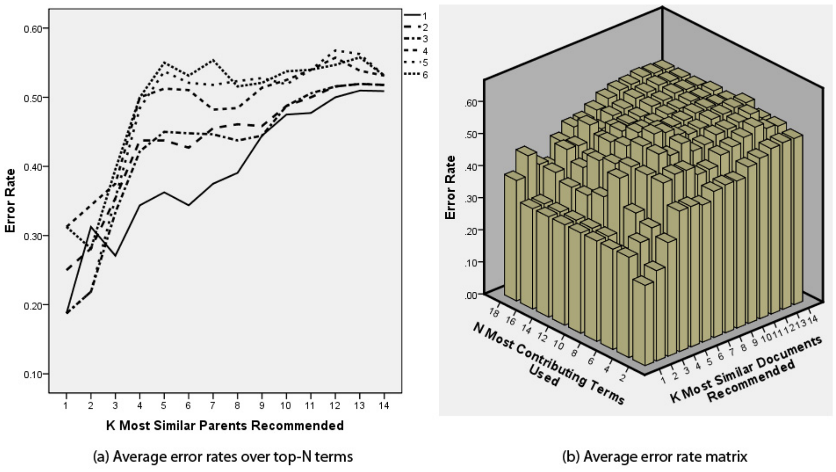


Fig. 5. Error rates of linking 18 autism assessment reports to top-K most similar assessment reports using top-N most contributing explanation terms

We start with a simple approach to generating similarity measures. The method is scoring each explanation term in one assessment with each explanation term in another assessment. In this approach, the similarity between two assessments is given by determining contributions made by explanations terms and semantic relationships between terms. The similarity between two explanations A and B are defined as follows:

$$s_{i,j} = \frac{1}{|A||B|} \sum_{i \in A} \sum_{j \in B - \{i\}} u(i,j)|d(\mathbf{A})_i||d(\mathbf{B})_j|$$

where $u(i,j)$ is the semantic similarity function that measures how close the term i in explanation A is to the term j in an explanation B where $i \neq j$, $d(\mathbf{A})_i$ and $d(\mathbf{A})_j$ are the amount of contributions of the features i and j, respectively. ConceptNet analogy space is used as the similarity function. Each semantic similarity between terms is the L1 similarity measure for social networks defined in [15]. That is, dot products of two vectors $\overrightarrow{i}$ and $\overrightarrow{j}$ in the analogy space, but weighted with contributions of terms. The parent link information is then generated by ranking assessments that are closed to an assessment and selecting N most similar assessments.

To test the effectiveness of this method, similarities between 16 assessments were measured: 8 assessments with autism diagnosis and 8 assessments with negative autism diagnosis. The average of the similarity measure between assessments with same diagnosis results was 8.66 and the average of the similarity measures between assessments with different diagnosis results was 6.83. The method of measuring similarities did not have information on class labels, but was able to distinguish positive cases from negative cases only using semantic similarity between the explanation terms. Figure 5 shows the error rates of linking parents to other parents with the same diagnosis results, where the error rate is defined as follows:

$$\text{Error Rate} = \frac{\text{Number of Incorrect Recommendations}}{\text{Number of Recommendations}}$$

The parents were linked by selecting top-K most similar assessments using top-N most contributing explanation terms. It shows that when one parent was linked to 32% or less proportions of the total community, the error rate is less than 35%, and it consistently gets better as K decreases further. The average error rate matrix in Figure 5 (b) clearly indicates that a small number of key explanation terms can provide good link information and that the relevance information of explanation terms is useful.

Using this similarity measure, we can recommend a parent p to a community in a set C of communities by selecting the community with the maximum average-similarity between the parent p and all parents p' in a community c:

$$R(p,C) = \underset{c \in C}{argmax} \frac{1}{|c|} \sum_{p' \in c} s_{p,p'}$$

6 Discussions and Future Work

This is the first report of a novel approach in providing social network-based health care services to families with ASD children. This is also the first report on the accuracy of social-linking using an explanation-based similarity measure. We showed that a semantic similarity measure between explanations of assessments has great potential for discovering close social communities of parents who can support each other for lifelong conditions like autism. It can also be used to find all potential hidden communities of patients and parents on the Internet. There is massive potential of incorporating these sophisticated information extraction technologies in social networking more generally.

Long term disabilities likes ASD pose a significant burden for families, and thus it is essential that health care services actively participate in communities to support patients' families. The community suggestion method facilitates social linking between parents with similar assessment reports to make information obtained through social networking more relevant and useful.

The approach of extracting some piece of knowledge using machine learning in explaining psychiatric assessments has the potential to provide early diagnosis and more standardized assessments, and to improve the usability of machine learning techniques in the medical and security domains. This approach of extracting explanations using some form of analysis on machine learning and associated parameters can be further expanded by using alternative feature representations of text data sets, such as concept terms or semantic terms.

References

1. Charman, T., Baird, G.: Practitioner review: Diagnosis of autism spectrum disorders in 2- and 3-year-old children. Journal of Child Psychology and Psychiatry 43(3), 289–305 (2002)
2. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
3. Deshpande, M., Karypis, G.: Item-based top-N recommendation algorithms. ACM Trans. Inf. Syst. 22(1), 143–177 (2004)
4. Diederich, J.: Rule extraction from support vector machines: An introduction. In: Rule Extraction from Support Vector Machines. SCI, vol. 80, pp. 3–31. Springer, Heidelberg (2008)

5. Diederich, J., Al-Ajmi, A., Yellowlees, P.: Ex-ray: Data mining and mental health. Applied Soft Computing 7(3), 923–928 (2007)
6. DSMIV: Diagnostic and Statistical Manual of Mental Disorders, Text Revision, 4th edn. American Psychiatric Association (2000)
7. Filipek, P.A., Accardo, P.J., Baranek, G.T., Cook, J.E.H., Dawson, G., Gordon, B., et al.: The screening and diagnosis of autistic spectrum disorders. Journal of Autism and Developmental Disorders 29(6), 439–484 (1999)
8. Filipek, P.A., Accardo, P.J., Ashwal, S., Baranek, G.T., Cook Jr., E.H., Dawson, G., et al.: Practice parameter: Screening and diagnosis of autism. report of the quality standards subcommittee of the american academy of neurology and the child neurology society. Neurology 55(3), 468–479 (2000)
9. ICD10: International Statistical Classification of Disease and Related Health. World Health Organization, Geneva (1992)
10. Joachims, T.: Making large-scale support vector machine learning practical, pp. 169–184 (1999)
11. Johnson, C.P., Myers, S.M.: The Council on Children with Disabilities: Identification and evaluation of children with autism spectrum disorders. Pediatrics 120, 1183–1215 (2007)
12. Klin, A., Lang, J., Cicchetti, D.V., Volkmar, F.: Brief report: Interrater reliability of clinical diagnosis and dsm-iv criteria for autistic disorder: Results of the dsm-iv autism field trial. Journal of Autism and Developmental Disorders 30(2), 163–167 (2000)
13. Robert Andrews, J.D., Tickle, A.B.: Survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge-Based Systems 8(6), 373–389 (1995)
14. Sarasohn-Kahn, J.: The wisdom of patients: Health care meets online social media. California HealthCare Foundation iHeath Reports (April 2008), http://www.chcf.org/publications/2008/04
15. Spertus, E., Sahami, M., Buyukkokten, O.: Evaluating similarity measures: a large-scale study in the orkut social network. In: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, KDD 2005, pp. 678–684. ACM, New York (2005)
16. Swan, M.: Emerging patient-driven health care models: an examination of health social networks, consumer personalized medicine and quantified self-tracking. International Journal of Environmental Research and Public Health 6(2), 492–525 (2009)
17. Tickle, A., Andrews, R., Golea, M., Diederich, J.: The truth will come to light: Directions and challenges in extracting the knowledge embedded within trained artificial neural networks. IEEE Transactions on Neural Networks 9(6), 1057–1068 (1998)
18. Wing, L., Gould, J.: Severe impairments of social interaction and associated abnormalities in children: Epidemiology and classification. Journal of Autism and Developmental Disorders. 9(1), 11–29 (1979)

Learning Belief Connections in a Model for Situation Awareness

Maria L. Gini[1], Mark Hoogendoorn[2], and Rianne van Lambalgen[2]

[1] University of Minnesota, Department of Computer Science and Engineering
200 Union Street SE, Minneapolis, MN 55455, United States of America
gini@cs.umn.edu
[2] VU University Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
{mhoogen,r.m.van.lambalgen}@few.vu.nl

Abstract. Situational awareness is critical in many human tasks, especially in cases where humans have to make decisions fast and where the result of their decisions might affect their life. This paper addresses the problem of learning optimal values for the parameters of a situational awareness model. The model is a complex network with nodes connected by links with weights, which connect observations to simple beliefs, such as "there is a contact", to complex belief, such as "the contact is hostile", and to future beliefs, such as "it is possible the pilot is being targeted". The model has been built and validated by human experts in the domain of F16 fighter pilots and is used to study human decision making. Given the complexity of the model, there is a need to learn appropriate weights for the connections, which, in turn, affect the activation levels of the beliefs. We propose the use of a genetic algorithm and of a sensitivity based approach to learn the weights in the model. Extensive experimental results are included.

1 Introduction

In order to create agents that exhibit human like intelligent behavior, one of the crucial elements to be addressed is the formation of an understanding of the current situation. In case an agent lacks such an understanding, it is very difficult to come to intelligent decisions. In psychological literature this process of becoming aware of the current situation is commonly referred to as Situation Awareness (see e.g. [1], [2]).

A variety of models for situation awareness have been proposed (see e.g. [3]). The main principle behind the majority of these models is that certain knowledge is present that expresses relationships between the various concepts in the world. The agent can utilize this knowledge by combining observation knowledge (which is most likely partial) with the knowledge about the aforementioned relationships. A complete picture of the situation can thus result. Such knowledge about relationships between concepts in the world is however mostly domain dependent, and for each new domain, a domain expert needs to specify the relationships that hold within the domain. Ideally, one would want an agent to learn these relationships based upon examples it has seen in the world. Since this is a challenging task, we leverage the availability of networks of the relevant concepts and their connections built for many domains by domain experts. We use such a network to simplify the learning task.

In this paper, an existing model for situation awareness (cf. [4]) is taken as a basis. The model basically consists of beliefs which have a certain activation value, and a network which expresses relationships between the beliefs in the form of connections with a certain strength (in this paper we will refer to the value of a connection strength as weight).These relationships are not only one-to-one, but can also be more complex whereby multiple beliefs are aggregated into more complex beliefs about the situation. In addition, the model also incorporates a time aspect, whereby influences of connections are calculated based upon their importance, and inference stops when the time limit has been reached. In order to learn the weights of the relationships, two algorithms are introduced, namely a genetic algorithm, and an algorithm which is based upon the importance of the weights. Both algorithms are compared based upon a case study from the domain of F16 fighter pilots.

This paper is organized as follows. First, related work is presented in Section 2. The model for situation awareness which is used in this paper is described in Section 3. Section 4 describes the learning algorithms that are utilized. The case study is in Section 5 and the results in Section 6. Finally, Section 7 concludes the paper.

2 Related Work

Many approaches exist to estimate parameters of a given model. Since many parameters of a system or model might not be known in advance and can only be determined by the observed behavior of a system, such techniques are essential in a variety of fields. Approaches that are based on analytical mathematical techniques can be used (see e.g. [5]), but especially when looking at highly complex models they can be difficult to apply. Other approaches include Genetic Algorithms (cf. [6]) but also algorithms such as Simulated Annealing (cf. [7]). In this paper, the main focus is on two approaches, namely Genetic Algorithms and a sensitivity-based approach.

Genetic Algorithms (GA) are a popular method to solve a variety of optimization problems (cf [6]; [8]) GAs have the advantage over other methods of being simple to use and capable of exploring a large part of the search space. GAs can converge to local minima, but appropriate choices for crossover points, methods to maintain the distribution of the population, and a higher probability of mutation typically enable converging to a good quality solution. GAs have been used, for instance, for mission planning ([9]) and situational awareness ([10]). Other machine learning methods have been used for situational awareness, such as particle filters that have been used for state estimation for Mars Rovers ([11]) and Adaptive Resonance Theory (ART) that has been used for information fusion ([12]).We use GAs for parameter optimization, specifically to find the weights of the links that connect simple beliefs, complex beliefs, and future beliefs. We have chosen to use GAs because there is a large number of links in the network and hence a large number of parameters we need to learn. Since we know from the experts the expected activation levels of the complex beliefs, we use those in the fitness function.

Sensitivity analysis can also be used to refine models. For instance, sensitivity analysis has been used to prune units in a feedforward neural network ([13]) or to update parameters in Bayesian networks ([14]). We use sensitivity analysis to estimate the weights on the links.

The model we use for situation awareness is discussed next. Other models, such as Fuzzy Cognitive Maps ([15]), could be refined in a similar way.

3 Model for Situation Awareness

The model for Situation Awareness which has been adopted (cf. [4]) consists of four main components. Three components are in line with the model of Endsley [1] which includes the perception of cues, the comprehension and integration of information and the projection of information for future events. The last component describes the mental model of the human which describes the connections between the various states in the situation awareness model. The model functions as follows. Initially, the agent starts to observe within the world, and obtains the results of these observations. Observations are obtained with a certain degree of certainty. Using these observations and the knowledge stored in the mental model, simple beliefs about the situation are derived. Simple beliefs concern simple statements about the current situation that have a one-to-one mapping to observations, or have a one to one mapping to another simple belief. An example of a mental model including such a mapping is shown in Figure 1 (whereby the lower part concerns the aforementioned mapping). This figure assumes a fighter pilot setting whereby the fighter pilot is trying to judge his current situation. In the figure, it can be seen that there are two observations: whether a plane is closing in (closing_in) and whether this plane comes from a hostile direction (from_hostile_direction). These are also present as simple beliefs within the mental model (with a direct connection between the observation and the belief). In addition, a simple belief is present that the plane is hostile (hostile_plane). Simple beliefs are represented by the following predicate:

simple_belief: INFO_ELEMENT x TIME x VALUE

In the predicate, the value presents the *activity* of the belief in the mind of the agent, which depends on a number of aspects, such as the certainty of the observation. In order to translate the certainty of an observation into an activation of a belief, the following rule is used (note that the elements that are part of the mental model are shown in gray). The arrow ($\twoheadrightarrow$) represents a temporal relationship, namely that the antecedent being true for 1 time point results in the consequent being true for 1 time point:

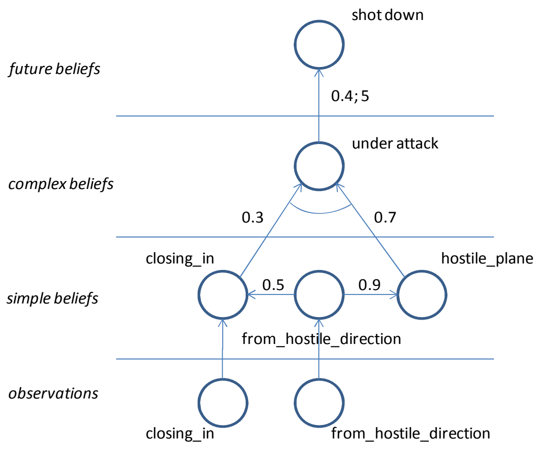


Fig. 1. Example mental model

LP1: Observations to simple beliefs
current_time(t) $\wedge$ observation_result(I, t, V1) $\wedge$ simple_belief(I, t-1, V2) $\wedge$ **simple_belief_decay(I, γ)** $\wedge$ **steepness(I, σ)** $\wedge$ **threshold_value(I, τ)** $\wedge$ **recency_influence(I, α)**
$\twoheadrightarrow$ simple_belief(I, t, $(1-\alpha)\cdot\gamma\cdot V2 + \alpha\cdot th(\sigma, \tau, V1)$)

This expresses that in the formation process of a simple belief, both the certainty of the observation and the old value of the belief are considered. Two parameters influence the value of a new observation compared to a previous belief value as well as how fast the belief decays (i.e., how fast the belief looses activation). Furthermore, a so-called threshold function is applied with parameters σ and τ, representing the steepness of the function and the threshold respectively. It enables the transformation of the certainty value of an observation to the activation value of a belief. Something might be observed with only a very low certainty but due to its importance, it might still result in a high activation. During times when no observations are made concerning a specific belief, the belief just decays.

After the new activation value of simple beliefs has been calculated, the influence of the simple beliefs among each other is determined. Hereby, influence weights reside in the interval [-1,1]. Figure 1 also shows the weights, whereby there is a strong connection (0.9) between the belief that the plane comes from a hostile direction, and the belief that it concerns a hostile plane. Furthermore, a somewhat weaker connection (0.5) exists between a plane coming from a hostile direction, and the plane closing in. In domains where there are clear relationships between beliefs, experts typically have stronger connections than novices. In order to calculate the new values for simple beliefs due to mutual influences, an iterative form of updating is used. This is based upon calculating all the influences that originate from the simple belief with the highest activation value:

Method 1: Updating simple beliefs

```
1.  Search for the simple belief with the highest value that has not been considered yet and
    whose value is above the threshold.
    •  For all connections originating from the selected belief:
       a.  Select the connection with the highest strength originating from the selected belief
           that has not been considered yet of which the absolute value is above the minimal
           connection threshold. In case none are left, go to (d). If none were present in the
           beginning, go to (e).
       b.  Perform calculations (LP2 shown below)
       c.  Mark the connection as considered and go to (a).
       d.  Add 1 to the time used.
       e.  Mark the selected belief as considered. In case the time has reached the
           maximum time the algorithm terminates, otherwise go to 1.
```

This algorithm has an anytime behavior, and stops when the available time has ended. The updating of the belief is expressed as follows:

LP2: From simple beliefs to simple beliefs
current_time(t) $\wedge$ simple_belief(I1, t, V1) $\wedge$ simple_belief(I2, t, V2) $\wedge$ **connection_strength(I1, I2, w1)** $\twoheadrightarrow$simple_belief(I2, t, $V2 + \gamma\cdot(Neg(V1\cdot w1\cdot V2) + Pos((1-V2)\cdot(V1\cdot w1)))$)

After simple beliefs are updated, complex beliefs are derived from them. Complex beliefs are aggregations of multiple beliefs (simple or complex) and describe the situation in a composed manner. Figure 1 shows an example, whereby the simple beliefs about a hostile plane and the fact that the plane is closing in results in the complex belief that you are under attack. In the model, it is assumed that the complex beliefs are calculated by taking a weighed sum of the relevant simple beliefs. An iterative form is used to update the complex beliefs, identical to method 1 sketched above. The updating of the value itself is expressed in LP3:

LP3: From simple to complex beliefs
complex_belief(CI1, t, VI1) ∧ belief(I1, t, V1) ∧ ….∧ belief(In, t, Vn) ∧
in_same_group(I1, ... In, CI1) ∧ **connection_strength(I1, CI1, w1)** ∧ ….. ∧
connection_strength(In, CI1, wn) ∧ **steepness(CI1, σ)** ∧ **threshold_value(CI1, τ)** ↠
complex_belief(CI1, t, VI1 + $\gamma_c \cdot$(f(w1V1, ... , wnVn) – VI1))

Here, the contributions of the beliefs that together form a connection to the complex belief are calculated using a combination function f (e.g. a logistic threshold function or weighted sum). In case no new information is present with respect to a complex belief, a simple decay of the activation value is assumed.

In order to project the complex beliefs to the future situation, they are forwarded to the component belief formation on future situation. To calculate activation values of future beliefs, time and delay parameters are an aspect of the connection strength used to derive the specific belief (see again Figure 1 for an example, shot down in this case refers to the pilot that is being modeled being shot down):

LP4: From complex to future beliefs
complex_belief(I1, t, V1) ∧ ..... ∧ complex_belief(In, t, Vn) ∧ future_belief(FI1, t+D, VI) ∧
in_same_group(I1, ... In, FI1) ∧ **delay_parameter(I1, ... In, FI1, D)** ∧
connection_strength(I1, FI1, D, w1) ∧ ..... ∧ **connection_strength(In, FI1, D, wn)** ∧
steepness(FI1, σ) ∧ **threshold_value(FI1, τ)**
↠ future_belief (FI1, t+D, VI+$\gamma_f \cdot$(f(w1V1, .. , wnVn) – VI))

Note that the future beliefs can be the same as the complex beliefs. An agent might for instance know that the belief refers to a state that will happen in 5 time points.

The judgment of the future situation that then follows is used to direct the observations of the agent, together with the goals of the agent at a specific point in time. Also, goals and complex beliefs are used by the agent to make decisions on the actions to take. However, these aspects are outside the scope of this paper.

4 Learning SA Model Parameters

The focus in this paper is to learn the weights of the mental model (i.e. not the other parameters part of the model) In order to learn these connections, two different approaches have been utilized, namely a genetic algorithm and a dedicated approach based upon measuring the importance of the weights. Before they are explained, the definition of the fitness function, which both approaches use, is addressed.

4.1 Fitness Function

In order for the learning to take place, a fitness function needs to be defined that expresses how well a solution complies with the desired state. In this case, the fitness can be measured in terms of how much the activation levels differ from the ideal activation levels (i.e. the activation levels an expert considers appropriate) since we want the model to exhibit a good performance on each state. The following algorithm can then be used to determine the fitness:

Algorithm 1. Calculate full fitness

```
fitness = 0;
for all time points t
  for all simple belief elements SB
    if the ideal simple belief value for SB is V1 at t
      and the current simple belief value for SB is V2 at t
        fitness = fitness + |V1-V2|
    end
  end
  for all complex belief elements CB
    if the ideal complex belief value for CB is V1 at t
      and the current complex belief value for CB is V2 at t
        fitness = fitness + |V1-V2|
    end
  end
  for all future belief elements FB
    if the ideal complex belief value for FB is V1 at t
      and the current complex belief value for FB is V2 at t
        fitness = fitness + |V1-V2|
    end
  end
end
fitness = fitness / (t * (|SB| + |CB| + |FB|))
```

In addition, a partial fitness function is also used, whereby more emphasis is placed on the formation of complex and future beliefs, as these are a true measure of the full understanding of the situation (since the complex and future beliefs are an aggregate of the simple beliefs). In this case, the approach as shown in Algorithm 1 can simply be reused except for the elements which concern the simple beliefs.

4.2 Genetic Algorithm

The genetic algorithm applied is a relatively standard GA from the Genetic Algorithm Toolbox[1]. Below, the most important aspects of the GA are explained.

- *Individual representation.* The population is composed of individuals that are represented by a binary string which represents real values (i.e. the weights in this case) with a certain precision.
- *Population initialization.* The population is in principle initialized randomly, but in some runs an individual with all zero weights has been explicitly added due to the fact that many connections between beliefs tend to be zero.
- *Selection.* The selection of individuals is performed by first ranking the individuals using linear ranking and then selecting individuals based upon stochastic universal sampling.
- *Mutation.* The mutation operator used is straightforward: each bit is simply mutated with a certain probability.
- *Crossover.* A single point crossover function is used to combine the individuals.

[1] Downloadable from `http://www.shef.ac.uk/acse/research/ecrg/gat.html`

4.3 Sensitivity Based Approach

The Sensitivity Based approach that is taken in this paper consists of two parts. In the first part sensitivities of all weights are calculated according to Algorithm 2. In the second part, weights are sorted from high to low sensitivity and the X most sensitive weights are adjusted (simulations can be performed with different values of X). The method to select and adjust the weights is shown in Algorithm 3. Each simulation cycle runs both parts, so that after adjustment of the weights the sensitivity of each weight is calculated again.

Algorithm 2. Determine Sensitivities

```
initialize weights
t = 0
sens_increment = 0.05

while t < end_time

run the SA model with current weights and
determine fitness value f_current

  for all weights W
     if the weight value for W is V1 at t
        V_old = V1
        V1 = V_old + sens_increment
        run the SA model with the current weights and the new V1
             and determine the fitness value f_add
        f_diff_add = f_add – f_current
        V1 = V_old – sens_increment
        run the SA model with the current weights and the new V1
             and determine the fitness value f_sub
        f_diff_sub = f_sub – f_current

        if f_diff_add > f_diff_sub,
          the value of upwards for W: U(W, t) = 1
          f_diff = f_add
        else
          the value of upwards for W: U(W, t) = -1
          f_diff=f_sub
        end
        the sensitivity S for W at t = f_diff / sens_increment
        V1=V_old
   end
 end
end
```

Algorithm 3. Adjust Parameters

```
for all weights W
   if W belongs to the X most sensitive weights
     if W has value V, sensitivity S and upwards value U
        if W should be adjusted upwards (U(W, t) == 1)
           V=V+tune_increment
        else if W should be adjusted downwards (U(W, t) == -1)
           V=V-tune_increment
        end
     end
   end
end
```

Simulations were performed using two different versions of Algorithm 3. In the first (Sensitivity_Based_fixed), a fixed number (fixed_increment) was added or subtracted to the dedicated weight. In the second (Sensitivity_Based_adjusted), this number was adjusted using the sensitivity according to the following formula:

tune_increment=(fixed_increment/(S*X))*speed

Here, S is the sensitivity and speed is the speed with which the weights are adjusted. X is the number of tuned weights.

5 Case Study: Fighter Pilots

The case study used to evaluate the approach is a short version of the case study presented in [4]. The idea of applying the model in this context is to develop human-like agents against which human fighter pilots can practice in a simulator.

In this simplified case study it is assumed that a pilot performs observations through a radar warning receiver and forms beliefs regarding the behavior of the opponent. The opponent of the pilot uses the radar to search for the agent, to track the flying pattern of the agent, or (when the opponent has found the agent) to lock it and eventually shoot a missile. The radar warning receiver can generate an occasional tone, a frequent tone or a continuous tone, which are translated into simple beliefs. These beliefs can be derived into other simple beliefs, indicating that the pilot is respectively searched, tracked or locked by the opponent. Each simple belief is connected to another simple belief and the values of these connections need to be learned. Connection values from the simple beliefs to complex beliefs (*not detected*, *detected*, *tracked* and *locked*) are learned in this paper as well as connection values from complex beliefs to future beliefs (*detected*, *ownship tracked*, *ownship locked* and *opponent missile release*).

6 Experimental Analysis

In this section, the results of the various approaches that have been introduced in Section 3 are shown. First, the overall setup of the evaluation is discussed, followed by results with the full fitness function (based on an evaluation of all activation values) and the results with a partial fitness function.

6.1 Evaluation Approach

In order to investigate how well the two approaches are able to find parameters of the model that describe the desired behavior well, they have been tested on a specific case study (as described in Section 4). The relevant states, connections between the states, and the strengths of those connections have been determined by domain experts. Furthermore, a scenario in which observations get a certain value at specific time points has been defined by domain experts as well. After running the simulation with these settings, the domain experts evaluated the resulting activation values, and they indicated the levels were indeed as they would expect. In the experiments, these activation levels have therefore been used as a golden standard. Hereby, it concerns a total of 23 states, 229 possible connections, and 50 time steps during which new activations come in. As a result, a total of 1150 activation values occur over the entire timeline. For the full fitness function, the average deviation of all these states is used as a measure to determine the fitness, whereas for the partial fitness function merely the activation level of the complex and future beliefs are used. In order to compare the activation values, a simulation of the model is performed with the current set of parameters and the resulting activation values are compared.

6.2 Results

Table 1. Algorithm settings

Genetic Algorithm		*Sensitivity-based Algorithm*	
Parameter	**Setting**	**Parameter**	**Setting**
population size	200	sens_increment	0.05
number of generations	1000	X	10
precision of variables (number of bits)	6	fixed_increment	0.1
generation gap	0.5	speed	0.1
crossover rate	0.7		
mutation rate	0.4		

The first series of experiments have been run based upon the *full fitness function*, in which all activations are used to determine the fitness of a certain set of parameters. For the Genetic Algorithm, two runs have been performed: one whereby the initial population explicitly includes an individual with all weights set to zero (and the rest initialized with random values), and a second whereby all individuals are initialized with random values. Both approaches have been included due to the fact that many of the weights to be learned are typically zero. The settings used for the parameters of the Genetic Algorithm in all experiments are shown in the first part of Table 1 and have been set to this value in accordance with the experiences obtained while running

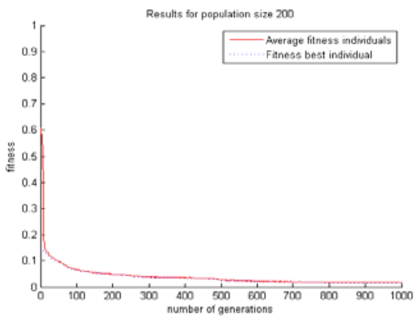


Fig. 2. GA with initial random population and full fitness function

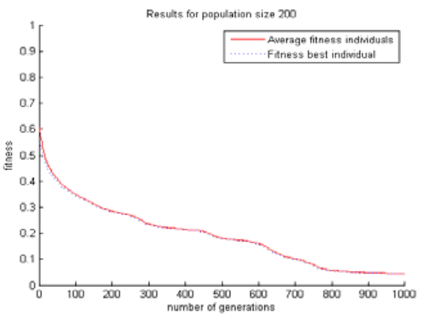


Fig. 3. GA with initial random population and zero and full fitness function

the algorithm. Considering the Sensitivity Based algorithm, simulations were done first to derive parameter settings for the optimal results. The parameter values are in the second part of Table 1. Both the Sensitivity_Based_fixed and the Sensitivity_Based_adjusted approach were used with initial weights set either to 0.5 or to 0.1.

Results for the GA are presented in Figure 2 and 3. Results for the Sensitivity_Based are presented in Figure 4 and 5. Overall results are given in Table 2. Figure 2 shows the fitness over the number of generations given a completely random initial population. The fitness gradually decreases as the number of generations goes up, and stabilizes at a fitness value around 0.044, which means that only a very small deviation of the activation values remains. Hence, the learning process has been very successful. Figure 3 shows the results with the full fitness function, and when an individual with all weights set to zero is included in the initial population. Hereby, the results are immediately a lot better, and also converge to an even higher precision: 0.017. The weights also deviate a bit less from the golden standard, but the standard deviation is a bit higher. Another interesting aspect is how close the learned weights are compared with the expert weights that have formed the basis of the golden standard. These results are shown in Table 2 (along with the detailed results of all experiments).

Table 2. Detailed results of parameter estimations

Setting	Full fitness	Partial fitness	Average weight dev.	Standard deviation weights
Training on full fitness				
GA	0.043956	0.030761	0.230297	0.316123
GA with zero	0.016647	0.004239	0.216074	0.343887
SBfixed	0.3723	0.5860	0.4592	0.3093
SBfixed 0.1	0.1582	0.0330	0.2948	0.3632
SBadjusted	0.3893	0.5772	0.4662	0.2802
SBadjusted 0.1	0.0624	0.0162	0.2459	0.3607
Training on partial fitness				
GA	0.220711	0.006288	0.406425	0.398423
GA with zero	0.149290	0.000428	0.303008	0.387562
SBfixed	0.3486	0.2460	0.3633	0.4198
SBfixed 0.1	0.1669	0.0115	0.1926	0.3055
SBadjusted	0.3253	0.2334	0.4070	0.4592
SBadjusted 0.1	0.1909	0.0178	0.2325	0.3328

The weights are shown to deviate quite a bit (certainly considering the fact that the majority of the weights cannot deviate more than 1 due to the chosen range), and the standard deviation is also quite high. This shows that despite the weights not being exactly as in the golden standard, the behavior of the model is still satisfactory. This is expected when using such a complex model.

Figure 4 and 5 show how the fitness decreases when using either the Sensitivity_Based_fixed or the Sensitivity_Based_adjusted method (final results are shown in Table 2). It can be seen from Figure 4 that the fitness is similar in the adjusted method as compared to the fixed method (around 0.375). When starting at 0.1 (Figure 6), the final fitness is much lower. The adjusted method performs better (fitness around 0.06) as compared to the fixed method (fitness around 0.15).

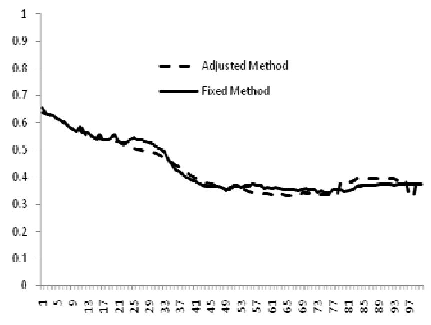


Fig. 4. Sensitivity based analysis using the full fitness function and initial weights set to 0.5

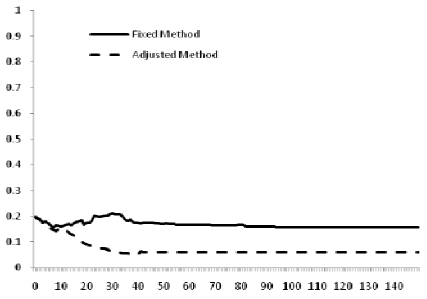


Fig. 5. Sensitivity based analysis using the full fitness function and initial weights set to 0.1

Besides the full fitness function, the algorithm has also been run with the *partial fitness function*. The results hereof are shown in Table 2. When tuning is performed on the partial fitness function, the performance of the different varieties of the algorithms is very good. The results generalize quite well for the full fitness function as well (going up to an accuracy of about 0.15). When tuning is performed on the full fitness function the performance on the partial fitness function is shown to vary a lot.

Note that the time and space complexity of the algorithms are not described for the sake of brevity. The complexity is mainly located in calculating the fitness of the parameter setting as this requires running the model itself.

7 Conclusions and Future Work

In this paper, an approach has been presented that enables the learning of parameters of a model for Situation Awareness. Models for Situation Awareness are crucial to enable agents to have a good understanding of the current state of the world and undertake appropriate actions. The models frequently have a large set of parameters and setting them manually is time consuming. Therefore learning these parameters is essential. Two learning approaches have been adopted, namely genetic algorithms and a sensitivity-based approach. Both showed very good results, with a very high accuracy of reproducing the judgment of the situation given certain observations in the world (although there is quite some deviation between the weights used to generate the test cases and the learned weights). The genetic algorithm performs significantly better than the sensitivity-based approach.

In the current research the connections strengths were learned through a model with ideal connections that was available beforehand and ideal results that were judged by experts as realistic. However, for future work we would like to investigate online learning of the mental model weights based upon the success of actions that are performed using the current judgment of the situation. This would mean that the entire input for learning the connections between states in the world would be feasible just by looking at the appropriateness of the actions. Furthermore, we would like to investigate the suitability of other learning approaches as well, such as Bayesian learning.

Acknowledgments. This research has been conducted and partially funded in the SmartBandits project in cooperation with the National Aerospace Laboratory in the Netherlands and the Royal Netherlands Air Force. We would like to thank the other participants of this project for their valuable input and Jan Treur for the fruitful discussions. Support for Maria Gini was provided by a KNAW Visiting Professorship.

References

1. Endsley, M.R.: Toward a theory of Situation Awareness in dynamic systems. Human Factors 37(1), 32–64 (1995)
2. Endsley, M.R.: Theoretical underpinnings of situation awareness: a critical review. In: Endsley, M.R., Garland, D.J. (eds.) Situation Awareness Analysis and Measurement. Lawrence Erlbaum Associates, Mahway (2000)
3. So, R., Sonenberg, L.: Situation Awareness in intelligent agents: foundations for a theory of proactive agent behavior. In: Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2004), pp. 86–92 (2004)
4. Hoogendoorn, M., van Lambalgen, R., Treur, J.: Modeling Situation Awareness in human-like agents using mental models. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (2011)
5. Koch, K.R.: Parameter Estimation and Hypothesis Testing in Linear Models. Springer, Heidelberg (1999)
6. Kirkpatrick, S., Gelatt, C.D., Vechhi, M.P.: Optimization by simulated annealing. Science 220, 4598, 671–680 (1983)
7. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley (1989)
8. Jin, Y., Branke, J.: Evolutionary optimization in uncertain environments-a survey. IEEE Trans. on Evolutionary Computation, 9(3), 303–317 (2005)
9. Rosenberg, B., Richards, M., Langton, J.T., Tenenbaum, S., Stouch, D.W.: Applications of multi-objective evolutionary algorithms to air operations mission planning. In: Proceedings GECCO (2008)
10. Salerno, J., Hinman, M., Boulware, D., Bello, P.: Information fusion for Situational Awareness. In: Proc. Int'l Conf. on International Fusion Cairns, Australia, (2003)
11. Dearden, R., Willeke, T., Simmons, R., Verma, V., Hutter, F., Thrun, S.: Real-time fault detection and situational awareness for rovers: report on the Mars technology program task. In: Proc. Aerospace Conference, vol. 2, pp. 826–840 (March 2004)
12. Brannon, N., Conrad, G., Draelos, T., Seiffertt, J., Wunsch, D., Zhang, P.: Coordinated machine learning and decision dupport for Situation Awareness. Neural Networks 22(3), 316–325 (2009)
13. Engelbrecht, A.P., Fletcher, L., Cloete, I.: Variance analysis of sensitivity information for pruning multilayer feedforward neural networks. In: Int'l Joint Conf. on Neural Networks, vol. 3, pp. 1829–1833 (1999)
14. Wang, H., Rish, I., Ma, S.: Using sensitivity analysis for selective sarameter update in Bayesian network learning. AAAI Spring Symposia, Tech. report SS-02-03 (2002)
15. Kosko, B.: Fuzzy Cognitive Maps. International Journal of Man-Machine Studies 4, 65–75 (1986)

An Integrated Agent Model Addressing Situation Awareness and Functional State in Decision Making

Mark Hoogendoorn, Rianne van Lambalgen, and Jan Treur

VU University Amsterdam, Agent Systems Research Group
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
{m.hoogendoorn,r.m.van.lambalgen,j.treur}@vu.nl

Abstract. In this paper, an integrated agent model is introduced addressing mutually interacting Situation Awareness and Functional State dynamics in decision making. This shows how a human's functional state, more specific a human's exhaustion and power, can influence a human's situation awareness, and in turn the decision making. The model is illustrated by a number of simulation scenarios.

Keywords: Situation awareness, functional state, agent model.

1 Introduction

An agent's decision making in realistic situations strongly depends on the situational awareness of the agent; e.g., [2]. When the agent is not aware of certain aspects of the situation that are relevant for the actions to undertake, this may result in actions that are ineffective or even counter-productive. In [5] a computational agent model was introduced to address situational awareness. Having a sufficient extent of situation awareness is a good basis for effective decision making. However, in demanding circumstances this easily may be compromised due to longer periods with high workload and high levels of stress, and due to this, accumulating exhaustion leading to a less optimal functional state; e.g., [1], [3], [4], [10], [12]. Therefore the extent of situation awareness is not constant, but may fluctuate over time. This was not taken into account in the model for situation awareness presented in [5].

The current paper addresses how situation awareness may be affected by increased exhaustion, and how this may lead to less optimal decision making. To this end the situation awareness model from [5] is integrated with a model for functional state in relation to exhaustion introduced in [7], and a decision model presented in [6]. The resulting model shows how depending on fluctuations in load, extra effort may be exerted, but if periods of high load have longer durations, due to the accumulated exhaustion the agent's situation awareness becomes less, and the decision making less optimal. Moreover, the model shows how in subsequent periods of lower load recovery from exhaustion takes place, and this results in higher extents of situation awareness and more optimal decision making.

The paper is structured as follows. In Section 2 a brief introduction of the background literature is presented. Section 3 summarizes the three existing models used in the integrated agent model. In Section 4 it is described how the models were integrated to obtain the integrated model. Section 5 shows some of the simulations that have been performed. Finally, Section 6 is a discussion.

2 Theoretical Background

In literature on workload and performance, it is often stated that in order to cope with situations of high task demands, people can make strategic choices [3], in order to protect performance degradation on the primary task. One of these choices is to increase the effort contributed to the task [11]. Unfortunately, as resources are limited, this can only be done for a limited amount of time. Another possibility is to make a shift to simpler strategies within the task, resulting in less use of working memory. An example of this can be found in driving behavior, where car drivers reduce their driving speed when faced with higher task demands [1].

Thus, while it can increase performance on the primary task, a reduced use of working memory can compromise secondary task goals, such as processing speed [3]. In addition, a decrease in the use of working memory can result in less attention available for peripheral cues [10]. Such attentional tunneling, together with less processing capacity available will result in a reduction of situation awareness in high workload conditions [2]. Finally, also decision making is affected by the contribution of effort (e.g., [12]). For example, research showed that in a situation with time-pressure people adjust their decision making strategy to less effortful strategies and will take more risky decisions [9]. The importance of information on human's own performance in the regulation of effort is shown in [11], where people invested higher levels of effort when they were informed of failure. Also, a lack of awareness of a human's own performance (e.g. as a consequence of low effort investment) may result in an impairment of effort regulation [3]. This is confirmed by Matthews and Desmond [8] who found an effort reduction as a consequence of a reduced awareness of performance impairment with the increase of fatigue. In the integration of the three models (as described in Section 3 below) the above described literature will be taken as a source of inspiration for making the connection between the models.

3 The Models Used as a Point of Departure

This section describes the three models that underlie the integrated agent model presented in this paper. First, the situation awareness model is described, followed by the decision making model, and finally, the model expressing the functional state.

Situation Awareness Model. The model for situation awareness used is taken from [5]. Fig. 1 shows the main concepts of the model in the upper left box, whereby the white circles denote the parameters of the model, whereas the dark circles represent processes. For the sake of brevity, the model will merely be described on a high level. For more details of the model, see [5].

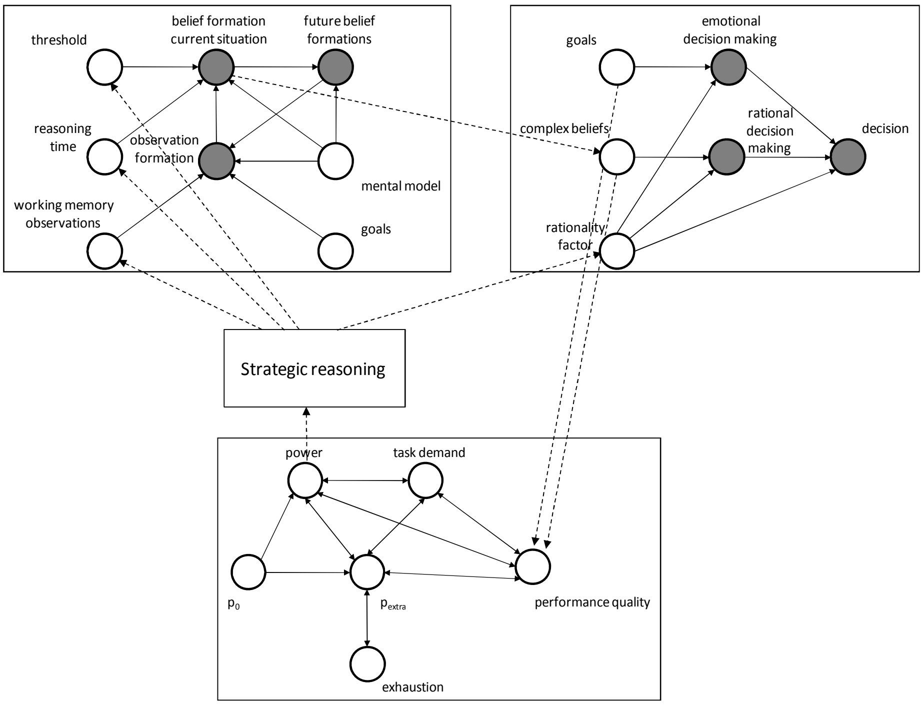


Fig. 1. Individual models and their integration

The essential idea behind the model (which tries to represent human situation awareness as defined in the literature) is that agents form beliefs about the current situation in the world. They do this based upon a mental model they have which expresses connections between these beliefs (e.g. if I have the belief that a holds, then I also believe that b holds). These beliefs are present on two levels: simple beliefs, which express simple facts about the world, and complex belief which encompass more complex statements about the world, and are triggered by combinations of multiple simple beliefs. Each of these beliefs has a certain activation value. The process of forming situation awareness starts when new observations are performed by the agent. This is then an input for the process *belief formation of current situation*. In the process, the observations cause updated activation levels of simple beliefs. Thereafter, the mental model is used to calculate new activation levels based upon the connections between beliefs. How many of such influence calculations are performed depends upon the reasoning time parameter. Furthermore, the threshold parameter expresses how high the activation level of a belief should be before being considered in the updating process. After the reasoning time has been reached, new activation levels for each of the beliefs are present, representing the judgment of the current situation of the agent. The next step is *formation of future belief*, which makes predications of occurrences in the future (in the form of time stamped beliefs with a certain activation level). Given that both the current and prospected situation are updated, a next round of observations can be performed (referred to as *observation formation*), the precise observations to be performed are also derived using the model. This depends upon the future beliefs (what does the agent think will happen), the goals (the agent focuses on important observations with respect to the goals), and the working memory size for observations (i.e. an agent cannot perform an infinite amount of observations). These observations are then performed, resulting in new input for the belief formation process, etcetera.

Decision Making Model. Based upon a certain judgment of the world (e.g. formed by the models for situation awareness described in Section 3.1), the agent can decide on what actions to perform. This is described in the model on decision making, following [6]. In the model (expressed in the upper right box of Fig. 1), emotions and rational utility are both considered as important elements in the decision making process. When looking at the emotional decision making part (the process *emotional decision making*), the options that can be decided upon are weighed based upon the feeling of the options with respect to the goals the agent has (e.g. a fighter pilot might have a negative feeling with the option of returning to base due to an enemy encounter bad for the goal of defeating the enemy). Each goal is hereby attributed with a certain weight, and each option has an emotional score with respect to each goal. A weighed sum is taken for each option. The more rational part (*rational decision making*) evaluated the options based upon the current situation (e.g. the complex beliefs such as part of the situation awareness model) and how well certain options are suited for this situation. Both processes result in a numerical evaluation of the options, and these are combined in the actual *decision making* process. How much each of the evaluations weighs depends on the rationality factor of the agent. For more details on the decision making model, see [6].

Functional State Model. The last model is a model representing the functional state of a human. A human's functional state can be defined as the combination of cognitive factors such as performance, effort and exhaustion (e.g., [4]). The model is shown graphically in the bottom part of Fig. 1 and presented in more detail in [7]. The model describes how an agent selects the amount of power to provided (i.e. how much effort does the agent want to put into a certain process). It is assumed that the agent strives for a certain *performance quality*. In order to achieve this quality, the agent must meet certain *task demands*. To meet these demands, the agent can input a certain *power* in the particular task at hand. This power is a combination of the basic power (p_o) and the extra power (p_{extra}). The basic power is inspired by a critical point, which is often seen in literature on exercises and sports. Once an agent needs to provide power above this point (i.e., the agent needs to provide extra power p_{extra}) the agent eventually will become exhausted. Once the exhaustion level reaches a certain level, the agent can no longer provide this additional power, and fall back to the basic power level.

4 Integrating the Three Models

In this Section, the three models that have been explained independently in Section 3 are combined into one model. This then results in a full agent to determine how much effort to spend, derive the situation using the selected appropriate effort within the situation awareness model, and derive an appropriate action given the perceived situation. How these models are connected will be explained in Section 4.1.

Thereafter, the strategic reasoning which takes place in various parts of the combined model is explained in more detail.

4.1 Interactions between the Models

Fig. 1 also shows the connections between the three models. The dashed arrows are the links between the concepts in the different models. Starting with the model for the *functional state* of the agent determines how much power to provide, this amount of power is an input for a *strategic component* that determines how to divide this power across the *situation awareness model* and the *decision making model*. To be more precise, it determines what value to select for:

1. The threshold in the situation awareness model (when are states considered).
2. The reasoning time in the situation awareness model (how much time is available to make calculations using the mental model).
3. The amount of working memory to be spent on observations to feed the situation awareness model.
4. The rationality factor in the decision making process (are more shortcuts used or is there more time to make a rational choice).

How these choices are precisely made and how these are quantified is explained in Section 4.2. Note that these choices are based upon the description of the relevant work as presented in Section 2 and are not trivial to define, especially due to the fact that the literature often does not describe these relationships in a very precise manner. The second element is to connect the *situation awareness model* with the *decision making model*. This combination is established by means of the complex beliefs about the current situation (i.e., the activation levels thereof). This judgment of the situation can be used to derive what options are appropriate.

The last link between the models is the derivation of the performance quality. The idea is that the performance quality can be determined by means of the goals that have been set by the agent (which are part of both the *decision making model* and the *situation awareness model*) and the current judgment of the situation (i.e. the complex beliefs), i.e. the performance quality expresses in how far the current situation (at least he situation perceived by the agent) contributes to the current goals. Note that this is not an objective measure, but the judgment of the agent itself, which is used as a steering instrument by the agent. It is assumed that each goal has a certain activation level (as already explained in the individual models):

goal_activation_level(goal, t)

Furthermore, the complex beliefs have a certain activation value as well:

complex_belief_activation_level(complex_belief, t)

In order to derive the performance quality, knowledge is present in the agent which expresses how much a certain complex belief contributes to a certain goal:

contributes_to_goal(complex_belief, goal, t)

The performance quality is determined by calculating per goal in how far the current situation fulfills this particular goal:

goal_contribution(complex_belief, goal, t) = complex_belief_activation_level(complex_belief, t) ·
contributes_to_goal(complex_belief, goal, t)

Then, the maximum is taken across all complex beliefs as these already provide an integrated view of the whole situation.

$$\text{overall\_goal\_contribution(goal, t)} = \\ \max(\text{goal\_contribution(complex\_belief}_1\text{, goal, t)}, ..., \text{goal\_contribution(complex\_belief}_n\text{, goal, t)})$$

Finally, the weighed sum is taken over all the goals to derive the performance quality:

$$\text{current\_performance\_quality(t)} = \sum_{G:GOALS} \text{goal\_activation\_leve(G, t)} \cdot \text{overall\_goal\_contribution(G, t)}$$

4.2 Strategic Reasoning

Strategic reasoning takes place in two parts of the model. First of all, in the model of the functional state of the agent as the agent needs to determine how much power is to be provided. The second strategic choice takes place in the strategic division of resources among the various elements that require power in the other models. Again, these choices made in this component are grounded within Psychology and are formalizations of high-level theories found in the literature.

4.2.1 Determining the Extra Power to be Provided

The first step that the agent needs to take is to determine how much power it wants to provide in order to achieve the task at hand. How much power the agent will deliver mainly depends on the performance quality the agent wants to deliver (*desired performance quality*), and the *current performance quality*. For the agent, the precise relationship between the power being provided and the performance quality is not crisp and clear: the agent needs to undergo a process of trying to put more power in, and seeing whether that results in a suitable performance quality (i.e. the *current performance quality* is approximately equal to the *desired performance quality*). In case the agent is underperforming, it will provide more power; in case it is performing above the desired quality, it will tend to reduce the provided power. Essentially, the agent only varies the power provided in addition to the basic power level (i.e. p_{extra}). How much the agent will change its power setting can be determined by means of a number of alternative algorithms. The simplest algorithm involves a standard increase/decrease of the power with a value γ.

$$p_{extra}(t+\Delta t) = \text{Pos}(p_{extra}(t) - \gamma \cdot \text{th}(\sigma, \text{desired\_performance\_quality(t)}, \text{current\_performance\_quality(t)}) \cdot \Delta t)$$

In this formula, the function $th(\sigma,\tau,V)$ is a threshold function that maps the value V to the interval [-1,1] whereby values of $V > \tau$ result in a value greater than 0 (and vice versa), and a value equal to the threshold results in an evaluation to 0. An example of such a function is for instance:

$$\text{th}(\sigma, \tau, V) = (2 \cdot (1/(1+e^{-4\sigma(V-\tau)}))-1)$$

The function $Pos(X)$ evaluates to X in case $X \geq 0$ and 0 otherwise.

A second option to determine the power setting is to use a more advanced sensitivity-based approach whereby the agent takes the previously experiences influence of the provided power upon the performance quality into account. This can be formulated by means of a mathematical equation as follows:

$$p_{extra}(t+\Delta t) = \text{Pos}(p_{extra}(t) + \\ \beta \cdot (\text{p\_pq\_sens(t)} \cdot (\text{desired\_performance\_quality(t)} - \text{current\_performance\_quality(t)}) \cdot \Delta t)$$

where

$$p\_pq\_sens(t) = (p_{extra}(t) - p_{extra}(t-\Delta t))/ (current\_performance\_quality(t)-current\_performance\_quality(t-\Delta t))$$

The idea of the above equation is that the effect of a difference in additional power with respect to the performance quality is calculated, and this sensitivity is used to adapt the extra power to be provided.

The approach mentioned above can work well, but the disadvantage is that the agent does not know in advance whether the power provided results in a reasonable performance quality. It is more a matter of trial and error. A more realistic agent would try a certain power in his mind, and project whether this effort would indeed be sufficient. This is therefore the way in which it is assumed to take place in this paper as well. The agent performs one fictive run of the model (thereby using an own world model) to see whether the intended power results in a sufficient quality. Based upon this the agent can still make adjustments based upon the strategies described above.

4.2.2 Strategic Distribution of Resources

The second strategic part about which the agent needs to reason lies within the strategic division of the resources which are spent by the agent over the various parts of the reasoning. More in specific, the component determines how to spread resources over: (1) the threshold used to update the beliefs for situation awareness, (2) the reasoning time within the situation awareness model, (3) the working memory available to perform observations, and (4) the rationality factor used in the decision making process. In order to facilitate the strategic reasoning process, for each of the factors a translation to power needs to be made. Table 1 shows a mapping from the dedicated values for the parameters in the model to an equivalent value which expresses the actual power value.

Table 1. Mapping of model parameter values to power

Parameter	**Values**	**Power equivalents**
threshold	[0,1]	[*max_threshold_cost*, 0]
reasoning time	[0, number_of_connections]	[0, number_of_connections · *power_per_connection*]
working memory observations	[0, $\sum_{\forall o:observations}$ cost(o)]	[0, $\sum_{\forall o:observations}$ cost(o)]
rationality factor	[0,1]	[0, *full_rationality_cost*]

It can be seen that the threshold normally has a value between 0 and 1, whereby 0 indicates that for all beliefs the connections should be considered whereas 1 expresses that this should only be done for beliefs that are completely activated. The power equivalent is precisely the opposite: the higher the threshold, the less power it costs (since fewer connections need to be considered). The maximum cost (performing all calculations) in terms of power is a constant which is called *max_threshold_cost*. The reasoning time to perform updates is expressed in the number of cycles that are being passed, which is limited to the number of connections. It is assumed that for each cycle (i.e. each connection that is being calculated) a certain power (*power_per_connection*) is required to obtain a mapping to a power value. With respect to the working memory for observations a cost value is already associated with each observation (in terms of power), and the maximum value is simply the sum of all cost of all possible observations. Finally, for the rationality factor in the decision making process, a value between 0 and 1 is possible, expressing fully non-rational decision making (which is assumed to cost no power) and fully rational decision making. Full rational decision making is assumed to be associated with a power of *full_rationality_cost*.

Given that these mappings are present, the agent first of all needs to determine how to spread the total power it has decided to spend on the task across the various parameters. Currently, a simple algorithm is assumed which simply assigns fixed weights to the different parameters: $w_{threshold}$, $w_{reasoning\_time}$, $w_{wm\_observations}$, $w_{rationality}$. Hereby, the sum of the weights is required to be 1. Once the total power *p(t)* has been derived, the power spent on the various aspects is calculated by a simple multiplication:

$$p_{threshold}(t) = w_{threshold} \cdot p(t)$$
$$p_{reasoning\_time}(t) = w_{reasoning\_time} \cdot p(t)$$
$$p_{wm\_observations}(t) = w_{wm\_observations} \cdot p(t)$$
$$p_{rationality}(t) = w_{rationality} \cdot p(t)$$

In the next step, a translation of these values to an appropriate parameter value can take place.

$$v_{threshold}(t) = 1 - (p_{threshold}(t)/\text{max\_threshold\_cost})$$
$$v_{reasoning\_time} = p_{reasoning\_time}(t)/\text{power\_per\_connection}$$
$$v_{wm\_observation}(t) = p_{wm\_observation}(t)$$
$$v_{rationality}(t) = (p_{rationality}(t)/\text{full\_rationality\_cost})$$

Also within the strategic component more advanced strategies can be deployed such as a sensitivity-based approach whereby the weight of the parameter in the weighed sum expressed above is determined by the sensitivity of that parameter. For the sake of brevity, this option has however not been explored within this paper.

5 Simulation

In this Section, an extensive case study is conducted to evaluate the behavior of the integrated model. First, the case study itself is described, followed by the results of the application of the model.

5.1 Case Study Description

In this case, the case study concerns a military scenario obtained from domain experts. In the scenario a pilot has to detect whether (enemy) contacts (i.e. other planes in this case) are near and if so, what kind of threat these contacts pose. As a result, the pilot has to decide what action to undertake. The detection of the other planes is performed by means of a radar warning receiver, which can provide a number of *observations*, including certain intensities of beeps coming from the receiver (expressing for instance whether the enemy is near, or has the ability to fire a missile due to a locked radar), the direction of the other plane. The more detailed mental model used in the situation awareness model that relates these observations into judgments on the current situation is expressed in Appendix A[1]. Complex beliefs that are formed involve element such as whether the plane is a possible target of a hostile attack.

[1] `http://www.cs.vu.nl/~mhoogen/sa/sa_appendix_A.pdf`

Given that the agent is aware of the situation, there are 5 possible decisions available in this scenario: *fly cap* (start flying in a circle to patrol a certain dedicated area) *beam* (maneuver to prevent enemy radar detection), *beam dive* (beam and dive to a lower altitude) *run* (move away from the potentially hostile plane) and *maintain cap* (remain flying the cap). In order to decide upon these actions, five different goals can be active within the agent: *fly cap* (patrolling a certain area), *avoid detected* (avoid an enemy plane from detecting you), *avoid track* (avoid an enemy plane from tracking your positions), *avoid lock* (avoid an enemy plane from locking a radar upon you, resulting in the possibility of firing a missile), *defeat missile* (try to defeat a missile being fired at you). Given this scenario, two elements are set dynamically, namely the goals (see Section 5.1.1 on the approach used), and the world model itself (i.e. how do action influence, the world, and how are observations obtained from this world), presented in Section 5.1.2.

5.1.1 Goals

The activity value of each goal is determined at each point in time, taking the activity value of complex beliefs into account, such that the agent adjusts its goals based on the situation. The influences of each complex belief to the available goals are known to the agent beforehand:

influences_goal(complex_belief, goal, t)

And the total influence of all complex beliefs to a goal is calculated by taking into account this influence and the activation value of all complex beliefs:

total_goal_influence(goal, t) =
$\Sigma_{\text{CB:Complex Beliefs}}$ complex_belief_activation_level(complex_belief, t) ·influences_goal(complex_belief, goal, t)

Finally, the relative activation value of each goal is calculated by dividing the total_goal_influence by the sum of the total goal influences for all goals.

goal_activation_level(goal, t)= (total_goal_influence(goal, t)/ ($\Sigma_{\text{G:Goal:}}$ total_goal_influence(goal, t))*2

5.1.2 World Model

A world model has been developed to complete the cycle from actions derived by the agents to observations in the world. The world model has been developed in two parts. First of all, there is a standard development of the world (in this case the enemy taking the necessary steps to perform a full attack). This standard development consists of a table which indicates how observations contribute to other observations (e.g. an observation of another pilot having a lock on the plane will contribute to the observation that a missile is fired). This consists of numbers on the interval [-1, 1] where -1 indicates a very negative influence whereas 1 expresses a positive influence. Assume two observations o_1 and o_2 whereby the influence of o_1 upon o_2 is calculated. The new value for o_2 is then calculated as follows:

activation_value(o_2, t + Δt) = activation_value(o_2, t) +
(Pos((1- activation_value(o_2, t) · activation_value(o_1, t) · influence_value(o_1, o_2, t) +
Neg(activation_value(o_2, t) · activation_value(o_1, t) · influence_value(o_1, o_2, t)·Δt

First, a single observation is selected, after which the above equation is sequentially applied for each of the influencing observations, followed by the second observation being selected for recalculation, etcetera.

The agent can of course influence this standard development by means of performing certain actions in the world, which is the second part of the world model. This expresses how actions influence observations (e.g. a dive results in a negative influence on an observation of a lock on the plane). This is done in an identical manner as presented before for the standard development (except that it of course now concerns actions that influence the observations). The combination of the standard development with the actions then results in appropriate observation results for the agent.

5.2 Simulation Settings

In this section, some simulations are presented. First, the setting of the key values are presented, followed by the results. Note that due to the fact that not all details of the individual models have been presented, some model specific parameters for these models are not explained further. In all simulations that are shown, the weights in the strategic component were divided based upon the following weight values: $w_{threshold}$ and $w_{reasoning\_time}$ were both set to 0.4 and $w_{rationality}$ and $w_{wm\_observations}$ were both 0.1. In the translation of the power values to parameter values, the following model settings were used: *max_threshold_cost = 6*, *power_per_connection = 0.15*, *full_rationality_cost = 5*. Furthermore, the exhaustion budget (the maximum amount of exhaustion that can build up in the functional state) has been set to 1000 and no recovery was allowed (according to the FS model, exhaustion builds up with the extra power provided).

Simulations were performed varying the basic power between 0 and 100 and the desired performance quality between 0.5 and 1. Graphs of simulation results are presented that best represent the integrated model behavior in Figs 2 to 6). Fig. 2 and 3 present simulations with a relatively high desired performance quality of 0.8. In the situation where the basic power is low (Fig. 2), the extra power that is contributed increases each point in time as the current performance quality is always lower than the desired PQ. After time point 32, no more extra power can be contributed because the maximum exhaustion budget of 1000 is reached. When the basic power is high (Fig. 3), this is not the case as only a low amount of P_{extra} needs to be contributed in order to achieve the desired Performance Quality.

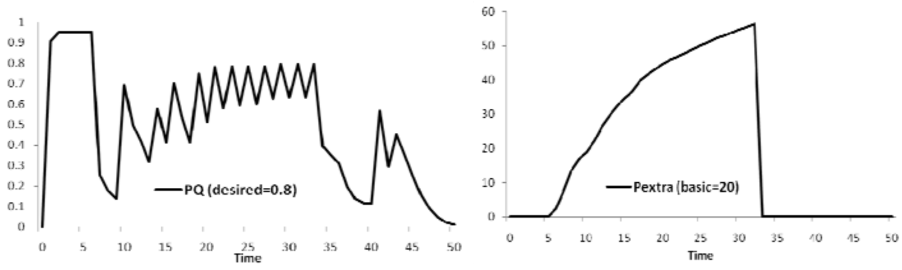
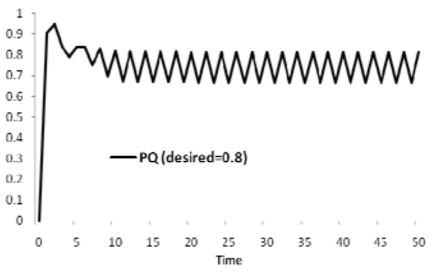


Fig. 2. Performance quality (a) and P_{extra} (b) with a desired PQ of 0.8 and a basic power of 20



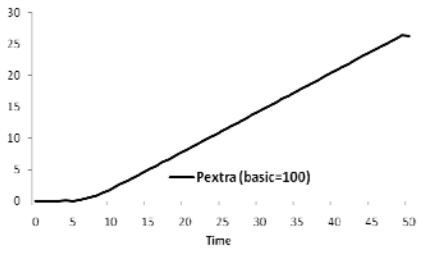
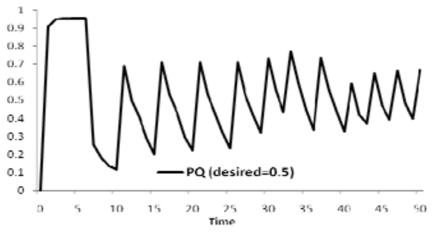


Fig. 3. Performance quality (a) and P_{extra} (b) with a desired PQ of 0.8 and a basic power of 100

In Fig. 4 it can be seen that when the desired performance quality is relatively low (i.e. 0.5), it is possible to achieve this level even though the basic power is low. Also, the extra power that is contributed is adjusted continuously, either downwards with the increase of performance quality or upwards with the decrease of performance quality. Fig. 5 shows the performance quality when the basic power is 100. In this case, no extra power needed to be contributed to achieve the desired performance quality of 0.5 (throughout the entire simulation, P_{extra} was zero).



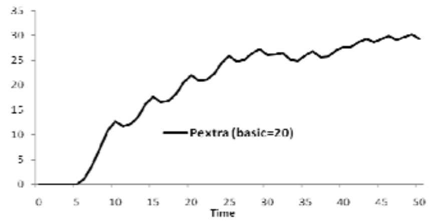
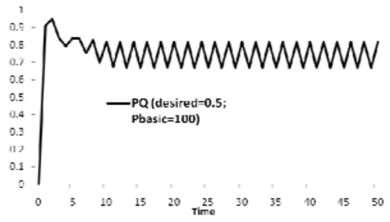


Fig. 4. Performance quality (a) and P_{extra} (b) with a desired PQ of 0.5 and a basic power of 20

In addition, a case was simulated where the basic power was very low (Fig. 6). The performance quality in this case stays very high, which shows the subjectivity of performance. As a consequence of the low power contribution, the agent's situation



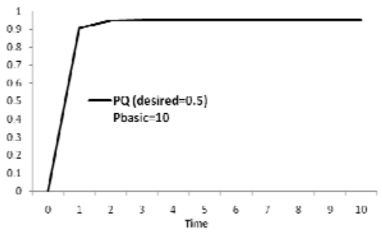


Fig. 5. Performance quality with a desired PQ of 0.5 and basic power of 100

Fig. 6. Performance quality with a basic power of 10

awareness is very low, which results in a low awareness on its own performance qualtiy. Since this is the case, no extra power will be contributed to improve the agent's situation awareness.

6 Discussion

In this paper, an integrated agent model was presented addressing the dynamics of mutually interacting situation awareness (e.g., [2]) and functional state (e.g., [1], [3], [4]) in decision making. By a number of simulation scenarios it was shown how a human's exhaustion and power, affect situation awareness and decision making. Although models exist for situation awareness or functional state separately, no models exist addressing the integrated process in a decision making context, as far as the authors know.

The integrated agent model was developed in the context of the national project Smart Bandits in cooperation with the National Aerospace Laboratory (NLR), aimed at developing simulation-based training facilities for fighter pilots. As a next step on the basis of the presented model it is planned to develop a software agent that can act as an automated enemy fighter for a trainee.

Acknowledgments. This research has been conducted and partially funded in the SmartBandits project in cooperation with the National Aerospace Laboratory in the Netherlands and the Royal Netherlands Air Force. We would like to thank the other participants of this project for their valuable input.

References

1. Cnossen, F., Rothengatter, T., Meijman, T.: Strategic changes in task performance in simulated car driving as an adaptive response to task demands. Transportation Research Part F 3, 123–140 (2000)
2. Endsley, M.R.: Toward a theory of Situation Awareness in dynamic systems. Human Factors 37, 32–64 (1995)
3. Hockey, G.R.J.: Compensatory control in the regulation of human performance under stress and high workload: a cognitive-energetical framework. Biological Psychology 45, 73–93 (1997)
4. Hockey, G.R.J.: Operator functional state as a framework for the assessment of performance degradation. In: Hockey, G.R.J., et al. (eds.) Operator Functional State. IOS Press (2003)
5. Hoogendoorn, M., van Lambalgen, R., Treur, J.: Modeling Situation Awareness in Human-Like Agents using Mental Models. In: Walsh, T. (ed.) Proc. of the Twenty-Second Intern. Joint Conference on Artificial Intelligence, IJCAI 2011, pp. 1697–1704 (2011)
6. Hoogendoorn, M., Merk, R.J., Treur, J.: An Agent Model for Decision Making Based upon Experiences Applied in the Domain of Fighter Pilots. In: Huang, X.J., et al. (eds.) Proceedings of the 10th IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2010, pp. 101–108. IEEE Computer Society Press (2010)

7. Klein, M.C.A., van Lambalgen, R., Treur, J.: An Agent Model for Analysis of Human Performance Quality. In: Huang, X.J., et al. (eds.) Proceedings of the 10th IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2010, pp. 181–188. IEEE Computer Society Press (2010)
8. Matthews, G., Desmond, P.A.: Task-induced fatigue states and simulated driving performance. The Quarterly Journal of Exp. Psy. 55A, 659–686 (2002)
9. Maule, A.J., Hockey, G.R.J., Bdzola, L.: Effects of time-pressure on decision-making under uncertainty: changes in affective state and information processing strategy. Acta Psychologica 104, 283–301 (2000)
10. Recarte, M.A., Nunes, L.M.: Mental workload while driving: effects on visual search, discrimination and decision making. Journal of Exp. Psy. Applied 9, 119–137 (2003)
11. Venables, L., Fairclough, S.H.: The influence of performance feedback on goal-setting and mental effort regulation. Motiv. Emot. 33, 63–74 (2009)
12. Wickens, C.D., Hollands, J.G.: Engineering Psychology and Human Performance. Prentice Hall, Upper Saddle River (2000)

Agent-Based Modelling for Understanding Sustainability

Sonja Pedell and Leon Sterling

Swinburne University of Technology
Faculty of Information and Communication Technologies, Hawthorn, Australia
{spedell,lsterling}@swin.edu.au

Abstract. Our aim is to demonstrate how agent-based models can play an important role in understanding sustainability. Here, we describe how the agent-based motivation models support the description of desirable outcomes and help to develop relevant and shared high-level goals in particularly complex areas such as sustainability. We focus on sustainable behaviour in households, and how to provide guidance for people to behave in a more environmentally-friendly manner. Our example demonstrates that the agent-based models are able to focus on the right questions when making decisions between alternatives in this multifaceted domain. With agent-based models, we aim to enable people to make the right choices for their particular circumstances and values. The agent paradigm is uniquely suited to understanding and representing the relevant goals, quality goals, and individual activities.

Keywords: Agent-based modelling, motivation models, sustainability, high-level goals, quality goals, values.

1 Introduction

We are using agent-oriented models from software engineering (AOSE) differently from goal models in the past. Former research projects have used the AOSE models for eliciting socially-oriented requirements and associated qualities specifically for the development of socio-technical systems [2, 24]. This research concluded that high-level goal models are well suited as an initial basis for shared understanding independent from a specific implementation, because these models represent the important characteristics of a domain. This representation can be useful when it is necessary to deal with complex topics, as agent models help people to think and focus on relevant aspects. Agent-based models constitute a shared basis for discussion and involve different stakeholders ranging from knowledgeable experts to novices. Also we are not developing the models with an insistence that the implemented system be agent-based, which is the case with methodologies such as Prometheus [18]. Here, our focus is not on developing a system. We aim to represent a complex socio-technical system and to help people to understand their role and their possibilities to act in it.

One area we have investigated is sustainable behaviour at home. With increasing use of energy, governments and universities have an increasing ambition to provide guidance, and to do research on sustainable behaviour and solutions to reach emission goals. For example The National Framework for Energy Efficiency aims to assist the market in supplying energy efficient technologies and processes [4]. Universities in particular have an important role to act in a sustainable way to lead by example and to teach necessary knowledge. Typically documents and processes that aim to guide sustainable behaviour from individuals are often hard to understand or to choose from, trivialised, not embedded in existing practise, or simply confusing for the reader as there are usually a lot different alternatives presented. We suggest here that AOSE models can help to represent sustainability activities, roles goals and quality goals that are connected to people's actual life situations, and therefore alleviate decisions, achieve behavioural change and consequently raise the likelihood to meet environmental targets.

We are interested in high-level goals for sustainable behaviour that are independent from specific technical solutions. We aim to direct the focus on relevant outcomes and necessary activities to achieve these outcomes before we think about technologies that might support these kinds of outcomes. For example, in an intergenerational relationship maintained over distance, goals such as *playing* and *gifting* and associated qualities goals such as *showing presence* and *share fun* are high-level goals. There are still many ways as to how this can be supported, but the social relationship stands at the centre and plays a crucial part that is independent of the technologies implemented [e.g. 20]. In order to influence people to show desirable behaviour we have to understand what is truly relevant for these people within the respective domain and make suggestions for behavioural change from there.

Here, we present a bottom-up approach in order to inform sustainable goals and quality goals derived from sustainability literature on effective sustainable activities. We use the models here for engaging in and raising awareness for this complex topic, to discuss important and shared goals and to facilitate decisions on which actions to take. Convincing people to change their behaviour has a lot to do with connecting with people and talking to them in a way that is relevant for them. Therefore it is not our aim to build detailed or exact representations of the world, but to discuss qualities and values surrounding the domain that are relevant for peoples' everyday lives. We aspire that with our approach people can embrace a topic on an intellectual and practice relevant level and increase their belief in self-efficacy [1]. The agent-based models represent activities that lead to a better outcome, as they account for and abstract human concepts such as individual attitudes, preferences and situations.

1.1 Benefits of Agent-Based Models for Representing Socio-technical Systems

Statistics show that people want to live a sustainable lifestyle and protect the environment [25]. However, Petkov et al., 2011 show that motivations for being sustainable are different [22]. Therefore, a socio-technical system needs to consider personal preferences and circumstances of people, but also represent facts and goals in a simple, understandable and flexible manner. Motivations and values need clear goals and solutions to be put into action. Here the agent-based goal models are extremely useful. In section 3 we will describe how agent-based models can be applied to enhance sustainable practices in helping to prioritise activities via goals. AOSE models are also suitable to show human values in form of quality goals. The concrete models deliver a shared basis to discuss values, interests, and self-efficacy.

We aim to come up with a process that helps us to find out how sustainable behaviour can be supported by agent-based diagrams. Here, we propose to use agent activities, roles and responsibilities, and goal models as an easy way to represent the complex relations that are sometimes long-term goals for e.g. house owners. AOSE models are very suitable as

- they are ideal for understanding complex topics because the concepts used in these models are suitable for expressing the behavioural aspects of individuals and their interactions [19].
- agents can be described in form of behavioural patterns. Different behaviours can still contribute to the same overall goal.

Agent-oriented models are suitable for modelling the socio-technical system of a sustainable household, because they represent the goals and motivations of roles and individuals, and quality goals can be used to discuss the quality of the high-level outcomes such as *saving water*. There are several possibilities to reach this goal depending on the circumstances: one can use less water, recycle water, have a rain water tank installed or all the three combined. Only the individual can decide what is the most fitting depending on the personal situation: what are the water targets, what can be done with grey water as one is not allowed to store it, and are there only indoor or also outdoor needs (e.g. vegetable garden) for water? In order to make a choice the person needs to understand what each of the solutions can provide. Our process leads from concrete best practice activities to the development of high-level quality goals that are shared by everyone, but can be substantiated and adapted for an individual context in a meaningful way. In the next section we provide the foundations of our approach and important definitions.

2 An Agent-Based Process for Quality Goal Formulation

Our process builds on the work of Sterling and Taveter [24]. Their work has focused on how to make high-level AOSE models palatable in design discussions. They define goal and role models that build part of a motivation layer. An agent is actively situated in an environment and is assumed as being purposeful in this environment. The models of goals and roles refer to knowledge about the problem domain. At the motivation layer, such knowledge is represented as a set of domain entities and relationships between them. A *goal* can be defined as *"a situation description that refers to the intended state of the environment"* (p. 30). Goals are based on motives and can have sub-goals. A *quality goal* is a non-functional or quality requirement of a socio-technical system.

We suggest several levels for using agent-based diagrams. Some of them are project specific in this case household specific (level one) and some are more general (level two, level three and level four). Starting with specific activities helps us to understand best practice of some people in specific circumstances living a more sustainable life. These different and alternative activities we can categorise in agent types and their activities for different roles. From there we can widen and generalise to the high-level goals and qualities that constitute a socio-technical system of sustainability. The different levels or steps include:

1) Activity plan for specific activities described in detail for an individual
2) Agent types for a specific area describing actual activities in less detail
3) Abstraction to roles – responsibilities and constraints
4) High-level motivational goal model (goals, quality goals and roles)

The next sections describe and define the different levels and the procedure of using agent-based models in more detail.

2.1 Agents and Activities

An *agent* is an entity that can act in the environment, perceive events, and reason. Reasoning means drawing inferences appropriate to the situation. Events that an agent perceives are caused by agents or other entities in the environment. Conversely, through acting, agents can affect entities in the environment. Agents can be humans as well as specialised hardware or software such as sensors.

2.2 Roles with Responsibilities and Constraints

Sterling and Taveter [24] define a *role* as some capacity or position that facilitates the system to achieve its goals. In their view, roles express functions, expectations and obligations of agents enacting them. They encompass these senses in the term *responsibilities*, which determine what an agent or set of agents enacting the role must do in order for a set of goals and quality goals to be achieved. In addition, a role may also have some *constraints* specifying conditions that the role must take into consideration when performing its responsibilities.

2.3 High-Level Goals and Quality Goals

Our end result is a simple model of motivations of a socio-technical system including goals and quality goals. By capturing and representing goals in AOSE models we make a commitment to important aspects of socio-technical systems. By externalizing them in a simple format the models become shared artefacts [17] that are able to sustain multiple interpretations across stakeholders and disciplines. Quality goals attached to goals allow a focus on understanding the reasons why people do things, or the essence of an attitude rather than describing a concrete action. This is what we call *values* here. In doing so, quality goals capture something that is more dynamic and fluid than other mechanisms found in usual software engineering practices. Non-functional goals do not generally have a direct relationship with functional goals [3]. In our approach there is a direct pairing between system goals and quality goals. Relating an abstract and unresolved quality attribute to a system goal enables a focus on a larger and shared social goal.

We use the construct of quality goals attached to functional goals as a way of representing quality attributes of socio-technical systems. Quality goals are designed to encapsulate aspects of the context into discussions. Garcia and Medinilla [8] describe high-level quality goals as a specific form of uncertainty that can be used as a descriptive complexity reduction mechanism and to model and discuss uncertainties in the environment. High-level goals associated with activities can act as a point of reference for discussing the usefulness of alternative activities to achieve these goals.

Instead of using the agent-based models in requirements elicitation for the development of a system we use them as shared artefacts for discussion [17] in the process of developing a shared understanding that can be used for goal formulation and strategies. The multi-agent paradigm offers benefits over other paradigms because the concepts used in modelling, such as roles, goals, and interactions, are part of everyday language and make it accessible for different stakeholders [21].

Goal models are useful at early stages of requirements analysis to arrive at a shared understanding [9, 14, 15]; and the agent metaphor is useful as it is able to represent the concepts that we want to capture for socio-technical systems, such as agents taking on roles associated with goals. These goals include quality attributes that are represented in a high-level pictorial view used to inform and gather input from stakeholders. In Sterling and Taveter's notation [24], goals are represented as parallelograms, quality goals are clouds, and roles are stick figures. These constructs can be connected using arcs, which indicate relationships between them (Figures 2-5).

3 Sustainable Households

3.1 Challenges for Guidance on Sustainable Domestic Behaviour

Usually the first step of interested people is to gain knowledge and facts on sustainability. We looked at literature, governmental guidelines and statistics providing information for the public [5, 11, 12, 25, 26]. We will illustrate how agent-based models can help to ground useful goals in such knowledge.

The whole area of sustainability is abstract and complex. How does a family motivated to live a sustainable lifestyle know what kind of sustainable behaviour realistically can be expected from them as part of a wider community? And on the other hand, how can a government that is interested in citizens behaving sustainably [5, 26] educate and encourage the right activities? One approach to support people in behaving sustainably is to showcase increasingly available applications and devices e.g. monitoring energy consumption [6, 7, 13]. These technologies only help when people understand what they are aiming for, have the right infrastructure in place and get more individualised feedback [10]. In addition, Fischer notes that monitors measure against a statistical average and people that are below this average often feel encouraged to use the resources that “they are entitled to” [6]. Competitiveness can lead to saving of resources, but the question is if people are not more successful if the main goal is *living sustainably* instead of comparing oneself to one’s neighbour. There is also a plethora of publications for environmentally friendly behaviour available for households [e.g. 5, 10, 11, 12, 26]. Most of these publications explain the need for sustainable behaviour and give concrete advice e.g. how to save energy and water at home. While all of this advice is useful and successful to some extent, it does not take into account the individual situation of different households, the climate, and personal preferences. Therefore, it is difficult for the individual to decide which actions are effective. When dealing with a complex area such as sustainability we would like concrete and simple advice. Yet, if the advice is too simplified it lacks relevance for the single household and its specific socio-economic situation.

Other specifics we have to consider when we focus within the large topic of sustainability on utility use in the home: firstly, set targets have to be supported by all people living in one home. Secondly there is a different level of insight into the topic (e.g. children might not understand the need straight away). Finally, non-home owners aren't able to make certain decisions on sustainability. This means that we have to consider several stakeholders within one home.

Again we see a role for the AOSE models in mapping the roles and responsibilities to overcome some of these challenges. The diagrams presented in the following sections show the different levels of abstraction: the first two lists are specific to one area or utility (level 1). Levels 2 to 4 are general to sustainability – here we aim to keep a light touch and stick to more general descriptions valid for all homes. The activities of level 2 are based on a body of literature on best practice sustainability accessible to the general public in libraries. The agent-based motivation models were created as a condensed version of the main and overlapping advice found in this literature – some of them containing 101 tips for sustainable living [e.g. 11, 12]. These tips differ largely in their effectiveness and their costs. For example, one book on water management advised on the same page "to cook vegetables in the microwave to save water" and "installing a rain water tank" for the same purpose [12]. If the high-level goal is *save water* then the latter advice is certainly more efficient unless it never rains in the region the rain water tank was installed. We suggest using the models for teaching people how to best accomplish high-level goals in utility management at home in accordance with the individual living situation. In order to ensure a high-level standard of the models, they were discussed in several discussion rounds with sustainability experts. On level 4 the specific agent activities are made more meaningful in two ways:

1. We are building motivational models that give high-level guidance. In formulating the specific advice in a more general way we prioritise and make a commitment to what goals we want to reach. That way people can think about which activities suit their lifestyle and contribute best to reaching their goals.

2. The attached quality goals express underlying values. Once there is a shared commitment among people it is easier to justify why certain activities should be given priority. Therefore the informed and shared models can be used in the future to motivate decisions on sustainability at home as goals are associated with values.

3.2 Specific Energy and Water Management Plan for One Household (Level 1)

Here we describe briefly the specific *energy management* plan for one household (level 1) that is located in a rather hot climate, is built with a lot of open areas, and has single pane windows. This *energy management* plan is based on the recommendations of a government certified sustainability consultant:

- Switch to hot water system with solar gas boost (eligible for rebates)
- Secondary glazing on windows for insulation
- Draft proofing (windows, doors, seal garage/office, self-sealing exhaust fans)
- Compartmentalize rooms so heating and cooling is minimized
- Switch devices off during night (stand-by), when not used (computer, lights)
- Use winter and summer settings on fans.

The following *water management* plan is based on above mentioned literature and is the actual plan of a specific Victorian household. It is the same household as above and the house owner has put together a plan after having an inspection by a plumber:

- Install grey water system for watering garden (governmental rebates)
- Connect rain water tank to toilet and dripping system (governmental rebates)
- Use double flush switch in toilet and install water saving shower heads (free)
- Use water filters, clean filters regularly and repair dripping taps

While in both cases best practice recommendations have been made and "theoretically" been embraced by the household owners it is unclear who is responsible for these changes and who is actually able to make decision on these changes. Therefore it is useful to group activities to different agent types as shown in the next section and then further to prioritise them regarding the efficiency of a goal and personal values expressed via the quality goals.

3.3 Agent-Types and Their Activities for *Energy* and *Water Management* (Level 2)

Specific agent types and actual activities for *energy management* and *water management* are shown in in Tables 1 and 2 respectively (level 2). Please note that the activities are far from complete but illustrate some of the most common examples found in a body of literature analysed by the authors [4, 5, 10, 11, 12, 25, 26].

Table 1. Agent-based activities for different roles (energy management)

Teacher: Teach knowledge about alternative energies (solar, wind, thermal…) and energy saving (e.g. insulation), give examples on consequences, provide best practice examples, communicate rules and policies, and discuss compromises.
Decision maker: use and subscribe to renewable and cleaner (non-carbon dioxide producers), insulate (seal windows, doors, roof), buy energy efficient appliances (fridge & freezer, washing machine).
Consumer/habitant: avoid fossil fuels, travel (travel together, walk, car pool, use public transport, cycle), eat and buy local products, save energy being energy smart (turn heat up early, turn down 1% to save 10%, switch off all lights, stand-by & computer).

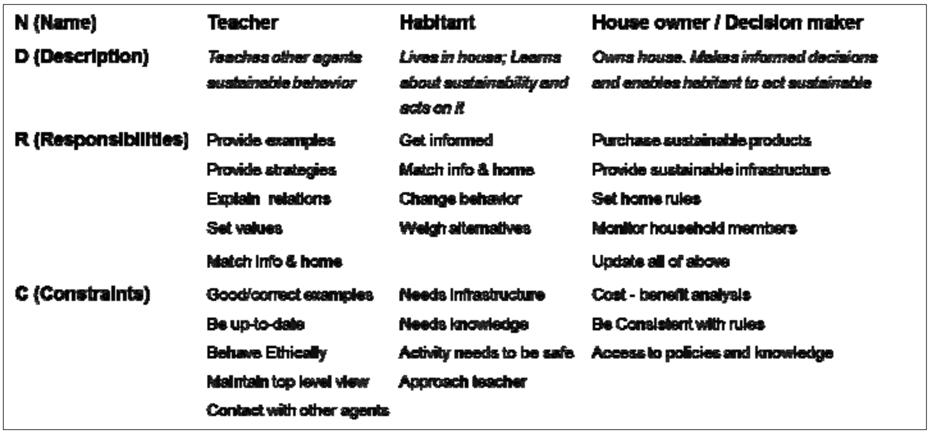
Table 2. Agent-based activities for different roles (water management)

Teacher: Teach knowledge about sustainability and ecological water systems (cycle of rainwater), give examples on consequences (salination, drought, pollution in larger context), best practice in household, communicate rules and policies, and discuss compromises.
Decision maker: Buy water saving head, water efficient appliances, rainwater tank, use biodegradable products, install grey water system, recycle water, reduce buying bottled water, install dual flush.
Consumer/habitant: Turn off taps, taps not dripping; short showers; follow '100 tips for water saving', drink tap water, use water again (bathwater for plants), no chemicals in water ways.

3.4 Responsibilities and Constraints (Level 3)

Figure 1 describes the responsibilities and constraints for the different roles involved in sustainable behaviour at home.

Note that the role 'teacher' can be played out by different resources of learning or acquiring best practice behaviour for the individual situation such as using literature, actual teachers, government consultants doing individual house inspections or other experts. In explicitly comparing the constraints and responsibilities of different roles the socio-technical system can be discussed and understood better in its completeness.



N (Name)	Teacher	Habitant	House owner / Decision maker
D (Description)	*Teaches other agents sustainable behavior*	*Lives in house; Learns about sustainability and acts on it*	*Owns house. Makes informed decisions and enables habitant to act sustainable*
R (Responsibilities)	Provide examples	Get informed	Purchase sustainable products
	Provide strategies	Match info & home	Provide sustainable infrastructure
	Explain relations	Change behavior	Set home rules
	Set values	Weigh alternatives	Monitor household members
	Match info & home		Update all of above
C (Constraints)	Good/correct examples	Needs infrastructure	Cost - benefit analysis
	Be up-to-date	Needs knowledge	Be Consistent with rules
	Behave Ethically	Activity needs to be safe	Access to policies and knowledge
	Maintain top level view	Approach teacher	
	Contact with other agents		

Fig. 1. Responsibilities and constraints for sustainable behaviour at home

3.5 High-Level Motivation Models (Level 4)

These motivation models cluster activities to high-level goals for *manage energy*, *manage water* and *manage waste* (figure 2, figure 3, and figure 4). These models are condensed to an overall top level goal model of the socio-technical system. The advantage of using a hierarchy of goal models is that no single model contains too much information. The high-level goals describe general activities such as *turn off* and *insulate*. People can think through what these goals mean for their specific situation at home.

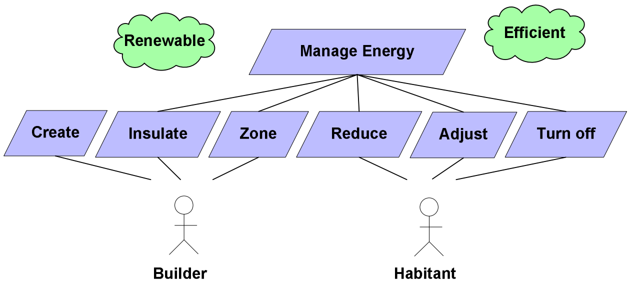


Fig. 2. Goal model specifically for *managing energy* at home

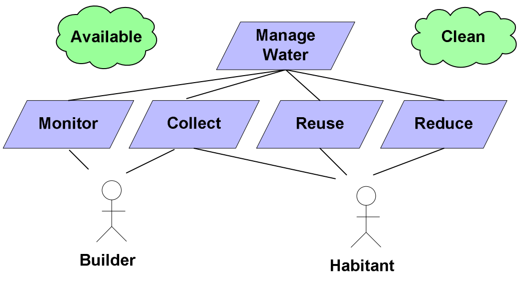


Fig. 3. Goal model specifically for *managing waste* at home

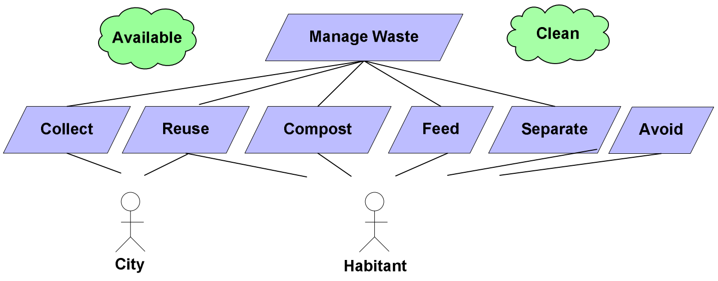


Fig. 4. Goal model specifically for *managing waste* at home

Figure 5 shows the high-level goal model for sustainable utility management at home. Again the model consists of goals, quality goals and roles. The quality goals take into consideration that while people want to live sustainably they are often not able to spend enough time and money on it (*manageable*).

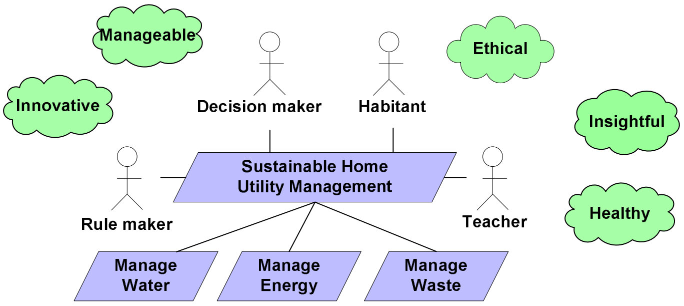


Fig. 5. High-level motivation model for sustainable behaviour at home (*utility mgt.*)

The quality aspect *innovative* encourages people to look actively for the latest solution and to keep up-to-date with sustainable developments. The different roles *decision maker* and *habitant* take into account that not every habitant in a home can make decisions on sustainability (e.g. children and people renting). The *rule maker* is an official body such as the city council or the regional government making rules on e.g. garden watering or recycling. The quality aspect *healthy* usually has high priority.

4 Discussion

We were treating and looking at sustainability as a social concept that can be supported by agent-based motivation models. We presented a method how to substantiate goals and quality goals in the complex socio-technical system of sustainable behaviour. In our method we used agent-oriented models as they are particularly suitable for modelling the social domain, representing the goals and motivations of individuals and their roles. We started with an approximation of sustainable household behaviour based on one specific household with a water and energy management plan based on expert advice. Then we derived best practice from literature on sustainable behaviour. In expert discussions we were categorizing the general activities to agent types and then to higher level roles with responsibilities and constraints. Based on these roles, we formulated goals and quality goals that are not tied to individual activities but to sustainable behaviour as a whole. These models matured over time in several internal expert discussions.

People require knowledge and awareness about complex topics. With the help of the motivation models, they can make a decision what they want to achieve and then choose how to achieve this. The different levels of agent-based diagrams help to structure what should be done and what can be done. The models are also useful in finding out the priorities and values of a person and continue from there to the next level of detail with a constant view on the main goals from level 4. The models enable people to choose activities and adjust their behavioural pattern according to set goals.

We see practical benefits of this proposed method of interleaving AOSE models and recommendations of sustainable behaviour. It was beneficial to have a shared explicit basis. Sustainability as many other complex social concepts has many facets and it is beneficial to agree on a high-level view when building a socio-technical system. A shared view helps to start communication (e.g. between different habitants of an household) and focus on the relevant goals in association with human and situational aspects within the socio-technical system. The motivational models allow the definition of quality goals. Often people jump too quickly into action as everyone has some ideas what sustainable behaviour means. Having a 'condensed' version and the whole socio-technical system represented in goal format on can derive and prioritize the activities in a top-down manner. The AOSE models helped to formulate high-level requirements for a wide audience that can be broken down or 'zoomed in' considering personal circumstances.

Certain value sets have so far been marginalized to date such as disclosing unsustainable behaviour and openly dealing with it. To harness these side by side with sustainable best practice can be positive. This method is promising to introduce a constructive, but critical way to provide sustainable education and discussions.

5 Conclusion

The role of the goal models is not simply the typical formal process of modelling to lead to the development of a system as in the traditional domain of software engineering. For us, they have become a way to think through problems, and to reach agreement about important objectives. However, a body of literature that looks at software engineering from a social science perspective recognises that models and other documentation in software engineering have been used for a long time as a way to think through problems, to reach agreements, and to elaborate the needs of stakeholders in a different way than simply feeding into a formal process of modelling for system design [16, 23]. In this sense it is not completely novel to use models as tools that are not directly connected to the development of a system. We use AOSE models to facilitate discussions around complex socio-technical systems. The models help us in discussions with externalising and making explicit the perspectives of different stakeholders on goals. In particular the quality goals help to explore different perspectives and values in a distinguishable manner.

Sustainability is the result of different behaviours based on multiple perspectives, varying sometimes contradictive knowledge and social values that needs to find a balance. In short, it is very complex and even though a lot of people aim for a more sustainable lifestyle it is difficult for them to find the more efficient and cheapest way to do so. Our discussions demonstrate that the AOSE models are easy to read and focus on relevant aspects. The high-level goal model needs to be flexible but also consistent with actual activities to ensure desirable outcomes. We are aiming for a match between the goal models and peoples' behaviour. Our discussions help us to include every stakeholder's perspective and include this perspective into the description of responsibilities and constraints. We also include qualities such as affordability (that is how *manageable* can be interpreted) of environmentally friendly solutions that is crucial for families but often not directly linked to the discussion of sustainability. This gives us a more realistic account if people can and will adapt their behaviour to live more environmentally friendly.

We believe that there are opportunities for agent-based motivation models to help governments to come up with useful guidelines and policies for its citizens. This contribution advances the state-of-the-art of agent computing practice in using agent-models in a novel way: we are providing an holistic approach looking at the whole socio-technical system (also containing context and individual values and priorities). This approach influences individual behaviour by involvement of different stakeholders in discussions represented by their roles. With this method, motivational models representing complex socio-technical systems can be created from extensive literature sources (bottom-up). Also high-level goals can then be broken down again in a target-oriented way in order to choose best practice activities through the examination of people's individual circumstances (top-down). As the next step, we are planning to present the models for sustainability to educators and embed the models as educational material in sustainability teaching.

Acknowledgments. We would like to thank all experts involved in the discussions providing us generously with critical feedback on the sustainability models.

References

1. Bandura, A.: Self-efficacy: Toward a Unifying Theory of Behavioral Change. Psychological Review 84, 191–215 (1977)
2. Boettcher, A.: Moving From Cultural Probes to Agent-Oriented Requirements Engineering. In: OZCHI 2006, pp. 20–24 (2006)
3. Chung, L.K., Nixon, B.A., Yu, E., Mylopoulos, J.: Non-Functional Requirements in Software Engineering. Kluwer Publishing (2000)
4. DRET (Department of Resources Energy and Tourism), What is the National Framework for Energy Efficiency? Department of Resources Energy and Tourism (2010), `http://www.ret.gov.au/Documents/mce/energy-eff/nfee/about/default.html` (viewed, June 2011)

5. DSE (Victorian Government Department of Sustainability and Environment). Energy Efficiency for Victoria Action Plan, The State of Victoria, Melbourne. Annual Report (2006), http://www.dse.vic.gov.au/CA256F310024B628/0/C4C97FE3979E7A88CA2575C4000A6869/$File/Energy+Efficiency+Action+Plan+2006.pdf
6. Fischer, C.: Feedback on household electricity consumption: a tool for saving energy? Energy Efficiency 1(1), 79–104 (2008)
7. Froehlich, J.: The Design of Eco-Feedback Technology. In: Proceedings of the CHI, Atlanta, GA, USA, April 10-15. ACM Press (2010)
8. Garcia, A., Medinilla, N.: The ambiguity criterion in software design. In: International Workshop on Living with Uncertainties. ACM (2007)
9. Guizzardi, R., Perini, A.: Analyzing requirements of knowledge management systems with the support of agent organizations. Journal of the Brazilian Computer Society (JBCS)-Special Issue on Agents Organizations 11(1), 51–62 (2005)
10. He, H.A., Greenberg, S., Huang, E.M.: One size does not fit all: Applying the Transtheoretical Model to Energy Feedback Technology Design. Department of Computer Science, University of Calgary, Calgary, Alberta, Canada (2009)
11. Healey, J.: Our energy future. Issues in Society, vol. 295. N.S.W. Spinney Press, Thirroul (2009)
12. Healey, J.: Water management. Issues in Society, vol. 288. N.S.W. Spinney Press, Thirroul (2009)
13. Holmes, T.: Eco-visualization: Combining Art and Technology to Reduce Energy Consumption. In: Proc. of the C&C, Washington, USA, June 13-15. ACM (2007)
14. Iqbal, R., James, J., Gatward, R.: Designing with ethnography: An integrative approach to CSCW design. Advanced Engineering Informatics 19, 81–92 (2005)
15. Jureta, I.J., Faulkner, S.: Clarifying goal models. In: Proc. ER 2007, pp.139-144 (2007)
16. MacLean, A., Bellotti, V., Young, R.M.: What rationale is there in design? In: Diaper, D., Gilmore, D.J., Cockton, G., Shackel, B. (eds.) Proceedings of the IFIP TC13 Third International Conference on Human-Computer Interaction, pp. 207–212 (1990)
17. Paay, J., Sterling, L., Vetere, F., Howard, S., Boettcher, A.: Engineering the Social: The Role of Shared Artifacts. IJHCS 67(5), 437–454 (2009)
18. Padgham, L., Winikoff, M.: Developing Intelligent Agent Systems: A practical guide. Wiley (2004)
19. Pavon, J., Arroyo, M., Hassan, S., Sansores, C.: Agent-based modelling and simulation for the analysis of social patterns. Pattern Recognition Letters 29(8), 1039–1048 (2008)
20. Pedell, S., Miller, T., Vetere, F., et al.: Having fun at home: interleaving fieldwork and goal models. In: Proc. of OZCHI 2009, pp. 309–312 (2009)
21. Pedell, S., Vetere, F., Howard, S., Miller, T., Sterling, L.: Shared artefacts as participatory Babelfish. In: Proceedings of PDC 2010, pp. 167–170. ACM (2010)
22. Petkov, P., Köbler, F., Foth, M., Krcmar, H.: Motivating domestic energy conservation through comparative, community-based feedback in mobile and social media. In: C&T 2011, June 29 -July 2. Brisbane (2011)
23. Randall, D., Hughes, J., Shapir, D.: Steps toward a partnership: ethnography and system design. In: Jirotka, M., Goguen, J. (eds.) Requirements Engineering: Social and Technical Issues, pp. 241–254. Academic Press (1994)
24. Sterling, L., Taveter, K.: The Art of Agent-Oriented Modelling. MIT Press (2009)
25. Sustainability Victoria, Report: (2009), http://greenlightreport.sustainability.vic.gov.au
26. The Australian Government's sustainability portal: http://www.livinggreener.gov.au

Modelling Joint Decision Making Processes Involving Emotion-Related Valuing and Empathic Understanding

Jan Treur

VU University Amsterdam, Agent Systems Research Group
De Boelelaan 1081, 1081 HV, Amsterdam, The Netherlands
treur@cs.vu.nl
http://www.cs.vu.nl/~treur

Abstract. In this paper a social agent model for joint decision making is presented addressing the role of mutually acknowledged empathic understanding in the decision making. The model is based on principles from recent neurological theories on mirror neurons, internal simulation, and emotion-related valuing. Emotion-related valuing of decision options and mutual contagion of intentions and emotions between agents are used as a basis for mutual empathic understanding and convergence of decisions and their associated emotions.

1 Introduction

An important aspect in group functioning is the ability for joint decision making. In recent years developments in neuroscience have clarified some of the mechanisms underlying such processes (e.g., [7, 13, 18]). Two interrelated core concepts in this discipline are mirror neurons and internal simulation. Mirror neurons are neurons that not only have the function to prepare for a certain action or body change, but are also activated upon observing somebody else who is performing or tending to perform this action or body change (e.g., [23, 32, 35, 39]). Internal simulation is mental processing that copies processes that may take place externally, for example, in another individual (e.g., [8, 10, 16, 17, 20]). On the one hand, mirror neurons and internal simulation have been put forward as a basic mechanism for imitation and contagion of actions and emotions; on the other hand, they have been related to empathy; e.g., [23]. In this way mirror neurons and internal simulation provide a basis both to mutually tune individual intentions and emotions and to develop mutual empathic understanding between persons (e.g., [16, 17, 33, 36]). Usually these two aspects are addressed separately, but in joint decision making processes they both play their roles in order to achieve solidly grounded joint decisions.

Empathic understanding can concern both cognitive (e.g., knowing or believing) and affective (e.g., feeling) aspects. Affective and cognitive understanding are often related to each other, as any cognitive state triggers an associated emotional response which is the basis of the related feeling (e.g., [8, 10, 11, 12]). Usually in an individual decision making process, before a decision option is chosen an internal simulation takes place to predict the expected effects of the option (e.g., [2, 8, 10, 11, 12, 28]). Based on these predicted effects a valuation of the option takes place, which may involve or even be mainly based on the affective state associated to this effect (e.g., [1, 8, 9, 11, 29, 31]). To achieve a solid joint decision, a shared feeling and valuation for the chosen option are important, and also mutual recognition of this sharedness. When this is achieved, a common decision has a strong shared emotional grounding as the group members do not only intend to follow that option, but they also share a good feeling about it, and they have (mutually acknowledged) empathic understanding of how other persons feel about the options. The latter may be important as well for acceptance of non-joint decisions.

The obtained social agent model can be used as a basis for the design of human-like virtual agents for simulation-based training or in gaming, or for virtual stories. For the first type of application the idea is to develop a number of virtual agents cooperating with a human trainee as a team in an decision making task. For the second type of application the idea is to design a system for agent-based virtual stories in which, for example, persons play a role which can be based on the presented model.

In this paper, first in Section 2 some core concepts used are briefly reviewed. Next, in Section 3 the social agent model is presented. In Section 4 some of the explored simulation scenarios are discussed. Finally, Section 5 is a discussion.

2 Mirroring, Internal Simulation and Emotion-Related Valuing

Two concepts used here as a basis are mirror neurons and internal simulation; in combination they provide an individual's mental function of mirroring mental processes of another individual (see also [39]). Mirror neurons are not only firing when a subject is preparing an action, but also when somebody else is performing or preparing this action and the subject just observes that. They have first been found in monkeys (cf. [15, 34]), and after that it has been assumed that similar types of neurons also occur in humans, with empirical support, for example, in [25] based on fMRI, and [14, 30] based on single cell experiments with epilepsy patients (see also [23, 24, 27]). The effect of activation of mirror neurons is context-dependent. A specific type of neurons has been suggested to be able to indicate such a context. They are assumed to indicate self-other distinction and exert control by allowing or suppressing action execution; e.g., [6, 19, 24], and [23], pp. 196-203.

Activation states of mirror neurons play an important role in *mirroring* mental processes of other persons by *internal simulation*. In [26] the following causal chain for generation of felt emotions is suggested (see also [12], pp. 114-116):

sensory representation $\rightarrow$ preparation for bodily changes $\rightarrow$ expressed bodily changes $\rightarrow$
emotion felt = based on sensory representation of (sensed) bodily changes

As a further step *as-if body loops* were introduced bypassing actually expressed bodily changes (cf. [8], pp. 155-158; see also [10], pp. 79-80; [11, 12]):

sensory representation $\rightarrow$ preparation for bodily changes = emotional response $\rightarrow$
emotion felt = based on sensory representation of (simulated) bodily changes

An as-if body loop describes an *internal simulation* of the bodily processes, without actually affecting the body, comparable to simulation in order to perform, for example, prediction, mindreading or imagination; e.g., [2], [16], [17], [20], [28]. The feelings generated in this way play an important role in valuing predicted or imagined effects of actions, in relation to amygdala activations; see, e.g., [29], [31]. The emotional response and feeling mutually affect each other in a bidirectional manner: an as-if body loop usually has a cyclic form (see, for example, [11], pp. 91-92; [12], pp. 119-122):

> emotion felt = based on sensory representation of (simulated) bodily changes →
> preparation for bodily changes = emotional response

As mirror neurons make that some specific sensory input (an observed action of another person) directly links to related preparation states, they combine well with as-if body loops; see also [39], or [12], pp. 102-104. In this way states of other persons lead to activation of some of a person's corresponding own states that at the same time play a role in the person's own feelings and decisions for actions. This provides an effective mechanism for how observed actions and feelings and own actions and feelings are tuned to each other. Thus a mechanism is obtained which explains how in a social context persons fundamentally affect each other's individual decisions and states, including feelings. Moreover, it is also the basis for empathic understanding of other persons' preferences and feelings. Both the tuning and convergence of action tendencies and the mutual empathic understanding (even when finally no common option is decided for) play a crucial role in joint decision making processes.

3 The Social Agent Model

The issues and perspectives briefly reviewed in the introduction and Section 2 have been used as a basis for the neurologically inspired cognitive agent model presented below (for an overview, see Fig. 1); in summary:

- Decision making is based on *emotion-related valuing* of the *predicted effects* of each action option
- Both the tendency to go for an action and the associated emotion are transferred between agents via *mirroring processes* using *internal simulation*
- These mirroring processes at the same time induce a gradual process of mutually *tuning* the considered actions and their emotion-related valuations, and the development of mutual *empathic understanding*
- The outcome of such a joint decision process in principle involves three elements:
  - a *common action* option
  - a *shared positive feeling* and *valuation* for the effect of this action option
  - mutually *acknowledged empathic understanding* for both the action and feeling
- In case of an outcome without a common choice for an action option, the process results in mutually *acknowledged empathic understanding*
- The mutually acknowledged empathic understanding is based on the following criteria:
  (a) Showing the same state as the other agent (nonverbal part of the empathic response)
  (b) Telling that the other agent has this state (verbal part of the empathic response)
  Assuming true, faithful nonverbal and verbal expression, these criteria are in line with the criteria of empathy for affective states formulated in [36].

In the model s denotes a *stimulus*, a an *option* for an *action* to be decided about, and e a world state which is an *effect* of the action. The effect state e is *valued* by associating a *feeling* state b to it, which is considered to be positive for the agent (e.g., in accordance with a goal). The state properties used in the model are summarised in Table 1.

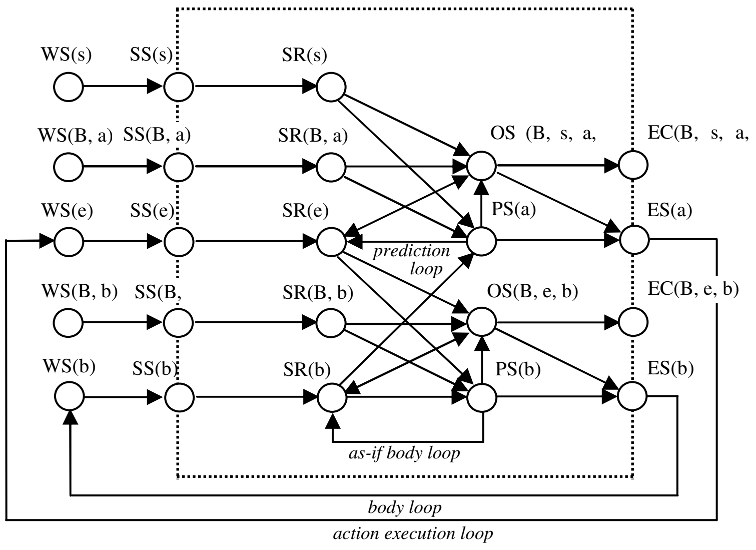


Fig. 1. Overview of the social agent model

The social agent model uses ownership states for actions a and their effects e, both for self and other agents, specified by OS(B, s, a, e) with B another agent or self, respectively (see Fig. 1). Similarly, ownership states are used for emotions indicated by body state b, both for self and other agents, specified by OS(B, e, b) with B another agent or self. As an example, the four arrows to OS(B, s, a, e) in Fig. 1 show that an ownership state OS(B, s, a, e) is affected by the preparation state PS(a) for the action a, the sensory representation SR(b) of the emotion-related value b for the predicted effect e, the sensory representation SR(s) of the stimulus s, and the sensory representation SR(B) of the agent B. Note that s, a, e, b, and B are parameters for stimuli, actions, effects, body states, and agents. In a given agent model multiple instances of each of them can occur.

Table 1. State properties used

notation	description
WS(W)	world state W: for an action a of agent B, a feeling b of agent B, a stimulus s, effect e, or an emotion indicated by body state b
SS(W)	sensor state for W
SR(W)	sensory representation of W
PS(X)	preparation state for X: action a or expressing emotion by body state b
ES(X)	execution state for X: action a or expressing emotion by body state b
OS(B, s, a, e)	ownership state for B of action a with effect e and stimulus s
OS(B, e, b)	ownership state for B of emotion indicated by body state b and effect e
EC(B, s, a, e)	communication to B of ownership for B of action a with effect e and stimulus s
EC(B, e, b)	communication to B of ownership for B of emotion indicated by b and effect e

Prediction of effects of prepared actions is modelled using the connection from the preparation PS(a) of the action a to the sensory representation SR(e) of the effect e. Suppression of the sensory representation of a predicted effect (according to, e.g., [3], [4], [28]) is modelled by the (inhibiting) connection from the ownership state OS(B, s, a, e) to sensory representation SR(e). The control exerted by the ownership state for action a is modelled by the connection from OS(B, s, a, e) to ES(a). Communicating ownership for an action (a way of expressing recognition of the other person's states, as a verbal part of showing empathic understanding) is modelled by the connection from the ownership state OS(B, s, a, e) to the communication effector state EC(B, s, a, e). Similarly, communicating of ownership for an emotion for effect e indicated by b is modelled by the connection from the ownership state OS(B, e, b) to the communication effector state EC(B, e, b). Connections between state properties (the arrows in Fig. 1) have weights, as indicated in Table 2.

Table 2. Overview of the connections and their weights

from states	to state	weights	LP
SS(W)	SR(W)	ω_{1W}	LP1
PS(a), OS(B, s, a, e), SS(e)	SR(e)	ω_{21e}, ω_{22e}, ω_{23e}	LP2
PS(b), OS(B, e, b), SS(b)	SR(b)	ω_{21b}, ω_{22b}, ω_{23b}	
SR(s), SR(b), SR(B, a)	PS(a)	ω_{31a}, ω_{32a}, ω_{33a}	LP3
SR(e), SR(b), SR(B, b)	PS(b)	ω_{31b}, ω_{32b}, ω_{33b}	
SR(B, a), SR(s), PS(a), SR(e)	OS(B, s, a, e)	ω_{41a}, ω_{42a}, ω_{43a}, ω_{44a}	LP4
SR(B, b), SR(e), PS(b), SR(b)	OS(B, e, b)	ω_{41b}, ω_{42b}, ω_{43b}, ω_{44b}	
OS(B, s, a, e), PS(a)	ES(a)	ω_{51a}, ω_{52a}	LP5
OS(B, e, b), PS(b)	ES(b)	ω_{51b}, ω_{52b}	
ES(a)	WS(e)	ω_{6e}	LP6
ES(b)	WS(b)	ω_{6b}	
WS(W)	SS(W)	ω_{7W}	LP7
OS(B, s, a, e)	EC(B, s, a, e)	ω_{8a}	LP8
OS(B, e, b)	EC(B, e, b)	ω_{8b}	

In this table the column LP refers to the (temporally) Local Properties LP1 to LP9 presented below. A weight usually has a value between *-1* and *1* and may depend on the specific instance for agent B, stimulus s, action a and/or effect state b involved. Note that in general weights are assumed non-negative, except for inhibiting connections, such as ω_{22e} which models suppression of the sensory representation of effect e, and ω_{22b} which models suppression of the sensory representation of body state b.

Below, the dynamics following the connections between the states in Fig. 1 are described in more detail. This is done for each state by a dynamic property specifying how the activation value for this state is updated based on the activation values of the states connected to it (the incoming arrows in Fig. 1). Note that in these property specifications s, a, e, b, and B are parameters for stimuli, actions, effects, body states, and agents, respectively; multiple instances for each of them can be used in a given agent model. The agent model has been computationally formalised using the hybrid modeling language LEADSTO; cf. [5]. Within LEADSTO a dynamic property or temporal causal relation a $\twoheadrightarrow$ b denotes that when a state property a (or conjunction thereof) occurs, then after a certain time delay, state property b will occur. Below, this delay will be taken as a uniform time step Δt. Each time first a semiformal description is given, and next a formal specification in the hybrid LEADSTO format. Parameter γ indicates the speed by which an activation level is updated based on received input from other states. During processing, each state property has an activation level represented by a real number between 0 and 1; variables V (possibly with subscripts) run over these values. In dynamic property specifications, this is added as a last argument to the state property expressions (an alternative notation activation(p, V) with p a state property has not been used for the sake of notational simplicity).

Below, f is a function for which different choices can be made, for example, the identity function $f(W) = W$ or a combination function based on a continuous logistic threshold function of the form

$$th(\sigma, \tau, X) = \left(\frac{1}{1+e^{-\sigma(X-\tau)}} - \frac{1}{1+e^{\sigma\tau}}\right)(1+e^{-\sigma\tau}) \quad \text{or} \quad th(\sigma, \tau, X) = \frac{1}{1+e^{-\sigma(X-\tau)}}$$

with σ a steepness and τ a threshold value, when $X \geq 0$, and 0 when $X < 0$. Note that for higher values of $\sigma\tau$ (e.g., $\sigma > 20/\tau$) the right hand side threshold function can be used as an approximation. In the example simulations, in LP1, LP6, and LP7, f is taken the identity function $f(W) = W$, and for the other states f is a combination function based on the logistic threshold function: $f(X_1, X_2) = th(\sigma, \tau, X_1+X_2)$, and similarly for other numbers of arguments; other types of combination functions might be used as well. For example values for τ and σ, see Table 3 in Section 4.

The first property LP1 describes how sensory representations are generated for any state W, indicating a stimulus s, an action a of an agent B, or a feeling b of an agent B.

LP1 Sensory representation of w based on a sensor state for w

If the sensor state for W has level V_1
and the sensory representation of W has level V_2
then after duration Δt the sensory representation of W will have level $V_2 + \gamma\,[f(\omega_{1W}V_1) - V_2]\ \Delta t$.

SS(W, V_1) & SR(W, V_2) $\twoheadrightarrow$ SR(W, $V_2 + \gamma\,[\,f(\omega_{1W}V_1) - V_2\,]\,\Delta t$)

The sensory representation of an effect state e is not only affected by a corresponding sensor state for e (affected by the world state), but also by two action-related states:

- via the *predictive loop* by a preparation state, as a way of *internal simulation* to predict the effect e of a prepared action a
- by an inhibiting connection from the self-ownership state, to *suppress* the sensory representation of the *effect* e of the action a, once it is going to be initiated; e.g., [3], [4]

This is expressed in dynamic property LP2. Note that for this suppressing effect the connection weight ω_{22e} from ownership state for action a to sensory representation for effect e is taken negative, for example $\omega_{22e} = -0.2$. Dynamic property LP2b specifies a similar temporal relationship for update of the sensory representation of a body state, and thus models *internal simulation* by an *as-if body loop*.

LP2e Sensory representation for an effect state e
If the preparation state for action a has level V_1
and the ownership of action a for B and s has level V_2
and the sensor state for state e has level V_3 and the sensory representation state of e has level V_4
then after Δt the sensory representation of e will have level $V_4 + \gamma[f(\omega_{21e}V_1, \omega_{22e}V_2, \omega_{23e}V_3) - V_4]\,\Delta t$.

$\text{PS}(a, V_1)\ \&\ \text{OS}(B, s, a, e, V_2)\ \&\ \text{SS}(b, V_3)\ \&\ \text{SR}(b, V_4) \twoheadrightarrow \text{SR}(b, V_4 + \gamma[f(\omega_{21e}V_1, \omega_{22e}V_2, \omega_{23e}V_3) - V_4]\,\Delta t)$

LP2b Sensory representation for a body state b
If the preparation state for body state b has level V_1
and the ownership of body state b for B and b, and e has level V_2
and the sensor state for state b has level V_3 and the sensory representation of state b has level V_4
then after Δt the sensory representation of b will have level $V_4 + \gamma[f(\omega_{21b}V_1, \omega_{22b}V_2, \omega_{23b}V_3) - V_4]\,\Delta t$.

$\text{PS}(a, V_1)\ \&\ \text{OS}(B, e, b, V_2)\ \&\ \text{SS}(b, V_3)\ \&\ \text{SR}(b, V_4) \twoheadrightarrow \text{SR}(b, V_4 + \gamma[f(\omega_{21b}V_1, \omega_{22ob}V_2, \omega_{23b}V_3) - V_4]\Delta t)$

Preparation for action a is affected by

- the sensory representation of stimulus s
- the body state b associated to the predicted effect e of the action,
- observation of the action (tendency) in another agent

The first bullet is an external trigger for the action. The second bullet models the impact of the result b of the *emotion-related valuing* of the action effect e. The third bullet models the *mirroring* effect for the action as observed as a tendency in another agent. Similarly for the preparation for a body state b; here the sensory representation of the effect e serves as a trigger, and the emotion state of another agent is mirrored.

LP3a Preparing for an action a
If sensory representation of s has level V_1 and sensory representation of body state b has level V_2
and sensory representation of B for a has level V_3 and the preparation for action a has level V_4
then after Δt preparation for action a will have level $V_4 + \gamma[f(\omega_{31a}V_1, \omega_{32a}V_2, \omega_{33Ba}V_3) - V_4]\,\Delta t$.

$\text{SR}(s,V_1)\ \&\ \text{SR}(b,V_2)\ \&\ \text{SR}(B, a)\ \&\ \text{PS}(a, V_4) \twoheadrightarrow \text{PS}(a, V_4 + \gamma[f(\omega_{31a}V_1, \omega_{32a}V_2, \omega_{33Ba}V_3) - V_4]\,\Delta t)$

LP3b Preparing for a body state b
If sensory representation of e has level V_1 and sensory representation of b has level V_2
and sensory representation of B for b has level V_3 and the preparation for action a has level V_4
then after Δt preparation for action a will have level $V_4 + \gamma[f(\omega_{31b}V_1, \omega_{32b}V_2, \omega_{33Bb}V_3) - V_4]\,\Delta t$.

$\text{SR}(e,V_1)\ \&\ \text{SR}(b,V_2)\ \&\ \text{SR}(B, b)\ \&\ \text{PS}(b, V_4) \twoheadrightarrow \text{PS}(b, V_4 + \gamma[f(\omega_{31b}V_1, \omega_{32b}V_2, \omega_{33Bb}V_3) - V_4]\,\Delta t)$

Ownership states for an action a or body state b are generated by LP4a and LP4b. They keep track of the agent's context with respect to the action or body state. This context concerns both the agent self and the other agents and their extent of ownership of the action or body change; in this sense it is a basis for attribution to an agent, and includes self-other distinction. Moreover, a self-ownership is used to control execution of prepared actions or body states, like super mirror neurons are assumed to do. For example, in case the agent B is self, the ownership state for action a strengthens the initiative to perform a as a self-generated action: executing a prepared action depends on whether a certain activation level of the ownership state for the agent self is available for this action. This is how control over the execution of the action (go/no-go decision) is exerted, and can, for example, be used to veto the action in a stage of preparation.

LP4a Generating an ownership state for B and a

If the sensory representation of (tendency for) action a in agent B has level V_1
and the sensory representation of s has level V_2 and the preparation for action a has level V_3
and the sensory representation of e has level V_4 and ownership of a for B, s and e has level V_5
then after Δt ownership of a for B, s and e will have
level $V_5 + \gamma\ [f(\omega_{41a}V_1, \omega_{42a}V_2, \omega_{43a}V_3, \omega_{44a}V_4) - V_5]\ \Delta t$.

$SR(B, a, V_1)\ \&\ SR(s, V_2)\ \&\ PS(a, V_3)\ \&\ SR(e, V_4)\ \&\ OS(B, s, a, e, V_5)$
$\twoheadrightarrow OS(B, s, a, e, V_5 + \gamma [f(\omega_{41a}V_1, \omega_{42a}V_2, \omega_{43a}V_3, \omega_{44a}V_4) - V_5]\ \Delta t)$

LP4b Generating an ownership state for B and b

If the sensory representation of B with body state b has level V_1
and the sensory representation of e has level V_2 and the preparation for body state b has level V_3
and the sensory representation of b has level V_4 and ownership of b for B and e has level V_5
then after Δt ownership of b for B and e will have
level $V_5 + \gamma\ [f(\omega_{41b}V_1, \omega_{42b}V_2, \omega_{43b}V_3, \omega_{44b}V_4) - V_5]\ \Delta t$.

$SR(B, b, V_1)\ \&\ SR(e, V_2)\ \&\ PS(b, V_3)\ \&\ SR(b, V_4)\ \&\ OS(B, e, b, V_5)$
$\twoheadrightarrow OS(B, e, b, V_5 + \gamma [f(\omega_{41b}V_1, \omega_{42b}V_2, \omega_{43b}V_3, \omega_{44b}V_4) - V_5]\ \Delta t)$

Note that in case that B is the agent self, the first condition in LP4a and LP4b indicates how far the agent has a certain willingness to come to an action or expression. For example, when no other agent is present the willingness to explicitly express emotions may be less, or when the agent is in a passive mood, willingness to come to an action a may be low. The use of ownership states in control of execution is modelled by LP5:

LP5a Action a execution

If ownership of a for B and s and e has level V_1 and preparation for action a has level V_2
and the action execution state for a has level V_3
then after Δt the action execution state for a will have level $V_3 + \gamma\ [f(\omega_{51a}V_1, \omega_{52a}V_2) - V_3]\ \Delta t$.

$OS(B, s, a, e, V_1)\ \&\ PS(a, V_2)\ \&\ ES(a, V_3) \twoheadrightarrow ES(a, V_3 + \gamma [f(\omega_{51a}V_1, \omega_{52a}V_2) - V_3]\ \Delta t)$

LP5b Body change b execution

If ownership of b for B and e has level V_1 and preparation for body state b has level V_2
and the execution state for b has level V_3
then after Δt the execution state for b will have level $V_3 + \gamma\ [f(\omega_{51b}V_1, \omega_{52b}V_2) - V_3]\ \Delta t$.

$OS(B, e, b, V_1)\ \&\ PS(b, V_2)\ \&\ ES(b, V_3) \twoheadrightarrow ES(b, V_3 + \gamma [f(\omega_{51b}V_1, \omega_{52b}V_2) - V_3]\ \Delta t)$

Note that these executions also function as the *nonverbal part of the empathic response*; e.g., showing a face expression with the same emotion as the other person.

Property LP6 describes in a straightforward manner how execution of action a or body change b affects the world state for effect e or body state b.

LP6e From action execution to effect state

If the execution state for action a has level V_1 and world state e has level V_2
then after Δt world state e will have level $V_2 + \gamma\ [f(\omega_{6e}V_1) - V_2]\ \Delta t$.

$ES(a, V_1)\ \&\ WS(e, V_2) \twoheadrightarrow WS(e, V_2 + \gamma [f(\omega_{6e}V_1) - V_2]\ \Delta t)$

LP6b From body change execution to body state
If the execution state for body state b has level V_1 and body state b has level V_2
then after Δt body state b will have level $V_2 + \gamma\ [f(\omega_{6b}V_1) - V_2\ ]\ \Delta t$.
ES(a, V_1) & WS(b, V_2) $\twoheadrightarrow$ WS(b, $V_2 + \gamma\ [\ f(\omega_{6b}V_1) - V_2\ ]\ \Delta t$)

The following property models how sensor states are updated. It applies to an action a of agent B, a feeling b of agent B, a stimulus s, effect e, or emotion indicated by body state b (covered by variable W).

LP7 Generating a sensor state for a world or body state W
If world state W has level V_1 and the sensor state for W has level V_2
then after Δt the sensor state for W will have level $V_2 + \gamma[\ f(\omega_{7W}V_1) - V_2]\ \Delta t$.
WS(W, V_1) & SS(W, V_2) $\twoheadrightarrow$ SS(W, $V_2 + \gamma\ [\ f(\omega_{7W}V_1) - V_2\ ]\ \Delta t$)

Communication of ownership of the other agent to the other agent represents acknowledgement of an agent that it has noticed the state of the other agent: a *verbal part* of the *empathic response*. These communications depend on the ownership states as specified in LP8.

LP8a Communication of the other agent B's intention a and e for s
If the ownership state of a and e for B and s has level V_1,
and communication of a and e for B and s has level V_2
then after Δt communication of a and e for B and s will have level $V_2 + \gamma\ [\ f(\omega_{8a}V_1) - V_2\ ]\ \Delta t$.
OS(B, s, a, e, V_1) & EO(B, s, a, V_2) $\twoheadrightarrow$ EO(B, s, a, e, $V_2 + \gamma\ [\ f(\omega_{8a}V_1) - V_2\ ]\ \Delta t$)

LP8b Communication of the other agent B's emotion b for e
If the ownership state of b for B and e has level V_1,
and communication of b for B and e has level V_2
then after Δt communication of b for B and e will have level $V_2 + \gamma\ [\ f(\omega_{8b}V_1) - V_2\ ]\ \Delta t$.
OS(B, e, b, V_1) & EO(B, e, b, V_2) $\twoheadrightarrow$ EO(B, e, b, $V_2 + \gamma\ [\ f(\omega_{8b}V_1) - V_2\ ]\ \Delta t$)

4 Simulation Results

In this section simulation results are discussed for scenarios that have been explored. Note that in this section for the sake of simplicity two agents A and B are considered and for each of s, a, e, b, just one instance is used, which is the same for both agents. In the first two scenarios mutual empathic understanding and convergence to a joint decision are achieved (for two different situations), and in the third scenario mutual empathic understanding is achieved but no convergence to a joint decision. In the scenarios discussed all connection strengths were taken *1*, except the inhibiting connections, which were taken *-0.2*, and the connection to the action effect in the world which was taken *0* as the focus here is on the process of decision making prior to the actual execution of the decision. The speed factor γ was set to *0.5* and $\Delta t = 0.2$. In the scenario shown in Fig. 2 both agents get stimulus s as input with level *1*. Here time is on the horizontal axis and activation levels as indicated are on the vertical axis. The upper graph shows agent A and the lower graph agent B. The threshold and steepness values used (for both agents) are shown in Table 3.

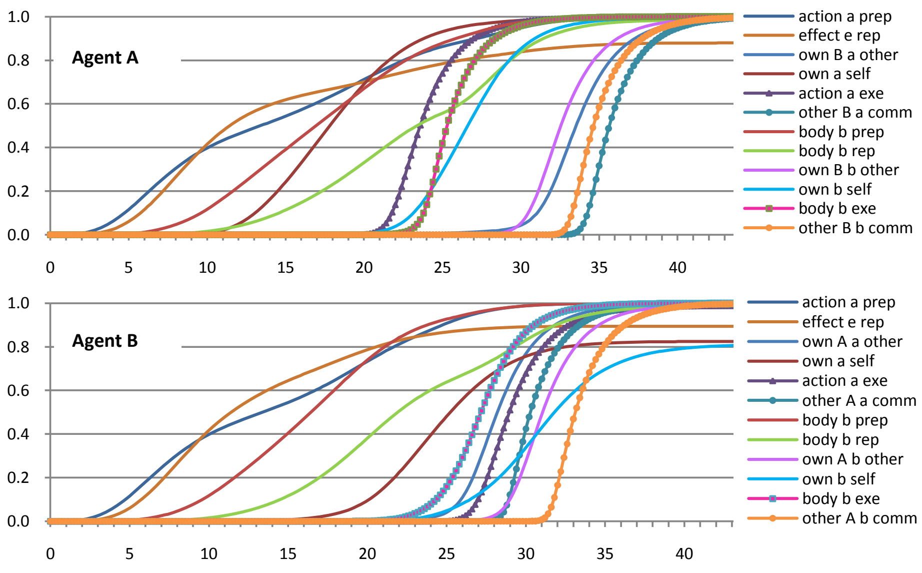


Fig. 2. Reaching a joint decision and mutual understanding for different self-contexts

Table 3. Threshold and steepness values used

LP	LP2e	LP2b	LP3a	LP3b	LP4a	LP4b	LP5a	LP5b	LP8a	LP8b
threshold τ	*0.2*	*0.7*	*1*	*0.7*	*3.2*	*3.2*	*1.6*	*1*	*0.6*	*0.6*
steepness σ	*4*	*4*	*4*	*4*	*8*	*8*	*20*	*20*	*20*	*20*

The only difference between the two agents is that agent A has level *1* context factor which indicates willingness to come to action and for agent B this is *0.5*. In Fig. 2 the following is shown:

- From time point 3 on, triggered by the stimulus s, both agents develop a *preparation for action option* a, which is immediately followed by activation of *predicted effect* e.
- Around time point 6 both agents start to develop an *emotional response preparation* for b triggered by the predicted effect e.
- As a consequence (by the as-if body loop), for both the *feeling* of this emotion starts from time point 9 on.
- Around time point 10 agent A starts to activate the *self ownership* state for *action* option a, whereas for agent B this only happens later, after time point 16, due to its lower self-context value.
- Due to this, agent A *expresses* (the tendency for) *action option* a from time point 20 on (marked line).
- Around time 21 agent A starts to develop a *self-ownership* state for *emotion* b
- From time point 22 on agent A *expresses* the *emotion felt* (marked line).

Note that at this point in time agent B does not yet show such reactions, due to the lower self-context for agent B. However, by B's mirroring of the two types of expression (action a tendency and body state b) from agent A, agent B is affected in its preparation levels for both the action a option and the bodily response b.

- Around time 16 agent B has started to develop a *self-ownership* state for the *action* a.
- From time point 21 agent B also starts to develop a *self-ownership* state for *feeling* b and *expresses* the *feeling* of b (marked line).
- From time 22 on agent B develops an *ownership* state for *agent* A *of action* a, and from time 24 on agent B develops an *ownership* state for *agent* A *feeling* b.
- The *expression* of a tendency for *action* option a is developed by agent B from time point 26 on (marked line).

This actually creates a joint decision for action option a, accompanied by a shared good feeling b for it. Moreover, this also provides the nonverbal part of B's empathic response on agent A's action tendency and feeling.

- After time 27 agent B starts to develop an *ownership* state for *agent* A *feeling* b.
- Agent B shows a *verbal empathic response* to A for both the *action* and the *feeling* starting at time points 28 and 31, respectively (marked lines).

The verbal empathic response from agent A to B comes later, which reflects the fact that some time was needed to get agent B in the proper state (due to mirroring) to show support for action option a and feeling b:

- Agent A develops *ownership* states for *agent* B of *action* a and B *feeling* b starting at time 28 and 29, respectively.
- At time points 33 and 34 (marked lines, upper graph), agent A shows a *verbal empathic response* to B for both the *action* and the *feeling,* respectively.

This shows how the process to reach a joint decision can b based on different processes within each of the agents, and their mutual impact on each other.

A second scenario addressed a case in which agent B and A both have self-context level *1*, and A has stimulus level *1*, and agent B *0.5*. Also in this case after some time a joint decision comes out, but now agent B depends on agent A for its activation of preparation for action option a and the associated emotional response and feeling. Therefore during the period from time point 5 to time point 25 the activation levels of action preparation, effect prediction, emotional response and feeling stay low. After time point 25 they move up due to agent A's expression starting at time 20 and 21.

A third scenario addressed a case in which agent A has self-context level *1*, but for agent B this level is *0*. The stimulus s for both has level *1*. In this case no joint decision comes out, as agent B does not follow A in the action option a, but still empathic responses are shown. As in the scenario in Fig. 2 agent B develops expressed states for the action a and feeling b, from time point 20 on. Also agent B shows the same pattern as in Fig. 2, up to time point 20. However, then a main difference is that in this scenario the self ownership state of B for action a does not develop; it gets a level not much more than *0.1*. As a consequence no tendency for action a is developed. Note that due to the emotion contagion still the feeling level of agent B becomes higher and as a result this feeling is expressed (from time point 22 on), thus contributing a nonverbal empathic response. Moreover, also verbal empathic responses of agent B are developed (after time points 27 and 30, respectively).

5 Discussion

In this paper a social agent model was presented based on mechanisms from Social Neuroscience. The model addresses the emergence of joint decisions, accompanied by shared emotions and mutually acknowledged empathic understanding. To this end it covers both cognitive and affective processes and their interaction in decision making, and social contagion. Core mechanisms adopted are mirror neurons (e.g., [23, 32, 35, 39]), internal simulation (e.g., [8, 10, 16, 17, 20]), and emotion-related valuing of predicted effects of action options (e.g., [1, 8, 9, 11, 29, 31]). It was shown how such social agent models can be used to perform simulation and analysis of the emergence of joint decisions grounded in shared emotion-related valuing, and together with mutual empathic understanding of agents.

The social agent model uses elements from the model presented in [37] for the empathic understanding, but in contrast to [37] where the empathic understanding was limited to emotions, in the current model it is applied to both (tendencies for) actions and emotions. Furthermore, the current model uses the idea of ownership states as in the model presented in [38]. However, in [38] ownership states are differentiated into prior and retrospective ownership states, which was not done in the current model. Moreover, in the current model the ownership states were used both for actions and for expressing emotions, whereas in [38] they were only focused on actions, and emotions were not addressed. Another difference to both [37] and [38] is the use in the current model of social contagion to affect both action tendencies and associated feelings in order to come to joint decisions accompanied by shared associated emotions. This purpose was also addressed at an abstract level in [21] and [22], but the models in these references do not address the underlying internal neurological mechanisms within the agents, and the mutually acknowledged empathic understanding as addressed in the model presented in the current paper.

Beliefs and explicit information exchange by means of verbal communication was left out of consideration in the presented model. In an extension this can be added and integrated, for example, in manner similar to what is described in [21] or [22].

References

1. Bechara, A., Damasio, H., Damasio, A.R.: Role of the Amygdala in Decision-Making. Ann. N.Y. Acad. Sci. 985, 356–369 (2003)
2. Becker, W., Fuchs, A.F.: Prediction in the Oculomotor System: Smooth Pursuit During Transient Disappearance of a Visual Target. Experimental Brain Res. 57, 562–575 (1985)
3. Blakemore, S.-J., Frith, C.D., Wolpert, D.M.: Spatio-Temporal Prediction Modulates the Perception of Self-Produced Stimuli. J. of Cognitive Neuroscience 11, 551–559 (1999)
4. Blakemore, S.-J., Wolpert, D.M., Frith, C.D.: Why can't you tickle yourself? Neuroreport 11, 11–16 (2000)
5. Bosse, T., Jonker, C.M., van der Meij, L., Treur, J.: A Language and Environment for Analysis of Dynamics by Simulation. Intern. J. of AI Tools 16, 435–464 (2007)
6. Brass, M., Spengler, S.: The Inhibition of Imitative Behaviour and Attribution of Mental States. In: Striano, T., Reid, V. (eds.) Social Cognition: Development, Neuroscience, and Autism, pp. 52–66. Wiley-Blackwell (2009)
7. Cacioppo, J.T., Berntson, G.G.: Social neuroscience. Psychology Press (2005)

8. Damasio, A.R.: Descartes' Error: Emotion, Reason and the Human Brain. Papermac, London (1994)
9. Damasio, A.R.: The Somatic Marker Hypothesis and the Possible Functions of the Prefrontal Cortex. Philosophical Transactions of the Royal Society: Biological Sciences 351, 1413–1420 (1996)
10. Damasio, A.R.: The Feeling of What Happens. In: Body and Emotion in the Making of Consciousness. Harcourt Brace, New York (1999)
11. Damasio, A.R.: Looking for Spinoza: Joy, Sorrow, and the Feeling Brain. Vintage books, London (2003)
12. Damasio, A.R.: Self comes to mind: constructing the conscious brain. Pantheon Books, NY (2010)
13. Decety, J., Cacioppo, J.T. (eds.): Handbook of Social Neuroscience. Oxford University Press (2010)
14. Fried, I., Mukamel, R., Kreiman, G.: Internally Generated Preactivation of Single Neurons in Human Medial Frontal Cortex Predicts Volition. Neuron 69, 548–562 (2011)
15. Gallese, V., Fadiga, L., Fogassi, L., Rizzolatti, G.: Action Recognition in the Premotor Cortex. Brain 119, 593–609 (1996)
16. Gallese, V., Goldman, A.: Mirror neurons and the simulation theory of mindreading. Trends in Cognitive Sciences 2, 493–501 (1998)
17. Goldman, A.I.: Simulating Minds: The Philosophy, Psychology, and Neuroscience of Mindreading. Oxford Univ. Press, New York (2006)
18. Harmon-Jones, E., Winkielman, P. (eds.): Social neuroscience: Integrating biological and psychological explanations of social behavior. Guilford, New York (2007)
19. Hendriks, M., Treur, J.: Modeling Super Mirroring Functionality in Action Execution, Imagination, Mirroring, and Imitation. In: Pan, J.-S., Chen, S.-M., Nguyen, N.T. (eds.) ICCCI 2010, Part I. LNCS, vol. 6421, pp. 330–342. Springer, Heidelberg (2010)
20. Hesslow, G.: Conscious thought as simulation of behaviour and perception. Trends Cogn. Sci. 6, 242–247 (2002)
21. Hoogendoorn, M., Treur, J., van der Wal, C.N., van Wissen, A.: Agent-Based Modelling of the Emergence of Collective States Based on Contagion of Individual States in Groups. Transactions on Computational Collective Intelligence 3, 152–179 (2011)
22. Hoogendoorn, M., Treur, J., van der Wal, C.N., van Wissen, A.: Modelling the Interplay of Emotions, Beliefs and Intentions within Collective Decision Making Based on Insights from Social Neuroscience. In: Wong, K.W., Mendis, B.S.U., Bouzerdoum, A. (eds.) ICONIP 2010, Part I. LNCS, vol. 6443, pp. 196–206. Springer, Heidelberg (2010)
23. Iacoboni, M.: Mirroring People: the New Science of How We Connect with Others. Farrar, Straus & Giroux, New York (2008)
24. Iacoboni, M.: Mesial frontal cortex and super mirror neurons. Behavioral and Brain Sciences 31, 30 (2008)
25. Iacoboni, M., Molnar-Szakacs, I., Gallese, V., Buccino, G., Mazziotta, J.C., Rizzolatti, G.: Grasping the intentions of others with one's own mirror neuron system. PLoS Biology 3, e79 (2005)
26. James, W.: What is an emotion. Mind 9, 188–205 (1884)
27. Keysers, C., Gazzola, V.: Social Neuroscience: Mirror Neurons Recorded in Humans. Current Biology 20, 253–254 (2010)
28. Moore, J., Haggard, P.: Awareness of action: Inference and prediction. Consciousness and Cognition 17, 136–144 (2008)
29. Morrison, S.E., Salzman, C.D.: Re-valuing the amygdala. Current Opinion in Neurobiology 20, 221–230 (2010)

30. Mukamel, R., Ekstrom, A.D., Kaplan, J., Iacoboni, M., Fried, I.: Single-Neuron Responses in Humans during Execution and Observation of Actions. Current Biology 20, 750–756 (2010)
31. Murray, E.A.: The amygdala, reward and emotion. Trends Cogn. Sci. 11, 489–497 (2007)
32. Pineda, J.A. (ed.): Mirror Neuron Systems: the Role of Mirroring Processes in Social Cognition. Humana Press Inc. (2009)
33. Preston, S.D., de Waal, F.B.M.: Empathy: its ultimate and proximate bases. Behav. Brain Sci. 25, 1–72 (2002)
34. Rizzolatti, G., Fadiga, L., Gallese, V., Fogassi, L.: Premotor Cortex and the Recognition of Motor Actions. Cognitive Brain Research 3, 131–141 (1996)
35. Rizzolatti, G., Sinigaglia, C.: Mirrors in the Brain: How Our Minds Share Actions and Emotions. Oxford University Press (2008)
36. Singer, T., Leiberg, S.: Sharing the Emotions of Others: The Neural Bases of Empathy. In: Gazzaniga, M.S. (ed.) The Cognitive Neurosciences, 4th edn., pp. 973–986. MIT Press (2009)
37. Treur, J.: A Cognitive Agent Model Displaying and Regulating Different Social Response Patterns. In: Walsh, T. (ed.) Proc. IJCAI 2011, pp. 1735–1742 (2011)
38. Treur, J.: A Cognitive Agent Model Incorporating Prior and Retrospective Ownership States for Actions. In: Walsh, T. (ed.) Proc. IJCAI 2011, pp. 1743–1749 (2011)
39. Treur, J.: From Mirroring to the Emergence of Shared Understanding and Collective Power. In: Jędrzejowicz, P., Nguyen, N.T., Hoang, K. (eds.) ICCCI 2011, Part I. LNCS, vol. 6922, pp. 1–16. Springer, Heidelberg (2011)

Negowiki: A Set of Community Tools for the Consistent Comparison of Negotiation Approaches

Ivan Marsa-Maestre[1], Mark Klein[2], Enrique de la Hoz[1], and Miguel A. Lopez-Carmona[1]

[1] Computer Engineering Department, University of Alcala, Spain
[2] Center for Collective Intelligence, Massachusetts Institute of Technology, USA

Abstract. There is a number of recent research lines addressing automated complex negotiations. Most of them focus on overcoming the problems imposed by the complexity of negotiation scenarios which are computationally intractable, be it by approximating these complex scenarios with simpler ones, or by developing heuristic mechanisms to explore more efficiently the solution space. The problem with these mechanisms is that their evaluation is usually restricted to very specific negotiation scenarios, which makes very difficult to compare different approaches, to re-use concepts from previous mechanisms to create new ones or to generalize mechanisms to other scenarios. This makes the different research lines in automated negotiation to progress in an isolated manner. A solution to this recurring problem might be to create a collection of negotiation scenarios which may be used to benchmark different negotiation approaches. This paper aims to fill this gap by providing a framework for the characterization and generation of negotiation scenarios intended to address this problem. The framework has been integrated in a website, called the *Negowiki*, which allows to share scenarios and experiment results with the negotiation community, facilitating in this way that researchers compare and share their advancements.

1 Introduction

Automated negotiation provides an important mechanism to reach agreements among distributed decision makers [10,11]. It has been extensively studied from the perspective of e-commerce, though it can be seen from a more general perspective as a paradigm to solve coordination and cooperation problems in complex systems, providing a mechanism for autonomous agents to reach agreements on, e.g., task allocation, resource sharing, or surplus division.

A variety of negotiation models has been proposed, yielding promising results in a wide range of negotiation problems [3]. However, most of these approaches are evaluated for negotiation scenarios meeting very specific requirements. Given the vast variety of negotiation problems, a recurrent challenge automated negotiation researchers have to face is how to justify the models and mechanisms they propose are suitable to solve or model different problems, or how to compare their approaches and methods with the ones of other researchers. In the best cases, there are a few number of previous works similar enough to the new proposal to make a comparison. In most cases, however, this comparison is not possible due to the diversity of scenarios the different research groups deal with, so the different research lines progress in an isolated manner. In addition, though there exist multiple surveys about negotiation in the literature [8,1], they are more intended to classify the different approaches (mediated, non-mediated, one-shot, iterative...) than to describe or classify the different negotiation problems.

We are developing a web-based system, called *Negowiki*, to help address this important gap, providing a framework which may be used by the research community to 1) create a openly accessible repository that systematically covers the space of potential negotiation scenarios and 2) test negotiation mechanisms against these scenarios. The need to have negotiation scenario testbeds to provide reproductivity and coherence to works from different authors has been acknowledged before [6]. Our goal in this line is to provide a framework which allows to characterize and generate negotiation scenarios according to high-level properties. The benefit of using such a framework would be threefold. First, it will facilitate the development of successively better negotiation mechanisms by providing a clear way to compare their performance. Second, it will ensure that mechanisms are tested on the full space of important problem types. Finally, this would open the door to the rigorous assessment of the applicability of negotiation approaches to real-world problems. For a given real negotiation problem, we could measure the high-level properties of the scenario and use them to find in the database the negotiation approach which performs better for scenarios matching these properties.

Developing such a tool requires that we be are able to meet three key challenges: characterizing the key properties of negotiation scenarios, measuring these properties in existing scenarios, and being able to generate new scenarios which have given properties. This paper describes how we are meeting these challenges, as well as how our emerging solutions are being made available to the research community through the *Negowiki* web site. The paper is organized as follows:

– We describe a set of tools which allow to measure high-level properties of a negotiation scenario. This includes both structural properties of the agents' utility functions and properties derived from the relationship between the different utility functions (Section 2).
– We describe a negotiation scenario generator which considers the properties outlined above (Section 3).
– We describe a community website, called the *Negowiki*, which makes these tools available to the negotiation research community and allows them to upload their negotiation scenarios to have them characterized, to download other researchers' scenarios for comparison with their approaches and to upload experiment results to share them with the community (Section 4).

2 Characterizing Negotiation Scenarios

The first key component of the *Negowiki* vision is to be able to characterize negotiation scenarios using high-level properties. To do this, we should first define what we understand as negotiation scenarios. Different authors agree that there are three main constituents in a negotiation model [8]: an interaction protocol which defines the rules of encounter among the negotiating agents, a set of decision mechanisms and strategies which govern agents' decision making, and the preference sets of the different agents which allow them to assess the different solutions in terms of gain or utility and to compare them. From these three components, we can easily see that both the interaction protocol and the decision mechanisms and strategies are more related to the way the model solves the negotiation problem than to the negotiation problem itself. Therefore, in the following we characterize a negotiation scenario according to the preferences of the agents taking part in the negotiation.

In particular, it is usual to define agent preferences by means of utility functions. Formally, for a given multi-attribute domain D, the *utility function* for each agent j is defined as

$$U^j : D \rightarrow \mathbb{R},$$

assigning to each possible combination of values in D or *deal* $s = \{s_i | i = 1, ..., n; s_i \in d_i\}$ a real number, which represents the utility that deal s yields for agent j.

There are vastly different utility functions in the negotiation literature. Monotonic negotiation scenarios are usually modeled with *Constant Elasticity of Substitution* (CES) utility functions [15], which are widely used in economics as production functions, and in consumer theory as utility functions. To represent non-monotonic utility spaces, we can use for instance k-additive utility functions [5]. Another widely used way to represent preferences and utility functions is the use of constraints over the values of the attributes. A particular case of constraint-based utility representation which has been used to model complex utility spaces for negotiation are *weighted constraints* [7]. This kind of utility functions produces nonlinear utility spaces, with high points where many constraints are satisfied, and lower regions where few or no constraints are satisfied. Due to the hypercube shape of the constraints, the utility functions defined in this way are discontinuous. An example of a utility representation for continuous, non-monotonic utility spaces can be found in [13], where the authors model the utility space of an agent as a sum of *bell-shaped* functions.

The above is just a brief review of a selection of the different kinds of preference representations used in the most relevant works in the field. From the formulations and descriptions we can see the inherent difficulty for the direct comparison of approaches which are intended to work in different kinds of scenarios, and for determining which of the existing approaches would be most effective to address a new scenario. We could wonder, for instance, whether the protocols proven successful for constraint-based utility spaces could be applied, for instance, to bell-based negotiation scenarios, or whether protocols intended to work with bell utility functions could be applied, with some modifications, to CES-based utility spaces. However, direct comparison of the approaches is often very difficult (if not unfeasible) due to the important differences between the scenarios. Even if a given negotiation mechanism could be applied to two different negotiation scenarios, it is very difficult to establish equivalencies between them, due to the vast differences between the settings of the different scenarios. For instance, [7] describes experiments for a negotiation scenario involving constraint-based utility spaces, where there are constraints with widths drawn uniformly from the interval $[3, 7]$ (in a domain $[0, 9]$). If we move onto a bell-based utility space, now we need to define it in terms of bell radii and heights. Which values would yield a utility function of similar complexity? We believe that such comparison of approaches could be made possible if there existed a framework for the characterization of negotiation scenarios according to a set of common properties. This is what we aim to provide in the following section.

2.1 High-Level Properties of Negotiation Scenarios

We have defined a negotiation scenario as a set of agent utility functions. Therefore, to characterize a given scenario we have to look at these utility functions and the relationships between them. The most immediate approach is to study the structural properties of a fitness landscape which are interesting regarding search complexity within the space, such as modality, ruggedness, smoothness and neutrality [17]. Most of the approaches we can find in the literature are based on the correlation between different samples of the fitness function f. A metric which is easy to compute in most scenarios and allows to make quantitative evaluations about the complexity of a fitness or utility landscape is *correlation length* or *correlation distance*. Correlation distance is defined as the minimum distance ψ_{th} which makes correlation fall below a given threshold th (usually 0.5), which gives an idea of the distance we can move throughout the solution space while keeping a certain correlation between samples [14]. The greater the distance, the smoother the utility function and the easier, in general, it will be to optimize.

A property which is usually related to negotiation complexity is issue interdependency. Negotiations with multiple, interdependent issues are assumed to be harder than those involving independent issues [9]. There are some recent works [16,4] suggesting to assess the degree of interdependency between issues in order to modify the negotiation strategy accordingly (e.g. by negotiating separately those issues which are less interdependent). However, in most cases issue interdependency is measured in an ad-hoc manner, usually restricted to the kind of utility representation used. Here we propose a measure based on information theory, analogous to the epistasis measure described in [18] for evolutionary computation:

$$\epsilon_{ij} = \begin{cases} \frac{I(i;U)+I(j;U)}{I(i,j;U)} - 1 & \text{if } I(i,j;U) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

where $I(i;U)$ and $I(j;U)$ are the amount of information that each issue i and j reveals about the value of the utility function U, and $I(i,j;U)$ is the amount of information *both* issues reveal about the value of U.

Another aspect which may be studied is the point distribution in utility space diagrams. These diagrams represent the utilities achieved by the different agents participating in the negotiation for each analyzed solution, allowing for instance to assess the distance of a solution from the Pareto front, which is used in many works as an evaluation metric for negotiation mechanisms. Intuitively, a negotiation will be easier as the ratio of mutable acceptable solutions against total potential solution in high, and will be more difficult in the opposite case. The same is true with negotiation efficiency and the ratio of solutions near the Pareto front. Finally, the very shape of the Pareto front may affect significantly the properties of the negotiation scenario, or even the way to analyze it. For the purpose of assessing the complexity of a given scenario, however, it is not enough to know if there exist solutions which yield a given set of utilities to the different agents. With the same utility diagram, a scenario where an 80% of the solutions are both mutually acceptable and are in the Pareto front would probably be much less challenging for a negotiation mechanisms than one with only a 10% of Pareto-efficient, mutually acceptable solutions. Therefore, we have to consider also the number (or ratio) of solutions corresponding to each point in the utility diagram. Taking this into account, we extend the concept of utility diagrams to utility *histograms* $H(\bar{u})$, where $\bar{u}$ is a vector of utility values for the different agents, and the histogram value at $\bar{u}$ represents the number of potential solutions which yield that combination of utility values for the agents. From these utility histogram we can easily derive properties like the ratio of mutually accepted solutions and the ratio of Pareto-efficient solutions.

3 A Scenario Generator for Complex Automated Negotiations

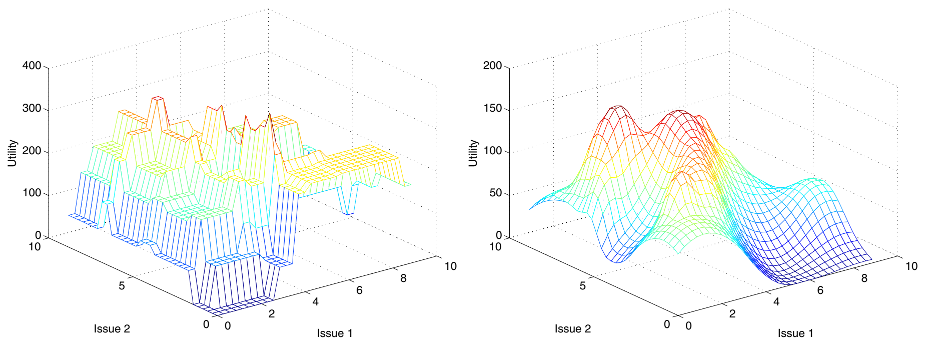
The second key component of the *Negowiki* vision is to be able to generate scenarios with specified high-level properties, i.e. that take into account both the structural properties of the agent utility functions and the relationships between the utility functions of the different agents. In the following, we first describe a parametric mechanism to generate utility functions, and then an approach to control the relationship between the utility functions through the utility diagram of the scenario.

3.1 Generation of Utility Functions by means of Hypervolumes

We aim to build a generator able to create utility functions which allow to test most of the negotiation approaches we can find so far. This is a rather ambitious goal, since, as we have seen, there are many different types of utility functions used in the negotiation literature. For the purposes of this work, we will assume cardinal utility functions, where contracts are mapped to real numbers which correspond to the utility values they yield. Note that cardinal utility functions may be used to represent ordinal preferences too, by restricting the range of the utility values to natural numbers.

Under this assumption, we can get a fully expressive representation of utility functions by aggregating *hypervolumes*. We define a hypervolume as a constrained cardinal function, where by constrained we mean that there may be constraints regarding when this cardinal function contributes to the utility value of the overall utility function. For instance, we can have a cardinal function $C_1(\bar{x}) = 5$, which is constrained so that it only applies for $\bar{x} \in S_1$, where S_1 is a given subset or region of the solution space S. In this way, we can use hypervolume C_1 as a weighted constraint. In a similar way, if we wanted to generate a linear utility function, we could use a hypervolume defined as a hyperplane, constrained so that it covers all the domain. Our scenario generator currently supports constant, cone, bell and CES cardinal functions as hypervolumes, though it has been designed so that new categories of hypervolumes may be added if needed.

Apart from adding hypervolumes to the utility function, we need to be able to define the aggregation operators we use to compute the overall agent utility from the hypervolumes. The generator covers a wide range of simple aggregation operators, like weighted sum (and average), maximum or minimum. Figure 1 shows two examples of utility functions generated using different kinds of hypervolumes and a weighted sum aggregation.



(a) hypercubes **(b)** bell hypervolumes

Fig. 1. Generation of utility spaces with weighted aggregation of hypervolumes

Finally, hypervolumes are defined to depend on a set of parameters (e.g. width, height, aspect ratio...), so that they can be varied to control the properties of the resulting utility functions. The *Negowiki* utility function generator allows to have complete control over all the parameters, though sometimes it is preferred to specify more wide-sense requirements for the generated utility functions. In order to do this, we provide sample templates which receive a set of higher-level parameters and generate utility functions from them. These templates allow us, for example, to specify such higher-level properties as the correlation length of the utility space, automatically selecting the lower-level properties needed to achieve this.

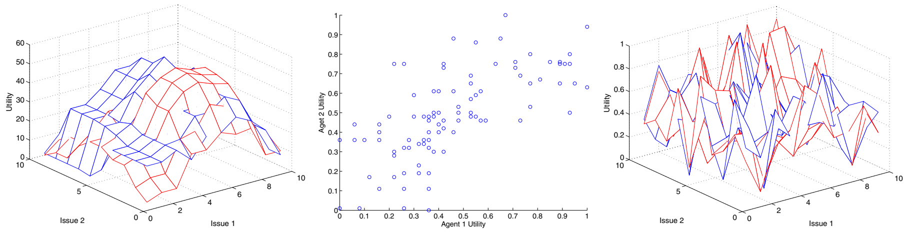
The generation of utility functions by means of aggregation of hypervolumes provides a flexible and expressive way to model different kind of agent preferences, and allows to control to a great extent the complexity of finding high utility regions within the utility space of an agents (by controlling, for instance, correlation length). The complexity of the individual agent utility functions, however, does not fully account for the complexity of the scenario. We may have, for instance, scenarios where agents may find very difficult to determine their high utility regions, but where once these regions have been found agreements are fairly straightforward, because high utility regions for the different agents coincide. On the other hand, we may have smooth utility functions for the agent which make very easy to locate high utility regions, but the negotiation may still be complex because *mutually acceptable* regions are hard to find. A mechanism to take into account the relationships between the utility functions of the different agents is described in the following section.

3.2 Generating Negotiation Scenarios from Utility Diagrams

As we stated above, utility diagrams are usually used to characterize negotiation scenarios, since they provide a graphical way to visualize the relationship between the potential solutions to the negotiation problem and the utility values these solutions would give to the negotiation agents. Utility diagrams are useful, for instance, to determine the existence of mutually acceptable solutions (that is, solutions with utilities above the reservation value for all agents), or to assess the relative efficiency of the solutions (that is, the distance from the solutions to the Pareto front). Finally, a wide range of notions for optimal solutions (e.g. Nash solution, Kalai-Smorodinsky, etc.) make use of the Pareto frontier.

The Negowiki scenario generator is able to take such utility diagrams as input for scenario generation, so that we are able to generate agent utility functions which match a given utility histogram. In order to generate utility functions for a given utility histogram, we propose to use *shared hypervolumes.* The idea behind shared hypervolumes is to include *similar* hypervolumes in the utility functions of the different agents, adjusting the parameters of the hypervolumes so that they generate appropriate points within the utility histogram. For instance, if we want to generate utility functions for a trivial utility histogram $H(\bar{u})$ for two agents, where the histogram value is v for $\bar{u} = \{a, b\}$ and 0 otherwise, we could achieve this by generating two utility functions which share a hypercube of volume v, with weight a for the first agent and weight b for the second. Of course, as the number of points in the utility diagram increases, the complexity of the generation process also increases, since we have to take into account the effect of the intersections between shared hypervolumes. What we do is to generate a first approximation of the utility functions by dividing the utility space in non-overlapping regions and assigning shared hypervolumes to each region, and then feed this first approximation of the utility diagram to a nonlinear optimizer which tries to minimize the approximation error.

An important property of this scenario generation strategy is that the shape and parameters of the shared hypervolumes may be varied so that additional properties of the generated functions are satisfied. For instance, we can vary the volume of the shared hypervolumes to adjust the correlation length of the utility functions. Figure 2 shows an example for two agents and weighted hypercubes, where we have generated two scenarios with identical utility diagrams and different correlation lengths. The type of hypervolumes or the aggregation operators used may be adjusted as well.



(a) $\psi_{0.7} = 5.9$ **(b)** utility diagram **(c)** $\psi_{0.7} = 1.7$

Fig. 2. Scenarios with the same utility diagram and different correlation lengths

4 The *Negowiki* Community Website

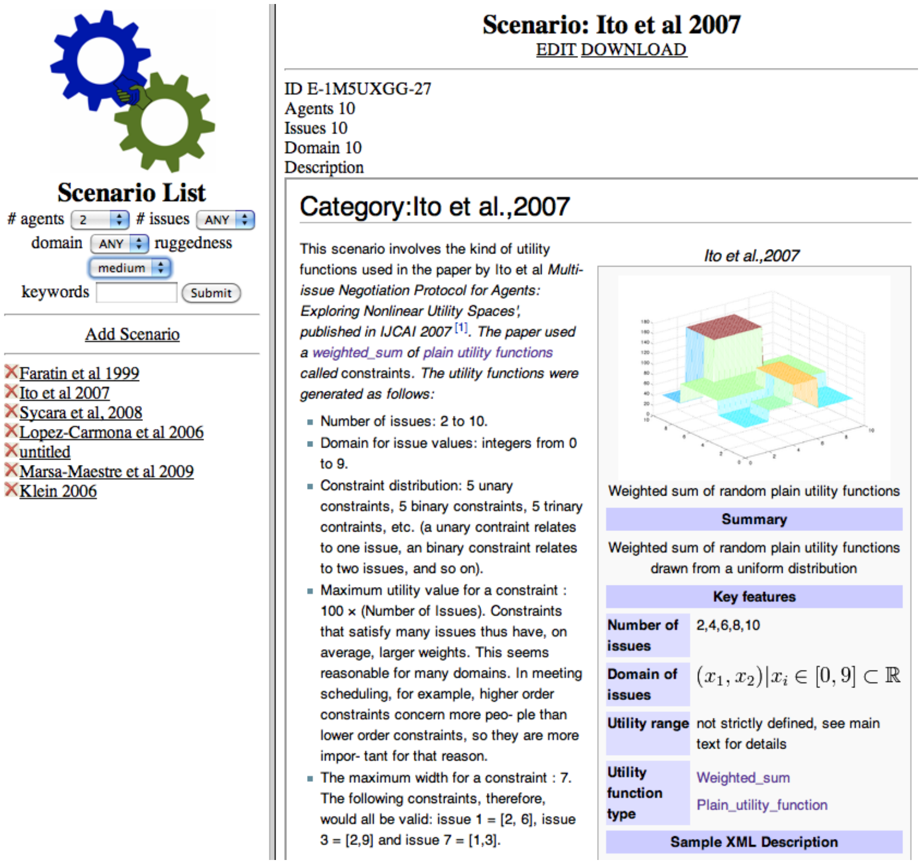
We have integrated the above tools in a community website, called the *Negowiki*. This web application helps address the research challenges we outlined in Section 1 by providing a community repository of nonlinear utility functions as well as, as time goes on, a repository of performance data for different combinations of protocols and utility functions. At the time this paper is written, the *negowiki* provides the following functionalities:

– *Download negotiation scenario.* For each negotiation scenario in the *Negowiki*, we provide an scenario description page, where users can find a document describing the rationale behind the scenario, some pictures of the associated agent utility functions (normally projections on two issues), the different metric values for both the scenario and the utility functions, and download links for the XML representations of the utility functions. Users can download simple function libraries (currently available in Matlab and Common Lisp, with other languages coming) that calculate the utility of a given contract for a given *Negowiki* scenario, making it straightforward for a research group to use the *Negowiki* scenarios in their experiments. The great advantage of this is that it allows researchers to easily test their protocols in a wide range of scenarios, without having to go to the trouble of designing and creating these scenarios themselves.
– *Upload negotiation scenario for analysis.* Users may upload their own negotiation scenarios, by submitting the XML representations of the associated

utility functions to the website. When uploaded, the website computes the appropriate scenario metrics and shows them as a result. These metrics can help researchers understand the properties of their scenarios, and therefore why their protocols work well in some circumstances but not in others. This can, in turn, help guide researchers towards developing better protocols. Finally, users have the option of archiving the scenario, which would then become part of the *negowiki* repository.

– *Upload negotiation results data for analysis.* If users perform experiments with the provided scenarios, they can upload the negotiation results data for analysis. When uploaded, the website computes a set of negotiation outcome metrics (individual agent utility, social welfare, fairness, distance from the Pareto front...) and shows them to the user. The *Negowiki* team has devoted substantial effort to ensuring that the site calculates accurate outcome metrics for negotiation experiment results. This is especially important for nonlinear scenarios, where calculating such measures as fairness and social welfare efficiency requires solving a challenging nonlinear optimization problem to estimate the Pareto front. In cases where the scenarios are generated by *Negowiki* starting from a utility histogram, we can *guarantee* that the outcome metrics are correct because the Pareto front is known up-front, rather than having to be estimated. Again, users may archive their results to make them available to the community.
– *Search for scenarios, utility functions or experiment results.* The *negowiki* is database-driven, which allows to perform advanced searches to prune the list of scenarios, utility functions or experiments. For instance, users may look for scenarios for a given number of agents and issues where utility function epistasis lies within a certain range of values. Users can also search for protocols that have performed well on scenarios they care about, so they can either identify the best protocols for solving the negotiation problem at hand or use the design of these top protocols as a inspiration for further improving their own protocols. Figure 3 show a screenshot of the results of a scenario search, and the description page of one of the scenarios found (in the right pane).

At this stage of development, *negowiki* is a beta project, and so we are offering access only upon request. Readers may contact the authors to get an account, or they can take a look at the main features of the website in the demo video at `http://negowiki.mit.edu/video`. At the time of writing this paper, we have uploaded to *Negowiki* a range of scenarios used in previous research works by different authors [2,9,7,13], and we have benchmarked the different negotiation approaches proposed in these works (e.g. similarity-based negotiation, region-based negotiation, auction-based negotiation...) in the full range of scenarios, showing that the relative and absolute performance of these protocols change depending on the kind of scenario they are applied to. These results are beyond the scope of this discussion, and will appear in a separate paper. See `http://negowiki.mit.edu/` for details on that.



Scenario List

agents 2 # issues ANY

domain ANY ruggedness medium

keywords Submit

Add Scenario

- Faratin et al 1999
- Ito et al 2007
- Sycara et al, 2008
- Lopez-Carmona et al 2006
- untitled
- Marsa-Maestre et al 2009
- Klein 2006

Scenario: Ito et al 2007

EDIT DOWNLOAD

ID E-1M5UXGG-27
Agents 10
Issues 10
Domain 10
Description

Category:Ito et al.,2007

This scenario involves the kind of utility functions used in the paper by Ito et al *Multi-issue Negotiation Protocol for Agents: Exploring Nonlinear Utility Spaces', published in IJCAI 2007* [1]. *The paper used a* weighted_sum *of* plain utility functions *called* constraints. *The utility functions were generated as follows:*

- Number of issues: 2 to 10.
- Domain for issue values: integers from 0 to 9.
- Constraint distribution: 5 unary constraints, 5 binary constraints, 5 trinary contraints, etc. (a unary contraint relates to one issue, an binary constraint relates to two issues, and so on).
- Maximum utility value for a constraint : 100 × (Number of Issues). Constraints that satisfy many issues thus have, on average, larger weights. This seems reasonable for many domains. In meeting scheduling, for example, higher order constraints concern more peo- ple than lower order constraints, so they are more impor- tant for that reason.
- The maximum width for a constraint : 7. The following constraints, therefore, would all be valid: issue 1 = [2, 6], issue 3 = [2,9] and issue 7 = [1,3].

Ito et al.,2007

Weighted sum of random plain utility functions

Summary	
Weighted sum of random plain utility functions drawn from a uniform distribution	
Key features	
Number of issues	2,4,6,8,10
Domain of issues	$(x_1, x_2)\|x_i \in [0, 9] \subset \mathbb{R}$
Utility range	not strictly defined, see main text for details
Utility function type	Weighted_sum Plain_utility_function
Sample XML Description	

Fig. 3. Scenario search results and scenario description page in *negowiki*

5 Conclusions and Future Work

One of the main problems in complex automated negotiation research is the difficulty to compare approaches from different authors, due to the vast diversity of scenarios considered by the different research groups working in this field. In this paper we present a framework for the characterization and generation of negotiation scenarios, with the aim to fill this gap. First, we provide a set of metrics to measure high-level scenario parameters, taking into account both the structural properties of the agent utility functions, and the complexity due to the relationships between the utility functions of the different agents. Then, we present a framework to generate scenarios in a parametric and reproducible way. The generator is based on the aggregation of hypervolumes to generate utility functions, and on the use of shared hypervolumes and nonlinear regression to generate negotiation scenarios from utility diagrams. Finally, we have developed a community website where generated scenarios and experiment results may be stored and searched for according to their parameters and metrics, and where users of the framework can contribute to its ongoing development both with scenarios to add to the library and with extensions to the framework code.

Though the experiments performed with the scenario generator yield satisfactory results, there is still plenty of research to be done in this area. We are interested in exploring new metrics, like smoothness or neutrality, and to refine the control of the generator over the current ones (e.g. fine-grained control of epistasis). We are interested in creating templates for the generation of the most usual scenarios in the literature, and in performing an exhaustive comparison of the most relevant related works in all those scenarios. Finally, we are exploring the possibility of using Negowiki as a complement to other benchmarking tools, such as GENIUS [12].

Acknowledgments. This work has been supported by the Spanish Ministry of Education and Science grants TIN2008-06739-C04-04 and JC-2010-035.

References

1. Buttner, R.: A classification structure for automated negotiations. In: WI-IATW 2006: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, pp. 523–530. IEEE Computer Society, Washington, DC (2006)
2. Faratin, P., Sierra, C., Jennings, N.R.: Negotiation decision functions for autonomous agents. Robotics and Autonomous Systems 24(3-4), 159–182 (1998)
3. Fatima, S., Wooldridge, M., Jennings, N.R.: An analysis of feasible solutions for multi-issue negotiation involving nonlinear utility functions. In: AAMAS 2009: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems, pp. 1041–1048. IFAAMAS, Richland, SC (2009)
4. Fujita, K., Ito, T., Klein, M.: An approach to scalable multi-issue negotiation: Decomposing the contract space based on issue interdependencies. In: 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), vol. 2, pp. 399–406 (2010)
5. Grabisch, M.: K-order additive discrete fuzzy measures and their representation. Fuzzy Sets Syst. 92(2), 167–189 (1997)
6. Hindriks, K.V., Jonker, C.M., Tykhonov, D.: A multi-agent environment for negotiation. In: El Fallah Seghrouchni, A., Dix, J., Dastani, M., Bordini, R.H. (eds.) Multi-Agent Programming, pp. 333–363. Springer, US (2009)
7. Ito, T., Klein, M., Hattori, H.: A multi-issue negotiation protocol among agents with nonlinear utility functions. Journal of Multiagent and Grid Systems 4(1), 67–83 (2008)
8. Jennings, N.R., Faratin, P., Lomuscio, A.R., Parsons, S., Sierra, C., Wooldridge, M.: Automated negotiation: Prospects, methods and challenges. International Journal of Group Decision and Negotiation 10(2), 199–215 (2001)
9. Klein, M., Faratin, P., Sayama, H., Bar-Yam, Y.: Protocols for negotiating complex contracts. IEEE Intelligent Systems 18(6), 32–38 (2003)
10. Kraus, S., Sycara, K., Evenchick, A.: Reaching agreements through argumentation: A logical model and implementation. Artificial Intelligence 1-2, 1–69 (1998)

11. Lai, G., Li, C., Sycara, K., Giampapa, J.: Literature review on multiattribute negotiations. Tech. Rep. CMU-RI-TR-04-66, Robotics Institute, Carnegie Mellon University, Pittsburgh, USA (December 2004)
12. Lin, R., Kraus, S., Tykhonov, D., Hindriks, K., Jonker, C.M.: Supporting the Design of General Automated Negotiators. In: Ito, T., Zhang, M., Robu, V., Fatima, S., Matsuo, T., Yamaki, H. (eds.) Innovations in Agent-Based Complex Automated Negotiations. SCI, vol. 319, pp. 69–87. Springer, Heidelberg (2010)
13. Lopez-Carmona, M.A., Marsa-Maestre, I., de la Hoz, E., Velasco, J.R.: A region-based multi-issue negotiation protocol for non-monotonic utility spaces. Computational Intelligence (2011)
14. Manderick, B., de Weger, M., Spiessens, P.: The genetic algorithm and the structure of the fitness landscape. In: Proceedings of the Fourth International Conference on Genetic Algorithms, San Diego, CA, pp. 1143–1150 (1991)
15. Mas-Colell, A., Whinston, M.D., Green, J.R.: Microeconomic Theory. Oxford University Press, New York (1995)
16. Robu, V., Somefun, D.J.A., La Poutré, J.A.: Modeling complex multi-issue negotiations using utility graphs. In: AAMAS 2005: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 280–287. ACM, New York (2005)
17. Vassilev, V.K., Fogarty, T.C., Miller, J.F.: Smoothness, ruggedness and neutrality of fitness landscapes: from theory to application, pp. 3–44. Springer-Verlag New York, Inc., New York (2003)
18. Ventresca, M., Ombuki-Berman, B.: Epistasis in multi-objective evolutionary recurrent neuro-controllers. In: IEEE Symposium on Artificial Life, ALIFE 2007, pp. 77–84 (2007)

A Stochastic Negotiation Approach to Power Restoration Problems in a Smart Grid

Von-Wun Soo[1,2] and Yen-Bo Peng[2]

[1] Institute of Information Systems and Applications, National Tsing Hua University, Hsinchu Taiwan
soo@cs.nthu.edu.tw
[2] Dept. of Computer Science, National Tsing Hua University, Hsinchu Taiwan
engypong@gmail.com

Abstract. In this paper we propose a negotiation protocol for multi-feeder agents who must resolve the conflicts in order to find an optimal allocation of power in order for the power restoration after a blackout in a power grid. Since the power distribution domain constraints and cost, the optimal power distribution criteria involves multi-objectives such as number of changes of switches, number of power zones restored, and etc. It is not usually easy to come up with an optimal solution in a scaled-up complicated power grid topology within a limited time. We implemented two stochastic decision functions PDF and ZDF for feeder agents to decide whether to accept a proposal of other feeder agent who requests a restoration zone and whether to request a candidate target zone to deliver the power respectively in the negotiation. We show that with ZDF the feeder agents can negotiate faster to come up with the optimal solution than without ZDF. Finally we also show that our negotiation is a real time algorithm and show the performance curve of negotiation in terms of the restoration rate.

Keywords: Stochastic negotiation, power restoration, smart grid.

1 Introduction

According to the Maximum Expected Utility (MEU) principle, a rational agent always chooses the action which has the maximum expected utility. Ideally, the MEU principle defines the proper action to take in every problem situation. Unfortunately, there are many difficulties for making decision based on the MEU to be feasible for a rational agent. First, the actual computation cost of estimating the MEU may be very expensive. Second, it is not easy to have a complete model of all problem states in terms of proper utility and probability values. Third, even if agents always choose an action that maximizes expected utility in a sequential decision-making, they may sometimes trap in a local optimal state as encountered in most cases of greedy hill-climbing search. There were researches [1][2] about how to find the expected utility efficiently and how to model the problems completely. However, even we can build a correct model and find expected utility quickly, we still have a chance to trap in a local optimal state. Negotiation is one of the most important abilities for agents to achieve coordination and resolving conflicts. Agents in negotiation usually assume agents are rational and they make decisions based on individual rationality and attempt to maximize the expected utility. However, if each negotiation agent insist in satisfying its individual rationality, it is in general very difficult to achieve global optimal solution. Sandholm [3][4] showed that without an oracle it is in general difficult to find an optimal re-allocation of tasks in terms of hill-climbing OCSM negotiation actions based on individual rationality [5]. Stochastic negotiation concept has been used in the sequential negotiation of organization choice as [6] that demonstrate the advantage of the negotiators who are willing to accept the short-term losses for better payoffs to come.

In the power restoration problem in the smart power grid, we need to find a set of feeder agents who are the most appropriate ones to be allocated for restoring the power to those load regions that demand the power in face with a power fault so that certain optimality criteria such as maximal number of power black-out regions, maximal amount of power loads are restored, minimal number of power switches changes, etc. can be achieved. Traditional power restoration problem is usually formulated in terms of a centralized combinatorial optimization problem that is given to an optimization algorithm such as genetic algorithm, integer programming, particle swarm algorithm [7][8], etc to find an optimal solution. In a scaled up problem of power restoration as in a large smart grid, the traditional centralized optimization methods becomes computational in-efficient and more and more difficult to maintain even they are theoretically feasible to find a very good solution that is close to the optimal one.

In this paper, we attempt to use a distributed optimization approach by using a multi-agent negotiation method to seek for the optimal solution to the power restoration problem. We treat the feeder agents as the negotiation agents who can negotiate to find proper power allocation to one who is responsible to the restoration of power demanded by the power load regions. Since agents are supposed to be rational, the negotiation based on the agent's individual rationality cannot guarantee to avoid trapping into a local optimal solution. Therefore, we allow the feeder agents to negotiate stochastically so that the trapping of local optimality can possibly be avoided in the negotiation. In other words, we do not always choose the action which has the maximum expected utility. Instead, we propose an approach to make decision softly and flexibly. This will imply the agent is not longer "absolute" rational. We propose a new scheme of stochastic negotiation and decision in a multi-agent system (MAS) in which lots of decisions have to be made before agents can conduct communication and negotiation. For example, in a negotiation, agents have to decide what candidate and feasible solutions to propose and whether a proposal should be accepted. They also need to decide the final agreement of a negotiation as the termination condition of the negotiation. Moreover, if there is more than one candidate agent involved in the negotiation, sometimes they even need to decide which candidate agent is most appropriate to negotiate with.

In the rest of this paper, we first present our method. Then we implement our method on power restoration problem and describe how it works. Finally, we will show some experiment results and make a simple conclusion.

2 Methods and Implementation

According to the MEU principle, a rational agent always chooses the action which has the maximum expected utility. However, our idea is that even if an action does not have the maximum expected utility, it should still have a chance to be chosen. In this way, we may avoid the bad decision caused by an incomplete model. On the other hand, we do not completely ignore the decision that has the potential high expected utility in the future, so that we could not trap in local optimal via a sequential decision-making.

The expected utility of action A_j can be formulated as below:

$$EU(A_j|E) = \sum_i P\big(Result_i\big(A_j\big)\big|Do\big(A_j\big), E\big)U(Result_i(A_j))$$

where $U(O)$ denotes the utility of the outcome O, $Result_i(A)$ denotes the i-th outcome of action A, and E denotes the agent's available evidence about the world. By the MEU principle, we can find maximum expected utility by the following formula.

$$MEU = Max\ (EU(A_j|E))$$

where the index j ranges over the different actions.

In our stochastic model, instead of calculate MEU, we can formulate a probability function F which is called Probability Expected Utility (PEU) function in terms of every $EU(A_j|E)$.

$$PEU_j = F(EU(A_1|E), EU(A_2|E), \ldots, EU(A_n|E)) \text{ subjected to } \sum_j PEU_j = 1$$

where n denotes the number of executable actions. In other words, PEU attaches every executable action a certain probability to be chosen to execute. Therefore, every action has a chance to be executed according to the expected utilities of all actions.

2.1 Domain Problem Formulation

A power restoration problem can be described as: After a blackout, on the premise that all power distribution constraints are met, a combination of the minimal number of changes of switch states is to be found to restore as many power load zones as possible. Simplifying the problem, we only keep two main constraints in this implementation.

1. Every feeder has limited power to be generated and to distribute.

   A feeder will decrease its power as it restores the power of a particular zone by the amount of the power demand of the zone. Therefore, it can only restore limited zones according to its own total power capability.
2. Radial power distribution structure has to be maintained.

   The zones can become connected or disconnected via switches that can be turned on and off respectively so that a proper feeder power can be delivered to the zones that demand the power. However, a feeder cannot restore an arbitrary zone. It has to restore zones connected one by one along the power grid topology. Moreover, a feeder can only restore the zones that do not break the tree structure. Besides, every zone can only be restored by one feeder. This means that a feeder may be blocked by other feeders in the restoration procedure because its reachable zones are all restored by other feeders.

Power restoration problem is in nature an NP Complete problem. Our goal is to build a distributed multi-agent systems (MAS) in which every agent is assumed to only have some local information to resolve the conflicts of the power restoration problem and achieve an solution as close to the optimal one as possible. However, by communicating with each other, they can hopefully find the optimal solution or an approximate solution to the optimal one within a time limit.

Figure 1 shows how restoration works in real situation. The black dots indicate the switches are ON (connected) while white dots indicate the switches are OFF (disconnected). Originally, power demand of all zones in the gird is supported by *feeder 0* as in Figure 1 (a). A fault happened and broke the power deliver system as in Figure 1 (b). Therefore, the whole region is blackout. By changing the states of switches, feeders can restore the blackout zones. After the restoration, the four feeders restore some zones in the grid respectively as an example shown in Figure 1(c). Our aim to implement a generic method to show how feeder agents could negotiate to find an optimal redistribution of powers from such a blackout situation of (b) to the restoration situation of (c).

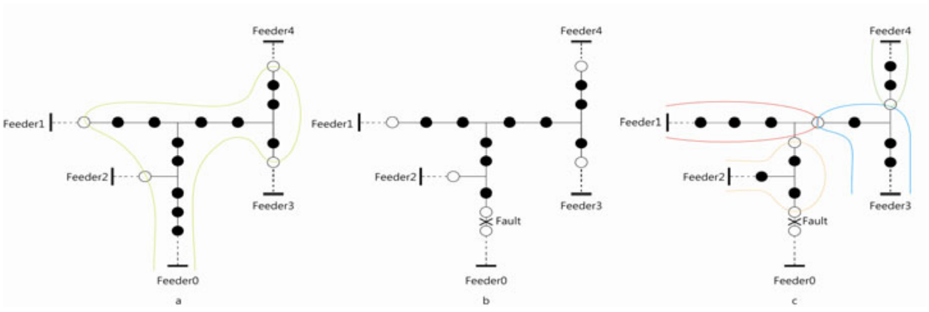


Fig. 1. An example of power system blackout and restoration

2.2 Initialization and Background Knowledge

In the MAS in a smart power grid, we formulate two kinds of agents, feeder agents and zone agents. Feeder agents represent a feeder which has ability to restore the blackout region. Zone agents represent a power load zone which demands power of certain amount to be restored. Each of them possesses some local information. A feeder agent knows its total amount of power can be generated and possesses a zone agent list which contains the zone agents that are reachable and restorable by the feeder agent. A zone agent knows its own power demand and its neighbor zone agents as well as their corresponding power demands. For example, in Figure 2, feeder agent f0 knows it has 33 units of power, and it can reach zone agent z5. Zone agent z1 knows that its power demand is 6. Besides, it also knows its neighbor zone agents z0, z2, z3 and z4 and their corresponding power demands as 5, 7, 8, 9 respectively.

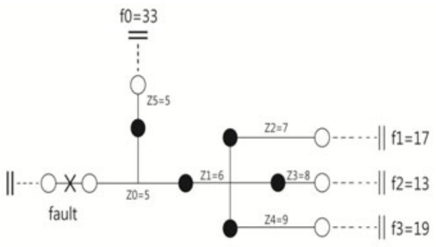


Fig. 2. An example of relations among feeder agents and zone agents

2.3 Stochastic Decision Functions ZDF and PDF

Before explaining the negotiation protocol in this MAS. We first define two PEU (Probability Expected Utility) functions that will be used in negotiation using the idea that we mentioned.

Zone Decision Function (ZDF)

As section 2.2, every feeder agent owns a zone agent list which contains all reachable and restorable zones as candidates to restore the power. Based on the MEU principle, the feeder agent should choose a zone agent that has the maximum expected utility to negotiate so that it can restore the zone. But in our idea, we do not always choose a MEU zone agent. Instead, we define PEU for every zone agent. We first simply define expected utility of every zone agent in the zone agent list of some particular feeder agent *a*.

$$\mathrm{EU_{i,a}} = \mathrm{h_{i,a}}$$

Where $h_{i,a}$ denotes the number of times that the i-th zone has been restored by feeder agent *a* during the negotiation. With the definition of EU, we can formulate PEU using EU now. We name this EPU function Zone Decision Function (ZDF). The following is the formulation of ZDF.

$$\mathrm{ZDF_{i,a}} = \frac{\frac{1}{\mathrm{h_{i,a}}+1}}{\sum_{j=1}^{n}\frac{1}{\mathrm{h_{j,a}}+1}} \text{ subject to } \sum_{i=1}^{n}\mathrm{ZDF_{i,a}} = 1$$

Where *n* denotes the number of zone agents in the feeder's zone agent list.

From the viewpoint of a feeder agent, if it restores a zone many times during the negotiation, it implies that there are other feeder agents that have been attempting to restore the same zone as well. It implies it is a popular zone to some extent. Our goal is to restore as many zones as possible from the global viewpoint. Therefore, the feeder agent had better try to seek other zone agents as a candidate to put in its zone-agent list. For this reason, we formulate ZDF as above so that the probability of restoring a popular zone becomes relatively lower.

ZDF gives every candidate zone agent a probability to be chosen but the probability is feeder agent dependent. Namely when a feeder agent wants to pick up one zone agent to restore its power demand, it bases on its own ZDF. For example, in figure 3, during negotiation, assume that *f0* has already attempted to restore *z2* two times, *z3* four times, and *z4* six times, then we can compute the ZDF as below.

$$\mathrm{ZDF}_{\mathrm{z2,f0}} = \frac{\frac{1}{3}}{\frac{1}{3}+\frac{1}{5}+\frac{1}{7}} = 0.493$$

$$\mathrm{ZDF}_{\mathrm{z3,f0}} = \frac{\frac{1}{5}}{\frac{1}{3}+\frac{1}{5}+\frac{1}{7}} = 0.296$$

$$\mathrm{ZDF}_{\mathrm{z4,f0}} = \frac{\frac{1}{7}}{\frac{1}{3}+\frac{1}{5}+\frac{1}{7}} = 0.211$$

Then, the feeder agent can stochastically choose the zone agent according to its corresponding probability. In this case, the feeder agent will choose more likely *z2* as the target zone agent.

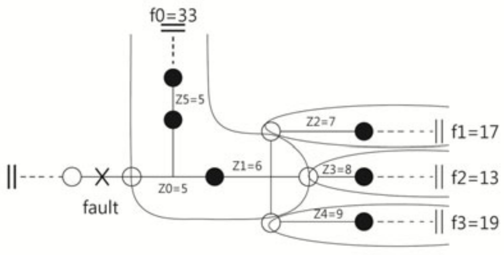


Fig. 3. An example of ZDF to show how feeder agent selects a target zone probabilistically

Proposal Decision Function (PDF)

Every feeder agent only keep the local information of their currently own restoration zones in the zone agent list, they actually don't know the states of the rest of zones in the smart grid. Therefore, when a feeder agent picks one target zone from the neighbors of the current restoration zones using its ZDF and try to include it into the restoration zone agent list, the target zone agent may have already been claimed by other feeder agents. We define this feeder agent the master feeder agent of the zone agent. In this situation, the two feeder agents have to negotiate to decide who should eventually restore the power for the target zone. The feeder agent who wants to restore the target zone will send a proposal to the master feeder agent which currently claims the restoration of the target zone. Master feeder agent will decide to accept or refuse according to a stochastic decision function. Just like ZDF, we first define the EU of the master feeder agent as below:

$$\mathrm{EU} = \frac{\mathrm{a}}{\mathrm{b}} * \left(1 - \frac{\mathrm{x}}{\mathrm{y}}\right)^{\mathrm{f}}$$

Where *a* denotes the sum of power demands of zone agents in the zone agent list of a master feeder agent, *b* denotes the remaining power of a master feeder agent, *x* denotes the sum of power demands of related downstream zones that the master feeder agent has to give up the claim of restoration if it accepts the proposal, *y* denotes the total consumed power for the current restoration claim of the master feeder agent, *f* is a flag, if there are downstream zones from the target zone, then *f* = 1. Otherwise, *f* = 0.

Then we combine the EU of the master feeder agent and the feeder agent who sends the request proposal as a PEU named Proposal Decide Function (PDF). The following is the formulation of PDF.

$$\mathrm{PDF} = \frac{\frac{a}{b}}{\frac{a}{b}+\frac{c}{d}} * \left(1 - \frac{x}{y}\right)^{f}$$

Where *c* denotes the sum of power demands of all zone agents in the zone agent list of the feeder agent who requests to restore the target zone, *d* denotes the remaining power of the feeder agent who wishes to restore the target zone.

PDF presents the probability that a master feeder agent will accept the proposal requested by other feeder agent who wishes to restore the target zone. We formulate PDF based on the following notions.

1. The more power the master feeder agent has, the lower is the acceptance probability. Because the master feeder agent is capable.
2. The higher the sum of power demands in the zone agent list of a master feeder agent, the higher acceptance probability. Because it means the master feeder can be more potential to run out of power.
3. The higher the sum of power demands of zones that a master feeder agent has to give up if it accepts the proposal, the lower acceptance probability. It means the master feeder agent has to lose a lot of utility.

We use an example to explain the parameters in PDF more clearly. In figure 4, *f0* restores *z5*, *z0*, *z1* and *z2*. *f3* restores *z4*. If *f3* sends a proposal to *f0* for *z1*, then *f0* can calculate the parameters for the proposal as in table 1. The PDF calculated accordingly is about 0.322. This means that the probability for *f0* to accept *f3's* proposal is less than 0.5.

Table 1.

a	b	c	d	x	y	f
17 (z3+z4)	10 (33-z5-z0-z1-z2)	6 (z1)	10 (19-z4)	13 (z1+z2)	23 (z5+z0+z1+z2)	1

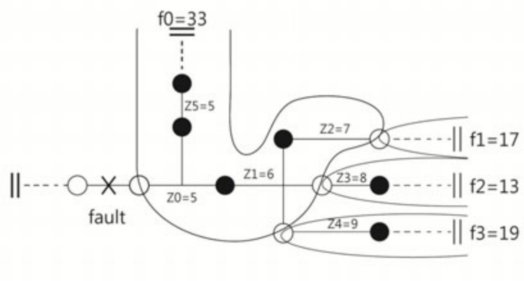


Fig. 4. An example of f0 is requested by f3 for z1 and f0 uses PDF to decide

2.4 Negotiation Protocol

After blackout, feeder agents start the restoration negotiation automatically. They request the zone agents who are in their zone agent list and try to restore them. If a zone agent does not belong to any feeder agent to restore the power, then the zone can be automatically allocated to the feeder agent. If the requested zone has already been allocated to be restored by some master feeder agent, the feeder agent who wishes to restore the zone, will have to negotiate with the master agent to see if it will accept to give away the zone. The complete negotiation protocol can be described as followed:

A. Feeder agent *FA* chooses a zone agent *ZA* from its zone agent list using ZDF and sends a proposal to *ZA*.
B. If *ZA* have not been restored, then *ZA* accepts the proposal. *ZA* sends the required information to *FA*. Both *ZA* and *FA* change their states. The negotiation is done. Otherwise, *ZA* gives *FA* the information about its master feeder agent *MFA*. Therefore, *FA* can communicate with *MFA*.
C. *FA* sends a proposal and the required information to *MFA*.
D. *MFA* accepts/rejects the proposal according to PDF. Then *MFA* sends the decision to the *ZA*.
E. If *MFA* accepts the proposal, *ZA* sends the required information to *FA* who becomes the new *MFA* of the *ZA*. Both *ZA* and *FA* change their states. If *MFA* rejects the proposal, *ZA* sends the reject information to FA. The negotiation is done.

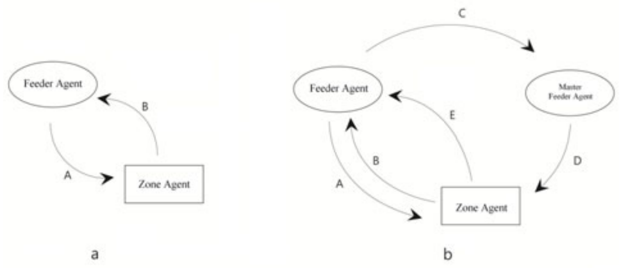


Fig. 5. (a) A feeder agent requests a zone from the zone agent if it does not belong to any feeder agent. (b) A feeder agent must negotiate with a master feeder if the zone he requests to the zone agent belongs to the master agent.

Briefly, all feeder agents are greedy and attempt to restore as much power zones as they could until they have no more power to restore any additional zone. They keep negotiating with zone agents. If a target zone has been restored, they need to negotiate with its master feeder agent. The job of a zone agent is simpler. They only need to response the required information according to their states, and keep the communication history data to be used in PDF and ZDF.

2.5 An Anytime Negotiation Algorithm

Finally, we discuss the decision to terminate the restoration negotiation procedure. Without time-limit constraint, intuitively, the restoration negotiation procedure will terminate in two conditions. First, when all blackout zones have been restored. Second, all feeder agents have distributed all their own power. Sometimes a large area of blackout may take a lot of time to find the optimal solution. However, a blackout may cause a huge economic loss if not be restored in time. For this reason, we usually hope to restore as soon as possible. An anytime-algorithm is one kind of algorithm that can be terminated and output an acceptable result in anytime. Because of its flexible feature, anytime algorithms are used in many domain applications. Our negotiation approach belongs to the anytime algorithms. Therefore, even if we cannot find the optimal solution in limited time, we still can output the "so far" best solution found at any time. This feature makes our approach more suitable for handling power restoration problem.

3 Experiments and Results

We design four special cases to test our approach. In order to test if our approach can reach the optimal solution, the cases are very subtlely designed. There could be many ways to deliver power partially to the blackout zones by the feeder agents, but only one optimal solution exists that can restore all blackout regions.

The first version of our negotiation protocol does not involve the ZDF (zone decision function). Instead, the feeder agent chooses a target zone agent to be restored randomly. Table 2 shows the test results of this version.

Table 2. Number of Negotiaitons without ZDF

	Case 1	Case 2	Case 3	Case 4
number of blackout zones	13	23	35	25
Average number of negotiation (request-response pairs)	24.9	50.5	59.6	60.9

- Average number of negotiation is the average over ten experiments
- One negotiation means a complete negotiation pairs in the negotiation protocol

In this experiment, we can reach the optimal solution in all four cases. Besides, we had two interesting findings. Firstly, comparing cases 3 and 4, we found that if the number of the overlapped zones of feeders' deliverable regions is larger, then it needs more negotiation steps to reach the optimal solution. Secondly, by tracing the negotiation protocol, we discover that some negotiations could be meaningless as shown in Figure 7. In Figure 7, there are four zone agents in the zone agent list of *feeder agent A* and only one for *feeder agent B*. Ideally, *feeder agent A* should choose *zone agents B*, *C*, or *D* instead of *zone agent A*. However, in real situation, *feeder agent A* may

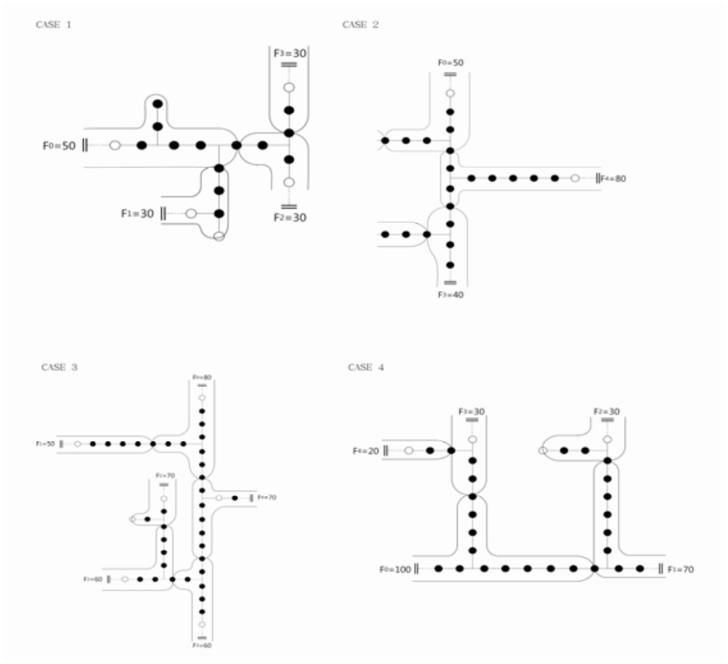


Fig. 6. Four designed test cases

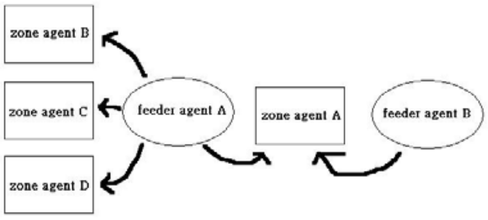


Fig. 7. Multiple zone agents might cause the imbalance choice of a zone agent

choose *zone agent A* because it chooses a zone agent randomly. This will cause *feeder agent A* and *B* keep negotiating for *zone agent A*.

To avoid making the meaningless negotiation, the feeder agents have to avoid the repeated negotiations. In the second version, we involve the ZDF into negotiation protocol. ZDF uses the number of restorations of a zone during the negotiation as the parameter that helps feeder agents to detect the repeated negotiations. Clearly, ZDF helps us to decrease the number of negotiations. Figure 8 shows the performance comparison between the two versions (with ZDF vs. without ZDF).

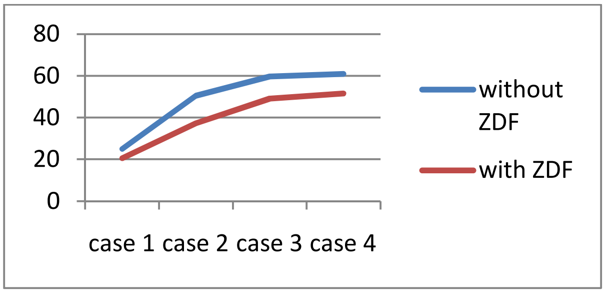


Fig. 8. The performance curves in terms of number of negotiations with ZDF vs without ZDF

Because our stochastic negotiation approach is an anytime algorithm, we are interested in the restoration rate during the whole negotiation procedure. For this reason, we record the number of negotiations at different restoration rate during the negotiation procedure. Figure 9 is our result in which the x-axis represents the restoration rate while the y-axis represents the number of negotiations. We discover that the number of negotiations and the restoration rate are linear relationship in the beginning. However, when the restoration rate is close to 1, it tends to demand lots of negotiations to increase the restoration rate. This result is intuitively correct. According to this result, how to balance the tradeoff between number of negotiations and restoration rate becomes an issue in some situation.

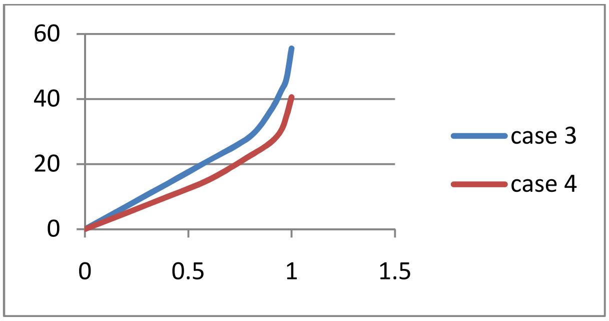


Fig. 9. The number of negotiations against the restoration rate in real time performance

4 Conclusion

Traditional negotiation protocol based on individual rationality of negotiators tends to have to the difficulty to avoid trapping into the local optimal solution. This is troublesome if the negotiation is used in solving the power restoration problems. In this paper, we propose a stochastic negotiation protocol to allow feeder agents to resolve the conflicts of finding the proper power zones to restore power in a smart grid. We implement a stochastic function PDF as decision function to decide if a power zone allocated to a master feeder agent should be given up upon the request of another feeder agent during negotiation. Another stochastic function ZDF supports the decision of a feeder agent to decide which neighbor zone agent should be the best target zone to request the power restoration. With these two functions, the negotiation protocol can find many optimal solutions in four subtle cases. We also show our negotiation algorithm is a real time algorithm and the negotiation protocol tends to improve slowly in the final stage of negotiation when the restoration rate is close to 1. We could use this fact to decide when to decide the continuation of negotiation in terms of quality of the solution under the time limit. We are currently conducting the experiments on benchmark power restoration problems to compare with traditional centralized optimization algorithms and methods in order to evaluation the performance of our stochastic negotiation protocol more rigorously.

Acknowledgements. This research is supported in part by National Science Council of Taiwan, ROC under the grant number NSC 100-3113-E-007-006. We also acknowledge Prof. Men-Shen Tsai at Institute of Automation Technology National Taipei University of Technology to introduce the power restoration problems to us.

References

1. Anders Levander, L.N.: A stochastic negotiation model. Gnosie, Stockholm (2007)
2. Levy, H., Markowitz, H.M.: Approximating expected utility by a function of mean and variance. American Economic Review 69(3), 308–317 (1979)
3. Sandholm, T.W.: Contract types for satisficing task allocation: I theoretical results. AAAI (1998)
4. Andersson, M.R., Sandholm, T.W.: Contract types for satisficing task allocation: II experimental results. AAAI (1998)
5. Weiss, G.: Multiagent systems: a modern approach to distributed artificial intelligence. MIT press (1999)
6. Sankaran, S., Bui, T.: A stochastic negotiation model for organizational choice. In: Proceedings of the 37th Annual Hawaii International Conference on System Sciences (2004)
7. Li, X.D., Xu, Y.Q., Zhang, L.: Distribution service restoration with DGs based on multi-agent immune algorithm. In: 2nd International Conference on Power Electronics and Intelligent Transportation System, PEITS (2009)
8. Lambert-Torres, G., et al.: Particle Swarm Optimization Applied to System Restoration. IEEE Bucharest Powertech, Vols. 1-5, 1992–1997 (2009)
9. Stuart Russell, P.N.: Artificial Intelligence: A modern approach, 2nd edn (2003)
10. Wooldridge, M.: An Introduction to Multi Agent System, 2nd edn (2009)

Coalition-Oriented Sensing in Wireless Sensor Networks

M. del Carmen Delgado-Roman and Carles Sierra

Artificial Intelligence Research Institute (IIIA), Spanish Scientific Research Council (CSIC), Universitat Autònoma de Barcelona, Bellatera E08193, Barcelona, Spain
{delgado,sierra}@iiia.csic.es

Abstract. Wireless Sensor Networks are generally composed of a large number of nodes that monitor their surrounding area. The monitoring capacity of sensors gets altered by the changing conditions of the environment and the sensors' internal state. Sensor coalitions, in which only the leader transmits information to a sink node, are a means to save resources when the conditions of the environment are similar around the sensors in the coalition. In this paper we analyse and formalise such sensor coalitions and propose an algorithm for coalition formation that allows the sensors to self-organise with the purpose of performing a good monitoring of the environment while maximising the life span of the sensor network as a whole. The algorithm uses the quality of the information fused at the coalition leader and the remaining energy of the sensors as the basic parameters to alter coalition membership and leadership.

Keywords: Wireless Sensor Networks, Sensor Coalitions, Resourse Saving Strategies.

1 Introduction

Wireless Sensor Networks (WSN) are becoming widespread thanks to the advances in electronics and wireless communication [1]. Nowadays, there is a plethora of small low-cost and low-energy consuming sensors able to communicate via wireless technology. These devices have made many monitoring tasks simpler and cheaper. Typical applications include environment monitoring, security control, military surveillance or traffic control.

The main challenge that WSNs put to scientists is the management of the hard constraints imposed on the sensors, like low communication bandwidth, little processing capacity and limited availability of energy. These restrictions make it necessary to use a large number of sensors such that there is always a minimum number of operational sensors to do the task. Multiagent system technologies can alleviate such constraints by introducing coordination between sensors to improve the performance of the network. The problem tackled in this paper is the development of energy-saving data treatment strategies based on the local activity of nodes in a WSN. In a generic scenario, the task of a sensor is to sense the environment and send the collected data to a server node, the *sink*. Although WSNs are typically deployed in dynamic environments, periods of stable conditions may not be infrequent. These periods do not require high sensing rates as this would entail transmitting the same data every time. The same argument holds in space: sensors situated close to one another may collect similar data. This phenomenon can be used to check the spatial coherence of the collected data and to save nodes' energy by avoiding the transmission of repeated samples. The inspirational scenario considered for this algorithm proposal is that of a WSN deployed over a waterway to monitor its state.

The distributed algorithm proposed in this paper implements a strategy for (not necessarily optimal) coalition formation in WSNs that trades-off *information-gain* and individual nodes' *energy*. The algorithm is fully distributed and embedded in the sensors functioning regime and we assume that coalitions don't serve the individual goals of agents but a shared common goal. The network's life span is increased at the cost of sending less data to the sink. The balance between quality of information collected and available energy is what determines the network division into coherent coalitions of agents.

In this paper we make two contributions. First, we formalise the problem of environmental monitoring in WSN through the network division into groups. Secondly, we propose an algorithm for coalition formation that allows for a tunable trade-off between information-gain at the sink and life-span of the whole network. To our knowledge no previous works followed this approach.

The rest of the paper is organised as follows. In Section 2, we briefly revise previous contributions to the important field of coalition formation in Multiagent Systems. Next, a formal description of the problem studied is presented in Section 3. Section 4 presents a detailed description of the algorithms designed and finally, we draw some conclusions and discuss future work in Section 5.

2 Related Work

Multiagent Systems (MAS) are composed of distributed autonomous entities that have to coordinate themselves to solve a common task. A simple coordination mechanism for agents is to organise themselves in coalitions and cooperate to share resources or reach goals that cannot be achieved individually. From a MAS perspective, coalitions represent a fundamental form of organisation.

According to the classical formulation, a coalition is defined as a set of self-interested agents that cooperate to achieve a common goal. Agents try to maximise their individual and groupal payoff while guaranteeing coalition stability. According to [14], Coalition Formation (CF) in MAS can be studied from three different perspectives:

– *Task allocation.* Many MAS applications require agents to join forces for a period of time to solve a task. Contract Net Protocol [13] represents one of the first proposals in this line.
– *Social networks.* This research line uses coalitions to study the emergence and behaviour of organisations in environments without clearly defined interaction mechanisms [4].

– *Game theory.* This approach to CF does not focus on the design of agents' strategies to reach beneficial coalitions [14] but on the coalitions stability.

CF in its traditional view, as a *static* and *centralized* problem, is NP complete [10]. However MAS and WSN do not satisfy this view as their application environments are typically *dynamic* and *distributed*. Hence, a number of alternative mechanisms for coalition formation in this kind of environments have been proposed in recent years.

A CF protocol for task accomplishment in situations of incomplete information was proposed in [8]. In the studied scenario, agents collaborate and reach decisions for task completion with incomplete information about the environment and the other agents. The algorithm is based on a negotiation process developed for a so called Request For Proposals domain. In the considered scenarios, business agents tackle complex decomposable tasks that require the formation of groups of provider agents to solve them.

Task oriented CF in dynamic environments faces the problem of high power and bandwidth consumption due to continuous configuration and reconfiguration processes to adapt to evolving system conditions and demands. To avoid that excessive consumption, [2] proposes a task oriented CF in which the coalition duration is calculated following some fuzzy rules applied to the historical behaviour of the agents and the characteristics of the tasks arriving to the system.

The previously presented approaches do not pay attention to individual agents cooperative or self-interested will. This dimension was added to the problem through the concept of clan [7]. A clan designates a set of agents that have similar aims and who also trust each other. In this case, group formation is not only determined by task accomplishment, but by the agents' motivation and trust relationships, originating mid-term duration coalitions.

The application of CF techniques to sensor networks has also been investigated by numerous researchers. For instance, a vehicle-tracking sensor network is modelled using disjoint coalitions of homogeneous agents in [12]. Coalitions are formed via a negotiation process based on local and social marginal utility calculations that take place in an incomplete information scenario. To maximise the system's performance, the proposed algorithm enables the self-organisation of the system by allowing the agents to discover their organizational relationships during the negotiation process. As a result, CF can also be observed as an organization method in MAS, as it naturally fits within the structure of a system without a central authority.

Assuming that sensor networks should be inherently adaptive, [9] proposed the Dynamic Regions Theory, whose objective is to optimize the overall operation of the network through its own partition into several regions that execute different algorithms. The network partition is derived from the individual nodes' role election according to their current circumstances and the system global policy. The goodness of this approach is shown for a specific gas plume detection scenario. A new dimension was added to the problem of CF in [5]: the study of the influence of the network topology structure in a MAS perfomance for task solving. The system divides itself into disjoint groups to accomplish the demanded tasks. During the execution, agents can rewire their connections to form better coalitions according to a degree of connectivity or a performance-based policy. A new network adaptation policy for the same situation was introduced in [3]. In this case, agents rewire the network according to their similarity to their neighbours. They also choose to which group they want to adhere and when to leave it based on some task and group success indicators. In the same line, [6] enriched the previous situation by considering a more realistic coalition model. The two rewiring policies developed in the system were based on performance and a different similarity definition. As in previous approaches, the implementation of these policies makes the system outperform an initial situation without rewiring capacity. However, none of these three approaches takes into account the energy consumption and cost derived from the rewiring pocilies.

Guided by the same objective of extending the lifetime of a glacial sensor network, [11] proposes an algorithm for adaptive sampling. Nevertheless, in contrast with our approach, nodes there follow an individual policy to reach the desired objective, while we focus on groupal strategies to save energy in sampling tasks.

In this paper, we propose a CF strategy for homogeneous nodes in a sensor network scenario. The sensor nodes' task is to monitor the behaviour of the environment in which they are deployed. In contrast with previous work, the CF strategy aims at saving energy to extend the network lifetime. This is achieved by allowing nodes in a coalition to delegate their sensing tasks to other neighbouring nodes, while restricting the maximum information loss so that the initial purpose of the system —faithfully monitoring the environment— is not missed.

3 Problem Formalisation

A WSN is composed of an initial set of cooperative and homogeneous nodes, from now on agents. Each of these agents has the same sensing capability, so they all can sample the variable x being observed at any time t. The basic behaviour of an agent consists of sensing the environment and sending this information to a server or sink. We will note by $A = \{a_1, \ldots, a_i, \ldots, a_N\}$ the set of sensing agents and by a_s the sink agent, $a_s \notin A$.

To save system resources, agents will organise themselves into disjoint groups. Nodes in a group accomplish their tasks together as an entity, avoiding redundant sensing by the members of the group and unnecessary routing among them. In this way, we will save energy from the batteries of the sensors. To find an appropriate division of the agents at time t, we take into account the similarity of the individual measurements and the topology of the neighbourhood structure. The unit distance assumed for this scenario is one radio hop. Let $d : A \times A \rightarrow \mathbb{N}$, be the distance between two nodes, measured as the minimum number of radio hops between them. The physical properties of wireless communication guarantees that d is a metric distance. In particular, d is commutative, $d(a_i, a_j) = d(a_j, a_i)$, and $d(a_i, a_i) = 0$.

Now we can define the notion of neighbourhood. Based on d, and given a set of agents A, we call $Ne : A \rightarrow 2^A$ a *neighbourhood function* if and only if $a_j \in Ne(a_i) \Leftrightarrow d(a_j, a_i) = 1$.

The coalition structure is then defined for a maximum distance α among its members. The α parameter influences the maximum achievable size of the groups in a coalition, and hence the minimum granularity of the network. Given a set of agents A, an *α-distance coalition structure*, $c_\alpha = \{g_k\}_{k:1..K}$, is a partition of A in K groups, such that $\forall a_i, a_j \in g_k$, $d(a_i, a_j) \leq \alpha$. We note by C_α the set of all possible α-distance coalition structures and the current coalition structure at time t, as c^t.

The criterion that guides the formation of the different coalition structures is to find (in a distributed manner) the best partition so that the quality of the information sent to the sink is somehow maximised and the energy consumption of the system somehow minimised. For such scenario, we propose the *Coalition Oriented Sensing Algorithm* (COSA), a tuneable algorithm able to fulfill these requirements. The adaptability of COSA is achieved through the definition of a set of parameters p (to be explained later) whose values are going to drive the agents' behaviour. For a certain p parameters configuration, agents take different kinds of sampling and *transmission actions*, represented as $m^j \in M_p$, where M_p is the set of existing actions available for that p configuration. The objective of minimising the system's energy consumption is formally expressed in the first part of (1). According to this, we try to find an optimal set of parameters $p*$, where m_i^j is the action j taken by agent i and E_j represents the energy consumption associated to that action.

$$p* = \arg\min_{p \in P} \Delta E = \arg\min_{p \in P} \sum_{m^j \in M_p} \sum_{a_i \in A} \# m_i^j E_j; \quad \text{s.t.} \quad QoI(t) \geq \epsilon \quad \forall t. \tag{1}$$

The identification of $p*$ is subject to guaranteeing that the Quality of Information received at the sink (QoI) is over a certain threshold (second part of (1)). Two different concepts can be used to *measure* the quality of the data sent to the sink: Pearson's coefficient of variation and Information Entropy.

– *Pearson's coefficient of variation* (CV) is a rough measure of relative dispersion. The value of this coefficient for a group g_k is $CV(g_k) = \frac{\sigma_k}{\bar{x}_k}$.
– *Information entropy* is a measure of the uncertainty or noise of a random variable. Applying this concept to this problem implies considering a group's entropy an indicator of the dissimilarity among the different group components. The information entropy associated to a generic group g_k is given by $H(g_k) = \ln(\sigma_k \sqrt{2\pi e})$.

According to these two concepts, the quality of the information of the system for a certain configuration can be expressed in two different ways.

$$QoI(t)\rfloor_H = \left(\sum_{\substack{c^t \in C_\alpha \\ c^t = \{g_1^t, .., g_n^t\}}} \frac{H(g_k^t)}{n^t} \right)^{-1}; \; QoI(t)\rfloor_{CV} = \left(\sum_{\substack{c^t \in C_\alpha \\ c^t = \{g_1^t, .., g_n^t\}}} \frac{CV(g_k^t)}{n^t} \right)^{-1} \tag{2}$$

3.1 Agents' Coalition Formation

A group in a coalition structure acts as an entity. The leader of the group senses and sends a groupal value to the sink. If this value is a good representative of the variable value in the region covered by the group then, the group, as a whole, saves energy and computational resources.

Group formation is based on a peer-to-peer negotiation protocol by means of which agents exchange information about their measurements (*adherence*) and their adequacy to represent their neighbours (*leadership*).

The *adherence degree* of an agent i to an agent j is a measure that indicates how much agent i intends to form part of a group led by agent j. The higher the degree, the higher the intention. The *adherence degree* is defined as the product of two factors. To evaluate those factors, we assume that the variable under observation follows a Normal distribution. Every agent i knows an initial approximation of that distribution, $\mathcal{N}_i$. This distribution is updated by agent i as it collects new samples, $\mathcal{N}_i(f(\bar{x}_i, x_i), g(\sigma_i, x_i))$ (for appropriate f and g functions).

The first factor in the adherence expression (3) captures the similarity between the measurements of agents i and j. It is defined as the quotient between the probability that the measure of an agent comes from the distribution of the neighbour (which could then be considered as the *same* value, perhaps with some noise) normalised by the maximum probability reachable in that distribution. To avoid unproductive computation, the similarity factor is only defined for neighbour agents that verify that $\|x_j - x_i\| \leq d_{max}\sigma_j$, where d_{max} is a parameter and x_i, σ_j are the corresponding sample and deviation of agents i and j. On the other hand, the second factor captures the *goodness* of the neighbour's distribution and avoids obtaining high adherence values to neighbours with wide distributions. To get this, this factor restricts the evaluation to those neighbours whose σ belongs to the interval $(\sigma_{min}, \sigma_{max})$ through the evaluation of the distribution's entropy normalized on that range.

As a result, the evaluation of the degree to which an agent a_i may be interested in being led by one of its neighbours a_j is calculated as follows:

$$adh(a_i, a_j) = \frac{p(x_i, \mathcal{N}_j(\bar{x}_j, \sigma_j))}{p(\bar{x}_j, \mathcal{N}_j(\bar{x}_j, \sigma_j))} \cdot (1 - \frac{e^{H_j} - e^{H_{min}}}{e^{H_{max}} - e^{H_{min}}}) \tag{3}$$

Note that the set of p parameters, as presented previously, can be identified now as $p = \langle d_{max}, \sigma_{min}, \sigma_{max} \rangle$ defined over the space $p \in \mathbb{R}^3$. The set of values to which these parameters are set influence the actions that a node can take.

When an agent receives an adherence value from a neighbour, it has to decide whether it is interested in becoming the leader of this agent or not. Let us call $P(a_i)$ (potential group) the group formed by a_i and the agents willing to become part of a group led by a_i. The attitude of a_i as a leader of this group depends on different factors that can be identified in (4). The first factor is called *prestige* and it is an average of the adherence level of the group's members. The *capacity* factor indicates the available energy of the node to act as a leader. This value is derived from the current energy level of the node minus the security energy level (E_{sl}) divided by the maximum energy level available E_{max}. E_{sl} defines the minimum energy that the node has to keep to ensure sending one last message before completely depleting its battery.

Finally, the last factor in (4), *representativeness*, indicates how well the potential leader's measurement fits as a representative of the potential group agents' measurements. So, a_i characterizes the set of data received together with its own data, that is, the set $\{x\}_{P(a_i)}$, with their mean and standard deviation, noted as $(\bar{x}_{P(a_i)}, \sigma_{P(a_i)})$. To encourage the formation of groups with very similar measurements, an exponential function establishes the divergence growing ratio: the larger the difference between the leader sample and the mean, the larger the penalty. Those potential groups whose measurement distribution is very disperse are also penalized through the inclusion of the Pearson's coefficient in the equation.

A good group leader is an agent who has enough energy and whose measurements are similar enough to the measurements of the other group members. In summary, the leadership capacity of an agent a_i for its potential group $P(a_i)$ is calculated as follows:

$$lead(a_i, P(a_i)) = \frac{\sum_{a_j \in P(a_i)} adh(a_j, a_i)}{N} \cdot \frac{E(a_i) - E_{sl}}{E_{max}} \cdot \frac{1}{e^{|x_i - \bar{x}_{P(a_i)}| CV_{P(a_i)}}} \quad (4)$$

4 Operational Protocol

Agents' preferences for coalition formation change due to the dynamics of the environment and the dynamics of the sensors. Based on the values of adherence and leadership, agents negotiate to form groups, trying to achieve their most preferred configuration at each time. The default situation is that of every agent alone constituting a group by itself (led by itself). In this section, we describe the algorithms that underpin the behaviour of the agents.

4.1 Coalition Formation Protocol

The coalition formation algorithm that all nodes execute can be divided in four processes that run simultaneously:

– *Sample information exchange.* This process corresponds to the variable sampling and measurements broadcast.
– *Adherence graph construction.* Once the agent has calculated the adherence degrees to its neighbours, it communicates the maximum adherence value to the corresponding most preferred neighbour.
– *Leadership information exchange.* Based on the current adherence relationships, the agent calculates and communicates its attitude as a leader towards the agents willing to adhere to it.

– *Group definition.* Depending on the information available for an agent at a certain moment, it decides whether to stay in its current group (as a leader or dependant of a leader node), to leave this group to join a different one or to constitute its own group.

The messages exchanged between sensors in this negotiation follow a classical agent communication format: *performative(sender, addresse(s), msgContent, [time]).* The time field is an optional item that is used depending on the kind of performative. The set of performatives that the agents use are:

– *inform*: to indicate the transmission of data (measurements, maximum adherence and leadership values).
– *firmAdherence*: to express the desire of the sending node to adhere to the addressee node.
– *ackAdherence:* to express the acknowledgment to a previously received firmAdherence message.
– *break*: for a leader node to break a leadership relationship.
– *withdraw*: for a dependant node to break a leadership relationship.

Note that within the first three processes listed before, only the inform performative is used, while the rest of the performatives are used within the group definition process.

4.2 Generic Agent Behaviour

From an external point of view, it can be said that agents behave in a proactive and reactive way. Proactive because the core behaviour of an agent is the continuous process of looking for the best group of neighbours that matches with its measurement and its state. To achieve this objective, an agent exchanges messages asynchronously with its neighbours. Reactive because their acts and decisions are triggered by the observation of the environment and the information they receive.

The CF protocol is embedded in the agent behaviour via the execution of the *Information Processing* thread and the actions corresponding to the agent role at any moment (leader or dependant). The role changes along time depending on the information available at a certain moment (collected and processed through the *Information Processing* thread). When an agent is a leader, it starts sampling the environment and sending the measure to both its neighbours and the sink. Awaken agents communicate with each other to find an adequate configuration in which some of them may end up asleep for a preestablished time. Leader agents continue sensing and sending data to the sink according to the frequency demanded by the application, but this time, they work on behalf of their depending neighbours. When an agent is not a leader, it can, according to a certain probability, periodically take part in the CF configuration process together with all leader agents. This way it can change its state to become a leader of itself and others, depending on its current conditions.

Algorithm 1. Information Processing

```
Data: $me$: focus node; $a_j$: generic neighbour; $a_l$: potential leader; $a_r$: potential
      dependant on me; $a_p$: dependant node on me; $a_L$: leader node of me;
      $D(me)$: set of dependant nodes on me
case rcvd(inform($a_j, me, meas, t$))
    updateNeighbourInfo();
    adherence2NeighbourEvaluation();
    updateOwnMaxAdherence();
    if changesOnOwnMaxAdherence then
        inform($me, a_l, maxAdh, t$);
    end
end
case rcvd(inform($a_j, me, maxAdh, t$))
    inform($me, a_r, lead$);
    updateNeighbourInfo();
    adherence2NeighbourEvaluation();
    updateOwnMaxAdherence();
    if changesOnOwnMaxAdherence then
        inform($me, a_l, maxAdh, t$);
    end
end
case rcvd(inform($a_l, me, lead$))
    if checkAgainstOwnLead then
        firmAdherence($me, a_l$);
    end
end
case rcvd(firmAdherence($a_r, me$))
    if checkAgainstOwnLead then
        ackAdherence($me, a_r$);
        updateOwnLeadValue();
        updateDependentGroup();
    end
end
case rcvd(ackAdherence($a_l, me$))
    if $\neg leader \wedge a_l != a_L$ then
        withdraw($me, a_L$);
    end
    if $leader \wedge D(me) != \emptyset$ then
        while $D(me) \neg = \emptyset$ do
            break($me, a_p$);
        end
    end
    updateRoleState(dependant);
    sleep($t$);
end
case rcvd(break($a_L, me$))
    updateRoleState(leader);
end
case rcvd(withdraw($a_p, me$))
    $D(me) \leftarrow D(me) \backslash a_p$;
    updateRoleState(leader);
end
```

The core of the CF process is contained in Algorithm 1. This algorithm has been designed following a simple reactive structure in which all the actions are triggered by an event that changes the available information of an agent. This perspective allows the agent to decouple the different negotiation dialogues it may be involved in and also the different stages of each process. From a global point of view, information flows among the agents and it is this flow that makes agents react.

4.3 Example of Coalition Formation

Figures 1–3 illustrate how the algorithm works and some of the different coalition structures it may originate for a simple scenario of three agents. We start the illustration in a point where each agent has already been informed of the adherence preferences of its neighbours. Figure 1 shows the adherence graph where each agent is linked to its most preferred neighbour at a certain instant. For instance, agent a_1 has informed a_2 about its maximum adherence. The

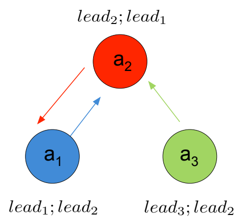


Fig. 1. Adherence relationships and lead information at each node

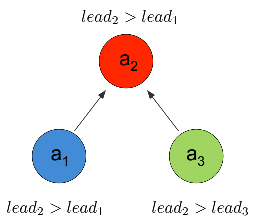


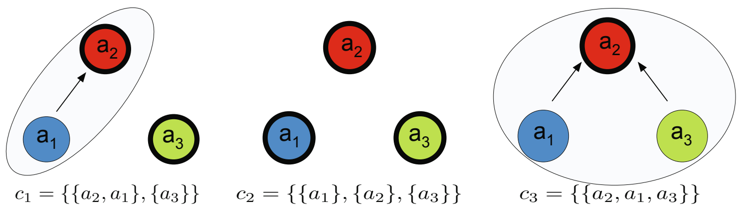
Fig. 2. Lead information evaluation and firm adherence messages emission

establishment of an adherence relationship implies enriching the information available for an agent with the leadership attitude of its preferred neighbour (Algorithm 1, line 10). So agent a_2 sends its current leadership attitude to a_1 and a_3. Agent a_2 receives the lead valueof a_1 and a_3 does not communicate its lead attitude to any other agent because no one wants to adhere to it. Once an agent gets the information about its neighbours' lead attitude, it decides whether it stands by this relationship or not (lines 18 – 22). This decision is reached by comparing its own lead value to the neighbour's one (Fig. 2). Depending on the agent's current role (leader or dependant) this comparison could be done with different thresholds to encourage different desired behaviour of the overall WSN, i.e., agents could be more or less reluctant to changes or demanding on their neighbours' leadership strength.[1] As a result of these comparisons, agents still willing to adhere to a neighbour send formal adherence messages that may or may not be answered by their potential leaders. When a potential leader receives a firm adherence message, it checks if it would be as good leader of the expanded group as it is of its current group. If so, it adds this new agent to its group, updates its lead attitude and sends an acknowledgement message to the new node to put it to sleep. On the other hand, if the agent's leadership would decrease by accepting the new neighbour, the agent will not answer to the potential new dependant, that may continue looking for a group to join (lines 23 – 29).

Different group configurations can be reached depending on the sequence of messages exchanged and the order in which they are received. Figure 3 shows three possible coalition structure configurations (with different number of groups each) reachable for this example. Each group of the coalition structure is described as a tuple formed by the leader and its set of dependent agents.

In the same way that coalition groups are formed, they can be broken by a leader or dependant agents (lines 42 – 48 in Algorithm 1) when new information about neighbours is recieved or when its internal state (e.g. remaining energy) makes the agent's preferences change (e.g. joining a different group).

[1] The addition of these thresholds would imply changing the set of parameters p to include them and also changing its definition domain.



$c_1 = \{\{a_2, a_1\}, \{a_3\}\}$ $c_2 = \{\{a_1\}, \{a_2\}, \{a_3\}\}$ $c_3 = \{\{a_2, a_1, a_3\}\}$

Fig. 3. Three possible resulting coalitions from the situation depicted in Fig. 2

It cannot be said that agents take their decisions and actions based on the actual *situation* of the environment (as such *information* may not be available), but on the current *information* about the environment. An agent is continuously involved in different negotiation processes with all its awaken neighbours and the decisions it takes are based on its most recent information. The intuition of the system function is that the more information an agent gets about what is happening, the more beneficial the coalition will be.

5 Conclusions and Future Work

In this work we have formalised and modelled the problem of coalition formation among distributed agents in a sensor network monitoring a wide area. For the considered scenario, agents join in coalitions in order to trade-off the quality of the sensed values and the life span of the network as a whole. The proposed distributed algorithm specifies the behaviour of the individual agents to accomplish their tasks and reach an appropriate group organisation. The algorithm's dynamics entail a changing distributed sampling of the environment, where the number of samples sent to the sink node depends on the current coalition structure of the network. The proposed algorithm is highly tuneable and eases the addition of an external control module on it. Currently, the experimentation to test the protocol is ongoing. Future work includes the extension of the approach over different sampling variable distribution models and different neighbourhood topologies. The influence of the social network derived from the preference graph structure on the topology and coalition distribution has also to be further explored. Finally, the comparison of the results of the algorithms for the two previously presented problem formulations (the first one based on Pearson's coefficient and the second one based on Information Entropy) will allow to identify which of the models better fits the experimental solution of the problem studied.

Acknowledgments. This work has been supported by the Agreement Technologies project (funded by CONSOLIDER CSD 2007-0022, INGENIO 2010).

References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Communications Magazine 40(8), 102–114 (2002)
2. Bai, Q., Zhang, M.: Rational, Robust, and Secure Negotiations in Multi-Agent Systems. In: A Fuzzy Logic-Based Approach for Flexible Self-Interested Agent Team Forming. SCI, vol. 89, pp. 101–113. Springer, Heidelberg (2008)
3. Barton, L., Allan, V.H.: Methods for Coalition Formation in Adaptation-Based Social Networks. In: Klusch, M., Hindriks, K.V., Papazoglou, M.P., Sterling, L. (eds.) CIA 2007. LNCS (LNAI), vol. 4676, pp. 285–297. Springer, Heidelberg (2007)
4. Gasser, L.: Social knowledge and social action: heterogeneity in practice. In: Proceedings of the 13th International Joint Conference on Artifical Intelligence, vol. 1, pp. 751–757. Morgan Kaufmann Publishers Inc., San Francisco (1993)
5. Gaston, M.E., des Jardins, M.: Agent-organized networks for dynamic team formation. In: Proceedings of the 4th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2005, pp. 230–237. ACM, New York (2005)
6. Glinton, R., Scerri, P., Sycara, K.: Agent-based sensor coalition formation. In: 2008 11th International Conference on Information Fusion, pp. 1–7 (July 2008)
7. Griffiths, N., Luck, M.: Coalition formation through motivation and trust. In: Proceedings of the 2nd International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2003, pp. 17–24. ACM, New York (2003)
8. Kraus, S., Shehory, O., Taase, G.: Coalition formation with uncertain heterogeneous information. In: Proceedings of the 2nd International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2003, pp. 1–8. ACM, New York (2003)
9. Mac Ruairí, R., Keane, M.T.: The dynamic regions theory: Role based partitioning for sensor network optimization. In: Proceedings of the 6th International Conference on Autonomous Agents and Multiagent Systems (2007)
10. Mas-Colell, A., Whinston, M.D., Green, J.R.: Microeconomic Theory. Oxford University Press (June 1995)
11. Padhy, P., Dash, R.K., Martinez, K., Jennings, N.R.: A utility-based sensing and communication model for a glacial sensor network. In: Proceedings of the 5th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2006, pp. 1353–1360. ACM, New York (2006)
12. Sims, M., Goldman, C.V., Lesser, V.: Self-organization through bottom-up coalition formation. In: Proceedings of the 2nd International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2003, pp. 867–874. ACM, New York (2003)
13. Smith, R.G.: The contract net protocol: High-level communication and control in a distributed problem solver. IEEE Transactions on Computers C-29(12), 1104–1113 (1980)
14. Vig, L., Adams, J.A.: Coalition formation: From software agents to robots. J. Intell. Robotics Syst. 50, 85–118 (2007)

Reducing the Environmental Impact of New Construction Projects through General Purpose Building Design and Multi-agent Crowd Simulation

Kieron Ekron, Jaco Bijker, and Elize Ehlers

University of Johannesburg, South Africa

Abstract. This paper presents a two stage process for using intelligent agent technology in designing space-efficient buildings in order to reduce their environmental impact. An environment editor for designing new construction projects is described, followed by a microscopic crowd simulation model that is used to test the operational efficiency of the designed building. Each member of the crowd is represented as an intelligent agent, which allows for more complex goal-directed behaviour, which in turn leads to more realistic crowd behaviour. Crowd simulations can be used to detect potential problem areas, as well as identify areas that may safely be made smaller.

1 Introduction

Recent ecological disasters such as the Deepwater Horizon oil spill and Fukushima nuclear disaster have alerted many people to the importance of environmental conservation. Yet with a current estimated global population of 6.9 billion people [17], new developments are needed to support an ever expanding global population. These developments include housing, office buildings and shopping malls to provide necessary groceries.

This paper focuses on the use of intelligent agents to assist in designing space-efficient buildings. Smaller buildings use fewer resources and require less land for construction, which helps to reduce the environmental impact of the construction project. A two-step process is used to accomplish this goal: an interactive, general purpose environment editor integrated with a microscopic crowd simulator.

Each member of the crowd is represented as an intelligent agent. The use of intelligent agents allows for more complex behaviours to be defined, since each agent maintains its own state and chooses its next course of action based on that state. This paper will describe an extensible method for managing and controlling the state of agents, which allows for a greater degree of customisation of simulations. The high degree of customisability greatly increases the generality of the proposed system.

The paper is structured as follows: section 2 presents a summary of related work; section 3 details the proposed integration of a environment editor and a crowd simulator; section 4 provides the results of two different simulations based on the proposed system; and finally, section 5 concludes the paper and focuses on potential future work.

2 Related Work

Before moving onto the contribution made by this paper, a survey of existing literature will be presented to place the contribution into context.

Section 2.1 discusses related work in measuring the environmental impact of construction projects. Section 2.2 focuses on advancements in computer-aided design. Section 2.3 provides details about related work in the field of crowd simulation.

2.1 Environmental Impact of Construction Projects

The performance of a construction project has traditionally been measured in terms of time, cost and quality. Recently, however, environmental concerns have also become an important performance measurement [4].

Construction projects require consumption of energy and non-renewable resources, and they can generate large amounts of pollutants, including emissions, solid waste, liquid waste and noise [5]. Various methodologies have been developed to help identify and assess the environmental impacts associated with construction. Quantification of environmental impacts can be calculated based on a combination of quantity, toxicity, affected volume, probability and persistence [5]. Some of these factors are dependent on the building site, whereas others are not. For example, the scale of the impact, the probability of occurrence and the duration of the impact are not dependent on the building site. Some of these factors are also easier to quantify than others. For example, water usage can be directly determined based on the amount of concrete used, since water is used to clean the concrete chutes, as well as mix the concrete. On the other hand, impacts such as habitat destruction have no directly observable correlation with the usage of construction resources [10].

UrbanSim [12] is a software system that is used to simulate the development of urban areas – including land use, transportation and environmental impacts – over periods of 20 or more years. It considers a number of factors – such as birth rates, movement of household locations, land price, job creation and traffic analysis – to determine the impact of developments in urban areas.

Computer-aided design is an important part of the design process for new construction projects. The following section describes some of the work that has been done to standardise CAD representations and software.

2.2 Computer-Aided Design

Within the building industry, there are many different disciplines, each with its own CAD software and design specifications [9]. Designs can range from simple 2D representations, such as those used by an architect, up to full 3D models used by interior designers and engineers. Neutral CAD formats for data exchange do exist, however they can lead to data loss, since the information deemed important by one party can be seen as extraneous or unnecessary by another [9].

Industry Foundation Classes (IFC) is a standard for building data presentation and exchange that provides an independent system that can support data management of all systems in the building profession [11]. IFC only requires the standard to be applied during data exchange; it does not need to be implemented inside the software system. It has an object model to describe how elements such as doors, fans and windows are represented. Each object consists of various attributes that can be used to describe every aspect of the object. For example, a fan can store its physical size, energy usage, estimated lifetime, etc. The benefit of this approach is that it allows for various specialists to make use of different properties of the same object. For example, engineers would focus on technical aspects, such as energy usage, whereas an architect would be more concerned with aesthetic aspects, such as size and location. This design, therefore, supports a high degree of interoperability between applications that support IFC.

One of the goals of this paper is to show how crowd simulation can be used to aid in the design process. The following section provides a brief overview on some of the existing models of crowd simulation.

2.3 Crowd Simulation

Microscopic models treat each member of the crowd as an individual; macroscopic crowd behaviour arises due to interactions between individuals in the crowd. A number of different models have been proposed, using simple rule-based systems [14], models based on laws of physics [7], cellular automata [3] and models inspired by robotics [18]. Microscopic models are often capable of producing the most accurate behaviour, but this accuracy comes at the price of high computational costs. As such, microscopic models do not typically scale as well as macroscopic models.

Macroscopic models, such as those used in [8,1,16], move away from individual, specialised behaviours in favour of a generalised view of the entire crowd. Because of this, macroscopic models tend to allow very large crowds to be simulated in real time but they are limited in the amount of diverse behaviours that can be simulated. These models tend to work well for large crowds where individuals have similar goals, such as crowds at a stadium or concert.

Having now discussed previous work in computer-aided design and crowd simulation, the next section introduces the contribution made by this paper: Simulacrum.

3 Simulacrum

Simulacrum is a system that utilises intelligent agent technology and combines computer-aided design with crowd simulation for the purpose of analysing and improving building designs. It consists of a graphical environment editor for constructing building designs, a simulation configuration system that allows a user to configure various aspects of the simulation, and an analysis module for viewing completed simulations and extracting feedback. Section 3.1 discusses the enviroment editor in more detail, while section 3.2 focuses on the crowd simulation model employed by Simulacrum.

3.1 Environment Editor

The environment editor was designed to be as general-purpose as possible, in order to support a wide variety of building types. To that end, the editor has several objectives:

- Simplicity
- Support for multiple floors
- Support for environmental objects
- Generality

Each of these objectives is discussed in further detail below.

Simplicity. A building can be viewed as a collection of interconnected rooms. It is therefore important to find both an appropriate representation for this system of interconnected rooms, as well as to provide authoring tools that simplify the task of creating a building design. The proposed environment editor provides a graphical interface with a top-down 2-dimensional view that allows the user to create rooms through simple drawing commands, such as plotting points. Rooms are represented as closed polygons, since these are simple to create and represent a large majority of room layouts. The implication of this representation is that the current model does not directly support circular rooms.

This method works very well for designing single floors, but very few modern buildings have only one floor. Therefore, a method of designing multi-floor buildings is also required.

Support for Multiple Floors. One of the best ways to conserve land usage is to make use of multi-story buildings. The environment editor must therefore provide a simple mechanism for designing buildings with multiple floors. In existing buildings, traversing between floors is accomplished through the use of either staircases or elevators; escalators are simply special types of stairways that move. To support multiple floors, the editor makes use of an attribute to mark a room as a multi-floor room that exists on two floors simultaneously. These rooms are typically either staircases or elevators.

Support for Environmental Objects. No building only consists of empty rooms. In order to represent real-world conditions better, there needs to be a mechanism for adding environmental objects, such as furniture.

Environmental objects can be obstructions or items inside an environment that are not part of the building foundation, such as furniture. Like rooms, objects are assigned to a specific floor, and it is possible for an object to be assigned to two floors if it is located in a multi-floor room. Objects are represented by their bounding boxes in 2-dimensional space. Objects support normal vector operations such as rotation, scaling and translation. Because rooms are represented as polygons, it is a simple matter of performing a point-in-polygon test to determine whether or not an object is inside a room. This allows the user the drag and drop objects, and the editor can determine the correct floor for the object.

Generality. One of the primary goals of this paper is to create a general-purpose model for assisting in building design. Just as IFC defines a multitude of attributes to better support information sharing, the editor allows the user to define new types of attributes that can be assigned to any element within the building design. Attributes can be text, number or Boolean values. Files can also be stored as attributes if they are first encoded as text. This allows data such as images to be used as attributes.

An important design goal is to make the editor indifferent to these attributes; the editor assigns no semantic value to these attributes. Instead, these attributes can be used by other applications that would consume them, such as analysis and simulation tools. As an example, a room can be marked as a staircase that wheelchair-bound people are unable to traverse. Only the simulator would assign a semantic value to the attribute and prevent people in wheelchairs from traversing the staircase; the attribute has no impact in the editor itself.

In addition to user-defined properties, building designs are saved in extensible markup language (XML). This makes the design understandable to humans, as well as supporting knowledge interchange since XML is an open format, which is easy to process and parse.

Having now discussed the editor used to create building designs, the following section discusses the model employed by the crowd simulator to produce simulations.

3.2 Crowd Simulator

Once the environment has been created, the next phase is to perform a crowd simulation in order to gauge the performance of the designed building. Simulacrum makes use of intelligent agents to represent individuals in the crowd; each person is represented as an intelligent agent. Because of this, the crowd simulator employs a microscopic simulation model, since microscopic models focus on simulating individual behaviours for each member of the crowd.

This section describes the general purpose, programmable crowd simulator used by Simulacrum. The simulator can be used to perform analyses on building designs. This section describes the three main components of the simulator: goal-directed behaviour, path planning and movement in the environment.

Goal-Directed Behaviour. Creating an accurate representation of crowd behaviour requires an accurate representation of the behaviour of people within the crowd. A number of logical constructs have been devised to facilitate the general-purpose simulation of crowd behaviour: world events, attributes and goals.

World events are used to represent external signals that agents can respond to, such as triggering a goal or updating an attribute. A fire alarm would be an example of a world event, since it triggers a goal (evacuate the building) and it can trigger an update in attributes (increased levels of fear). Each world event has a list of activation and deactivation conditions, which are used to determine when a world event should become active and inactive, respectively.

Attributes are used to represent internal state information about an agent. Agent attributes can represent simple physical properties, such as fatigue, or they can represent complex emotional states, such as fear and anxiety. An attribute is represented as a floating point number, combined with a collection of update policies. Update policies are used to determine when an attribute should be updated, and the amount that the attribute is updated by.

Goals are used to define objectives for agents in the simulation. In order to keep goals as generic possible, each goal acts as a container for a number of conditions and satisfiers. Conditions are used to determine whether or not the goal should become active. This allows users to specify when an agent should attempt to satisfy a specific goal. Since it is not always possible to satisfy multiple goals simultaneously, goal priorities are needed to determine the order in which goals should be satisfied. If more than one goal is active, the goal with the highest priority will be satisfied first. Goal priorities are assigned by the user when the simulation is configured. Goal satisfiers are used to perform two actions. First, they are used to determine when a goal has been satisfied. Second, they are used to instruct the agent how to satisfy a goal. The most common method for satisfying a goal is to have the agent navigate to a specific location in the environment. For example, evacuation goals require agents to leave the building, whereas satisfying an agent's hunger is accomplished by moving to a food stall to purchase food.

Generality was an important design goal while designing the crowd simulator. One of the methods used to accomplish this is to make use of a generic `ICondition` class. All condition instances include a reference to the controlling `Simulation` object, which provides full, read-only access to information regarding the simulation. This information includes agent properties, environment layout and information regarding world events.

Through the use of generic conditions, world events, attributes and goals, it is possible to simulate a wide variety of crowd behaviour. The next section will examine how agents plan a path through the virtual environment in order to reach their goal.

Path Planning. Once an agent has determined its goal, the next step is to calculate a path from its current location to its goal. Path planning is the process of finding a series of points that the agent can follow in order to move from its current location to its goal. The path planner calculates a path through the environment, avoiding environment objects and boundaries. Path planning is based on a combination of two techniques: cell and portal graphs [13] and navigation meshes [15], illustrated in Fig. 1.

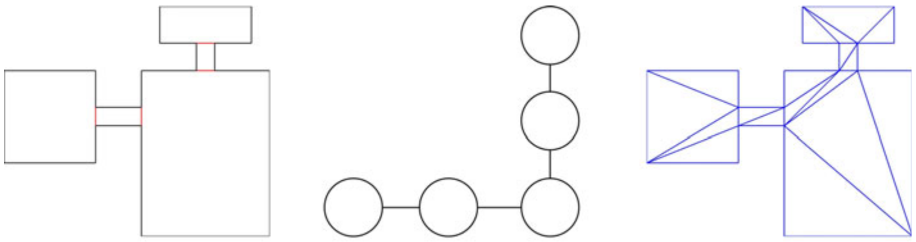


Fig. 1. Original floor plan, cell and portal graph and navigation mesh

Cell and portal graphs provide a simple, easy to understand abstraction of the environment, where each node of the graph represents a room, and an edge between nodes indicates a door connecting the two rooms. However, cell and portal graphs do not provide enough information to navigate effectively.

Navigation meshes [15] provide a high level of detail that may be used to determine a path accurately. Cell and portal graphs are used to provide extra information about the environment, based on properties that were assigned in the editor; navigation meshes are used purely for navigation. Combining these two methods allows each agent to have detailed navigation information, as well as information about the area that the agent is traveling through.

A cell and portal graph is automatically constructed from an environment created in the environment editor. A navigation mesh is constructed by performing a constrained Delaunay triangulation [2] on a cell and portal graph, using the boundaries and the edges of environmental objects' bounding boxes as constraints. The mesh is stored in a graph, where each node represents a triangle, and an edge between nodes indicates that the triangles share a common edge. The cell and portal graph and the associated navigation mesh are constructed as a pre-processing step; they are not reconstructed during the course of the simulation.

The path planner operates by performing an A* search on the navigation mesh to find the series of triangles that must be traversed to reach the goal. The heuristic function used by the A* search is the straight-line distance from the centroid of the triangle (since each node represents a triangle) to the goal point. The cost function is the same as the heuristic function, taking the distance between the centroids of adjacent triangles. From this triangle path, it is a trivial matter to extract a path of points in 2-dimensional space. The path that is returned will be suboptimal due to the number of unnecessary points contained. To address this issue, a smoothing algorithm is used to optimise the path. While an agent is moving, it constantly looks ahead to the second point in its path. If there is no obstruction, the first point can be safely removed, and the second point can be moved to the front of the path. This process is illustrated in Fig. 2.

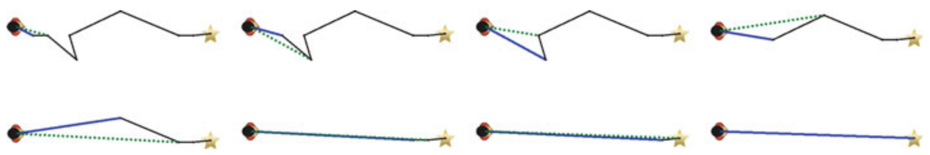


Fig. 2. Path smoothing algorithm

The path planner is only responsible for finding a path through the environment; the following section discusses the movement model used to advance agents along this path.

Movement. Agent movement is based on the social forces model of Helbing et al. [6]. Motivations and influences on agents are modelled as physical forces, which are ultimately used to calculate the resulting acceleration of an agent. It is important to keep in mind that social forces do not represent actual, physical forces. However, these forces influence the movement of an agent in exactly the same way. That is, an agent moves as if there were forces acting on it.

Each agent i has a desire to adapt its actual velocity, $\boldsymbol{v}_i(t)$, to some desired direction $\boldsymbol{e}_i^0(t)$ and speed $v_i^0(t)$ within a certain time τ_i. The force describing this desire is given by the equation

$$\boldsymbol{f}_i^{\text{desired}}(t) = \frac{v_i^0(t)\boldsymbol{e}_i^0(t) - \boldsymbol{v}_i(t)}{\tau_i} \ . \tag{1}$$

Each agent also tries to keep a certain minimum distance from other agents. This is given by the repulsive social force equation

$$\boldsymbol{f}_{ij}^{\text{social}}(t) = \begin{cases} A_i \text{exp}\left[\frac{r_{ij}-d_{ij}}{B_i}\right] \boldsymbol{n}_{ij} & \text{if } \theta_{ij} \leq \frac{\alpha_i}{2} \\ A_i \text{exp}\left[\frac{r_{ij}-d_{ij}}{B_i}\right] \boldsymbol{n}_{ij} \times C_i & \text{if } \theta_{ij} > \frac{\alpha_i}{2} \end{cases} \ . \tag{2}$$

A_i and B_i are constants representing the interaction strength and range of repulsive interactions, respectively, r_{ij} is the sum of the radii of agents i and j, d_{ij} is the distance between their centers and

$$\boldsymbol{n}_{ij}(t) = \frac{\boldsymbol{x}_i(t) - \boldsymbol{x}_j(t)}{d_{ij}(t)} \tag{3}$$

is the normalised vector pointing from agent j to i. In general, people are more influenced by obstructions and people in front of them than those to the sides or behind them. The field of view α_i defines a cone-shaped area in front of the agent. Any agent outside of this cone exerts a force with a weight of C_i where $0 < C_i < 1$. The angle θ_{ij} is defined as the angle between $\boldsymbol{e}_i(t) = \boldsymbol{v}_i(t)/\left\|\boldsymbol{v}_i(t)\right\|$, the direction of motion, and $-\boldsymbol{n}_{ij}$, the direction of the agent exerting the repulsive force.

Secondly, although agents try to keep a certain minimum distance, there are situations when this is no longer possible and agents will be forced into physical contact. Examples of this include evacuation scenarios and heavily congested areas, such as crowds exiting a concert hall. This physical interaction is modelled by the equation

$$\boldsymbol{f}_{ij}^{\text{physical}}(t) = k\Theta(r_{ij} - d_{ij})\boldsymbol{n}_{ij} + \kappa\Theta(r_{ij} - d_{ij})\Delta v_{ji}^{t}\boldsymbol{t}_{ij} \ . \tag{4}$$

k and κ are constants representing the repulsive force of a body and force of sliding friction, respectively, and Θ is a function, where $\Theta(x) = x$ if $x \geq 0$, otherwise 0. $\boldsymbol{t}_{ij}$ is the direction tangential to $\boldsymbol{e}_i(t)$ and $\Delta v_{ji}^{t} = (\boldsymbol{v}_j - \boldsymbol{v}_i) \cdot \boldsymbol{t}_{ij}$ is the tangential velocity difference between agents i and j..

Finally, agents avoid collisions with boundaries and environment objects. Since environment objects are represented using bounding boxes, they can be treated in exactly the same way as boundary walls. The force modelling the interaction with obstructions, similar to Eq. 2, is given below:

$$\boldsymbol{f}_{io}^{\text{obstruction}}(t) = \left(A_i \exp\left[\frac{r_i - d_{io}}{B_i}\right] + k\Theta(r_i - d_{io})\right)\boldsymbol{n}_{io} - \kappa\Theta(r_i - d_{io})(\boldsymbol{v}_i \cdot \boldsymbol{t}_{io})\boldsymbol{t}_{io} \ . \tag{5}$$

d_{io} is the distance to the closest point on obstruction o, $\boldsymbol{n}_{io}$ is the direction perpendicular to obstruction o, towards the agent, and $\boldsymbol{t}_{io}$ is the direction tangential to it.

In summary, the total force acting on the agent is given by the following equation:

$$\boldsymbol{f}_i(t) = \boldsymbol{f}_i^{\text{desired}}(t) + \sum_{j(\neq i)} \left[\boldsymbol{f}_{ij}^{\text{social}}(t) + \boldsymbol{f}_{ij}^{\text{physical}}(t)\right] + \sum_{o} \boldsymbol{f}_{io}^{\text{obstruction}}(t) \ . \tag{6}$$

In order to extract results for analysis, other than through an informal inspection of a running simulation, an additional piece of information is recorded during the course of a simulation: a usage density matrix. The usage density matrix is a 2-dimensional grid that records areas with a high amount of foot traffic, which is used to determine which areas of the building are the most heavily traversed. This is useful for finding areas that are under-utilised or have a high degree of congestion.

The next section discusses the implementation details of the above model, as well as some of the results that have been obtained.

4 Results

To demonstrate the proposed solution, two different scenarios were designed and simulated: a multi-level office building, illustrated in Fig. 3, and a shopping mall, illustrated in Fig. 4.

The office building was used to demonstrate an evacuation scenario where a building is fully populated at the start of simulation and each agent has only a single goal: to exit the building as quickly as possible. This simulation was also

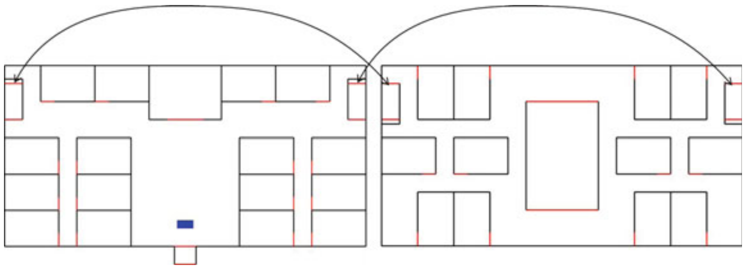


Fig. 3. Office building. Arrows show connected staircases between floors 1 and 2; the solid rectangle is a receptionist's desk.

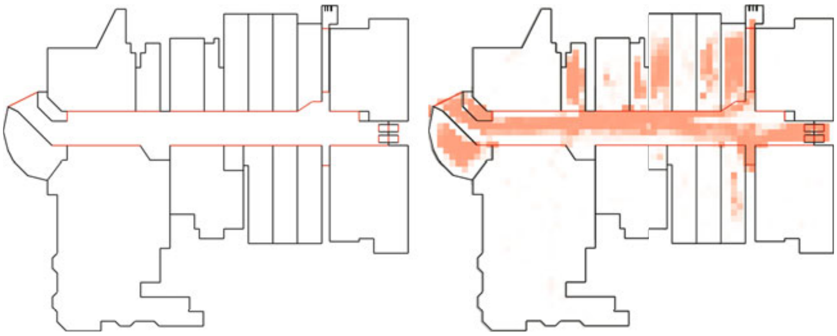


Fig. 4. Floor plan of shopping mall with resulting usage density

used to demonstrate the effect of building design on evacuation times. In one simulation, only a single exit was open, and a receptionist's desk was placed in front of the door, as illustrated in Fig. 3. A second simulation involved moving the desk to the side of the main entrance and adding a second exit at the other end of the floor. This simple change resulted in an improvement of 16.5% in the time taken to evacuate.

The shopping mall was used to demonstrate agents that enter a building during the course of simulation. This simulation serves as an analysis of usage under normal operating conditions. Each agent has a number of optional goals – such as visiting various shops and wandering around the shopping mall – and each agent will leave the mall once all of its other goals have been completed. Figure 4 provides the resulting density matrix of the shopping mall.

5 Conclusions and Future Work

This paper has presented a two-stage process for using multi-agent technology to assist in designing buildings. We have presented a general-purpose environment editor that may be used to design a large variety of building types. We have also discussed a crowd simulation model that incorporates goal-directed behaviour, high-level navigation processes and movement with collision-avoidance. This simulation model can be employed to simulate a multitude of crowd scenarios, such as general every-day usage, extreme congestion and evacuation.

The goal of the system is to use intelligent agents to assist in designing space-efficient buildings. Therefore, it is useful to know of areas where there is an abundance of space that may be safely reduced, or areas that result in bottlenecks or congestion that may need to be made larger. Simulations can help identify these areas by determining how people would use the available floor space of the building. Identified problem areas can be quickly rectified through the use of the integrated environment editor.

Combining an environment editor with a crowd simulation increases the amount of customisation options that are available to define the behaviour of agents. Almost every aspect of an environment can be given data tags that may be used as part of the crowd simulation.

Although the current system is capable of producing useful results, there are several areas that could benefit from future work.

One possible augmentation for the environment editor is to support automatic generation of building plans. This would allow designers to generate an initial building layout from user-specified details – such as the perimeter and the number and type of rooms needed. The generation algorithm should be able to learn from existing designs, to ensure that generated layouts are realistic. However, the generator is not meant to replace an architect. Rather, it is meant to complement and assist with building design.

The crowd simulator was designed to be as general-purpose as possible. However, by increasing the generality of the simulation, the knowledge required to configure a simulation and the amount of time required to do so is also increased. Currently, the user is responsible for assigning goals, determining their priority, as well as creating attributes and defining how they are updated over time. For complex simulations, this can be a time-consuming process. New methods of configuration, or research into behavioural templates, would simplify the process greatly and reduce the time taken to configure a simulation.

Acknowledgement. The financial assistance of the National Research Foundation (NRF) towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attributed to the NRF.

References

1. Chenney, S.: Flow tiles. In: Proceedings of the 2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation, pp. 233–242 (2004)
2. Chew, L.P.: Constrained delaunay triangulations. In: Proceedings of the Third Annual Symposium on Computational Geometry, SCG 1987, pp. 215–222. ACM, New York (1987)
3. Dijkstra, J., Jessurun, J., Timmermans, H.J.P.: A multi-agent cellular automata model of pedestrian movement. In: Pedestrian and Evacuation Dynamics, pp. 173–181 (2001)

4. Gangolells, M., Casals, M., Gasso, S., Forcada, N., Roca, X., Fuertes, A.: A methodology for predicting the severity of environmental impacts related to the construction process of residential buildings. Building and Environment 44(3), 558–571 (2009)
5. Gangolells Solanellas, M., Casals Casanova, M., Gassó Domingo, S., Forcada Matheu, N., Roca Ramon, X., Fuertes Casals, A.: Identifying potential environmental impacts at the pre-construction stage. Development (2009)
6. Helbing, D., Farkas, I.J., Molnár, P., Vicsek, T.: Simulation of pedestrian crowds in normal and evacuation situations. In: Pedestrian and Evacuation Dynamics, pp. 21–58 (2002)
7. Helbing, D., Molnár, P.: Social force model for pedestrian dynamics. Physical Review E 51(5), 4282–4286 (1995)
8. Hughes, R.L.: A continuum theory for the flow of pedestrians. Transportation Research Part B: Methodological 36(6), 507–535 (2002)
9. International Alliance for Interoperability: An Introduction to the International Alliance for Interoperability and the Industry Foundation Classes. Tech. rep., International Alliance for Interoperability (1999)
10. Johnston, A., Hutchison, J., Smith, A.: Significant environmental impact evaluation: a proposed methodology. Eco-Management and Auditing 7(4), 186–195 (2000)
11. Lingjiang, H.: Technology in Computer Aided Architectural Design. In: Second International Conference on Information and Computing Science, ICIC 2009, vol. 2, pp. 221–223 (May 2009)
12. Noth, M., Borning, A.: An extensible, modular architecture for simulating urban development, transportation, and environmental impacts. Computers, Environment and Urban Systems 27(2), 181–203 (2003)
13. Pelechano, N., Allbeck, J.M., Badler, N.I.: Virtual Crowds: Methods, Simulation, and Control. Morgan & Claypool (2008)
14. Reynolds, C.: Flocks, herds and schools: A distributed behavioral model. Computer Graphics 21(4), 25–34 (1987)
15. Snook, G.: Simplified 3d movement and pathfinding using navigation meshes. In: Game Programming Gems, vol. 1, pp. 288–304. Charles River Media (2000)
16. Treuille, A., Cooper, S., Popović, Z.: Continuum crowds. ACM Trans. Graph. 25(3), 1160–1168 (2006)
17. U.S. Census Bureau: World POPClock Projection (2011), `http://www.census.gov/ipc/www/popclockworld.html`, (accessed June 13, 2011)
18. Van den Berg, J., Lin, M., Manocha, D.: Reciprocal velocity obstacles for real-time multi-agent navigation. In: IEEE International Conference on Robotics and Automation, pp. 1928–1935 (2008)

An Investigation of Emergent Collaboration under Uncertainty and Minimal Information in Energy Domains[⋆]

Radu-Casian Mihailescu, Matteo Vasirani, and Sascha Ossowski

Rey Juan Carlos University,
Centre for Intelligent Information Technologies,
Madrid, Spain
{raducasian.mihailescu,matteo.vasirani,sascha.ossowski}@urjc.es

Abstract. We study the phenomenon of evolution of cooperation in the electricity domain, where self-interested agents representing distributed energy resources (DERs) strategize for maximizing payoff. From the system's viewpoint cooperation represents a solution capable to cope with the increasing complexity, generated by the introduction of DERs to the grid. The problem domain is modelled from a multi-agent system high-level perspective. We report on experiments with this model, giving the underlying understanding for the emergent behavior, in order to determine if and under what conditions such a collaborative behavior would hold. Finally we suggest how insights from this model can inspire mechanisms to instill cooperation as the dominant strategy.

1 Introduction

Conventional methods for energy generation, transmission and distribution are about to experience a radical change. This is on one hand due to the ever increasing demand in energy consumption (e.g. electric vehicles) and secondly, due to the proliferation of distributed generators (e.g. renewable energy) to be connected to the grid. Current power networks will no longer be able to provide the required level of reliability and robustness and thus there is a need for a more flexible connection and management of the system.

Various approaches for the advent of agent technologies to this domain have been proposed thus far, that range from micro-grid architectures [3,10], demand-side management [6,16] and micro-storage solutions [17] to plug-in hybrid vehicles coordination [8]. The benefits of applying the multi-agent systems paradigm as an approach for distributed control of the Grid entails primarily: autonomy, scalability, flexibility, extensibility, fault tolerance and reduced maintenance [10]. The actors existing in the grid (i.e. consumer loads, distributed generators) represent different owners with particular user behaviors, hence deploying an agent-based distributed control over the system becomes highly suitable for this

⋆ This work was supported by the projects AT (CONSOLIDER CSD2007-0022, INGENIO 2010) and OVAMAH (TIN2009-13839-C03-02).

scenario. Moreover, decentralization increases the system's reliability in case of failures, enables local adaptability to dynamic situations at runtime and allows coordination as opposed to the more complex task of centralised management.

The outline for the rest of the paper is as follows. In Section 2 we review related work in this area. Section 3 describes the agent-based framework used for modelling the problem domain. Then, in Section 4 we discuss implementation details and outline a series of experiments that exhibit the phenomenon of emergent cooperation. Section 5 suggests directions for future research and concludes the paper.

2 Background

Generically, these intelligent electricity network technologies support the vision of a *Smart Grid* that aims at reducing the carbon footprint, while increasing energy efficiency and clean energy usage. A key approach in doing so, focuses on decreasing the high energy costs and emissions during peak demand periods by means of intelligently coordinating the variable output of wind or solar energy generators. In particular the cooperative concept of virtual power plants has been advocated as a viable organizational model [2,11], from the grid operator's viewpoint, allowing to cope efficiently with the integration of the many distributed energy resources.

In [2], the authors start from the assumption of an existing VPP and propose a novel pricing mechanism, that addresses the question of allocating payoff amongst the VPP members, so that it can guarantee that no subset of agents has an incentive to break away. Additionally, the payoff scheme elicits truthful behavior from the VPP members, while their production estimates are evaluated by means of statistical methods. Similarly, the PowerMatcher described in [9] is a market-based multi-agent tree-architecture for balancing supply and demand within clusters of DERs.

From an organizational perspective earlier works [13,14] suggested centralized structures for managing the VPP, that come short in addressing an open system setting and the inherent stochastic nature of DERs. These issues are captured in [12], where a dynamic coalition formation mechanism is proposed for the creation of VPPs. The authors consider a cooperative scenario and introduce a decentralized algorithm according to which, agents efficiently self-organizing into coalitions representing the actual VPPs.

Alternatively, in this work we take a different perspective and question the emergence of such a phenomenon of evolution of cooperation itself in a population of self-interested agents. We model the problem as a repeated game and conduct an analysis in order to determine the context under which collaborative behavior could result. Gaining an understanding of what drives cooperation under the assumption of rational agents is particularly important in designing system-level mechanisms and policies that could incentivize efficient resource allocation.

The emergence of cooperation amongst self-interested agents has received a lot of attention from various research areas including social sciences, behavioral economics or evolutionary biology [1,15]. Several notable efforts have looked at different variations of the Prisoner's Dilemma game[1] and studied the influences of parameters such as underlying network topology, interaction rules or updated rules [18,7]. In this work we take a different outlook on this issue and address the problem of stochastic environments, specifically represented here by the electricity domain, showing how collaborative behavior can emerge as an adaptive strategy for handling uncertainty.

3 Agent-Based Model

Given that the penetration of distributed energy resources (DERs) is expected to increase significantly [4], integrating these devices to the grid poses difficult challenges. As a solution for reducing the complexity of managing the system at large, the aggregation of DERs as virtual power plants (VPPs) has been proposed. A VPP is conceived as a bundle of distributed energy generators that are connected through an informational infrastructure and act in a coordinated way as a single entity, being represented as one resource to the system operator. Generally, they represent renewable energy resources such as wind or solar power, which accounts for high variability depending on environmental conditions.

Now, considering the distributed nature of DERs and their selfishly driven behavior it comes natural representing them as autonomous agents interacting in an open an highly dynamic environment. Also we consider agents to be controlling identical DERs in terms of their capacity profile. Agents are thus interested in maximizing the payoff obtained by selling their available energy. There are several day-ahead power markets where agents may choose to bid: (i) *Baseload Power Market*: typically this power is currently provided by large-scale power plants round-the-clock and at low costs per kWh; (ii) *Peak Power Market*: this represents the additional power necessary during high-demand intervals of time; (iii) *Spinning Reserves Market*: designed for ensuring the reliability of the grid in case of transmission line failures or similar contingencies, in practice they are rarely used but are being payed for the duration they are available; (iv) *Regulation Markets*: required in order to regulate the frequency and voltage in the grid, must be capable to respond to frequent real-time signals from the grid operator.

Taking into account the profile of DERs, in our model we consider for the agents the options of participating in the either the *Baseload* or *Peak Power Market*, reflecting two opposing strategies, which denote exposure to risk. There is clearly an uncertainty regarding energy availability for the following forecasted day. On one hand selecting to bid in the *Baseload Market* ensures a lower profit (lower kWh rates) regardless of the amount of energy provided, whilst

[1] Well-known two-player game-theoretic framework where agents have to chose between two strategies: cooperating or defecting. While defecting is the dominant strategy and the only Nash Equilibrium, it is also Pareto-inefficient as cooperation would make both players better off.

the *Peak Power Market* would guarantee higher profits (during high-demand intervals) given sufficient energy availability. Thus, the former option represents risk-aversion, as the latter denotes a risk-seeking behavior.

Secondly, agents need to decide whether they prefer cooperation, which takes the form of a VPP, where agents reallocate energy in order to mitigate day-ahead predictions and fulfill bids and thus maximize profit or rather, choose non-cooperative behavior.

Therefore, essentially we modelled a game where the action space for each agent is a two dimensional binary state space $\mathcal{A} = s_1 \times s_2$, characterizing willingness to cooperate as the strategy set $s_1 = \{$*cooperate(1), defect(-1)*$\}$ and aversion to risk by $s_2 = \{$*risk-seeking(1), risk-averse(-1)*$\}$. Moreover, the stochastic and dynamic nature of the system is due to the uncertainty of available energy and the varying number of agents participating in the system at each iteration of the game. Specifically, the n-player game proceeds in rounds, each consisting of two phases: (i) the game playing phase, where the payoffs are computed for each agent based on their particular choice of strategies and (ii) the strategy update phase.

Let $\mathcal{P}_1$ represent for each agent the probability for choosing cooperation, whilst the complementary probability corresponds to defecting. Similarly, we associate $\mathcal{P}_2$ with the probability of risk-seeking behavior and $1 - \mathcal{P}_2$ with risk-aversion. The update rule computes new probabilities for the next iteration for each agent a_i of the set of all agents $\mathcal{N}$ based upon previous round results, by applying a *satisfying gradient ascent rule*:

$$\mathcal{P}_i^{t+1} \leftarrow \mathcal{P}_i^t + \alpha \cdot s_i^t \cdot (\mathcal{R}_t(\bar{\mathcal{A}}_{a_i}) / \operatorname{argmax}_{a_j \in \mathcal{N}} \mathcal{R}_t(\bar{\mathcal{A}}_{a_j}) - \varepsilon)$$

where $\mathcal{P}_i^t$ is the current probability associated with strategy s_i; $\mathcal{P}_i^{t+1}$ is the probability associated with strategy s_i for the following iteration; α represents the learning rate; $\mathcal{R}_t(\bar{\mathcal{A}}_{a_j})$ is the payoff of agent a_j at iteration t corresponding to its chosen strategy set $\bar{\mathcal{A}}_{a_j}$. Intuitively, the probability of a particular strategy is updated according to the percent payoff difference between the agent's current strategy and the best performing strategy in the system, minus the margin acceptability ε. Parameter ε represents the trade-off between the payoff difference and the probability to switch.

Now, assuming our previous considerations about participation in different energy markets, we represent the agents payoff as a function of the energy that the agents are able to provide, which is in turn the stochastic variable of the system θ. Specifically, by adopting a risk-averse strategy the payoff (refered in terms of kWh rates) is moderate, being less dependent on the amount of energy availability. On the contrary a risk-seeking strategy would return a considerably higher payoff in case of high energy availability but, a significantly lower one otherwise. For the sake of simplicity the abstracted payoff functions for the two strategies are empirically defined to match the above description representing a close-to-flat rate for the former strategy, as for the latter generating twice the profit for large values of θ and close to zero otherwise (as depicted in Figure 2). Also important to remark is that in the vicinity of the intersection of these two functions the uncertainty for choosing strategy s_2 is highest.

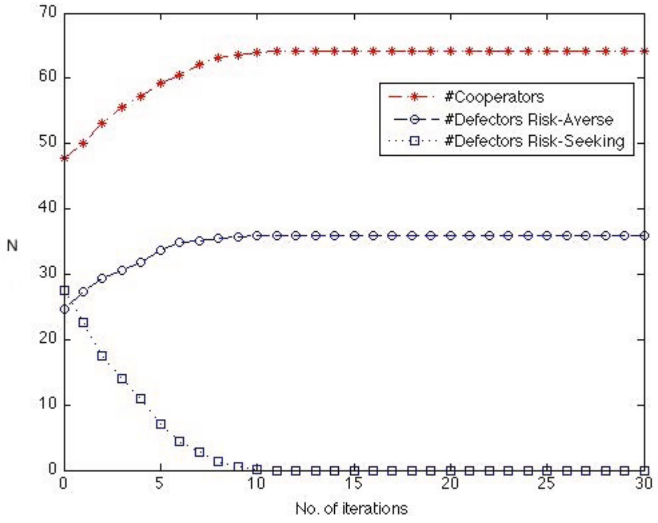


Fig. 1. Evolution of the number of agents with respect to their strategy space for θ=5

4 Simulation Results

4.1 Complete Information Games

The purpose of our experiments is analysing the conditions under which there is evolution of cooperation for our domain specific setting and understand whether the system-level goals and those of our rational agent population align or on the contrary, they are conflicting.

We start under the assumption of complete information, where all aspects of the game are considered common knowledge for all players and the payoffs of the other agents are directly observable. We simulated a system consisting of a population of at most 100 agents. The initial probabilities, based on which agents are determining their strategies are allocated randomly with uniform probability. The variability of the environment is represented as the stochastic value θ, which determines the amount of energy available for each agent. The simulation then proceeds for each round according with the two phases previously described in Section 3.

What the experiments show is that for each θ value the system converges to stable configurations, in terms of the ratios with which strategies are selected. Convergence is reached in approximately 20 rounds. Figure 1 highlights for a population of 100 agents and a particular value of θ the fraction of agents that have selected to cooperate and for those that have defected, their strategy regarding risk. We take this last stationary state of the system as the final result. Following, we plot this data in Figure 2, representing agents' payoff as a function of θ, by averaging each data point over 1000 runs.

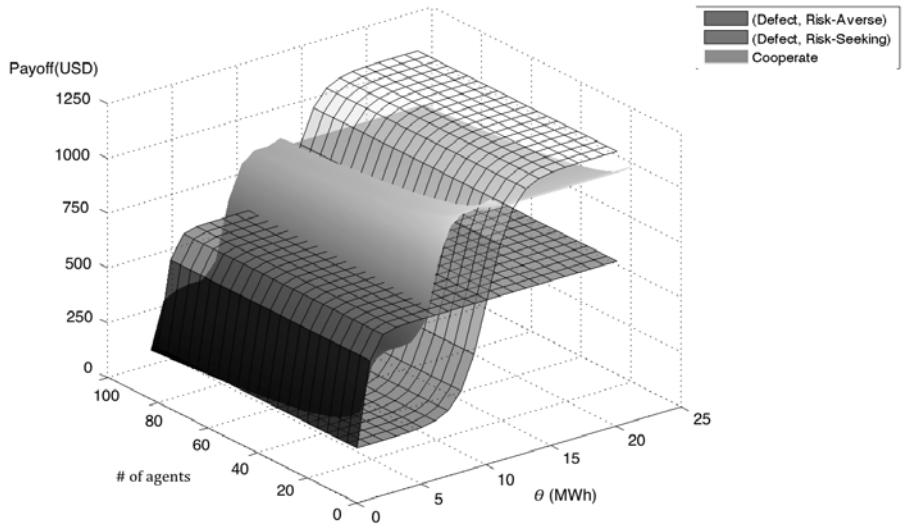


Fig. 2. Payoffs as a function of variable energy availability θ and number of agents in the system

It appears that a cooperative strategy yields the highest payoff for the agents in a situation where neither risk-seeking nor risk-averse strategy is clearly dominant. Thus, in case of uncertainty cooperation proves to be the optimal choice, accounting for a minimization of risk. However, when there is a higher level of certainty with regard to strategy s_2 for selecting a suitable energy market, defecting outperforms cooperation. The underlying reason for this counter-intuitive result is the fact that the reallocation of energy between cooperative agents produces also better results for the agents with an incorrect choice for strategy s_2, than would defecting. This misguided feedback is causing them to react suboptimal, and thus, in detriment of the coalition at large. In comparison defecting agents show better adaptability.

Moreover, we take into account an open system scenario where the number of agents is varying. In Figure 2 the y-axis shows how the number of agents is influencing the outcome of the game. Consistent with the abovementioned explanation, in conditions of higher certainty about risk, a lesser number of cooperating agents are capable to better adapt their strategies, rather than a larger coalition. Therefore we can conclude that the limitation of the number of cooperating agents proves here to represent a solution for promoting cooperation as a dominant strategy. This would allow for members of a coalition to better respond to variations and adapt their respective strategies, while maintaining a dynamic equilibrium with the environmental changes.

4.2 Minimal Information Environments

In this section we set to investigate the emergence of cooperation for games played with minimal information, wherein the global knowledge assumption of observable payoffs for all players (as opposed to complete information games, see Section 4.1) does no longer hold.

Thus, we consider the agents representing DERs deployed over a spatial distribution and restrict observability to their vicinity. Formally, agents are connected via an underlying network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, denoted as an undirected graph, where the set of vertices $\mathcal{V}$ is the set of agents and edges $\mathcal{E}$ are the set of links in $\mathcal{G}$, which connect the agents in $\mathcal{V}$. An agent a_i may only directly observe agent's a_j payoff if they are neighbours. $\mathcal{N}_{a_i}$ represents the set of a_i's neighbours, where $\mathcal{N}_{a_i} = \{a_j | (a_i, a_j) \in \mathcal{E}\} \subset \mathcal{V}$.

For generating the random graph structure we have used the model proposed in [5], where undirected edges are placed randomly between a fixed number of vertices, resulting in a network where each possible edge is present with equal probability p. Similarly to our previous experimental setting, we consider a population of $n = 100$ agents. We set $p = log(n)/n$ to ensure the graph connectedness and an average connectivity per node equal to 4, corresponding to topological configurations for generic meshed suburban network models[2].

Strategy Selection and Convergence. The strategy selection rule determines which strategy to play from the agent's strategy space. Considering the given minimal information scenario, we use the following types of strategy selection rules:

- *The gradient ascent strategy rule* performs an identical estimation to the one detailed in Section 3, revising its strategy selection probabilities $\mathcal{P}_i$ according to the reward gradient, with one important difference. Namely, each agent can only perceive the local highest payoff in its vicinity, as opposed to the global highest payoff in the system:

  $$\mathcal{P}_i^{t+1} \leftarrow \mathcal{P}_i^t + \alpha \cdot s_i^t \cdot (\mathcal{R}_t(\bar{\mathcal{A}}_{a_i}) / \underset{a_j \in \mathcal{N}_{a_i}}{\operatorname{argmax}} \mathcal{R}_t S(\bar{\mathcal{A}}_{a_j}) - \varepsilon)$$

- *Win-stay, lose-shift rule* maintains the current strategy selection probabilities only if the current payoff is at least as high as in the previous iteration round. Otherwise, revises $\mathcal{P}_i$ proportional to the difference of its current and last payoff:

  $$\varDelta\mathcal{R}_t = \mathcal{R}_t - \mathcal{R}_{t-1};\ \mathcal{P}_i^{t+1} \leftarrow \mathcal{P}_i^t + \alpha \cdot s_i^t \cdot \varDelta\mathcal{R}_t$$

  This approach is highly suitable for minimal information environments as it only requires keeping track of short-term previous payoffs.

- *Imitate best strategy rule* identifies the agent with the highest payoff in its neighbourhood and adopts the same probabilities for strategy selection.

  $$\mathcal{P}_i^{t+1}(a_i) \leftarrow \mathcal{P}_i^t(a_j), \text{ where } \mathcal{R}_{t-1}(\bar{\mathcal{A}}_{a_j}) = \underset{a_k \in \mathcal{N}_{a_i}}{\operatorname{argmax}} \mathcal{R}_{t-1} S(\bar{\mathcal{A}}_{a_k})$$

[2] Identified as the most suitable setting for the deployment of medium-scale VPPs, thus the particular spatial distribution considered.

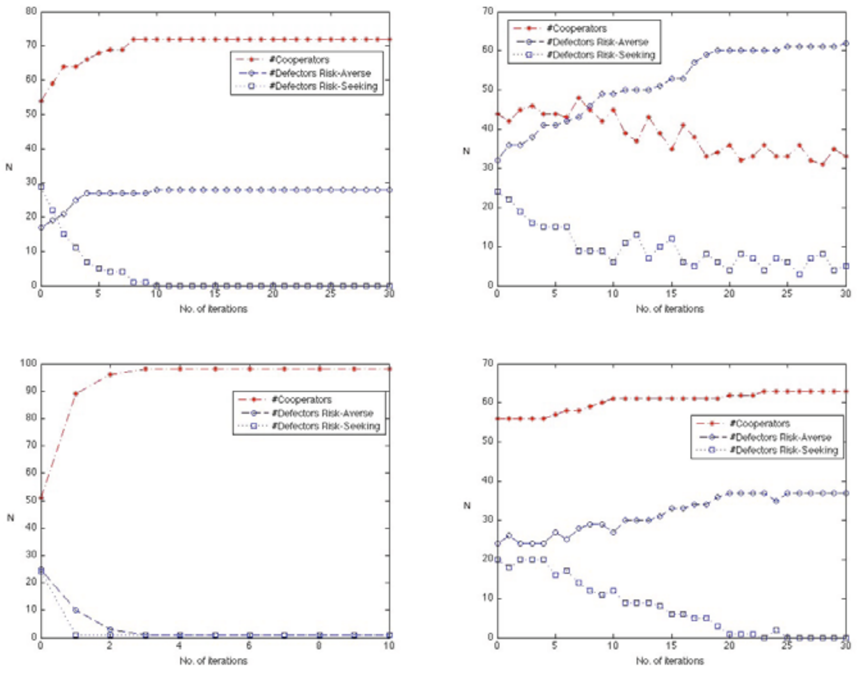


Fig. 3. The evolution of the agents behavior with respect to the strategy selection rule for θ=9. a) *gradient ascent* ; b) *win-stay, lose-shift* ; c) *imitate best strategy* ; d) *regret minimization* ;

- *Regret minimization strategy rule* consists of playing the strategy that maximizes its minimum possible expected payoff.

$$\mathcal{P}_i^{t+1} \leftarrow \mathcal{P}_i^t + \alpha \cdot s_i^t \cdot \min_{(s_1,s_2)} \max_{\theta}(\mathcal{R}_{t-1} - \mathcal{R}_t)$$

Evaluation. In the previous section we ran our simulation under the assumption of global system knowledge. However, in a practical scenario, it is more likely that agents are capable to acquire only partial knowledge of the system. With these considerations in mind we deploy agents over a spatial distribution and investigate to what extend this may affect results. Moreover we derived several strategy selection rules for minimal information environments and performed a comparative analysis. The value used for θ was chosen to reflect highly uncertain scenarios where cooperation is ought to emerge as the dominant strategy.

It is interesting to observe from the plots in Figure 3 that the system converges to similar results, obtained for games with minimal information. Particularly, the *gradient ascent* strategy attains just about the same stationary states in terms of the fraction of cooperating agents via local estimations only, though convergence occurs after more iterations of the game.

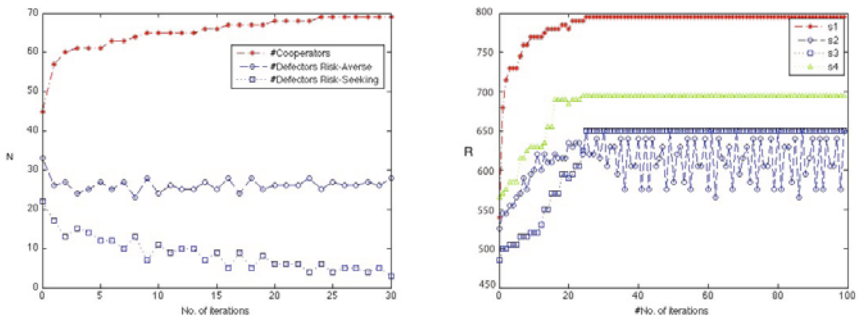


Fig. 4. a) Population share in mix-strategy repeated games b) Average payoffs in repeated games

For the *win-stay, lose-shift rule* we find a lower fraction of cooperators, denoting a lower efficiency for adopting the optimal strategy. Here strategy selection solely relies on the agent's own past results. This enables applying it to settings where payoffs of other agents are unobservable, while still producing satisfactory outcomes. We found that the *win-stay, lose-shift rule* was outperformed by the *gradient ascent* strategy by a margin of approximately 20%.

In contrast, the *imitate best strategy rule* proves a rapid convergence to optimal for the majority of the agent population. However, note that the success of this rule depends on the agents' capability of perceiving the internal states of neighbouring agents for copying their probabilities of strategy selection, which may not always hold as a reasonable assumption.

Finally, the *regret minimization strategy rule* shows a foreseeable result. Given the highly uncertain conditions chosen for this experiment, the number of agents that adopt a *risk-seeking* behavior is dramatically reduced, as *risk-averse defectors* and *cooperators* become majoritarian.

Also, it is interesting to analyse the impact on performance considering a heterogeneous population of agents. We conducted experiments for mix-strategy repeated games, by having each of the previously detailed strategy selection rules equally represented in the population. Similarly, we evaluated the percentage of the population that converged to optimal behavior for high uncertainty conditions. Figure 4 a) shows that a heterogeneous population of agents achieves almost about the same level of cooperation for minimal information domains as opposed to complete information scenarios. Additionally, it is interesting to see which strategies have been the top performers of the game in terms of the payoff obtained against the different types of participating agents. As depicted in Figure 4 b), the highest average payoff after 50 rounds of the game was achieved by the *imitate best strategy rule*, where all agents employing it converged to the highest payoff possible (which was also the case for the self play game). The *win-stay, lose-shift rule* population share returned the second highest payoff, while the *gradient ascent* and *regret minimization strategy rules* were respectively significantly outperformed. These results suggest that even in minimal information settings agents can learn optimal behavior under the more strong assumption of local complete information, while moderate performance can still be achieved relying solely on the agent's own past results.

5 Conclusion and Future Work

In this paper we have described an agent-based model for a smart electricity grid that assumes the presence of large number of distributed energy resources in the system. A novel mechanism aimed to cope with the increasing complexity of the system proposes an organization of DERs in the form of VPPs. In this work we studied the phenomenon from the perspective of self-interested agents, that look to maximize their payoff, in order to determine if and under what conditions such a collaborative behavior might emerge.

Experiments have shown that cooperation amongst rational agents is an emergent phenomenon in this setting for a large fraction of the agent population, including for games played with minimal information. In fact cooperation is the optimal strategy in situations of high uncertainty, where agents adopt it as an adaptation mechanism to variable environmental conditions. However, when uncertainty is decreased defecting may produce better results for good predicting agents. Whereas collaboration becomes more susceptible to suboptimal gains as some cooperating agents may still return acceptable payoffs although selecting suboptimal strategies, due to the redistribution of available energy. Therefore, in such instances, in order to instill cooperation as the dominant strategy further mechanism need to be implemented at the coalition level in order to ensure optimality.

As future work we plan to extend the model, in order to relax some of the underlying assumptions. In particular we intend to experiment with a heterogeneous population of agents, in the sense of DERs' output capacity and expand the strategy space in order to reflect more accurately realistic scenarios. Also we are interested in studying large-scale scenarios where not only the grand coalition would form, but also agents could reason about joining different competing coalitions and understand the type of emergent equilibrium that results from their interactions.

References

1. Axelrod, R.: The Evolution of Cooperation. Basic Books, New York (1984)
2. Chalkiadakis, G., Robu, V., Kota, R., Rogers, A., Jennings, N.: Cooperatives of distributed energy resources for efficient virtual power plants. In: Proc. of the Tenth International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), pp. 787–794 (May 2011)

3. Dimeas, A.L., Hatziargyriou, N.D.: Operation of a multiagent system for microgrid control. IEEE Transactions on Power Systems 20(3), 1447–1455 (2005)
4. Department of Energy and Climate Change: Smarter grids: The opportunity (2009), http://www.decc.gov.uk/assets/decc/smartergridsopportunity.pdf
5. Erdos, P., Renyi, A.: On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci. 5, 17–61 (1960)
6. Hammerstrom, J., Brous, J., Chassin, D.P., Horst, G.R., Kajfasz, R., Michie, P., Oliver, T.V., Carlon, T.A., Eustis, C., Jarvegren, O.M., Marek, W., Munson, R.L., Pratt, R.G.: Pacific northwest gridwise testbed demonstration projects part ii grid friendly appliance project. Tech. rep., Pacific Northwest National Laboratory, Richland, WA, tech. Rep. PNNL-17079 (October 2007), http://gridwise.pnnl.gov/docs/op_project_final_report_pnnl17167.pdf
7. Hofmann, L.M., Chakraborty, N., Sycara, K.: The evolution of cooperation in self-interested agent societies: A critical study. In: Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2011, pp. 685–692. International Foundation for Autonomous Agents and Multiagent Systems (2011)
8. Kamboj, S., Kempton, W., Decker, K.S.: Deploying power grid-integrated electric vehicles as a multi-agent system. In: Proceedings of the Tenth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011), pp. 13–20 (May 2011)
9. Kok, J.K., Warmer, C.J., Kamphuis, I.G.: Powermatcher: multiagent control in the electricity infrastructure. In: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2005, pp. 75–82. ACM, New York (2005)
10. Mcarthur, S.D.J., Davidson, E.M., Catterson, V.M., Dimeas, A.L., Hatziargyriou, N.D., Ponci, F., Funabashi, T.: Multi-Agent Systems for Power Engineering Applications Part I: Concepts, Approaches and Technical Challenges. IEEE Transactions on Power Systems 22(4), 1743–1752 (2007)
11. Mihailescu, R.C., Vasirani, M., Ossowski, S.: Dynamic coalition formation and adaptation for virtual power stations in smart grids. In: Proc. of the 2nd Int. Workshop on Agent Technologies for Energy Systems, pp. 85–88 (2011)
12. Mihailescu, R.-C., Vasirani, M., Ossowski, S.: An organizational approach to agent-based virtual power stations via coalitional games. In: Proceedings of International Conference on Practical Applications of Agents and Multi-Agent Systems, Salamanca, Spain, pp. 125–134 (April 2011)
13. Pielke, M., Troschel, M., Kurrat, M., Appelrath, H.J.: Operation strategies to integrate chp micro units in domestic appliances into the public power supply. In: Proceedings of the VDE- Kongress (2008)
14. Pudjianto, D., Ramsay, C., Strbac, G.: Virtual power plant and system integration of distributed energy resources. IET Renewable Power Generation 1(1), 10–16 (2007)
15. Santos, F.C., Pacheco, J.M., Lenaerts, T.: Evolutionary dynamics of social dilemmas in structured heterogeneous populations. Proc. Natl. Acad. Sci. USA 103, 3490–3494 (2006)
16. Trudnowski, D., Donnelly, M., Lightner, E.: Power-system frequency and stability control using decentralized intelligent loads. In: 2005/2006 PES TD Conference and Exposition, pp. 1453–1459 (2005)

17. Vytelingum, P., Voice, T.D., Ramchurn, S.D., Rogers, A., Jennings, N.R.: Agent-based micro-storage management for the smart grid. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2010, pp. 39–46. International Foundation for Autonomous Agents and Multiagent Systems (2010)
18. Zimmermann, M.G., Eguiluz, V.M.: Cooperation, social networks, and the emergence of leadership in a prisoner's dilemma with adaptive local interactions. Physical Review E (Statistical, Nonlinear, and Soft Matter Physics) 72(5), 56118 (2005)

A Simulator Integration Platform for City Simulations

Yuu Nakajima and Hiromitsu Hattori

Department of Social Informatics, Kyoto University
Yoshida-Honmachi, Sakyo-ku, Kyoto, 606-8501, Japan
{nkjm,hatto}@i.kyoto-u.ac.jp
http://www.ai.soc.i.kyoto-u.ac.jp/~nkjm

Abstract. Multiagent-based simulations are regarded as an useful technology for analyzing complex social systems and have been applied to various problems. Tackling the problems of a city involves various levels of abstraction and various target domains. Different types of human behaviors are studied separately by specialists in their respective domains. We believe that we need to integrate simulators that offer different levels of abstraction and cover various target domains. This paper introduces the architecture of a simulator integration platform and demonstrates the capability of the platform in that domains of city traffic and city electricity.

1 Introduction

Multiagent-based simulations (MABS) are regarded as the most powerful technology for analyzing complex social phenomena. Actors, who have various characteristics, play out their daily lives autonomously, and the summation of their actions produces social phenomena. The MABS approach, which observes interaction among diverse agents that model individual humans, is suitable for social simulations[4]. MABS are being applied various domains, e.g. traffic engineering, disaster management, sociology[1,6,3].

Vehicular traffic is one of the most complex systems in modern society. Comprehensive traffic simulations must involve various levels of abstraction and various target domains.

In the traffic domain, the behaviors of humans are captured at different levels of abstraction. For example, traffic simulations of a wide-area road network are based on highly abstract models such as traffic flow models[2,10], while those of a road are based on less abstract models such as driving behavior model[5,7]. Few simulators can deal with different levels of abstraction.

Additionally, traffic is related to various other city problems: evacuations after a disaster or enhancement of the spread of a flu pandemic. Traffic and other city problems impact each other, but there is no simulation platform that can combine multiple (different) simulators.

This paper has two goals:

1. Integration of simulators having different abstraction level Interpreting the activities of a city involves different levels of abstraction. For city traffic simulations, the movements of people are expressed by highly abstract models such as flow over a road network. For driving simulations at the level of individual roads, people are described in more specificity as actors who observe the environment and then decide their action. Integrating simulators with different levels of abstraction is the first issue. Our solution is the layered architecture, where each simulator shares the data of contact points and a high level simulator calls a low level simulator when it needs precise details of a particular phenomenon.
2. Integration of different domain simulators.
   Activities in a city are interpreted as different domains. In the traffic domain, people traveling to a hospital are interpreted as vehicle movements; in the flu pandemic domain, they are interpreted as possible contacts of infected and uninfected people. Since these two domains are only weakly related, we propose an external simulator integration architecture that loosely connects different simulators by establishing mutual contact points through which messages are passed.

The remainder of this paper is as follows. Section 2 describes our approach to designing the simulator integration platform. Section 3 shows how to combine simulators that have different levels of abstraction. Section 4 describes the integration of simulators in different domains: traffic simulation and electricity simulation. Section 5 provides an analysis of the platform's performance and Section 6 places the proposed method within extant research.

2 Architecture

We classify the difference between simulators from two viewpoints: abstraction and target domain. First, the simulator components, the parts of the architecture, are described. Second, the layered architecture for simulators having different abstraction level is explained. Last, the external simulator integration architecture for different domain simulators.

2.1 Simulator Components

This architecture can handle multiple simulators, each of which captures a specialized aspect of city phenomena. Each simulator is built as an independent system module (Fig. 1, Fig. 2) and includes a simulation controller. Each simulator has its own time step because its components are executed independently.

The simulation controller requests the simulation model to calculate the state of the next step. The simulation models writes its estimate of the next time step. Simulation models consist of agent models to capture human behavior and environmental models to capture city environments.

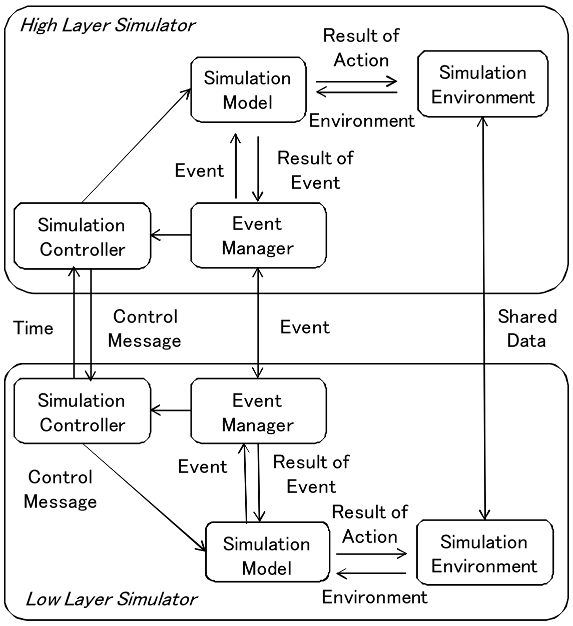


Fig. 1. Layer based integration architecture

The points connection among simulators are established by the event managers initiating and responding to event messages. When an event that involves another simulator occurs, the event is sent to the event manager of the appropriate simulator.

2.2 Layered Integration Architecture

Activities of citizens are interpreted on different levels of abstraction when they are modeled in simulations. For example, a high level of abstraction is used to examine gross movements, i.e. the morning movement of people from the suburbs into the city center. A low level of abstraction would be used to examine crowd movement in a shopping area.

Simulators that have different abstraction levels deal with the same phenomenon from different aspects. Because of this, these simulators are strongly linked to each other. We propose here a layered architecture where the higher level layer controls the lower layer and the simulators are connected by data exchanges (Fig. 1).

The simulation model in the higher level layer opens an extensible point as events type for replacing its simulation model to less abstracted simulation model. The simulator of high level abstraction sends the event to the event manager when the simulation model requires more precise calculation. The simulation controller in the higher level simulation sends a request to the appropriate

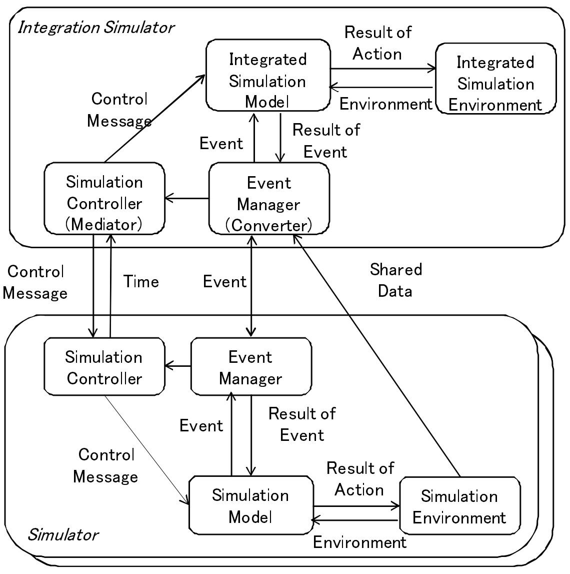


Fig. 2. External simulator integration architecture

lower level simulator. The data of each simulation environment is accumulated in each simulator but the simulation models exchange the data associated through shared environments.

2.3 External Simulator Integration Architecture

The activities of citizens must be interpreted from different point of views. In the traffic domain, people rushing to a hospital are interpreted as the movement of vehicles. In the flu pandemic domain, they are interpreted as contacts between infected and uninfected people. We describe the integration of different domain simulators in this section.

Different domain simulations have a weak relationship and share a little data. Loose coupling is desirable for connecting the various kinds of simulators. We proposed the external simulator integration approach that uses an external integration simulator (Fig. 2). The integration simulator manages the simulation processes and events representing the interaction between target simulators.

The integration simulation controller controls the simulation process of each simulator. Each simulator receives requests to calculate the next state from the integration simulation controller and returns the simulation time of the next state. The integration simulation controller decides the next controller to activate considering the simulation times of each simulation.

Objective simulators for integration are registered with the event manager in the integration simulator. The event manager gathers the events that occur in each simulator and sends them to the appropriate simulators after converting them into events that can be handled by the simulators.

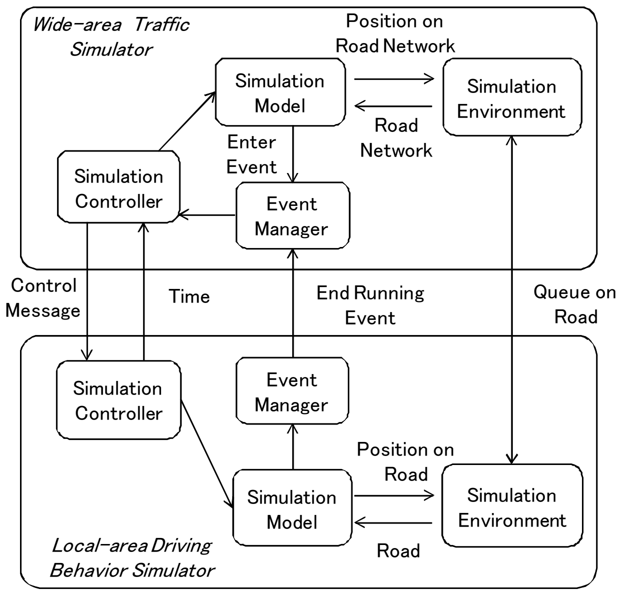


Fig. 3. Integration of wide-area traffic simulator with local-area driving simulator

Each simulator stores a different simulation environment. A simulation environment can be altered only by its own simulator, while some part can be read by the other simulators. The connection between simulators is realized by message passing to decrease the dependency between simulators and protect the simulation environments from external simulators.

3 Integration of Different Abstraction Level Simulators

To explain our approach, we describe the example of connecting a traffic flow simulator for wide-area road networks, with a driving behavior simulator for individual roads. Fig. 3 provides a system diagram of the integration.

Traffic flow simulations using simple agents are popular in the traffic domain. A simulator with a high level of abstraction uses simple nodes and links to model road networks. This is suitable when estimating the relation between traffic demand and traffic flow across wide areas[2]. However, this approach fails to provide realistic driving behavior simulations on particular roads. This is because details of the road structure (*e.g.*, the width of lanes) or surrounding environment including neighboring vehicles cannot be represented, the simulator fails to consider such local factors.

3.1 Simulator for Global Traffic

The agents in a global traffic simulation select appropriate routes and then pass along the routes.

The route selection module reads road network data and Origin-Destination (OD) data of agents. Road network data mainly describes the structure of the road network while the OD data consists of tuples of the starting point and the destination point of each agent. The agent selects the route that has minimum cost as identified from map information and the average trip time of each road. A route plan consists of paths, mode choice, daily activity, and so on.

The route execution module deals with abstracted road networks, not two-dimensional spaces. The route execution module realizes a queue-based simulator; that is, the road network is represented as a network of FIFO (First-In, First-Out) queues. Each agent moves over this queue-network between queues according to its scheduled routing plan given vacancies in the next queue. Traffic flows in this simulator are composed of agent transfers between queues.

3.2 Simulator for Local Traffic

In the simulator for local traffic, the agent is regarded as a virtual driver and vehicle. They move in a two-dimensional space rather than the abstract road network. The module reads agent ID and road ID from the simulation environment and gets details of the road's structure and surrounding environment including neighboring vehicles.

3.3 Integrating Simulators of Different Abstraction Levels

The integration of a local road simulator and a wide-area traffic simulator proceeds as follows (Fig.4).

1. At the initial step, a set of initial plans (routes) is generated based on free speed travel times generated by the wide-area traffic simulation.
2. The traffic flow simulation is run using the generated plans. On the abstract road network in the wide-area traffic simulation, agents move from entering queue to running queue or leaving queue to entering queue.
3. When an agent enters a running queue in the wide-area simulation, an entering event is generated. The wide-area simulator calls the appropriate local area simulator and specifies the event.
4. The local driving simulation calculates the driving behavior of the agents in the specified running queue for the next step. The wide-area traffic simulation and local driving simulation are jointly aware of all running queues.
5. If an agent in a running queue arrives at the end of the road (queue), the agent is moved from the running queue to leaving queue in the wide-area traffic simulation.
6. If the simulation time of the local driving simulation is earlier than that of the wide-area traffic simulation, the local driving simulator updates the time and goes to step 4.
7. Step 2 to step 6 must be iterated before the simulation time is at the end.

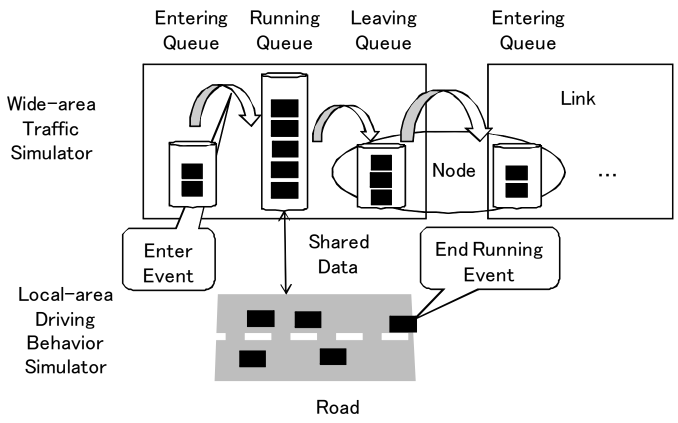


Fig. 4. Behavior of cars on road network and road

4 Integration of Different Domain Simulators

We connect traffic and electricity-consumption simulators to elucidate our integration technology. This section explains how these two simulators can be integrated. Fig. 5 provides a system diagram of the integration.

Previous papers did not try to combine with city traffic simulators and electricity consumption simulators which were developed independently. However, the rapid penetration of electric vehicles is forcing the emergence of a loose relationship. The charging of an electric vehicle is interpreted as vehicle movement (to the charging station) i.e. traffic domain, and as an electricity consumer i.e. electricity domain.

4.1 Traffic Simulation

The traffic simulation used in this section is the one described in Section 3. Traffic agents in the simulation leave their home, go to shopping areas or/and working places and return home. When the agents arrive at or leave a facility, arriving/leaving events are generated and sent to the event manager. The traffic simulator transmits the arriving/leaving events and destination change events to the electricity-consumption simulator.

4.2 Electricity-Consumption Simulation

The electricity-consumption simulator has several models: electric consumption, electric generation, and electric charging of electric appliances connected to facilities. The simulator calculates electric consumed in each facility step by step. The electricity simulator transmits connecting/disconnecting events to the traffic simulator. The electricity-consumption simulator has a time-step of 10 seconds, which is different from that of the traffic simulator.

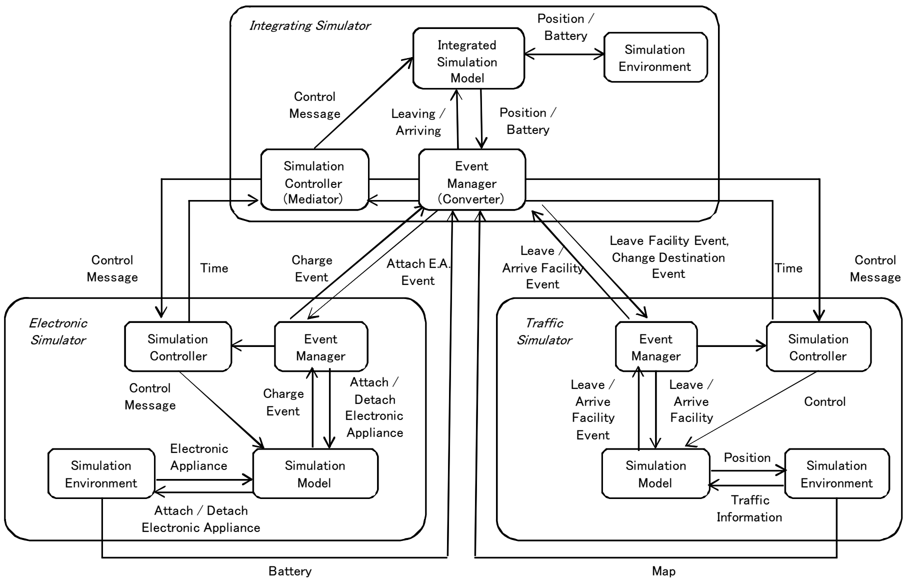


Fig. 5. Integration of traffic simulator with electronic simulator

4.3 Integration of Traffic Simulation and Electricity Simulation

The integration controller (IC) calls the simulator that has the earlier simulation time. The traffic and electricity-consumption simulations are linked by the arriving/charging/leaving behavior at charging facilities. IC calculates electric consumption based on the movement distance of electric vehicle and let the agent decide whether to go to the charging station or not. IC can match objects (agents and facilities) in the traffic simulation with those in the electricity-consumption simulation because the simulator has object ID tables.

IC proceeds as follows.

– When IC receives a “leaving facility” message from the traffic simulator, it records vehicle ID and facility ID. If the battery reserve of the electric vehicle is under the threshold that triggers a search for a charging station, IC locates the charging station nearest the vehicle and sends a destination change message to the traffic simulator. IC reads and records the battery reserve of the electric vehicle.
– When IC receives an “arriving at facility” message from the traffic simulator, it calculates the electricity consumption based on the distance moved, gained from map information, and updates the battery status of the vehicle. If the battery reserve is under the threshold that triggers charging, IC sends a charging event to the electricity-consumption simulator.

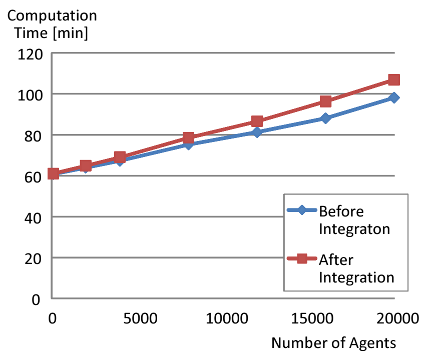


Fig. 6. Performance of layer based integration

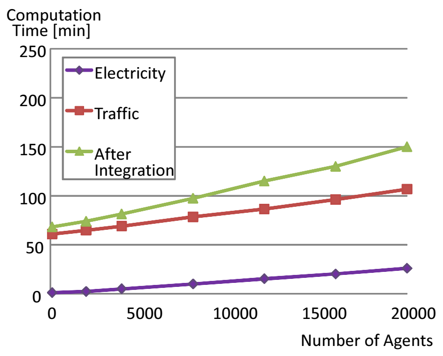


Fig. 7. Performance of external simulator integration architecture

– When the vehicle is fully recharged in the electricity-consumption simulator, the vehicle disconnects and the appropriate event is sent to IC. Additionally when the electric vehicle does not have another activity in the current facility, it leaves the facility and a "leaving facility" event is sent to IC.

5 Evaluation

5.1 Layered Integration

We investigate how our layered integration approach affects the computational overhead time. The computational time of integration simulation (after integration) and only wide-area simulation (before integration) are compared. We changed the number of agents from 0 to 20,000.

This evaluation was conducted to estimate the relationship between driving behavior on local roads and route selection in a wide area network. Each agent was given two simple driving rules "If current speed is slower than own desired speed, accelerate", "If there are slow cars in front, pass them if possible".

In this experiment, we used the road network of Kyoto city; it consisted of about fifty thousand links and one hundred thousand nodes in a square 20km x 20km area. We generated ODs (origin-destination) pairs randomly and assigned each to a different traffic agent. Simulation time step for wide-area traffic was one second and for local traffic it was 0.5 seconds. The simulation was executed over a 24 hour period.

The computer used in the following experiment had a Core i7 (8 core) 3.02GHz CPU and 12GB of memory. Simulators are executed on JVM (Sun JRE1.6.0_23 (64bit)) on Windows 7 (64bit). The version of MATSim was 0.1.1. These simulators use only one thread except initialize stage.

Fig. 6 plots the computation time versus the number of agents. As you can see, the computation time is directly proportional to the number of agents and no penalty was created by the integration.

5.2 External Simulator Integration

We investigate how the external simulator integration architecture affects the computational time for evaluating the proposed platform. The computational time of the integrated simulation and those of each simulation (traffic and electricity-consumption) are compared. We changed the number of agents; from 0 to 20,000.

This simulation scenario assumes that it is conducted to estimate the peak change in electric consumption caused by the use of electric vehicles. All traffic agents in the simulation use electric vehicles. Both simulation environments have the same homes and work locations facilties.

The electricity-consumption simulation had 10 second time steps. The simulation machine and the traffic simulation used the same specifications as the simulation described in Section 5.1.

Fig. 7 plots the computation time versus the number of agents. As you can see, the computation time is directly proportional to the number of agents and the integration incurred no penalty.

6 Discussion

6.1 Simulation Module Coupling

The most open approach to integrating multiple simulators is to release all interfaces to control the simulators and share all simulation environments. The integrated simulators would have a high degree of coupling in this case but a small degree of coupling is desirable for maintenance and reuse of each simulator.

We proposed the layered integration architecture for connecting simulators having different levels of abstraction. There is a strong relationship between the simulators because each simulator deals with the same phenomena from a different point of view. Therefore, our architecture takes the approach that a higher layer simulator calls the lower layer simulator and both exchange data via connection point. Our approach to integration has lower cost than the full integration approach because interactions between simulators are very simple. However, simulator coupling can be very high.

We proposed integration method by external integration simulator for combining different domain simulators. Each phenomenon in different domain has weaker relationship than that of different abstraction level simulators. External simulator integration is suitable because the simulation processes and simulation environments can keep independent each other. The implementation cost for integration with the approach is larger than layered approach because interactions between each simulators becomes complex. However, the couplings of simulators become low.

The proposed architectures require the creation of simulator components for controlling the simulation process, interfaces for observing some part of the simulation environments, and event types. IC can be reused because it also can be regards as simulator component.

6.2 Related Work

Some platforms that execute in distributed environment have been proposed. ASON[1], Repast[2] and ZASE[9] are large-scale multiagent simulation environments. These platforms provide support for multi-threading and agent data managing. The platforms help developers to implement simulator which can be used in parallel and distributed environments. Platforms that reuse existing simulators have been proposed [8]. The platform integrates Repast and Ascape[3] modules as reusing existing simulators.

These simulators are focused on extending scalability of simulators. While on the proposed architecture is focused on integration of simulators which capture various phenomena in a city with different abstraction level.

One future direction of this study is to support parallel and distributed environment for increasing computational resources because precise simulation of various phenomena in a city required taking much of calculation. We try to implement a simulation integration platform on the large-scale multiagent simulation platforms listed in this section.

7 Conclusion

Multiagent simulations are increasingly seen as the most attractive approach to analyze complex phenomena in a city and are being applied to various domains. The phenomena captured by the different simulators can be related. There is no platform that can integrate different simulators so that they can support and complement each other.

Many simulators have already been developed and they contain knowledge and technology created from previous work in their respective application areas. We proposed a simulation platform that can integrate multiple simulators. We classified the difference in simulators from two viewpoints: abstraction and target domain. Our contributions are as follows.

1. Integration of simulators having different abstraction level.
   We proposed a layered architecture wherein each simulator shares data across contact points and the higher level simulator calls the lower level simulator when the former needs precise data about a target phenomenon.
2. Integration of different domain simulators.
   We proposed an architecture based on an external integration simulator controller for loosely connecting different simulators via points for message passing.

One future direction of this study is to apply the proposed platform to tackle realistic problems. In particular, as shown in Section 4, we will try to capture the effect of electric vehicles which behave as actors who in the traffic domain and as electricity consumers across an entire city.

[1] http://cs.gmu.edu/~eclab/projects/mason/
[2] http://repast.sourceforge.net/
[3] http://ascape.sourceforge.net/

Acknowledgment. This work was supported by Panasonic Corp. - Kyoto University Joint Research: Crowd Navigation for Region EMS Considering Individual Behaviors and Preferences and Kyoto University Global COE Program: Informatics Education and Research Center for Knowledge-Circulating Society.

References

1. Balmer, M., Cetin, N., Nagel, K., Raney, B.: Towards truly agent-based traffic and mobility simulations. In: Proceedings of the 3rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004), pp. 60–67 (2004)
2. Balmer, M., Meister, K., Rieser, M., Nagel, K., Axhausen, K.W.: Agent-based simulation of travel demand: Structure and computational performance of matsimt. In: Proceedings of the 2nd TRB Conference on Innovations in Travel Modeling, pp. 1–30 (2008)
3. Deguchi, H., Kanatani, Y., Kaneda, T., Koyama, Y., Ichikawa, M., Tanuma, H.: Social simulation design for pandemic protection. In: Proceedings of The 1st World Congress on Social Simulation (WCSS 2006), vol. 1, pp. 21–28 (2006)
4. Epstein, J., Axtell, R.: Growing Artificial Societies: Social Science from the Bottom Up. MIT Press (1996)
5. Halle, S., Chaib-draa, B.: A collaborative driving system based on multiagent modelling and simulations. Journal of Transportation Research Part C 13, 320–345 (2005)
6. Kitano, H., Tadokor, S., Noda, H., Matsubara, I., Takahasi, T., Shinjou, A., Shimada, S.: Robocup rescue: search and rescue in large-scale disasters as a domain for autonomous agents research. In: Proceedings of the IEEE Conference on Systems, Men, and Cybernetics, vol. VI, pp. 739–743 (1999)
7. Paruchuri, P., Pullalarevu, A.R., Karlapalem, K.: Multi agent simulation of unorganized traffic. In: Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002), pp. 176–183 (2002)
8. Scerri, D., Hickmott, S., Padgham, L., Drogoul, A.: An Architecture for Modular Distributed Simulation with Agent-Based Models. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), pp. 541–548 (2010)
9. Yamamoto, G., Tai, H., Mizuta, H.: Consideration on realizing a hundred millions agent-based simulation. The IEICE Transactions on Information and Systems (Japanese edetion) 90(9), 2423–2431 (2007)
10. Yamashita, T., Izumi, K., Kurumatani, K., Nakashima, H.: Smooth traffic flow with a cooperative car navigation system. In: Proceedings of the 4th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), pp. 478–485. ACM Press, New York (2005)

Human Relationship Modeling in Agent-Based Crowd Evacuation Simulation

Masaru Okaya and Tomoichi Takahashi

Meijo University, Aichi, Japan
m0930007@ccalumni.meijo-u.ac.jp, ttaka@ccmfs.meijo-u.ac.jp
http://sakura.meijo-u.ac.jp/ttakaHP/Rescue_index.html

Abstract. Crowd evacuation simulations are becoming a tool to analyze and assess the safety of occupants in buildings. Agent-based simulation provides a platform on which to compute individual and collective behaviors that occur in crowds. We propose a human behavior model in evacuation based on the Belief-Desire-Intention (BDI) model and Helbing's agent behavior model. Human relationships affect the states of BDI at each simulation step, and altruism forces among agents are introduced in Helbing's model to affect agents' intentions in calculating agent movements. Two evacuation scenarios are examined so that the results match quantitatively and qualitatively with past disasters. The simulations reveal typical behaviors in a crowd evacuation; for example, family-minded human behaviors that lead to interactions in the crowd and other behaviors. The simulation indicates that due to the interaction it takes a longer time to evacuate from buildings in actual situations.

1 Introduction

After disasters, disaster-related measures have been taken in various countries and regions[11]. In the aftermath of Hurricane Katrina and the September 11 attacks, evacuation simulation has attracted attention to decrease the amount of damage resulting from disasters, and in particular, to save more human lives. Various types of crowd evacuation simulation approaches have been presented. Helbing et al. proposed a particle model that can simulate jamming by uncoordinated motion in a crowd[3]. Whereas the fluid-flow model and other macro-level simulations are modeled on the basis of precedent cases or experiments, they do not compute the interpersonal interactions that occur in a crowd or the behavior that involves people returning to the site. For example, at the Great East Japan earthquake at 2011 March 11, teachers waited parents come to school to bring their children for a while. And the teachers and students evacuated when they heard the tsunami coming. However, they had little time to go to safe places.

Agent-based simulation (ABS) provides a platform on which to compute the individual and collective behaviors that occur in crowds. The evacuation of people in emergencies or disasters is a complex task. Many systems have been presented to solve the following problems using the features of ABS [14]:

1. In Helbing's model, all agents perceive the accident at the same time. In real situations, people are not likely to perceive the emergency situation simultaneously, except when a global alarm system exists.
2. Depending on events that happen during the simulation, agents adopt one of the possible behaviors classified into normal life, escape, risk avoidance, rescue and other states.
3. Psychological states and knowledge of the agent affect the choice of actions. The level of knowledge about the environment or indications from other agents makes the agents change their evacuation routes.

Evacuation simulations are becoming a tool to analyze and assess the level of safety for human life provided in buildings. It is essential to verify the results of simulations. One method is to compare the simulations to real-life data, but it is difficult to conduct experiments of evacuations. Another method is to compare the data of past disasters at real situations. A detailed report on occupant behavior in the World Trade Center (WTC) disaster has been published and a related study has been carried out by Galea et al. [4]. They noted some features that are not supported by existing simulations. One of these features is that people either come together or break apart during an evacuation.

We believe that human relationships cause behaviors that make people either form a group to evacuate together or fall away from the group. We apply BDI-based agent model in which human relationships affect behavior. Human relationships result in a sense-reason-act cycle at each simulation step. Agent movement in a crowd is calculated using Helbing's model. The model is modified to calculate the factor of agent intentions as well as other physical factors. The remainder of this paper is organized as follows. Related works are introduced in Section 2. Section 3 describes the architectures of the evacuation system composed of BDI and crowd behavior models, taking human relations into consideration. Simulation scenarios and results are discussed in Section 4. Finally, a summary is provided in Section 5.

2 Related Works

Kuligowski reviewed 28 egress models and stated that there is a need for a conceptual model of human behavior in time of disaster so that we can simulate actions such as route choice, crawling, and even group sharing of information in order to make decisions on any kind of itinerary[5] [6]. Most crowd simulations model crowds as groups of people with common characteristics or objective. Some authors attempt to model an agent with individual characteristics to create a crowd evacuation behavior [13]. Steunebrink et al. reported how emotions can be used to specify constrains on the sense-reason-act cycle of an agent[1].

The study of Galea et al. on the WTC disaster presents the following five points that are required to simulate egress from buildings. They note that most of the current evacuation simulation models do not incorporate these points.

a) Travel speed model: It is well known that congestion occurs as people's move. For example, people congest at exits when they evacuate though a

Table 1. Research issues pointed in [4] and comparison to actual works [7] [9]

	issue	Pelech.	Pan
a	congestion at exit	✓	✓
	pass the injured (same direction)	✓	✓
	meet rescues (counter flow)	*	*
	join at staircase landing	*	*
b	sensing model of people	*	*
	information share among people	✓	✓
	communication among people		
	psychological model of people		✓
c	group evacuation	✓	✓
	group formation & break		
	human relation		
d	rescue agents		
	rescue headquarters		
	announce on evacuation		
e	exit routes barred by (debris, smoke, heat, water)	debris	debris

* Some of the issue can be handled.

narrow space, rescue teams that go to victims collide against people who are evacuating from buildings, and heavy congestion occurs at staircase landings where people from upper and lower floors come together.

b) Information-seeking task: People who are unfamiliar with buildings want to know how they can exit. They look for iconic warning signs, exchange information with people nearby, or follow others. Their perception abilities or behavior patterns change according to their psychological states caused by anxiety.

c) Group formation: Guidance from well-trained leaders enables evacuation flow to occur smoothly[7]. Schools drill their students to follow the instructions of teachers and evacuate together. In a time of disaster, people evacuate under different scenarios, and various factors of the scenarios result in people forming a group or breaking away from the group.

d) Experience and training: In the WTC disaster, announcements affected the evacuation behaviors of occupants. Proper announcements save lives; incorrect announcements increase the amount of damage resulting from disasters. How well information is gathered to rescue and how well the information is announced change the behaviors of occupants.

e) Choosing and locating exit routes: There are various kinds of obstacles in disaster situations that threaten safe and smooth evacuation. Choosing evacuation places and selecting routes affect evacuation behaviors.

Pelechano et al. examined the effectiveness of guidance by trained leaders. They suggested that simulations based on grids are limited in term of simulating crowd evacuation, and presented a system that simulates the local motion and global way finding behaviors of crowds moving under psychological and geometrical rules in a social and physical forces model [10]. Pan proposed a framework that deals with human and social behavior [9]. Table 1 presents the issues cited in Galea et al. and makes a comparison with the systems of Pelechano et al. and Pan.

Recently, human relationships among agents have been taken into consideration [8] [15]. These works show the difference in evacuation behaviors between agents with and without relationships. The communication model has not been taken into consideration.

3 Agent Intentions Selection by Human Relations and Evacuation Behavior Models

3.1 Requirements for Evacuation Simulation

The issues in groups (b), (c), and (d) in Table 1 are related to each other. We believe that human relationships are one of the factors that cause group formation. People either evacuate together or they fall away from the group. For example, occupants in buildings hear an emergency bell. Some people evacuate to safe places at once whereas others stay and begin to evacuate after confirming the reason why the bell rang. During an evacuation, some people may miss others and return to the room when they feel no anxiety.

We also assume that the following points are essential to simulate the aforementioned behaviors.

Behavior decision of agents: Agent behaviors during disasters make the difference between life and death[12]. There is no guarantee that people will know all the exits and split up to go to them to ensure efficient evacuation. Evacuation guidance or instructions from trained leaders are important when evacuations are begun or in deciding how to evacuate.

Micro-level evacuation simulation: Evacuation simulations consist of movements of humans - men and women of all ages. Differences in physical condition cause jams in human flows. When many people escape from an exit at one time, their behaviors cause congestion at the doors and collisions with people who go to other exits.

Verification of simulation results: The simulations are matched both quantitatively and qualitatively to the precedent cases to verify their results before using them in real applications. Qualitative considerations include the question of how the phenomena that occurred in the real cases can be simulated. The rate of evacuation is an example of quantitative measures.

3.2 Human Relations and Decision of Intentions

In emergencies, human behaviors differ from those in ordinary times. Behaviors are affected by people's mental condition. For example, when we fear for our physical safety, we think only of ourselves and will run away from a building with no thought to anything else. However, when we feel no anxiety, we think of other people and we evacuate together. Perry et al. summarizes these human

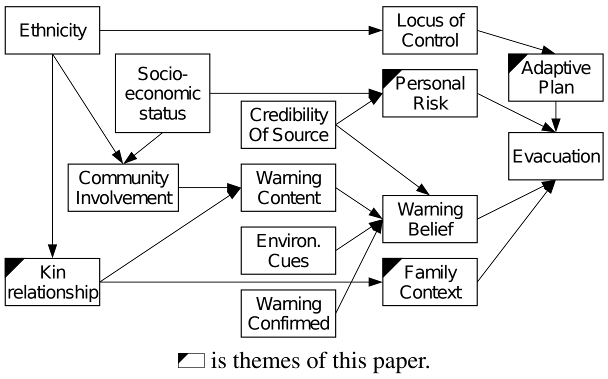


Fig. 1. Human relations factors in evacuation [11]

relation factors in decision making on the basis of empirical findings (Fig. 1). The marked factors are considered in this paper.

Herein, a BDI model in which human relation factor affects the choice of behaviors is presented. In our model, human relations affect the stages of the sense-reason-act cycle (Fig. 2).

Agents receive visual and auditory information according to their environment conditions. S_t is a set of sensor input at time step t. B_t, D_t, and I_t are the sets of belief, desire, and intention of the BDI model, respectively.

$$S_t = \{s \mid s \in \mathit{sensor\ input}\}$$
$$B_t = \mathit{belief}(S_t, B_{t-1})$$
$$D_t = \mathit{option}(B_t, I_{t-1})$$
$$I_t = \mathit{filter}(D_t, I_{t-1})$$

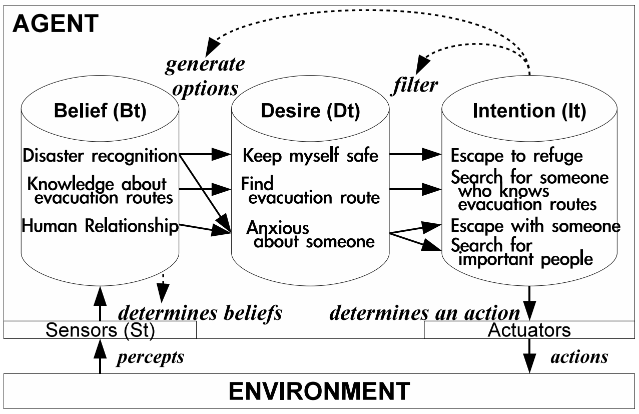


Fig. 2. BDI model of evacuation behavior that takes human relationships into consideration

belief: S_t updates B_t.

$B_t = \{\mathit{PersonalRisk} \cup \mathit{FamilyContext} \cup \mathit{AdaptivePlan}\}$

$\mathit{FamilyContext} \in \{\mathit{KinRelationship}\}$

B_t consists of human relation factors described in Figure 1. $\mathit{KinRelationship}$ handles the relations of agents and affects their evacuation behavior.

$AdaptivePlan$ are actions related to rescue actions including guiding others to safe places and coordinating the evacuation from rooms.

option: D_t is created from previous intentions and present beliefs.

$D_t = \{RiskAvoidance,\ Anxiety\}$

$RiskAvoidance = f(PersonalRisk,\ AdaptivePlan)$

$Anxiety = f(FamilyContext)$

$RiskAvoidance$ is one kind of desires that relates to the agents themselves.

$Anxiety$ is a set of desires that are related to the agent's family.

filter: I_t is updated and the action determined.

$I_t = I_{RiskAvoidance} \cup I_{Anxiety}$

$I_{RiskAvoidance}$ and $I_{Anxiety}$ are intentions dragged from the variant desires regardless of whether they are their own or others. Their preference criteria are different from those of the agents.

3.3 Crowd Evacuation Behavior Simulation

Some people evacuate by themselves and others in groups. The behaviors are based on various intentions, which differ among people. These differences in intentions result in various movements, and are taken into consideration in the simulation of people behavior. In a crowd, agent behaviors are influenced by the behaviors of others.

The agent's actions are selected from their I_t. The intention is calculated in the sense-reason-act cycle at every simulation step Δt. The motions of the agent are micro simulated according to a force calculated by the following equation. The micro simulation step $\Delta\tau$ is finer than the step Δt.

$$m_i \frac{d\mathbf{v}_i}{dt} = \mathbf{f}_{social} + \mathbf{f}_{altruism} \tag{1}$$

$$\mathbf{f}_{social} = m_i \frac{v_i^0(t)\mathbf{e}_i^0(t) - \mathbf{v}_i(t)}{\tau_i} + \sum_{j(\neq i)} \mathbf{f}_{ij} + \sum_{W} \mathbf{f}_{iW}$$

$$\mathbf{f}_{altruism} = \sum_{j \in G} \mathbf{f}_{ij}$$

$\mathbf{f}_{social}$ is a social force in Helbing's model. The first term is a force that moves the agents to their target. $\mathbf{f}_{ij}$ and $\mathbf{f}_{iW}$ are repulsion forces to avoid collision with other agents and walls, respectively. $\mathbf{e}_i^0$ is a unit vector to the targets , $\mathbf{v}_i(t)$ is a walking vector at t, m_i is the weight of agents i, and v_i^0 is the speed of walking. m_i and v_i^0 are set according to the age and sex of the agent$_i$. In our model,

1. $\mathbf{e}_i^0$ is derived from the agents' intentions I_t. The targets are places or humans. When child agents follow their parent, the targets are their parent whose positions change during the simulation step Δt.
2. $\mathbf{f}_{altruism}$ is an attractive force that keeps the agents in a group. It works when a parent waits till his or her child catches up. Group G is a unit in which members physically recognize each other. So it becomes zero when parents lose sight of their child. In this case, parents intend to look for their child. This change of intentions I_{t+1} causes the setting of a different $\mathbf{e}_i^0$.

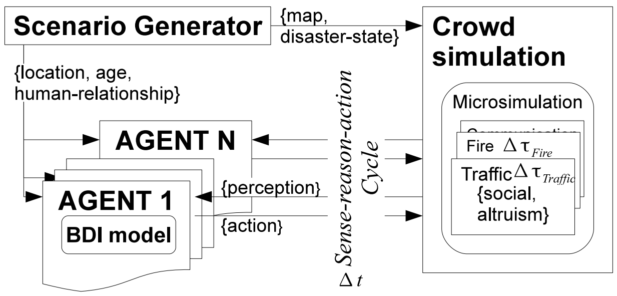


Fig. 3. Architecture of BDI based crowd evacuation system. $\Delta t > \Delta\tau_{Traffic}$

4 Evacuation Scenarios and Simulations

4.1 Prototype System and Behavior Model of Agents

Figure 3 shows the architecture of our system. Agents (in the left part) send their own properties to the crowd simulator at the start time and their targets at every sense-reason-action cycle. The targets present the agent's intentions that are selected from their BDI models. The crowd simulator calculates the movements of agents according to equation (1). The results of micro simulation are returned to the agents as its own and other agents' positions that are within their visible area.

RoboCup Rescue Simulation v.1 (RCRS) is used as a platform [2]. RCRS comprehensively simulates agents' behavior in a simulated disaster world and supports two kinds of agents: a civilian agent and a rescue agent. We implement three types of civilian agent with different BDI models (Table 2) and modify related simulators.

adult agents move autonomously and have no human relations with others. This type of agent can look for exits even when they have no knowledge of escape routes.

parent agents are adult agents who have one child. They are anxious about their child and have methods related to *anxiety*.

child agents have no data on escape routes and no ability to understand guidance from others. They can distinguish and follow their parent.

Two scenarios have been simulated to examine the effect of human relations on evacuation behaviors.

Table 2. Agent types and their behavior model

type	Belief	Desire	Intention
adult	personal risk	risk avoidance	evacuate to refuge, hear guidance
parent	personal risk	risk avoidance	evacuate to refuge, hear guidance
	family context	anxiety	seek child, evacuate with child
child	personal risk	risk avoidance	follow parent

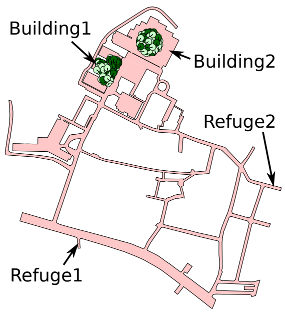


Fig. 4. A snapshot of an event (left) and campus layout and location of agents (right)

4.2 Evacuation from an Event Hall

Figure 4 presents a snapshot of an event site at a hall in which many families participate. Children enjoy the events and their parents watch them from distant places. The first scenario is one of which agents evacuate from a hall that is $70m$ by $50m$ and has one $4m$ wide exit. The parameters of the scenario are the number of agents and whether they are family members.

Figure 5 presents snapshots of evacuation simulations in two cases.

a) The 150 agents are all adults. They are divided in two groups, the left group composed of 100 adults and the right group, 50 adults.
b) The 150 agents comprise 50 adults and 50 parent-child pairs (50 parents and 50 children). The left group is composed of 50 adults and 50 parent agents the right, 50 children.

The following can be seen from the figure:

a) All agents move to the exit and congest there.
b) Where adult agents move to the exit, parent agents move to their child. When they move to their child, some parents collide with other agents who move to the exit.

At 120 steps, there are approximately 140 agents in the hall in both cases. However, it is clear that their behaviors differ. At 240 steps, the number of agents in case b) is twice as many as that in case a).

Figure 6 shows the number of agents that go out of the hall. The left and right columns correspond to case a) and b), respectively. The second row shows the results of simulation with 600 agents. In case b) of 600 agents, they are 200 adults and 200 parent-child pairs (200 parents and 200 children). The figures show that the congestion caused by the behavior of the parent agents takes more steps evacuate from the hall. Three lines in case b) are following settings of the parent agent;

without $\mathbf{f}_{altruism}$: When the parents lose their child, they look for their child at the level of BDI cycle. The $\diamond$ marked line shows a case that v_i^0s of parent agent and child agent are the same. The $\square$ marked line is that v_i^0s of child agent is 0.8 of that of parent agent.

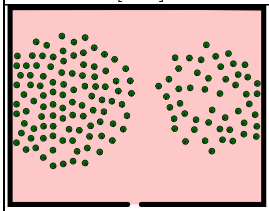
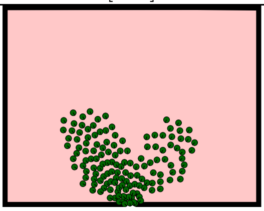
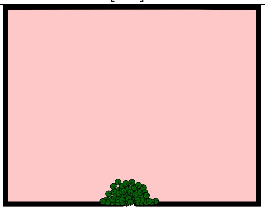
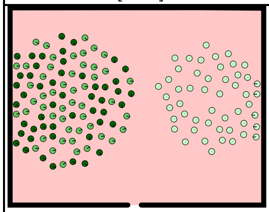
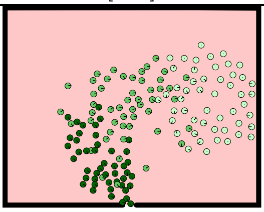
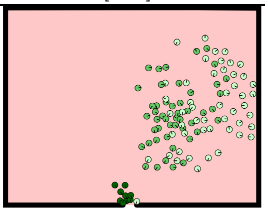
case	initial state	120 steps	240 steps
a)	[150]	[141]	[53]
			
b)	[150]	[146]	[107]
			

Fig. 5. Snapshots of evacuation. The number in [ ] is the number of agents in the hall. a) all agents are adults without human relations. b) 100 of 150 agents are parent-child relations.

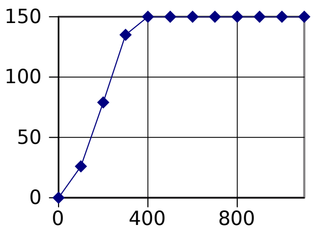
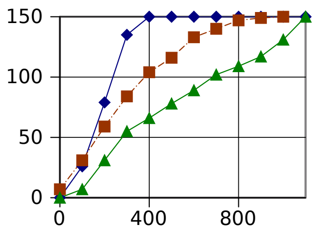
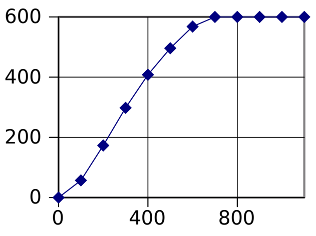
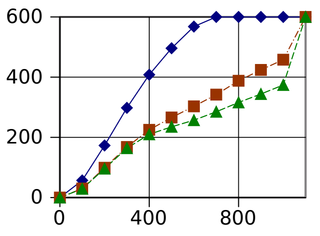
all agents are adults	adults and parent-child pairs
the number of agents:150	
	
the number of agents:600	
	

Fig. 6. The number of the agents that go out of the hall in time sequence

with f$_{altruism}$: The parent - child pair moves together keeping with their distance constant ($\triangle$ marked line).

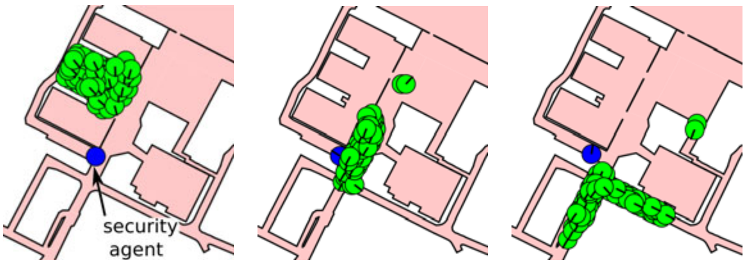


Fig. 7. Agents take different routes to Refuge2. One group goes down the road and the others up the road.

It is shown that the behaviors of the parents who care about their child take more time to evacuate.

4.3 Evacuation from a Campus with Guidance

The second scenario is evacuation from a campus. An event is held at the campus, and agents in two buildings evacuate to two refuges (Fig. 4). In an emergency, notices that urge people to evacuate to refuge locations are announced on the campus. Some agents hear the announcements and follow the instructions, and others do not. One hundred adult agents in Building 1 start to evacuate to Refuge 1, which is near Building 1. They know that there are two refuges and where the locations are. A security agent is at an intersection where agents that go to Refuge 1 pass. The security agent announces that it is safer to go to Refuge 2 than Refuge 1, so they go to Refuge 2. The agents can hear the guidance when they are within $30m$ from the security agent.

Five cases are simulated; the percentage of agents that follow the security agent varies from 0% to 100%, at every 25%. Figure 7 depicts the behavior of agents who hear the guidance. The agents at the front of the group turn left at the intersection to go to Refuge 2, whereas some others take a different route. They go up and turn right to reach the refuge.

Figure 8 presents the number of agents who arrive at Refuge 1 and 2. The numbers of agents who arrive at the refuge locations differ according to the percentage of those following the security agent's guidance. The rates and arrival time complement each other. There is one worth noting point around 1650 time steps. It takes a longer time for all agents to arrive at Refuge 1 in the cases of the 50% and 75% than 0% and 25% group, even though less agents evacuate to Refuge 1 in the 50% and 75% than 0% and 25% groups.

Figure 9 illustrates interesting snapshots that occur in the case of the 75% group. Some agents are involved in the flow of the other group that moves to the other refuge. The agents are divided into two groups: one (yellow) goes to Refuge1, and the other (green) goes to Refuge 2 from (1) to (3). Figure 9 (4) illustrates emergent behaviors that three agents are involved in the other group and return to go to Refuge 1 after some time. The behaviors explain the late arrival at Refuge 1 in the cases of the 50% and 75%.

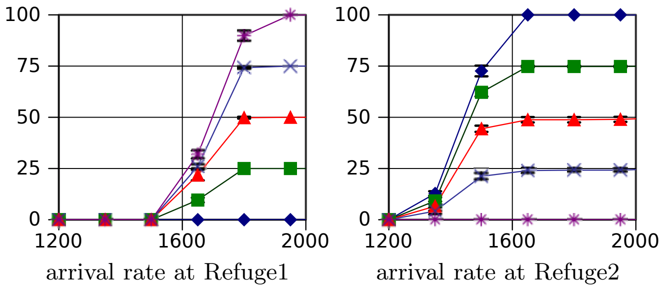


Fig. 8. Numbers of agents who arrive at refuge locations. $\diamond$, $\square$, $\triangle$, $\times$, and $*$ correspond to 0%, 25%, 50%, 75%, and 100% cases, respectively.

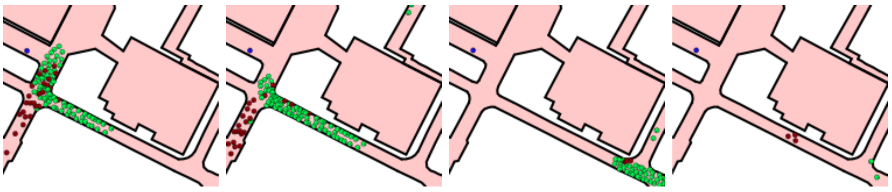

(1) agents separate to refuges
(2)
(3)
(4)some agents return to previous paths.

Fig. 9. Several agents are involved in the movements of a number of people that go to different destinations

5 Summary

The analysis of building evacuation has recently received increase attention as people are keen to assess the safety of occupants. Agent based simulation provides a platform on which to compute individual and collective behaviors that occur in crowds. We believe that psychological conditions must be taken into consideration in order to produce accurate evacuation simulations. We also assume that human relationships are factors that influence psychological conditions. We propose a model in which human relationships affect the states of BDI at each simulation step. We modify Helbing's model to include the factor of agent intentions as they are affected by their human relationships as well as other physical factors.

The model is implemented in an agent based simulation system using RCRS. Two evacuation scenarios are presented. The results of evacuation simulations reveal the following:

1. Family-minded human behaviors result in family members evacuating together, which causes interactions in the crowd.
2. Evacuation guidance affects crowd evacuation behaviors. The movements of a small number of agents are involved in a number of agent behaviors.
3. As in real life, evacuation takes more time when congestion occurs.

These are not programmed explicitly in the code of agents. The emergent behaviors occur as a result of agents' behavior-decision stages implemented as part of human relationships. These results demonstrate that our model provides an effective method of crowd simulation in an emergency.

References

1. Steunebrink, B.R., Dastani, M.D., Meyer, J.C.: Emotions to control agent deliberatoin. In: Proc. Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010), pp. 973–980 (2010)
2. Cameron Skinner, S.R.: The robocup rescue simulaiton platform. In: Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems, AAMAS 2010 (2010)
3. Kaup, D.J., Lakoba, T.I., Finkeistein, N.M.: Modifications of the helbing-molnar-farkas-vicsek social force model for pedestrian evolution. Simulation 81(5), 339–352 (2005)
4. Galea, E.R., Hulse, L., Day, R., Siddiqui, A., Sharp, G., Boyce, K., Summerfield, L., Canter, D., Marselle, M., Greenall, P.V.: The uk wtc9/11 evacuation study: An overview of the methodologies employed and some preliminary analysis. In: Klingsch, W.W.F., Rogsch, C., Schadschneider, A., Schreckenberg, M. (eds.) Pedestrian and Evacuation Dynamics 2008, pp. 3–24. Springer, Heidelberg (2008)
5. Kuligowski, E.D.: Review of 28 egress models. In: NIST SP 1032; Workshop on Building Occupant Movement During Fire Emergencies (2005)
6. Kuligowski, E.D., Gwynne, S.M.: The need for behavioral theory in evacuation modeling. In: Pedestrian and Evacuation Dynamics 2008, pp. 721–732 (2008)
7. Pelechano, N., Badler, N.I.: Modeling crowd and trained leader behavior during building evacuation. IEEE Computer Graphics and Applications 26(6), 80–86 (2006)
8. Okaya, M., Takahashil, T.: Bdi agent model based evacuation simulation. In: AAMAS Demo (2011)
9. Pan, X.: Computational Modeling of Human and Social Behaviors for Emergency Egress Analysis. Ph.D. thesis, Stanford (2006), http://eil.stanford.edu/xpan/
10. Pelechano, N., Allbeck, J.M., Badler, N.I.: Controlling individual agents in high-density crowd simulation. In: Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA 2007, pp. 99–108. Eurographics Association, Aire-la-Ville (2007), http://portal.acm.org/citation.cfm?id=1272690.1272705
11. Perry, R.W., Mushkatel, A. (eds.): Disaster Management: Warning Response and Cummunity Relocation. Quorum Books (1984)
12. Ripley, A.: The Unthinkable: Who Survives When Disaster Strikes - and Why. Three Rivers Press (2008)
13. Shendarkar, A., Vasudevan, K., Lee, S., Son, Y.J.: Crowd simulation for emergency response using bdi agent based on virtual reality. In: Proceedings of the 38th Conference on Winter Simulation, WSC 2006, pp. 545–553 (2006), http://portal.acm.org/citation.cfm?id=1218112.1218216
14. Thalmann, D., Musse, S.R.: Crowd Simulation. Springer, Heidelberg (2007)
15. Tsai, J., Tambe, M.: Escapes - evacuation simulation with children, authorities, parents, emotions, and social comparison. In: AAMAS (2011)

A Web Service Recommendation System Based on Users' Reputations

Yu Furusawa, Yuta Sugiki, and Reiko Hishiyama

Graduate School of Creative Science and Engineering, Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo 169-8555, Japan
19881226@moegi.waseda.jp, sugi-yuta@asagi.waseda.jp, reiko@waseda.jp

Abstract. In recent years, as the Internet spreads, the use of the Web Service has increased, and it has diversified. The Web Service is registered with UDDI, and the user selects service there and can use it for the provider by making a demand. In future, if the Web Service comes to be used more widely, the number of Web Services will increase, and the number of registrations at the UDDI will also increase. The user examines the large number of available services, and needs to choose the service that best matches their purpose. Quality of Service (QoS) is used as an index when a user chooses a service. Many studies show that the scoring of QoS for service selection is important. Quality of Service is registered by the provider and is treated as an objective factor. However, subjective evaluation, the evaluation of the user after the service use, is also needed to choose the best service. In this study, we use a new element, evaluation, in addition to QoS for service selection. We have expanded the existing filtering technique to make a new way of recommending services. Our method incorporates subjective evaluation. With this model, we apply the technique of information filtering to the Web Service recommendation and make an agent. Also, we simulate it after having clarified the behavior and tested it. The results of testing show that the model provides high levels of precision.

1 Introduction

As the Internet has spread in recent years, the Web Services have increased, and diversified. Web Services have come to be used at companies and in the home. Web Services are registered with Universal Description, Discovery and Integration (UDDI) by the service provider. Requesters select a service from the UDDI registry, and make a request to the provider. The requester can then use the service [2]. In the future, the number and diversity of Web services will increase even more, and it is expected that there will be more providers who register with their Quality of Service (QoS). Therefore, it is necessary to select a service that matches the users own particular needs from the high number of candidates of Web Services. When a user selects a Web Service, the QoS becomes an important index. The QoS includes various items such as cost, response time, and reliability, and the user searches for the factor that is right for them and then chooses a service. The QoS is handled as registered objective quality by a provider. Therefore, there are several studies on using QoS to select a Web Service.

In addition, not only the QoS but also the subjective Quality of Service, that is evaluation after use, is an important element in service selection. In this study, we used not only the QoS but also the reputation, i.e., subjective quality, for an index of the service selection. We thus recommend a Web Service that reflects both subjective and objective quality. We made a recommendation system in which the preference of the user is considered.

The paper is organized as follows. Section 1 introduce the research topic and outline its methodological implications. We discuss related research works in Section 2 and provide context to our work in Section 3. We overview of our multi-agent model for Web service recommendation and outline the algorithm based upon collaborative filtering and decision tree. In Section 4, we present the experimental settings, before proceeding to an empirical test of our model. Then we conduct several comparative experiments and discuss the results. Some ideas for future work are discussed in the conclusion.

2 Related Work

Various trust and reputation mechanisms have been proposed, and in particular, QoS has been discussed a lot in the literature and seen as the major criteria for selecting web services [9]. Most related work on QoS-based web service selection involves calculating the matching score between user's quality requirements and candidate web services. Wang et al. [8] and Sha et al. [7] studied the web service selection with QoS-based matching score. They use a quality set which includes cost, response time, reliability, accuracy, security, and reputation. It is normalized between user requirements and candidate web services, and then a matching score is calculated. However, as QoS is objective, this is not a subjective evaluation.

In other fields, there are studies using subjective evaluation. Murakami et al. [4] studied a web page recommend system with collaboration filtering. In this system, web pages are recommended from a similar user's viewing history. Similar users are decided by the viewing history of the page. In a study by Iwahama et al. [3], music is recommended by using a decision tree. Users evaluate music they listened to earlier. Each user's decision tree is made from the evaluation level and characteristic level of music. New music that fits each decision tree is then recommended. However, there is at present no similar work on using subjective evaluation in web service selections.

3 Recommendation Model

3.1 Web Service Based Recommendation System

We made a reputation repository as a database for accumulating reputation and select the Web Service based on it. We thus derive a preference of the user from the evaluation of the service that was used before. We propose a multi-agent model, i.e., a service recommendation system considering the preference of the user. This system consists of two agents, a broker one and a filtering one. They recommend new services based on accumulated evaluation and QoS. The overall behavior of the agents is Fig. 1, and the behavior of each agent is Fig. 2. We use both collaborative filtering [6] and a decision tree [5], and thereby enable recommendations that reflect both subjective and objective evaluations. However, a precondition is to use some services and evaluate their past performance.

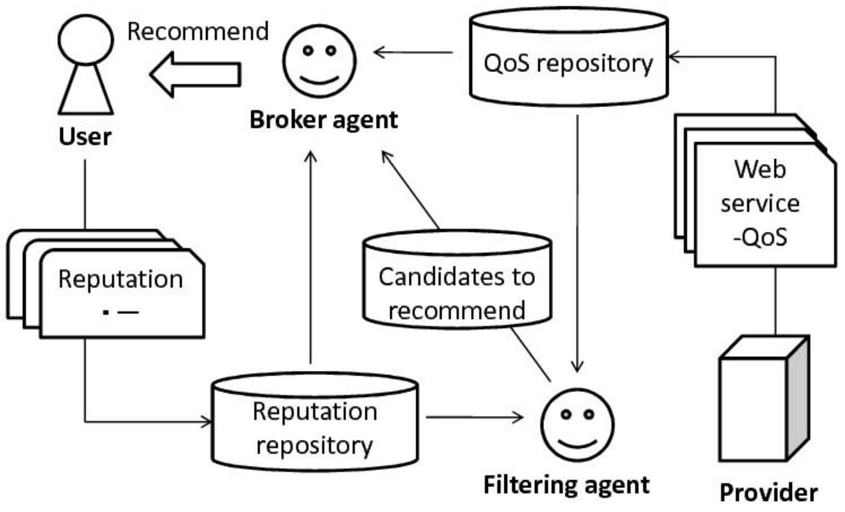


Fig. 1. Recommendation Model

The QoS, which was registered by a provider, is an assumed characteristic of the Web Service, and this system is used to recommend a Web Service based on the reputation which registered by users. The reputation data are accumulated in the reputation repository, and the QoS is saved in the QoS repository. A filtering agent is used to extract candidates of the Web Service for recommendation with reference to QoS and reputation. This agent uses collaborative filtering to do this. These candidates are sent to the broker agent to recommend. The agent make a user profile based on QoS and reputation and they receives the candidates to recommend from filtering agent. They compare the candidates with the user profile, and decide on a Web Service in accordance with the choicephilia of the user. The broker agent thus recommends a Web Service.

In this system, two agents use the collaborative filtering and decision tree, so a recommendation is made on the basis of reputation.

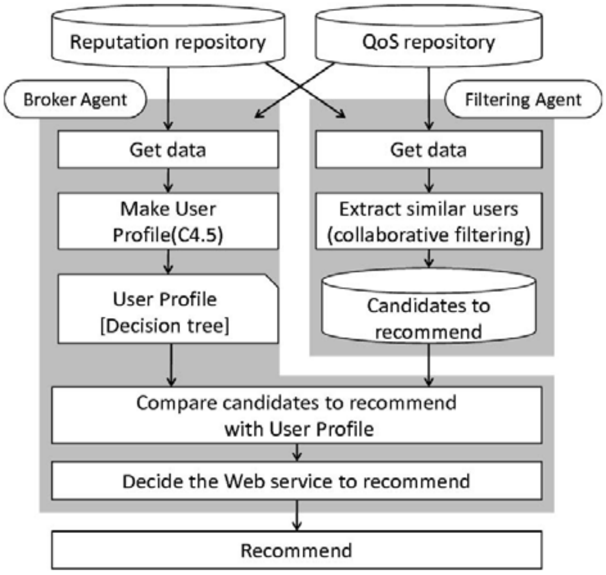

Reputation repository
QoS repository
Broker Agent
Filtering Agent
Get data
Get data
Make User Profile(C4.5)
Extract similar users (collaborative filtering)
User Profile [Decision tree]
Candidates to recommend
Compare candidates to recommend with User Profile
Decide the Web service to recommend
Recommend

Fig. 2. Behavior of Agents

3.2 Filtering Agent

A filtering Agent searches for users of a similar tendency regarding choicephilia to select, and makes candidates of Web Service to recommend. These are the roles of the filtering agent. The agents' behavior is described below.

1. The filtering agent gets data from the QoS repository and reputation repository. The data is the reputation of the Web service scored by the user who request the Web service and scored by the other users, and also the QoS of the Web Service.
2. The filtering agent uses reputation data to calculate similarity of the past preferences of the target user and the neighbors. The agent extracts several users with the same tendencies.
3. Make candidates to recommend services based on the reputation that was posted by similar users.
4. Send these candidates to the broker agent to recommend. This agent finishes the behavior.

3.3 Broker Agent

The role of the broker agent is as follows. They make a user profile from reputation data and compare the candidates which received from the filtering agent with the user profile. They then decide which Web service to recommend. Their behavior is described below.

1. The agent gets the data from the QoS and the reputation repository. The data is the reputation of the Web service scored by the user who requests Web service, and QoS of the Web service which was scored by the target user.
2. The agent make a user profile with the form of the decision tree. An algorithm of C4.5[5] is used on this occasion. The attribute to classify it is QoS, and the class is the evaluation level, i.e., reputation.
3. The agent compares the candidates received from the filtering agent with user profile, and classifies the candidates in every evaluation level.
4. Recommend services classified in best class.

Figure 3 shows an example of a made User Profile.

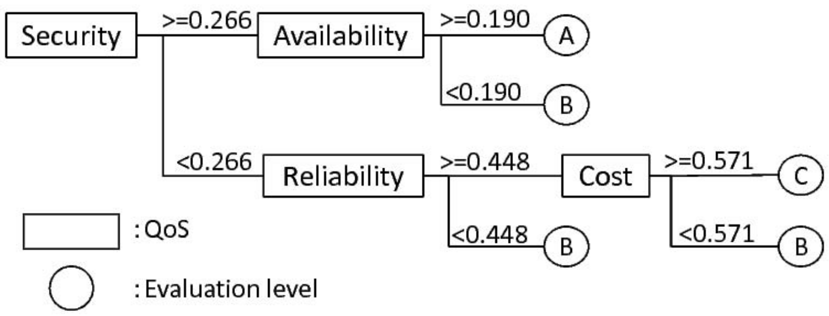


Fig. 3. Example of User Profile (Decision tree)

3.4 Collaborative Filtering

The filtering agent uses collaborative filtering when it searches for users of a similar tendency to select, and makes candidates of Web service to recommend. The number of services that the user who requests the Web service used for the past is one, and the number of services that the other user used for a past is m. The number of services in both is n. The evaluation level vector of the requester is $\overrightarrow{X} = \{x_1, x_2, \cdots, x_n\}$, the evaluation level vector of the other user is $\overrightarrow{Y} = \{y_1, y_2, \ldots, y_n\}$. The similarity between the requester's evaluation and the evaluation of the others is calculated by the cosine function of the requester's vector $\overrightarrow{X} = \{x_1, x_2, \cdots, x_n\}$ and the other's vector $\overrightarrow{Y} = \{y_1, y_2, \cdots, y_n\}$. The filtering agent is used to decide candidates from the use history of users with a high level of similarity.

$$Similarity(\overrightarrow{X}, \overrightarrow{Y}) = \frac{\overrightarrow{X} \cdot \overrightarrow{Y}}{|\overrightarrow{X}||\overrightarrow{Y}|} = \frac{\sum(x_i) \cdot (y_i)}{\sqrt{\sum x_i^2}\sqrt{\sum y_i^2}} \tag{1}$$

3.5 Decision Tree

The broker agent make a user profile with the form of the decision tree based on reputation. Algorithm of C4.5, suggested by Quinlan[5], is used on this occasion. Algorithm of C4.5 is as following three steps.

```
< Step1 >
if all data of the root node belong to the same class
then make a class node, and finish
otherwise choose one attribute by the criteria for selection of the attribute
and make a distinction node
< Step2 >
Set up the branch of a chosen attribute
< Step3 >
Perform < Step1 > and < Step2 > in all nodes recursively
```

In $< Step1 >$, entropy for the criteria is used to select the attribute. A unit of the entropy is a bit, and can be calculated from expression(2).

$$-\log_2 P(x_i) \tag{2}$$

In a random variable X, each x_i(i=1,2, $\cdots$, n), that is the element of a random variable X, has probability $P(x_i)$. The entropy, expected value of a set X, is info(X). The expression of info(X | A) is the entropy necessary to decide the class of the element x in a set X. The gain(A,X) is the information gain when attribute A is selected; split_info(A,X) is the value to normalize a bias of the set X, it is entropy when it was divided X into the subset of the m unit. The gain_ratio(A,X) is the ratio of part which is useful among the gain(A,X). Each value is calculated in the following expressions(3) $\sim$ (7).

$$info(X) = -\sum_{i=1}^{n} P(x_i)\log_2 P(x_i) \tag{3}$$

$$info(X|A) = -\sum_{j=1}^{m} \frac{X_j}{X} \times info(X_j) \tag{4}$$

$$gain(A,X) = info(X) - info(X|A) \tag{5}$$

$$split\_info(A,X) = -\sum_{j=1}^{m} \frac{X_j}{X}\log_2 \frac{X_j}{X} \tag{6}$$

$$gain\_ratio(A,X) = \frac{gain(A,X)}{split\_info(A,X)} \tag{7}$$

The attribute that is the highest value of gain_ratio is when the condition branches. The attribute to become the condition branching of each node of the decision tree is QoS, and the class to classify a set is the evaluation of the user. The Algorithm 1 is the algorithm to use numerical data in the decision tree. In this algorithm, the numerical data are sorted in ascending order, and a set is divided with the value of the data in ascending order as the border. It is used to calculate the gain_ratio, and the data of the highest value of gain_ratio is selected as the border ω to divide a set. Set X is divided into two groups by the border ω. The high group is a new set, H, and the low group is a new set, L. It is divided by whether it meets the following condition: values of attribute A are more than those of ω.

Algorithm 1. Algorithm to use numerical data in decision tree

```
$X = \{x_1, x_2, \ldots, x_n\}$ // $\forall i, j, i < j \rightarrow x_i \geq x_j$
$\omega \leftarrow x_1$
for $i = 2$ to $n$ do
    if $gain\_ratio(x_i, X) > gain\_ratio(\omega, X)$ then
        $\omega \leftarrow x_i$
    end if
end for
$H \leftarrow \{\}$
$L \leftarrow \{\}$
for $i = 0$ to $n$ do
    if $x_i \geq \omega$ then
        $H \leftarrow H + x_i$
    else
        $L \leftarrow L + x_i$
    end if
end for
return $H, L$
```

4 Experiment

4.1 Setting

We made the agents to recommend the Web service, and conducted verification experiments for this system.

In this experiment, the number of Web services is 100, and the number of users is 300. This is because we predict that the number of users is more than the number of Web services. The number of the similarity users that is extracted by the filtering agent is 10, and we set an evaluation level with 3 phases of the A, B, and C. The reason an evaluation level has three phases is that the decision tree cannot be used to treat the same grade. The number of entry of QoS is 5 on the pattern of the previous study [8], and it is a random value.

The result is an average of ten trials. We compared the proposed method with two conventional recommendation methods. The first method is decision tree [5] as (Existing method 1 in Fig.5 and 6), and the second method is collaborative filtering [6] (Existing method 2 in Fig.5 and 6).

4.2 Preparation

We made the data of a QoS repository, the QoS data of the Web service, and the data of reputation repository, it is reputation of users. First, we made 100 QoS data of Web service. The value of QoS is a random value between 0 and 1 as the value that was the standardized QoS. This is because that realistic value of QoS differs in range, for example, cost, response time, and reliability.

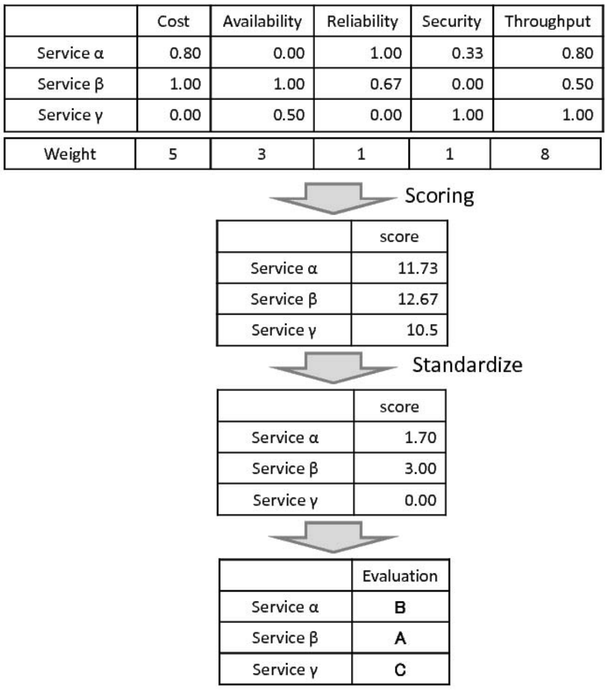
The following is an explanation of how we make reputation data registered with a reputation repository. First, we gave a degree of importance for each QoS with 15 phases of random numbers. The example of reputation data processing is shown in Fig. 4. We then calculated the product sum of QoS and degree of importance and standardized it to between 0 and 3. The score is this value, the evaluation level is A when this value is more than 2, the evaluation level is B when this value is 1-2, and the evaluation level is C when this value is lower than 1. Au Yeung [1] showed that the evaluation of similar products in the Web is similar. In their evaluation method, a similar score is scored to similar service and evaluation is also similar. In addition, evaluations of user of similar tastes were also found to be similar in their study. We used only the number of learning data as data registered with a reputation repository in random from made data, and the other data are used as the data which have not been evaluated.

5 Results

A comparison was made between the average score, accuracy, number of recommendations for every number of services used in the past for the suggested method, and the existing methods 1 and 2. Accuracy can be defined as the rate at which the recommended services have been rated A and it can be determined from the following equation (8).

$$Accuracy = \frac{number\ \ of\ \ recommended\ \ services\ \ rated\ \ A}{number\ \ of\ \ recommended\ \ services} \tag{8}$$

Also, the score is the score of service calculated by the users' tastes and value of QoS. These have been standardized to 0-3. The average score is the average of score for the service that have been recommended. The number for the service recommendation is the number of services recommended by this recommendation system.



	Cost	Availability	Reliability	Security	Throughput
Service α	0.80	0.00	1.00	0.33	0.80
Service β	1.00	1.00	0.67	0.00	0.50
Service γ	0.00	0.50	0.00	1.00	1.00
Weight	5	3	1	1	8

	score
Service α	11.73
Service β	12.67
Service γ	10.5

	score
Service α	1.70
Service β	3.00
Service γ	0.00

	Evaluation
Service α	B
Service β	A
Service γ	C

Fig. 4. Example of Reputation Data Processing

5.1 Recommendation Accuracy

Fig. 5 shows the recommended accuracy by the difference of learning data. This figure expresses the accuracy of recommendation by each method in the number of the learning data 50, 25, and 10 from the top.

As seen in Fig. 5, the accuracy for the suggested method is over that of the existing methods. Also, irrespective of the number of learning data, the recommended accuracy showed a very high rate for the suggested method. When the learning data is 10, the recommended accuracy is 10.

5.2 Average Score

Next, the average score for each method will be shown in Fig. 6. From Fig. 6, irrespective of the number of learning data, the suggested method shows the highest score. As the number of learning data increases, the average score seems to be decreasing in existing method 2. However, the score appears to be stable for the suggested method. From this, it can be said that the recommendation in the high stable area is possible for the suggested method. As for the case with recommended accuracy, the learning data of 10 showed a higher accuracy than the learning data of 25 and 50 in dependence of collaborative filtering. However, this kind of effect could not be seen with the average score. From this, it can be

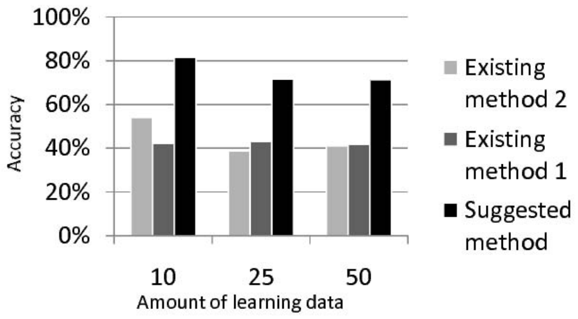


Fig. 5. Accuracy of each method by number of learning data

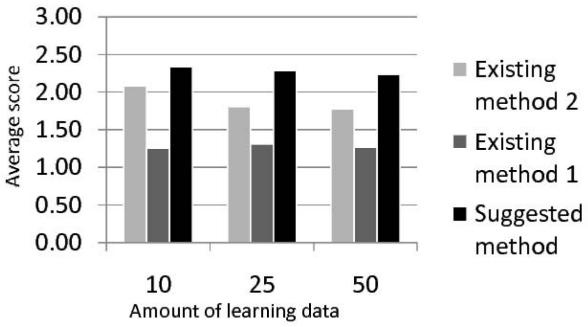


Fig. 6. Average score of each method by number of learning data

assumed that suggested method can be recommended with a stable quality of service without depending on any other methods. Also, since the score is stable with a high score of over 2, a good quality service could be recommended.

5.3 Number for Service Recommendation

The number of recommendations of each method is shown in Table 1.

From this table 1, the number for recommendations for the suggested method is lower than that in both the existing methods. This result is an inevitable outcome because what is extracted accords with the requirements of the user profile given by the candidates recommended by collaborative filtering of the Web Service.

After it was recommended service, the user needs to think what is necessary from the different elements. In this case, the number for recommendation is not low. Using our system significantly decreases the burden of selecting the services for users.

Table 1. Number for service recommendations for each method by the number of learning data

Number of learning data	Existing method 1	Existing method 2	Suggested method
10	20.0	17.7	5.2
25	17.1	28.1	7.0
50	21.5	23.3	4.6

6 Discussion

We used our method and a recommended Web service to combine two methods, collaborative filtering and a decision tree together by the suggested method and recommended Web service. We constructed recommendation system that reflected the choicephilia of the user.

Recently, when users select their Web services, they select services that match their needs in terms of cost, response time, and reliability. When the number of Web services increases it becomes more difficult to select the Web service that best fits these needs. Our purpose in this study was to make better recommendations by making the burden of choice smaller.

Therefore, we point out that user is recommended a service that best matches their needs and interests and to make the load less for selecting services. Recommended accuracy, average score, the number for recommendations is used as indexes for discussion. The result shows that the suggested method showed a high rate in 3 of the indexes: recommendation accuracy, average score, and the number of recommendations. Thus, services that fit the users' tastes can be recommended and selecting of services is simplified. Also, the result is not expected to be affected by the number of services is used in the past because the data shows that result does not change with the number of learning data.

7 Conclusion and Future Work

We suggested a multi-agent system that recommends a Web service from a large number of candidates of the Web Service. The recommendation reflects both subjective and objective quality. We inspected the effect of the suggested method by doing a simulation experiment. The results show that a recommendation system that uses the new index, subjective evaluation, can provide a recommendation that reflects the choicephilia of the users.

We used the subjective evaluation to recommend, but the subjective evaluation is accumulated equally and is static. Future works include the expansion of our method to a dynamic recommendation system that varies with changes in the level of choicephilia.

References

1. Au Yeung, C., Iwata, T.: Trust relation and product rating on the web. WebDB Forum (2010)
2. Erl, T.: Service-oriented architecture: concepts, technology, and design. Prentice Hall (2005)
3. Iwahama, K., Hijikata, Y., Nishida, S.: Content-based filtering system for music data. In: Application and the Internet Workshops, pp. 480–487 (2004)
4. Murakami, E., Terano, T.: Collaborative Filtering for a Distributed Smart IC Card System. In: Yuan, S.-T., Yokoo, M. (eds.) PRIMA 2001. LNCS (LNAI), vol. 2132, pp. 183–197. Springer, Heidelberg (2001)
5. Quinlan, J.: C4.5: Programs for machine learning. Morgan Kaufmann (1993)
6. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: an open architecture for collaborative filtering of netnews. In: Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW 1994), pp. 175–186 (1994)
7. Sha, L., Shaozhong, G., Xin, C., Mingjing, L.: A qos based web service selection model. In: International Forum on Information Technology and Applications (IFITA 2010), pp. 353–356 (2009)
8. Wang, X., Vitvar, T., Kerrigan, M., Toma, I.: A Qos-Aware Selection Model for Semantic Web Services. In: Dan, A., Lamersdorf, W. (eds.) ICSOC 2006. LNCS, vol. 4294, pp. 390–401. Springer, Heidelberg (2006)
9. Wang, Y., Vassileva, J.: Toward trust and reputation based web service selection: a survey (2007), http://bistrica.usask.ca/madmuc/papers/yaojulita-ws-mas-survey.pdf

Human-Centered Planning for Adaptive User Situation in Ambient Intelligence Environment

Makoto Sando and Reiko Hishiyama

Graduate School of Creative Science and Engineering, Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo 169-8555, Japan
sando@fuji.waseda.jp, reiko@waseda.jp

Abstract. The new intelligence system named Ambient Intelligence (AmI) is now in the limelight. In this AmI environment, when multiple people in various situations cooperate mutually, they need to to be supported simultaneously and effectively. Furthermore, peoples' context and activities change continuously, so it is also necessary to achieve dynamic correspondence between people. Therefore, a planning agent has been developed to enable context-aware service composition. The planning agent creates plans dynamically to support multiple people in different environments. In the experiment, the dish event scenario was adopted and the planning agent allotted multiple people ingredient collection tasks.

1 Introduction

Various devices that surround us, including information appliances, are becoming more advanced. Along with this, a network environment is beginning to be achieved that includes the sensor that perceives user's state. The user becomes able to connect with the network from various environments like domestic information appliances, mobile devices outdoors, or ID information in the office. In addition, if user's behavioral context and purpose can be understood highly accurately in the environment, it is thought that this will lead to the achievement of a beneficial support service depending on the user's situation. For these reasons, a new intelligence system named Ambient Intelligence (AmI) [4] [1] that intellectually and beneficially supports users from an environmental side has begun to be paid attention to. AmI is expected to be merged into people's daily lives. For example, the "smart house" that automatically controls the appliances when the user is in unawareness state[13] [2]. However, achieving this AmI environment is challenging and difficult. People live their lives in rapidly changing environments. Thus, these environments must be sensed at a high level. Moreover, it needs to correspond to the processing of huge information obtained from the ambient surrounding. Furthermore, an advanced action plan must be made that corresponds to a dynamically changing situation to enable AmI to support the user as much as possible. Because of these factors, to introduce AmI into our lives, some researchers have tried to enable complex processing by using the agent technology [9]. Moreover, the AI Planning assists the problem solving by producing a plan of action to achieve a particular goal [12]. Therefore we set out to introduce the planning-agent that dynamically generates and executes the plan corresponding to the situation into the AmI environment. The planning agent offers beneficial support by considering the user's situation from information such as RFID and GPS data. The plan is refined with the process of the multi planning interaction of a human-agent and machine-agent, and it aims at *Acquisition of the human-centered planning* through this process. The scenario in this study was for users to collect ingredients and cook dishes. In Section 2, we discuss related research works. In Section 3, we propose the action forecast process of the planning agent and we overview of our planning agent model in Section 4. In Section 5, we present three experiments that we tested the effectiveness of our model and then we discuss the results. In Section 6, we analyze the evaluation of the planning agent. Finally, in Section 7, we conclude the work and discuss some ideas of future work.

2 Related Works

An important follow up to Ubiquitous Computing [8] has been developed under the term Ambient Intelligence [7]. The ambient intelligence framework to support service by refining the planning task have been discussed in precedent works. Patkos *et al.*[10] need to seek ways to represent device and service profiles, plans and goals in a manner that is mutually apprehended and correctly interpreted by all participating entities. Bajo *et al.'s* applied the planning agent into the ambient intelligence environment [3]. This planning agent dynamically generated the medical treatment plan by using the RFID log data (the place of nurse or patient) and the treatment scheduler data. Hattori *et al.'s* proposed an agent that supports user's grocery shopping [6]. This agent obtains the user and shop place information and supplies the list of ingredients. However, the stock control of a refrigerator is not automatic, so effort is needed to control the stock of ingredients. Moreover, only one user can receive this agent system service, so this research has not yet considered the problems when two or more users interact with the agent.

3 Action Forecast

3.1 Learning Phase

(1) Generating the Set of Place

$L_i = \langle l_{i1}, l_{i2}, \ldots, l_{in} \rangle$ is the GPS data obtained on day i. $l_{ij} = [t_{ij}, x_{ij}, y_{ij}]$ is the data obtained in one measurement. t_{ij} is time, x_{ij} is latitude, and y_{ij} is longitude. We applied DBSCAN algorithm [5] to extract the place of users from L_i. DBSCAN algorithm can extract the arbitrary shape of a cluster by changing the distance of points and the threshold number of target points. From this algorithm, we can find the place $C = \{c_1, c_2, \ldots, c_m\}$ where users stayed a block of time. m is the number of extracted clusters.

(2) Generating the Frequent Action Pattern

$R_i = \langle r_{i1}, r_{i2} \ldots, r_{i|R_i|} \rangle$ is the line of place where users stayed in order of time. This is the action pattern of day i. r_{ij} is $[c_{ij}, T^b_{ij}, T^e_{ij}]$ and c_{ij} is the jth place where the users stayed on day i. T^b_{ij} and T^e_{ij} are the start and end times of the stay. We apply the prefixspan algorithm [11] to obtain the frequent action pattern $P_k = \langle v_{k1}, v_{k2}, \ldots, v_{k|P_k|} \rangle$. Prefixspan algorithm can extract the frequent line pattern. Each v_{kj} is $[c_{kj}, \bar{T}^b_{kj}, \bar{T}^e_{kj}]$. $\bar{T}^b_{kj}$, $\bar{T}^b_{kj}$ is the average start time of a stay in c_{kj} and its average end time when the pattern P_k appeared in each $\{R_1, R_2, \ldots, R_N\}$. N is the number of learning days.

(3) Co-occurrence Relation with Scheduler Words

The next step is defining the co-occurrence relationship with the scheduler words and the cluster by using the calculating formula of posterior probability of a Bayesian classifier (1). For instance, calculating the highest probability of a cluster when the scheduler word is HOME, then the cluster's place will be HOME.

$$P(c_i|w_j) = \frac{P(c_i)P(w_j|c_i)}{\sum_{i=1}^{|c|} P(c_i, w_j)} \tag{1}$$

$P(c_i|w_j)$: posterior probability of appearance of place c_i after the scheduler words were described. $P(c_i)$: probability of appearance of place c_i over the leaning time. $P(w_j|c_i)$: probability of appearance of scheduler word w_j was described when the user was in the place c_i. $P(c_i, w_j)$: probability of conjunction of the place c_i and the scheduler word w_j.

3.2 Precast Phase

To precast the action after the time of t_{now}, the GPS log data by the time of t_{now} and the scheduler data after the time of t_{now} are necessary. $\hat{R} = \{\hat{r}_1, \hat{r}_2, \ldots, \hat{r}_{|\hat{R}|}\}$ is the line of place by the time of t_{now}, and $\hat{S} = \{\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_{|\hat{S}|}\}$ is the line of scheduler words where the starting time is after the time of t_{now}($\hat{r}_i = [\hat{c}_i, \hat{T}^b_i, \hat{T}^e_i]$ and $\hat{s}_j = [\hat{w}_j, \hat{T}^b_j, \hat{T}^e_j]$). Moreover, the set of these two lines is the action line of the day. Concordance rate of the pattern P_k and $\hat{R}$,$\hat{S}$ is calculated by the score matching formula (2).

$$\begin{aligned} score(P_k, \langle \hat{R}, \hat{S} \rangle) = Prob(P'_k)\big((1-w)\sum_{i}^{|\hat{R}|} rscore(P_k, \hat{r}_i) \\ + w \sum_{j}^{|\hat{S}|} sscore(P_k, \hat{s}_j)\big) \end{aligned} \tag{2}$$

$rscore(P_k, \hat{r}_i)$ is the concordance rate of the place line by the time t_{now} and the pattern. The score is 1 when the time (start to end) in the same place $\hat{r}_i$ overlaps, and the length of time is less than the threshold. If this not the case, the score is 0. $sscore(P_k, \hat{s}_j)$ is the concordance rate of the place line after the time t_{now} and the pattern. The score is $P(c_i|w_j)$ when the time (start to end) in the same place $\hat{s}_j$ overlaps, and the length of time is less than the threshold. If this not the case, the score is 0. w $(0 \leq w \leq 1)$ is the weight that controls the effect of $\hat{R}$ and $\hat{S}$. $Prob(P'_k)$ is the appearance probability of each action pattern depending on the day of the week.

$$P'_k = \alpha P_{k,1} + \beta P_{k,2} + \gamma P_{k,3} \tag{3}$$

$$Prob(P'_k) = \frac{P'_k}{P'_1 + P'_2 + \ldots + P'_M} \tag{4}$$

M is the total number of action patterns and $P_{k,l}$ is the number of action pattern appearances ($l = 1$: the number of appearances on the same day of the week, $l = 2$: the number of appearances on a different day of the week, but the same type of day (weekday or weekend day), $l = 3$: the number of appearances on a different day of the week and a different type day). When the forecasting day is a weekday, α is the weight against the number of appearances on the same day of the week, β is the weight against the number of appearances on the different weekday, and γ is the weight against the number of appearances on a weekend day (when the forecasting day is holiday weekend day, β is the weight against the number of appearances on the other weekend day and γ is the weight against the number of appearances on a weekday.). Now we can evaluate the concordance rate of the schedule (past line of the place and the future scheduler data) and the frequent action patterns from above. We regard the largest number of the score as the next action after t_{now}.

4 Planning Agent

4.1 Agent Model

In this study, we eliminated manual stock control of the refrigerator by the automatic recognition technology of RFID. Moreover, we enabled the planning agent to understand how many ingredients the refrigerator has stocked and what the user's is doing. The concept of the planning agent is shown in Fig. 1 below. The planning agent generates the plan by cooperating with the action forecaster, the ingredient informer, and the event informer, and requests the users to collect the ingredient. Action forecaster acquires the user's location information from the GPS log data and determines the user's behavior pattern from the co-occurrence relationship between the GPS log data and the scheduler data. The user's next action is forecast by matching the accumulated behavior pattern and the user's behavior pattern up to this time. In the ingredient informer, the ingredient's name or amount is recognized from the RFID tag, and when the ingredient comes in and goes out, the data is stored into the refrigerator database. The event informer manages information about event attendees or the date of the events that users have registered, and when the planning agent asks for the event information, it supplies and renews the information.

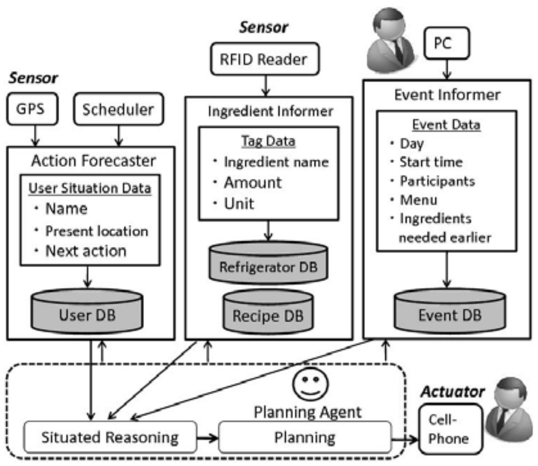


Fig. 1. Concept of Planning Agent

4.2 Planning Agent Module

The planning agent has five functions.

1. User Context Information Inquirer

The user context information inquirer is the function by which the planning agent inquires to the action forecaster about the user's present and forecast place information.

2. User Classifier

The user classifier is a function by which the planning agent classifies users into three groups by using the information from the user context information inquirer. These three groups all receive request to collect ingredients.

Group 1 (From home to event)

Group 1 has the users who can take the ingredients from their homes to the event. If the ingredients in the home can be taken, the cost of the event reduces, so the agent requests this group to collect the ingredients.

Group 2 (From anywhere except home to event)

Group 2 has users who can buy the ingredients on the way to the event. Having someone in this group buying the ingredients on the way to event is more effective than having someone already at the event go shopping.

Group 3 (From event and back again)

Group 3 has users who are already at the event and can go out and buy the ingredients. It is ineffective if someone already at the event goes out and buys the ingredients, so this group is the last group to receive requests.

3. Necessary-Ingredients-List Maker

The necessary-ingredients-list maker is the function with which the planning agent makes the list of ingredients that are necessary for the event. This list is inferred from the menu data, participant data from the event informer and the refrigerator stock data, recipe data from the ingredient informer. The information of ingredients necessary for the event but are not in the refrigerator is sent to the missing-ingredients-list maker.

4. Missing-Ingredients-List Maker

The missing-ingredients-list maker is the function with that the planning agent makes the list of ingredients necessary for the event but are not in the refrigerator. The ingredients' data is read by the RFID reader every time the ingredients are stored in the refrigerator. Thus, the missing-ingredients-list is also updated every time the ingredients stored in the refrigerator.

5. Missing-IngredientCollector

The missing-ingredients collector is the function with which the planning agent requests the users who have been classified in three groups to collect the ingredients on the missing-ingredients-list. The timing of the request depends on the beginning time of the event. The planning agent sends a request to a user's cell phone and re-plans in accordance with the contents of the reply.

4.3 Processing of Planning

In this study, not only the agent's action but also the user's action needs to be planned. Therefore, it is necessary to consider the interaction between the planning agent and the users in the planning process. In this study, we construct a plan in the order of (1) to (3) below.

(1) Making an Initial Plan

The first step of the planning process is the initial plan, such as the sequence and the interaction of each action. This initial plan is executed when there are no changes in the environment.

(2) Setting of Pre-during-post Conditions

After the initial plan is made, next the pre-during-post conditions are set. Each action in the initial plan is only executed when these pre-during-post conditions are met. The planning agent has a during-condition for messaging. This is because, for instance, when Group 1 replies to a request sent to Group 2, the amounts of necessary ingredients may change, and accurate amounts of ingredient might not be collected. Therefore, this during-condition is set so that all three groups are not requested to collect the missing ingredients at once.

(3) Countermeasure to Dynamic Environment

To take measures against a situation not initially planned for or that has no designated the pre-during-post condition, for instance if a request is not replied to, the countermeasure is an "if-then" conditional statement.

5 Experiment

5.1 Experimental Outline

We experimented three times in different users' situations to test the effectiveness of this method. We carried out a questionnaire after every experiment and use the results to improve the planning agent. The planning agent re-plans and requests by using the information from the users' messages, but it is important not to make the users do too much, so we limited the number of requests to two.

5.2 Results

Fig. 2, Fig. 3, and Fig. 4 are the plans that the planning agent generated and the trajectory of the requests in the three experiments. In addition, here are the details of the planning agent's improvements and the new functions that we added after each experiment.

Experiment 1

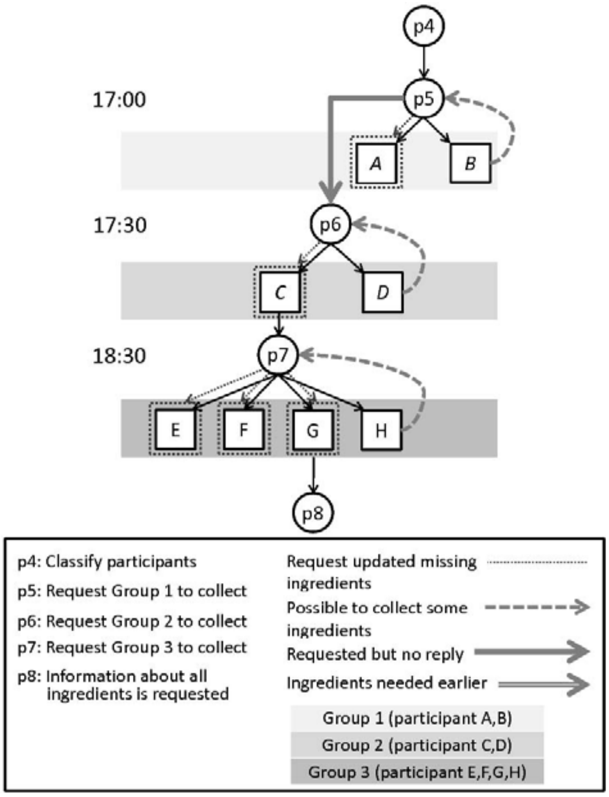

p4
17:00
p5
A
B
17:30
p6
C
D
18:30
p7
E
F
G
H
p8
p4: Classify participants
p5: Request Group 1 to collect
p6: Request Group 2 to collect
p7: Request Group 3 to collect
p8: Information about all ingredients is requested
Request updated missing ingredients
Possible to collect some ingredients
Requested but no reply
Ingredients needed earlier
Group 1 (participant A,B)
Group 2 (participant C,D)
Group 3 (participant E,F,G,H)

Fig. 2. Generated Plan: Experiment 1

Request Time and Classification of Users

The first experiment had eight participants and its event was held in a laboratory at Waseda University, Tokyo. Estimated start time was 19:00, and the first requests to Group 1 (2 users), Group 2 (2 users), Group 3 (4 users) were at 17:00, 17:30, and 18:00, respectively.

Generated Plan

All ingredients needed for the event were collected by five users, but they came 35 minutes after the estimated event start time. In the generated plan (Fig.2), there were many changes in the situation "Enable collection of only some ingredients", so five more requests were made than initially planned.

Questionnaire and Agent's Improvements

In the questionnaire, some participants said that "The planning agent requested more ingredients than could be brought" and "It would be convenient to have a automatic checkout system", so we decreased the amount of ingredients each user had to collect (a third of total weight) and added an automatic checkout system that can calculate the cost for one event attendee. Moreover, we modified the first request time to 14:00.

Experiment 2

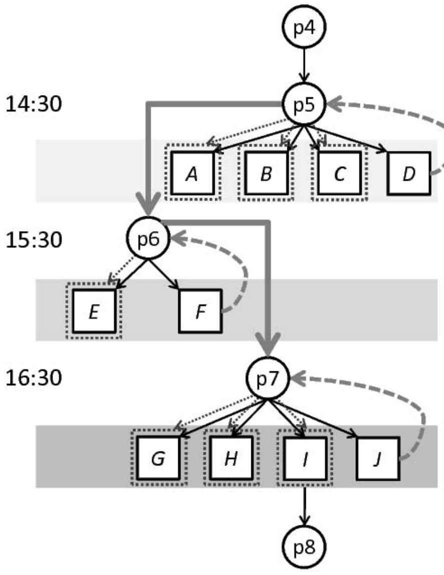


Fig. 3. Generated Plan: Experiment 2

Request Time and Classification of Users

The second experiment had 10 participants, and its event was held in a laboratory at Waseda University, Tokyo. Estimated event start time was 19:00, and the first request times for Group 1 (4 users), Group 2 (2 users), and Group 3 (4 users) were 14:30, 15:30, and 16:30, respectively.

Generated Plan

The event start time was 16 minutes after the estimated event start time, which was an improvement on experiment 1. In the generated plan (Fig.3), there were many changes in the situation "Enable collection of only some ingredients", as was the case in experiment 1, so 7 more requests were made than initially planned. In addition, "no responses from the users" occurred twice, but the planning agent flexibly responded to this change in situation.

Questionnaire and Agent's Improvements

In the questionnaire, some users pointed out that even though the amount of ingredients decreased by one-third of total weight, there were still too many ingredients. Therefore, we decreased the amount of ingredients to one-third of the total number of ingredients. Moreover, we added the preferential-collecting system to collect the ingredient that need to be cooked for a long time before the event. This system was expected to ensure that the event could start by the estimated event start time.

Experiment 3

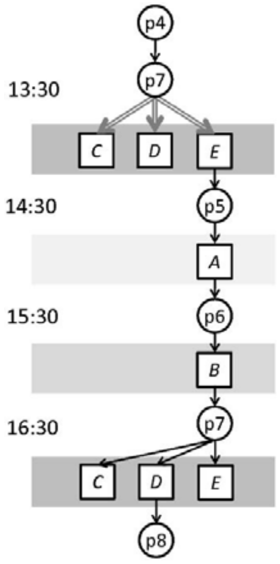


Fig. 4. Generated Plan: Experiment 3

Request Time and Classification of Users
The third experiment had participants, and its event was held in a laboratory at Waseda University, Tokyo. Estimated event start time was 19:00, and the first request times for Group 1 (1 user), Group 2 (1 users), Group 3 (3 users) were 14:30, 15:30, and 16:30, respectively. However, some ingredients needed to be cooked for a long time, so at 13:30 the planning agent requested Group 3 to collect these first.

Generated Plan
All the ingredients were collected and prepared by the estimated event start time. In the generated plan (Fig.4), there were no changes like in experiments 1 or 2, but some ingredients need to be cooked for a long time, so the planning agent made two more requests than initially planned.

Questionnaire and Agent's Improvements
In responses, three participants said "It will be much better if the planning agent sends requests to all of us at the same time" (participants C, D, and E. Fig.4). This will be a useful reference to make a better planning agent in future work.

6 Discussion

We carried out a questionnaire to test the effectiveness of the planning agent that we introduced. The analytic view is effectiveness and weak point of the agent system that we introduced. We made 20 evaluation items for the planning agent, for instance "context-sensitive support", and asked the users to score each evaluation item in its level of importance and their level of satisfaction with it from 0 to 5 points. We also let the participants write personal evaluations of the planning agent.

6.1 Analysis of Evaluation

Table 1 shows the averages and coefficient of correlation between importance and satisfaction of the planning agent. Fig.5 is a scatter diagram of three experiments on which the planning agent's evaluations are plotted. Fig.5 or Table 1 shows that the average satisfaction increased as the number of experiments grew. From this, we can conclude the planning agent improvements that we made after each experiment made the performance better or generated more appropriate plans. The dotted line regions in Fig.5 show the evaluation items for the capability that

Table 1. Average and Coefficient of Correlation about Importance and Satisfaction

	Average Importance	Average Satisfaction	Coefficient of Correlation
Experiment 1	3.66	3.00	0.43
Experiment 2	3.62	3.32	0.41
Experiment 3	3.42	4.00	0.62

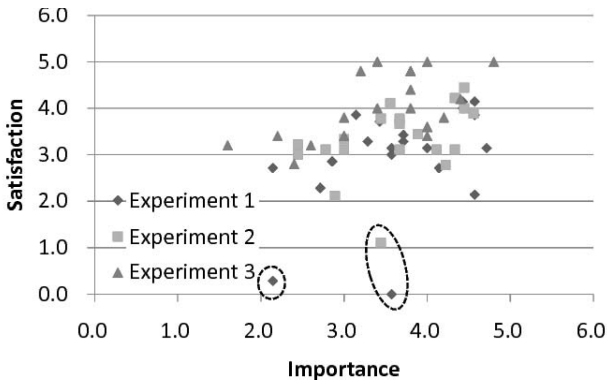


Fig. 5. Importance and Satisfaction

the planning agent did not have at that time, so the level of satisfaction is very low. The coefficient of correlation in experiment 3 was the closest to 1.00, so importance of and satisfaction with the planning agent were strongly correlated. This means the satisfaction advanced in proportion to the level of importance. Therefore, it is thought that the planning agent responded accurately to the wants of users over the three experiments. Table 1 shows that the satisfaction advanced from experiment 1 to experiment 2. Nonetheless, the coefficient of correlation decreased. The improvements we added after experiment 1 (decreasing the amount of ingredients to one-third of total weight and adding an automatic checkout system) might not have been the point all users wanted. Table 1 also shows that the average of satisfaction decreased a little over three experiments. The users who took part in the experiment more than once might have gradually come to think the evaluation items they were already satisfied with were not very important.

7 Conclusion and Future Work

In this paper, we proposed the planning-agent that dynamically generates and executes plan corresponding to certain situations. The plan was generated with the process of the multi planning interaction of a human-agent and machine-agent, and we obtained *human-centered planning through this process.* The result of the experiments showed that the planning agent was effective in the ambient intelligence environment.The next step in this research is to enhance the design of the planning agent so that it can improve autonomously by studying past generated plans. We will also try to allot various tasks for which the planning agent will have to cooperate with two or more agents. Moreover, we could not put the GPS log data into the action forecaster, so we used only the scheduler data to forecast the user's situation. Inferring from the GPS data and the scheduler data will achieve not only the accurate forecasts, but also the bring closer the idea of Ambient Intelligence.

Acknowledgement. Thanks to Kazuya Hamaguchi and Hisayoshi Fujimoto for useful discussions. We also want to thank four anonymous reviewers for their helpful comments.

References

1. Augusto, J.C.: Ambient intelligence: the confluence of ubiquitous/pervasive computing and artificial intelligence. In: Schuster, A. (ed.) Intelligent Computing Everywhere, pp. 213–234 (2007)
2. Augusto, J.C., Nugent, C.D.: The use of temporal reasoning and management of complex events in smart homes. In: 16th European Conference of Artificial Intelligence (ECAI 2004), pp. 778–782 (2004)
3. Bajo, J., Tapia, D.I., Rodr-iguez, S., Corchado, J.M.: Planning agent for health care: ambient intelligence in practice. In: Encyclopedia of Artificial Intelligence, pp. 1316–1322 (2008)
4. Bikakis, A., Antoniou, G.: Defeasible contextual reasoning with arguments in ambient intelligence. IEEE Transactions on Knowledge and Data Engineering 22(11), 1492–1506 (2010)
5. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: 2nd International Conference on Knowledge Discovery and Data Mining (KDD 1996), pp. 226–231 (1996)
6. Hattori, M., Cho, K., Ohsuga, A., Isshiki, M., Honiden, S.: Context-aware agent platform in ubiquitous environments and its verification tests. In: IEEE International Conference on Pervasive Computing and Communications (PerCom 2003), pp. 547–552 (2003)
7. Jürgen, B., Vlad, C., Marc, L., Friedemann, M., Michael, R.: Social, economic and ethical implications of ambient intelligence and ubiquitous computing. In: Weber, W., Rabaey, J., Aarts, E. (eds.) Ambient Intelligence, pp. 5–29 (2005)
8. Mark, W.: The computer for the 21st century. ACM SIGMOBILE Mobile Computing and Communications Review 3(3) (1999)
9. Ohsuga, A., Nagai, Y., Irie, Y., Hattori, M., Honiden, S.: PLANGENT: An approach to making mobile agents intelligent. IEEE Internet Computing 1(4), 50–57 (1997)
10. Patkos, T., Bikakis, A., Antoniou, G., Papadopouli, M., Plexousakis, D.: Distributed AI for Ambient Intelligence: Issues and Approaches. In: Schiele, B., Dey, A.K., Gellersen, H., de Ruyter, B., Tscheligi, M., Wichert, R., Aarts, E., Buchmann, A. (eds.) AmI 2007. LNCS, vol. 4794, pp. 159–176. Springer, Heidelberg (2007)
11. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.-C.: Prefix-span: mining sequential patterns efficiently by prefix-projected pattern growth. In: 17th International Conference on Data Engineering (ICDE 2001), pp. 215–224 (2001)
12. Ramos, C., Augusto, J.C., Shapiro, D.: Ambient intelligence - the next step for artificial intelligence. IEEE Intelligent Systems 23(2), 15–18 (2008)
13. Xuemei, L., Gang, X., Li, L.: RFID based smart home architecture for improving lives. In: Proceedings of the 2nd International Conference on Anti-Counterfeiting, Security, and Identification (ASID 2008), pp. 440–443 (2008)

Author Index

Ab Aziz, Azizi 83
Artel, Arsun 113
Artikis, Alexander 202
Arulanandam, Rexy 36

Baarslag, Tim 143
Bijker, Jaco 460
Brey, Eric M. 113

Chang, Tao-Hsuan 234
Cinar, Ali 113
Cornejo, Alejandro 19

Davidsson, Paul 97
de la Hoz, Enrique 159, 424
Delgado-Roman, M. del Carmen 448
Demazeau, Yves 218
Dillon, Denise 361
d'Inverno, Mark 263

Ehlers, Elize 460
Ekron, Kieron 460

Frantz, Christopher 275
Furusawa, Yu 508

Ghoneim, Ayman 51
Gini, Maria L. 373
Goh, Tze Jui 361

Hanada, Kenta 174
Hattori, Hiromitsu 484
Hindriks, Koen 143
Hirayama, Katsutoshi 174
Hishiyama, Reiko 508, 520
Hoogendoorn, Mark 373, 385
Hsu, Jane Yung-jen 234, 299
Hu, Cuiyun 287

Iwasaki, Atsushi 4

Jevinger, Åse 97
Jørgensen, Bo Nørregaard 218
Jonker, Catholijn 143

Kagifuku, Ryuya 68
Klein, Mark 159, 218, 424
Kloukinas, Christos 311
Kuo, Yen-Ling 234, 299

Lekeas, George K. 311
Lopez-Carmona, Miguel A. 159, 424
Luck, Michael 263
Lynch, Nancy 19

Maheswaran, Rajiv T. 247
Mailler, Roger 187
Mao, Xinjun 287
Marsa-Maestre, Ivan 159, 424
Matsubara, Shigeo 68
Mehdizadeh, Hamidreza 113
Mihailescu, Radu-Casian 472
Minato, Shin-Ichi 4

Nakajima, Yuu 484
Noda, Itsuki 349
Noriega, Pablo 263
Norman, Timothy J. 323
Nowostawski, Mariusz 275

Okaya, Masaru 496
Ossowski, Sascha 472

Pedell, Sonja 398
Peng, Yen-Bo 436
Persson, Jan A. 97
Pitt, Jeremy 202
Purvis, Martin K. 275
Purvis, Maryam 36

Rodriguez-Aguilar, Juan A. 263
Rus, Daniela 19

Sakurai, Yuko 4
Sanchez, Romeo 247
Sando, Makoto 520
Savarimuthu, Bastin Tony Roy 36
Schaumeier, Julia 202
Sen, Sandip 2
Sharpanskykh, Alexei 129

Sierra, Carles 263, 448
Smith, Melanie 187
Sørensen, Jan Corfixen 218
Sonenberg, Liz 3
Song, Insu 361
Soo, Von-Wun 436
Stathis, Kostas 311
Sterling, Leon 398
Sugiki, Yuta 508
Sung, Min 361
Sycara, Katia 323
Szekely, Pedro 247

Takahashi, Tomoichi 496
Tambe, Milind 1
Toniolo, Alice 323
Treur, Jan 83, 129, 385, 410

Ueda, Suguru 4

van der Wal, C. Natalie 83
van Lambalgen, Rianne 373, 385
Vasirani, Matteo 472
Volkov, Mikhail 19

Wakaki, Toshiko 336

Yokoo, Makoto 4

Zhou, Huiping 287