Luis Antunes
Mario Paolucci
Emma Norling (Eds.)

LNAI 5003

Multi-Agent-Based Simulation VIII

International Workshop, MABS 2007
Honolulu, HI, USA, May 2007
Revised and Invited Papers

Springer

Lecture Notes in Artificial Intelligence 5003

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

Luis Antunes Mario Paolucci
Emma Norling (Eds.)

Multi-Agent-Based Simulation VIII

International Workshop, MABS 2007
Honolulu, HI, USA, May 15, 2007
Revised and Invited Papers

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

Luis Antunes
Universidade de Lisboa, Faculdade de Ciências
Grupo de Estudos em Simulação Social (GUESS)
Campo Grande, 1749-016 Lisboa, Portugal
E-mail: xarax@di.fc.ul.pt

Mario Paolucci
Institute for Cognitive Science and Technology
Laboratory of Agent Based Social Simulation
Via San Martino della Battaglia 44, 00185 Rome, Italy
E-mail: mario.paolucci@gmail.com

Emma Norling
Centre for Policy Modelling, Manchester Metropolitan University
Aytoun Building, Aytoun Street, Manchester, M1 3GH, UK
E-mail: norling@acm.org

Library of Congress Control Number: 2008931477

CR Subject Classification (1998): I.2.11, J.4, I.6

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-540-70915-0 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-70915-2 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12442881 06/3180 5 4 3 2 1 0

Preface

This volume contains selected papers that were presented at the eighth international workshop on Multi-Agent-Based Simulation (MABS 2007), a workshop co-located with the 6th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), held in Honolulu, Hawaii, on May 15, 2007. These papers have been revised and extended, based on discussions at the workshop, and reviewed once more.

Agent technology is now a mature paradigm of software engineering. Complex systems, which are irreducible to their components in isolation, are instead heavily characterized by the interaction between their components. Agent-based simulation is the natural way to model systems with a focus on interaction, and the circle closes by considering how the social sciences show this kind of complexity. The focus of this workshop series lies in this confluence of social sciences and multi-agent systems.

Simulation has been proposed by Axelrod[1] as a *third way of doing science*, in contrast with deduction and induction: generating data that can be analyzed inductively, but coming from a rigourously specified set of rules rather than direct measurement of the real world. In this sense, to simulate a phenomenon is to generate it – constructing artificial (agent) societies. This in turn leads to questions that have already been asked for human societies. Computer scientists have adopted general terms like emerging behavior, self-organization, and evolutionary theory; even specific social terms such as norms, reputation, trust, tags, institutions; but all of them in an intuitive manner. Multi-agent system researchers have started to develop agents with *social* abilities and complex *social* systems, recognizing that the design of open multi-agent systems can benefit from abstractions analogous to those employed by our robust and relatively successful societies and organizations. However, most of these systems lack the foundation of the *social sciences*. MABS fills the gap, posing as a point of encounter between these diverging forces.

As this workshop marked the 10th anniversary of the initiation of the workshop series, Jaime Simão Sichman, who was one of the founders of the series, was invited to present a historical perspective. We start the volume with the paper which arose from his presentation. The remainder of the papers are organized into three broad areas.

In the first section, we present four papers on architectures, methodological or technical issues. Vidit Bansal, Ramachandra Kota, and Kamalakar Karlapalem focus on scalability issues for systems that should be able to accommodate a realistic number of agents to describe situations like a disaster scenario for a large city using a database approach that shifts representation between micro

[1] *Advancing the Art of Simulation in the Social Sciences*, in Conte R., Hegselmann R. and Terna P. (eds.), Simulating Social Phenomena, Berlin: Springer, 1997.

and macro levels; Dawit Mengistu, Paul Davidsson, and Lars Lundberg connect multi-agent systems to the grid and propose a supportive middleware that brings performance improvement to MABS on the grid; Vera Lucia da Silva, Maria das Graças, Bruno Marietto, and Carlos H. Costa Ribeiro propose a multi-agent architecture based on principles from Luhmann's social theorization; H. Van Dyke Parunak, Sven Brueckner, Danny Weyns, Tom Holvoet, Paul Verstraete, and Paul Valckenaers present a review on the theme of polyagents – models where the representation of an agent is augmented by a "swarm of ghosts" – comparing them with similar approaches.

The second part proposes two papers on the themes of teams, learning, and education. Maartje Spoelstra and Elizabeth Sklar simulate group learning, modelling characteristics outlined in pedagogical literature and comparing the outcomes of simulated learners operating with different goal structure, also in relation with factors such as group size and rewards. Yuqing Tang, Simon Parsons, and Elizabeth Sklar examine the relationship between the investment that a society makes in education, exploring the effects of different parameter settings on the education investment of a society and the resulting economic growth.

In the third part we include papers on economics, trust, and reputation. Tibor Bosse, Martijn C. Schut, Jan Treur, and David Wendt tackle the renowned question of altruistic behavior – the claim is that to be able to make reasonable decisions, an agent needs a cognitive system for intertemporal decision making, in relation to a model of the environment; James G. McCarthy, Tony Sabbadini, and Sonia R. Sachs simulate the effects of a disruptive technological change in an industry setting; and, finally, Isaac Pinyol, Mario Paolucci, Jordi Sabater-Mir, and Rosaria Conte discuss the dynamics of information in a system where reputation and image are moderated by retaliation.

To conclude, let us consider a measure of how this challenge was met by AAMAS researchers. MABS 2007 was in the top five workshops in AAMAS (out of a total of 19) for number of registrations. The workshop had 20 submitted papers, each reviewed by 3 anonymous Program Committee members, showing an extremely high standard, which was reflected in the relatively high (for this workshop series) acceptance rate – 50% of the submissions were accepted for the workshop and then subsequently revised based on the workshop discussion, and reviewed once more. Already we are seeing signs that the workshop continues to strengthen, with MABS 2008 extended to a 1.5-day workshop to accommodate more papers with a lower acceptance rate. It was our pleasure to have Jaime Sichman discussing further the strengths of this series in his invited paper. As one of the founders and Chairs of the first MABS workshop in 1998, we asked him to present his personal view on how the field has evolved over the 10 years of the workshop series. We look forward with pleasure to ongoing developments in this community.

April 2008

Luis Antunes
Mario Paolucci
Emma Norling

Organization

The 8th International Workshop on Multi-Agent-Based Simulation (MABS 2007) was organized by the *Institute of Sciences and Technologies of Cognition* – CNR, Italy; the *Centre for Policy Modelling*, UK; and *GUESS/LabMAg/Universidade de Lisboa*, Portugal.

Workshop Chairs

Luis Antunes
Mario Paolucci
Emma Norling

Program Committee

Luis Antunes
Joao Balsa
Cristiano Castelfranchi
Sung-Bae Cho
Helder Coelho
Rosaria Conte
Paul Davidsson
Alexis Drogoul
Bruce Edmonds
Nigel Gilbert
Laszlo Gulyas
David Hales
Wander Jager
Marco Janssen
Satoshi Kurihara
Jorge Louçã
Akira Namatame
Paulo Novais
H. Van Parunak
Juan Pavon Mestras
Emma Norling
Mario Paolucci
Juliette Rouchier
David Sallach
Keith Sawyer
Jaime Sichman
Elizabeth Sklar
Liz Sonenberg
Keiki Takadama
Oswaldo Teran
Takao Terano
Jan Treur
Harko Verhagen

Additional Reviewers

Annerieke Heuvelink
Gennaro Di Tosto

Table of Contents

MABS Celebrates Its 10th Anniversary!

Jaime Simão Sichman

Laboratório de Técnicas Inteligentes (LTI)
Escola Politécnica (EP)
Universidade de São Paulo (USP)
jaime.sichman@poli.usp.br

Abstract. The MABS (multi-agent based simulation) workshop had its first edition in 1998, and hence is celebrating its 10th anniversary. The aim of this paper is to present an historical perspective of the event, by showing some details about each of its editions. Additionally, a statistical summary of submitted and accepted papers is presented.

Keywords: Multi-agent based simulation, MABS workshop.

1 Introduction

In the late 90's, some researchers from the multi-agent systems domain had the idea of creating an international workshop on multi-agent based simulation, whose main goal was to serve as a bridge between social and economic sciences researchers, on one hand, and computer science, artificial intelligence and multi-agent systems experts on the other. The result has been a workshop series called MABS - Multi-Agent-Based Simulation (http://www.pcs.usp.br/~mabs/).

The MABS community is closely related to research in the social simulation domain. The main world scientific associations related with social simulation are the following:

- ESSA - European Social Simulation Association (http://www.essa.eu.org/), founded in 2003.
- NAACSOS - North American Computational Social and Organization Sciences (http://www.casos.cs.cmu.edu/naacsos/sections.php), also founded in 2003.
- PAAA - Pacific Asian Association for Agent-based Approach in Economic & Social Complex Systems (http://www.paaa.econ.kyoto-u.ac.jp/), founded in 2001.

The above-mentioned associations have each organized an annual event since their foundation: the ESSA and NAACSOS events have the same name as the association, whereas the PAAA event is called the International Workshop on Agent-based Approaches in Economic and Social Complex Systems (AESCS). In 2006, the three associations together launched the World Congress on Social Simulation (WCSS), which will be held every two years, and whose second edition will happen in 2008.

L. Antunes, M. Paolucci, and E. Norling (Eds.): MABS 2007, LNAI 5003, pp. 1–7, 2008.

The MABS workshop series is hence the eldest workshop still active in the area. It is celebrating its 10th anniversary in 2008, and some details of its history are described below.

2 MABS History

When it was created, the idea was to associate the MABS workshop to the most prestigious MAS conference at that time, called ICMAS (International Conference on Multiagent Systems). As the ICMAS conference happened every two years, the first editions of the workshop were also biannual.

The first workshop of the series, *MABS 1998*, was associated with ICMAS 1998. The conference was held in Paris, France, 3-7 July 1998. The organizers of this first workshop were Nigel Gilbert, (University of Surrey, UK), Rosaria Conte (ISTC/CNR, Italy), and Jaime Simão Sichman (University of São Paulo, Brazil). The workshop was structured in three sessions, presented in the afternoons of 4-6 July 1998. At the closing session, a round table on "Representing cognition for social simulation" was presented by Nigel Gilbert, Rosaria Conte, Jim Doran (University of Essex, UK) and Scott Moss (Manchester Metropolitan University, UK).

Scott Moss and Paul Davidsson (Blekinge Institute of Technology, Sweden) were the organizers of the second edition, *MABS 2000,* which was held two years later in Boston, USA, associated with ICMAS 2000 (4th International Conference on Multiagent Systems), 8-12 July 2000. The presentation of the papers was structured around six sessions, held on 8-9 July, starting in the afternoon of the first day: (i) structural issues in model design, (ii/iii) simulation and applications (two sessions), (iv) simulating social relations, (v) simulating observed and prospective social processes and (vi) simulation formalisms. Scott Moss opened the event giving an introductory talk, called "MAS $\cup$ ABSS $\supseteq$ MABS", and an open discussion panel closed the activities.

In 2002, the ICMAS conference was merged with two other agent-related conferences: AA (International Conference on Autonomous Agents) and ATAL (International Workshop on Agent Theories, Architectures, and Languages). This new joint annual conference was named AAMAS (International Conference on Autonomous Agents and Multiagent Systems), and was intended to be the most respected international forum for research in the theory and practice of autonomous agents and multiagent systems. From this year forward, the MABS workshop has become an annual event.

MABS 2002 was held on 15 July 2002, as part of AAMAS 2002 (1st. International Conference on Autonomous Agents and Multiagent Systems), in Bologna, Italy, 15-19 July 2002. This time, the event chairs were Jaime Simão Sichman, François Bousquet (CIRAD, France) and Paul Davidsson. The presentations were classified in four sessions: (i) emergence, alliances and groups, (ii/iii) applications (two sessions), and (iv) platforms and languages. The event also had an invited talk given by Alexis Drogoul (University of Paris VI, France), the title of which was "MABS: Past, Current and Future Research".

David Hales (Manchester Metropolitan University, UK), Juliette Rouchier (GRECAM/CNRS, France), Bruce Edmonds (Manchester Metropolitan University, UK), Emma Norling (University of Melbourne, Australia) and Roberto Pedone

(ISTC/CNR, Italy) chaired *MABS 2003*. This edition was held on 14 July, in Melbourne, Australia, associated with AAMAS 2003, 14-18 July 2003. It was organized in three sessions: (i) MABS techniques for MAS, (ii) economics, exchange and influence in virtual worlds and (iii) MABS techniques for real world modeling. The workshop started with an invited talk given by Giovanna Di Marzo Serugendo (University of Geneva, Switzerland) about “Engineering Emergent Behaviour: A Vision”.

On 19 July 2004, a joint workshop called *MAMABS* 2004 was held in New York, USA, during AAMAS 2004, 19-23 July 2004. This event joined two workshop proposals submitted to the AAMAS 2004 program: MABS – proposed by Paul Davidsson and Keiki Takadama (University of Electro-Communications, Japan) – and Multi-Agent Simulation - proposed by Brian Logan (University of Nottingham, UK) and Les Gasser (University of Illinois at Urbana-Champaign, USA). The papers were divided in five sessions: (i) simulation of MAS, (ii) techniques and technologies, (iii) methodologies and modeling, (iv) social dynamics and (v) applications.

Luis Antunes (University of Lisbon, Portugal), and Jaime Simão Sichman chaired the sixth edition of the workshop, *MABS 2005*, which was held in Utrecht, the Netherlands, on 25 July 2005 as an associated workshop of AAMAS 2005, 25-29 July 2005. This time, the program was organized in four sections: (i) coalition emergence, (ii) theories and models, (iii) applications and (iv) environments. The workshop also included an invited talk by Scott Moss titled “Having Fun Being Useful”.

The seventh edition, *MABS 2006*, was held in Hakodate, Japan, on 8 May 2006 during AAMAS 2006, 8-12 May 2006. The organizers, Luis Antunes and Keiki Takadama, divided the papers presentations in seven sections: (i) empirical cross studies, (ii) social dependence and decision theory, (iii) experimental ecology, (iv) learning, (v/vi) foundations and methodologies (two sessions) and (vii) experimental economics. Takao Terano (Tokyo Institute of Technology, Japan) gave the invited talk of this edition, whose title was “Exploring the Vast Parameter Space of Multi-Agent Based Simulation”.

MABS 2007 occurred on 15 May 2007, associated with AAMAS 2007, which was held in Honolulu, USA, 14-18 May 2007. The event chairs were Mario Paolucci (ISTC/CNR, Italy), Emma Norling (Manchester Metropolitan University, UK) and Luis Antunes. The workshop was organized in four sections: (i) architectures, (ii), teams, learning and education and (iii/iv) economy, trust and reputation (two sessions). An invited talk, titled “Celebrating MABS’ 10th Anniversary: Some History and Perspectives” was given by Jaime Simão Sichman and this is the reason why this paper makes part of this volume.

Finally, *MABS 2008* is already scheduled to happen on 12-13 May 2008 in Lisbon, Portugal, during AAMAS 2008, 12-16 May 2008. The ninth edition chairs, Nuno David (Lisbon University Institute, Portugal) and Jaime Simão Sichman, have organized the workshop along seven sessions: (i) simulation of economic behaviour, (ii) applications, (iii) infrastructure and technologies, (iv/v) methods and methodology (two sessions) and (vi/vii) simulation of social behaviour (two sessions). An invited talk and a closing round table are also foreseen.

Since its first edition, in addition to the local proceedings, MABS has had a post-proceedings volume published in LNAI series [5, 4, 6, 3, 2, 7, 1]. These volumes contain revised versions of the papers presented in the workshops. Authors were requested to take into account the discussion and suggestions made during their presentation. Some of these volumes contain also some additional invited papers, including the invited speakers contributions, and in some cases, contributions from others who attended the workshop.

3 MABS Statistics

Table 1 presents the number of submitted and accepted papers in all MABS editions, as well as the final acceptance rate and the duration of the workshop (in full days). In the 2006 edition, three short papers were also presented, but they were not considered for statistical purposes.

Table 1. Number of submitted and accepted papers

Year	Local	Duration	Submitted	Accepted	Acceptance rate (%)
1998	FR	1,5	50	15	30,0
2000	US	1,5	25	15	60,0
2002	IT	1	26	12	46,2
2003	AU	1	27	13	48,1
2004	US	1	32	20	62,5
2005	NL	1	28	12	42,9
2006	JP	1	25	12	48,0
2007	US	1	20	11	55,0
2008	PT	1,5	44	16	36,4

One can notice that the acceptance rate of the workshop has been on average 48%, with exceptions in some years. On the other hand, the average number of submissions was around 30 papers, whereas the average number of accepted papers was around 14. Significantly, the first and last editions were the ones where there was an outstanding number of submitted papers: 50 in 1998 and 44 in 2008. Regarding the first workshop, this may be explained by the fact that the submissions were then made by abstracts. With respect to the last edition, there may have been an augmented interest for the field. This can be also observed by the workshop duration, since in 2008 the workshop has gained an extra period of half day, which has occurred for the first time since it has been associated with AAMAS.

Table 2 presents the first author's country affiliation for every accepted paper in each MABS edition. It was considered in this table only the papers that effectively were reviewed for the workshop: additional invited papers for the post-proceedings volumes, such as the contributions of the invited speakers, were not taken into account.

Some interesting facts can be inferred from this table. Considering all the workshop editions, the 126 accepted papers' main authors came from 22 different countries in almost every inhabited continent (North and South America, Europe, Asia and Oceania). However, authors from four countries have been responsible for more than

Table 2. First author's country affiliation

Year	**Local**	**AU**	**BE**	**BR**	**CA**	**CH**	**DE**	**E S**	**FR**	**IN**	**I T**	**HK**	**J P**	**MX**	**NL**	**P T**	**S E**	**S G**	**TR**	**UK**	**U S**	**VE**	**V I**	**TOTAL**
1998	FR								5		3		1	1			1			3	1			15
2000	US	1				1	1		3		1	1				1	1			1	3	1		15
2002	IT			2	1				1		1				2	2	1			2				12
2003	AU	1		2					3		1		4							1	1			13
2004	US	1	1		1						2		5		2		1	1		2	4			20
2005	NL			2	1		1	1	1						1	2				1	1		1	12
2006	JP			1					1		1		4			1				2	2			12
2007	US			1				1		1			1		2		1			1	3			11
2008	PT			1				1	1	1			1		2	1	1		1	3	3			16
TOT		3	1	9	3	1	2	3	15	2	9	1	16	1	9	7	6	1	1	16	18	1	1	126
%		2.4	0.8	7.1	2.4	0.8	1.6	2.4	11.9	1.6	7.1	0.8	12.7	0.8	7.1	5.6	4.8	0.8	0.8	12.7	14.3	0.8	0.8	100

the half of the total number of accepted papers (65 papers, corresponding to 51,6%): USA (18 papers, 14,3%), UK (16 papers, 12,7%), Japan (16 papers, 12,7%) and France (15 papers, 11.9%). Moreover, if we add authors of three additional countries – Brazil (9 papers, 7.1%), Italy (9 papers, 7.1%), and the Netherlands (9 papers, 7.1%) – we reach to more than 70% of the accepted contributions (92 papers, 73.0%). These numbers may suggest at a first glance that most work on MABS is carried on by well-established research groups both in universities and in research centers in these countries. However, as stated before, only the first author affiliation was taken into account to create this table, and many papers are co-signed by colleagues working in different countries. This last fact shows that MABS research has also helped to shorten the distance between research groups in different countries. Finally, regarding the publication regularity, the only country that has had papers accepted in all MABS editions was the UK. In addition, nine different countries have had papers published in more than a half of the total editions of the workshop (five or more): the above-mentioned seven countries, Sweden (papers in six editions) and Portugal (papers in five editions)

4 MABS Topics

As stated in the introduction, the goal of the MABS workshop series was to serve as a link between researchers from social/economic sciences and computer science, artificial intelligence and multi-agent systems. This fact has been reflected in the various different topics covered in the workshops' sessions. It is out of scope of this paper to make a deep analysis of this difficult issue.

Adopting a rather very simplistic vision, however, one can easily see at least four different classes of topics that have been dealt with in the several workshops' editions:

- MABS as a *means* for reaching conclusions in social/economic sciences
    - MABS methods and/or methodologies
    - Description/analysis of social/economic phenomena using MABS
- MABS as a research *goal* for building better computer based systems
    - MABS tools, languages and computer environments
    - Problem solving using MABS social/economic metaphors

Obviously, these different classes may not be or-exclusive ones, i.e., if we analyze in more detail the papers motivations some overlapping may occur. However, these classes may serve as a first tentative to differentiate the huge number of topics that were presented in the workshops editions.

5 Conclusions

In this paper, a brief history of the 10 years of the MABS workshop was presented. In addition to the formal presentations and technical discussions, the goal of a scientific workshop is to connect people (professors, students, researchers) working around the same themes. I am sure that several european/international research or collaboration projects in multi-agent based simulation, as well as several PhD subjects on the field, were born during some coffee breaks or evening drinks in MABS workshops.

I'd like to finish by thanking Nigel Gilbert, Rosaria Conte, Scott Moss, Paul Davidsson, François Bousquet, David Hales, Juliette Rouchier, Bruce Edmonds, Emma Norling, Roberto Pedone, Keiki Takadama, Luis Antunes, Mario Paolucci and Nuno David, chairs of the several editions of the MABS workshop, who have undoubtedly given their experience, knowledge and effort to organize the event in all these 10 years.

Considering the great quality and amount of research in the multi-agent based simulation domain, I'm pretty sure that a paper describing the next 10 years of the workshop will be written by a colleague in the future!

References

1. Antunes, L., Takadama, K. (eds.): MABS 2006. LNCS (LNAI), vol. 4442. Springer, Heidelberg (2007)
2. Davidsson, P., Logan, B., Takadama, K. (eds.): MABS 2004. LNCS (LNAI), vol. 3415. Springer, Heidelberg (2005)
3. Hales, D., Edmonds, B., Norling, E., Rouchier, J. (eds.): MABS 2003. LNCS, vol. 2927. Springer, Heidelberg (2003)
4. Moss, S., Davidsson, P. (eds.): MABS 2000. LNCS (LNAI), vol. 1979. Springer, Heidelberg (2001)
5. Sichman, J.S., Conte, R., Gilbert, N. (eds.): MABS 1998. LNCS (LNAI), vol. 1534. Springer, Heidelberg (1998)
6. Sichman, J.S., Bousquet, F., Davidsson, P. (eds.): MABS 2002. LNCS (LNAI), vol. 2581. Springer, Heidelberg (2003)
7. Sichman, J.S., Antunes, L. (eds.): MABS 2005. LNCS (LNAI), vol. 3891. Springer, Heidelberg (2006)

System Issues in Multi-agent Simulation of Large Crowds

Vidit Bansal, Ramachandra Kota, and Kamalakar Karlapalem

Center for Data Engineering
International Institute of Information Technology
Hyderabad, India
vidit@research.iiit.ac.in, ramachandra@students.iiit.ac.in,
kamal@iiit.ac.in

Abstract. Crowd simulation is a complex and challenging domain. Crowds demonstrate many complex behaviours and are consequently difficult to model for realistic simulation systems. Analyzing crowd dynamics has been an active area of research and efforts have been made to develop models to explain crowd behaviour. In this paper we describe an agent based simulation of crowds, based on a continuous field force model. Our simulation can handle movement of crowds over complex terrains and we have been able to simulate scenarios like clogging of exits during emergency evacuation situations. The focus of this paper, however, is on the scalability issues for such a multi-agent based crowd simulation system. We believe that scalability is an important criterion for rescue simulation systems. To realistically model a disaster scenario for a large city, the system should ideally scale up to accommodate hundreds of thousands of agents. We discuss the attempts made so far to meet this challenge, and try to identify the architectural and system constraints that limit scalability. Thereafter we propose a novel technique which could be used to richly simulate huge crowds.

1 Introduction

Simulating large crowds in rescue simulation systems throws up many challenges. For one, crowd events and their associated phenomenon are difficult to model. Crowds demonstrate a variety of emergent behaviours based on the behaviour of individuals in the crowd. Crowd dynamics have been extensively studied in the past and various socio-psychological and physiological theories have been put forth to explain crowd behaviour. The complexity also stems from the fact that compared to a simulation with limited parameters, the level of detail that could be incorporated in the model for a realistic social simulation can be quite high. Analyzing crowd dynamics has been an active area of research. Different types of crowd simulation systems have been developed, ranging from those based on force-modelling approaches [1, 2] to cellular automata based simulations [7, 8, 9] and rule-based architectures [5, 6]. Recently, many agent-based architectures have been proposed [13, 18, 19]. The multi-agent paradigm is adequately suited

L. Antunes, M. Paolucci, and E. Norling (Eds.): MABS 2007, LNAI 5003, pp. 8–19, 2008.

to a crowd simulation application. Social factors can be better modelled as human characteristics can be objectively mapped to agent behaviour.

In this paper we present a multi-agent based crowd simulation system developed on a continuous field force model. Our model supports - heterogeneity in agents to model the demographics of a population, a complex navigational behaviour including obstacle avoidance and navigation in a terrain with partial information and flocking behaviour. As scalability is a vital criterion for the system to realistically model a disaster scenario for a large city with hundreds of thousands of people, we evaluate the issue of scalability for multi-agent based crowd simulation systems. The organization of the paper is as follows. We discuss the environment model that we use for our simulations in Section 2. We then move on to propose a novel technique for simulation of large crowds in Section 3. We present our results in Section 4.

2 Background and Related Work

Amongst the different approaches to model crowd behaviour, one which has been widely put to use is the force based model developed by Dirk Helbing et al.[1, 2]. This model tries to simulate the motion of each individual in a crowd (henceforth referred to as a civilian) under forces that are exerted by other civilians and inanimate objects. Each civilian feels, and exerts on others, two kinds of forces, “social” and physical. The social forces are not exactly physical forces such as a *push* or a *pull*; they reflect the intentions of a civilian to avoid collisions and to move in a chosen direction. The movement of the civilian can be tracked by equations defining his motion under the sum of all forces. Further refinements were suggested in [3]. One problem with Helbing’s model is with its computational complexity. Every civilian must be tracked with respect to every other civilian to calculate the net force acting on it. Alternative approaches proposed[3] avoid computing every agent’s effect on all the others. While the model has been successful in simulating a number of crowd behaviours demonstrated by real-life crowds, particularly arch formation at exits in egress situations, scalability issues were not tackled. The model by itself does not incorporate navigation models, or any form of social interaction.

Cellular Automata based models have also been proposed most notably in [7, 8, 9]. Cellular automata based simulations also model forces on a civilian, but are discrete in space and time. Efforts have been made to build multi-agent based systems for crowd simulation. Prominent among these are [11, 12, 13, 18, 19]. However, scalability issues in multi-agent based crowd simulation systems have not been rigorously analyzed.

2.1 Background

We have attempted to develop a crowd simulation model which preserves the granularity of simulation at the individual level and at the same time is scalable and can richly simulate behaviour of huge crowds. The problem that we are trying to address is scaling up in terms of the number of civilians that can be simulated on a single computer given a set of system constraints. Typically agents are implemented as threads. For any multi-agent based architecture, there are two key system constraints – the size of the RAM, which determines the number of agents that can be kept in the main memory and the number of threads that can be handled by the processor. Threads by themselves require both the main memory and the CPU usage, thereby imposing a dual constraint. We show the results of our experiments on our simulation model described below.

2.2 Description of the Experimental Model

Our simulation model is based on the continuous force model by Helbing [2]. The map of the environment is a two dimensional continuous grid. A complex terrain can be specified with obstacles and walls. There may be multiple safety exits towards the periphery of the map. Civilians are placed at different locations on the map at the beginning of the simulation. Each civilian c_i has body mass m_i and a maximum velocity v_i^{max}. The civilian occupies a circular area of size r_i which is proportional to the the square root of its body mass m_i . Each civilian also has a defined visibility range e which determines a sphere of influence. A civilian is affected by all objects i.e both by other civilians and walls within their visibility range. A civilian experiences a repulsive force from other civilians in its visibility. This force decreases as the distance increases and is null beyond the visibility. The civilian experiences a similar repulsive force from a wall or a blockade. The repulsive force along the x-direction on civilian i due to civilian j has the form:

$$f_{ij}^x = \begin{cases} C_x/(x_j - x_i) & \text{if } |x_j - x_i| \leq e \\ 0 & \text{otherwise} \end{cases}$$

Walls exert a force f_{iw}^x on civilian c_i :

$$f_{iw}^x = W_x/d$$

where d is the distance of the closest point of the wall from the civilian. Obstacles avoidance is achieved through the application of a repulsive force f_{io}^x on civilian c_i :

$$f_{io}^x = O_x/d_o$$

where d_o is the distance of the obstacle from the civilian. C_x , W_x and O_x are constants and are set to ensure appropriate ranges for the forces.

The civilians may have partial or complete knowledge of the terrain. Their knowledge is in the form of a sequence of landmarks or points they might remember to reach the safety exits. This sequence can also be updated by external factors such as a communication from another agents. The civilian agent can also follow another agent. In an emergency evacuation scenario there might rescue agents to coordinate rescue activity. Let F_{dir}^x be the force which determines the direction the agent wants to move in.

Therefore the total force acting on the civilian c_i along the x-direction is

$$F_i^x = \sum_j f_{ij}^x + \sum_w f_{iw}^x + \sum_o f_{io}^x + F_{dir}^x$$

Similarly

$$F_i^y = \sum_j f_{ij}^y + \sum_w f_{iw}^y + \sum_o f_{io}^y + F_{dir}^y$$

The model described above is the basic minimum model, and it can be enriched by specification of other parameters. For instance, we have also modelled injuries, simple communication models and specialized rescue agents. In this paper however we'd like to focus on the scalability aspects of our system.

For our experiments, we used MASON multi-agent simulation library [22]. MASON is coded in Java and provides a comprehensive functionality set for simulations. We conducted the experiments on a Linux server with the following configuration: 2100 MHz CPU, with a 512 MB RAM.

2.3 A Micro-agent Architecture

We start with a simple architecture. In this architecture, each civilian is modelled as an agent. Hence, each civilian has a thread dedicated to it. The CPU would run as many threads as there are civilians. In every simulation step, we compute the movement of each civilian agent under the influence of the forces on it. Also, each agent stores its state information and hence needs memory. If the memory required for one civilian is M_x, then the total memory required would be nM_x where n is the number of civilians. Henceforth we refer to this architecture as the Micro-Agent architecture in this paper. With the model described in the previous subsection, we are able to show common egress behaviour of crowds from an enclosed room. Arch formation at the exit is shown in figure 1(b).

We conducted further experiments trying to scale up by increasing the number of civilians in the map. For this set of experiments our simulation environment map consisted of a huge hall with one exit and a few randomly placed obstacles. We observed that an increase in the number of agents (which is equivalent to the number of threads) corresponded to an almost linear increase in the time to the run the simulation for a fixed number of cycles. This was expected as a civilian agent has to consider only the agents within its sphere of influence as against every other civilian in Helbing's model. But the simulation hits a major block

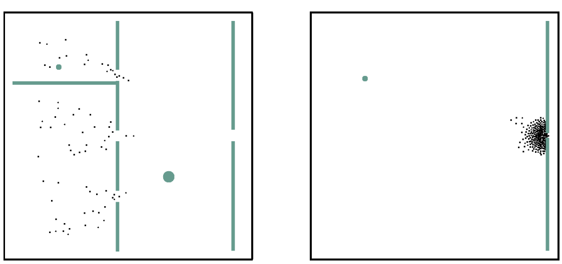


Fig. 1. Evacuation from a hall:: Fig (a): Civilians(black dots) rushing towards the exits in an evacuation scenario Fig (b): Arch formation at exit : A common egress behaviour exhibited by crowds

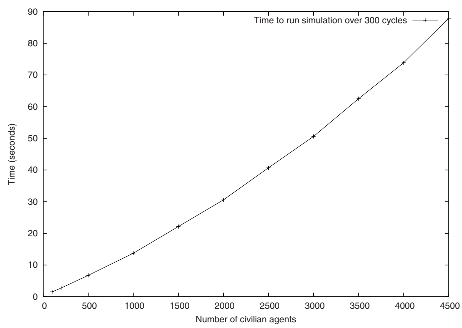


Fig. 2. Time taken to run simulation for 300 cycles

when it comes to the number of threads that can be run on the system. In our experiments we hit a top of approximately 5000 threads initially. The maximum number of threads that can be can be simulated depend on the operating system and the Java Virtual Machine(JVM). The available stack size on the system is one the factors that determine the maximum number of threads that can be run. The default stack size for a thread is 512Kb on Linux (kernel versions 2.6) but it can be set to a lower value. In our experiment we were able to set the minimum value of 105Kb. With this setting we were able touch close to 9600 threads. Note that these values were specific to our experiment's environment and these could vary under different settings. There are however two important aspects we would like to convey – *i*) that there is an upper limit on the number of threads that can be run on a system with given specifications, and *ii*) the more complex the application is, the fewer number of threads it would be able to support. This is because given a fixed stack size on a system, if the stack space occupied by a thread increases, fewer threads would be accommodated in the stack.

3 A Macro-micro Architecture

3.1 Using a Database

We see that in the Micro-Agent architecture, the limited resources put a constraint on the number of civilians that can be created and sustained in the environment. The limiting factor is the size of the main memory. The standard approach to counter a memory limitation would involve getting an external memory with a paging or swapping scheme. This approach was successfully demonstrated by Yamamoto and Nakamura [20] in an distributed electronic commerce scenario.

We could conceptualize a solution to this problem, overcoming the main memory limitations by using a database, with each civilian existing as a tuple in a *civilian* table in a database. In each time cycle, the simulation program would retrieve each civilian's details from the database, get its location, identify its neighbours, compute the net force on the civilian and calculate its next step and update the corresponding tuple of the civilian table. It would then move onto the next civilian. After one complete scan it could update positions of all civilians. Since, we are using secondary memory storage in this method, the memory constraints would not limit the number of civilians that can be simulated. Also, since only one process is being executed, there couldn't be any constraints on the number of civilians due to the limited processing capability. However, as the secondary memory would have to be accessed for each agent for each time cycle and since there is no parallelism, each step in the simulation would take a large amount of time. Thus it would be too slow to be of any significance.

3.2 A Macro-micro Architecture

We propose an architecture, which we call the Macro-Micro architecture which uses the database to bypass the limitations of the memory, and at the same time uses the abstraction of a crowd to reduce the complexity. The multi-agent system architecture we propose in the following discussion essentially is *augmented* by a database to help it scale with respect to the number of agents that can be created and sustained in a given environment.

A crowd can be said to be a transient group of individuals, sharing some common space and environment and moving together, with all individuals having nearly the same velocity. Individuals who are deep within a crowd have a restricted freedom of movement. Their movement is decided by the movement of their neighbours. However the individuals who are at the periphery of the crowd are considerably more free to move. Thus the individuals on the periphery of the crowd shape its boundary and dictate its movement. The individuals within a crowd have an alignment towards the average direction of motion of outer individuals [21]. Further the crowd can be considered as a single entity moving in a certain direction with a certain velocity, with individuals inside the crowd sharing the same motion characteristics.

In our simulation, we maintain a database table of all civilians. Consider a moving crowd, physically separated from other crowds where the distance between two individuals in the same crowd is less than between individuals of different crowds, i.e. typically a cluster of civilians. This crowd can be considered to be a single entity and we represent it by a *Crowd* agent. Since the individuals at the boundary play an important role in determining the shape and movement of the crowd, we consider it necessary to accord a special status to them - distinct from their respective Crowd agent. We represent them as *Boundary* agents *belonging* to the that particular Crowd agent. We designate the set of Boundary agents by the cover E.

In our approach, we essentially differentiate between those civilians who are within a crowd and share the motion of the crowd, and those which are at the periphery and may have significant motion characteristics of their own. The definition of a crowd in our model is based on physical proximity of individuals in a limited space. Models explaining crowd formation or how individuals organize themselves into groups are not yet available but any such model would be consistent with our definitions. The architecture proposed has two basic components, the database and a set of agents running on the system. The database stores the state of all the civilians, who at any point of time might be activated as a boundary agent if they come to lie at the periphery of a crowd. Apart from the database we have two distinct sets of agents - Crowd agents which model the behaviour of the crowds and a set of Boundary agents who are essentially the activated civilians from the database. If at any point of time, a boundary agent is displaced inwards such that it comes to lie deep within the crowd, we deactivate the agent and it continues its existence as just a tuple in the database table. In effect, our approach is centered at activating and deactivating agents as the simulation proceeds. As the simulation proceeds, any of the following may happen:

– The crowd may disintegrate.
– A crowd might merge with another crowd.
– Individuals might leave or join a crowd.

In all these scenarios, the action takes place at the boundary as an agent can join or leave the crowd only at the periphery. In section 3.4 we present a set of incremental algorithms which trace the changes to the outer cover E for each crowd.

We are able to achieve scalability as the agents running on the system are either the civilians at the periphery or the crowd agents. Typically we were able to create and sustain up to twenty times more agents than we could by using the Micro-Agent architecture. The more important thing, however, is that we are still able to retain the granularity at the individual level since each individual's characteristics are stored in the database. We only update a small fraction of the tuples, that too only when a civilian leaves a crowd or joins a new one.

3.3 Components of the Architecture

Let us now examine each of the components of the architecture in detail.

1. **Database Table:** The table stores the details of all the civilians in the simulation. The schema of the table is –
   $\{CIV_{id}, X_{init}, Y_{init}, X_{rel}, Y_{rel}, CROWD_{id}, FLAG_b\}$.

   Each civilian is assigned an ID (CIV_{id}). It's initial position is stored as $\{X_{init}, Y_{init}\}$. $\{X_{rel}, Y_{rel}\}$ store its relative position with respect to the crowd

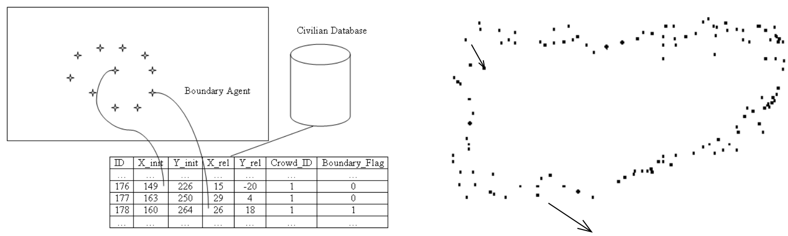


Fig. 3. (a) The system architecture. (b) An illustrative diagram showing a moving crowd. The civilian at the bottom moves away from the crowd. Another one at the top is moving in to join the crowd.

center, $CROWD_{id}$ marks the ID of the crowd to which the civilian currently belongs. The $FLAG_b$ (boundary flag) is set whenever the civilian is activated as a boundary agent and is unset when it loses that status.

2. **Boundary Agent:** Each boundary agent behaves as a single free civilian. Each boundary agent is a civilian and hence has a tuple in the database to represent it. This tuple is updated whenever the agent changes the crowd that it belongs to or loses its boundary agent status. It interacts with other agents in its visibility and moves as a result of the forces acting on it.
3. **Crowd Agent:** The crowd agent is an agent which represents all the civilians that are part of it. The movement of the crowd agent is the vector mean of the movements of the boundary agents. Each crowd agent can know all the civilians belonging to it by querying the database. It also maintains a list of all its boundary agents E_c.

3.4 Simulating the Behaviour of Boundary and Crowd Agents

1. Boundary Agent Parting away:
   In every iteration, the crowd agent checks for each boundary agent, whether it still belongs to the crowd. Those boundary agents whose distance from all the neighboring boundary agents is more than twice the visibility are considered no longer belonging to the crowd. They are considered as a new crowd with just a single agent. Such agents are removed from the boundary agent list of the crowd agent and their corresponding tuple in the database is updated. It also creates a new crowd agent which contains only one civilian corresponding to the run-away boundary agent. Thus the new crowd agent only has one boundary agent. The crowd agent also checks whether any boundary agent has been taken away by another crowd (it may happen during the merger of two crowds). It would come to know this through the *crowdID* stored in each boundary agent. The crowd agent also removes such snatched-away agents from its boundary agent list.
2. Boundary Agent joining in:
   At every time step, the crowd agent checks in the vicinity (visibility) of its boundary. If it finds a boundary agent not belonging to it and if such an agent belongs to a single boundary agent crowd or a crowd agent of smaller size (size = total civilians i.e total tuples of the crowd), then the crowd agent grabs the boundary agent as one of its own. It updates the *crowdID* of the boundary agent and also the corresponding tuple in the database. It also inserts the new boundary agent in its boundary agent list. Thus, the crowd agent has effectively snatched away a boundary agent from another relatively weaker crowd agent.
3. Change of Shape of Crowd:
   There are two ways in which the shape of the crowd agent can change. Either two boundary agents move apart and there is need for a new boundary agent between them or a boundary agent goes in and is no longer on the boundary and its boundary status has to be unset.

   At each step, the crowd agent traverses in order through all its boundary agents to check whether the distance between any two consecutive agents is more than the visibility. If so, then the crowd queries the database for the civilians whose relative positions in the crowd lie between these two boundary agents, visible to both. It picks one amongst them, sets its boundary flag in the database and creates a new boundary agent to represent this civilian. This boundary agent belongs to the crowd and is inserted into the boundary agent list accordingly. At each step, the crowd agent also checks whether any boundary agent has moved inwards into the crowd. A boundary agent is considered to have moved into the crowd, if another boundary agent lies on the line which extends radially outwards from the center towards the boundary agent. In such a case, the boundary agent that lies closer to the center is no longer on the boundary and must be dealt with accordingly. For the civilian represented by this boundary agent, the boundary flag is unset in the database and its relative position with respect to the crowd center is also updated in the database. Finally, the boundary agent is deactivated as it serves no purpose since it is no longer on the boundary and has to move as the crowd does.

4. Splitting and Merging of Crowds:
   Splitting of a Crowd can be considered as a continuous process of boundary agents moving away from the crowd agent as in (1) above. Merging of two crowds can be considered as a continuous process of boundary agents joining one crowd from another. The stronger crowd agent would snatch away the boundary agents of the weaker crowd. At the end of the merger, the weaker agent would be left without any boundary agent or any civilian belonging to it and hence would terminate itself.

4 Results and Analysis

We ran multiple simulations in scenarios involving evacuation from a hall with varying number of civilians. The results are tabulated in Table 1.

In our experiments we were able to achieve a scale up in the number of civilians simulated by a factor of five to twenty. The results tabulated are different random runs. It is difficult to provide a quantitative analysis of the results based on a standard metric as it is not possible to recreate an experiment run exactly. Depending on the structure of the crowd clusters which are formed and how individuals decide to move in each step, the number of instructions executed to maintain the boundary could vary widely over experiments.

An important aspect is that the complexity of the system depends upon the number of boundary agents rather than the number of civilians. The number of crowd agents is typically quite less compared to the number of boundary agents. It is difficult to keep a tab on the number of boundary agents. Their number could vary with time in a particular scenario, and more generally could vary across different scenarios. In our experiments we create some well defined crowd clusters at the beginning of the simulation and mark the civilians at the periphery as the boundary agents. Thereafter the boundary is updated as per the algorithms specified above. To illustrate the effect of the number of boundary

Table 1. 1a: Time taken to run simulation for 300 cycles. 1b: Time taken to run simulation for 300 cycles for 30000 civilians.

Number of civilians	Time taken (for 300 cycles)	Initial number of boundary agents	Final number of boundary agents
1000	0m03.560s	90	85
3000	0m10.970s	300	267
9000	0m34.483s	900	880
15000	0m45.283s	1500	1372
18000	1m17.640s	1800	1631
30000	2m8.647s	3000	2754
50000	12m28.001s	4500	4282
75000	18m17.522s	5400	5177
100000	23m51.707s	6000	5652

Number of civilians	Time taken (for 300 cycles)	Initial number of boundary agents	Final number of boundary agents
30000	0m18.136s	12	128
30000	0m16.688s	30	143
30000	0m19.179s	90	199
30000	0m20.393s	300	24
30000	0m29.537s	600	530
30000	1m59.602s	900	880
30000	2m8.647s	3000	2769
30000	26m40.784s	6000	5904
30000	29m57.944s	7500	7235
30000	49m27.345s	9000	8877

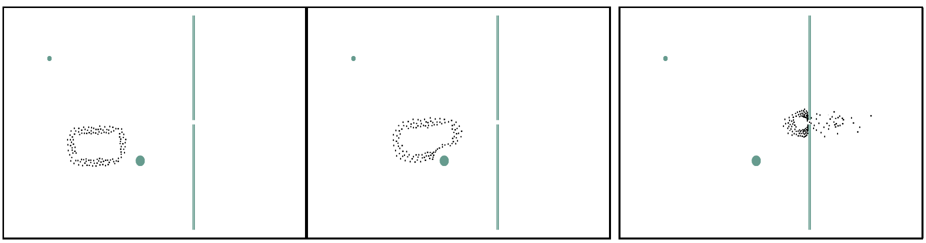


Fig. 4. Evacuation from a hall:: Fig (a): A rectangular crowd → Fig (b): Crowd expands, avoids obstacle → Fig (c): At the exit: Crowd broken into a number of small crowds(including unit crowds)

agents on the simulation time we tried varying the number of initial boundary agents while keeping the number of civilians constant. The results are shown in Table 1(b). The worst case scenario for the macro-micro architecture would be when the civilians are distributed and far apart to form any large clusters. In that event, the number of boundary agents would roughly equal the number of civilians. On the other hand, the best case scenarios would be when the civilians are distributed into thick and distinguishable clusters.

4.1 Keeping Track of Individual Civilians

Each civilian exists as a tuple in the database. Thus, its identity is preserved. Its relative position with respect to the crowd is stored and so is its crowd ID stored in the database. Thus at any point of time, its actual position on the map can be approximately known since the coordinates of the crowd center are available and the relative position with respect to the crowd center is also available through the database. The basic assumption here is that the relative position of a civilian with respect to the center of the crowd will remain unchanged. This assumption is justifiable because a civilian inside a crowd is constrained by its neighbours. When a civilian becomes a boundary agent, then too its movement can be tracked since it exists as an agent on the map. When it joins another crowd, then the crowd ID, relative positions etc are updated accordingly and therefore its position is still known. Thus at any point of time, the location of any civilian on the map can be obtained. Thus all civilians can be tracked.

5 Conclusion and Future Work

System issues limit the scope of applying multi-agent systems for massive crowd simulation. In this paper we proposed a macro-micro simulation system based on a database to richly simulate massive crowds. We have presented our results demonstrating how our technique can be used to overcome the system constraints. More significantly we believe that this idea can be extended to a generic architecture where it is possible to richly simulate the behaviour of a large body of agents by focusing on a relatively smaller number of significantly active agents. Further research can focus on how database systems can be effectively used for augmenting multi-agent system architectures to attain scalability. It is possible to consider boundary agents as the leaders in the crowd. We plan to simulate emergency evacuation scenarios analyzing the role of these leaders in such cases.

References

1. Helbing, D., Farkas, I.J., Molnr, P., Vicsek, T.: Simulation of pedestrian crowds in normal and evacuation situations. In: Schreckenberg, M., Sharma, S.D. (eds.) Pedestrian and evacuation dynamics, pp. 21–58. Springer, New York (2002)
2. Helbing, D., Farkas, I., Vicsek, T.: Simulating dynamical features of escape panic. Nature v. 407, 487–490 (2000)
3. Lakoba, T.I., Kaup, D.J., Finkelstein, N.M.: Modifications of the Helbing-Molnar-Farkas-Vicsek Social Force Model for Pedestrian Evolution In Simulation, vol. 81, pp. 339–352 (2005)
4. Henein, C.M., White, T.: Agent-Based Modelling of Forces in Crowds. In: Proceedings of the Workshop on Multiagent and Multiagent-based Simulation (MAMABS) at AAMAS 2004, July 19-23 (2004)
5. Schweiss, E., Musse, S.R., Garat, F., Thalmann, D.: An Architecture to Guide Crowds Using a Rule-Based Behavior System. In: Proceedings of Autonomous Agents (1999)
6. Farenc, N., Musse, S.R., Schweiss, E., Kallmann, M., Aune, O., Boulic, R., Thalmann, D.: A Paradigm for Controlling Virtual Humans in Urban Environment Simulations. Applied Artificial Intelligence 14(1) January 1 (2000)
7. Kirchner, A., Schadschneider, A.: Simulation of evacuation processes using a bionics-inspired cellular automaton model for pedestrian dynamics. Physica A 312, 260–276 (2002)

8. Dijkstra, J., Jessurun, A.J., Timmermans, H.J.P.: A Multi-Agent Cellular Automata System for Visualising Simulated Pedestrian Activity. In: Proceedings of the Fourth International Conference on Cellular Automata for Research and Industry, pp. 29–36. Springer, Berlin (2000)
9. Hamagami, T., Hirata, H.: Method of crowd simulation by using multiagent on cellular automata. In: Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology (IAT 2003) (2003)
10. Pan, X., Han, C.S., Law, K.: A multi-agent based simulation framework for the study of human and social behavior in egress analysis. In: The Proceedings of The International Conference on Computing in Civil Engineering, Cancun, Mexico, July 12-15 (2005)
11. Murakami, Y., Minami, K., Kawasoe, T., Ishida, T.: Multi-Agent Simulation for Crisis Management. In: Proceedings of the IEEE Workshop on Knowledge Media Networking (KMN 2002) (2002)
12. Ishida, T.: Social Agents in Digital Cities
13. Ohta, M., Koto, T., Takeuchi, I., Takahashi, T., Kitano, H.: Design and implementation of the kernel and agents for the RoboCup-Rescue. In: Proceedings of IEEE, Fourth International Conference on MultiAgent Systems (2000)
14. Koto, T., Takeuchi, I.: The Distributed Simulation of RoboCupRescue Version 1. In: Proceedings of SICE 2002 (2002)
15. Koto, T., Takeuchi, I.: A Distributed Disaster Simulation System That Integrates Sub-simulators
16. Tadokoto, S., et al.: The RoboCup-Rescue Project: A Robotic Approach to the Disaster Mitigation Problem. In: Proc. IEEE International Conference on Robotics and Automation, pp. 87–96 (2000)
17. Brenner, M., Wijermans, N., Nussle, T., de Boer, B.: Simulating and Controlling Civilian Crowds in Robocup Rescue. In: RoboCupRescue Simulation: Infrastructure competition (2005)
18. Toyama, M., Bazzan, A., da Silva, R.: An Agent-Based Simulation of Pedestrian Dynamics: from Lane Formation to Auditorium Evacuation. In: Proceedings of International Joint Conference on Autonomous Agents and Multi-Agent Systems (2006)
19. Osaragi, T.: Modelling of pedestrian behaviour and its applications to spatial evaluations. In: Jennings, N., Sierra, C., Sonenberg, L., Tambe, M. (eds.) International Joint Conference on Autonomous Agents and Multi-Agent Systems, pp. 836–843 (2004)
20. Yamamoto, G., Nakamura, Y.: Architecture and Performance Evaluation of a Massive Multi-Agent System. In: Proceedings Autonomous Agents 1999 (1999)
21. Reynolds, C.W.: Flocks, Herds, and Schools: A Distributed Behavioral Model. In: Computer Graphics (SIGGRAPH 1987 Conference Proceedings), vol. 21(4), pp. 25–34 (1987)
22. MASON: Java based multiagent simulation library. MASON has been developed by George Mason University's ECLab Evolutionary Computation Laboratory and the GMU Center for Social Complexity,
http://cs.gmu.edu/~eclab/projects/mason/

Middleware Support for Performance Improvement of MABS Applications in the Grid Environment

Dawit Mengistu, Paul Davidsson, and Lars Lundberg

Department of Systems and Software Engineering, School of Engineering,
Blekinge Institute of Technology,
372 25 Ronneby, Sweden
{dawit.mengistu,paul.davidsson,lars.lundberg}@bth.se

Abstract. The computational Grid is an infrastructure which enables the execution of applications demanding huge computing resources. Hence, it can be the right environment for large-scale Multi-agent based simulation (MABS) applications. However, due to the nature of the Grid and the characteristics of MABS, achieving optimum performance poses a great challenge. Performance study of MABS applications is therefore a necessary undertaking which requires an understanding of these characteristics and the extent of their influence. Moreover, owing to the dynamicity and heterogeneity of the Grid, it is difficult to achieve performance gains without a middleware support for application deployment and dynamic reconfiguration. This paper presents a study of the key features of MABS applications that affect performance and proposes a supportive middleware to MABS platforms. Experiments show that the proposed middleware can bring performance improvement for MABS applications on the Grid.

Keywords: MABS, Performance Improvement, Grid Computing, Middleware.

1 Introduction

The computational Grid is a hardware and software infrastructure that provides access to a possibly huge pool of heterogeneous computing resources. It is a resource rich distributed computing platform, that can support the execution of complex applications in various fields. Applications in climatic simulations, earth science, medical research, etc., which would take a lot of time or be unmanageable in other environments can now be executed on the Grid within a reasonable time. One major area where the benefit of Grid computing is evident is in the field of simulation, particularly in multi-agent based simulation (MABS).

MABS applications involve modeling real world entities as software agents that interact with each other. In most of the cases, MABS applications are limited, partial, conducted on a smaller scale, with fewer agents, making use of many simplifying assumptions, etc [2]. MABS models often are designed as down-sized versions of the actual problem, or are implemented as separately run components executed at different time and/or premises, due to lack of computing and visualization resources.

L. Antunes, M. Paolucci, and E. Norling (Eds.): MABS 2007, LNAI 5003, pp. 20–35, 2008.

Recent developments in computing have created an opportunity to make advances in large scale MABS. High performance computing resources such as the Grid have become available for solving problems that were once thought to be intractable. Agent programming tools and multi-agent platforms are developed to deploy multi-agent applications on distributed resources with less programming effort. Although multi-agent platforms designed for distributed computing are maturing, the work done so far to use them in the Grid environment is very limited.

One challenge when implementing MABS applications on the Grid concerns the application performance that is caused by the relatively high communication-to-computation ratio of multi-agent systems. There are also other inherent features that affect the performance of MABS when it is deployed as a distributed application in heterogeneous environment such as the Grid.

The purpose of this work is to study the performance characteristics of MABS applications on the computational Grid and propose a middleware to be used for performance improvement. Using a synthetic MABS application as a workload, we conducted an experiment to study the behaviour of MABS, investigate application design issues, and ways of overcoming performance bottlenecks. We also developed a prototype of the middleware which was used in the experiment. The rest of this paper is organized as follows: the motivations to study performance issues for MABS deployment on the Grid is presented next, followed by a discussion on performance issues of MABS applications and the Grid. We then present an architectural framework of the proposed middleware and an overview of the envisaged MABS deployment in the Grid. In Section 5, the experimental environment and the characteristics of the workload used are explained. Section 6 presents the results of the experiment with a qualitative discussion and the performance improvement achieved using a prototype of the middleware. We conclude the paper by summarizing the findings of our study and citing directions for future work.

2 Motivations for This Study

Although there are many circumstances where partial simulations suffice to understand the real world phenomena, there exist a number of problems that cannot be adequately understood with limited-scale simulations [1]. Partial simulations may not always yield complete results, require additional time and effort to assemble, analyze and interpret. The domain being studied can be better understood if a large, or even full scale simulation where each real world entity is modeled as an agent and the relations between the entities is represented accordingly. In problems where the number and type of entities is large, it will be very difficult to represent the dynamics in its entirety due to lack of adequate computing resources. A very large scale simulation would require the availability of computational power large enough to run the simulation within an acceptable span of time.

Technically, a MABS application may be implemented as a multi agent system comprising several agents working in parallel performing computing, communication and data access tasks asynchronously. Therefore, MABS models can be conveniently ported to the Grid as parallel applications. In particular, the Grid can be an ideal computational environment for MABS because:

1. Simulations involving thousands or millions of agents and highly complex and data intensive tasks can benefit from the Grid's high throughput resources.
2. The distributed nature of agent-based computing makes distributed platforms like the Grid a natural working environment for multi-agent systems.

However, MABS applications designed without properly considering their execution characteristics (particularly the communication behaviour) in a distributed computing environment are unlikely to be good candidates for the Grid. It is therefore necessary to identify the key features of MABS applications that affect performance on the Grid. By understanding the impact of these features, isolating and eliminating the performance bottlenecks they cause, simulations can be executed efficiently. A MABS platform supported by a middleware that monitors the execution environment and adapt the application for best performance on the Grid is desired. A performance-aware middleware that works with the MABS platform to support dynamic application tuning and optimization for the Grid is required to achieve this.

3 Performance of MABS Applications on the Grid

A MABS platform provides the physical structure in which agents are deployed. It generally consists of agent support software, agent management system, a directory service, a message transport service and the agents themselves [15]. The architecture of the application running on the platform affects performance significantly.

3.1 Key Features of MABS Affecting Performance

Although it is not possible to list all items that potentially contribute to performance problems, we have identified relevant key features of MABS to be examined in this experiment. The following architectural features have significant influence on the performance of MABS applications.

1. *Multi-threading*. The autonomous behaviour of agents is conveniently realized by writing the simulation application in such a way that each agent runs within its own thread of control. Platforms that do not provide multi-threading may hamper modeling MABS applications in distributed environments. If the simulation size is large, i.e., involves too many agents, it would require substantial multi-threading. This undermines system performance severely since running too many threads causes a lot of context switching, more overhead and complicates scheduling tasks.
2. *High communication-to-computation ratio*. Agents simulating real world entities should mimic the interactions among the entities they model. The interaction is inter-agent messaging, where the message originating and destination agents are deployed on the same or different nodes. Messaging between agents running on the same node is essentially data movement within the same physical memory, while that between agents on different nodes involves local or wide area communication. A key requirement of the simulated problem may be that the agent originating a message not perform any action (hence should be in a waiting state without executing any useful operation) until it receives the reply to the message it sent out.

If intense interaction between agents running on separate nodes is involved, this forced waiting undermines performance.

3. *Time-step synchronization*. For an agent based simulation, to reflect the chronological sequence of events in the simulated problem and guarantee causality of actions, the timing of the agent executions throughout the simulation is almost all cases of great importance. This is normally carried out by forcing a currently running thread to yield involuntarily, to avoid running out of synchronism with the other agents. The yielding thread will stay in the wait state until all agents are brought to the same temporal situation. If the size of the task executed within a unit of time is very small (i.e., the application is fine grained), this would entail a lot of context switching overhead which may even exceed the actual useful computation. This problem may further be exacerbated depending on the scheduling policy of the run-time environment.

The above factors may occur in combination, making performance analysis of MABS applications very complex. Another inherent feature of agent based applications affecting performance is that unlike object oriented applications, execution time cannot be predicted from the size of the executable code alone. Deploying such applications on a dynamic execution environment like the Grid will make the problem even more complex. The MABS application would therefore require proper architectural considerations.

3.2 Performance Analysis of Grid Applications

Performance analysis for efficient application execution and scheduling is a major challenge to Grid application developers [13]. Although recent Grid monitoring tools and schedulers come with improved features, they do not adapt to application specific architectural details in many cases due to their generic features. The performance of domain specific Grid applications can be enhanced by using application specific middleware which assist Grid schedulers. A *performance-aware-middleware* supports application modeling, prediction, optimization based on information acquired from Grid monitoring services. However, due to the dynamic behaviour of the Grid and the heterogeneity of resources, the implementation of performance analysis techniques in the context of Grid computing is more complex than in other environments. Because of this, static performance support tools do not provide effective support. As a result, the use of a middleware for dynamic performance analysis and prediction is becoming more common. Our plan to use a middleware support for MABS applications is therefore based on the inherent nature of the Grid environment.

4 Proposed Architecture

The proposed Grid middleware is to be used as an interface between a MABS platform and the Grid resources and services. As explained earlier, the major purpose of employing this layer is to optimize a MABS application run-time code to execute efficiently in a given Grid environment. This middleware can interact with Grid monitoring services to improve performance using the information it receives from the services. In a Grid environment where monitoring tools are not available or accessible, the middleware can perform several useful tasks such as:

- Partitioning the simulation application into balanced tasks that can be dispatched to available Grid nodes. The simulation size is determined by the number of entities simulated or the number of agents to be deployed. Balancing tasks corresponds to dividing the number of agents equally into the number of Grid nodes taking part in the simulation.
- Distributing the task to the nodes. The task to be run on a node is deployed as a web service launched through the middleware.
- Monitoring the performance of the application. The middleware collects application information such as messaging statistics, thread waiting times, etc., from the Grid nodes through web service invocations.
- Reallocates agents and migrating tasks to improve performance. Based on the application performance data collected from Grid nodes, the middleware will decide on an optimal job redistribution strategy and reallocate the tasks to the nodes as necessary.

When implemented in conjunction with Grid infrastructure monitoring tools, the Middleware will interface the platform functionalities with the Grid resources. It uses performance data collected by Grid monitoring services for performance modeling and prediction purposes. It will then carry out essential tasks such as load balancing and reallocation strategies for job optimization and match making. The Grid scheduler will then be able to receive an optimized job matching predicted performance and execution requirements such as deadlines, if any. Figure 1 shows the architecture of the proposed performance aware middleware.

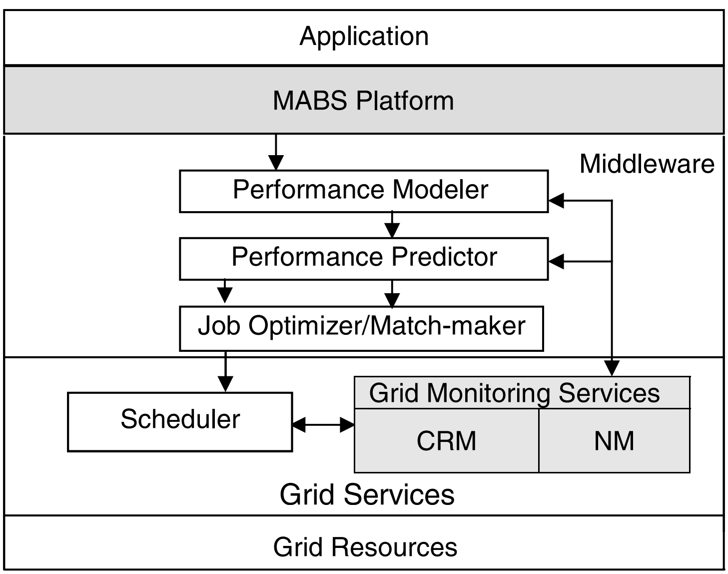


Fig. 1. The proposed middleware architecture

The Simulation Modeler assists users by taking away most of the system level programming tasks from them so that they can focus only on the functional logic of the simulated entities. According to [2], the simulation modeler is implemented as a customizable framework for modeling social phenomena in different domains. It should thus consist of a domain specific model editor and a user interface that enables capturing of the entities and the problem to be simulated itself.

The middleware consists of the following components:

1. The *Performance Modeler* uses performance data collected through Grid monitoring services, i.e., Compute Resource Monitoring (CRM) and Network Monitoring (NM), to build the execution performance model of the simulation workload. The model contains information needed to make scheduling decision, reallocation strategy, etc.
2. The *Performance Predictor* predicts the execution time of the simulation based on the output of the performance modeler and data from Grid monitoring services on current status of the Grid infrastructure. The prediction is bound to change with changes in the Grid environment. The predictor can also decide the optimal application deployment/ reallocation strategy.
3. The *Job Optimizer/ Match Maker* undertakes the necessary modifications on run-time parameters of the tasks for optimal use of the Grid resources on which each task is to be deployed. This component will thus assist the generic Grid scheduler to make more refined scheduling decisions.

The MABS platform consists of the components necessary to run a meaningful simulation application in a distributed environment. Figure 2 shows the components in detail.

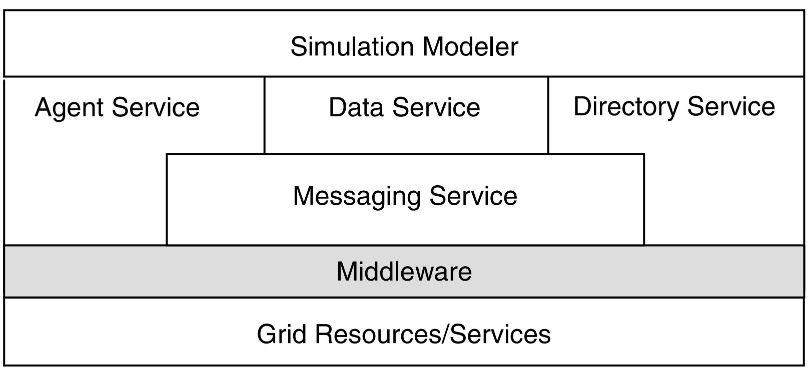


Fig. 2. Major components of a MABS application platform in the Grid environment

Since the Grid employs service oriented technologies, it is convenient to implement these platform components as distributed services to facilitate interoperability with other services.

The Directory Service manages information about the agents such as naming, locating, etc. Upon launching, agents automatically register in the directory service with a unique identifier. This service has a hierarchical architecture distributed over the Grid simulation nodes, with each node managing the information of the agents running on it. A central directory service is responsible for coordinating the directories of individual nodes.

The Data Service is an agent that keeps the repository of the data produced in the simulation. The data is converted into a knowledgebase which the agents regularly use to update their beliefs about the environment.

The Agent Service handles functionalities such as the launching and termination of agents, execution of agent code, registration and deregistration of agents, etc.

The Messaging Service is responsible for routing and delivery of messages between agents. In general, messages could be destined to peers residing on the same node (inbound) or elsewhere on the Grid (outbound). This service interacts with the Directory Service to decide whether messages are inbound or outbound.

5 Experiment

An experiment is designed to study the effect of the key MABS features discussed in Section 3.1 and to implement a middleware support for performance improvement. The MABS application used in the experiment is a synthetic simulation workload which incorporates these key features. The implementation details of the workload and the experimental environment are discussed next.

5.1 Workload Characterization

The workload is a multi-threaded MABS application. Each agent in the simulation is characterized by the following features:

- Has its own thread of execution and is recognized as a distinct process or thread by the underlying operating system;
- Communicates with other agents through inter-thread communication APIs implemented at the application level.
- A program code that captures the behaviour of the real world entities it is intended to represent.

For the purpose of this experiment, the agent code will have two components, computational and communication, clearly separated to manipulate the control variables and to study the effect of the earlier discussed features of MABS. The computational part mainly represents the role (task) of the agent, while the communication part handles the inter-agent communication activities. The actions of an agent are therefore realized by a thread code that performs a computation, followed by a messaging and yields to enforce the time-step synchronization.

If the workload is implemented on a stand-alone machine, since all threads run on that same machine, the inter-thread communication is essentially inbound. On a Grid version, however, the threads are launched on separate machines and the communication can be outbound too. The Grid workload should be partitioned into tasks of equal size, to be launched on the nodes.

The communication characteristics are of primary interest and should be well defined in the workload model. We defined the intensity of outbound communication as a parameter. We also study the effect of time-step synchronization in two situations. First, we see the extent to which the presence of time-step synchronization affects performance by taking two simulation runs with and without synchronization. We will then investigate the effect of performance for different levels of application granularity. In this experiment, the application granularity is considered to be the time it takes to execute one computational loop at the end of which synchronization between agents is enforced. Fine-grained applications perform smaller computational task in each time step while coarse-grained applications have larger size of code to be executed between successive time steps.

It is desired that performance instrumentation causes as little perturbation as possible to the instrumented application. Therefore, data manipulation and transfer tasks were deferred to the end of the workload execution not to interfere with the normal course of the application. In the case of the Grid, it is believed that putting measurement data on top of every messaging data will increase the overhead and should be avoided or minimized. Therefore, the measured data are stored in elementary data structures until the end of the simulation and communicated to the master node only thereafter.

The workload is modeled in terms of known and predictable computational and communication tasks to be carried out by each thread. The effect of known external factors that would bias the outcome should be minimized. For this reason, the application does not contain such tasks as I/O operations other than the communication explicitly required for inter-agent messaging.

An important requirement of the measurement process in multi-threaded applications is capturing the desired performance metrics with minimal overhead. To achieve this, individual threads maintain their respective copies of performance data, but share a single instrumentation code [8].

The validity of the experimental setup was verified, and show that the workload and performance metrics chosen conform to the envisaged objective and are equally valid for both the stand-alone and Grid environments. The workload was run in both environments several times to see if the experiment is repeatable and the outcomes are predictable. It was observed that factors that had not been accounted for have very little effect on the experiment and the workload model is indeed valid for the study.

5.2 Grid Environment

The Grid version of the experiment was conducted on a Globus (GT4) Grid testbed with machines having PIII 1000MHZ processors with 512MB RAM running Linux. These machines are connected via a 100Mbps switch to the Internet. Since the main purpose of the study is the effect of MABS characteristics on performance, it was necessary to run the experiment under a controlled environment to isolate and observe the effect of each of MABS features explained earlier. For this reason, the Grid nodes do not have additional loads. Furthermore, some components of the Globus toolkit such as GFTP and security features, although installed, are not used here since their presence or absence bears no relevance to this experiment.

Each node hosts an identical copy of the MABS application as a set of Web Services deployed in its Globus container. Another machine used as a master node serves as a launching pad where the simulation application is divided into sub tasks to be invoked as web services on each Grid node.

The stand-alone version of the application was run on a Linux machine having identical configuration with the worker nodes used in the Grid version of this experiment. Since the applications run in both environments are functionally the same, we only discuss the Grid version in the rest of this paper.

5.3 Implementation of Workload

The workload is a Java multi-threaded application written in accordance with the requirements of the workload model explained in 5.1 above. The Grid version of the application is a web service launched from a master (client) node where the workload is partitioned into pieces of balanced sizes distributed to worker nodes on the Grid. For example, if the real world problem consists of 100 similar entities modeled by 100 agents and the simulation is run on 2 Grid nodes, the master node launches the web services on each worker node such that the worker hosts 50 agents. Inter-agent communications can occur between those agents residing on the same or different nodes.

The agents are instantiated as individual threads in the Grid web service. As explained earlier, the workload has a fixed size and is executed as computational loops interleaved with messaging operations. Since granularity is a parameter, simulation runs are conducted with different granularity levels. The execution time of one loop is thus a measure of the application's granularity. Inter-thread messaging normally takes place following this. The product of the granularity and the number of loops is constant and represents the size of the workload.

If an agent sends out a message and does not receive a reply immediately or within a reasonable time, it should yield control and wait until the reply comes instead of advancing the computational loop. Since all agents take control of the CPU turn by turn, the expected reply will be received sooner or later.

Since the threads need to maintain coherence with respect to the simulated time, they should be made to advance execution of loops at a synchronized pace. If a thread finishes a task ahead of the others, it should stay in a waiting state by yielding control of the CPU until other threads come to the same level.

A shared memory area is reserved for managing the inter-agent messaging. Each agent would be able to read messages destined to it and also send out messages to others. If an inbound message is sent, (to an agent on the same node) it will be stored in the message buffer of the destination agent. If, however, it is destined to an agent on another node, it is kept in a different area from which the messaging web service collects outbound messages and delivers them to the client. The web service on the destination will then collect the message from the client through its messaging service and places it in the destination agent's messaging buffer.

The ratio of outbound messages to the total number of dispatched messages is an important parameter for studying the effect of network latency. We have therefore defined it as outbound (OB) communication ratio. The ratio is given as the number of messages sent out by an agent to agents residing on nodes other that hosting the sender, out of every 100 messages initiated by that agent, expressed in percentage.

Time-step synchronization is enforced at two levels, local and global as follows: first, all agents running on the same node are internally synchronized. When the node level (local) synchronization is completed, the master node is notified through a response to a Web Service request initiated by the master node itself. After all nodes reported their status to the master (global) in this way, they will receive another Web Service invocation as a command to proceed with the next time step or computational loop, for a specified number of times.

To launch the application, the master code is initiated on the client machine and invokes the Web service on each node by passing the required parameters of the simulation. This code also monitors the overall execution of the application and facilitates the delivery of outbound messages in cross node communications.

Using different values of the experiment parameters, i.e., simulation size (number of agents), granularity and outbound communication ratio, several runs of the experiment were conducted. The parameters were varied independently and together, to understand their individual and combined effects. The observations made from the experiment, the analysis of the measurement data and the performance improvement obtained using the middleware are discussed in the following section.

6 Results

The observations made from the experiment are presented below in this section according to the order in which they appear in Section 3.1. The plots from the repeated runs of the experiment summarize these observations. A prototype of the middleware proposed in Section 4 is implemented to see how it improves performance by addressing one of the factors, namely, the outbound communication. The performance gain with the middleware is presented at the end of this Section.

6.1 Effect of Multi-threading on Performance

If the program codes of individual agents are realized as separate threads, the application will be massively multi-threaded when the size of the simulation is large. Because such level of multi-threading causes a lot of overhead on the computational resource, execution time increases in a quadratic fashion and performance reduces significantly. To illustrate this, several runs of the experiment were conducted with different number of agents deployed as individual threads, taking part in the simulation. The outcome of the experiment is shown in figure 3.

As can be seen from the plot, implementing the agents as individual threads degrades performance significantly. It follows that, if the simulation is distributed over *M* nodes, given other conditions equal, the speed up gain in the best case can be as high as M^2. Several reasons are attributed to this, such as poor performance of the JVM, operating system policies, presence of synchronized methods in the agent codes to ensure data integrity, etc.

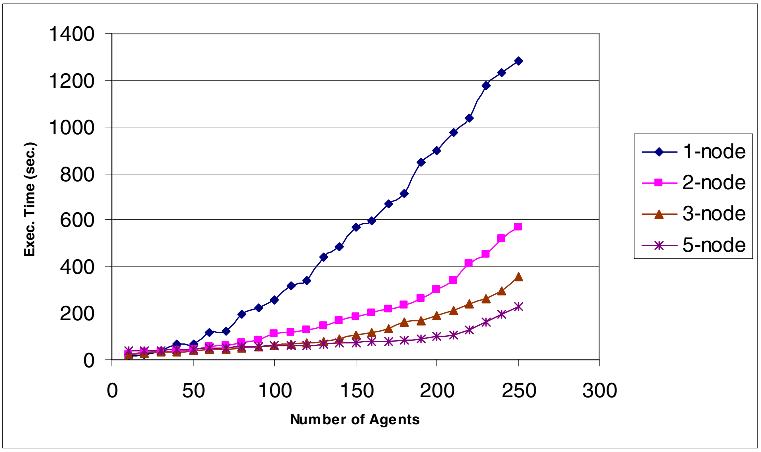


Fig. 3. Execution time of MABS applications on the Grid increases non-linearly with simulation size

6.2 Effect of Outbound Communication

The effect of outbound communication is typically discernible when the requirement of the simulation is that agents expecting replies for messages they sent out cannot proceed with their tasks until the reply arrives. In such cases, the agents will be forced to stay in a *wait* state until the *reply* arrives. The contribution of this waiting time could even be well larger than that of the time needed to execute the agent code depending on the intensity of the communication and the granularity of the MABS application. Coarse-grained applications have generally low communication-to-computation ratio and will not be affected as seriously as fine-grained ones. As the simulation size increases, however, other factors such as synchronization, multi-threading, etc., also come into the picture. The effect of outbound communication is felt most if the size of inter-agent messages is too large to fit into the communication buffers.

Instead of sending out messages instantly as they are produced, storing them in one location until the messaging web service collects them from that node for routing reduces network traffic significantly. This technique, known as *message aggregation,* is a very useful strategy employed in other communication intensive distributed applications also.

6.3 Effect of Time Step Synchronization

As explained earlier, the simulation cannot mimic the real world behaviour unless the operations of all simulating agents are synchronized in time steps. Time step synchronization undermines performance by introducing additional overhead as can be seen in the following figures.

The plots show that fine-grained simulations are affected most by time-step synchronization. Furthermore, the performance loss is more significant in small-scale simulations. For larger simulations however, the significance of this loss will be less since other factors related to the operating environment take a larger share of the performance drop. In limited cases, MABS applications may not need synchronization at all. Such cases arise when the simulated system does not involve any communication between entities or the chronological sequence of simulated events is of no interest. In such cases, it is possible to achieve significant performance gains by avoiding time-step synchronization.

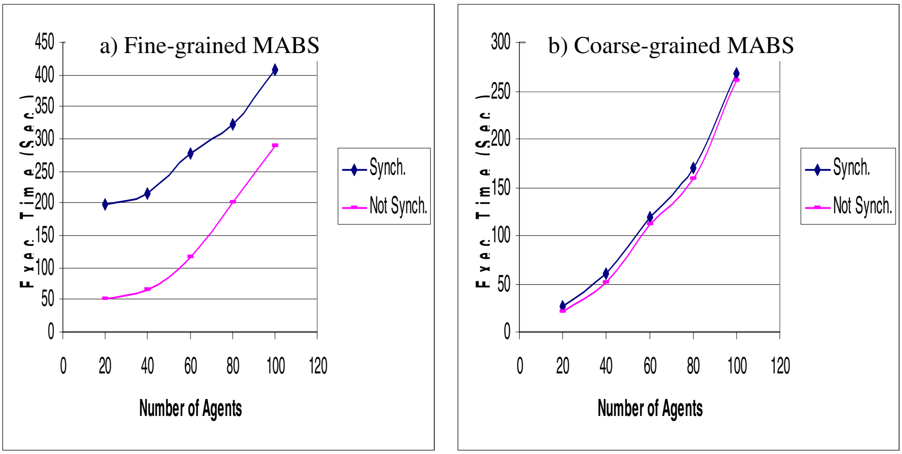


Fig. 4. Comparison of simulations with and without time-step synchronization on a 5-node Grid, with no outbound communications. Fine grained simulations are affected most.

6.4 Middleware Support for Performance Improvement

The middleware was used to improve load distribution with the view to reducing outbound communication. It studies the communication pattern of the agents, with whom they communicate most. It will then regroup the agents such that as many peer agents (those that are likely to have frequent communication with each other) as possible are run on the same node, effectively reducing most of the communication traffic to an inbound one. The effectiveness of the middleware was examined with an experiment where agents are initially located arbitrarily without any knowledge of their communication patterns and location of their peers. During the execution, the middleware analyzes the messaging behaviour of the agents, with whom they communicate very often. It then uses the analysis to perform task reallocation by moving as many peer agents as possible to a common node so that most of the inter-agent communication can be internal.

In this experiment, it is assumed that the list of peers an agent communicates with does not change with time. However, it is not necessary that any two peers have an identical set of peers. Because of this, it is not possible to eliminate outbound communication completely.

If the simulation involves N agents running on M machines, the probability that an agent finds its peers on its own node is 1/M. The number of agents launched per node will be *N/M* and the rate of outbound communication will then be:

$$OB = (1-1/M)*100\% \tag{1}$$

The middleware can reduce the *OB* above significantly depending on the number of peers an agent has. The table below shows the reduction in the number of outbound messages as a result of the middleware's intervention. The experiment was run with 500 agents and different inter-agent communication intensities (different number of peers per agent) on a 5-node Grid.

Those runs with larger reduction percentage correspond to cases where agents have relatively fewer peers, so that after reallocation, they find most of their peers on the same node and the communication becomes essentially inbound. If, however, agents interact with too many peers, the middleware may only be able to move a limited number of these peers to the same node and a good portion of the communication remains outbound.

Table 1. Comparison of outbound traffic with and without middleware support

Run No.	Number of OB Messages		Reduction
	Without middleware	With middleware	
1	1564	458	70.7%
2	5716	3840	32.8%
3	15825	4721	70.2%
4	28691	13393	53.3%
5	32552	8496	73.9%

7 Conclusions and Future Work

The experiment has shown the architectural issues to be considered in designing MABS applications for the Grid and also how a middleware can be used to improve application performance. The findings of this work are useful for platform development, resource planning and simulation modeling. A Grid based platform making use of a performance-aware middleware can support fine tuning of its overlying application if the simulation size, the communication characteristics and the granularity of the application can be determined. Simulation modelers can decide on the architecture of the simulation, complexity of the agent codes, etc., anticipating the performance benefits that can be achieved by designing an efficient simulation.

The messages used in the experiment have small sizes (few bytes). The achieved improvement in performance, i.e., reduction in execution time, however, also depends on the size of the messages. It is therefore necessary to perform additional experiments to determine the performance gain for different sizes of messages, to determine whether the gain in outbound message reduction is translated into a saving in execution time.

We extended the experiment with the previous workload, to build performance models. The model predicts the simulation execution time if it is given the simulation size (number of agents), task granularity, the rate of outbound communication and the number of Grid nodes on which the simulation is executed. An important benefit of the prediction model is that it can facilitate the tasks of Grid resource allocation and scheduling.

The accuracy of the model largely depends on the granularity of the simulation. While execution times for coarse grained simulations (like in figure 4b) are fairly predicted, fine-grained simulations were not accurately estimated. We believe this problem can be partly solved if the model is made comprehensive enough using additional experimental data with fine-grained simulations. However, if the granularity is extremely fine, it will be difficult to perform stable experiments and take accurate measurements due to instrumentation and related overheads.

Although it was attempted to make the compute resources heterogeneous by exposing them to different load conditions, the initial study would have been more comprehensive in an environment with heterogeneity in hardware and software. To address this problem, we are currently expanding our initial experiment setup. Future experiments will be performed on an already established Globus 4 testbed with around 40 machines over 3 sites in Germany and Sweden.

A synthetic workload with generic features was used to represent a basic MABS application so far in a homogeneous Grid environment. This workload differs from an actual MABS because its model contains several simplifying assumptions. In our current work, we are studying how the middleware can be used in realistic scenarios in a dynamic execution environment consisting of heterogeneous resources. For this purpose, a practical simulation application showing the prominent features of MABS used in social simulation modeling should be empoyed. We thus replaced the workload with an existing MABS application from practice in the transport and logistics domain.

The simulation application focuses on effects of control policies on freight transport chains. It is used as a decision support system for the various stakeholders of a transport chain (customers, buyers, suppliers, production, transport coordinators, etc.). It captures important information such as production capacity, storage, vehicle loading and unloading time, vehicle capacity, speed, environmental performance, and others. The simulation model also incorporates the interaction between all the entities, which are rightly modeled as agents.

A version of the simulator was implemented using the JADE platform on a standalone machine. JADE can be used in a way that fits the middleware architecture given in figures 1 and 2. It simplifies the implementation of MABS applications through a middleware that complies with FIPA agent specifications and services such as white-page services, yellow-page services, message transport and parsing services. The JADE platform can also be used across Java-enabled heterogeneous machines on the Grid. It is based on an execution container, which provides a complete run-time and communication environment for agent execution.

Initially, JADE was not developed with simulation applications in mind [16]. The messaging service of JADE also indicates that it was not designed for high performance computing systems. In order to carry out a large-scale simulation on the Grid using JADE, it would be necessary to identify the features of JADE that can be sources of performance bottlenecks.

One cause of performance degradation while implementing the workload as a distributed simulation with JADE is that of inter-agent messaging. We are therefore working on extending JADE to be suitable for the Grid environment, to efficiently execute communication-intensive simulations. One extension is the decentralization of the messaging service so that inbound messages can be delivered without congesting the network and burdening the services on the main JADE platform. Another enhancement is the adaptation of our middleware for task reallocation and agent migration as explained earlier in section 6.4. We employ decentralized directory facilitators and yellow page services to provide the necessary information for the middleware to make appropriate decisions on task redistribution. The middleware studies the messaging and deployment patterns of peer agents by interacting with the messaging and directory services as the simulation progresses.

References

1. Cioffi-Revilla, C.: Invariance and universality in social agent-based simulations. In: Proc. National Academy of Science USA, vol. 99 (suppl. 3), pp. 7314–7316 (2002)
2. Davidsson, P., Holmgren, J., Kyhlbäck, H., Mengistu, D., Persson, M.: Applications of multi-agent based simulations. In: Antunes, L., Takadama, K. (eds.) MABS 2006. LNCS (LNAI), vol. 4442. Springer, Heidelberg (2007)
3. Sansores, C., Pavon, J.: A framework for agent based social simulation. In: The Second European Workshop on Multi-Agent Systems, Barcelona, Spain (2004)
4. Gasser, L., Kakugawa1, K., Chee1, B., Esteva1, M.: Smooth scaling ahead: Progressive MAS simulation from single PCs to Grids. In: Davidsson, P., Logan, B., Takadama, K. (eds.) MABS 2004. LNCS (LNAI), vol. 3415, pp. 1–10. Springer, Heidelberg (2005)
5. Baker, M., Apon, A., Ferner, C., Brown, J.: Emerging Grid standards, vol. 38(4), pp. 43–50. IEEE Computer Press, USA (2005)
6. Sotomajor, B.: The Globus toolkit 4 programmer's tutorial (visited, March 2006), http://gdp.globus.org/gt4-tutorial/
7. Helsinger, A., Lazarus, R., Wright, W., Zinky, J.: Tools and techniques for performance measurement of large distributed multi-agent systems. In: Second International Joint Conference on Autonomous Agents and Multi-Agent Systems, pp. 843–850. ACM, New York (2003)
8. Xu, Z., Miller, B.P., Naim, O.: Dynamic instrumentation of threaded applications. In: Proc. 7th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, Georgia, USA (1999)
9. Author, N.: The behaviour of Java threads under Linux NPTL (visited, July 2006), http://www.amitysolutions.com.au/documents/NPTL_Java_threads.pdf
10. Tichy, W.F.: Should computer scientists experiment more? IEEE Computer 31(5), 32–40 (1998)
11. Timm, I.J., Pawlaszczyk, D.: Large Scale Multiagent Simulation on the Grid. In: Proceedings of 5th IEEE International Symposium on Cluster Computing and the Grid. IEEE Computer Society Press, Washington (2005)

12. Jarvis, S.A., Spooner, D.P., Keung, H.N.L.C., Nudd, G.R.: Performance prediction and its use in parallel and distributed computing systems. In: Proceedings of the 17th International Symposium on Parallel and Distributed Processing. IEEE Computer Society Press, Los Alamitos (2003)
13. Badia, R.M.: Performance Prediction in a Grid environment. In: 1st European Across Grids Conference, Santiago de Compostela (2003)
14. Puppin, D., Tonellotto, N., Laforenza, D.: Using Web Services to Run Distributed Numerical Applications. In: Kranzlmüller, D., Kacsuk, P., Dongarra, J. (eds.) EuroPVM/MPI 2004. LNCS, vol. 3241, pp. 207–214. Springer, Heidelberg (2004)
15. Marieto, M.B., et al.: Requirements Analysis of Agent-Based Simulation Platforms. In: Sichman, J.S., Bousquet, F., Davidsson, P. (eds.) MABS 2002. LNCS (LNAI), vol. 2581. Springer, Heidelberg (2003)
16. Page, B., Kreutzer, W.: The Java Simulation Handbook. Shaker Verlag, Aachen (2005)

E Pluribus Unum: Polyagent and Delegate MAS Architectures

H. Van Dyke Parunak[1], Sven Brueckner[1], Danny Weyns[2], Tom Holvoet[2], Paul Verstraete[2], and Paul Valckenaers[2]

[1] NewVectors LLC, 3520 Green Court, Suite 250, Ann Arbor, MI 48105 USA
{van.parunak,sven.brueckner}@newvectors.net
[2] Katholieke Universiteit Leuven, 3001 Leuven, Belgium
{danny.weyns,tom.holvoet}@cs.kuleuven.be,
{paul.verstraete,paul.valckenaers}@mech.kuleuven.be

Abstract. For the past few years, our research groups have independently been developing systems in which a multi-agent system (typically of lightweight agents) provides some functionality in service of a higher-level system, and often of a higher-level agent in that system. This paper compares our approaches to develop a more generic architecture of which our individual approaches are special cases. We summarize our existing systems, describe this architecture and the characteristics of problems for which it is attractive, and outline an agenda for further research in this area.

1 Introduction

Great ideas often occur to several researchers at the same time. At AAMAS06 and its workshops, NewVectors (NV) reported a modeling construct that represents a single domain entity with multiple agents, a "polyagent" [13, 14]. Katholieke Universiteit Leuven (KUL) described how individual agents in a manufacturing system could delegate certain tasks to a swarm of ant-like agents, a "delegate MAS" [5, 6].

The use of multiple agents to model a single agent is not new, but typically each of the multiple agents has a distinct function, which will be lost if that agent is eliminated. An example of this functional decomposition is the CODAGE system developed at the Laboratoire d'Informatique of the Université de Paris [8]. What sets our systems apart is that they use multiple agents *with the same function* to explore some combinatorial space through which the single agent must move—a planning space, or a space of possible futures, or a space of alternative decisions. The multiple agents conduct concurrent agent-based simulations to guide the decisions of the single agent. The number of agents modulates the performance and efficiency of the system, but not the functionality that is achieved.

The Latin phrase in our title applies to our topic in two ways. First, each of our systems uses a swarm of agents to provide a unified function, producing "one out of many" within the setting of a single application. Second, in seeking to unify our technical vision, we wish to produce "one (higher-level architecture) out of many" (or at least two) previously independent approaches. Realizing this second unification will

L. Antunes, M. Paolucci, and E. Norling (Eds.): MABS 2007, LNAI 5003, pp. 36–51, 2008.

enable us to pursue a common research agenda, develop common tools, and leverage off of one another's results.

Section 2 reviews our two approaches, with examples of how they have been applied in practice, and compares their motivations and foci. Section 3 discusses design considerations. Section 4 looks at future work. Section 5 concludes.

2 Polyagents and Delegate MAS

We begin by reviewing the two independent systems that motivate this analysis, and exploring their complementarities.

2.1 Polyagents at NewVectors

The Polyagent Model. Agent-based modeling conventionally associates a software agent with each entity in the domain. For example, an entity might be a soldier in a military domain, or a vehicle in a traffic model, or a person in a social simulation. A polyagent represents each domain entity with multiple agents: a single avatar that links it to the entity, and a swarm of ghosts that explore its alternative behaviors. Figure 1 shows the conceptual architecture of a polyagent.

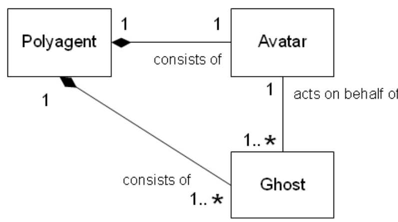


Fig. 1. A polyagent represents a domain entity with one avatar and multiple ghosts

The avatar persists as long as its entity is active, and maintains its entity's state. It may use sophisticated reasoning. Each avatar generates a stream of ghosts. Ghosts die after a fixed period of time or after some defined event. Each avatar controls the rate it generates ghosts, and typically has several concurrent ghosts. The ghosts are the "multiple agents with the same function" mentioned in our introduction.

Ghosts explore alternative behaviors for their avatar. In the applications constructed by NewVectors researchers, they are computationally simple, and interact through a digital pheromone field, a vector of scalars that depends on both location and time. Each ghost chooses its actions stochastically based on a weighted function of nearby pheromones, and optionally deposits its own pheromone. A ghost's "program" is the vector of weights.

The main benefit of representing a single domain entity with multiple agents is to multiply the number of interactions that a single run of the system can explore. Instead of one trajectory for each avatar, we now have one trajectory for each ghost. If each avatar has k concurrent ghosts, we explore k trajectories concurrently, leading to an increase in the number of interactions being explored at each step of an n-avatar system from n to k^n [13]. The avatar can base its decisions within a single run of the system on the multiple possible futures explored by its ghosts. In effect, the ghosts form an agent-based model that supports the decisions of the higher-level avatar agent.

The avatar can

- Modulate the number of its ghosts, their rate of generation, and the distribution of their parameters to control the exploration of alternative futures;
- Evolve them to learn the best parameters for a given situation;
- Review their behavior to estimate its own future experience.

NewVectors researchers have applied the polyagent model to three distinct domains: factory scheduling, robotic routing, and combat prediction.

Factory Scheduling. Our earliest application of polyagents did real-time job-shop scheduling [1] with three types of agents: processing resources, parts, and policy agents. Avatars of processing resources with different capabilities and capacities and avatars of parts with changing processing needs (due to rework) coordinate to optimize material flow through a complex, high-volume manufacturing transport system. Only part avatars deploy ghosts. Policy agents and resource (machine) avatars are traditional single agents, whose loads the ghosts explore in order to choose assignments for the parts.

Robotic Routing. Robotic vehicles must continuously replan their paths, as their knowledge of the environment changes due to limited sensor range and environmental change. In military applications, vehicles must navigate dynamically changing sets of targets and threats. Ants solve a similar problem in forming paths between nests and food sources [9]. Ants searching for food deposit nest pheromone while climbing the food pheromone gradient left by successful foragers. Ants who find food deposit food pheromone while climbing the nest pheromone gradient left by outbound ants. The pheromone fields collapse into a path as the ants interact. We have emulated this behavior to route robotic aircraft [16]. The agent controlling the robot sends out a stream of ghosts that execute the ant path planning algorithm in real time. The ghosts deposit nest pheromone as they move from the robot to seek out targets while avoiding threats, and target pheromone after they have found a target and are making their way home. Positive reinforcement among ghosts through their target pheromone leads to the emergence of high-density target pheromone paths that guide the robot.

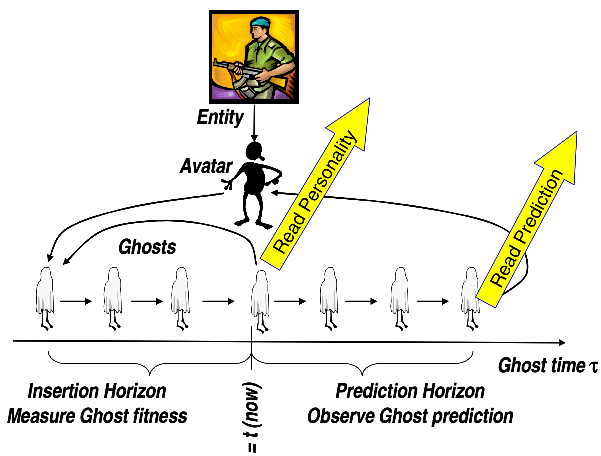


Fig. 2. Each avatar generates a stream of ghosts that sample the personality space of its entity. They evolve against the entity's recent observed behavior, and the fittest ghosts run into the future to generate predictions.

Combat Prediction. A commander in urban combat may have observations of the recent behavior of the adversary, and want to extrapolate these to predict future behavior. We use polyagents to evolve a model of the internal personality of each real-world entity and predict its future behavior [11]. Figure 2 shows the process. Ghosts live on a timeline of discrete pages indexed by τ (distinct from real time t) that begins in the past and runs into the future. The avatar inserts ghosts at the insertion horizon (say $\tau - t = -30$, the state of the world 30 minutes ago), sampling each ghost's parameters to explore alternative personalities of its entity.

The avatars record pheromones representing the observed state of the world on each page between the insertion horizon and $\tau = t$. The inserted ghosts interact with this past state. Their fitness depends not just on their own actions, but also on the behaviors of the rest of the population, which is also evolving. τ advances faster than real time, so eventually $\tau = t$, when the avatar compares each ghost with its entity's actual state.

The fittest ghosts have three functions.

1. Their personality estimates the personality of the corresponding entity.
2. They breed, and their offspring reenter at the insertion horizon.
3. They run into the future, exploring possible futures of the battle that the avatar analyzes to predict enemy behavior and recommend friendly behavior. In the future, the pheromone field is generated by other ghosts rather than avatars. Thus it integrates the various futures that the system is considering, and each ghost interacts with this composite view of other entities.

While many of the applications summarized in this paper deal with robotic systems, this application is important in showing the relevance of polyagents to social systems. In the past, multiple ghosts can be evolved against an entity's outward behavior to discover the underlying personality (including emotion). In the future, the ability of multiple ghosts from different avatars to interact with one another lets us explore multiple possible social interactions efficiently in a single run of the system.

2.2 Delegate MAS at KUL

Delegate MAS Model. A delegate MAS is a modeling construct that consists of a swarm of lightweight agents (ant agents) that provide a service for a higher level agent (the issuing agent) to support this agent in fulfilling its functions (Figure 3).

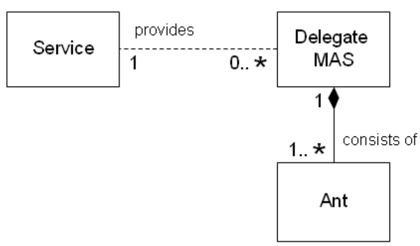


Fig. 3. A delegate MAS is a swarm of ant agents, and can provide various kinds of services

The issuing agent, representing a domain entity, may simultaneously have several delegate MAS, each rendering a specific service, and it may use a combination of delegate MAS to handle a single one of its concerns. For example in resource allocation problems, two distinct delegate MAS often prove useful: a swarm of exploration ants that seek out possible routings among resources on behalf of a task, and a swarm of intention ants that communicate a task's likely routing back to the resources. The issuing agent controls the number of ant agents, their program, and their parameter settings. The number is bounded at any instant. Each ant agent may only perform a bounded computational effort within its bounded lifetime and has a bounded footprint (memory). In other words, a delegate MAS is (computationally) efficient by design. However, the 'program' of an ant agent is not constrained otherwise.

The ants in a delegate MAS deposit, observe, and modify information (pheromone) in the virtual counterpart of the real world. This information can be any kind of data structure; it is not limited to vectors of scalars. Moreover, the environment in which the information is deposited may transform this information. For instance, bookings made by intention ants are inserted into a resource agent's planning scheme. All pheromone information has an expiration time (evaporation).

Finally and most importantly, a delegate MAS delegates in two manners. First, the issuing agent assigns a responsibility to the delegate MAS. Second, the ant agents delegate to the environment in which they travel and evolve. For instance, exploration ants query resource agents about expected processing times, processing results, transportation times, etc. Intention ants delegate the local scheduling to the resource agents. Exploring ants use product agents to evaluate routing options. This extreme usage of delegation enables a delegate MAS to cope with a dynamic, heterogeneous and unpredictable world. Its design nowhere assumes that data structures suffice to capture the diversity of the problem domain.

Real-Time Resource Allocation and Task Execution. Delegate MAS have been developed for applications that perform real-time resource allocation and that supervise the execution of the tasks requiring these resources. The resources and tasks are diverse and heterogeneous. Furthermore, competitive performance requires resource allocation and task execution to account for the specific nature of both the resources and the tasks, and their interactions. Moreover, the system must be able to deal with multiple allocations and task execution steps ahead in time. The applicability of delegate MAS outside this domain remains an open issue.

Until now, Manufacturing Execution Systems (MES) constitute the main application area for delegate MAS research. The first application targeted by the research addressed car body painting. The design of this pioneering implementation has been improved in subsequent developments, addressing a confection flow shop, a machine tool shop and a heat treatment facility respectively [18, 23]. Other application areas that have been explored are railway systems, traffic control and supply networks [3, 20]. We briefly discuss two example applications.

Manufacturing Execution Systems. In the MES prototypes, the issuing agents are PROSA (Product-Resource-Order-Staff Agent) [22] agents . All PROSA agents have counterparts in reality, which facilitates integration and consistency (indeed, reality is fully integrated and consistent). The main PROSA agents in the MES are:

- Resource agents reflecting the actual factory. They offer a structure in cyber space on which other agents can virtually travel through the factory.
- Order agents reflecting manufacturing tasks.
- Product agents reflecting task/product types.

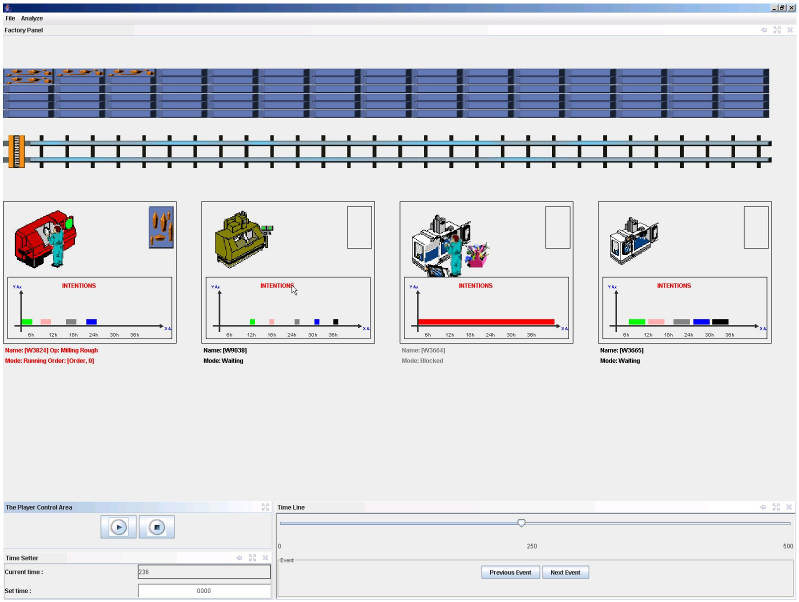


Fig. 4. Intention ants notify resource agents about the intentions of their respective order agent to occupy resources at a specific time slots in the near future. Resource agents use this information to self-schedule.

Both resource agent and order agents issue delegate MAS. A single agent may have several delegate MAS. Each delegate MAS has its own responsibility.

Resource agents use a delegate MAS to make their services known throughout the manufacturing system. Ant agents collect information about the processing capabilities of resources while virtually traveling through the factory. These *feasibility ants* deposit this information (pheromone) at positions in cyber space that correspond to routing opportunities.

Each order agent is an issuing agent for a delegate MAS in which *exploring ants* scout for suitable task execution scenarios. In addition, each order agent is an issuing agent for a second delegate MAS that informs the resource agents of its intentions: *Intention ants* regularly reconfirm bookings for slots at resources (Figure 4). Specific manufacturing execution systems employ additional delegate MAS to deliver case-specific services [21].

Traffic Control System. We have applied delegate MAS in an experimental traffic control system that proactively tries to predict and avoid road congestion [7]. Each car in the system is represented by a task agent, while road segments and crossroads are represented by resource agents. Task agents use resource agents to book a certain road in advance trying to avoid road congestion. Three types of delegate MAS are used: (i) resource agents issue feasibility ants to gather information about the underlying environment (which roads lead to which destinations). (ii) task agents issue exploration ants to gather information about the costs of possible routes; (iii) task agents issue

Fig. 5. A car at the circle (at the bottom right) has three possible routes: straight ahead, left, and straight on and then right. In the current situation, the car follows the route straight ahead, which has the minimal cost.

intention ants to book the best possible route. A booking must be refreshed regularly to maintain the reservation.

We have applied the delegate MAS approach to the Leonard crossroad, a well-known Belgian congestion point between the Brussels Ring and the E411 Motorway (Figure 5). Tests for a realistic morning peak scenario show a reduction of 26% of congestion time for an increase of only 4% of extra traveled distance.

2.3 Complementarities

The Polyagent and Delegate MAS models were developed independently of one another, and thus have differing objectives.

The main insight in the Polyagent model is that multiple representatives of a single agent can be used to explore alternatives for that agent. Thus it emphasizes the relation between the single persistent avatar and the multiple transient ghosts.

The main insight in the Delegate MAS model is that a swarm of agents can perform some service in support of a larger system. A delegate MAS does not include an avatar, though it is typically used to support the reasoning of a single agent. Thus a delegate MAS can be viewed as a part of a polyagent, the swarm of ghosts that is associated with an avatar.

Figure 6 makes this orthogonal relationship explicit.

The left-hand side of the picture shows the agent types in a conventional MAS, the PROSA architecture for manufacturing control [2]. The right hand shows the three delegate MAS that KUL has developed to support a PROSA system. Feasibility Ants explore connected sequences of resources, Exploration Ants estimate the quality of a particular sequence of resources on behalf of a task agent, and Intention Ants propagate the task agent's current intentions back to the resources so that they can schedule their availability.

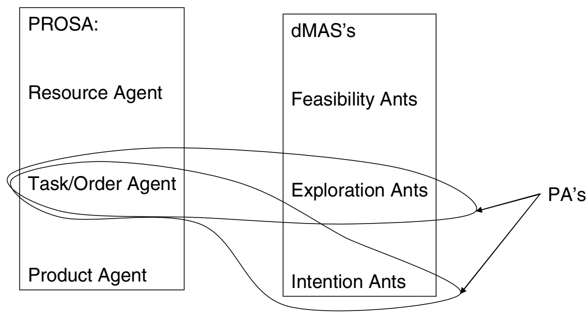


Fig. 6. A delegate MAS is a swarm of functionally homogeneous agents that explore multiple alternatives concurrently. A Polyagent uses a delegate MAS to explore alternatives for a single, usually more complex, agent, the Avatar.

This figure contains two polyagents (or a single polyagent with two types of ghosts), and three different relations between conventional agents and delegate MAS.

The Exploration Ants and Intention Ants both work on behalf of the Task Agent, which therefore constitutes an Avatar under the definition of a polyagent.

For Feasibility Ants two alternative designs can be considered. In [6], Feasibility Ants do not represent a single resource agent, but construct virtual routes through the network of resource agents. Thus they support the system as a whole, but are not part of a polyagent. An alternative implementation is possible in which each Resource agent sends out its own Feasibility Ants to deposit a quantitative pheromone on upstream resources. The relative strength of this pheromone would encode the relative distance of the node from the issuing resource, and thus enable Exploration Ants to assess the sequence of resources available from a given node. In this alternative implementation, a resource and its feasibility ants would constitute a polyagent.

Product Agents do not use the services of any delegate MAS, either directly or indirectly. They show that delegate MAS may be applied to part of a MAS, while other parts function using conventional MAS mechanisms.

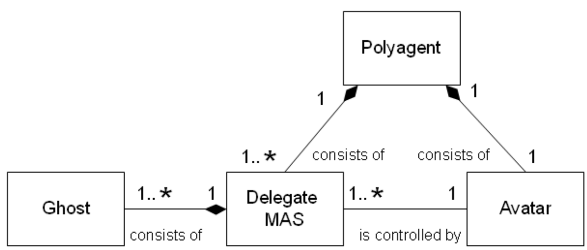


Fig. 7. An EPU system may use several Polyagents. Each Polyagent is the combination of an Avatar and one or more delegate MAS. Each delegate MAS may render a specific service for the Avatar, and the Avatar may use a combination of delegate MAS to handle a single one of its concerns.

To discuss these families of systems together, we need to establish some common vocabulary, which at points may differ from the vocabulary used in our previous papers. We propose (Figure 7):

A **delegate MAS** is a swarm of agents that provide some service for a higher-level agent system.

A **Ghost Agent** is one agent in a delegate MAS (where the term "ant" was used in the original Delegate MAS papers).

A **Polyagent** is the combination of a high-level agent with one or more delegate MAS. The recognition that a single polyagent can include several delegate MAS is an extension of the original polyagent model. All of the ghosts in a single delegate MAS have the same function, but the different delegate MAS in a single polyagent support different functions.

An **Avatar** is the agent paired with a one or more delegate MAS in a polyagent.

An **Entity** is something in the domain that is represented by an agent.

An **EPU system** is any system that draws on the constellation of ideas that we bring together in this paper. The acronym EPU recalls our motto, *e pluribus unum*.

When we need to refer to the characteristics of specific systems that one or the other of our teams has previously constructed, we will designate them by NV (NewVectors) or KUL (Katholieke Universiteit Leuven).

Integrating these two models yields clear benefits to both of our teams.

From the dMAS perspective, the concepts of Avatar and Polyagent in an EPU system provide explicit architectural constructs for designing systems.

From the Polyagent perspective, the delegate MAS construct encapsulates a swarm of ghosts; this providing a clean approach to associate different types of ghosts (in terms of different delegate MAS) with a single Avatar/Polyagent.

3 Design Considerations

As developers of real-world applications, we want to distill our experience into engineering guidelines for future exploitation of EPU systems. In this section we develop an integrated list of the domain characteristics for their application, which reflect similarities between our respective systems. The differences between our systems reveal the variability along which EPU systems can be developed.

3.1 Domain Characteristics

Some of the domain characteristics for applying EPU systems are shared with other multi-agent systems. Other characteristics are peculiar to our approach.

3.1.1 Common Domain Characteristics

Dynamism. Many domains are in constant change, and a MAS to manage them must be able to consider alternatives rapidly in order to adapt to this change. Using the parallelism of a delegate MAS is one way to support this dynamism.

Locality of Decision and Action. Agent systems naturally lend themselves to domains in which the primary information sources and the loci of action available to an agent are localized in some topology. This characteristic is especially valuable for EPU systems. The idea of using many representatives (the ghosts) to explore alternatives concurrently requires that the ghosts execute extremely efficiently, and both of our approaches use environmentally-based coordination via digital pheromones to enable light-weight ghosts. Such techniques are most effective when there is a strong correlation between an agent's location and its information and potential actions.

Going Concerns. Systems can conveniently be divided into problem solvers (typically activating a tightly coupled community of agents to reach an achievement goal, at which point the system can shut down) and going concerns (using a more loosely coupled set of agents to support a maintenance goal that requires constant attention) [10]. The ability of EPU systems to deal with dynamism makes them particularly valuable for handling going concerns.

3.1.2 Specific Domain Characteristics for EPU Systems

Temporal Constraints. The delegate MAS must run fast enough to be of service to the real-world system it is supporting. This requirement usually means that the ghosts must be able to more faster than the avatar. Otherwise, one might just as well let the avatar do the search.[1] One broad class of systems that satisfies this condition is systems dealing with the movement of physical entities. Physical constraints usually slow the movement of such entities so that ghosts, which can move at cyber-speed, can explore alternative trajectories faster than real time.

Space of Multiple Options. The replication of ghosts in an EPU system has the purpose of exploring alternatives concurrently. Such techniques are more useful as the problem space presents higher levels of combinatorial complexity.

Simulation-Friendly. The ghosts must be able to simulate the problem domain efficiently. This requirement is supported by two further characteristics:

- **Simulation Models.** Ghosts need efficient simulation models or other mechanisms to evaluate single options quickly, at least to a rough level of accuracy.
- **Constrained Problem Space.** If a system is highly constrained, its behavior may be relatively insensitive to details of individual agent actions, permitting the use of simplified models. We call this characteristic "universality" [17]. Such constraints may arise in two ways. First, the static structure of the environment may reduce the options that agents can follow. Second, the dynamics of the system may exhibit a few large basins of attraction leading to large equivalence classes in the space of possible solutions.

3.2 Variability in Applying EPU Systems

Contrasts between our systems reveal a number of degrees of freedom that can be exploited in engineering an EPU system for a particular application.

Locus of Functionality. KUL provides a plug-in architecture that allows the behavior of ghosts to be modulated by injecting additional functionality, while NV's implementations tend to have monolithic ghosts.

How Smart is a Ghost? Previous NV applications tend to use stigmergic agents, but this is not a requirement for application of EPU system's. The KUL plug-in architecture can use any computational method, so long as it respects the temporal constraints outlined in Section 3.1.

[1] This restriction is not strictly true. Even if ghosts only move at the same speed as their avatars, an EPU system may still be of some value on physically parallel hardware, enabling multiple alternatives to be evaluated in parallel with one another.

Ghost Interaction. Past NV applications of polyagents tend to take advantage of interactions among ghosts of the same avatar, mediated environmentally. For example, path planning depends on positive feedback among ghosts representing the same entity. In this approach, the ghosts in a polyagent not only *retrieve* knowledge from the environment, but actually *generate* new knowledge, reducing the decision-making load in the avatar. KUL ghosts could behave this way, but currently do not. The result is to place more responsibility for decision-making on the avatar. However, KUL's ghosts do interact with those of other types, in that exploration ghosts read symbolic pheromones written by feasibility ghosts.

Writing to the Environment. Closely related to ghost interaction is the question of whether or not ghosts can change the state of the environmental nodes that they visit. Three alternatives are available.

1. Ghosts can read from the environment but not write to it. This is the approach taken by the exploration ghosts in the KUL factory control system.
2. Ghosts can leave information in the environment for use by other ghosts, either of the same types or of different types. The first approach is used in NV's path-planning application, where it enables the generation of information (a routing) by positive feedback among the ghosts. The second is used in KUL's factory control system, where exploration ghosts read the accessibility information left by feasibility ghosts.
3. Ghosts can leave information for consumption by the environment, as do the intention ghosts in KUL's system.

Ghost Speciation. In KUL's factory control system, a single task agent has two separate delegate MAS, one for exploration and the other for propagating intentions. NV's applications have had a single type of ghosts for each avatar, and Brueckner's factory architecture [1] used pheromones deposited by a task agent's ghosts to estimate the level of intention that the task agent has for a given resource.

Time Management. Frequently, the space of alternatives over which ghosts are searching extends over time, and ghosts need some way to distinguish different future times from one another. One alternative, used by KUL and in Brueckner's early work, associates a timeline with each entity in the system that the ghosts may encounter, an approach we call *entity priority*. The other, used in NV's more recent systems, maintains a book of pheromone pages, each for a successive moment in time, and all entities are represented on each page. We call this approach *time priority*. The two alternatives pose an interesting trade-off. In an entity-priority system, a ghost on an entity's agent can easily compare the state of that entity at different times, but to compare different entities at the same time, it must move from one to the other. With time priority, a ghost can efficiently see the effect of multiple entities at a moment in time (to the degree that their pheromone fields overlap), but to see the state of one entity at different times, it must move through time.

These two models have evolved naturally in the domains in which KUL and NV have developed their systems. Two factors have motivated the respective decisions: the kinds of entities involved, and the nature of the reasoning to be done.

In most factory settings, resource agents and task agents behave in space-wise local but time-wise spread-out ways. This distinction makes it natural that timelines are local to agents representing application entities (resources, tasks). Indeed, the manufacturing machines are independent of one another, so entity priority enables faster selection of candidate time slots without significant sacrifice. In the combat modeling addressed in NV's latest systems, the most important entities are all combatants, and whatever Red does to Blue, Blue may also consider doing to Red. So the asymmetry that makes it feasible to manage time on one type of entity and not the other in the factory setting is not available. In addition, in combat, it is more important to understand how entities are interacting with one another than it is to examine a single entity's evolution over time, so time priority is preferable.

4 Future Work

Up to this point, EPU systems have been driven by the needs of specific problems. As we refine the approach into a reusable architectural approach, several issues require further investigation. We record them here as a roadmap for our own activity, and to invite other researchers to join us in extending this powerful approach.

Architectural Patterns for EPU Systems. The conceptual architecture of EPU systems described in Section 2.3 (Figure 7) represents a generic architectural pattern to develop agent systems for domains that satisfy the characteristics described in Section 3.1. The conceptual architecture describes the essential architectural elements (avatars, delegate MAS, ...) and relation types together with a set of constraints on how it may be used to build EPU's.

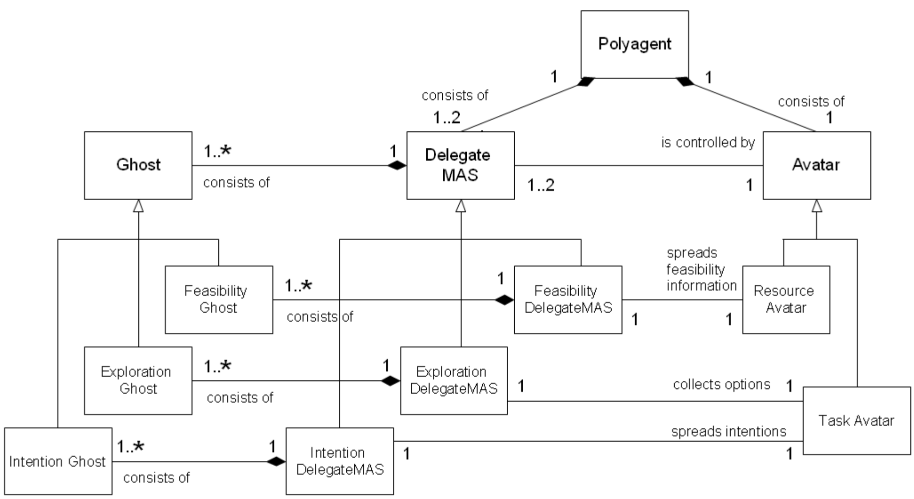


Fig. 8. Resource Avatars issue Feasibility Delegate MAS and Task Avatars issue Exploration and Intention Delegate MAS to obtain BDI-like functionality through the environment

From experience with particular classes of EPU systems, we have derived concrete instances of the general architectural pattern. Figure 8 shows one recurring pattern.

This architectural pattern consists of two specific Avatar types: Task and Resource Avatars that extend the general Avatar type. Three specialized delegate MAS provide BDI-like functionality through the environment to the Task Avatar:

- **Feasibility delegate MAS** are managed by Resource Avatars and build up beliefs about the environment (e.g., feasible production paths through a factory or a map in a traffic environment)
- **Exploration delegate** MAS are managed by Task Avatars and build up desires (e.g., useful paths through the factory to produce a particular product or useful routes in a traffic application)
- **Intention delegate** MAS are managed by Task Avatars and build up intentions (e.g., paths of booked resources to produce a product or selected routes in a traffic application)

This architectural pattern is one specific instance of the general EPU pattern. Such an architectural pattern provides a reusable asset for engineers to build new EPU applications. An interesting venue for future work is to derive other architectural patterns for EPU systems.

Dynamics. Any system with multiple interacting nonlinear components has the potential for complex dynamics that may either support or compromise the purpose for which the system has been constructed. EPU systems multiply the problem by embedding many MAS within a single system. Thus special attention must be paid to issues such as convergence, stability, and catastrophic discontinuities in behavior.

When an EPU system is used to reason into the future, one particular dynamic concern is of special interest. Nonlinear systems can exhibit chaotic behavior that causes the trajectories emanating from nearby points in state space to diverge arbitrarily far from one another, making long-range prediction impossible. Short-range prediction is still possible, and by continuously generating short-term predictions, one can reliably move ahead, as we have shown elsewhere [15]. But it is important to estimate just how far ahead predictions are meaningful, and at what point (the "prediction horizon") they become no better than random noise. We discuss some preliminary steps to studying this problem elsewhere [12]. Much remains to be done in enabling individual avatars to estimate how far into the future they should let their ghosts search, thus improving both their efficiency and the accuracy of the information that they produce.

Mechanism Design. Current EPU systems are closed, with a single developer who can ensure honest behavior on the part of avatars and ghosts. We expect the technique to become more widespread and enter the general toolbox of MAS developers, a prospect enhanced by the modular plug-in architecture being developed at KUL. EPU systems may have components developed by different parties, and in this case we need to give attention to the possibility that a delegate MAS may conduct deception on behalf of its avatar (or be deceived by the ghosts of another avatar). This problem is a generalization of the problem of deception and reliability in MAS in general, and techniques for mechanism design need to be adapted to this setting.

One promising approach is the use of reputation maintenance mechanisms. The attractiveness of EPU systems to going concerns (discussed in Section 3.1.1) means that avatars (or the organizations that issue them) can be expected to appear repeatedly, and other entities can learn their reliability. For example, in the KUL factory system, if task agents from a particular source regularly renege on an unusually high percentage of their reservations, it would be natural for resource agents to discount further reservations from those agents to avoid being undersubscribed. In this example, an agent's reputation transforms its statements into an expectation based on the statement, with an associated level of uncertainty.[19].

Guarantees vs. Adaptability. The large space of alternatives that a delegate MAS can explore can make a system more adaptable, but also less predictable. In many commercial settings, users require firm guarantees, at least on lower-bound behavior. Providing such guarantees requires ways to balance the exploration of the system against fixed behaviors that may be less adaptable but more predictable.[4].

Integration with Legacy Systems. Rarely will an EPU system completely supplant an existing system. It is more likely to be applied to part of the system, to give some advantage such as improving performance, adapting more rapidly to changes, or reducing variance. The techniques for achieving such integration are a rich field for study. In some cases, it may be possible to run the EPU system alongside the existing system and use its outputs selectively to adjust the legacy system. In other cases, one may embed components of the legacy system into the EPU system to provide specific functions, which must then interact with the functions being provided by the EPU system. Both approaches (and others) invite investigation.

Interference. The need for ghosts to run efficiently makes pheromone-based coordination attractive for delegate MAS, but leads to a problem. As ghosts explore alternative futures, they may deposit pheromones along different paths. Sometimes both paths are reasonable, but in other cases they are mutually exclusive, and it can be difficult to distinguish the two cases. More generally, the problem is that pheromones can accumulate and decay, but cannot interfere with one another, and over time the pheromone space can become muddy. As a result, when many futures are being explored, the space becomes muddy. If competing options could interfere, one could cancel out the other, avoiding the muddiness.

Autonomic Capabilities. Delegate MAS are a promising approach to addressing the self-X capabilities required for autonomic computing: self-monitoring, self-adjusting, self-healing, and so forth. In this case, the ghosts' focus is inward, on the components of the system, rather than outward toward the application domain. Developing idioms and mechanisms for these functions is an important research objective, and will greatly increase the potential for open EPU systems.

5 Conclusion

Traditionally, agents in a MAS are mapped to domain entities either one-to-one, or according to a functional decomposition. Sometimes it is advantageous to assign multiple agents with the same function to a single domain entity, in order to explore alternatives concurrently. When the single agent is controlling a physical system, its multiple representatives constitute a set of interacting agent-based simulations exploring the combinatorial space through which the single agent moves. Such an approach yields an EPU system. Between our two research teams, we have constructed EPU systems in several domains, including manufacturing control in several industries, supply chains, traffic control, robotic routing, and combat prediction. By examining our techniques together, we have been able to identify both the conditions under which such an approach is useful, and a number of design choices that are available to engineers who wish to exploit this technique for future applications. The approach opens up a range of interesting and important questions for further research.

References

[1] Brueckner, S.: Return from the Ant: Synthetic Ecosystems for Manufacturing Control. Dr.rer.nat. Thesis at Humboldt University Berlin, Department of Computer Science (2000), http://dochost.rz.hu-berlin.de/dissertationen/brueckner-sven-2000-06-21/PDF/Brueckner.pdf

[2] Brussel, H.V., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P.: Reference Architecture for Holonic Manufacturing Systems: PROSA. Computers In Industry 37(3), 255–276 (1998), http://www.irit.fr/TFGSO/DOCS/TFG2/TFGIISO_Valckenaers.pdf

[3] DeSwert, K., Valckenaers, P., Saint German, B., Verstraete, P., Hadeli, K., Van Brussel, H.: Coordination and Control for Railroad Networks Inspired by Manufacturing Control. In: Proceedings of IEEE 2006 Workshop on Distributed Intelligent Systems, pp. 201–206. IEEE, Los Alamitos (2006), http://doi.ieeecomputersociety.org/10.1109/DIS.2006.21

[4] Hadeli, K.: Bio-Inspired Multi-Agent Manufacturing Control System with Social Behaviour. Thesis at Katholieke Universiteit Leuven, Department of Department of Mechanical Engineering (2006)

[5] Holvoet, T., Valckenaers, P.: Beliefs, desires and intentions through the environment. In: Proceedings of 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, pp. 1052–1054 (2006)

[6] Holvoet, T., Valckenaers, P.: Exploiting the Environment for Coordinating Agent Intentions. In: Weyns, D., Van Dyke Parunak, H., Michel, F. (eds.) E4MAS 2006. LNCS (LNAI), vol. 4389. Springer, Heidelberg (2007)

[7] Huard, E., Gorissen, D., Holvoet, T.: Applying delegate multi-agent sytems in a traffic control system. CW 467, Katholieke Universiteit Leuven, Leuven, Belgium (2006), http://www.cs.kuleuven.be/publicaties/rapporten/cw/CW467.abs.html

[8] Kant, J.-D., Thiriot, S.: Modeling one Human Decision Maker with a MultiAgent System: the CODAGE Approach. In: Proceedings of Fifth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2006), Hakodate, Japan, ACM Press, New York (2006)

[9] Parunak, H.V.D.: 'Go to the Ant': Engineering Principles from Natural Agent Systems. Annals of Operations Research 75, 69–101 (1997), http://www.newvectors.net/staff/parunakv/gotoant.pdf

[10] Parunak, H.V.D.: From Chaos to Commerce: Practical Issues and Research Opportunities in the Nonlinear Dynamics of Decentralized Manufacturing Systems. In: Proceedings of Second International Workshop on Intelligent Manufacturing Systems, Leuven, Belgium, pp. k15–k25, Katholieke Universiteit Leuven (1999), http://www.newvectors.net/staff/parunakv/ims99.pdf

[11] Parunak, H.V.D.: Real-Time Agent Characterization and Prediction. In: Proceedings of International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), Industrial Track, Honolulu, Hawaii. ACM Press, New York (2007)

[12] Parunak, H.V.D., Belding, T., Brueckner, S.: Prediction Horizons in Polyagent Models. In: Proceedings of Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), Honolulu, HI (submitted, 2007)

[13] Parunak, H.V.D., Brueckner, S.: Concurrent Modeling of Alternative Worlds with Polyagents. In: Antunes, L., Takadama, K. (eds.) MABS 2006, at AAMAS 2006. LNCS (LNAI), vol. 4442, Springer, Heidelberg (2007), http://www.newvectors.net/staff/parunakv/MABS06Polyagents.pdf

[14] Parunak, H.V.D., Brueckner, S.: Modeling Uncertain Domains with Polyagents. In: Proceedings of International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2006), Hakodate, Japan. ACM Press, New York (2006), http://www.newvectors.net/staff/parunakv/AAMAS06Polyagents.pdf

[15] Parunak, H.V.D., Brueckner, S., Matthews, R., Sauter, J., Brophy, S.: Real-Time Evolutionary Agent Characterization and Prediction. In: Proceedings of Social Agents: Results and Prospects (Agent 2006), Chicago, IL, Argonne National Laboratory (2006)

[16] Parunak, H.V.D., Brueckner, S., Savit, R.: Universality in Multi-Agent Systems. In: AAMAS 2004, pp. 930–937. ACM, New York (2004), http://www.newvectors.net/staff/parunakv/AAMAS04Universality.pdf

[17] Parunak, H.V.D., Brueckner, S., Sauter, J.: Digital Pheromones for Coordination of Unmanned Vehicles. In: Weyns, D., Van Dyke Parunak, H., Michel, F. (eds.) E4MAS 2004. LNCS (LNAI), vol. 3374, pp. 246–263. Springer, New York (2005), http://www.newvectors.net/staff/parunakv/E4MAS04_UAVCoordination.pdf

[18] Peeters, P., Van Brussel, H., Valckenaers, P., Wyns, J., Bongaerts, L., Kollingbaum, M., Heikkila, T.: Pheromone based emergent shop floor control system for flexible flow shops. Artificial Intelligence in Engineering 15(4), 343–352 (2001)

[19] Saint Germain, B., Valckenaers, P., Verstraete, P., Bochmann, O., Van Brussel, H.: Supply Network Control, An Engineering Perspective (2004)

[20] Saint Germain, B., Valckenaers, P., Verstraete, P., Van Brussel, H.: Resource coordination in supply networks. In: Proceedings of IEEE International Conference on Systems, Man and Cybernetics, pp. 1972–1978 (2004)

[21] Valckenaers, P., Hadeli, K., Saint Germain, B., Verstraete, P., Van Brussel, H.: Emergent short-term forecasting through ant colony engineering in coordination and control systems. Advanced Engineering Informatics 20(3), 261–278 (2006)

[22] Valckenaers, P., Van Brussel, H.: Holonic manufacturing execution systems. CIRP Annals of Manufacturing Technology 54(1), 427–432 (2005)

[23] Zamfirescu, C., Valckenaers, P., Van Brussel, H., Saint Germain, B.: A case study for modular plant control. In: Mařík, V., McFarlane, D.C., Valckenaers, P. (eds.) HoloMAS 2003. LNCS (LNAI), vol. 2744, pp. 268–279. Springer, Heidelberg (2003)

A Multi-agent Model for the Micro-to-Macro Linking Derived from a Computational View of the Social Systems Theory by Luhmann

Vera Lúcia da Silva[1], Maria das Graças Bruno Marietto[2], and Carlos H. Costa Ribeiro[1]

[1] Technological Institute of Aeronautics, Computer Science Division, São José dos Campos, São Paulo, Brazil
{verals,carlos}@ita.br
[2] Federal University of ABC, Centre of Mathematics, Computation and Cognition, Santo André, São Paulo, Brazil
graca.marietto@ufabc.edu.br

Abstract. Even though questions related to the micro-macro relationship are fundamentally important for understanding and modeling social systems, important theoretical gaps on this issue continue to exist in the Agent-Based Social Simulation (ABSS) area. To address it we consider a model that explicitly links the micro-level individual communications with the macro-level social phenomena, building up an important formal tool for analyzing social processes and their dynamics in a bottom-up approach. The model lays out a series of key elements from Niklas Luhmann's Systems Theory and concepts from ABSS.

Keywords: micro-macro link, emergence, social systems.

1 Introduction

Two important research foci in Agent-Based Social Simulation (ABSS) are the analysis of social phenomena formation from individual behavior, and the study of the social influence upon individual cognitive processes ([14], [11]). In focusing such questions ABSS is in contact with a long standing problem in Social Science, namely the micro-macro link. In sociological studies, this link is usually presented under three perspectives [14]: micro-to-macro, macro-to-micro and the dialectic issue regarding emergence and social causation. The micro-to-macro perspective (emergence) refers to the upsurge of macro phenomena from interactions among individuals. The macro-to-micro perspective (social causation) refers to the influence of the macro-structural phenomena upon individual members of the society. Finally, the dialectic between emergence and social causation refers to the mutual interactions between individuals and macro-social structure.

Studying the micro-macro relationship is of paramount importance for multi-agent systems (MAS) and ABSS, as it helps building up more general models for social organizations by dealing with issues such as self-organization, social structures and norms, emergence of cooperation and coordinated action, and so on. However, despite

L. Antunes, M. Paolucci, and E. Norling (Eds.): MABS 2007, LNAI 5003, pp. 52–68, 2008.

the importance of an explicit modeling for micro-macro relationship, standard MAS models often assume an individualistic perspective. Simulations and theories based on such characteristic do not provide a conceptual and systemic structure that accounts for explaining how micro-properties play a role on forming macro-phenomena and vice versa ([14], [11]).

Recent ABSS studies have considered multi-agent models for macro-phenomena emergence (*e.g.* [2], [4], [10], [12], [14], [15]) considering that social reality emerges from an extensive plethora of actions at the individual level. However, there is no model yet that fully represents the interaction between the micro and macro levels. In fact, such a general model does not seem to be close to come up either from Sociology or ABSS theories. This can be justified for this interaction is a broad and complex question, generating investigations from a myriad of interdisciplinary fields.

According to Turner [16], an attempt at minimizing the complexity of such a model could separately consider each perspective from the sociological theory, and only after that consider an unifying model. Following Turner's proposal, in this paper we present a formal model to represent the micro-to-macro link (emergence) in a multi-agent organization.

The proposed model is based on a semantic framework encompassing the fundamental elements of the Social Systems Theory by Niklas Luhmann and concepts from ABSS.

Luhmann's Social Systems Theory can be characterized as an appropriate theory for modern social relations such as those elicited by the concept of frontierless societies. It is also argued that it is a general theory for social systems. For the development of the work presented herein, Luhmann's Theory is indeed adequate, as it presents aspects that allows for a computational understanding of the micro-macro link. Among those aspects we emphasize the view of society and social systems provided by the theory, which presents society not as composed by agents (people), but rather by the results of the communications carried out by these agents. This permits a focus on the communications generated by the so-called psychic systems of the agents, and on the way such communications modify the expectation structures of the social systems.

In this sense, the first contribution of this model is related to Social Sciences. We argue that, from this model and implementations of its simulation, it is possible to analyze Luhmann's Theory *vis-à-vis* its consistency, precision and predictive power for the formation of macro-phenomena. A second contribution is related to ABSS, and considers that the proposed model might be a basis for a more objective analysis of social simulations with respect to formation and development of emerging phenomena in social systems.

The rest of this paper is organized as follows. The second section presents the general ideas of Luhmann's Theory, and based on such elements we construct a model for how micro-properties assist on the emergence of macro-properties. In the third section, a comparative analysis of models that focus on the micro-macro relation is presented. The paper is ended with some final considerations and proposals for future work.

2 A Multi-agent Framework for the Micro-to-Macro Link

The theoretical model proposed in this section aims at an explicit mapping for emergence relations in social systems. It is based on the Social Systems Theory by Niklas Luhmann [8] and on concepts from ABSS area.

2.1 The Social Systems Theory by Luhmann

Luhmann's Social Systems Theory contrasts with other sociological theories by not considering the society as composed by individuals, but only by the communications generated by their respective psychic systems (PSs). Thus, society is possible because PSs share some understandings and perspectives, via communication mechanisms.

According to Luhmann, the PS self-produces a systematic unit, called consciousness, through the recursive production of thoughts. Then, PSs consist of consciousness, with thoughts as the elements of reproduction: I think, therefore I am (*Cogito, ergo sum*). On the other hand, social systems are shaped by the result of the interactions among PSs. Luhmann considers society as a broad social system, split into diverse functional social systems, such as Economy, Religion and Science. PSs and their interaction belong to the micro level, whereas social systems are emergent results at the macro level.

Social systems and PSs coevolve, each being necessary for the other's evolution. Thus, there is neither communication without consciousness nor consciousness evolution without communication. This co-evolution is possible because both systems use meaning or sensing as evolutionary elements [8]. Meaning is constituted by distinction, i.e., the denomination of "something" as relevant and its differentiation from other phenomena. Meaning constitution is self-referenced, because the denomination needs other meanings a priori. Distinction is a system's internal event from which it can observe the world, making a differentiation between "this" and the "rest" (meaning).

Social systems and PSs are autopoietic and operationally closed. Autopoiesis means that the system produces its own elements and structures in an operationally closed process [7], with the aid of their own elements. In Luhmann's proposal, the autopoiesis concept substitutes the open/closed system's concept. The concept of operational closure means that operations that produce new elements depend on former operations of the same system. Thus, neither system can act out of their boundaries. However, according to [8] the systems are *not* closed for cognitive (PS) and structural (social systems) evolution. Autopoiesis and operational closure assures to the system the autonomy for selecting and interpreting only relevant information.

The relationship between the system and the environment is mediated by the Structural Coupling (SC). The SC only irritates the system, but it cannot determine its internal processes. The system makes the reading of the SC through the recognition of perturbations, confronting intra-system expectations with identified perturbations. The SC between social system and PS is called interpenetration, and among PSs is called interaction. Systems can be perturbed by the environment, and can dissipate information over the environment.

Communications are the constituent elements in a social system. The form of communication can be thought of in a traditional manner, either using the standard emitter-channel-receptor idea from Information Theory, or following Luhmann's approach of viewing communication as a socially built operation, needing at least two Agents-P to complete a communication cycle. In this approach, agents send messages to the environment and not directly to receiving agents, because each agent cannot control other agent's attitudes and expectations. Dissipated messages for the environment can either perturb other agents or not. The receiving agent is in charge of accepting and interpreting messages according to its internal rules.

According to Luhmann, communication is autopoietic because it is created only in a recursive context from other communications, in a network whose reproduction requires cooperation from each isolated communication. A communication is the synthesis of three selection processes. First there is information selection, i.e., selection of the message content related to a topic of interest. Then there is utterance selection, that is, selection of the form in which the agent transmits the information-content message. This expression is represented by an action, such as speech, gesticulation, etc. Finally, there is selection regarding comprehension of the information contained in the received message. This completes the communication cycle. The third selection process is performed by the receiving agent and others selections by the sender agent. In isolation, none of the constituent selection processes constitute a communication. On the other hand, once a message is dissipated to the environment, it is viewed simply as data. An Agent-P (receiver) must select and understand the message for that data to become information.

Social systems reproduce their basic elements (communications) as events. An event is the minimal socially possible temporal atom, happening uniquely and for a period necessary and sufficient for its identification [8]. Communication events happen simultaneously and in great numbers in a society. Thus, they cannot be accumulated indiscriminately in a social system for obvious computational reasons. It is therefore necessary to create mechanisms which efficiently keep the information from such events, generating social structures. Such structures correspond to social forms that a society, in its evolution, can assume, being variable according to conditions of time and place.

In contrast to events, social structures can be temporally extended and yet modifiable. They are composed by expectation networks and are formed via event repetition. To allow for such repetition, events must follow patterns in a communicative process, as communication patterns reinforce selection mechanisms as far as world interpretation is concerned [8]. In fact, according to [8] the concept of a "structure" is based on an organization by patterns. In our context, social systems are identified as expectation structures. Expectations are happenings or behaviors expected by an agent which directs its attitudes. Behaviors that are different from the expectations are treated as expectation deviations.

Communications with different topics are generated by PSs. This implies a need by social systems to select communications related to topics that are relevant to their functional activities. Communication event selection is based on two elements: binary code and symbolically generalized communication media (SGCM). Each social system has a unique code with which it filters, processes and generates communications. The outside world vision of a functional system is the one that its binary code identifies.

On the other hand, SGCM are semantic devices which motivate the acceptance of a communication by PSs, using knowledge based on the reality of a social system. Each social system establishes a specific SGCM to make it possible the internal operations, as boundaries of the systemic differentiation. As an example, power is a SGCM which increases the acceptance of a communication in Politics. Its binary code is *To Have* or *Not to Have*. Theory and method is a SGCM which increase the acceptance of a communication in Social Sciences. Its binary code is *Valid* or *Not Valid*.

According to [8] interactions are temporally organized into episodes. An episode is a sequence of communication events through a timeline, arranged by expectation scenarios and governed by a central topic. For instance, we could consider an episode related to a client visit to a bank. Such episodes model standard behavioral situations in a social system, considering its expectations structure.

2.2 Agent Architecture for Psychic Systems

An agent for representing psychic systems (Agent-P) is composed by four modules, as depicted in Figure 1: Perturbation Module, Memory Module, Psychic System Module and Dissipation Module.

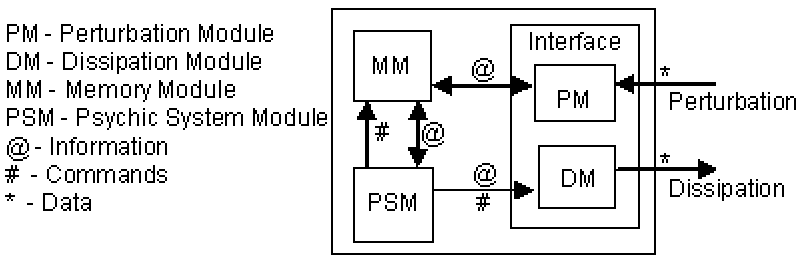


Fig. 1. Architecture of Agent-P

A. Perturbation Module (PM). The PM manipulates the perturbations identified by the agent. Such perturbations result from the interaction with other agents.

This module makes it possible that three actions can be executed on the data received: a) data selection, b) recognition of the form of the received message, c) understanding of the data by the agent (as information), either for generation or modification of internal meanings. Notice that if any of these actions is performed without success, the data is discarded and the perturbation process is finished.

A PM is formed by a set of submodules, as illustrated in Figure 2: Data Analyzer and Selector, Temporary Information Base, Utterance Manager and Perturbation Analyzer. The agent performs data selection from the environment using the Data Analyzer and Selector. Perceived data are those present in its perceptual field and whose content is a perturbation, that is, data that provide information which is somehow relevant for the agent. Data selection is guided by the rules of systemic closure present at the Rules Base in the Memory Module (see item B).

When data is selected, it is kept in quarantine while being analyzed by the other perturbation submodules. The Temporary Information Base keeps this information until when the Utterance Manager (UM) submodule starts analyzing it. Such analysis

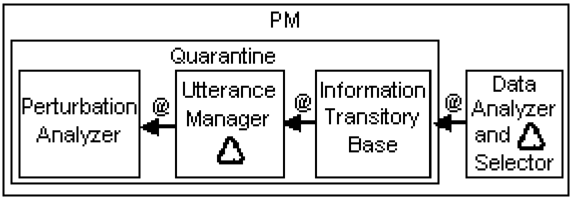


Fig. 2. Perturbation Module

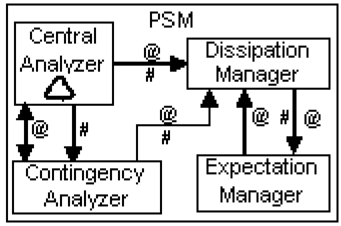


Fig. 3. Psychic System Module

aims at verifying and validating the form of the message, in such a way that it can be understood by the agent. For instance, UM verifies if the message is a gesture, an image, etc. Once an information is identified, UM send it to the Perturbation Analyzer, whose function is to comprehend the information. If the information is considered to be valid, it is added to the agent's Belief Base. A valid information is an information that is understood and that produces or modifies meanings in memory.

B. Memory Module (MM). MM is composed by three databases: Knowledge Base, Belief Base and Rules Base. Knowledge stored in the Knowledge Base can be of one of two kinds: knowledge of the agent about itself (goals, personal data, etc) and social knowledge, that is, information about how the agent understands society. Notice that both kinds of knowledge are socially constructed, as meaning is derived from social interaction among agents.

The Belief Base has yet unconsolidated information in the network of meanings that are considered to be true. Such beliefs can be either about the agent itself or about society. Beliefs are generated by the Perturbation Analyzer, as soon as an information is understood by the agent. The Rules Base contains a set of rules which establishes the systemic closure of Agent-P, thus defining a limit for what is interesting and evolutionary. Rules can be reactive, functional or dynamic. Reactive rules refer to information selection related to survival and existence. These rules determinate to Agent-P that there is information that must be processed timely.

Functional rules establish which characteristics are necessary to attend the agent's expectations, defining its identity. Such rules can determinate a larger interest for certain information, and therefore a weight can be established for each rule, in proportion to a degree of desirability of information. Finally, dynamic rules are those that allow for the input of new or transitory relevant information. Those are generated by the PSM (see item C) when, during a processing to reach a goal, the agent faces a topic too different from the one(s) defined for its role.

C. Psychic System Module (PSM). The PSM manages the agent autonomous processing, and is formed by the submodules Central Analyzer, Contingency Analyzer, Dissipation Manager and Expectation Manager. Figure 3 depicts its organization.

The Central Analyzer manipulates data and directs decisions and actions towards agent goals and objectives. This submodule has the characteristics of an agent modeled according to the traditional lines of Distributed Artificial Intelligence that is, developed for performing functions according to its role in a society of agents. During the operation of the Central Analyzer some uncertainties can be generated. According to Luhmann uncertainties lead to communication because, to reduce uncertainty, the agent establishes a communication process with its environment to acquire additional (and hopefully disambiguating) information. To take care of this situation, the Central Analyzer verifies the possible behaviors which can become uncertain. The Contingency Analyzer (CA) receives this information and defines which must be considered as uncertainties. If it identifies an uncertainty, it triggers the Dissipation Manager, sending the information to the environment.

The Dissipation Manager controls agent dissipations to the environment. It is triggered by the Central Analyzer when there is a need of an interaction between the agent and the environment, or by the CA in the search for additional information. Once the message receivers are identified, this submodule activates the Expectation Manager to take care of possible communication alternatives.

The Expectation Manager considers the agent motivations for communication, and its desired expectations. Then, it lists a set of alternatives for communication, based on its intentions, experience and former knowledge about the agent that must receive the message. This list is sent to the Dissipation Manager. Finally, once the alternative is selected, the Dissipation Manager directs the information to be dissipated and a pointer to the receiver to the Dissipation Module.

D. Dissipation Module (DM). The DM sends information from the agent to the environment. This dissipation can be motivated by a) internal and autopoietic agent deliberation; and b) selection of new information that stimulates new communication for uncertainty reduction.

Information to be transmitted is encapsulated in a message, which is processed or materialized via an action. To treat the phases of information transmission from the agent to the environment, the DM was divided in two submodules: Message Analyzer and Actuator. The Message Analyzer converts internal information into communication information. For this, it first chooses the best way of externalizing information, considering the characteristics of the receiver agent. Then, it formally generates the message to be dissipated and delivers it to the submodule Actuator, which finally sends the information to the environment via realization of an action.

2.3 Agent Architecture for Social Systems

The architecture of an agent for representing a social system, namely Agent-S, is composed by four main modules: Perturbation Module, Dissipation Module, Memory Module and Social System Module. Figure 4 illustrates this architecture.

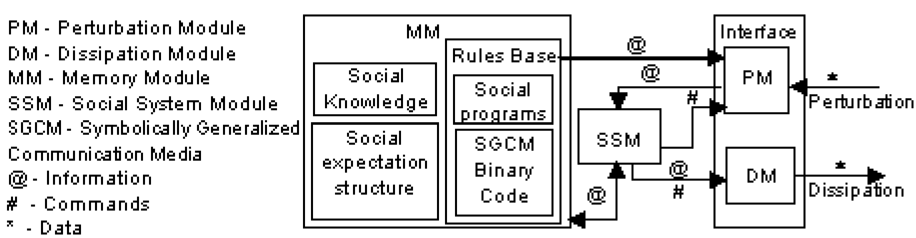


Fig. 4. Architecture of Agent-S

A. Perturbation Module (PM). This module selects communication events from the environment that are interesting for the Agent-S. It is divided into three submodules (Figure 5): Analyzer and Selector of Events, Perturbation Analyzer and Event Transitory Base.

In the model proposed herein, a communication event is composed by context, communication, participants and binary code. The context refers to the contour situation where communication occurs. It is composed by the communication scope (e.g leisure, school, family) and time, which defines the period when the communication took place by a time variable t, of variable granularity. For instance, three levels of granularity can describe communications that occur along the morning, afternoon or evening. The participants are the Agents-P involved in the communication, more specifically a single emitter and one or more receiving agents. Notice that a single message in the environment can be understood by more than a single Agent-P, and each understanding corresponds to a new event. Any event can then be selected and assessed by an Agent-S, which can by turn modify its expectation structure. A binary code is used by the receiving agent-P to identify the message with a meaning which characterizes the function of a specific social system.

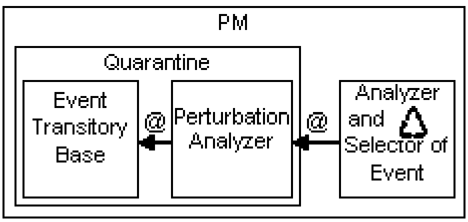


Fig. 5. Perturbation Module of Agent-S

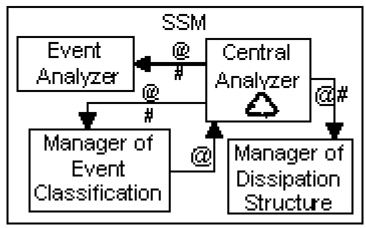


Fig. 6. Social System Module

The submodule Analyzer and Selector of Events selects the events which are interesting for the Agent-S. Event selection is based on the SGCM and binary code present in the Rules Base of the MM (see item B). Such events are originated from the interpenetration between Agent-S and Agent-P. For instance, in the formation of the SS *Economics* every event whose SGCM is *money* or *property* with binary code *To Have* or *Not To Have* would be selected. The Perturbation Analyzer filters and fine tunes the captured events. For this aim, it uses social programs composed by rules accepted by the SS. Finally, the Event Transitory Base stores transient events so that they can be analyzed by the Social System Module (item C).

B. Memory Module (MM). The MM stores information from the Agent-S, and it is composed by three submodules: Social Expectations Structure, Rules Base and Social Knowledge.

The submodule Social Expectations Structure stores expectation structures generated as result of many social interactions, which can be consolidated, for instance, as norms, directives or social values. From a computational point of view, such structures can be implemented as expert systems, Petri nets, state machines, etc. In our model expectations are represented as production systems with probabilistic production rules (i.e., with an associated degree of uncertainty). The rules are generated via abstraction of information contained in events.

We do not consider here the genesis of the evolutionary process of the expectation structures, and therefore we assume the existence of a predefined base of structures. It is not our aim to analyze the origin of an evolution process in a SS, but rather to analyze how expectation structures can be modified. Rule modifications are originated by the upsurge of new expectations in the context of interactions among Agents-P. The changes in a rule can occur in one of the following ways:

- Creation of a then: From a set of prior expectations, changes and insertions are allowed only in the consequent part of an expectation rule; and
- Modification of a rule probability: From a set of prior rule probabilities, value changes can occur according to interactions among Agents-P.

The rules are separately organized in modules, according to specialization. For instance, a "Law" SS can have modules for rules from Civil Law, Criminal Law, and so forth.

The Rules Base stores the binary codes and SGCM, which together with the social program establish the identity of the Agent-S with respect to its functional characteristics and systemic closure. The Social Knowledge Module contains information related to other agents, their relationships and social roles.

C. Social System Module (SSM). The SSM coordinates the execution of the remaining modules, and allows for the observation and manipulation of events for modification of expectation structures. It is composed by four submodules, as illustrated in Figure 6: Central Analyzer, Event Classification Manager, Event Analyzer and Dissipation Structure Manager.

The Central Analyzer is constant processing, allowing the observation by the agent of the selected events. For this aim, it retrieves events to be processed from the Event Transitory Base. It then directs these events to the submodule Event Classification Manager, which performs an initial classification of events based on the subareas which compose the social expectations structure. This initial classification is then fed back to the Central Analyzer, which then directs the classified event to the Event Analyzer. The Event Analyzer then performs the understanding of the selected event and inserts the relevant information in the structure of Agent-S. More specifically, it alters the expectation rules, either creating a new *then* proposition or modifying a rule weight.

To manage an episode, the Agent-S has social programs that control the event sequence. The episode controller program is triggered every time that an event is manipulated by the Event Analyzer. Once operative, the program verifies if the event is part of an ongoing episode, or if it is the beginning of a new episode. In both cases it accesses the structure of the Agent-S and verifies the possible expected behaviors for the event, according to the agent's roles and prior experiences. With this information possible interaction scenarios among the Agents-S are assembled, and during the simulation it is verified if any of these scenarios took place. If that is the case, the modification of the expectations structures is performed immediately. Otherwise, an expectation deviation took place, and the Agent-S learns from this experience modifying its expectation structure or even dismiss the episode as "non-understandable". The control of event change operation in an episode can be performed via techniques such as scripts, state machines and so forth.

The Central Analyzer submodule is also responsible for verifying the need for making social expectation structures available. If that is the case, it activates the submodule Dissipation Structure Manager which operates the release of expectation structures to the environment.

D. Dissipation Module (DM). DM makes the social expectation structures available to the environment.

2.4 Life Space

One of Luhmann's Theory contributions for reducing systems analysis complexity is the substitution of the distinction part/total by the distinction system/environment. In this new approach, everything that is not a part of a system, according to its systemic closure, belongs to its environment. Thus, the environment of a PS or social system is composed of all the others social systems and PSs.

In the model proposed in this work, the distinction between system and environment becomes operational through the component denominated Life Space (LS). Besides Luhmann Theory, the construction of the LS uses the Field Theory concepts by Kurt Lewin [6]. Agents-P and Agents-S are inserted in the LS, providing a structure that (i) allows for relationships among the agents, via structural coupling; (ii) defines outline conditions related to the place where the agents are allocated.

The LS exists independent of each agent's limitations of representing it internally, or even of perceiving it totally. It has an updated and objective representation of the agency state, and makes it possible to implement the communication model proposed by Luhmann, where there is no the direct emitter-channel-receptor communication. The LS makes the link between the dissipated message by an agent and the possibility of that message to be observed by one or more environment agents, according to each agent's autonomy and interest.

The LS possesses three basic functions. The first is to represent, in an explicit way, the MAS complete topological environment. It is important to emphasize that, for each MAS developed with base in that model, it may be necessary to change the meaning of this topological representation. The second function is to allow explicit structuring of the information dissipation to the environment. All messages (in its several expression forms) sent by MAS agents for the environment stay stored in different and specific channels of the LS. For instance, for the agents modeling that represents human beings, the LS can present channels for speech messages, visual messages, and so on (see Section 2.4.1). The third function is explicitly to model the agents' perturbation process. For that, the LS makes available stored information in specific channels to Agents-P.

2.4.1 Interactions of Agents-P in Life Space

As said before, interaction allows for relationships among PSs. Agents-P interact among themselves and, starting from these interactions, communications are generated. Such communications are constituent elements of social systems, forming Agents-S.

The interaction among Agents-P involves dissipation and perturbation actions, as illustrated in Figure 7. An interaction happens when a message dissipated by an Agent-P is selected by a Perturbation Module of another Agent-P. For Interacting Agents-P use the LS as support for the communications.

Regarding dissipation, when an agent wants to communicate he transforms its thoughts, intentions and expectations in information that can be understood by other agents. It sends a message for the environment, which is a result of an action (to speak, to do signs or gestures, to write, and so on). That message is sent for the LS, that stores it in the specific channel for the expressed form type. Regarding the perturbation, the agent observes its environment constantly, maintaining contact with the LS (and not with each agent directly). The Data Analyzer and Selector Module of each Agent-P informs to the LS which information type it wants, passing details such as the detection range of their sensors. If the obtained data is of the Agent-P's interest, it will begin to be processed in the other execution phases of the Perturbation Module (see Section 2.2, item A).

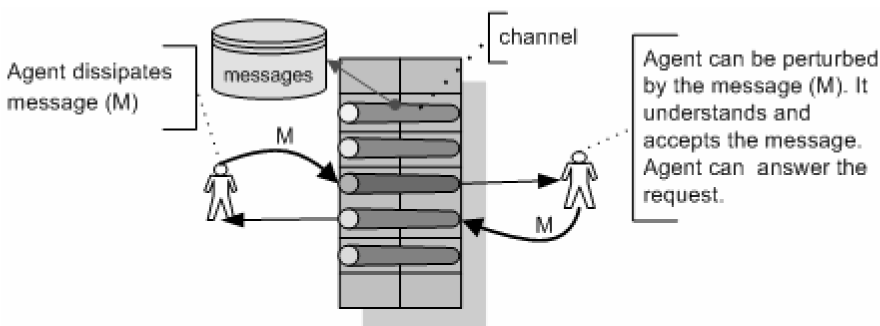


Fig. 7. Interaction among Agents-P through the LS

2.4.2 Interactions of Agents-P and Agents-S in Life Space

As said before, interpenetration allows for relationships among PSs and Social Systems through a transmitting system [8]. Social and psychic systems coevolve. However as they are autopoietic systems, they evolve from separate instances, although they are structurally coupled and belong to the environment of each other.

In our model, Agents-P internalize social expectations acquired through participations in communications, whereas expectation structures are formed in Agents-S. Agents-P retain in its knowledge base a simplified and personalized representation of each social system that they participate in. Interfacing between Agents-P and Agents-S is treated by a temporary communication media, more specifically a blackboard system. Communication events (as described in section 2.3, item A) generated by interactions among Agents-P are sent to this area, where Agents-S can select them. It is a function of LS to compose the data structure to represent the communication event and to send it for this area.

3 Application of the Proposed Multi-agent Model: Case Studies

The computable theoretical model presented in this work includes fundamental elements of Luhmann Theory. In this sense, it can be considered a "general" model, because it aims at describing the guiding ideas related to social systems. Therefore, in case this model is adopted as base for the modeling and implementation of simulations and/or applications, elements such as dissipation, perturbation, operational closure, autopoiesis and social structures can be taken into account.

However, for each domain and problem considered, it will be necessary that the proposed model modules are adapted to the context and corresponding contour conditions. As mentioned in Luhmann [9], "*...it is only in this way that one can confront general theories with the realities of concrete areas of investigation to see whether the theories are functional and what modifications they might need*".

With the objective of discussing the viability of implementing social systems and having as base the proposed model, in the next subsections some practical situations are presented where the semantic framework of Luhmann Theory can be applied.

3.1 Cyberspace as a Social System

Currently the connections through the Internet among folks, organizations and individuals not only occur almost instantly, but also the quantity of the possible connections grows of exponentially. As highlighted in [5], this implicates that changes in the form and frequency of the flows among two points have repercussions in roads and other very distant connections from the initial point. In contrast to a previous world structured and analyzed by geographical boundaries, currently there is a global configuration having as base a complex digital net of communications and cooperation. This virtual global world, denominated cyberspace, forms a social system with new constructions, for instance, in the social, cultural, political and economical realms.

In cyberspace the subject is not another person or machine (psychic systems), but the communications generated through e-mails, forums, discussions lists, chats, and so on. As psychic and social systems are autopoietic and operationally closed, the cyberspace doesn't regulate machines and humans' thoughts and behaviors directly. It just guides the communicative process that it will make the acceptance of determined messages and information more probable than others.

Formation of the cyberspace's social structures depends on the dynamics of the social communication generated by the psychic systems, and this dynamics is different from that of "traditional" social systems. Thus, one of the challenges in the modeling of cyberspace's expectation nets is to characterize new boundaries of interaction of spatial, temporal and cultural orders. These borders will interfere directly in the sense attribution performed by the systems, as well as in the understanding of actions (and posterior communication) in the virtual environment.

3.2 Organizational Climate as a Social System

Human organizations like social groups, tribes and enterprises possess an own and singular identity, formed by the several combinations of their variables and their assumed values, generating culture and organizational climate. The climate formation [1] in organizations is a subject of study in Social Sciences.

The proposed model can be used for the simulation of the climate formation in an organization. The climate is modeled as a social system formed by the generated communications from interactions among organization's members. The organizational climate can be represented by an Agent-S and belongs to the macro level, and the individual climate [13] can be formed as cognitive schemes into the psychic systems of the organizations' members (Agents-P). Individual climate represents the meaning that the individual attributes to the elements of its working environment as a consequence of the answers of their judgment systems, sustained in their basic values. Both climates emerge from the interactions among organization agents.

For model validation, we developed a MAS prototype for simulating the formation of an organizational climate in a small company which operates on Mechanics and Machining. This represents a typical case study for social systems formation based on interactions among members of the organization, leading directly to the micro-macro problem. The main issue is that the result of agents behaviour can affect the collective climate of a group and, consequently, of the company.

For the simulation we selected the sector of the company which deals with part production. It encompasses the fabrication process itself and the group of employees that take part in this process. The group consists of four employees: production manager (PM), quality control expert (QCE), and two workers: machinist and miller. These roles are organized in a hierarchical manner: the PM is in the top and has the workers as subordinates. The QCE is a consultant on process and final product quality.

The proposed theoretical model is used as for designing the agents that represent the company's employees, i.e., their respective psychic systems (Agent-P). The organizational climate is modeled as an emerging social system, represented in the MAS as an Agent-S agent. Agent communications directives are also provided by the model, via message exchanging and the interpenetration space among agents-P and the agent-S.

The prototype implementation was carried out in the Swarm-Java platform, and FIPA-ACL was used for Agents-P communication.

4 Comparative Analysis of Models for the Micro-macro Linking

An analysis of the literature on ABSS makes it possible to classify models that consider the micro-macro link into two categories, based on the level of explicitation of the constituent elements of this link. In both categories the models are usually sociologically inspired, based on theories such as symbolic interactionism, Luhmann's Theory among others.

In the first category linkages between individual and social structures are still under-specified, but there is a concern on representing agents with rules and mechanisms for internal generation of knowledge and structures that are socially built and shared. Thus, the micro-macro link is considered although there is no explicit representation of the macroscopic level. Such models are therefore individualistic, in the sense that social subjects are constructed and monitored having as sole basis individual consciousness. Symbolic interactionism is a source of inspiration for such models (e.g. [4], [15]).

However, this individualistic approach is rather limiting, as it does not consider that the society is more than a sum of individual interactions. In an attempt to surpass such limitation, models in the second category try explicitly to represent the macro level as autonomous and as a reality on its own, not deduced just from individual realities. The model proposed in this paper and in [12] are representatives of this category.

A comparison between models in such categories can help understanding how ABSS studies are using sociological theories as a base for technological design in MAS. Such analysis is important insofar, as mentioned in [3], sociological theories can be instrumental in redesigning the use of algorithmic models and simulation techniques. In the context herein, such a comparison also evidences how the proposed model is inserted in the state-of-the-art of the micro-macro problem in ABSS.

Table 1 presents this comparative overview. The following elements are considered: communication forms, micro level explicitation, macro level explicitation, micro-to-macro link and macro-to-micro link.

The communication form refers to agent's interaction inside the agency. Some possible forms of communication in MAS are direct and indirect message exchanging (standard), dissipation and perturbation (proposal by Luhmann).

The elements micro level explicitation and macro level explicitation refer to the existence of components that represent objectively characteristics of agency's micro and macro level, respectively. The micro level describes agents and their interaction behaviors. The level macro describe and explain which forces are responsible for stability (processes of reproduction) and change of societies (dynamical aspects), making explicit social structures and generated collective processes.

Table 1. Comparative analysis of MAS models considering micro-macro linking

	First category		**Second category**	
	[4]	**[15]**	**[12]**	**Proposed model**
Communication forms	indirect and direct message exchange	direct message exchange	Exchange of messages among agents and use of one whiteboard	Perturbation, dissipation, and blackboard
Micro level explicitation	Producing agents	Agents and local variables	There is not an agent model, just considering their communication	Agents-P
Macro level explicitation	Not available	Not available	Mirror-Holons agents	Agents-S
Micro-to-macro link	Not available	Not available	Messages capture and other observable actions, via whiteboard	Message capture and other observable actions, via blackboard
Macro-to-micro link	Not available	Not available	Communication of social structures, via whiteboard	Not available

The micro-to-macro link refers to the existence of components that make explicit the macro-phenomena emergence, starting from agents and their interactions. On the other hand, the macro-to-micro link requests the explicitation of the social structures and macro-phenomena influence in individual agents.

For model comparison, in the first category we analyzed the models presented in [4] and [15]. In the second category we analyze both the model presented in this paper and the one presented in [12].

Starting with the first category, we notice that in [4] the agent role formation process is modeled in a society called SISTER in which producing and consuming agents simulate a simple exchange economy. Communication is performed via direct and indirect (signal exhibition and reading) message exchange. In the micro level the producing agents and respective interactions are defined explicitly, with knowledge acquired by genetic algorithms and coevolution processes for implementing a double induction of signal meaning. There is no social structure that stores any global society knowledge. Each agent has private knowledge, acquired from many interactions with other agents. As there is no macro level explicitation, micro-to-macro and macro-to-micro interactions are not directly evaluated. However, common symbol meanings can emerge in the knowledge representations for each agent.

The work reported in [15] analyzes conflicts in a society, considering that mass conflict is given by the intensity of collective mobilization (CM). CM intensity is an emergent property that results from interaction among agents. With respect to the form of communication, there are interactions among agents in two groups, and randomly connected in a social network. Communication is performed via direct message exchange among agents. Emerging factors are represented by global variables, and therefore there is no explicitation of a macro level, as global variables do not fully characterize social structures and collective behaviors.

In [12] a MAS (HolOMAS – Holonic Open Multiagent System) is proposed to represent the social level. This MAS is composed by Mirror-Holons agents, which observe the communication among agents and, from such observations, derive social expectation structures. Mirror-Holons agents access a shared memory (whiteboard) which has two functions: a) to allow agents to store messages and other observable actions; and (b) to allow a HolOMAS to send events to other agents. This model does not present the agents explicitly, but only the observed communications. The micro-to-macro link is modeled via the observation of communication among agents, and the macro-to-micro link is done through communication of the social expectation structures to the agents.

The model proposed in this paper focuses social construction using as a basis Luhmann's Theory, considering that the macro level (Agents-S) originates from interactions among Agents-P (micro level), thus inferring a micro-to-macro link. This link can be tracked considering the communications established among the PSs, their capture by social systems in a *blackboard*, and posterior formation of expectation structures from communication. In this model, there is no explicit macro-to-micro link. Communication among Agents-P occurs by perturbation and dissipation processes, in which agents in their interactions send messages to the environment. However, there are no guarantees that the message will be selected, analyzed, understood and accepted, being a responsibility of the receiving PS to accept and interpret the message according to its internal rules.

5 Conclusion

A general observation from what has been said so far suggests that a critical challenge for future research is to further work out a clarification of the micro-macro link when developing ABSS models. Social Sciences theories – more specifically from Sociology – can be an inspiring source for building up wider and more adaptive solutions. However, as pointed out in [10], the symbiosis between Sociology and ABSS requires formal and computable models: "*...for stating theory precisely, connecting its concepts in rigorous intellectual structures, and identifying both hidden assumptions and unexpected consequences*".

Such formalization is not only a matter of scientific rigor, but a dialectic way of building up structures and semantic architectures from a social theory. This dialectical perspective considers that algorithmic modeling of social theories can help on creating a critical and renovating thinking of social abstractions and constructs. Likewise, a better understanding of social universe formation can be instrumental for answering questions regarding ABSS issues such as scalability, cooperation, coordination, negotiation and conflict resolution, among others.

The model proposed in this paper is a formal model of fundamental elements of Luhmann's Social Theory. It is a semantic framework that can be used, for instance, in computational platforms for multi-agent simulation (*e.g.* Swarm, CORMAS), allowing a more objective analysis of the formation of macro-phenomena from relationships in the microscopic level. This is important because most classical simulation programs and contemporary models of agent-oriented software engineering are not concerned with the exploration of emergent, non-anticipated structures. Indeed, such active exploration of different results and qualitative concepts is the focus of socio-cognitive models in the ABSS area [11], with models in which many emergent macro-phenomena are visible only after extensive execution of simulations.

The comparative analysis of Section 3 reinforces the importance of the use of social theories in ABSS, aiming at revitalizing multi-agent technologies more adapted to social modeling. Amongst the basic elements of a social approach, in the comparison we considered the level of explicitation of constituent elements of the micro-macro link. As a result two model categories were identified, and we performed a comparison of the models pertaining to each of them. Specifically, the proposed model is in the second category, which considers an explicit representation of the macro level.

Acknowledgments. We are grateful to CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico – for the financial support for carrying out this research (Grant no. 14227/2003-6).

References

1. Ashforth, B.E.: Climate Formation: Issues and Extensions. Academy of Management. The Academy of Management Review 10(4), 837–847 (1985)
2. Barber, M., Blanchard, P., Buchinger, E., Cessac, B., Streit, L.: Expectation-Driven Interaction: a Model Based on Luhmann's Contingency Approach. In: JASSS, pp. 9(4) (2006)
3. Castelfranchi, C., Conte, R.: Cognitive and Social Action, London (1995)

4. Duong, D.V., Grefenstette, J.: SISTER: a Symbolic Interactionist Simulation of Trade and Emergent Roles. In: JASSS, vol. 8(1) (2005)
5. HargittaI, E., Centeno, M.A.: Defining a Global Geography. In: American Behavioral Scientist, vol. 10(44) (2001)
6. Lewin, K.: Field Theory in Social Science. Tavistock Publications, London (1952)
7. Luhmann, N.: The Autopoiesis of Social Systems. In: Geyer, F., van der Zouwen, J. (eds.) Sociocybernetic Paradoxes. Sage, London (1986)
8. Luhmann, N.: Social Systems. Stanford University Press, Stanford (1995)
9. Luhmann, N.: Law as a Social System. Northwestern University, School of Law Northwestern University Law Review 83(1&2) (1988-1989)
10. Malsch, T.: Naming the Unnamable: Socionics or the Sociological Turn of/to Distributed Artificial Intelligence. In: AAMAS, vol. 4, pp. 155–186 (2001)
11. Marietto, M., et al.: Requirements Analysis of Agent-Based Simulation Patterns: State of the Art and New Prospects. In: Sichman, J.S., Bousquet, F., Davidsson, P. (eds.) MABS 2002. LNCS (LNAI), vol. 2581, pp. 125–141. Springer, Berlin (2003)
12. Nickles, M., Weiss, G.: Multiagent systems without agents: mirror-holons for the compilation and enactment of communication structures. In: Fischer, K., Florian, M., Malsch, T. (eds.) Socionics. LNCS (LNAI), vol. 3413, pp. 289–313. Springer, Heidelberg (2005)
13. Parker, C.P., et al.: Relationships between psychological climate perceptions and work outcomes: a meta-analytic review. Journal of Organizational Behavior 24, 389–416 (2003)
14. Sawyer, R.K.: Social Emergence: Societies as Complex Systems. Cambridge University Press, Cambridge (2005)
15. Situngkir, H.: On Massive Conflict: macro-micro Link. In: BFI Working Paper no. WPD 2004. SSRN (2004), http://ssrn.com/abstract=601784
16. Turner, J.H.: Handbook of Sociological Theory. Kluwer Academic - Plenum Publishers, Dordrecht (2002)

Agent-Based Simulation of Group Learning

Maartje Spoelstra[1,2] and Elizabeth Sklar[3]

[1] Dept of Computer Science
Vrije Universiteit Amsterdam
De Boelelaan 1081a, 1081 HV Amsterdam, NL
[2] TNO Defence, Security and Safety
Department of Command and Control
Oude Waalsdorperweg 63
2509 JG The Hague, NL
maartje.spoelstra@tno.nl
[3] Dept of Computer and Information Science,
Brooklyn College, City University of New York
2900 Bedford Ave, Brooklyn, NY, USA 11210
sklar@sci.brooklyn.cuny.edu

Abstract. We construct groups of *simulated learners* that model the behaviour of humans acting in various learning environments, with the aims of studying *group learning* and focusing on the effects of different *goal structures* on individuals and groups of learners. Three sets of research objectives are investigated: (1) imitating the behaviour of human learners with a multiagent simulation by modeling characteristics outlined in pedagogical literature; (2) comparing the outcomes of simulated learners operating with different goal structures; and (3) exploring factors that influence the behaviours of simulated learners acting in groups, such as group size and composition, as well as the inclusion of team rewards. We ran a series of experiments as part of this investigation, which are outlined herein.

1 Introduction

Multiagent simulations based on computational representations of human actors and characteristics of social environments can provide useful approximations of large-scale population studies or fine-grained behavioural studies. Although necessarily abstracted to varying degrees, these types of simulations can be useful either as a pre-cursor to experiments involving real humans or as a means of analyzing previously collected data sets [17]. The work described here examines *group learning* and focuses on the effects of different *goal structures* on individuals and groups of *simulated learners*. Three research objectives are investigated: first, imitating behaviours of human learners in a multiagent simulation by modeling characteristics outlined in pedagogical literature; second, comparing the behaviours of simulated learners responding to different goal structures; and third, exploring factors that influence the behaviours of simulated learners acting in groups, such as group size and composition and the inclusion of team rewards.

Earlier related work describes "SimEd", an environment that emulates interactions between simple artificial learners and abstract knowledge domains [12]. Students and teachers are modeled as agents acting within a complex social system, namely the

L. Antunes, M. Paolucci, and E. Norling (Eds.): MABS 2007, LNAI 5003, pp. 69–83, 2008.

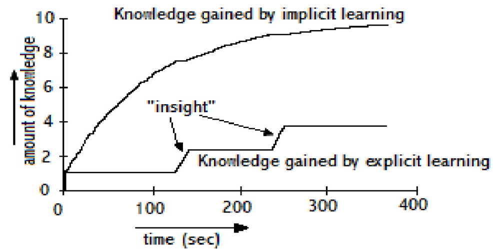
education system; and their behaviours are controlled by features such as emotion, motivation and ability [9]. Here we expand upon this line of work in two main ways. First, we model peer-to-peer interactions—whereas the previous work only models the results of student-teacher interactions. Second, we base the details of the present simulation on the large body of existing research on "group learning" that has been conducted by developmental psychologists, education researchers and cognitive scientists. Thus our models of human learners are grounded in empirical and controlled experimental studies well-documented in the literature—whereas the previous work abstracted many of the details of the human "agents" and was based on canonical views of classroom activity. Our work is related to the fields of cognitive modeling and user modeling; however the goal here is not to build or augment an intelligent tutoring system but rather to build a simulation system in which we can explore the interplay between various characteristics of learners and the environments in which they progress.

Our approach differs from other work that describes "simulated students". VanLehn *et al.* [18] present an analysis of machine learning systems that behave like human students, identifying two inputs of such systems (a student's knowledge prior to the learning event that will be simulated and the instructional intervention that led to the learning event) and two outputs (the student's behaviour during and updated knowledge after the learning event has occurred). Subsequent work employs this notion for analyzing skill acquisition, for example emulating learning from error correction [11]. Uses for systems that simulate students can be grouped into three categories [18]: *teacher training* [4,3], *peer tutoring* (where the peer is a simulated student) [20], and *instructional design* [19]. Peer tutoring is the most closely related to the work described here.

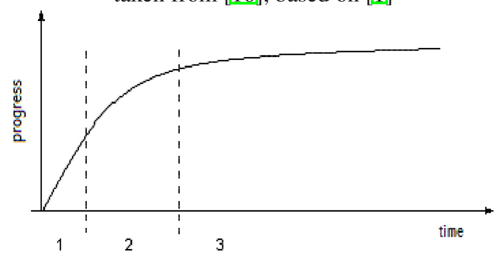
In the work presented in this paper, we examine aspects of group learning, comparing a range of different reward mechanisms for individuals and groups, as well as various heterogeneous (vs homogeneous) group compositions. First, we provide an overview of relevant pedagogical literature describing the characteristics of individuals and goal structures in group learning situations. In section 3, we describe our *group learning model* and the design of a simulator which we constructed for experimenting with the model. Section 4 presents some results, and we close with a brief discussion.

2 Background

Our group learning model is based on several important pedagogical theories of human learning and skill acquisition as well as applications of these theories to implementations of instructional and learning processes in a classroom. Fitts [6] describes a theory, involving three phases for physical skill learning in adults, which has influenced many others studying skill acquisition. Fitts' theory claims that when learning a skill, human development goes through an "early" phase, an "intermediate" phase and a "late" phase. In the early phase, the emphasis of the learning task is on understanding instructions and on establishing the proper cognitive set for the task, resulting in a better grasp of the task at hand. The latter is done by performing a series of short, simple tasks and trials, like an introduction to the task to be learned. In the intermediate phase, people learn to associate parts of the skill they are acquiring with different stimuli. The late phase involves the perfection of the task learned.



(a) implicit and explicit learning models, taken from [16], based on [1]



(b) stages of cognitive development combined with the stage-theory of [6] and [1], inspired by [16]

Fig. 1. Models of knowledge acquisition during learning, i.e., "progress". The horizontal axes represent the passage of time; the vertical axes represent the amount of knowledge acquired by the learner. In figure (b), "1" represents the initial stage of learning; "2" is the associative stage; and "3" is the autonomous stage.

Anderson [1] describes three similar stages in the context of the acquisition of cognitive skill. He names and explains the three phases slightly differently: the first phase is called the "cognitive" stage. A characteristic of this phase is verbal mediation, which enables the learner to clarify instructions for herself. The second stage is the "associative" stage, in which skill performance is "smoothed out": errors in the initial understanding are detected and overcome. In this phase, no verbal mediation is necessary anymore. The last phase is the "autonomous" stage, in which the learner gradually improves in performance of the skill. As a part of this stage, Anderson mentions the "procedural stage" which applies purely to the increase in speed with which the skills are performed.

Taatgen [16] expands on Anderson's learning model and describes the outcomes of learning in terms of "explicit" and "implicit" learning (see figure 1a). He uses the term "implicit learning" for unconscious and unintentional learning, whereas in "explicit learning", goals and intentions determine what is learned. In an educational system, we can say that explicit learning gives rise to the cognitive outcomes of goal structures and implicit learning gives rise to the affective outcomes.

In many pedagogical studies, researchers distinguish between several levels of *ability* because some learners progress more quickly than others: some studies mention three levels ("high", "medium" and "low") [10], but the most common are two levels of ability ("high" and "low") [2]. In our study we chose to focus on two levels of ability.

Another factor influencing learning behaviour is the level of difficulty of the information being processed in comparison to the level of development of the learner. The *zone of proximal development* is defined by Vygotsky as "the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under adult guidance or in collaboration with more capable peers" [21]. In order for a learner to process new information optimally, the level of the information should be such that it can be grasped by the learner's present zone of proximal development. *Collaborative* activity amongst learning peers promotes growth because peers are likely to operate within each other's zones of proximal development and interactions can help reinforce knowledge and smooth learners' transitions from the early to later stages of skill acquisition.

Throughout much of the pedagogical literature, three factors are cited as influencing individual human learning: *cognition*, *motivation*, and *emotion*. These are often referred to as the "trilogy of mind" [9]. All three elements influence the learning process equally. The zone of proximal development can be seen as the cognitive component of this trilogy. Motivation and emotion are factors that depend for a large part on the learning environment for a learner and on interaction with others while learning.

The design and implementation of one or more *goal structures* is a part of the educational process within the classroom and can focus on (1) individual, (2) cooperative and/or (3) competitive aspects. With *individual goal structures*, each student can set his or her own learning goals, regardless of the goals of others. With *cooperative goal structures*, students work together on a task. One inherent feature of this cooperation is that students only obtain their personal goals if the students with whom they work also obtain their own goals. If implemented correctly, the cooperative goal structure is generally believed to be beneficial for students' learning processes [7,13,2] because they not only learn the concept that is in fact the objective of their cooperation, but also the interactive skills necessary to cooperate. With *competitive goal structures*, students working individually can obtain their goal by scoring well in relation to others, even if others fail to achieve their goals and even if students block others' successes; not always negative, competitive goal structures can be very motivating for some students [7]. The three goal structures all vary in the amount and type of interaction that takes place among learners: with an individual goal structure, there is no interaction; with a competitive goal structure, there are only competitive interactions; with a cooperative goal structure, interactions are designed to help all participants.

Goal structures can be implemented in different ways, according to how an instructor wishes to use them to help teach concepts and motivate her students. One teaching methodology that can implement all of the aforementioned goal structures is the STAD learning method [13]. The STAD (Student Teams Achievement Divisions) method has five major characteristics, which can be implemented as 4-5 sequential phases in the learning process that, collectively, are performed iteratively:

1. *Teacher presentations*—the initial phase of the learning process in which a teacher explains the concept to be acquired;
2. *Student teamwork or individual work*—the phase in which activities designed to facilitate learning are undertaken by one or more students, working alone or in groups;

3. *Quizzes*—the phase in which the teacher evaluates the progress made by each student;
4. *Individual improvement*—the phase in which individuals receive recognition (from the teacher and/or their peers) for any progress they have made; and, optionally,
5. *Team recognition*—the phase in which teams are ranked and "prizes" (or some other form of recognition) are bestowed upon team members—this phase is only relevant when the "cooperative goal structure" is in place and students are working in teams.

A typical feature of the STAD learning method is that before learning a concept, students are each given individual "targets" to reach, customized according to their ability. Because these targets are personalized, every student has as much chance of performing well on her quiz as her peers do on theirs. Team recognition is based on collective performance as well as individual performance relative to personalized targets. This means of assessing progress and determining rewards was used in the current research to simulate the learning of a series of concepts by groups of students in an environment with various reward structures.

3 Group Learning Model

Our *group learning model* is designed based on the pedagogical theories highlighted above. In this section, we first outline the parameters that define *agents* acting in the simulator; each agent represents an individual human learner. The model of cognitive development—i.e., progress made by individual learners—which underlies our simulation is illustrated in figure 1b. In the initial stage of learning (labelled "1" in the figure), a large amount of new knowledge is introduced to the learner in a short amount of time, mostly in the form of instructions; hence the slope of the curve is quite steep. In the associative stage ("2" in the figure), instructions are formalized and made part of the learner's own skills; the slope of the curve decreases because it takes more time to formalize and associate actions with the new information and because the amount of new information that is presented also decreases. In the autonomous stage ("3"), the learner does not learn new things but constantly elaborates the present knowledge.

The second part of this section discusses the *learning environment* (or *instructional model*) in which the agents interact. The instructional model that we simulate is based on the notion that first a student is exposed to some new knowledge (we use the term "concept" to indicate a unit of knowledge)—this could be like reading about it in a textbook chapter or hearing a teacher give a lecture on the topic—then the student has a chance to practice working with the knowledge, such as, for example, doing a homework assignment or lab work, writing a computer program, answering question at the end of a textbook chapter; and finally the student's new knowledge is assessed. We label this evaluation a "quiz", but it could also mean the teacher marking a homework or lab assignment—the term "quiz" here refers to the stage at which the teacher (or instructional module in an intelligent tutoring system) gains feedback on whether the student is acquiring the new knowledge or not.

The final part of this section describes the simulator and how it demonstrates the group learning model.

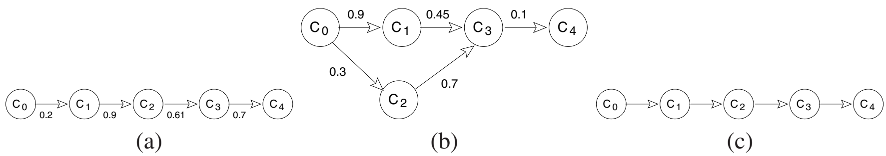


Fig. 2. Sample graphs of concepts; the first two (a and b) are taken from [12] and the third (c) is adapted to illustrate the implementation described here

The learning environment resembles a classroom context, in which students have to progress through a certain number of *concepts*, with varying difficulties, within a time frame indicated by "ticks" (units of time). The notion of *concept difficulty* is based on an abstract representation of a knowledge domain introduced in [12]: A concept comprises a small bit of information, such as the spelling or meaning of a word or an arithmetic equation, and is represented as a node in a graph (see figure 2a and 2b). Each concept has a difficulty value between 0 (easiest) and 1 (most difficult). Concepts are related to each other, and the closeness of their relationship is indicated by a weight on the link between concept nodes.

In the work presented here, a concept represents more information than in [12]—instead a concept is comparable to a topic in a geography class or a mathematics principle. Here, the difficulty of each concept is defined as one of three values: easy ($= 0.3$), intermediate ($= 0.6$) or hard ($= 0.9$). Another difference between the concepts used in [12] and the current research is the dependency between the concepts; in the current research, the concepts are not related to each other. In other words, learning and understanding one concept does not have an influence on a learner's understanding of the next concept. This can be seen in figure 2c, in which the first five concepts (numbered C0 to C4) are shown. The fact that the concepts are not related, could be compared to a series of class sessions in a day at school, in which sessions of different subjects follow each other, and the understanding gained from geography class has no bearing on a learner's understanding of mathematics.

3.1 Agents

Each agent is defined by a number of parameters, as detailed below.

- **ability**—indicates whether the agent has "high" or "low" aptitude. This value does not change during the simulation and can be thought of like IQ (intelligence quotient), i.e., a value that indicates a learner's innate aptitude and remains constant over their lifetime.
- **improvement**—reflects the general increase in knowledge throughout the learning of a new concept.
- **progress**—is the cumulative value of improvement (shown in figure 1b).
- **base_score**—represents the score of the quiz the student took before the first concept and after each concept.
- **improvement_score**—is the outcome of the quiz taken by the learner in the evaluation phase of each concept cycle (see below) and is the value used to increment

the **base_score** after completing a concept. It is a combination of the improvement of the learner during the concept presentation and the gained **understanding** (see below).

– **understanding**—is gained by the learner when learning a concept of which the **difficulty** (explained in the next subsection) falls within her **zone** (see below). Another way in which a learner gains extra understanding is when explaining things to peers in the cooperative goal structure. The understanding gained by a learner depends on the current improvement of that learner and the help provided to others. The gain in a learner's understanding may be at the cost of that learner's improvement; therefore, it does not always pay off for a learner to help others. At the beginning of each concept, understanding is set to 0 again, indicating that the subsequent concepts are independent.
– **zone**—resembles the center of a frame, bounded by **zone** $\pm\ \epsilon$, and represents the "zone of proximal development". This is a cumulative variable, to which the **learning_rate** (see below) is added after each concept. Note that the size of the frame stays the same throughout the development of the learner; as the value of **zone** increases when the learner improves, the entire frame shifts accordingly.
– **learning_rate**—is calculated as the average change in **improvement_score**, per **tick** (one time unit in the simulation). This variable is used to indicate the overall development of the learner (added to **zone** after each concept), because students take different time spans to learn a concept, as can be seen in the simulation from the way they progress.
– **motivation**—attempts to capture in an abstract way whether a learner is motivated to do well or not, i.e., if a student has the ability to learn a concept, does she actually acquire it? The value of **motivation** changes as the simulation runs and depends on whether the **difficulty** of the current concept lies within the learner's **zone** and whether or not the learner passed the quiz at the end of the previous concept presentation. If the learner "fails", she becomes motivated to do better next time if she failed by a little (**motivation** increases), but demoralized (**motivation** decreases) if she failed by a lot. In the case of cooperative learning, motivation is also influenced by the motivation of the teammates. In case of competitive learning, a learner's motivation increases if both the learner and her opponent have a competitiveness factor above a certain threshold.
– **emotion**—attempts to capture in an abstract way whether a learner is paying attention to the lesson and able to absorb all the input given during the initial presentation phase, i.e., if a learner is unhappy or depressed, she may not listen to everything her teacher says. In our simulation, the value of **emotion** changes over time and depends on how well the student performs on the quiz after progressing through a concept. For cooperative learning, **emotion** also depends on teammates' **emotion** and the rank of the team after the learners all complete the quiz. In the pedagogical literature, researchers often remark on the fact that in a competitive setting, students tend to prefer that others do not get benefits if they themselves do not receive any [7,8]. This tendency led us to implement an increase in emotion when learners compete; if learners are close together in **zone**, they form a threat to each other and competing gives them a means to try and get ahead of each other.

- **target**—is the individual target for each concept. It is a goal only for the learners in the individual the cooperative goal structures (as explained in [13]); in the competitive goal structure everyone strives for the same goal.
- **likeliness_to_help**—is the help others can give to a learner in a cooperative context and depends on how likely they are to help their peers. The help provided to other learners is calculated as the product of **likeliness_to_help** and **improvement** and represents the amount that is subtracted from the helper's **improvement**, as that learner "stays behind" to help a peer. However, the lost **improvement** is invested in **understanding**. An important factor to note is that the help provided by one learner and the help received by another learner are not necessarily the same. The amount of help given by a high ability learner to a peer depends on that learner's improvement and the **likeliness_to_help** of the learner. The receiver of the help is also responsible for the cooperation: the effort invested in the learner by the other is received according to the receiving learner's motivation; if the receiving learner is unmotivated, the help is not fully effective. Another factor influencing the learner's cooperation is whether the help provided falls within the zone of proximal development of the learner receiving the help. If the help does not fall within the **zone** of the learner helped, only a fraction of the help provided reaches the learner.
- **competitiveness**—is similar to **likeliness_to_help**, but it applies to interactions in a competitive goal structure. Very competitive learners might become motivated because of this competitiveness, in which case competitiveness has a positive influence on the learner. When a competitive and a non-competitive learner compete, this might have negative influences on the motivation of the non-competitive learner.

3.2 Environment

When implementing each goal structure in a human classroom, the setting of instructional targets for individuals or groups of learners and the design of the evaluation phase are all tasks that belong to the teacher. In our simulation, the role of the teacher is part of the environment: concept difficulties are set randomly, individual and group targets are set according to the previous outcomes of the learners, and evaluation is done according to individuals' targets and progress. The teacher is represented implicitly in the simulation as an agent bearing a unilateral dependence relation with her students; learners' behaviours depend on their teacher, but the teacher's behaviour does not change in response to the learners. This is an example of the *lecture model* of teacher behaviour described in [12].

The performative structure [5] (i.e., a description of the sequence(s) of activity in the model) of the classroom environment is divided into three phases: **initialization**, **learning** and **evaluation**. This three-phase "concept cycle" executes for each concept in the simulated curriculum. Following the STAD learning method, in the **initialization** phase, teachers present an introduction to a new concept and defines individual targets. In the simulation, this translates into setting values for each agent such as **base_score** and **zone**.

Then, agents enter the **learning** phase where they progress through the concept; this is the phase in which students are given the chance to acquire new knowledge—whether they actually progress or not (and how much) depends on their **motivation**,

emotion, **ability**, **zone** and the **difficulty** of the concept. In a competitive goal structure, this is the phase in which learners interact by competing. In a cooperative goal structure, the learners interact by helping each other to progress and by influencing each other's motivation and emotion. The learning phase is the longest phase in the concept cycle.

In the **evaluation** phase, students' **progress** is measured and combined with **understanding**, which together add up to a learner's **improvement_score**. This value is compared to the learner's target for the concept. The **improvement_score** forms the learner's new **base_score** in the initialization phase for the next concept. If the improvement score is equal to or higher than the target, the learner has passed the quiz. This results in an increase in **motivation** and **emotion** (the learner becomes "happy"), which will have a positive influence on the performance of the learner during the next learning phase. With the cooperative goal structure, the evaluation phase is used for calculating the ranks of the participating groups: in a situation in which five groups compete, the two groups that contain the learners who scored best on their individual quizzes are rewarded, having a positive influence on their motivation and emotion, and the lower scoring groups become disappointed, resulting in a negative effect on their motivation and emotion.

Two variables that drive the learning process are the **difficulty** of each concept to learn (described earlier) and the number of **ticks** spent on each concept. In the simulation, learners have a certain amount of time to master each concept, measured in **tick**s; if time runs out while learners are still working, they have to stop and move on to the evaluation phase, after which they start a new concept.

3.3 Simulator

The group learning model was simulated using NetLogo[1] [22], depicted in figure 3 [14,15]. Programming in NetLogo is inherently agent-based, and its robust and easy-to-use graphical user interface makes it an ideal environment for prototyping and running relatively small scale experiments.

The change in **zone** was monitored for each kind of learner, within each kind of goal structure, group size and composition. The average change in **zone** depends on an agent's **learning_rate**. In the evaluation phase, at the end of a concept, the learner's **zone** is incremented by the **learning_rate**, which incorporates **improvement_score**, which, in turn, encompasses **understanding**, **motivation**, **emotion** and the value of **zone** after the previous concept to be learned. In this way, all variables that are mentioned in the pedagogical literature influence the learning behavior of each agent. The differences in **ability** are implemented by using a different mean and standard deviation for a Gaussian curve describing possible improvement for each level of **ability**.

The algorithm implemented for simulating learning centers around the variable **improvement**, which is modeled as a curve (following figure 1b), using a normal distribution, with a different mean and standard deviation for learners with high or low ability, indicating the general increase in knowledge throughout the learning of a new concept.

[1] `http://ccl.northwestern.edu/netlogo/`

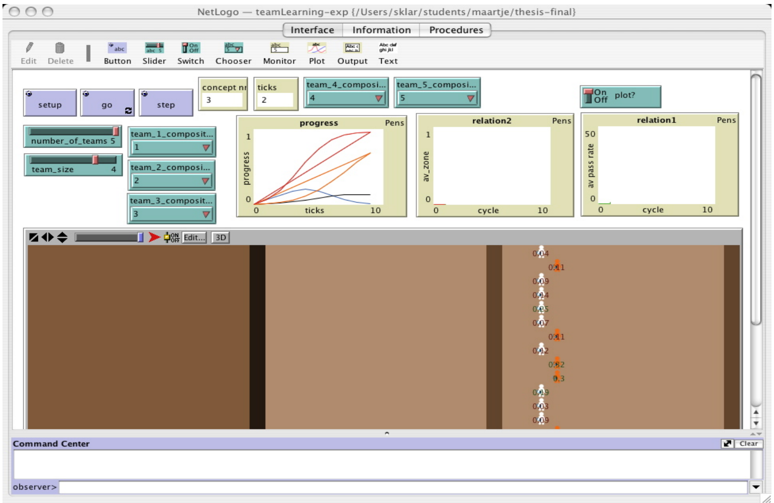


Fig. 3. Screenshot of the simulator. In the bottom portion of the screen, rows of simulated learners are shown; they begin on the left edge of the window and move to the right as they progress through each concept (there are three concepts per screen width, displayed in a horizontal scrolling window). Each dark vertical stripe indicates the evaluation phase of the previous concept and the initialization phase of the next concept. Between the dark stripes is a learning phase, shaded according to the **difficulty** of that concept: the darker the color of the concept, the harder it is. All agents start the learning phase at the same time. Interactions between agents in the cooperative and competitive goal structures are indicated by drawing lines between collaborating agents or competing agents. The interface, at the top of the screen, can be run interactively and allows the user to modify parameter settings before each run: the number of concepts to be learned and the goal structure. In the case of a cooperative goal structure, the user can choose the number of groups participating, the group size and composition for each group, and whether or not team rewards are present.

Cognitive development is measured as the change in **zone**, which is calculated by adding the learning rate to the present value of **zone**. The value of **learning_rate** is calculated as **improvement** per **tick**, and **improvement** is determined mathematically as:

$$f(\mathbf{tick}, \mu, \sigma) = \frac{1}{\sigma \cdot \sqrt{2} \cdot \pi} \, exp \left(- \frac{(\mathbf{tick} - \mu)^2}{2 \cdot \sigma^2} \right)$$

where μ and σ are the mean and standard deviation, respectively, of the improvement curve.

As suggested by the literature, improvement depends on the trilogy of mind: cognition, motivation and emotion. Cognition is represented by **zone**; the rest of the trilogy is represented directly as **motivation** and **emotion**. Another variable that influences **improvement** is concept **difficulty**. These factors are combined and used to modulate **improvement**:

$$\mathbf{improvement} \cdot = (\mathbf{motivation} \cdot \mathbf{emotion} \cdot \mathbf{zone} / \mathbf{difficulty})$$

This indicates that if **motivation**, **emotion** and **zone** are optimal, then **improvement** is maximized. For the individual goal structure, no other variables contribute to **improvement**. For cooperative and competitive goal structures, **improvement** is also influenced by help (given and received) and competition, respectively, through the variables **likeliness_to_help**, **understanding** and **competitiveness**.

4 Experiments and Results

We conducted a series of experiments designed to monitor the development (i.e., change in **zone**) of individuals within each of the three goal structures. The behaviour of the simulated learners in the individual goal structure was used as a reference for learner behaviour in the cooperative and competitive goal structures. For the cooperative goal structure, we experimented with different settings of the following parameters: group size (number of learners in each group), group composition (homogeneous and heterogeneous with different mixes of high and low ability students), and the influence of team rewards on the learning behaviour of high and low ability learners. The experiments involved 10 runs of 99 concepts each, for each goal structure. Table 1 contains the change in **zone** for both high and low ability learners in all group compositions, averaged over all runs. Values within the sections of the table can be compared, but note that it is not meaningful to contrast the change in **zone** between high and low ability learners; by definition, high ability learners will progress more quickly due to the different implementation of their improvement.

4.1 Goal Structures

We compare the results for the three goal structures simulated.

Individual and Competitive goal structures produce similar results. When we compare the development of individual learners in a competitive goal structure with the development in an individual goal structure, in table 1a, the values lie too close together to point out significant differences between the behaviors. The only difference that can be pointed out is that the standard deviations of the learners in a competitive goal structure are smaller than in an individual goel structure, which might indicate that competitiveness creates more coherence among the learners. But generally, for both high and low ability simulated learners, it can be said that individual and competitive goal structures give rise to similar learning behavior.

Cooperative goal structures benefit high ability learners. As can be seen from table 1b, all values of the development of high ability learners are higher than the value for individual learning. We can therefore say that a cooperative learning environment tends to be beneficial for the development for high ability learners.

Cooperative goal structures only benefit low ability learners some of the time. On the other hand, when comparing the results of low ability learners in an individual versus a cooperative environment we can see that learners in some group compositions do not seem to benefit from working in groups. Some of the team compositions result in the learners performing worse than in an individual goal structure. It cannot be said that the cooperative environment would therefore not be beneficial for low ability learners; it

Table 1. Experimental results: goal structures, group composition and size. Mean change in zone and standard deviation are shown. Different group compositions are illustrated by combinations of (H) and low (L) ability learners. Groups sizes (2, 3 and 4) are represented implicitly in the number of learners denoted in each group composition.

(a) **high** and **low** ability learners, for both **individual** and **competitive** goal structures:

	individual goal structure	competitive goal structure
H	**0.2670** (0.0253)	0.2620 (0.0273)
L	**0.1517** (0.0233)	0.1542 (0.0185)

(b) **high** ability learners, **cooperative** goal structure:

	without team rewards	with team rewards
HH	0.3738 (0.1509)	0.3201 (0.0576)
HHH	**0.4162** (0.2085)	0.2955 (0.0958)
HHHH	0.2972 (0.0360)	0.3627 (0.0891)
HL	0.3062 (0.1011)	**0.4959** (0.3448)
HHL	0.2916 (0.0744)	0.3621 (0.2213)
HLL	0.3514 (0.1015)	0.3210 (0.1084)
HHHL	0.3082 (0.0535)	0.2899 (0.0632)
HHLL	0.3295 (0.0703)	0.2610 (0.0763)
HLLL	0.2807 (0.1153)	0.2792 (0.1790)

(c) **low** ability learners, **cooperative** goal structure:

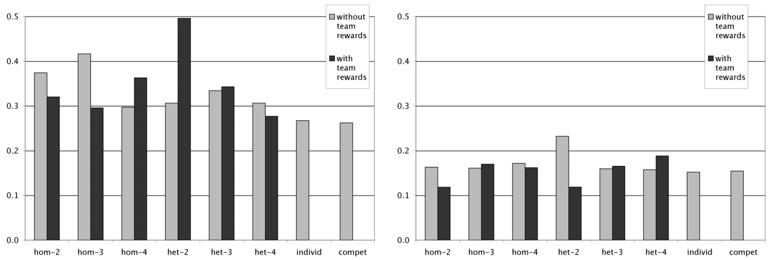
	without team rewards	with team rewards
LL	0.1631 (0.0713)	0.1182 (0.0646)
LLL	0.1611 (0.0526)	0.1698 (0.0509)
LLLL	0.1714 (0.0455)	0.1619 (0.0429)
HL	**0.2321** (0.1279)	0.1185 (0.0986)
HHL	0.1679 (0.1060)	0.1806 (0.0933)
HLL	0.1546 (0.0695)	0.1568 (0.0653)
HHHL	0.1819 (0.1236)	**0.2247** (0.0956)
HHLL	0.1441 (0.0840)	0.1478 (0.0708)
HLLL	0.1464 (0.0574)	0.1921 (0.0660)

does however become clear that other factors might influence the success or failure of the cooperative goal structure for low ability learners.

4.2 Group Composition and Size

We compare the results for group composition within cooperative goal structure.

High ability learners benefit most from working in small groups of homogeneous composition, while low ability learners benefit most from heterogeneous groups. The reason for the latter result could be that the low ability learners benefit from cooperation with high ability learners, since the high ability learners can help them progress. One result that illustrates this very well are the results for learners of both abilities in a group with a composition with a small number of low ability learners and a large number of high ability learners (like **HHHL**). This composition is most fruitful for low ability learners; working together with only high ability learners will provide the low ability learner with a lot of help. The high ability learners benefit in turn from cooperating with low ability learners because they gain understanding. This trade-off can be seen in table 1b, for the same team composition. Where the low ability learner scored relatively very well, the high ability learner does not develop much more than in an individual goal structure. The results do show, however, that both high ability learners and low ability learners can thrive in heterogeneous groups, whereas homogeneous



(a) high ability learners (b) low ability learners

Fig. 4. Experimental results: team rewards. Results are averaged over each group size (2, 3 or 4), for each type of group (homogeneous vs heterogeneous).

group compositions only pay off for high ability learners. A possible explanation for this could be that high ability learners only benefit from helping low ability learners in certain circumstances; low ability learners, on the other hand, are always helped by high ability learners.

4.3 Team Rewards

We conducted an experiment examining the influence of team rewards in the cooperative goal structure.

Team rewards do not always have the intended effect of improved development; very often, both high and low ability learners perform worse than without team rewards. As can be seen from figure 4(a), team rewards work especially well for high ability learners in large homogeneous groups and small heterogeneous groups. This can be explained by the increased chances of high ability learners to rank highly in a learning environment where group performance is compared. Low ability learners, especially in small groups, cannot "outrank" the groups with more high ability learners and will therefore lose motivation. An interesting result shown in figure 4(b) is therefore the development of low ability learners in a homogeneous group of three; they seem to benefit from team rewards, while many other groups would seem to be better cognitively. The influence of team rewards on the simulated learners is closely related to group size.

By introducing team rewards, the pedagogical literature predicts that group members are more responsible for their group members' progress. Team rewards can therefore be an important motivator for group members, and can be compared to "team spirit" amongst members of a sports team [13]. Based on this motivational aspect, our prediction was that team rewards would have a positive effect on the learning behaviour of the simulated learners in the cooperative goal structure. In the simulation, team rewards have an influence on the motivation and emotion of the group members: when a cooperative group improves a lot compared to the other groups, the motivation of all its members will increase; the motivation will decrease if a group is ranked last. As a result of the increased motivation, the emotion will also increase: the learners become "happier".

5 Discussion and Summary

Our experimental results show differences in learning, measured by a model of each student's zone of proximal development. Additional experiments and details of this work can be found in [14]. Summarizing the results presented here, we can say that group composition, team rewards and team size have clear influences on the development of simulated learners in a cooperative environment. Different variable settings may help to overcome the apparent negative influences of this goal structure for low ability learners. This can be compared to a real-life situation, in which a teacher implements a goal structure in such a way that it enables her students to develop optimally.

The results also show that there appears to be no single optimal group size for either high or low ability learners; however group size is a very powerful factor in combination with other variables, like group composition or the presence of team rewards. The hypothesis that a larger group would give rise to more development is proven to hold only for homogeneous groups with team rewards, or for high ability learners in heterogeneous groups without team rewards. One observation that can be made from watching the visualization of the learners in the simulation is that team rewards have a positive effect on group *coherence*, although this was not measured formally. The learners seem to progress more "together" in a situation with team rewards (than without). This is related to the helping principle, which enables a high ability learner to gain understanding by helping a low ability learner.

We have presented the background for and design of a *group learning model* and simulation system in which theoretical human learners are modeled as artificial agents whose behaviours are influenced by a wide range of individual and environmental parameters. Using this simulator, we have investigated three different goal structures in groups of simulated learners, characterized by features such as size, homogeneity and reward structures. A number of the parameters defined in the simulation have significant effects on learning outcomes, corresponding to trends observed in empirical studies of human learners described in the pedagogical literature. Even though computational modeling will always be an abstraction of the behaviour of human subjects, agent-based simulation can be a powerful tool for examining aspects that are difficult to study *in situ* and can provide better understanding of individual and environmental characteristics that influence the progress of human learners.

Acknowledgments

This work was partially supported by NSF #ITR-02-19347, NSF #ITR-05-52294 and by PSC-CUNY #68525-00-37.

References

1. Anderson, J.R.: Acquisition of cognitive skill. Psychological Review 89, 369–406 (1982)
2. Azmitia, M.: Peer interaction and problem solving: When are two heads better than one? Child Development 59, 87–96 (1988)
3. Brown, A.H.: Simulated Classrooms and Artificial Students: The Potential Effects of New Technologies on Teacher Education. Journal of Research on Technology Education (formerly Journal of Research on Computing in Education) 32(2), 307–318 (winter, 1999)

4. Daum, S.C.: Using simulated students to improve residents' teaching. Acad Med. 72(5) (May 1997)
5. Esteva, M., Rodriguez, J.A., Sierra, C., Garcia, P., Arcos, J.L.: On the formal specifications of electronic institutions. In: Sierra, C., Dignum, F.P.M. (eds.) AgentLink 2000. LNCS (LNAI), vol. 1991, pp. 126–147. Springer, Heidelberg (2001)
6. Fitts, P.M.: Perceptual-motor skill learning. Categories of human learning (1964)
7. Johnson, D.W., Johnson, R.T.: Learning Together and Alone: Cooperation, Competition and Individualization. Prentice-Hall, New Jersey (1975)
8. Kohn, A.: No Contest: The Case Against Competition. Houghton Mifflin, Boston (1986)
9. Lazarus, R.S.: Cognition and Motivation in Emotion. American Psychologist 45(4), 352–367 (1991)
10. Mugny, G., Doise, W.: Socio-cognitive conflict and structure of individual and collective performances. European Journal of Psychology 8, 181–192 (1978)
11. Ohlsson, S.: Learning from performance errors. Psychological Review 103, 241–262 (1996)
12. Sklar, E., Davies, M.: Multiagent simulation of learning environments. In: Fourth Int'l Conf on Autonomous Agents and Multi Agent Systems (AAMAS 2005) (2005)
13. Slavin, R.: Cooperative Learning: Theory, Research and Practice. Alley & Bacon, Boston (1990)
14. Spoelstra, M.: Simulating the Effects of Goal Structures in Human Learning Environments. Master's thesis, Artificial Intelligence Section, Department of Computer Science, Faculty of Sciences, Vrije Universiteit Amsterdam, The Netherlands, Advisors: Martijn Schut and Elizabeth Sklar (2006)
15. Spoelstra, M., Sklar, E.: Using simulation to model and understand group learning. In: Agent-Based Systems for Human Learning and Entertainment (ABSHLE) Workshop at Autonomous Agents and MultiAgent Systems (AAMAS 2007) (2007)
16. Taatgen, N.A.: Explicit Learning in ACT-R. In: Mind modelling: a cognitive science approach to reasoning, learning and discovery, ch. 3.1, pp. 233–252. Pabst, Lengerich (1999)
17. Tang, Y., Parsons, S., Sklar, E.: Modeling human education data: From equation-based modeling to agent-based modeling. In: Multi-agent Based Simulation (MABS) Workshop at Autonomous Agents and MultiAgent Systems (AAMAS 2006) (2006)
18. VanLehn, K., Ohlsson, S., Nason, R.: Applications of simulated students: An exploration. Journal of Artificial Intelligence in Education 5(2), 135–175 (1996)
19. Virvou, M., Manos, K.: A Simulated Student-Player in Support of the Authoring Process in a Knowledge-Based Authoring Tool for Educational Games. In: Third IEEE Int'l Conf on Advanced Learning Technologies (ICALT 2003) (2003)
20. Vizcaino, A., du Boulay, B.: Using a simulated student to repair difficulties in collaborative learning. In: Int'l Conf on Computers in Education (2002)
21. Vygotsky, L.S.: Mind in society: The development of higher psychological processes. Harvard University Press, Cambridge (1978)
22. Wilensky, U.: Modeling nature's emergent patterns with multi-agent languages. In: Proceedings of EuroLogo 2002, Linz, Austria (2002)

An Agent-Based Model That Relates Investment in Education to Economic Prosperity

Yuqing Tang[1], Simon Parsons[1,2], and Elizabeth Sklar[1,2]

[1] Department of Computer Science
Graduate Center, City University of New York
365, 5th Avenue, New York, NY 10016, USA
ytang@gc.cuny.edu
[2] Department of Computer & Information Science
Brooklyn College, City University of New York
2900 Bedford Avenue, Brooklyn, NY 11210, USA
{parsons,sklar}@sci.brooklyn.cuny.edu

Abstract. This paper describes some experiments with an agent-based model designed to capture the relationship between the investment that a society makes in education in one generation, and the outcome in terms of the health of the society's economy in ensuing generations. The model we use is a multiagent simulation derived from an equation-based model in the economics literature. The equation-based model is used to establish parameterized sets of agent behaviors and environmental characteristics. Agents are divided into three chronological categories: students, adults and pensioners; and each responds to and affects the environment in different ways, in terms of both human and physical capital. We explore the effects of different parameter settings on the education investment of a society and the resulting economic growth.

1 Introduction

We are working towards creating tools that can be used in determining the effects of particular choices in education policy. Our aim is to be able to use such tools to inform the debate about initiatives like the US "No Child Left Behind" Act [10], and illuminate the controversies that such initiatives have created. To this end we have been extracting predictive models from sets of data related to human education, and implementing predictive models [12,14].

Typically, data on education is collected in one of two ways. It is either very large, aggregrate data sets over entire populations (like whole cities, school districts, states or provinces) or it is very small, localized experimental samples. In both cases, the data is usually analyzed using standard statistical methods. Often, the most highly publicized statistics are the simplest, for example the mean and standard deviation of standardized test scores in mathematics and language arts. These values are frequently the ones used to make policy decisions. More occasionally, the data is analyzed in such as way as to examine how multiple factors influence each other, such as the relationship between student-teacher ratios and test scores, dollars per student and test scores, or class size and test scores.

L. Antunes, M. Paolucci, and E. Norling (Eds.): MABS 2007, LNAI 5003, pp. 84–95, 2008.

Where this data is extracted into models, it is formulated in traditonal terms, as sets of interelated differential equations. In contrast to such models, commonly called *equation-based models* (EBMs), we are building agent-based models (ABMs) which are constructed in terms of a set of autonomous interacting entities. Such models have been successfully used to generate useful predictions about the behavior of populations made up of individuals [11], especially where such individuals make their own decisions about how to act [15].

A particular strength of agent-based models [3] is that they allow one to identify *emergent phenomena*. Emergent phenomena result from the actions and interactions of individual agents, but are not directly controlled by the individuals. Indeed, they have an existence that is partly independent of those individuals—the classic example of an emergent phenomenon is a traffic jam, which, while caused by the actions of drivers moving in one direction, may even travel in the opposite direction.

Emergent phenomena simply do not show up in EBMs, but knowing about them can be crucial. Bonabeau [3] gives a nice example of emergent behavior with the agent-based model used by NASDAQ to identify the effects of changing some of the market rules. This model showed that reducing "tick" size (the minimum possible change in the price of an offer to trade) would lead to a larger bid-ask spread (the difference between offers to buy and offers to sell), a result that was completely counter-intuitive. Cases of emergent behavior also appear in [2,5,7], and in our prior work [14]. Such findings are also echoed in ecology, as in [6,13] for example, and agent-based models have been used quite extensively in ecology where they go by the name "individual-based models". Since one can not only examine the behavior of individuals in an agent-based model, but can also look at the statistics across a population, agent-based modeling can help bridge the gap between macro and micro data sets, and thus provides the perfect tool for our work.

In this paper, we describe results of our work on one specific agent-based model, a model developed from an equation-based model published in the economics literature [9]. The model relates the effectiveness of education to economic productivity, and money spent on education to the effectiveness of education. It therefore provides a means to tie models like those we have developed in our previous work [12,14] — models which concentrate on the education obtained by individuals — into the wider economic picture.

The remainder of this paper is organized as follows. Section 2 describes the model that we have implemented, both the equation-based original and the agent-based model we derived from it. Section 3 then describes a set of experiments that we performed using the model, Section 4 gives the results, and Section 5 analyzes them. Section 6 then concludes.

2 The Model

The model that we have implemented is taken from the work of Laitner [9]. This section describes the main features of this model, and the main aspects that needed to be adapted to create an agent-based model.

2.1 Laitner's Equation-Based Model

The setting for the model is a simple economy that has two sectors. Each of these sectors produces one good. The goods are:

- Units of education that are used to train individuals in the population; and
- Units of a numeraire good.

"Numeraire" is defined as "a basic standard by which values are measured, as gold in the monetary system" [4]. In [9] it is a good that is produced (see below) and then traded for things that individuals consume. Presumably these things are produced by a different economy that trades with the one we are studying.

The individuals that inhabit this economy live for three time periods, periods in which they are students, adults and pensioners. Consider an individual who is a student during period $t-1$. She spends this period living with her parent[1] and studying. Parents provide the numeraire good that supports the child during this period, but the child selects her own units of schooling, borrowing the money to finance this. In the period t, the now adult individual forms her own household, rears a child (paying for the child's consumption but not the child's schooling), and chooses how much of the numeraire good, c_t^l, that she earns during this period will be consumed by the household during the same period. In the period $t+1$, the individual is a pensioner, and chooses her consumption for that last period, c_{t+1}^2, from the numeraire good than she has saved. An individual's utility, u, is:

$$u = (1-\alpha)\ln(c_t^l) + \alpha \ln(c_{t+1}^2) \tag{1}$$

where $\alpha \in (0,1)$, and all individuals have the same α.

While working, period t in our example, our individual earns w_t per unit of human capital she possesses. Her human capital depends on her innate ability and the amount of schooling she chose as a child. An individual with ability a who purchased e_{t-1} units of education will have human capital:

$$h = a\left(\frac{(e_{t-1})^\gamma}{\gamma}\right) \tag{2}$$

where $\gamma \in (0,1)$ and all individuals have the same γ. The relationship between e and a allows education to raise human capital, but in a way that is subject to the law of diminishing returns. Innate ability is randomly assigned at the birth of individuals, with values being taken from a stationary distribution given in [9].

The model does not include inheritance and bequests, so every individual has to pay for her consumption and education out of what she earns during the period t during which she works. If r_t is the interest rate on savings made during period $t-1$ and held until period t, every individual is constrained by

$$c_t^1 + \left(\frac{c_{t+1}^2}{r_{t+1}}\right) + p_{t-1} r_t e_{t-1} \leq w_t a \left(\frac{(e_{t-1})^\gamma}{\gamma}\right) \tag{3}$$

[1] In this model, every individual has one child and raises that child alone — in terms of real world accuracy, this is equivalent to the model in [8], implemented as an agent-based model in [14], which assumes every family is a perfect nuclear family of a mother and a father and a son and a daughter.

where p_{t-1} is the cost of a unit of education in period $t-1$. In other words, the total amount that an individual consumes, including their education, suitably discounted over time, must be less that their earnings. Any earnings that are not consumed in an individual's lifetime are lost.

In every period, m individuals are born, and so there are $3m$ individuals in total in every period in time — m of these are being educated, m are working, and m are retired.

Considering the sector of the economy that produces the numeraire good, the model assumes constant returns to scale, so that the output per individual in a given generation is:

$$n = (K_t^n)^{\beta^n} (\lambda_t^n H_t^n)^{1-\beta^n} \tag{4}$$

where $\beta^n \in (0,1)$, K_t^n is the physical capital the sector has per working individual at time t, and H_t^n is the average human capital per individual in the generation that is currently working. $\lambda^n > 1$ models the tendency of technological change to increase the effect of human capital in the sector of the economy that generates the numeraire good. The other sector of the economy produces education. Here we have:

$$e = (K_t^e)^{\beta^e} (\lambda_t^e H_t^e)^{1-\beta^e} \tag{5}$$

and the model allows for λ^e and β^e to be different from, or the same as, λ^n and β^n, their counterparts in the numeraire sector of the economy.

For both the numeraire and education sectors, the assumption is that all physical capital is consumed in a single period, so the numeraire good produced in period t has to equal all consumption plus the physical capital used at time $t+1$.

Taken together, these equations and the values of the constants provided in [9] provide a rather standard economic model.

2.2 The Agent-Based Model

As described in [3,11,14], it is possible to generate agent-based models from equation-based models, by equipping individuals with decision processes that make decisions in line with the equations.

For this model, the decisions faced by an individual are:

1. How much education to purchase[2].
2. What proportion of wages to save.

The first of these is, in essence, an investment decision. Given (2), for a given level of ability, the more education that an individual purchases, the greater their productivity. All other things being equal — and in particular, the performance of the other individuals in the economy — this greater productivity will turn into greater production of numeraire goods, and, once the cost of education is paid off, greater utility for the individual. Because (2) captures diminishing returns, an individual who spends too much

[2] Since every unit of education that an individual undergoes must be paid for from its later wages, it seems appropriate to think of choices about education as a purchase.

on education will not recoup their investment. The second decision is the same decision faced by anyone who has considered their own retirement — how much of one's lifetime earnings, minus cost of living and any debts accumulated, should be saved for retirement rather than spent while one is working.

In addition to these decisions, there are decisions faced by the economy as a whole. In the current version of the model, these decisions are taken by a single agent, representing the government. These are:

1. What proportion of numeraire production should be turned into physical investment.
2. What proportion of physical investment should be put into the numeraire sector rather than the education sector.
3. How to allocate workforce between the education and numeraire sectors.

The first of these decisions can be considered as the effect of taxation — some money is taken out of the income pool and is distributed by the government.

The second decision determines how much of this taxation is invested in education rather than into the production of numeraire goods — this provides the K_t^n and K_t^e in (4) and (5). Since this investment amortises over a single time-step — which is reasonable given that each time step represents a third of a lifetime, or approximately 25 years given average life expectancy in the United States — it needs to be renewed at every timestep. With this second decision under the control of some central authority, the model looks like a command economy. A more capitalist model, in which firms compete for investment from individuals and use that to provide physical investment for the numeraire sector, while leaving the government to deal with education investment, is a topic for future work.

Allowing the government to directly determine what proportion of workers to place in each sector also looks like something one would expect to find in a command economy. However, all governments exercise some control over aspects like this through their actions — in many economies the government has a large say in the organization of the education sector and can encourage people to work in the education sector by, for example, spending money raised through taxation to increase the wages of education workers.

We have implemented a number of ways that each of these decisions can be taken, and these are explored in the next section, which also gives a description of the experiments we have run.

3 Experiments

The experiments that we will describe here were intended to explore the properties of the model described in the previous section, examining whether our agent-based version could run successfully. That is, whether the decision-making functions with which we had equipped the model were sufficient to create a healthy economy and to approximate the behaviors of real economies. Before we explain how the experiments were run, we need to say what the decision functions are.

In our current implementation, individuals only have one decision to make because the proportion of wages that are saved is kept fixed. The decision they have to take, then, is how much education e_d to purchase, and the implementation provides two ways for individuals to do this:

1. Randomly: e_d is drawn from a normal distribution with mean 13 and standard deviation 2.1. This is the distribution used in [8,14], and was originally taken from recent US census data.
2. Maximum utility: e_d is chosen by:

$$e_d = argmax_e \left(w_t \cdot a \cdot \frac{e^\gamma}{\gamma} - p_{t-1} \cdot r_t \cdot e \right) \tag{6}$$

As described above, the government has to decide:

1. $pr_{n,k}$: the proportion of numeraire production to be used as physical investment in the next period;
2. $pr_{k,e}$: the proportion of physical investment to be put into the education sector; and
3. $pr_{h,e}$: the proportion of the working population to move into the education sector.

The implementation provides several ways that each of these decisions can be made. There are three ways to decide on $pr_{n,k}$:

1. Constant proportion: $pr_{n,k}$ is set to 0.4.
2. Self adjustment: if numeraire production exceeds demand then $pr_{n,k}$ is decreased by 5%, otherwise $pr_{n,k}$ is increased by 1%.
3. Z policy: The policy that [9] uses for this decision.

Laitner's Z policy first computes an intermediate variable Z_t which describes a relationship between physical investment K_t and capital value H_t, then the policy computes an estimate of H_{t+1} from the education students have received at time t and their ability, and then computes the physical investment K_{t+1} from the estimate of Z'_{t+1}. These computations are the following:

$$Z_t = \frac{K_t}{\lambda^t \cdot H_t} \tag{7}$$

$$Z'_{t+1} = \theta \cdot (Z_t)^\xi \tag{8}$$

$$\theta = \left(\frac{\alpha \cdot (1-\gamma)}{\lambda^{1/(1-\gamma)}} \cdot \frac{1-\beta}{1+\gamma \cdot [(1-\beta)/\beta]} \right)^{1-\gamma} \tag{9}$$

$$\xi = (\beta - 1) \cdot (1-\gamma) + 1 \tag{10}$$

$$K'_{t+1} = Z'_{t+1} \cdot \lambda^{t+1} \cdot H_{t+1} \tag{11}$$

$$pr_{n,k} = \frac{K_{t+1}}{n} \tag{12}$$

The implementation includes two methods for choosing $pr_{k,e}$:

1. Constant proportion: $pr_{k,e}$ is set to 0.1.
2. Self adjustment: if education production exceeds demand, $pr_{k,e}$ is decreased by 5%, otherwise $pr_{k,e}$ is increased by 1%.

and there are two methods implemented for choosing $pr_{h,e}$:

1. Constant proportion: $pr_{h,e}$ is set to 0.1.
2. Self adjustment: if education production exceeds demand, then $pr_{h,e}$ is decreased by 5%, otherwise $pr_{h,e}$ is increased by 1%.

We ran experiments for each combination of these decision mechanisms.

4 Results

The results of these experiments, which were run over 100 timesteps, or just over 30 generations, are given in Figures 1 and 2, which show, for each economy:

– The average utility of individuals.
– The total earnings of all individuals in the economy, along with their savings for retirement, and the unpaid debt for their education.
– The education that is produced, per individual in the economy, along with the average demand for education.
– The number of numeraire goods that are produced, per individual in the economy, along with the average demand. Demand is measured by the amount of goods and individual chooses to consume, an amount that may not be satsified if the economy does not produce enough.
– The wage rates, broken down across the numeraire and education sectors.
– The number of individuals who cannot generate enough wages during their lifetime to pay for their education and their consumption as a worker or as a retiree, broken down across the numeraire and education sectors.

By all these measures, the economy in Figure 1 (Experiment 10) is healthy. The overall utility of individuals grows over time, as do wages (which reflect production). Education production flucuates over time, but fits well with demand — note that when demand exceeds supply, then individuals only receive a proportion of the education they want, and the surplus demand is spread across the population. Numeraire production grows over time. Wages in the numeraire sector grow steadily over time, as do those in the education sector, but these latter are also affected by spikes in demand. Finally, no individuals go bankrupt.

In contrast, the economy in Figure 2 (Experiment 18) is dramatically unhealthy. Once we get past the start-up effects, which are responsible, for example, for the same modest jump in average utility in both Figure 1 and 2 (note that Figure 1 (a) and Figure 2 (a) are on rather different scales), utility enters a long slump, total earnings are static while debt mounts, demand for education consistently outstrips supply by a factor of around 3, average wages have a downward trend, and after about six generations (20 timesteps) become insufficient to support the whole population — indeed after around 15 generations (40–50 timesteps) the entire population cannot meet its needs. The only apparent bright spot is that numeraire production exceeds demand, but this is because individuals do not have enough money to consume any of the goods — at the end, production is 40 times less than that in the healthy economy.

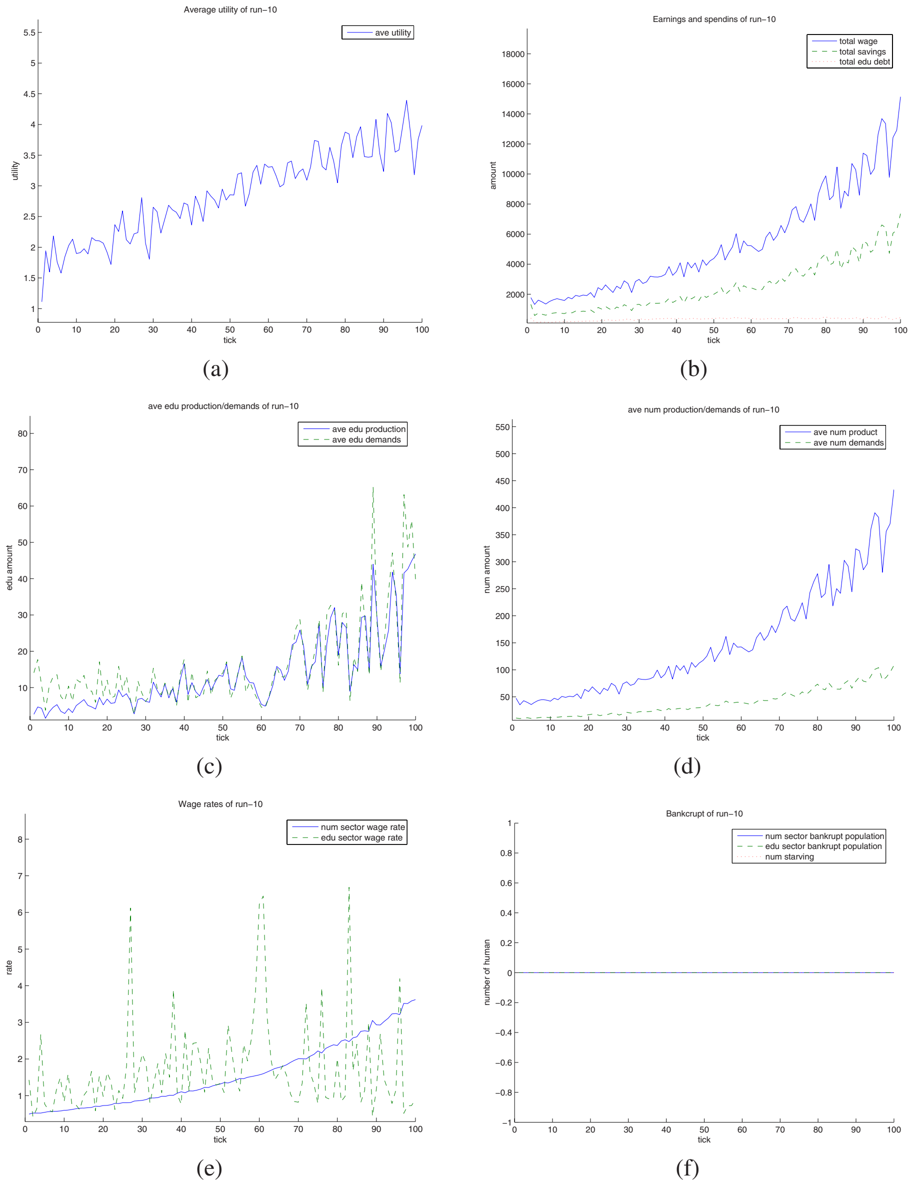


Fig. 1. Experiment 10, an example of a healthy economy under the model. (a) Average utility of individuals. (b) Earnings and savings. The solid line shows total earnings. The dashed line shows total savings. The dotted line shows debt due to education. (c) Education production per individual. The solid line shows actual production. The dashed line shows demand. (d) Numeraire production per individual. The solid line shows actual production. The dashed line shows demand. (e) Wage rates. The dashed line shows wages in the education sector. The solid line shows wages in the numeraire sector. (f) Bankruptcy. The solid line shows the number of workers in the numeraire sector who are bankrupt. The dashed line shows the corresponding number for the education sector. The dotted line shows the number of individuals who cannot afford to consume.

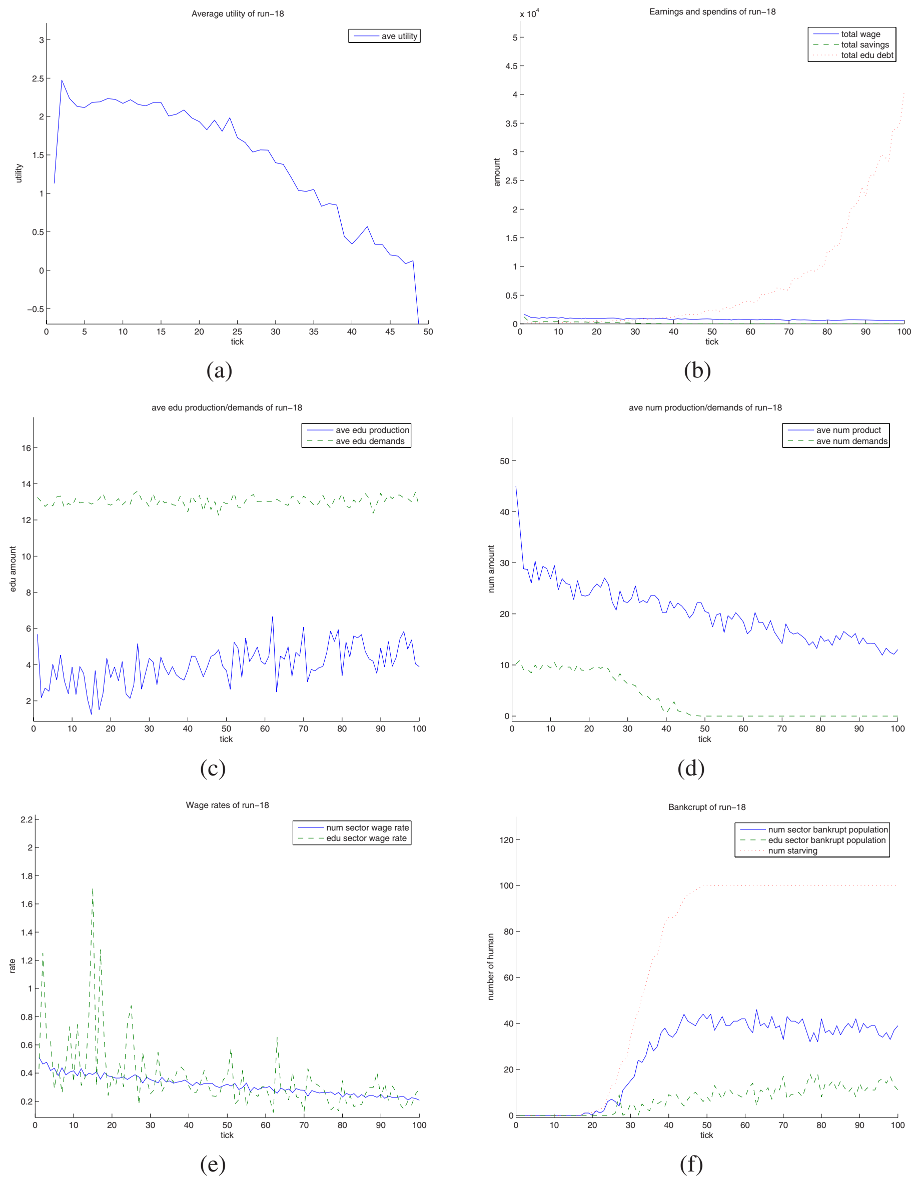


Fig. 2. Experiment 18, an example of an unhealthy economy. (a) Average utility of individuals. (b) Earnings and savings. The solid line shows total earnings. The dashed line shows total savings. The dotted line shows debt due to education. (c) Education production per individual. The solid line shows actual production. The dashed line shows demand. (d) Numeraire production per individual. The solid line shows actual production. The dashed line shows demand. (e) Wage rates. The dashed line shows wages in the education sector. The solid line shows wages in the numeraire sector. (f) Bankruptcy. The solid line shows the number of workers in the numeraire sector who are bankrupt. The dashed line shows the corresponding number for the education sector. The dotted line shows the number of individuals who cannot afford to consume.

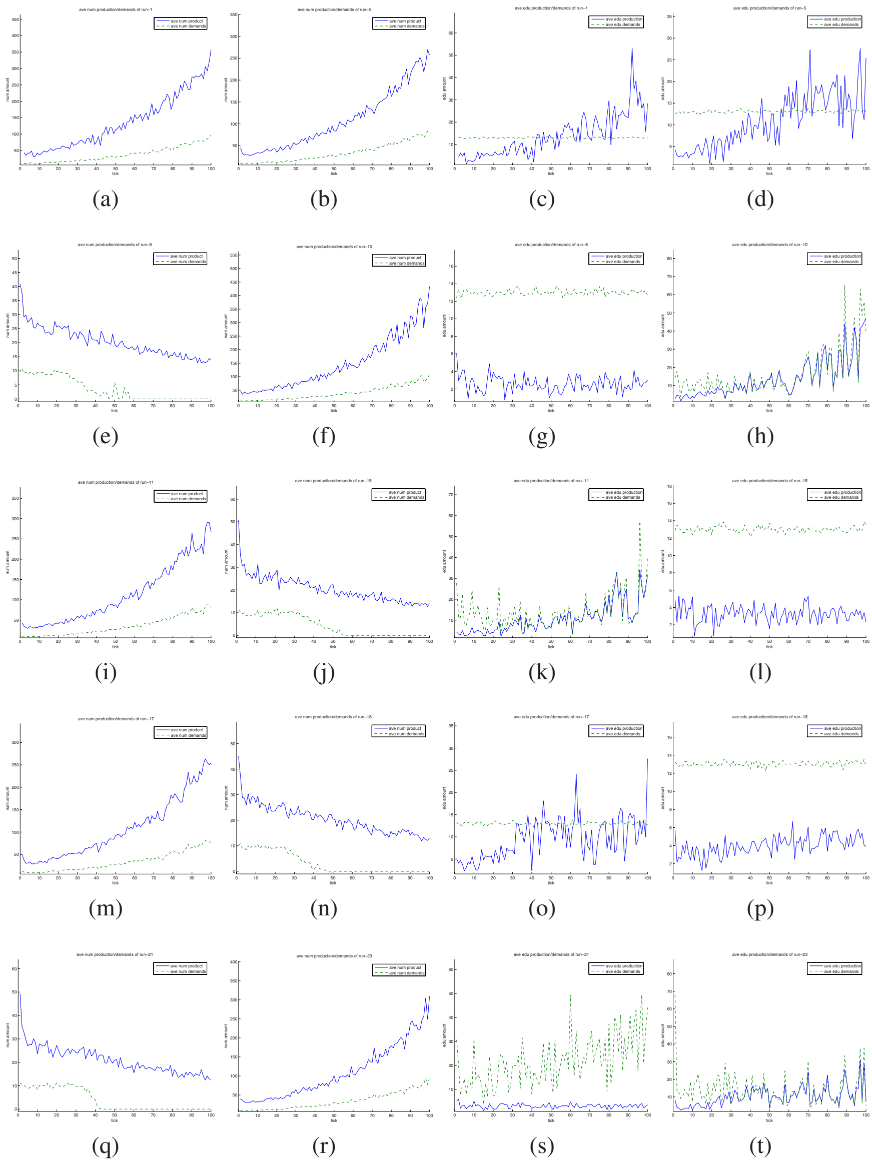


Fig. 3. The relationship between the education produced by the economy and the production of the numeraire good in selected experiments. In all graphs, demand is given by a dotted line, supply by a solid line. The left two columns give the numeraire production, the right two columns the education production. (a) and (c) are taken from the same economy, as are (b) and (d), and so on.

5 Discussion

The results in the previous section are taken from only two examples of the 24 outlined in Section 3, but they are typical. To back up this claim, Figure 3 gives the average production of numeraire good and the demand for that good (which is a useful measure of economic health) against the demand and supply of education for 10 of the models. The results are presented in pairs, so Figure 3 (a) and (c) give numeraire production and education production for one model, Figure 3 (b) and (d) for the next model, and so on.

The broad trends shown in Figures 1 and 2 are repeated in these other models — the results in Figures 1 and 2 are those in Figure 3 (f) and (h) and Figure 3 (n) and (p) respectively. All of the other runs have results that fall into the same two broad classes — not only are all healthy economies healthy in exactly the same way, but all unhealthy economies are unhealthy in the same way.

The question, of course, is "why do the failing economies fail?", and it seems to us that the reason for the failure is clear from Figure 3. All the economies that fail have a consistently unmet demand for education. Over time, if economies lack the ability to educate the workforce, productivity falls, there is no basis for capital investment, and so demand for education remains unmet.

Of course, this feedback effect is written into the equation-based model, so it is no great surprise that it surfaces in the agent-based model. Indeed, we would be worried if it did not. However, note that in all the economies, even the successful ones, the demand for education initially outstrips supply. It is those economies responding to this mismatch by pumping resources into education and thus growing education production, that manage to bootstrap themselves out of the initial surplus demand for education (which will, of course, limit the productivity of the education sector since future educators themselves will be less productive if their education demands are not met). Interestingly, all the economies in Figure 3 that fail are economies that use the self-adjustment mechanism to set investment. This mechanism is much more short-term than the others, cutting investment at the first suggestion that production exceeds demand. It is tempting to interpret the failure of this approach in the models depicted as a failure for short-termism in economic policy, but we need to run more experiments before we can be confident in making any judgement on this.

6 Summary

This paper has described the creation of an agent-based model of an economy from an equation-based model, and the results of some experiments intended to establish the behavior of the model under a range of conditions. These experiments have shown that the model tightly couples investment in education to production, and, through production, to the overall health of the economy.

Our next step with this model is to extend it towards the policy evaluation tool that we described in the introduction. To do this, we first envisage combining it with the model we described in [14] — an agent-based model that was developed from the equation-based model in [8]. The model in [14] will give us a mechanism that individuals use to determine the level of education that they desire (a level that is based on that of their parents), a model that, as [8] describes, is a good fit for real data. With that done, we want to couple in models like that in [12] which relate policy changes in education, like class size, to the quality of education that is provided.

Acknowledgments

This work was partially supported by NSF #ITR-02-19347 and by PSC-CUNY #68525-00-37.

References

1. Antunes, L., Takadama, K. (eds.): MABS 2006. LNCS (LNAI), vol. 4442. Springer, Heidelberg (2007)
2. Bernard, R.N.: Using adaptive agent-based simulation models to assist planners in policy development: The case of rent control. Working Paper 99-07-052, Sante Fe Institute (1999)
3. Bonabeau, E.: Agent-based modelling: Methods and techniques for simulating human systems. Proceedings of the National Academy of Science 99(3), 7280–7287 (2002)
4. Flexner, S.B. (ed.): The Random House Dictionary of the English Language. Random House, New York (1987)
5. Greenwald, A., Kephart, J.: Shopbots and pricebots. In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sweden, August 1999, pp. 506–511 (1999)
6. Grimm, V.: Ten years of individual-based modeling in ecology: what have we learned and what could we learn in the future? Ecological Modeling, 129–148 (1999)
7. Kephart, J., Greenwald, A.: Shopbot economics. Autonomous Agents and Multi-Agent Systems 5(3), 255–287 (2002)
8. Kremer, M.: How much does sorting increase inequality? The Quarterly Journal of Economics 112(1), 115–139 (1997)
9. Laitner, J.: Earnings within education groups and overall productivity growth. The Journal of Political Economy 108(4), 807–832 (2000)
10. No Child Left Behind (July 2001), http://www.whitehouse.gov/news/reports/no-child-left-behind.html
11. Van Dyke Parunak, H., Savit, R., Riolo, R.L.: Agent-based modeling vs. equation-based modeling: A case study and users' guide. In: Proceedings of the Workshop on Multiagent-based Simulation, pp. 10–25 (1998)
12. Sklar, E., Davies, M., Co, M.S.T.: imEd: Simulating Education as a MultiAgent System. In: Proceedings of the Third International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2004), pp. 998–1005 (2004)
13. Sole, R.V., Gamarra, J.G.P., Ginovart, M., Lopez, D.: Controlling chaos in ecology: From deterministic to individual-based models. Bulletin of Mathematical Biology 61, 1187–1207 (1999)
14. Tang, Y., Parsons, S., Sklar, E.: Agent-based modeling of human education data. In: Proceedings of the Fifth International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2006) (2006); An extended version of this paper appeared in [1]
15. Wooldridge, M., Jennings, N.R., Kinny, D.: The Gaia methodology for agent-oriented analysis and design. Autonomous Agents and Multi-Agent Systems 3(3), 285–312 (2000)

Trust-Based Inter-temporal Decision Making: Emergence of Altruism in a Simulated Society

Tibor Bosse, Martijn C. Schut, Jan Treur, and David Wendt

Vrije Universiteit Amsterdam, Department of Artificial Intelligence,
de Boelelaan 1081a, 1081 HV, The Netherlands
{tbosse,schut,treur,dwendt}@cs.vu.nl
http://www.cs.vu.nl/~{tbosse, schut, treur, dwendt}

Abstract. This paper contributes to the analysis of the question how altruistic behaviour can be in an agent's own interest. The question is addressed by involving a temporal dimension in which altruist behaviour at one point in time can be in the agent's interest at a future point in time, depending on the environment. The claim is that to be able to make reasonable decisions, an agent needs a cognitive system for intertemporal decision making, in relation to a model of the environment. To address this, a society of agents has been modelled, simulated and analysed. Some of the agents have a cognitive system including a model of the environment based on a dynamic model for trust in other agents. This environment model is combined with a model for intertemporal decision making. Agents with this cognitive system show more altruistic behaviour, they get a larger social network, and become in the end healthier than agents without such a cognitive system.

1 Introduction

A basic assumption in the evolutionary explanatory framework is that an organism's behaviour serves its own interests, to improve its own well-being and production of offspring. For the explanation of the development of altruistic behaviour from an evolutionary perspective, one easily encounters a paradox; see, e.g., [12], pp. 17-23. As by definition altruistic behaviour is behaviour that is against one's own interests, this paradox can be formulated as: 'an organism serves its interests by behaviour which is against its interests'. One way to solve this paradox is by widening the scope in the temporal dimension. Then the occurrences of the concept 'interest' in the paradox get different time points attached, as follows: altruistic behaviour serves the organism's interests at a future time point by the organism's behaviour which is against its interests at the present time point. So, the organism's present behaviour is seen as an investment to obtain future revenues for the organism itself. As long as the future revenues are at least as important for the organism as the present investment, this may work out fine. It is this approach that is analysed further in this paper; see also [6], Chapter 7.

In this case a basic assumption is that the environment of an organism has the potentiality or regularity to provide future revenues in return for present investments.

L. Antunes, M. Paolucci, and E. Norling (Eds.): MABS 2007, LNAI 5003, pp. 96–111, 2008.

This is a nontrivial characteristic of an environment, that often depends on the presence of other organisms in the environment. For example, for species developing agriculture, the activity of sowing in the present, depending on the potential of the seed, leads to growth of food or other products that are in the organism's interest. Another example, which is taken as a case study in this paper, is that other agents are present in the environment that offer the future returns when they are favoured by an agent, depending on their own intertemporal decision making.

Godfrey-Smith [7], p. 3 relates environmental complexity to the development of cognition, as briefly formulated in his Environmental Complexity Thesis as: 'The function of cognition is to enable the agent to deal with environmental complexity'. For the case considered here, the agent needs a cognitive system that is able to make a decision where a current investment has to be compared to a future revenu. So, it needs cognitive facilities to predict future revenues based on the present world state and the world's regularities, and to compare such predicted future revenues to investments to be made in the present. These processes require nontrivial cognitive capabilities, the more so as the world's regularities usually have a probability aspect in them, that also has to be accounted for in the decision. These cognitive processes are usually called 'intertemporal decision making'; cf. [11]. To cope with the world's risks that in some cases predicted revenues will not come true, in such decision making the future revenues have to be estimated higher than the present investment, for example, by taking into account a certain interest rate. In the literature on intertemporal decision making, the environmental regularity or probability to indeed provide revenues in return usually is not modelled in a detailed manner, and not adapted on the basis of the agent's experiences. Experiments and models often focus on one subject and its expectations, and do not address how these relate to the real environment. In fact, to estimate the risk of not getting the future revenues in return, the model of intertemporal decision making for the subject should be combined with an environment-dependent model describing how based on its experiences the subject estimates when the environment indeed returns revenues for investments of the subject. In this way the agent can learn and adapt itself to the world's regularities or potentialities.

In this paper, these issues are analysed and tested by creating an artificial society. As part of the model, for any of the agents also the environment is modelled in a detailed manner as the rest of the society. By formal analysis and simulation it is investigated how agents endowed with a cognitive model for intertemporal decision making can choose for altruistic behaviour by providing services (for free) to other agents in the present and provide revenues in their own interest in the future. The guarantee or probability that revenues indeed are returned by the environment, depends in this case on other agents in the environment receiving the services, that in the future may or may not provide services in return. To estimate the risk of not getting the future revenues in return, the model of intertemporal decision making is combined with an environment model, which in this case is a model for evolution of trust in other agents based on experiences with them (adopted from [9], [10]). If the agent experiences over time that another agent does not provide services, the trust in this agent becomes lower; if it does provide services, trust becomes higher. Having such a dynamic environment model enables the agent to become better adapted to the environment. One of the main properties to verify is whether indeed agents with a cognitive system for trust-based intertemporal decision making do well over time, in contrast to agents that lack such a cognitive system. In other words, do agents with a more sophisticated cognitive system become healthier (or fitter) than their less developed colleagues?[1]

To create an artificial society of multiple agents performing trust-based inter-temporal decision making, several modelling approaches have been used. First, the LEADSTO simulation environment [4] was used for rapid prototyping, i.e., to create a high-level declarative specification of the simulation model, involving a small number of agents (six in this case). When this specification turned out to show coherent behaviour, it was used as a blueprint to create a large-scale version of the model in the NetLogo environment [14]. Finally, to analyse the simulation model and its outcomes, the predicate logical Temporal Trace Language (TTL, see [5]) was used to specify a number of global dynamic properties that are relevant for the domain in question. An example of such a property is "agents that anticipate on the future eventually become fitter". Using the TTL Checker Tool, these properties have been checked automatically against the traces resulting from the LEADSTO and NetLogo simulations.

In Section 2 the model for trust-based inter-temporal decision making is presented in detail. Section 3 describes the simulation model, which was designed and formally specified at a conceptual, temporal logic level in the form of local dynamic properties (in LEADSTO) for the basic mechanisms, and implemented in NetLogo. Section 4 discusses some of the simulation results. In Section 5 global dynamic properties are formulated and formalised in TTL, and by formal analysis logical interlevel relations between such a global property and local properties specifying the basic mechanisms are established, providing insight in their dependencies. Section 6 is a discussion.

2 Trust-Based Intertemporal Decision Making

The adaptation mechanism introduced in this paper is a decision theoretic function that includes a dynamic trust factor. The function is used for inter-temporal decisions, i.e., those decisions that compare investments or revenues at one point in time to those at another point in time; e.g., deciding whether to buy a car with a not-so-nice colour which you can take home immediately or waiting half a year for a car in your desired colour. Often uncertainty is involved about events that are more distant in time; they may depend on the environment's dynamics which can be unpredictable. For example, the event of making an investment in the present is certain, while an expected revenue in the future may be uncertain, and depend on the environment. The particular decisions concerned here involve the cooperation between agents, where the inter-temporal aspect is the expected reciprocity of cooperation: if I help you now, you will help me later. For such patterns to occur, as part of the agent's cognitive system an adequate decision model is essential. Such a model should include an environment model to predict future revenues upon present investments. In our model, the environment model has the form that agents maintain trust in other agents: they adapt their trust in other agents' willingness to help based on experienced (non)cooperations over time. Thus our agent model consists of two main parts: one part concerns the intertemporal decision making (to cooperate or not), and a second part concerns the updating of trust in other agents based on experiences. Both are described below.

---

[1] We acknowledge that this research question has some relationships to well-known studies by Trivers [13] and Axelrod [2], [3]. See Section 6 for an extensive discussion about this topic.

We are concerned with the following decision situation. Consider a set of agents, where each agent is working towards its own benefit. Each agent has a certain *fitness*, which is represented as a real number. At every point in time, agents are able to ask each other whether they want to cooperate or not. Let us assume that agent *x* requests agent *y* to cooperate. Such a cooperation has cost *c* for agent *y* and provides reward *f* to agent *x*. In response to this request, agent *y* evaluates the benefit of cooperation (i.e., calculates if the reward outweighs the cost). Based on this evaluation, agent *y* either accepts the request (agent *x* receives *f* and agent *y* pays *c*) or declines the request (agents *x* and *y* neither receive nor pay anything), and both agents' fitness is updated accordingly (taking into account *f* and *c*). Note that the evaluation function contains a reciprocity factor: if agent *y* cooperates with agent *x* then agent *y* can reasonably assume that agent *x* will later return the favour.

Inter-temporal choice is a decision in which the realisation of outcomes may lie in the imminent or remote future. Recently, inter-temporal choice has caught the attention in the literature on behavioural decision-making [11]. Before this, results on the subject were mainly due to the research contributions in related fields, like economics and animal psychology. The standard agent model for decision-making over time is a framework called time discounting [11], which works according to the same principles as interest that one receives on a bank account: I calculate a delayed reward back to its current value based on the interest that I would receive for it.

We use a similar agent model for inter-temporal decision making here, extended to our particular decision situation (involving reciprocity for cooperation) in two main ways. Firstly, the decisions involve an explicit model the agent has of (regularities in) the environment, in this case incorporating the other agents. This results in parameters for *trust* of the agent in other agents. As explained below, the value of this parameter evolves over time as a consequence of monitoring (regularities in) the environment over time, i.e., the experienced (non)cooperations. Secondly, the individual decisions are concerned with choosing between (1) a possible reward in the remote future and (2) having no immediate cost, rather than choosing between an immediate and delayed reward (as investigated traditionally in time discounting). In the model, the discounted value $f_{discounted}$ of a future reward is calculated by: $f_{discounted} = f * 2^{-\beta * (1 - (tr + 1)/2)}$, where

f : REAL	= future reward
$\beta \in [0,1]$	= discount factor (i.e., the higher β, the lower the expected future reward)
$tr \in [-1,1]$	= trust in the agent who asks you to cooperate (thus, the higher tr, the higher the expected future reward)

If the discounted future reward evaluates higher than (or equal to) the current (immediate) cost, the agent decides to cooperate[2]. In other words:

If ($f_{discounted} \geq c$), then cooperate, else do not cooperate

where c : REAL is the immediate cost.

---

[2] Note that the model as presented here does not explicitly take the time point at which the future reward is expected into account. In order to do this, the ratio between the period after which the agent expects to receive its reward and the period after which it expects to lose its costs can be incorporated in the exponent of the formula.

The next sections show how it was tested how agents that use this decision function develop in a multi-agent society. The prediction is that these agents will show altruistic behaviour, will establish a larger social network than agents without such a decision function (i.e., agents that are not able to estimate the future reward, and thus never cooperate), and will eventually get a higher fitness.

Agents adjust their trust values in other agents according to the following principle: if I ask you to cooperate and you accept, then I increase my trust in you; if you decline, then I decrease my trust in you. For modeling such adaptation of trust over time, we use a trust function that was presented in [9], [10]. This function, as applied here, takes the response of an asked agent (accept/decline) to determine how to revise the trust value. Such a response $e \in [-1,1]$ evaluates to 1 if the agent accepts or -1 if the agent declines. A scaling factor $\delta \in [0,1]$ (which is constant throughout the experiments) determines how strongly an agent is committed to its trust values: a higher δ means that an agent puts much weight on its current trust value and lets an (non)cooperation experience not weigh so heavily; and vice versa. In the model, when the outcome of a request to cooperate is known, we calculate the trust value tr_{new} as follows: $tr_{new} = \delta * tr + (1 - \delta) * e$, where

$tr \in [-1,1]$	= current trust value,
$\delta \in [0,1]$	= scaling factor (constant),
$e \in [-1,1]$	= the response of the agent who you asked to cooperate.

Thus, each agent maintains a list of trust values for all other agents in the environment. The model also includes a *cooperation threshold* $ct \in [-1,1]$ such that agent *x* only requests cooperation with agent *y* if trust of agent *x* in agent *y* is above this threshold.

3 Simulation Model

This section describes the simulation model that was used to investigate the impact of trust-based inter-temporal decision making on fitness. In accordance with the explanation in Section 2, a society of agents is analysed, where each agent can ask the other agents for a favour. An example of such a favour is that an agent asks another agent for help during a house removal event. Figure 1 briefly sketches the different possible scenarios in one round of interaction between two agents x and y.

First, a conceptual model has been created of this domain in LEADSTO, modelling a society of six agents. In this model, three agents used the decision function introduced in Section 2, whereas the other three agents were not equipped with this function. These three agents simply always selected the current reward, thus they never cooperated. Next, large-scale simulations of the domain have been implemented in NetLogo, involving societies up to 200 agents. In these simulations, again half of the agents used the inter-temporal decision function, whilst the other half did not. Other parameter settings (both in the LEADSTO as in the NetLogo simulation) were as follows. In the decision function, c=10, f=14, β=0.80. In the trust function, δ=0.9. The request threshold ct (see Figure 1) is -0.3, and the daily decrease in fitness d is 3.

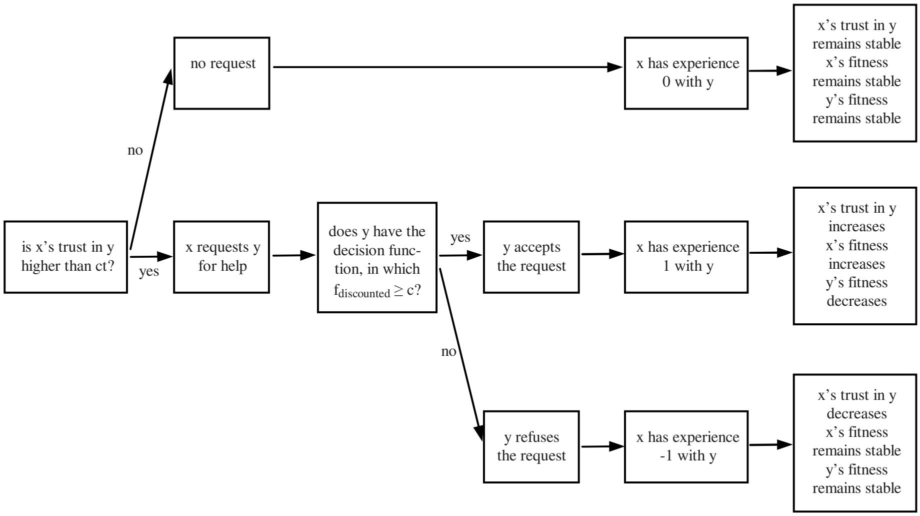


Fig. 1. Scenario sketch of one round of interaction

The basic building blocks of a LEADSTO simulation model are so-called Local Properties (LP's, in contrast to the Global Properties that are shown in Section 5). The format of these local properties is defined as follows. Let α and β be state properties of the form 'conjunction of atoms or negations of atoms', and e, f, g, h non-negative real numbers. Then, the notation $\alpha \twoheadrightarrow_{e, f, g, h} \beta$, means:

If state property α holds for a certain time interval with duration g, then after some delay (between e and f) state property β will hold for a certain time interval of length h.

The local properties (LPs) that were used for our conceptual model are shown below. When this declarative specification turned out to show coherent behaviour, it was used as a blueprint to implement the model in NetLogo.

LP1 Trust Adaptation

Trust is adapted on the basis of experiences. Here, 'delta' is a constant, e.g. 0.9.

∀x,y:agent ∀tr:real ∀e:real

has_trust_in(x, y, tr) ∧ has_experience_with(x, y, e) $\twoheadrightarrow_{0,0,1,1}$

has_trust_in(x, y, delta × tr + (1 - delta) × e)

LP2a No Request Making

An agent x makes no request to y if its trust in y is below the cooperation threshold. Here, 'ct', is a constant, e.g. –0.25.

∀x,y:agent ∀tr:real

has_trust_in(x, y, tr) ∧ cooperation_round ∧ tr < ct ∧ x \= y $\twoheadrightarrow_{0,0,1,1}$ not request_from_to(x, y)

LP2b Request Making

An agent x makes a request to y if its trust in y is above the cooperation threshold. Here, 'ct', is a constant, e.g. –0.25.

∀x, y:agent ∀tr:real

has_trust_in(x, y, tr) ∧ cooperation_round ∧ tr >= ct ∧ x \= y $\twoheadrightarrow_{0,0,1,1}$ request_from_to(x, y)

LP3a Decision Function (1)
If you have the ability to evaluate the future reward, and the discounted value of the future reward is evaluated higher than the current cost, then accept a request for cooperation.
∀x,y:agent ∀tr:real ∀b:real
has_trust_in(x, y, tr) ∧ request_from_to(y, x) ∧ has_decision_function(x) ∧
has_beta(x, b) ∧ current <= future $\times$ $2^{-b * (1 - (tr + 1) / 2)}$ $\twoheadrightarrow_{0,0,1,1}$
accepts_cooperation_with(x, y)

LP3b Decision Function (2)
If you have the ability to evaluate the future reward, and the discounted value of the future reward is evaluated lower than the current cost, then refuse a request for cooperation.
∀x,y:agent ∀tr:real ∀b:real
has_trust_in(x, y, tr) ∧ request_from_to(y, x) ∧ has_decision_function(x) ∧
has_beta(x, b) ∧ current > future $\times$ $2^{-b * (1 - (tr + 1) / 2)}$ $\twoheadrightarrow_{0,0,1,1}$
refuses_cooperation_with(x, y)

LP3c Primitive Decision Function
If you don't have the ability to evaluate the future reward, then refuse a request for cooperation.
∀x,y:agent ∀tr:real
has_trust_in(x, y, tr) ∧ request_from_to(y, x) ∧ not has_decision_function(x) $\twoheadrightarrow_{0,0,1,1}$
refuses_cooperation_with(x, y)

LP4 No Interaction
There is no interaction if no requests are done.
∀x,y:agent
not request_from_to(y, x) $\twoheadrightarrow_{0,0,1,1}$ no_interaction_between(x, y)

LP5a Good Experience
If agent x cooperates with agent y, then y has a good experience with x.
∀x,y:agent
accepts_cooperation_with(x, y) $\twoheadrightarrow_{0,0,1,1}$ has_experience_with(y, x, 1)

LP5b Neutral Experience
If there is no interaction between agent x and y, then y has a neutral experience with x.
∀x,y:agent
no_interaction_between(x, y) $\twoheadrightarrow_{0,0,1,1}$ has_experience_with(y, x, 0)

LP5c Bad Experience
If agent x does not cooperate with agent y, then y has a bad experience with x.
∀x,y:agent
refuses_cooperation_with(x, y) $\twoheadrightarrow_{0,0,1,1}$ has_experience_with(y, x, -1)

LP6a Fitness Adaptation
If agent x cooperates with i agents, then its fitness will decrease with the current reward multiplied with i.
∀x,y:agent ∀f:real ∀i:integer
has_fitness(x, f) ∧ $\Sigma_{y=a(1)}^{agents}$ accepts_cooperation_with(x, y) = i $\twoheadrightarrow_{0,0,1,1}$
has_fitness(x, f - i $\times$ current)

LP6b Fitness Adaptation
If i agents cooperate with agent x, then x's fitness will increase with the current reward multiplied with i.
∀x,y:agent ∀f:real ∀i:integer
has_fitness(x, f) ∧ $\Sigma_{y=a(1)}^{agents}$ accepts_cooperation_with(x, y) = i $\twoheadrightarrow_{0,0,1,1}$
has_fitness(x, f + i $\times$ future)

LP7 Fitness Decline
After each round all agents in the population become older and lose some fitness. The daily decrease in fitness d is a constant, e.g. 3.
∀x:agent ∀f:real
has_fitness(x, f) ∧ cooperation_round $\twoheadrightarrow_{0,0,1,1}$ has_fitness(x, f - d)

4 Simulation Results

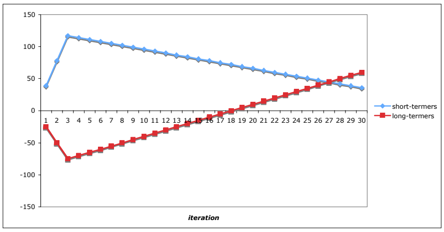
This section discusses the results of the simulations. As mentioned before, first a test simulation has been performed in LEADSTO, involving a society of 6 agents. Next, large-scale simulations have been performed in Netlogo, both for 6, 25 and 200 agents. Due to space limitations, only the results of the Netlogo-6 and -200 simulations are shown here. Nevertheless, all simulations show the same global trend: initially, all agents request each other for help. However, only the agents with the inter-temporal decision function are willing to cooperate; they accept all requests, thereby showing some kind of altruistic behaviour. The other agents show egoistic behaviour: they refuse all requests. As a result, the trust in the cooperating agents increases, whilst the trust in the non-cooperating agents decreases. This development continues for a while, until the trust in the non-cooperating agents is so low that even the agents with the inter-temporal decision function (the cooperating agents) are not willing to help them anymore. However, they still continue helping the other cooperating agents. Thus, a group emerges of agents that are helping each other, whilst the other agents get isolated: they do not interact with any agent anymore. As a consequence, the fitness of the agents with the decision function (which was first rather low, since these agents were initially exploited by the other agents) recovers, and the fitness of the agents without decision function gets lower and lower. These results confirm the hypothesis that agents that have the ability for inter-temporal decision making will show altruistic behaviour, which leads to a bigger social network, and eventually to a higher fitness.

Figure 2 and 3 show the results for the NetLogo simulation with 6 agents and 200 agents, respectively. In these figures, the following measurements are shown: fitness – the average fitness of the agents for the short-termers[3] and long-termers, respectively; trust – the average trust of each type of agent (short-term or long-term) in the other type of agent; requests – total number of requests that agents have done (cumulative over time); cooperations – total number of cooperations (cumulative over time). A cooperation ($type_1$ -> $type_2$, e.g., long -> long) is an accept from agent *y* on a request from agent *x*, where $type_1$ is the type of agent *x* and $type_2$ is the type of agent *y*.

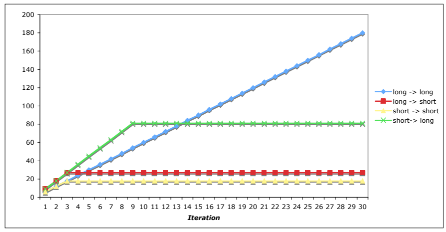
We can do a number of observations looking at the 6-agents and 200-agents Netlogo traces. Firstly, we see that the results of the simulation with 6 agents are consistent with the results of the LEADSTO simulation. Most importantly, we observe 1) the same trends and 2) that there is a turning point at the third iteration, after which the trust of the long-termers in the short-termers is sufficiently low that they do not cooperate anymore.

Secondly, the results are scaleable (from 6 to 200 agents) with some notable differences for the various measures. For *fitness*: we see that the fitness of the short-termers increases rapidly at first, and then decreases – but slower for 200 agents than for 6 agents. This is a result of the fact that short-termers receive reward from many more agents in the first three iterations and (because the metabolism is equal for both

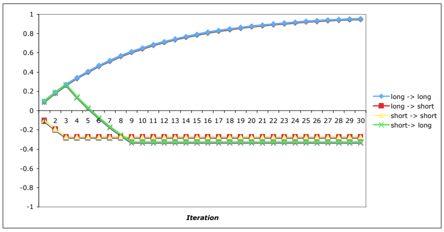
[3] This section uses the term 'short-termers' to indicate those agents that do not operate based on a decision function, but simply never accept cooperation requests. The 'long-termers' use the decision and trust-update function presented earlier to decide on accepting cooperation requests.



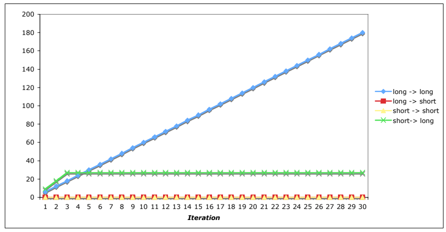
Development of fitness over time



Number of requests over time

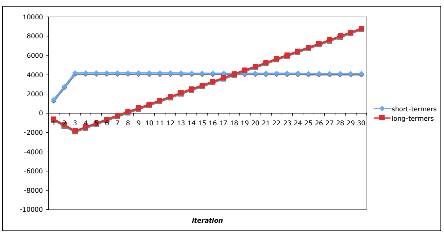


Development of trust over time

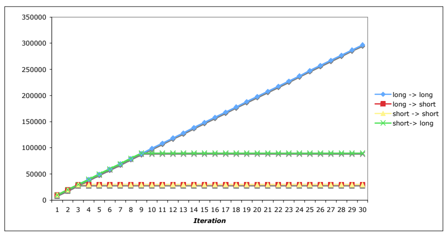


Number of cooperations over time

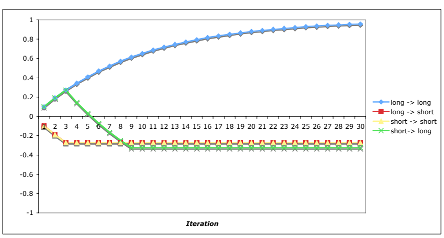
Fig. 2. NetLogo results for simulation with 6 agents



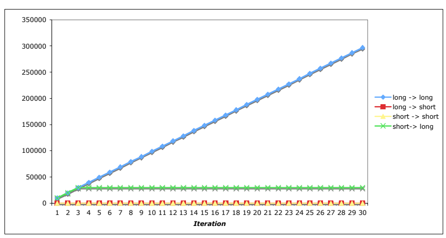
Development of fitness over time



Number of requests over time



Development of trust over time



Number of cooperations over time

Fig. 3. NetLogo results for simulation with 200 agents

traces) can live off this much longer. The point at which the fitness curves cross each other is also earlier for 200 agents than with 6 agents. This is because the long-termers also benefit from the fact that after iteration 3, they receive cooperation from many more other long-termers, leading to faster fitness increase. For *trust*: we do not observe differences between the two traces. For *number of requests*: most noteably, we see that the (short -> long) curve crosses the (long -> long) curve much later for the 6 agents trace (at iteration 14 instead of 9). Initially, the (short -> long) curve grows quicker than the (long -> long) curve because the growth-factor of the (short -> long) is the number of long-termers and the growth-factor of the (long -> long) is the number of long-termers – 1 (since a long-termer does not do a request to itself). Eventually they cross, because the (short -> long) curve does not grow further after iteration 9 because of the fixed cooperation threshold. For *number of cooperations*: in both traces we see that the long-termers never cooperate with the short-termers, and short-termers never cooperate with each other. The long-termers always cooperate with each other. The short-termers cooperate with the long-termers up till iteration 3.

To better illustrate the emergence of groups in the society, the cooperation between the agents can be visualised, using the organisation visualisation tool by [8]. A screenshot of this tool is depicted in Figure 4, for the results of the NetLogo simulation of 25 agents. Here, the nodes denote the agents, the edges denote cooperation. The size of the nodes and the thickness of the edges indicate the number of cooperations the agents were involved in during the whole simulation. Figure 4 clearly illustrates that the long-termers have established a network, whilst the short-termers have become isolated.

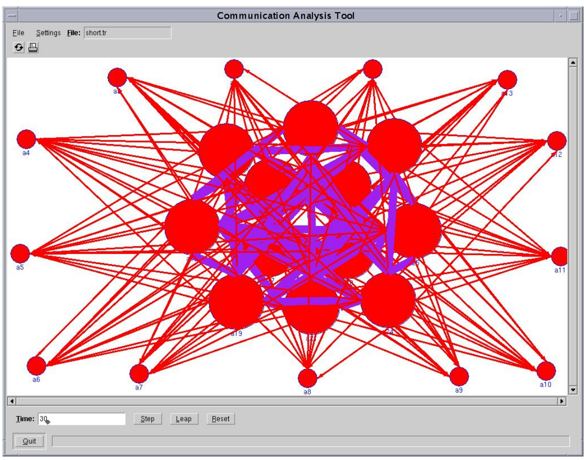


Fig. 4. Emergence of groups in the society

5 Analysis

This section addresses further formal analysis of the simulation model and its results. In particular, a number of global dynamic properties have been formalised in TTL, and it was verified whether the local temporal properties defining the simulation model entail these global properties. This type of analysis can be performed in two ways: (1) using a checker tool, it can be verified automatically whether the global properties hold for the generated simulation traces, and (2) interlevel relations can be established between the local properties (see Section 3) and the global properties, which can be verified by mathematical proof. These two types of analysis are addressed by Section 5.1 and 5.2, respectively.

5.1 Checking Global Properties

A number of global properties have been identified that are relevant for the domain of trust-based inter-temporal decision making. These properties have been formalised in the TTL language. Three of them are shown below:

FM Fitness Monotonicity

If x has the cognitive system for decision making, then there exists a time t such that for all t1 and t2 after t, with t1 < t2, the fitness of x at t2 is higher than the fitness of x at t1.

∃t ∀x:AGENT ∀t1,t2≥t ∀f1,f2:REAL
[[∀t state(γ, t) |= has_decision_function(x)] & t1<t2 & state(γ, t1) |= has_fitness(x, f1)
⇒ ∃f2≥f1 state(γ, t2) |= has_fitness(x, f2)]

Here, for example, state(γ, t1) |= has_fitness(x, f1) denotes that in trace γ in the state at time point t the agent x has fitness f1.

DMAF Decision Making Agents get Fitter

Eventually, all agents with the cognitive system for decision making, will be healthier than the agents without this system.

∀t ∀x,y:AGENT ∀f1,f2:REAL
[state(γ, t) |= has_decision_function(x) ∧ not has_decision_function(y) &
state(γ, last_time) |= has_fitness(x, f1) ∧ has_fitness(x, f2)
⇒ f1>f2]

NDMA Network of Decision Making Agents

All agents with the cognitive system for decision making will always cooperate with each other.

∀t ∀x,y:AGENT
[state(γ, t) |= has_decision_function(x) ∧ has_decision_function(y) ∧ request_from_to(x, y)
⇒ state(γ, t+1) |= accepts_cooperation_with(y, x)]

A specific software environment has been built, which takes as input a set of traces and a formalised property in TTL, and verifies whether the property holds for the traces [5]. Using these kinds of checks, the above properties have been checked against the traces mentioned in Section 4 involving 6 and 25 agents. They all turned out to hold, which validates the above statements, such as "decision making agents get fitter", for the simulation traces.

5.2 Interlevel Relations

This section aims at getting insight in why on the basis of the mechanisms as modelled in the local properties (see Section 3) the global properties are obtained. This is done by an analysis, for a given global property, based on a hierarchical AND-tree of dynamic properties at different levels, in which the branching specifies interlevel relations. This tree represents an argumentation why the global property holds, given as premises that the local properties hold. The tree considered here focuses on the highest level property FM shown in the previous paragraph, and is shown in Figure 5.

Roughly spoken the argumentation runs as follows. There are two ways to affect fitness. One way is (1) to increase it by earning profit by investing in cooperations with other cooperative agents. Another way to affect fitness is (2) to decrease it by

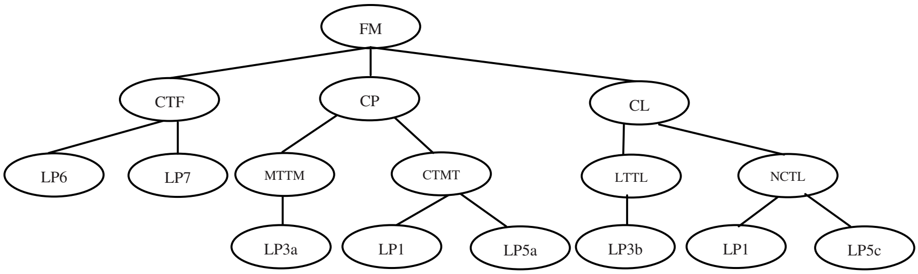


Fig. 5. AND-tree of interlevel relations for Global Property FM

investing in cooperations with non-cooperative agents so that the investment gives no return. Initially, it can be unclear which contribution to fitness dominates. However, by maintaining trust in other agents based on the experiences in cooperation, the agent learns to discriminate the agents in the categories (1) and (2), and thus decides not to invest in the noncooperative agents anymore. Therefore, after some time point contribution (1) to the fitness dominates, and thus fitness becomes monotonically increasing. This rough outline of the argumentation was detailed as follows, making use of properties at one level lower:

CTF Fitness Change by Cooperation Results

Cooperation profit contributes to fitness increase, and cooperation loss contributes to fitness decrease.

$\forall$t1,t2 $\forall$x:AGENT $\forall$f1,d1,d2
[state(γ, t1) |= has_fitness(x, f1) &
cooperation_profit_over(γ, x, d1, t1, t2) & cooperation_loss_over(γ, x, d2, t1, t2)]
$\Rightarrow$ state(γ, t2) |= has_fitness(x, f1+d1-d2-d*(t2-t1))]

CP Cooperation Profit

If x has the cognitive system for decision making, then there exists a time t such that for all t1 and t2 after t, with t1 < t2 and length L1 $\leq$ t2-t1 $\leq$ L2, between t1 and t2 the amount of revenues contributed by other cooperative agents y to x in cooperations is at least M higher than the amount of expenses invested by x in cooperation with these agents y.

$\forall$x [has_decision_function(x) $\Rightarrow$
$\exists$t $\forall$t1,t2$\geq$t [L1 $\leq$ t2-t1 $\leq$ L2 $\Rightarrow$ $\exists$d cooperation_profit_over(γ, x, d, t1, t2) & d $\geq$ M]

Here L1, L2 and M are constants that can be given specific values.

CL Cooperation Loss

If x has the cognitive system for decision making, then there exists a time t such that for all t1 and t2 after t, with t1 < t2 and length L1 $\leq$ t2-t1 $\leq$ L2, between t1 and t2 the amount of loss due to other noncooperative agents z to x in cooperations becomes lower than M.

$\forall$x [has_decision_function(x) $\Rightarrow$
$\exists$t $\forall$t1,t2$\geq$t $\forall$d [L1 $\leq$ t2-t1 $\leq$ L2 & cooperation_loss_over(γ, x, d, t1, t2) $\Rightarrow$ d$\leq$M]

The property CP relates to the following two lower level properties (of which the formalisation has been omitted):

CTMT Investment in Cooperation Leads To More Trust

For an agent y with the cognitive system for decision making, if x invests in cooperation with y, then trust of y in x will increase

MTTMC More Trust Leads to More Cooperation

For an agent x with the cognitive system for decision making, more trust of x in y in leads to more investment of x in cooperation with y.

The property CL relates to the following two properties (of which the formalisation has been omitted):

NCTLT Non-Cooperation with a Cooperative agent Leads To Less Trust
For an agent y with the cognitive system for decision making, if x does not invest in cooperation with y, then trust of y in x will decrease.

LTTLC Less Trust Leads to Less Cooperation
For an agent x with the cognitive system for decision making, less trust of x in y in leads to less investment of x in cooperation with y.

Based on the properties defined above, the logical relationships are as follows. Suppose x and y both have the cognitive system for decision making, and an initial amount of trust of x in y and of y in x is available, and y is cooperative. Moreover, assume that the properties CTMT and MTTMC hold. Then:

1. The initial trust of x in y leads to some investment by x in cooperation with y by property MTTMC.
2. This investment of x in cooperation with y leads to more trust of y in x, by CTMT.
3. The increased trust of y in x leads to more investment of y in cooperation with x, by MTTMC.
4. This positive feedback process continues for some time, and thus will increase the amount of revenues of cooperation of agent x (and of y) after some time point to a level that profit is obtained from the cooperation, which shows that property CP is implied by CTMT and MTTMC.

A similar argument can be made for x with respect to a noncooperative agent z. Assume that the properties NCTLT and LTTLC hold. Then:

5. The initial trust of x in z leads to some investment by x in cooperation with y by property MTTMC.
6. As z is noncooperative, this does not lead to more investment of z in cooperation with x.
7. Because of noncooperation of y with x, x will decrease its trust in z by NCTLT.
8. The decreased trust of x in z leads to less investment of x in cooperation with z, by LTTLC.
9. This negative feedback process continues for some time, and thus will decrease the amount of loss of cooperation of agent x with z after some time point, showing that property CL is implied by NCTLT and LTTLC.
10. If after some time point over any interval there is profit from cooperation, and the loss becomes lower, and profits and losses affect fitness in positive, respectively negative manners, then after a point in time the fitness at t2>t1 will be higher than the fitness at t1. This expresses that property FM is implied by CP, CL, CPFI, and CLFD.

By this argumentation the logical interlevel relationships as shown in Figure 5 are obtained. The semantics of this tree is as follows: if a certain property is connected to a number of lower level properties, then the conjunction of the lower level properties (logically) entails the higher level property. As the picture shows, eventually the intermediate properties can be related to the local properties as shown in Section 3.

6 Conclusion

The work reported in this paper contributes to the analysis of a paradoxical question from an evolutionary perspective: how can altruistic behaviour be in an agent's own interest? The question is addressed by involving a temporal dimension in which altruistic behaviour at one point in time can be in the agent's interest at a future point in time, depending on the environment. The claim is that to be able to make reasonable decisions, an agent needs a cognitive system for intertemporal decision making, in relation to a model of the environment to predict when indeed it may expect to provide revenues for the agent's investments by its altruist behavior.

To address this, a society of agents has been modelled, simulated and analysed. Some of the agents have a certain cognitive system for decision making that enables them to choose for altruistic behaviour. Part of this cognitive system is a model of the environment to predict whether future revenues may be expected in return, which is based on a dynamic model for trust in the other agents based on experiences with them over time, adopted from [9], [10]. This environment model is combined with a model for intertemporal decision making taken from the literature; e.g., [11].

It turned out that the agents with this cognitive system enabling them to anticipate on the future show more altruistic behaviour. As a result, these agents get a bigger social network, and in the end become healthier than the agents without such a cognitive system. This is in accordance with the theory of how altruism emerged in Nature as a result of more elaborated capabilities of mankind for inter-temporal decision making; e.g., [6]. Among the agents with the ability to anticipate on the future, different variants can be identified, for example, by taking different values for the discount factor β. In future work, it will be investigated how a society with such different variants develops. An interesting question will then be to explore what is the 'optimal' inter-temporal decision function under different circumstances.

We are aware of the fact that there exists a substantial body of work on the evolution of cooperation, with most notably the seminal works by Trivers [13] and Axelrod [2], [3]. Trivers elaborates on the mathematics of *reciprocal altruism* – the form of altruism in which an organism provides a benefit to another in the expectation of future reciprocation – and includes human reciprocal altruism as a case study to illustrate the model. Axelrod's work investigates the question under what conditions cooperation will emerge in a world of egoists without central authority. He demonstrates that 'altruistic' strategies such as Tit-for-Tat perform well in iterated prisoner's dilemma (IPD) tournaments. Although these works address the same fundamental question on reciprocal altruism as in this paper (how can altruistic behaviour be in an agent's own interest?), our perspectives are significantly different. First, the context of our simulations differs from the IPD in a number of ways. For example, it is more realistic (the domain of providing services is less artificial than that of prisoners) and it is not symmetric (it is possible that agent A does a request to agent B, while B does not do a request to A). Second, we addressed a question that is narrower than the one posed by Axelrod. Whereas Axelrod investigated which types of strategies performed best in a heterogeneous population with many different strategies, we took the theory of Dennett [6] as inspiration in order to explicitly compare two types of agents: those with a decision function, and those without. For the former type, we consider the decision function of the involved individuals to be based on the decision-theoretic notion of utility discounting. Ainslie [1] reports on the existence of such a function (exponential or hyperbolic) based on collected field data. Dennett [6] then uses such function as the basis of an evolutionary explanation of free will. To our best knowledge, to include such a discount utility function in the context of reciprocal altruism has not been addressed explicitly before. Also, we explicitly include an adaptive trust parameter to the decision function, which can be considered a novel contribution to the research on this topic. Nevertheless, we predict that strategies such as Tit-for-Tat can be modelled as special cases of our trust-based intertemporal decision function (e.g., using specific parameter settings such as $\delta=0$). In the future, we will perform experiments with heterogeneous populations, including, e.g., Tit-for-Tat agents and intelligent cheating agents. In this respect, an interesting challenge is to explore whether there are any circumstances in which our trust-based intertemporal decision agents actually perform better than Tit-for-Tat agents.

Another possible direction for future research is to take a ‘truly’ evolutionary perspective. In the current version of the model, each simulation run only corresponds to the lifetime of one generation of agents. It would be interesting to extend the model with an evolutionary component, i.e., by taking the resulting population of the current simulations, and creating offspring based on their fitness functions. Although we expect that the general trend of such simulations will not be different from the ones shown here, it would strengthen our claims of reproducing the claims made by [6] and [7] from an evolutionary computation perspective.

Finally, the model can be made more realistic by incorporating more concepts that are characteristic for human cooperation and trust. For example, one can distinguish between more and less successful cooperation. Also, the trust update function can be replaced by a reputation mechanism, where agents adapt their trust in others not only based on direct experiences, but also on received communication about other agents.

With respect to scaling to larger populations, it is obvious that the complexity of the approach increases with the amount of agents. Since each agent has to maintain a list of all other agents and the associated trust scores, the complexity of the approach is $O(n^2)$. This may become problematic in case populations of millions of agents are considered. However, since the main goal of this research is to provide an answer to a philosophical question (i.e., what is the impact of trust-based inter-temporal decision making on fitness?) rather than providing a model that can be applied in real world applications, this problem is beyond the scope of this paper.

References

1. Ainslie, G.: Breakdown of Will. Cambridge University Press, Cambridge (2001)
2. Axelrod, R.: The Evolution of Cooperation. Science 211(4489), 1390–1396 (1981)
3. Axelrod, R.: The Evolution of Cooperation. Basic Books, New York (1984)
4. Bosse, T., Jonker, C.M., van der Meij, L., Treur, J.: A Language and Environment for Analysis of Dynamics by SimulaTiOn. Operating Systems 16(3), 435–464 (2007); shorter version In: Eymann, T., Klügl, F., Lamersdorf, W., Klusch, M., Huhns, M.N. (eds.) MATES 2005. LNCS (LNAI), vol. 3550, pp. 435–464. Springer, Heidelberg (2005)

5. Bosse, T., Jonker, C.M., Meij, L., van der Sharpanskykh, A., Treur, J.: Specification and Verification of Dynamics in Cognitive Agent Models. In: Proceedings of the Sixth International Conference on Intelligent Agent Technology, IAT 2006, pp. 247–254. IEEE Computer Society Press, Los Alamitos (2006)
6. Dennett, D.C.: Freedom Evolves. Viking Penguin, New York (2003)
7. Godfrey-Smith, P.: Complexity and the Function of Mind in Nature. Cambridge University Press, Cambridge (1996)
8. Hoogendoorn, M., Treur, J., Yolum, P.: A Labelled Graph Approach to Support Analysis of Organizational Performance. In: Proceedings of the International Workshop on Agent-based Technologies and Applications for Enterprise Interoperability, ATOP 2005, pp. 49–60 (2005)
9. Jonker, C.M., Treur, J.: Formal Analysis of Models for the Dynamics of Trust based on Experiences. In: Garijo, F.J., Boman, M. (eds.) MAAMAW 1999. LNCS, vol. 1647, pp. 221–232. Springer, Berlin (1999)
10. Jonker, C.M., Treur, J.: A Framework for Specification and Analysis of Models for the Dynamics of Trust based on Experiences. Technical Report, Vrije Universiteit Amsterdam (2005)
11. Loewenstein, G.F., Elster, J.: Choice over time. Russel Sage Foundation, New York (1992)
12. Sober, E., Wilson, D.S.: Unto Others: The Evolution and Psychology of Unselfish Behaviour. Harvard University Press, Cambridge (1998)
13. Trivers, R.L.: The Evolution of Reciprocal Altruism. Quarterly Review of Biology 46, 35–57 (1971)
14. Wilensky, U.: NetLogo. Center for Connected Learning and Computer-Based Modeling. Northwestern University, Evanston, IL (1999), http://ccl.northwestern.edu/netlogo

Multi-agent Model of Technological Shifts

James G. McCarthy, Tony Sabbadini, and Sonia R. Sachs

jim@agiledesigntech.com,
tonysabbadini@gmail.com,
soniasachs@gmail.com

Abstract. We present a multi-agent simulation model of a concentrated industry undergoing technological change. The simulation consists of heterogeneous firm agents, heterogeneous consumer agents, and one intermediating agent, the auctioneer, who aggregates market information to transmit to agents and to arrive at prices. Firms seek to maximize profit by estimating market demand, entering and exiting product markets, and optimizing output based on strategic conjectures of competing firm behavior. Consumers maximize utility by optimizing consumption within their budget sets. The auctioneer matches firm research with shifting consumer preferences, diffuses product knowledge among firms and consumers, and sets prices. We study . technological shifts; the diffusion of new technologies that directly compete with and replace existing ones. Sensitivity analysis is discussed in terms of the effects that model parameters have on product adoption, demand, output, spending, price, and profits. We validate our model against actual industry data on film cameras versus digital cameras.

1 Introduction

A great deal of research has been done on how technology diffuses within an economy. Early models incorporated technological change at the level of national economies, but made no attempt to model the process of innovation or explain how or why it occurred. Until recently, most models have abstracted the behavior of individual agents (firms, consumers, governments) and relied on closed form expressions of aggregate effects. This has inhibited research on how innovation is affected by several likely important factors including heterogeneous agents, strategic (gaming) behavior, and the competitive environment faced by firms.

This paper describes a multi-agent system model that was developed to assist in a broader research project that studies causes, features and effects of technological innovation. This model aims to enable researchers to input relevant parameters that specifically identify economic factors such as industry concentration, rivalrous vs. cooperative firm behavior, effects of customer loyalty, market segmentation, different cost structures, and many other economically important characteristics at the level of individual agents. We have completed work on a baseline version and subjected it to parameter sensitivity testing, including different individual firm cost structures, different output/pricing regimes, and different consumer utility functions. This paper

L. Antunes, M. Paolucci, and E. Norling (Eds.): MABS 2007, LNAI 5003, pp. 112–127, 2008.

describes the current capabilities of the model, the results of our parameter sensitivity testing and validation, and the direction of our ongoing development effort.

In Section 2, we review related work on models for innovation and technological change, including recent multi-agent models. We describe the model that we developed in a narrative format in Section 3. In Section 4, we discuss some preliminary results and parameter sensitivity analysis. We also discuss validation results. We present conclusions in Section 5.

2 Related Work

Research on innovation and technological change, diffusion, and adoption has been active since at least the early twentieth century, when Joseph Schumpeter advocated a theory that emphasized waves of technological change as the prime factors driving economic growth. Technological change was incorporated into early models of innovation as something that happened at the level of national economies. Recently, however, there has been an increased focus by macro-economists on the micro-economic behaviors that cause higher level phenomena. Robert Solow's Nobel Prize winning work [1] illustrated that technological improvement is responsible for about four-fifths of the growth in US output per worker. Natural extensions of this work examined how and why individual agents (firms, consumers, governments) engage in research and adopt technological innovations. Most models of innovation that deal with effects at the level of the individual firm incorporate a diffusion process of some sort. In a diffusion model, agents learn of the innovation from other agents and adopt it when some criterion is met (usefulness, profitability, etc.).

Two seminal and widely cited diffusion models are Mansfield [2] and Bass [3]. Mansfield presented a deterministic model that explains the different rates at which an innovation is diffused and adopted by firms. Mansfield showed that the number of firms that introduce an innovation grows over time following a logistic function and that the rate of imitation is controlled by a coefficient, whose expected value is shown to be a linear function of 1) the profitability of introducing the innovation relative to that of alternative investments and, 2) the investment required to introduce the innovation as a percent of the average total assets of the firms. Bass studies the innovation process from a different perspective than Mansfield. The Bass model assumes that there is a number of people who have already adopted before time *t*, and that the probability that someone adopts given that he or she has not yet adopted depends on people's intrinsic tendency to adopt the new product (coefficient of innovation), and a "word of mouth" or "social contagion" effect (coefficient of imitation).

Stoneman [4] organizes the earlier models of diffusion into epidemic, rank, stock, and order models. Rank models assume that adopters have different characteristics, such as firm size, and therefore experience different benefits from the adoption of the new technology. Examples of this type of model are [5] and [6]. Stock models use the assumption that for a given cost of acquisition, adoption is not profitable beyond a certain number of adopters. They also assume that the cost of acquisition reduces over time, which results in additional adoptions. Examples in this category are [7] and [8]. Order models assume an order of adoption among the firms and that the decision that firms make to adopt take into account how moving down the adoption order will affect their profits (see [9]).

Early models of diffusion are not equilibrium models. They do not require, for example, that inputs equal outputs or that money spent on one item must reduce spending on another. Equilibrium diffusion models use the idea that diffusion results from a decline in price along the reservation price distribution, assuming that there is a distribution of reservation prices below which firms will adopt the new technology. These models consider the expectations of agents and the interaction between supply and demand of new technologies. Several authors have formalized this idea in their models [10], [11].

Unlike its predecessors, Nelson's model [12], which started the era of evolutionary models, used agents of bounded rationality. Many authors used the evolutionary approach in their models, incorporating the study of many interesting variables of technological change, such as the study of network externalities [13] and the study of product diversification [14], among others. See [15] and [16] for reviews of evolutionary models. Agent-based models in this area have contributed to extensions such as procedural decision-making behavior of firms that use feedback learning rather than the perfect rationality assumptions of analytical evolutionary models. However, early agent-based models of diffusion only include firm agents, leaving out both the consumer agents and the market. Also, they assume a stationary environment, where technologies cannot appear after the process has already started. Another limitation mentioned is that results of these earlier models lack statistical rigor and do not include sensitivity analysis of the model variables.

Recent models attempt to overcome some of these limitations. Fitness landscape models help answer the question of search strategies that agents apply when choosing among alternative technologies. Percolation models model dynamics of adoption, where agents communicate among each other in order to decide on adopting new technologies. Network models answer questions on the role of network relations in the rate of innovation and diffusion. Several publications have demonstrated how network characteristics affect technological change (see [15]).

Agent-based models have also been developed to study economic growth, which is closely related to the study of technology diffusion. Aspects of technological change and growth, such as exploitation-vs-exploration of innovation, diffusion speed, and knowledge structure are incorporated into several agent-based models of growth (see[17]).

Product innovations in an oligopolistic market with the goal of predicting market response is studied in[18]. Evolution of demand and its relationship to product innovations is modeled in [19], where consumer preferences co-evolve with new products offered by firms. As mentioned in [18], several agent-based models have studied the effects of heterogeneous strategies on technological change.

A few modeling aspects of technology diffusion are still open problems, such as the study of the relationship between mode of competition and innovation, the study of the co-evolution of innovations and demand, and the use of such models as guidance to firm strategies and policies. In this paper, we focus on the study of interactions between firms and consumers within the micro-economy based on the application of old and new technologies. We are interested in studying technological shifts, i.e., the diffusion of new technologies that directly compete and replace existing ones under the assumptions of a mode of competition (oligopoly with Cournot behavior). We pursue sensitivity analysis of model parameters, and, in a systematic way, model validation and testing of simulation results.

3 The Multi–agent Simulation Model Narrative

The model currently has three types of agents: firms, consumers, and a single auctioneer. After appropriate initializations, agents in the model act in the following sequence:

1. The auctioneer uses observed market outcomes to estimate a market demand function for each good. For a new product, external information is used for the estimate. He transmits this function to firms as well as total industry output for the previous period.
2. Firms combine total industry output, their own last period output, and the market demand function to estimate a residual demand function for their own output. They then produce a level of output in the current period that maximizes their expected profit in the period given these assumptions, and transmit the production information to the auctioneer. This method of setting output is referred to as using Cournot conjectures.
3. The auctioneer totals the output information from each firm and uses the estimated market demand function to set an estimated market clearing price.
4. Consumers choose consumption bundles, at the prices set by the auctioneer, which maximize their utility given their budget and other constraints. These bundles are reported to the auctioneer, who uses the information to update the demand function.

The auctioneer is a common feature in economic models. Its purpose is to aggregate supply, demand, and other market information to transmit to agents and to arrive at a price or prices at which transactions will occur. In equilibrium models the auctioneer arrives at a price at which supply exactly equals demand – a so called market clearing price. Our model is not an equilibrium model, although prices and quantities should converge to equilibrium values when they exist. We have chosen to use the artifice of an auctioneer to abstract away from the specific mechanisms by which firms learn about markets so that we can focus on the impact of strategic decisions about output and prices. There are numerous counterparts to the auctioneer in real world markets: the specialist in securities markets, industry and trade organizations, and government regulators. In practice, firms in concentrated industries know a lot about their competitors. In fact, it is frequently the case that public announcement of intended actions is in the firm's own interest, especially when competitors' actions make it clear that they want to co-operate to maintain high prices. Since our goal is to model a concentrated industry with large players, we feel justified in assuming that the information provided by the auctioneer is available.

Updating the market demand function is not a trivial matter. When prices change or when new products are added, price/quantity outcomes will generally include wealth and substitution effects. Wealth effects occur when prices change in a way that enables a consumer to purchase his utility maximizing bundle from last period and have money left over this period, or when he can no longer afford the bundle he purchased last period. Alternatively, a consumer's wealth changes when he can consume at a higher level of utility, or must consume at a lower level, with the same nominal budget. Substitution effects occur when a change in prices causes relative quantities of goods consumed to change. These effects must be estimated and backed out of observed outcomes to arrive at the desired downward sloping demand function. In this model, the auctioneer takes into account substitution and wealth effects when estimating demand functions.

The model incorporates diffusion processes for both technology and consumer awareness of new products. Diffusion in the model is conducted on an agent-to-agent basis, with each agent acting as a node in a dynamic communication network. Because of the heterogeneity of the agents, the means by which each agent, or node, communicates (sends, receives) and processes information varies. The variation between agents in communication and processing of information is controlled by agent parameters, which are set at model initialization and vary according to random distributions.

In the current version of the model consumers have preferences defined by Constant Elasticity of Substitution (CES) utility functions: $U_i(x_{i1}, \ldots, x_{in}) = (a_{i1}x_{i1}^r + \ldots + a_{in}x_{in}^r)^{1/r}$ where U_i represents utility of consumer i and x_{ij} represents the quantity of good j he consumes. The parameter r determines the extent to which consumers consider products to be substitutes for each other. Several familiar utility functions can be represented using the CES form with different values of r; for example, if r=1 we get linear indifference curves (perfect substitutes), if $r \to 0$ we get Cobb-Douglas utility, and if $r \to -\infty$ we get Leontief utility (perfect complements).

The CES form has been widely used in studies of consumer behavior. It gives qualitatively reasonable results in a variety of settings and has several convenient computational features. We chose the CES form because we are studying situations in which technology results in new products being substituted for old ones, and we want to be able to easily specify the extent to which products satisfy the same needs.

Consumers learn about new products either from a firm's advertisement or from another consumer via word of mouth. Upon learning of a new product the consumer transmits this new product information to other consumers with a certain probability each period, to a number of other consumers, and for a number of periods, all of which are parameters in the model. When consumers learn about a new product from a firm's advertisement, they are the first to know about it in the market, and they evaluate the product and determine their personal intensity coefficient, distributed randomly from the base intensity coefficient. The consumer will add the new product to his choice set and inform other consumers about the product with a certain probability each period, to a certain number of other consumers, and for a given number of periods.

If the consumer learns of a new product from another consumer, and has not learned about the product in a previous period, he will evaluate the product, establish his personal intensity coefficient, and add it to the list of products he may consume.

Firms in the model produce goods that they have the technology to produce and for which they think they will earn a profit. We are primarily interested in behavior of concentrated industries, so we assume there are a limited number of competing firms. Inter-firm behavior in concentrated industries (oligopoly) is richer and more interesting than in industries with many small firms (pure competition) or with only one firm (monopoly). We have tested the model under the assumption of pure competition and achieved reasonable results but will not discuss them further in this paper. The average number of firms used in our runs is 10, generally with a few dominant firms.

Firms have different cost structures of the form $C_{ij} = F_{ij} + MC_{ij}Q_j^2$ where C_{ij} is the cost to firm i to produce Q_j units of good j, F_{ij} is the fixed cost associated with this production, and MC_{ij} is a coefficient associated with variable cost. Fixed cost is needed to ensure that firms will shut down production at a low level of demand. The quadratic term represents inefficiency costs associated with high levels of utilization. For convenience we have not included a term linear in Q; this is equivalent to assuming that economies of scale exactly balance out unit input costs.

Revenue is price $\boldsymbol{P}$ = (p1, p2, …, pn) times (dot product) quantity $\boldsymbol{Q}$ = (x1, x2, …, xn). Price is estimated by an inverse demand function of the form $\boldsymbol{P}(\boldsymbol{Q}) = \boldsymbol{A} - \boldsymbol{B}*\boldsymbol{Q}$ where vector $\boldsymbol{A}$ and matrix $\boldsymbol{B}$ are estimated by a regression. Past total industry production of each good is common knowledge, and firms know their own past production, so they can compute an estimated residual inverse demand function for their own production in the current period. Residual demand is demand that is unsatisfied after consumers have purchased all output produced by a firm's competitors. It is worth noting that we are not assuming that demand is linear; indeed, we know that with CES utility demand will not be linear. Rather, we are assuming that firms in the model use linear functions to estimate unknown current period demand from observed previous period production/price outcomes.

Firms use Cournot conjectures to set their output, meaning that they take their competitors' output as fixed and maximize their profit under this assumption. This represents behavior that is neither rivalrous nor collusive. With Cournot conjectures firms recognize that their output will affect price, but they neither try to collude to raise price above this level nor try to undercut their competitors to gain market share. One of the important goals of the model is to enable us to study the effects of the competitive environment on technology innovation and adoption, and Cournot conjectures represent a base case for comparison. In future versions of the model we will use a method called "conjectural variation" to study different competitive environments. This method can be parameterized so that varying the parameter inputs changes the nature of competition between firms in a range from perfect collusion to extreme rivalry, with neutral Cournot conjectures separating the two. We will also include parameters that characterize product differentiation, so that we can model varying degrees of monopolistic competition. Our goal is to enable a firm to input paremeters that describe its competitors so that it can evaluate it various strategic alternatives.

Firms learn new technologies either from their own basic research efforts or by leakage from other firms. A firm's research effort discovers a new technology when two randomly generated keys match. One key per period is generated to represent (changing) consumer preferences, and a number of keys (0-n) are generated by each firm: the more resources a firm commits to research versus capacity expansion, the higher the expected number of keys), and the more keys generated by the firms, the higher the likelihood of a match with the consumer key. If a match occurs, the firm advertises its new product for exactly one period to a number of consumers proportional to the discovering firm's share of all goods produced. After a firm has discovered a technology (either through original research or via a leak of the new technology information from another firm), parameters determine the probability of accidentally leaking this information to other firms, the number of firms that may acquire the technology by leak, and the number of periods during which leaks may occur.

4 Model Results, Parameter Sensitivity Analysis, and Validation Results

We have studied how the average probability of consumers' and firms' diffusion parameters affect technological shifts. For each parameter set, we have used 30 runs to compute averages and confidence intervals. Graphs shown here do not include the display of confidence intervals to facilitate visualization. The time scale on the x axis for all graphs is 20 years. The graphs labeled with "Slow Rate of Diffusion" were created with firms' and consumers' average probability of diffusion = *0.01*. The ones labeled with "Fast Rate of Diffusion" depict output created with firms' and consumers' average probability of diffusion = *0.05.*

By changing just these two parameters by *5X*, we see in Figures 1 thru 16 the profound overall market effects of how quickly a new product is adopted, the way prices adjust to supply and demand, and how firms strategically influence the long-term market outcomes by their entry, exit, and output decisions. The colors of the curves correspond to different product's intensity coefficients. The blue (color 1) and red (color 2) products are the oldest products, with the lowest intensity coefficients. The next product adopted is green (color 3), with higher intensity coefficient than the red and blue, and the newest product adopted is purple (color 4), with the highest intensity coefficient.

Figures 1 and 2 show the diffusion of product adoption among consumers, gradual with the slow rate of consumer and technology diffusion parameters and rapid with the fast rate of consumer and technology diffusion parameters.

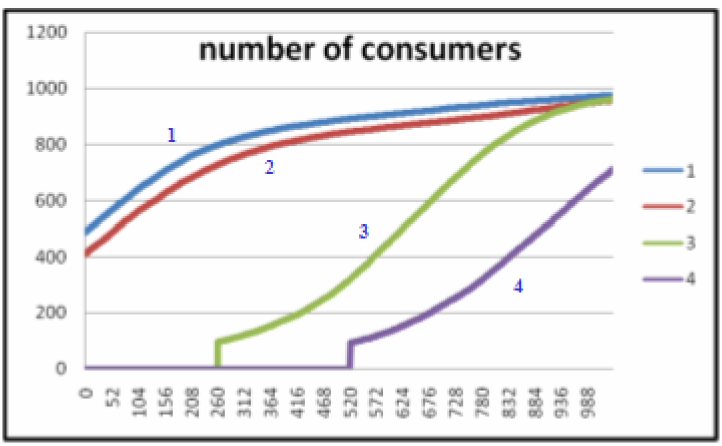


Fig. 1. Effect of Slow Rate of Diffusion on Product Adoption

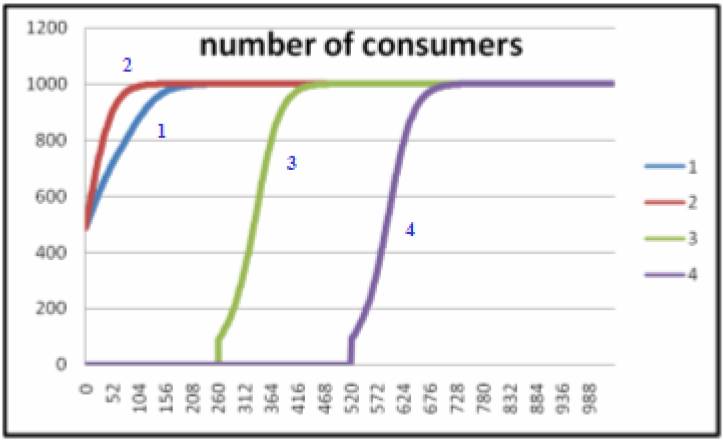


Fig. 2. Effect of Fast Rate of Diffusion on Product Adoption

Figures 3 and 4 show consumer quantity demand for each product. Under slow diffusion, the first new product, despite having better intensity coefficient than the older products, returns lower total utility for some time because the older products are cheaper. The slow diffusion gives firms that are producing the older products enough time to reduce marginal production costs so much that, for som time, the older products are actually slightly more desirable to consumers, reflected in their higher quantity demanded.

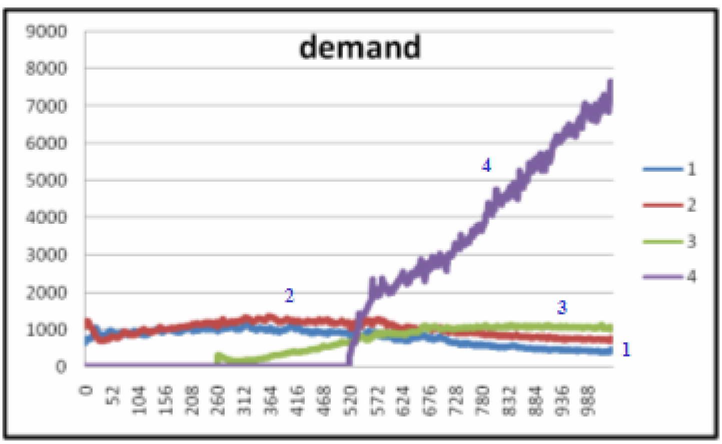


Fig. 3. Effect of Slow Rate of Diffusion on Demand

Under faster diffusion, however, we see the quantity demanded of new products eclipsing that of older products – a direct result of the lack of time for marginal production cost reductions and the price-adjusted utility principle built into each

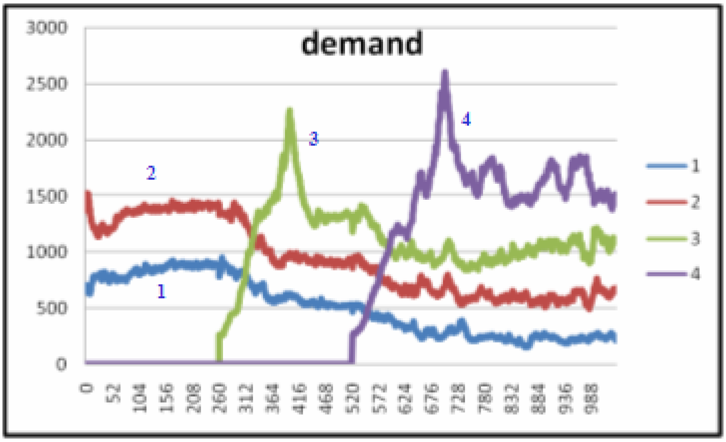


Fig. 4. Effect of Fast Rate of Diffusion on Demand

consumer's demand functions. In both cases, total quantity demand rises every period as reduced marginal production cost causes a corresponding reduction of the overall price level of each product.

The graphs on Figures 5 and 6 show consumer spending on each product, defined as *price*demand*. As expected, spending after a given level of adoption has been reached, increases with the intensity coefficient of the products. Changes in spending levels track consumer adoption of new products; i.e. spending changes when consumers are adopting new products and adjusting their demand functions. Spending stabilizes when no additional consumers are adopting new products. As expected, the rate of adoption varies proportionally to the diffusion rate.

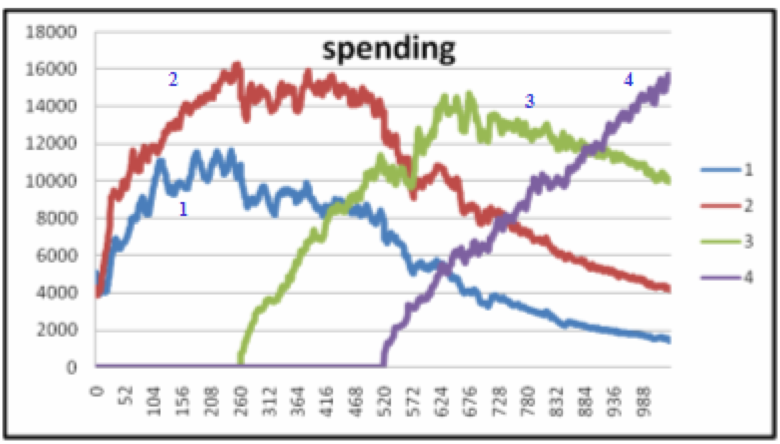


Fig. 5. Effect of Slow Rate of Diffusion on Consumer Spending

Price illustrates the short term dynamics of firm supply and consumer demand and offers a long-term roadmap for how marginal production cost declines contribute to overall lower price and how new product introductions disrupt stable market price trends. Figures 7 and 8 are best understood in conjunction with Figures 9 and 10 (number of producers). With slow consumer and technology diffusion, the number of producing firms grows slowly for two reasons: first, and most obvious, firms adopt new technology slowly; second, and less obvious, even firms that know about the new technology are less likely to develop a product because consumers are adopting the product slowly. So, firms projected profits are lower and the net result is that the number of firms producing the new product remains small for a longer time. Fewer firms mean less competition, which means more pricing power and thus higher prices. Price declines as the number of producer increases.

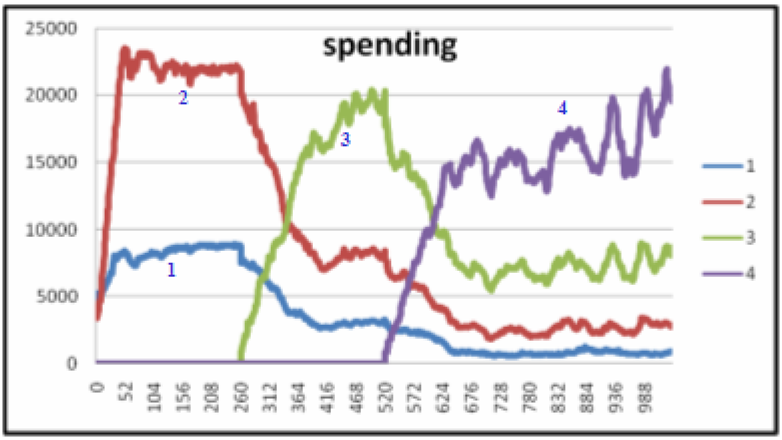


Fig. 6. Effect of Fast Rate of Diffusion on Consumer Spending

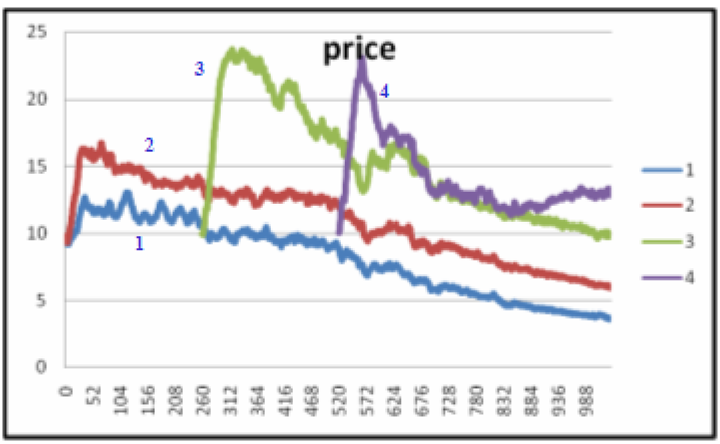


Fig. 7. Effect of Slow Rate of Diffusion on Price

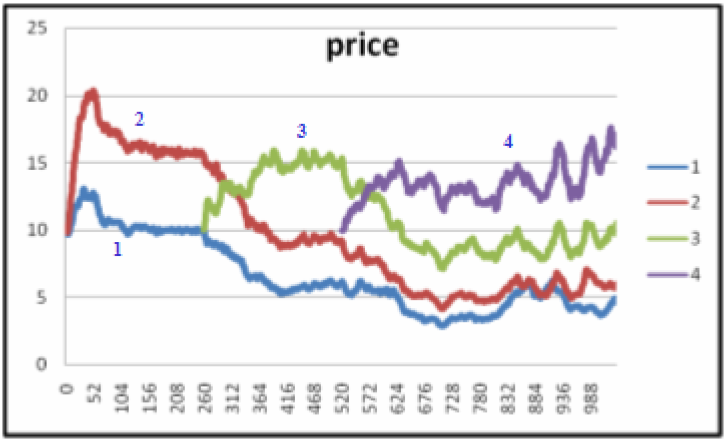


Fig. 8. Effect of Fast Rate of Diffusion on Price

In the faster diffusion case, more firms start producing the new product earlier, so the higher level of competition creates a more stable price-demand relationship.

Figures 9 and 10 show the number of firms producing each product. The time until firms *exit* older product markets takes much longer with the slower diffusion, shown on left, than with faster diffusion, shown on right. This again is directly attributable to the ability of firms to reduce marginal production costs at a competitive pace with a slow consumer adoption rate of new products.

Figures 11 and 12 show firm supply of each product; it approximates the dynamics seen in the consumer demand graphs, where supply of older products remains competitive in the slower diffusion case because of greatly reduced marginal production costs, and uncompetitive in the faster diffusion case.

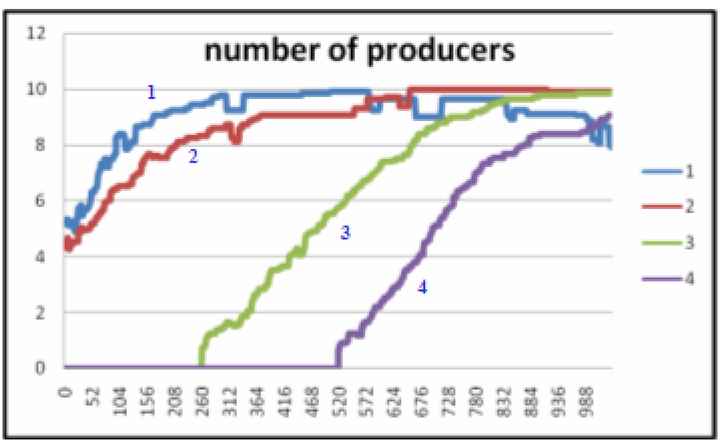


Fig. 9. Effect of Slow Rate of Diffusion on Number of Producers

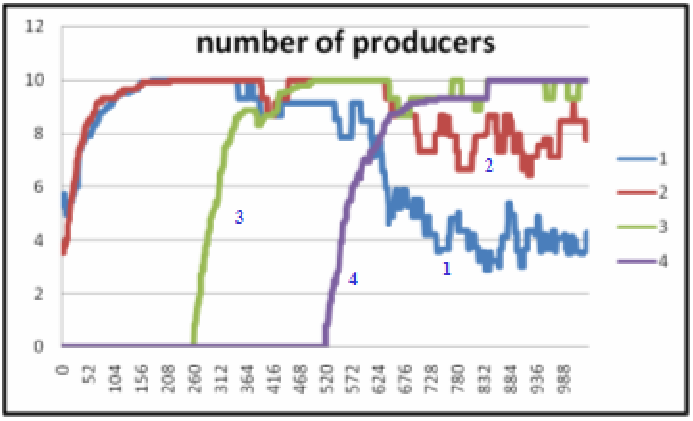


Fig. 10. Effect of Fast Rate of Diffusion on Number of Producers

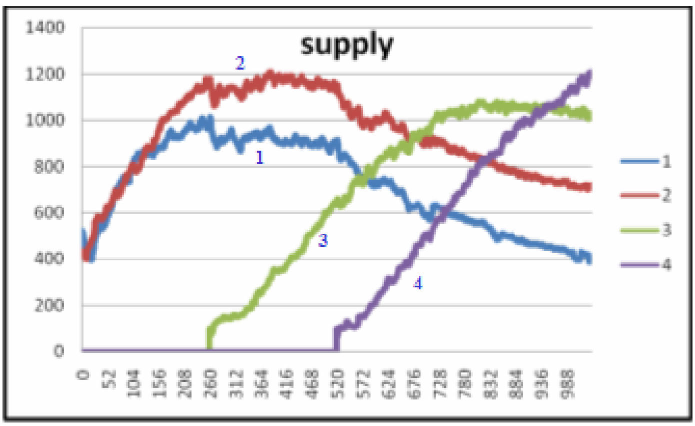


Fig. 11. Effect of Slow Rate of Diffusion on Supply

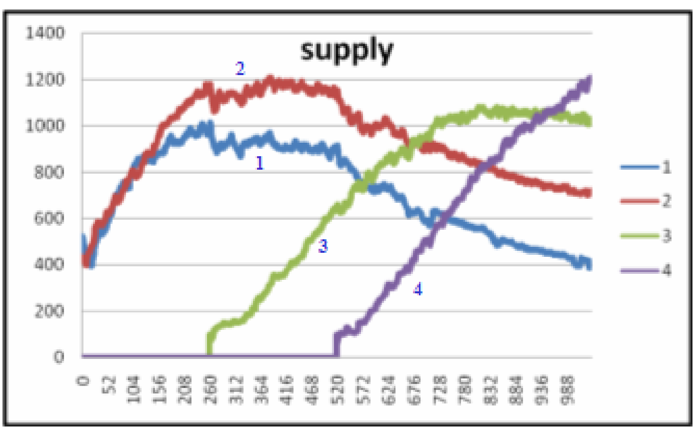


Fig. 12. Effect of Fast Rate of Diffusion on Supply

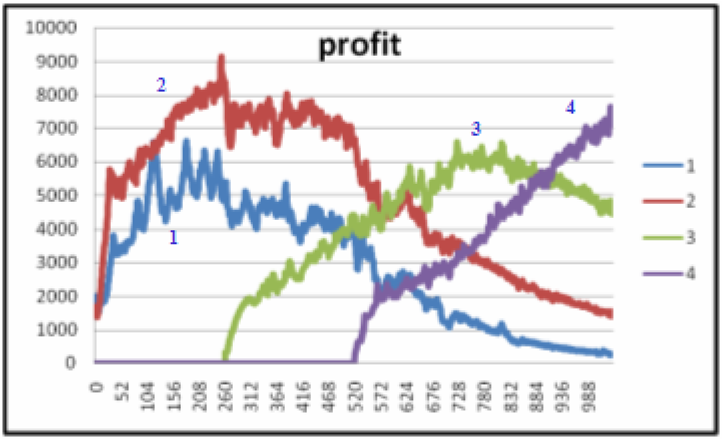


Fig. 13. Effect of Slow Rate of Diffusion on Profit

Figures 13 and 14 show the effect of diffusion on firm profits. It is interesting to note the small but noticeable spikes in profits that follow the exit of a firm from a product market; these can be seen in both graphs. Profits move roughly in line with aggregate consumer spending changes, so as spending habits change gradually in the slow diffusion case, so do profits; and as spending habits change rapidly in the fast diffusion case, so do profits.

Examining Figures 13 and 14 (profits) in conjunction with Figures 9 and 10 (number of producers) also yields useful insights. Although total profit is lower in the slow diffusion case, product 3 commands a higher proportion of total profit far longer than in the rapid diffusion case. The higher proportion of profits is a direct result not only of fewer competitors entering the market for product 3, but also of fewer producers in the market for product 4.

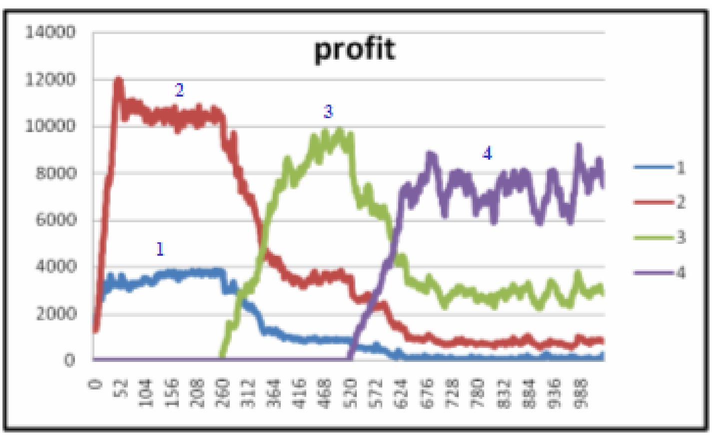


Fig. 14. Effect of Fast Rate of Diffusion on Profit

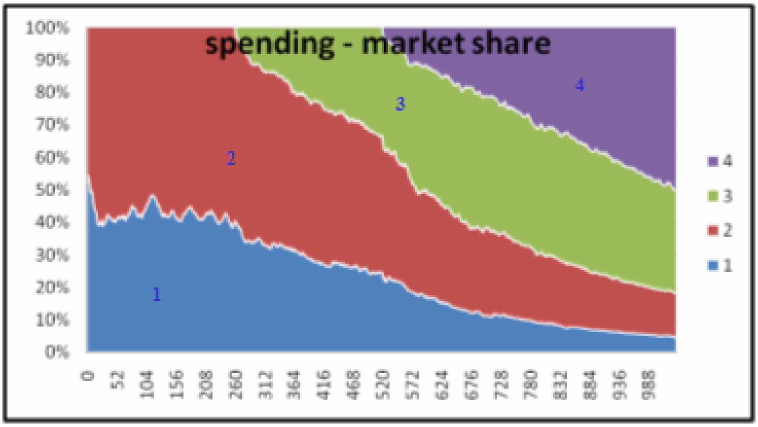


Fig. 15. Effect of Slow Rate of Diffusion on Market Share

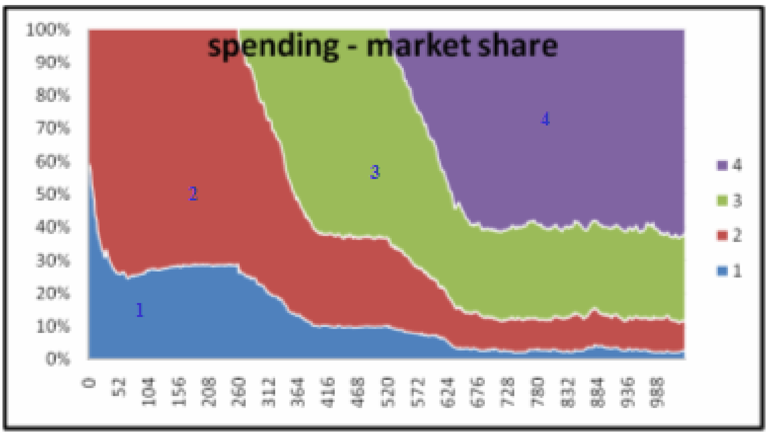


Fig. 16. Effect of Fast Rate of Diffusion on Market Share

The extended period of high profits dramatically illustrates the value to companies of aggressively defending patents and trade secrets (to slow technological diffusion in their product area) and of strong social institutions to protect intellectual property (to slow technology diffusion among firms producing near substitute products).

Figures 15 and 16 show the effect of diffusion on market share. Looking at these graphs, we can compare the rate at which the market share of products converges to a stable "steady state." The fast diffusion example illustrates that the relative spending level of a given product reflects its intensity coefficient.

Taking historical imaging industry data [20], [21] from 1994 to 2004 we arrived at an estimate for the diffusion coefficients for the imaging industry during the given time period of *0.05*. Using this coefficient, we averaged the results of 10 simulation runs using the same parameter set as the one measured/estimated for the industry data. Plotting market share figures for incumbent products film and film cameras versus the newly introduced digital cameras, we verified that the simulation results approximate well the actual data.

We analyzed the fit of the actual data with the simulated data, comparing the actual data series of each product to the simulated data series, arriving at $R^2 = 0.938$ for film, $R^2 = 0.959$ for the film camera, and $R^2 = 0.961$ for the digital camera.

5 Conclusion

Multi-agent simulation offers the opportunity to study complex systems at a level of detail not possible with other techniques. We have presented motivations and background of a research program that includes the modeling of a system of producers and consumers within a micro-economy, with the goal of revealing important characteristics and effects of technological innovation. We described the model developed to support the study of technological shifts, presenting preliminary results, parameter sensitivity analysis, and validation. The model currently supports variation of key properties of producers, consumers, and markets. We expect to make use of the model features to study the effect of different preferences among different wealth segments on different strategies for dealing with the shifts.

We have shown that the consumer and technology diffusion parameters strongly affect how quickly a new product is adopted and how firms strategically influence the long-term market outcomes by their entry, exit, and output decisions. Under slow diffusion, we showed that even if new products have higher intensity coefficient than the older product, quantity demanded of the older product can stabilize at a relatively high level. Under faster diffusion, we show that the quantity demanded of new products eclipses that of older products. In both cases, total quantity demand rises every period as reduced marginal production cost causes a corresponding reduction of the overall price level of each product. This result illustrates the importance of firms aggressively publicizing innovative products. If they do not capture market share before competitors have time to react, they may never achieve a dominant position. Important results were also presented on the effect of consumer diffusion on consumer spending, number of firms, and supply.

One interesting aspect of the micro-economy represented by this model relates to total profits from each product. In the slow diffusion case, the older product remains extremely profitable well after the new products have been introduced. The life cycle of each product is diffusion rate dependent on all the other products, and thus the investment decision of each producer in new plant capacity is also affected. The rate at which both consumers and producers learn about technology, not just adopt it, is critical.

By selecting simulation parameters that are similar to market parameters in the time period under study, we showed that the simulation results are similar to the actual industry data in the case of market adoption of digital vs. film photographic technology.

Future work will extend the model to include a number of additional parameters that further refine behavior and interaction of agents. In order to calibrate the parameters used, we plan to pursue empirical studies of consumer preferences. Extensions will also include parameters that identify economic factors such as industry concentration, rivalrous vs. co-operative firm behavior and effects of different cost structures. Extensions to many other economically important characteristics at the level of individual agents will follow. Much work also remains in validating the model and in sensitivity analysis in order to gain additional insight into technological shifts.

References

1. Solow, R.: A Contribution to the Theory of Economic Growth. Quarterly Journal of Economics 70, 65–94 (1956)
2. Mansfield, E.: Technical Change and the Rate of Imitation. Ecomometrica 29(4), 741–766 (1961)
3. Bass, F.A.: New Product Growth for Model Consumer Durables. Management Science 50(12), 1825–1832 (2004)
4. Stoneman, P.: The Economics of Technological Diffusion. Blackwell Publishers Ltd., Malden (2002)
5. Ireland, N., Stoneman, P.L.: Technological Diffusion, Expectations and Welfare. Oxford Economic Papers 38, 283–304 (2002)
6. Davies, S.: The Diffusion of Process Innovations. Cambridge University Press, Cambridge (1979)
7. Reinganum, J.: Market Structure and the Diffusion of New Technology. Bell Journal of Economics 12, 618–624 (1981)
8. Quirmbach, H.C.: The Diffusion of New Technology and the Market for an Innovation. RAND Journal of Economics 17, 33–47 (1986)
9. Fudenberg, D., Tirole, J.: Preemption and Rent Equalization in the Adoption of New Technology. Review of Economic Studies 52, 383–401 (1985)
10. Stoneman, P., Ireland, N.J.: The Role of Supply Factors in the Diffusion of New Process Technology. Economic Journal, Conference Papers 93, 65–77 (1983)
11. Silverberg, G., Dosi, G., Orsenigo, L.: Innovation, Diversity and Diffusion: A Self-Organization Model. Economic Journal 98(393), 1032–1054 (1988)
12. Nelson, R.R., Winter, S.G.: An Evolutionary Theory of Economic Change, Belknap, Cambridge, MA (1982)
13. Jonard, N., Yildizoglu, M.: Technological diversity in an evolutionary industry model with localized learning and network externalities. Structural Change and Economic Dynamics 9, 35–53 (1998)
14. Kwasnicki, W., Kwasnicka, H.: Market innovation, competition. Journal of Economic Behavior and Organization 19, 343–368 (1992)
15. Frenken, K.: History, state and prospects of evolutionary models of technical change: a review with special emphasis on complexity theory. In: Casti, J. (ed.) Complexity Focus, `http://www.complexityscience.org`
16. Dawid, H.: Agent-based models of innovation and technological change. In: Tesfatsion, L., Judd, K. (eds.) Handbook of Computational Economics, pp. 1237–1272. Elsevier, Amsterdam (2006)
17. Silverberg, G., Verspagen, B.: Evolutionary theorizing on economic growth. In: Kuusi, O. (ed.) Innovation Systems and Competitiveness. ETLA, Helsinki (1996)
18. Dawid, H., Reimann, M.: Evaluating Market Attractiveness: Individual Incentives vs. Industrial Profitability. Computational Economics 24, 321–355 (2005)
19. Windrum, P., Birchenhall, C.: Is product life cycle theory a special case? Dominant designs and the emergence of market niches through coevolutionary-learning. Structural Change and Economic Dynamics 9, 109–134 (1998)
20. Stanly, M.: Imaging Handbook (February 14, 2002)
21. J.P. Morgan – Precision Instruments Industry (June 25, 2002)

Beyond Accuracy. Reputation for Partner Selection with Lies and Retaliation

Isaac Pinyol[1], Mario Paolucci[2], Jordi Sabater-Mir[1], and Rosaria Conte[2]

[1] Artificial Intelligence Research Institute, Barcelona, Spain
[2] Institute for Cognitive Science and Technology, Rome, Italy

Abstract. In an unpredictable, heterogeneous world, intelligent agents depend on accurate social information; reputation, among the preeminent artifacts to transmit social evaluations, has been receiving growing attention by social scientists. A realistic description of reputation must include inaccurate information; in this paper, based on the distinction between image (agents' believed evaluation of a target) and reputation (circulating evaluation, without reference to the evaluation source), we model the spreading of information in a simple market with the presence of liars and the possibility of retaliation. While fear of retaliation inhibits the spreading of image, the detached character of reputation can be a cause of inaccuracy; The two forces could balance in different settings. In a set of simulations, with agents using the Repage platform for management of image and reputation, we compare the usage of image alone with the usage of both image and reputation. Reputation is shown to be preferable over image to allow for faster discover of scarce good sellers.

1 Introduction

In an unpredictable world, intelligent agents are shown to depend on accurate information [1,2,3,4] for acting adaptively. More specifically, they depend on accurate social information for interacting into a heterogeneous multiagent world. Memory of past experience is a precious source of information, but is usually acquired at own expenses. As obtaining experience may be fatal in a world of cheaters, agents depend on one another to indirectly acquire information for partner selection, before interacting with, and in order to avoid, the bad guys.

In the last ten years or so [5,6,7], the role of indirectly acquired social information has been appreciated by social scientists to a fairly realistic degree. Indeed, reputation has received a growing attention as network-based social evaluation [8]. However, by this means the rate of inaccurate information circulating in a multiagent system increases, and the question as to how put up with such inaccuracy starts to be posed. Stated otherwise, if reputation is a mechanism for finding out the material cheaters, i.e. nonreciprocators in exchange of material goods, how find out the informational cheaters, i.e. deceitful informants?

Solutions to this problem in the MAS field usually rely on learning strategies [9,10], by means of which agents more or less gradually discover trustable informers and gain accurate information. Applied to detection of informational

L. Antunes, M. Paolucci, and E. Norling (Eds.): MABS 2007, LNAI 5003, pp. 128–140, 2008.

cheating, learning techniques are rather more efficient than is the case with material cheating, since facing deception in reputation is less costly than trading with a gouger. However, this process results in shrieked social networks where agents get stuck, to the expenses of less experienced agents which are "new to the environment" [11]. But if by learning we mean feedback from experience that provides more accurate information, there are some distinct and interacting problems that need clarification and solution.

- On the side of the input, personal experience does not spare the costs of its acquisition: people do often pay dearly the privilege of telling who is a cheater and who is not!
- On the side of the output, once agents have got this precious information, why should they come to share it with others? In this sense, the idea of agents resorting to local and total beliefs [9] is somewhat naive.

These questions cannot be addressed by means of "try and test" techniques, but need theory-driven experimentation and implementation. Learning is not the key to solve any problem of adaptation: suffice to have a look at natural societies to find out that human agents neither always nor often converge on trustworthy informers, nor on rational expectations.

In this paper, we deal with deception by means of a theory of reputation, and closer to what is found to happen in natural societies [12,13]. We will tackle the issue by means of a reputation system (REPAGE, [14]), that can metabolyse inaccurate information rather than discard it thanks to a fundamental distinction between image (own opinion of a target) and reputation (others' opinion of a target). In substance, image is passed on as accurate evaluation, reputation is passed on as reported, hencefore not necessarily accurate evaluation (although in fact liars may choose either modality). The interplay between these two objects in the agents' minds and in social transmission allows both accurate and inaccurate information to spread over the population. We will simulate the effects of these assumptions over a simplified market, where buyers choose sellers of variable quality, endowed with a variable number of non-replenishing stock units.

Two assumptions about how REPAGE affects agents are made:

- In order to select information, agents are more conservative in image than reputation acceptance; in other words, they may accept as a reputation what they do not believe as an image.
- In order to avoid retaliation if found out to be inaccurate, agents transmit evaluations that they do not believe or are uncertain about, but only as reputation (others' opinion).

Thanks to image, incoming information affects agents' beliefs, and consequently partner selection, to a lower extent and in a more controlled way than it affects their message passing. Thanks to reputation, evaluations circulate into the system, providing continuous input to belief formation and revision. Reputation makes the system more dynamic and alert. Image makes it more selective and controlled. Reputation releases uncertain information, which might turn out to be more or less accurate, and which agents will check before using it for their own purpose. A system based on image only is expected to hinder the information flow to the benefit of accuracy; a reputation-prone system is expected to collapse under uncertainty and inaccuracy.

2 Theory Introduction

2.1 Image and Reputation

Our proposal is based on a model of imAGE, REPutation and their interplay developed in [12]. Although both are social evaluations, image and reputation are distinct objects. Image is a simple evaluative belief [15], it tells that the target is "good" or "bad" with respect to a norm, a standard, or a skill. Reputation is a belief about the existence of a communicated evaluation. Consequently, to assume that a target t is assigned a given reputation implies only to assume that t is reputed to be "good" or "bad", i.e., that this evaluation circulates, but it does not imply to share the evaluation.

To select good partners, agents need to form and update own social evaluations; hence, they must exchange evaluations with one another. If agents should transmit only believed image, the circulation of social knowledge would be bound to stop soon. On the other side, agents that believe all the informations that they receive would be no more autonomous; in order to preserve their autonomy, agents need to *decide* independently whether to share or not and whether to believe or not others' evaluations of a given target. Hence, they must

– Form both evaluations (image) and meta-evaluations (reputation), keeping distinct the representation of own and others' evaluations, before
– Deciding whether or not to integrate reputation with their own image of a target.

Unlike other current systems, in REPAGE reputation does not coincide with image. Indeed, agents can either transmit their own image of a given target, which they hold to be true, or report on what they have "heard" about the target, i.e. its reputation, whether they believe this to be true or not. Of course, in the latter case, they will neither commit to the information truth value nor feel responsible for its consequences. Consequently, agents are expected to transmit uncertain information, and a given positive or negative reputation may circulate over a population of agents even if its content is not actually believed by the majority.

To remark the difference between the effects of REPutation and imAGE, we will examine all the simulation scenarios in the rest of the paper under the two main experimental conditions

– L1, where there is only exchange of image between agents
– L2, where agents can exchange both image and reputation.

Note that while L1 is comparable with a large body of similar literature (ex. [16]), the introduction (L2) of reputation as a separate object in a simulative experiment will be presented in this paper for the first time.

3 The Repage Model and Architecture

Repage [14] is a computational system based on a cognitive theory of reputation [12] that proposes a fundamental distinction between image and reputation. The Repage architecture has three main elements, a memory, a set of detectors and the analyzer. The memory is composed by a set of references to the predicates hold in the main memory of the agent. Predicates are conceptually organized in levels and inter-connected. Each predicate that belongs to one of the main types (including image and reputation) contains a probabilistic evaluation that refers to a certain agent in a specific role. For instance, an agent may have an image of agent T (target) as a seller (role), and a different image of the same agent T as informant. The probabilistic evaluation consist of a probability distribution over the discrete sorted set of labels: {Very Bad, Bad, Normal, Good, Very Good}.

The network of dependences specifies which predicates contribute to the values of others. In this sense, each predicate has a set of precedents and a set of antecedents. The detectors, inference units specialized in each particular kind of predicate, receive notifications from predicates that changes or that appear in the system and uses the dependences to recalculate the new values or to populate the memory with new predicates.

Each predicate has associated a strength that is function of its antecedents and of the intrinsic properties of each kind of predicate. As a general rule, predicates that resume or aggregate a bigger number of predicates will hold a higher strength.

At the first level of the Repage memory we find a set of predicates not evaluated yet by the system. *Contracts* are agreements on the future interaction between two agents. Their result is represented by a *Fulfillment*. *Communications* is information that other agents may convey, and may be related to three different aspects: the image that the informer has about a target, the image that, according to the informer, a third party agent has on the target, and the reputation that the informer has about the target.

In level two we have two kind of predicates. *Valued communication* is the subjective evaluation of the communication received that takes into account, for instance, the image the agent may have of the informer as informant. Communications from agents whose credibility is low will not be considered as strong as the ones coming from well reputed informers. An *outcome* is the agent's subjective evaluation of a direct interaction, built up from a fulfillment and a contract.

At the third level we find two predicates that are only fed by valued communications. On one hand, a *shared voice* will hold the information received about the same target and same role coming from communicated reputations. On the other hand, *shared evaluation* is the equivalent for communicated images and third party images.

Shared voice predicates will finally generate *candidate reputation*; shared evaluation together with outcomes will generate *candidate image*. Newly generated candidate reputation and image aren't usually strong enough; new communications and new direct interactions will contribute to reinforce them until a threshold, over which they become full-fledged image or reputation. We refer to [14] for a much more detailed presentation.

From the point of view of the agent strucuture, integration with the other parts of our deliberative agents is strightforward. Repage memory links to the main memory of the agent that is fed by its communication and decision making module, and at the same time, this last module, the one that contain all the reasoning procedures uses the predicates generated by Repage to make decisions.

4 Description of the Experiment

We have designed the simulation experiment as the simplest possible setting where accurate information is a *commodity*, meaning that information is both valuable and scarce. Since the system will be used as a proof of concept, we will not ground it with micro or macro data, but we will instead use a simplified generic economic metaphor of an agent-based market setting with instability. This simplified approach is largely used in the field [16], both on the side of the market design and of the agent design. We release the simplification on the agent design while keeping it on the side of the market design.

The experiment includes only two kind of agents, the buyers and the sellers. All agents perform actions in discrete time units (turns from now on). In a turn, a buyer performs one communication request and one purchase operation. In addition, the buyer answers all the information requests that it receives.

Goods are characterized by an utility factor that we interpret as quality (but, given the level of abstraction used, could as well represent other utility factors as quantity, discount, timeliness) with values between 1 and 100.

Sellers are characterized by a constant quality and a fixed stock, that is decreased at every purchase; they are essentially reactive, their functional role in the simulation being limited to providing an abstract good of variable quality to the buyers. Sellers exit the simulation when the stock is exhausted or when for certain number of turn they do not sell anything, and are substituted by a new seller with similar characteristics.

The disappearance of sellers makes information necessary; reliable communication allows for faster discover of the better sellers. This motivates the agents to participate in the information exchange. In a setting with permanent sellers (infinite stock), once all buyers have found a good seller, there is no reason to change and the experiment freezes. With finite stock, even after having found a good seller, buyers, should be prepared to start a new search when the good seller's stock ends.

At the same time, limited stock makes good sellers a scarce resource, and this constitutes a motivation for the agents not to distribute information. One of the interests of the model is in the balance between these two factors.

There are four parameters that describe an experiment: the number of buyers NB, the number of sellers NS, the stock for each seller S, and the distribution of quality among sellers. In 2 we defined the two main experimental situations, L1 where there is only exchange of image, and L2 where both image and reputation are used.

4.1 Decision Making Module

In our experiments the decision making procedure is a key point that determines the performance of the whole system. From the seller side, this procedure is quite simple since they are limited to *sell* the products that buyers require and to disappear when the stock gets exhausted. From the point of view of the buyers, at each turn they have to ask one question to another buyer and buy some item from a seller. They may also answer a question from other buyers. Each of these actions requires the buyer to make some decisions:

Buying Action: In this action the question is: which seller should I choose? The Repage system provides information about image and reputation of each one of the sellers. The easiest option would be to pick the seller with *better* image, or (in L2) better reputation if image is not available. We set a threshold for an evaluation (actually, for its center of mass, see [14] for definitions) to be considered *good enough* to be used to make a choice. In addition, we keep a limited chance to explore other sellers, controlled by the system parameter *risk*[1]. Figures 1 and 2 describe the reasoning procedure that agents use to pick the seller in the situations L1 and L2 respectively. Notice that image has always priority over reputation, since image imply an acknowledge of the evaluation itself while reputation only an acknowledge of what is said.

Asking Action: As in the previous action, the first decision is the choice of the agent to be queried, and the decision making procedure is exactly the same than for choosing a seller, but dealing with images and reputation of the agents as informers (*informer image*) instead of as sellers. Figures 3 and 4 describe these procedures in situations L1 and L2 respectively.

Once decided who to ask, the kind of question must be chosen. We consider only two possible queries: Q1 - Ask information about a buyer as informer

1. *Candidate_Seller* := Select randomly one image's seller
2. If *Candidate_Seller* is empty or decided to risk then *Candidate_Seller* := select randomly one seller without image
3. Buy from *Candidate_Seller*

Fig. 1. Buying action: Decision procedure for L1

1. *Candidate_Seller* := Select randomly one good enough seller image.
2. If *Candidate_Seller* is empty then *Candidate_Seller* := select randomly one good enough seller reputation
3. if *Candidate_Seller* is empty or decided to risk then *Candidate_Seller* := select randomly one seller without image
4. Buy from *Candidate_Seller*

Fig. 2. Buying action - Decision procedure for L2

[1] Risk is implemented as a probability (typically between 5% and 15%) for the buyer to try out unknown sellers.

1. *Candidate_Informer* := Select randomly one *good enough* informer image
2. If *Candidate_Informer* is empty or decided to risk then *Candidate_Informer* := select randomly one buyer without image as informer
3. Ask to *Candidate_Informer*

Fig. 3. Asking action - Decision procedure for L1

1. *Candidate_Informer* := Select randomly one good enough informer image
2. If *Candidate_Informer* is empty then *Candidate_Informer* := select randomly one good enough informer reputation
3. if *Candidate_Informer* is empty or decided to risk then *Candidate_Informer* := select randomly one buyer without image as informant
4. Ask to *Candidate_Informer*

Fig. 4. Asking action - Decision procedure for L2

(basically, how honest is buyer X as informer?), and Q2 - Ask for some good or bad seller (for instance, who is a good seller, or who is a bad seller?). Notice that this second possible question does not refer to one specific individual, but to the whole body of information that the queried agent may have. This is in order to allow for managing large numbers of seller, when the probability to choose a target seller that the queried agent have some information about would be very low. The agent will ask one of these two questions with a probability of 50%. If Q1 is chosen, buyer X as informer would be the less known one, that is, the one with less information to build up an image or reputation of it.

Answering Action: Let agent S be the agent asking the question, R the agent being queried. Agents can lie, either because they are cheaters or because they are retaliating. When a buyer is a cheater whatever information being answered is changed to its opposite value. Retaliation is accomplished by sending inaccurate information from the point of view of the sender (for instance, sending "Idontknow" when really it has information, or simply giving the opposite value)when R has a bad image of S as informer. In L1 retaliation is done by sending an "Idontknow" message even when R has information. This avoids possible retaliation from S since an "Idontknow" message do not imply any commitment. If reputation is allowed, (L2) retaliation is accomplished in the same way as if the agent were a liar, but converting all image to send into reputation, in order to avoid as well possible retaliation from S.

Because of the fear of retaliation, sending an image will take place only when an agent is very secure of that evaluation, in the sense of the REPAGE *strength* parameter included in every evaluation. This is yet another parameter *thStrength*, that allows to implement *fear of retaliation* in the agents. Notice that if *thStrength* is zero, there is no fear since whatever image formed will be a candidate to be sent, no matter its strength. As we increase *thStrength*, agents will become more conservative, less image and more reputation will circulate in the system.

1. ImgX := Get image of agent X as informant
2. if ImgX exists and strength(ImgX) >= *thStrength* then send ImgX to agent S, END
3. else send "Idontknow" to agent S, END
4. if ImgX does not exist then send "Idontknow" to agent S

Fig. 5. Answering Q1 - Decision procedure for agent R, L1

1. ImgX := Get image of agent X as informant
2. if ImgX exists and strength(ImgX) >= *thStrength* then send ImgX to agent S, END
3. else convert ImgX to RepX and send RepX to S, END
4. if ImgX does not exist then RepX := Get reputation of agent X as informant
5. if RepX exists then send RepX to S, END
6. if RepX does not exist then send "Idontknow" to agent S

Fig. 6. Answering Q1 - Decision procedure for agent R, L2

1. IG := Get good enough images of sellers; IB := Get bad enough images of sellers
2. if IG is not empty then CandImage := Pick one randomly from IG1
3. else if IB is not empty then CandImage := Pick one randomly from IB
4. if CandImage is not empty and strength(CandImage) >= *thStrength* then sent CandImage to S, END
5. if CandImage is not empty and strength(CandImage) < *thStrength* then send "Idontknow" to S, END
6. if CandImage is empty then send "Idontknow" to agent S

Fig. 7. Answering Q2 - Decision procedure for agent R, L1

1. IG := Get good enough images of sellers; IB := Get bad enough images of sellers
2. RG := Get good enough reputations of sellers;
3. RB := Get bad enough reputations of sellers
4. if IG is not empty then CandImage := Pick one randomly from IG
5. else if IB is not empty then CandImage := Pick one randomly from IB
6. if CandImage is not empty and strength(CandImage) >= *thStrength* then send CandImage to S, END
7. if CandImage is not empty and strength(CandImage) < *thStrength* then convert CandImage to CandRep and send it to S, END
8. if RG is not empty then CandRep := Pick one randomly from RG
9. else if RB is not empty then CandRep := Pick one randomly from RB
10. if CandRep is not empty send CandRep to S, END
11. if CandRep is empty send "Idontknow" to S, END

Fig. 8. Answering Q2 - Decision procedure for agent R, L2

Figures 5 and 6 describe the decision making process that agents use to answer the question Q1 in situations L1 and L2 respectively. In figures 7 and 8 is shown the processes agents use to answer Q2 in situations L1 and L2 respectively.

5 Research Questions

We have two experimental conditions, with image only (L1) and with both Image and Reputation (L2). We will explore several values of the parameters in order to show how and where there is an advantage in using reputation. The hypotheses are:

H1. Initial advantage: L2 shows an initial advantage over L1, that is, L2 grows faster.
H2. Performance: L2 performs better as a whole, that is, the average quality at regime is higher than L1. Note that to obtain this result we are hardwiring a limitation in image communication, based on the theory that foresees large amounts of retaliation against mistaken image communications but not on the reputation side.
H3. Cheaters effect: with a high number of cheaters, L2 tends to drop to L1 levels.

There are other hypotheses that we do not treat yet. They regard the relationship between efficiency, fairness, and the presence of cheaters. Actually, these are not yet really formulated as hypotheses but as questions.

H3.B. Cheater effect: are cheaters always detrimental to the system? In particular, is the performance of the system always decreasing in the number of cheaters?
H4. Fairness: what is the order relation between L1 and L2 in terms of fairness? For the calculation of fairness we can use simple measures of distribution in quality (averaged, accumulated).
H5. Cheaters' advantage: do the cheaters effectively reach a significant advantage from their behavior?

6 Simulation Runs and Result Analysis

We have run simulations to examine the relationship between L1 and L2 with different levels in some parameters. The stock is fixed at 50, the number of buyers to 25, and the number of sellers at 100. We included cheaters as well with percentages of 0%, 25% and 50%.

We run the simulations for 100 steps, and we explored the variation of good and bad sellers, from the extreme case of 1% of good sellers and 99% of bad sellers(A1), going trough 5% good sellers and 95% bad sellers(A2),and 10% good sellers and 90% bad sellers(A3), and finally, to another extreme where we have 50% of good sellers and 50% of bad sellers(A4). For each one of these conditions and for every situation (L1 and L2) we run 10 simulations. In figures we present the accumulated average per turn of a concrete condition in both situations, L1 and L2.

6.1 Experiments without Cheaters

In figure 9 we show results for the four conditions without cheater; both hypotheses H1 and H2 are verified. With the increase of good sellers the difference between L1 and L2 gets smaller, until in condition A4 there is no difference. Because of the good sellers increase, they can be reached by random search and the necessity of communicating social evaluations decreases. In the extreme condition A4, statistically every buyer would find a good seller at the second turn

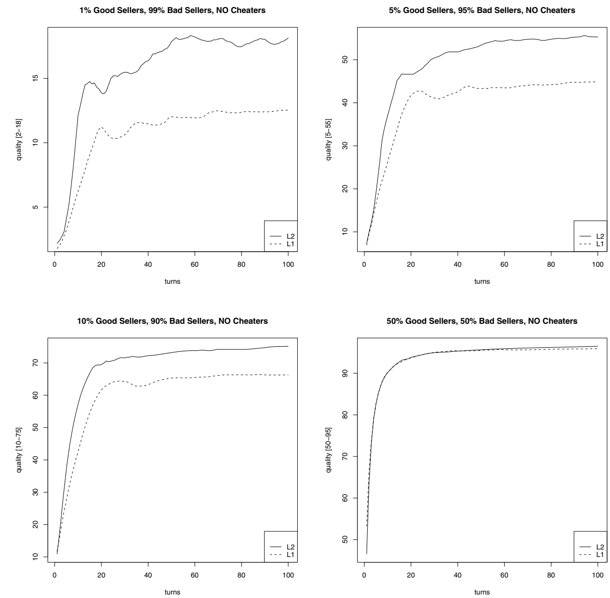


Fig. 9. Accumulated average quality per turn without cheaters in condition A1, A2, A3 and A4 respectively

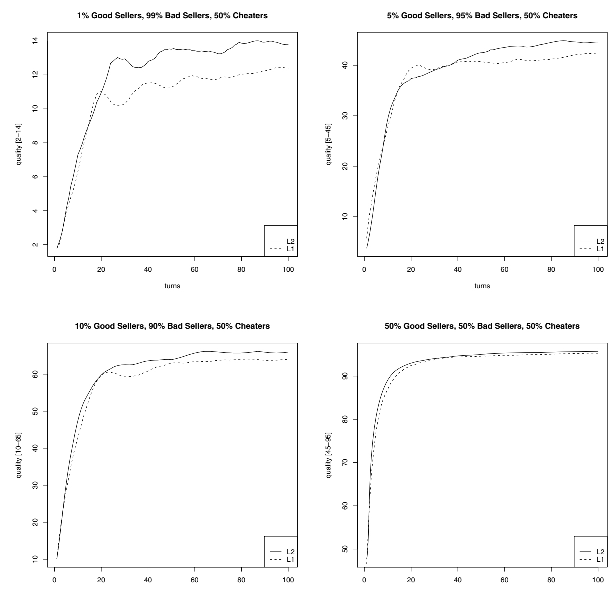


Fig. 10. Accumulated average quality per turn with cheaters in condition A1, A2, A3 and A4 respectively

(there is a probability of 50% to get one in one turn). In condition A3 the probability to reach one good seller per turn is 0.1, then, in 10 turns approximately every one would reach a good one. In L1 the amount of useful communications (different from *"Idontknow"*) is much lower that in L2, due to the fear of retaliation that governs this situation. In conditions where communication is not important, the difference between the levels disappears.

6.2 Experiments with Cheaters

Figure 10 shows results for conditions A1, A2, A3 and A4 with 50% of cheaters. The increased amount of false information produces a bigger impact in situations and conditions where communication is more important. Quality reached in L1 shows almost no decrease with respect to the experiment without cheaters, while L2 quality tends to drop to L1 levels, supporting the hypotheses H3. This shows how the better performance of L2 over L1 is due to the larger amount of information that circulates in L2.

7 Conclusions and Future Work

The results we obtained indicate that using reputation and image instead of only image improve the average quality reached in the whole system. We consider these results as a proof of concept about the usefulness of the reputation model [12], under a set of assumption that we discuss with a perspective on future works:

Retaliation: The presence of retaliation is crucial for the present results. We claim that the fact of communicating a social evaluation that is an image implies a commitment from the source agent. From the theory, image is a believed evaluation (see section 2) and sharing it implies that the source agent is informing of what he/she *accepts* as true. This differs from reputation, since accepting a reputation do not imply to accept the nested belief. Because of that, sharing what an agent acknowledge as a reputation does not imply a personal commitment. Here we assume that the personal commitment associated to image transmission exposes the agent to a possible retaliation if inaccurate information was sent.

As a future work we will study in more depth the effect of cheaters over the whole system, considering the presence of a norm that prescribes agents to tell the truth, and the reputation mechanism as a social control artifact to exclude agents that do not follow the norm. This is where the hypothesis H3, H4 and H5 we described in section 5 take relevance.

Communication and Reputation: There is no reputation without communication. Therefore, scenarios with lack of communication or few exchange of information cannot use reputation. However, in virtual societies with autonomous agents that have the freedom to communicate, that need to cooperate and have the right to choose partners, we consider that keeping a separation between image and reputation considerably increases the circulation of information and improves the performance of their activities. In our experiments, even when there is no panelization for direct interactions and considering at each turn only one possible question, the introduction of this difference already improves the average quality per turn. In scenarios where *quality* is scarce and agents are completely autonomous is where this social control mechanism make the difference.

Decision Making Procedure: The decision making schema we implemented (see section 4) determines the performance of the system. In fact, this is where the agent is taking advantage of the distinction between Image and Reputation [12]. We will elaborate on this distinction, possibly reformulating it in terms of textitmeta decision making, a very promising future line of work to better ground and exploit the Image and Reputation artifacts.

Acknowledgments

This work was supported by the European Community under the FP6 programme (eRep project CIT5-028575, OpenKnowledge project FP6-027253 and SocialRep project MERG-CT-2005/029161) and the project Autonomic Electronic Institutions (TIN2006-15662-C02-01), and partially supported by the Generalitat de Catalunya under the grant 2005-SGR-00093. Jordi Sabater-Mir enjoys a Ramon y Cajal contract from the Spanish Government.

References

1. Laffont, J.J.: The Economics of Uncertainty and Information, 4th edn. MIT Press, Cambridge (1993)
2. Kudenko, D., Kazakov, D.: Adaptive Agents and Multi-Agent Systems. Springer, New York (2003)
3. Kluegl, F., Bazzan, A.: A case study on the role of information for implicit coordination. In: AAMAS 2004 (2004)
4. Molnr, K.: Learning with expert advice. JEL (2006)
5. Hirshleifer, J.: The private and social value of information and the reward to inventive activity. American Economic Review 61(4), 561–574 (1971); In: P.A. Diamond, M. Rothschild. (ed.). Uncertainty in Economics: Readings and Exercises. Academic Press, San Diego (reprinted, 1989) ISBN: 0122148517
6. Carter, J., Bitting, E., Ghorbani, A.: Reputation formalization for an information sharing multiagent system. Computational Intelligence 18(4), 514–515 (2002)
7. Yamagishi, T., Matsuda, M., Yoshikai, N., Takahashi, H., Usui, Y.: Solving the lemons problem with reputation. An experimental study of online trading (2004)
8. Weesie, J., Buskens, V., Raub, W.: The management of trust relations via institutional and structural embeddedness. In: Doreian, P., Fararo, T. (eds.) The Problem of Solidarity: Theories and Models. Gordon and Breach 1998, Amsterdam, pp. 113–138 (1990)
9. Yu, B., Singh, M.P.: Detecting deception in reputation management. In: AAMAS 2003: Proceedings of the second international joint conference on Autonomous agents and multiagent systems, pp. 73–80. ACM Press, New York (2003)
10. Fullam, K., Barber, K.S.: Learning trust strategies in reputation exchange networks. In: AAMAS 2006 (2006)

11. Pradchayakool, P., Daengdej, J., Supannika, K.: An approach for detecting deception in agents. In: Aerospace Conference IEEE (2006)
12. Conte, R., Paolucci, M.: Reputation in artificial societies: Social beliefs for social order. Kluwer Academic Publishers, Dordrecht (2002)
13. Conte, R., Paolucci, M.: Social cognitive factors of unfair ratings in reputation reporting systems. In: Proceedings of the IEEE/WIC International Conference on Web Intelligence - WI 2003, pp. 316–322 (2003)
14. Sabater, J., Paolucci, M., Conte, R.: Repage: Reputation and image among limited autonomous partners. Journal of Artificial Societies and Social Simulation 9(2) (2006)
15. Miceli, M., Castelfranchi, C.: The Role of Evaluation in Cognition and Social Interaction. In: Human cognition and agent technology. Benjamins, Amsterdam (2000)
16. Sen, S., Sajja, N.: Robustness of reputation-based trust: boolean case. In: AAMAS 2002: Proceedings of the first international joint conference on Autonomous agents and multiagent systems, pp. 288–293. ACM Press, New York (2002)

Author Index