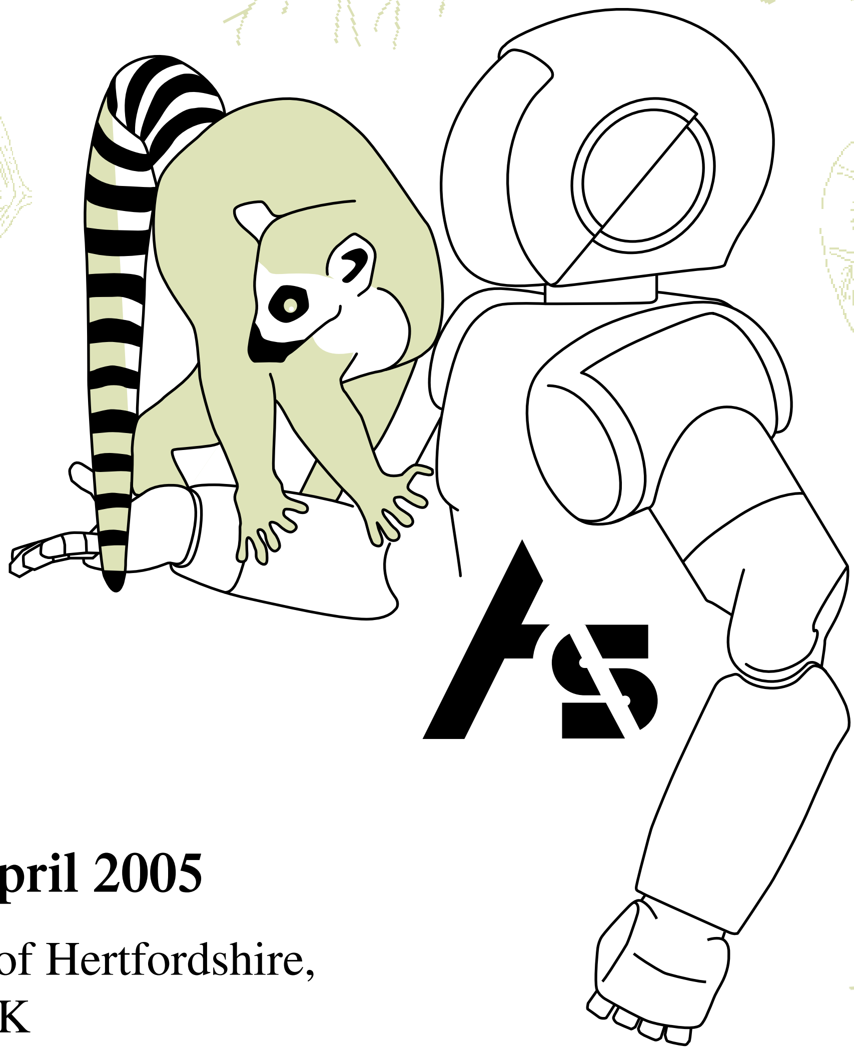
AISB'05: Social Intelligence and Interaction in Animals, Robots and Agents

Proceedings of the Joint Symposium on Socially Inspired Computing



12 - 15 April 2005

University of Hertfordshire,
Hatfield, UK

SSAISB 2005 Convention

AISB UH EPSRC Engineering and Physical Sciences Research Council

AISB’05 Convention

Social Intelligence and Interaction in Animals, Robots and Agents

12-15 April 2005
University of Hertfordshire, Hatfield, UK

Proceedings of the

Socially Inspired Computing Joint Symposium

Memetic Theory in Artificial Systems & Societies
Emerging Artificial Societies
Engineering with Social Metaphors

Published by

The Society for the Study of Artificial Intelligence and the Simulation of Behaviour
www.aisb.org.uk

Printed by

The University of Hertfordshire, Hatfield, AL10 9AB UK
www.herts.ac.uk

Cover Design by Sue Attwood

ISBN 1 902956 48 4

AISB'05 Hosted by

The Adaptive Systems Research Group
adapsys.feis.herts.ac.uk

The AISB'05 Convention is partially supported by:

Engineering and Physical Sciences Research Council

The proceedings of the ten symposia in the AISB’05 Convention are available from SSAISB:

Second International Symposium on the Emergence and Evolution of Linguistic Communication (EELC'05)
1 902956 40 9

Agents that Want and Like: Motivational and Emotional Roots of Cognition and Action
1 902956 41 7

Third International Symposium on Imitation in Animals and Artifacts
1 902956 42 5

Robotics, Mechatronics and Animatronics in the Creative and Entertainment Industries and Arts
1 902956 43 3

Robot Companions: Hard Problems and Open Challenges in Robot-Human Interaction
1 902956 44 1

Conversational Informatics for Supporting Social Intelligence and Interaction - Situational and Environmental Information Enforcing Involvement in Conversation
1 902956 45 X

Next Generation approaches to Machine Consciousness: Imagination, Development, Intersubjectivity, and Embodiment
1 902956 46 8

Normative Multi-Agent Systems
1 902956 47 6

Socially Inspired Computing Joint Symposium (Memetic theory in artificial systems & societies, Emerging Artificial Societies, and Engineering with Social Metaphors)
1 902956 48 4

Virtual Social Agents Joint Symposium (Social presence cues for virtual humanoids, Empathic Interaction with Synthetic Characters, Mind-minding Agents)
1 902956 49 2

Table of Contents

The AISB'05 Convention
Social Intelligence and Interaction in Animals, Robots and Agents

Above all, the human animal is social. For an artificially intelligent system, how could it be otherwise?

We stated in our Call for Participation "The AISB'05 convention with the theme *Social Intelligence and Interaction in Animals, Robots and Agents* aims to facilitate the synthesis of new ideas, encourage new insights as well as novel applications, mediate new collaborations, and provide a context for lively and stimulating discussions in this exciting, truly interdisciplinary, and quickly growing research area that touches upon many deep issues regarding the nature of intelligence in human and other animals, and its potential application to robots and other artefacts".

Why is the theme of Social Intelligence and Interaction interesting to an Artificial Intelligence and Robotics community? We know that intelligence in humans and other animals has many facets and is expressed in a variety of ways in how the individual in its lifetime - or a population on an evolutionary timescale - deals with, adapts to, and co-evolves with the environment. Traditionally, social or emotional intelligence have been considered different from a more problem-solving, often called "rational", oriented view of human intelligence. However, more and more evidence from a variety of different research fields highlights the important role of social, emotional intelligence and interaction across all facets of intelligence in humans.

The Convention theme *Social Intelligence and Interaction in Animals, Robots and Agents* reflects a current trend towards increasingly interdisciplinary approaches that are pushing the boundaries of traditional science and are necessary in order to answer deep questions regarding the social nature of intelligence in humans and other animals, as well as to address the challenge of synthesizing computational agents or robotic artifacts that show aspects of biological social intelligence. Exciting new developments are emerging from collaborations among computer scientists, roboticists, psychologists, sociologists, cognitive scientists, primatologists, ethologists and researchers from other disciplines, e.g. leading to increasingly sophisticated simulation models of socially intelligent agents, or to a new generation of robots that are able to learn from and socially interact with each other or with people. Such interdisciplinary work advances our understanding of social intelligence in nature, and leads to new theories, models, architectures and designs in the domain of Artificial Intelligence and other sciences of the artificial.

New advancements in computer and robotic technology facilitate the emergence of multi-modal "natural" interfaces between computers or robots and people, including embodied conversational agents or robotic pets/assistants/companions that we are increasingly sharing our home and work space with. People tend to create certain relationships with such socially intelligent artifacts, and are even willing to accept them as helpers in healthcare, therapy or rehabilitation. Thus, socially intelligent artifacts are becoming part of our lives, including many desirable as well as possibly undesirable effects, and Artificial Intelligence and Cognitive Science research can play an important role in addressing many of the huge scientific challenges involved. Keeping an open mind towards other disciplines, embracing work from a variety of disciplines studying humans as well as non-human animals, might help us to create artifacts that might not only do their job, but that do their job right.

Thus, the convention hopes to provide a home for state-of-the-art research as well as a discussion forum for innovative ideas and approaches, pushing the frontiers of what is possible and/or desirable in this exciting, growing area.

The feedback to the initial Call for Symposia Proposals was overwhelming. Ten symposia were accepted (ranging from one-day to three-day events), organized by UK, European as well as international experts in the field of Social Intelligence and Interaction.

- Second International Symposium on the Emergence and Evolution of Linguistic Communication (EELC'05)
- Agents that Want and Like: Motivational and Emotional Roots of Cognition and Action
- Third International Symposium on Imitation in Animals and Artifacts
- Robotics, Mechatronics and Animatronics in the Creative and Entertainment Industries and Arts
- Robot Companions: Hard Problems and Open Challenges in Robot-Human Interaction
- Conversational Informatics for Supporting Social Intelligence and Interaction - Situational and Environmental Information Enforcing Involvement in Conversation
- Next Generation Approaches to Machine Consciousness: Imagination, Development, Intersubjectivity, and Embodiment
- Normative Multi-Agent Systems
- Socially Inspired Computing Joint Symposium (consisting of three themes: Memetic Theory in Artificial Systems & Societies, Emerging Artificial Societies, and Engineering with Social Metaphors)
- Virtual Social Agents Joint Symposium (consisting of three themes: Social Presence Cues for Virtual Humanoids, Empathic Interaction with Synthetic Characters, Mind-minding Agents)

I would like to thank the symposium organizers for their efforts in helping to put together an excellent scientific programme.

In order to complement the programme, five speakers known for pioneering work relevant to the convention theme accepted invitations to present plenary lectures at the convention: Prof. Nigel Gilbert (University of Surrey, UK), Prof. Hiroshi Ishiguro (Osaka University, Japan), Dr. Alison Jolly (University of Sussex, UK), Prof. Luc Steels (VUB, Belgium and Sony, France), and Prof. Jacqueline Nadel (National Centre of Scientific Research, France).

A number of people and groups helped to make this convention possible. First, I would like to thank SSAISB for the opportunity to host the convention under the special theme of *Social Intelligence and Interaction in Animals, Robots and Agents*. The AISB'05 convention is supported in part by a UK EPSRC grant to Prof. Kerstin Dautenhahn and Prof. C. L. Nehaniv. Further support was provided by Prof. Jill Hewitt and the School of Computer Science, as well as the Adaptive Systems Research Group at University of Hertfordshire. I would like to thank the Convention's Vice Chair Prof. Chrystopher L. Nehaniv for his invaluable continuous support during the planning and organization of the convention. Many thanks to the local organizing committee including Dr. René te Boekhorst, Dr. Lola Cañamero and Dr. Daniel Polani. I would like to single out two people who took over major roles in the local organization: Firstly, Johanna Hunt, Research Assistant in the School of Computer Science, who efficiently dealt primarily with the registration process, the AISB'05 website, and the coordination of ten proceedings. The number of convention registrants as well as different symposia by far exceeded our expectations and made this a major effort. Secondly, Bob Guscott, Research Administrator in the Adaptive Systems Research Group, competently and with great enthusiasm dealt with arrangements ranging from room bookings, catering, the organization of the banquet, and many other important elements in the convention. Thanks to Sue Attwood for the beautiful frontcover design. Also, a number of student helpers supported the convention. A great team made this convention possible!

I wish all participants of the AISB’05 convention an enjoyable and very productive time. On returning home, I hope you will take with you some new ideas or inspirations regarding our common goal of understanding social intelligence, and synthesizing artificially intelligent robots and agents. Progress in the field depends on scientific exchange, dialogue and critical evaluations by our peers and the research community, including senior members as well as students who bring in fresh viewpoints. For social animals such as humans, the construction of scientific knowledge can't be otherwise.

Beppu, Japan.

Dedication:

I am very confident that the future will bring us increasingly many instances of socially intelligent agents. I am similarly confident that we will see more and more socially intelligent robots sharing our lives. However, I would like to dedicate this convention to those people who fight for the survival of socially intelligent animals and their fellow creatures. What would 'life as it could be' be without 'life as we know it'?

Kerstin Dautenhahn

Professor of Artificial Intelligence,
General Chair, AISB'05 Convention *Social Intelligence and Interaction in Animals, Robots and Agents*

University of Hertfordshire
College Lane
Hatfield, Herts, AL10 9AB
United Kingdom

Symposium Preface
Socially Inspired Computing Joint Symposium

SYMPOSIUM OVERVIEW

Many social scientists are thinking about social phenomena as emergent properties of complex adaptive processes. Neither determined by the individual behaviour nor social level structures, social phenomena are seen as emerging from the interaction of the two over time. One way to understand such phenomena is with the use of computer simulation and experimentation (often using agent-based modelling).

In tandem with these developments computer scientists are required to understand and engineer ever more complex, distributed and loosely coupled systems (such as the internet). In these types of systems (such as multi-agent systems and peer-to-peer systems) the individual sub-systems interact to form an artificial social system with all the concomitant benefits and problems.

The Socially Inspired Computing Symposium brings together those working in these areas to explore algorithms producing novel emergent social phenomena. Such work can benefit both the understanding and engineering of artificial and human social systems.

The Symposium comprises three one day themes:

Day 1: Memetic Theory in Artificial Systems and Societies - focusing on novel computational models of culture using memes.
Day 2: Emerging Artificial Societies - focusing on the role of emergence in artificial social systems.
Day 3: Engineering with Social Metaphors - focusing on applying socially inspired methods to engineering next generation information systems.

ORGANISING COMMITTEE

Bruce Edmonds (Manchester Metropolitan University, UK)
Nigel Gilbert (University of Surrey, UK)
Steven Gustafson (Nottingham University, UK)
David Hales (Bologna University, Italy)
Natalio Krasnogor (Nottingham University, UK)

Memetic Theory in Artificial Systems and Societies

in the

Socially Inspired Computing Joint Symposium

Theme Preface
Memetic Theory in Artificial Systems and Societies (METAS)

Natalio Krasnogor
University of Nottingham
School of Computer Science & IT
natalio.krasnogor@nottingham.ac.uk

Steven Gustafson
University of Nottingham
School of Computer Science & IT
steven.gustafson@nottingham.ac.uk

ABSTRACT

Memetic Theory in Artificial Systems and Societies (METAS) is the first edition of a series of international symposia dedicated to qualitative and quantitative aspects of memetic research as applied to artificial (and natural) systems and societies. This symposium will bring together researcher working at the cutting-edge of memetic theory, cultural algorithms and the transmission of culture-like information as applied to artificial systems and societies. METAS aim is to promote multidisciplinary studies and the best science on memetics.

INTRODUCTION

Since Dawkins inception in 1976 of the "meme" concept, we have witnessed enormous advances in computational and communication technologies, not least the creation and popularisation of the Internet. These computational and communication advances allow researchers to simulate large and complex systems of interactive agents in scales not dreamt-of a short time ago. At the same time, these same resources represent sophisticated evolving computational substrates in which artificial societies (could) exist and where the science of memetics can be tested, developed and exploited.

The science of memetics encourages a common framework where cultural evolution and the transmission of culture-like information in artificial systems and societies can be studied. Some of the themes we would like to see covered in the METAS series are:

- Fundamental concepts on memetics and theoretical frameworks for Memetics (e.g., evolutionary, cognitive, societal and computational mechanisms, etc.)
- Memetics as an evolutionary model of information transmission
- Qualitative and Quantitative issues of memetics in artificial and natural societies (e.g. the impact of memes in the individual VS the society, etc.)
- Computer simulations of memetics systems and dynamics
- The memetics nature of information processing in networks (in general) and the Internet (in particular)
- The memetics of software evolution
- Memetics simulations in economy, marketing, policy-making, conflict resolution, game playing
- Memetics in artificial and natural problem solving, software engineering and multi-agent systems
- Requirements for effective memetics systems (computational substrates, communication mechanisms, etc.).

This symposium series will provide a unique opportunity for researchers in artificial intelligence, artificial life, robotics, cognitive science, biologist, social sciences, political studies and distributed systems engineering to interact with memetic scientist and to share a forum for discussion. The symposium will also serve as a common publication outlet for interdisciplinary research in these areas.

PAPER SUMMARIES

We are pleased to include in this years Memetic Theory in Artificial Systems and Societies symposium six exciting papers which represent a broad scientific agenda which ranges and inter-weaves operational definitions of memetics with robotics, network flow models to distributed evolutionary processes in the Internet and stigmergetic multi-agent systems.

In *Operationalization of Meme Selection Criteria: Methodologies to Empirically Test Memetic Predictions*, Chielens and Heylighen briefly review recent attempts to make memetic theory more quantitative and predictive rather than only qualitative and metaphoric. This report on an internet-based pilot study serves as a solid proof of concept for the potential in their approach for operationalizing the concept of meme selection criteria.

In the paper *Simulation Models for Biological and Cultural Evolution*, Fog argues that both genetic and cultural models have a built-in complexity which renders them more suitable to simulation models rather than analytical ones. Moreover, it is argued in the paper that in the case of cultural evolution this problem is exacerbated by the fact that cultural processes are even more complex than genetic ones, thus making genetic analytical model unsuitable for cultural processes. The paper draws various conclusions regarding the construction of models of cultural systems.

The paper *A Dynamic Network Flow Theoretical Model of Information Diffusion* by P.T. Breznay presents a conceptual framework, based on dynamic network flow theory, which can be used to model various diffusion mechanisms on a network of interconnected nodes that can act either as senders or receivers of information. The model, which uses coupled differential equations, is applied to a series of paradigmatic network topologies where the effects of diffusion processes are investigated.

The paper by W.B. Langdon, *Pfeiffer – A Distributed Open-ended Evolutionary System*, describes the implementation of an internet-based interactive evolutionary algorithm which is used to, in the jargon of R.Dawkins, produce designoids. Designoids are objects that seem to have been rationally design while in fact they have been evolved. Langdon describes the potential for open-ended evolution and its connections with cultural systems.

Priesterjahn, Goebels and Weimers *Sigmergetic Communication for Cooperative Agent Routing in Virtual Environments* investigates the advantages and disadvantages of local versus global knowledge repositories in scenarios where agents can exchange information and their success depends on effective communication and coordination strategies. The authors report that, although a global knowledge repository produces more robust behaviour, local knowledge is sufficient for effective survival strategies to emerge.

In their paper *Towards the Emergent Memetic Controlof a Module Robot Weimer*, Priesterjahn and Goebels employ a memetics as information carrier metaphor for the control of a self-reconfigurable robot. They implement both declarative and procedural knowledge as memes, and they show that an imitation of behaviour based strategy can robustly selfreconfigure a robot into a target shape.

PROGRAMME COMMITTEE

We would like to express our gratitude to the programme committee members:

- Yaneer Bar-Yam - New England Complex Systems Institute, Boston, USA
- Mark Bedau - Editor in Chief of Artificial Life Journal, USA
- Elhanan Borenstein - Dept. of Computer Science, Tel-Aviv University, Israel
- Larry Bull - School of Computer Science, Univ. of the West of England, UK
- Agner Fog - Engineering College of Copenhagen, Denmark
- Liane Gabore - Dept. of Psychology, Univ. of CA, Berkeley, USA
- Nigel Gilbert - Dept. of Sociology, Univ. of Surrey, UK
- William Hart - Sandia National Laboratories, USA
- Eytan Ruppin - Dept. of Computer Science, Tel-Aviv University, Israel
- Sorin Solomon - RACAH Institute of Physics, Hebrew University, Israel
- Jim Smith - University of the West of England, UK

METAS was part of the joint-symposium Socially Inspired Computing, along with the two symposia Emerging Artificial Societies and Engineering with Social Metaphors, in the 2005 AISB Convention Social Intelligence and Interaction in Animals, Robots and Agents.

Pfeiffer – A Distributed Open-ended Evolutionary System

W. B. Langdon*
*Computer Science, University of Essex
Colchester CO4 3SQ, UK
W1angdon@essex.ac.uk

Abstract

Pfeiffer contains a population of fractals which has been evolving continuously for more than three years. The animations are developed from embryos using a Lindenmayer grammar (L-System). These open generative representations potentially allow gene duplication and the evolution of higher order genetic operators and might be a step towards the emergence of social intelligence in swarms of artificial life (alife) agents. The fitness function is simply do the snowflake patterns appeal to the users: interactive evolution (IEC). To this end, images are placed in animated snow globes (computerised snowstorms) by world wide web (www) browsers (Netscape, Mozilla, Internet Explorer, Firefox, etc.) anywhere on the planet. More than 600 people have used http://www.cs.ucl.ac.uk/staff/W.Langdon/pfeiffer.html.

1 Introduction

For more than three years we have been running an experiment in distributed open-ended interactive evolution in which small local populations within each user's web browser communicate via Javascript with a central server holding a variable sized global population (see Figure 2). (Initial results were reported in Langdon (2004a).) Pfeiffer is intended to show the feasibility of evolving agents on many small computers running across the Internet under the user's actions as a fitness measure. The agents are intended to be attractive and therefore they are animated in a snowstorm. Their form is given by a D0L deterministic context free L-system Pruskinkiewicz and Lindenmayer (1990) (see Figure 1), whose initial seed is a Koch fractal snow flake.

L-systems have the advantage over traditional programming in that they are inherently parallel. This is analogous to growing plant tissue (for which they first used to model) where each cell grows and divides in parallel with its neighbours and like DNA strands where, in principle, all genes can be expressed simultaneously. Karl Sims was perhaps the first person to combine L-systems with interactive evolution, e.g. Sims (1991).

The next section describe the evolutionary L-system. Section 3 summarises its usage (more details are given in Langdon (2004a) and Langdon (2004b)) while section 4 considers what lessons can be drawn. The penultimate section (5) discusses where evolutionary agents might lead us. We conclude, in Section 6.

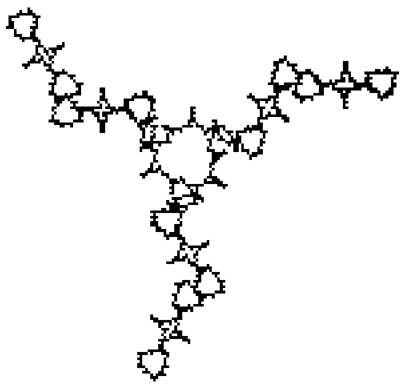


Figure 1: Example L-system fractal pattern. The picture is the phenotype developed from the 435^{th} genotype (seed) saved by users in the global population. The seed defines the L-system's initial grammar symbol as `F++F++F++F++F` and the replacement rule as `F`$\Rightarrow$`FF+FF--F`. It also specifies that start symbol be expanded four times.

2 How Pfeiffer Works

Pfeiffer (cf. Figures 2 and 3) evolves agents and displays them moving in two dimensions across the screen of a world wide web (www) browser. The visual phenotype of each agent is given by a Lindenmayer (L-system) grammar. As the agents are moved or tumble across the screen they are subject to random encounters and changes which may effect their grammar. Each time the grammar is changed the agent's new shape is drawn on the screen. The user can save pretty shapes and delete ugly ones.

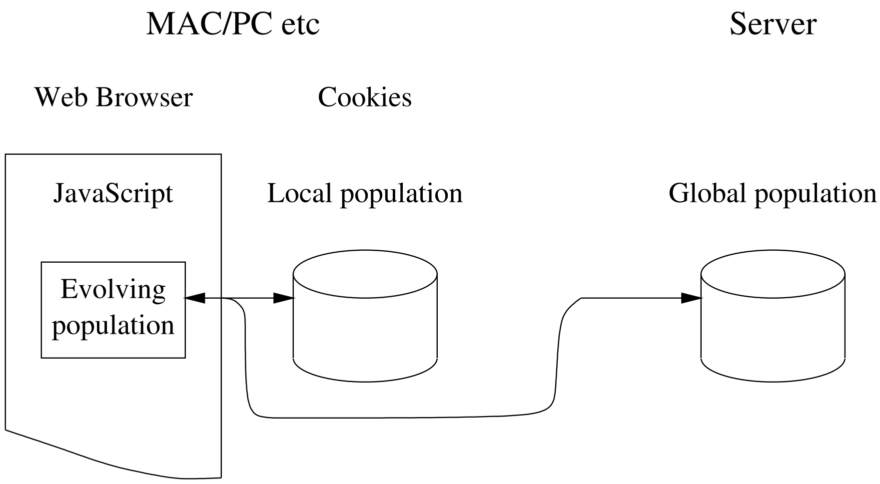


Figure 2: Overview of Pfeiffer. The user interacts via HTML and Javascript run in their web browser (left hand side). Initial seeds (chromosomes) are either retrieved from earlier runs via cookies or down loaded across the Internet via cgi programs from the global population. After evolution in the user's browser, the user may save new seeds both locally (as cookies) and in the global population.

2.1 Global and Local Populations

The active local population is stored in Javascript objects within the user's web browser. However these are lost when a new HTML page is selected. Therefore, "cookies" (if they are enabled) are used to provide off line storage of the local population.

Each time the Pfeiffer web page is down loaded, the initial value of each agent's chromosome is read from the corresponding cookie. However, if there is no cookie, the initial chromosome is down loaded from the global population across the network.

2.2 User Interaction

The primary goal of the user intervention is to use the user to provide selection pressure to drive the evolution of the agents. Passing the mouse over an agent causes its menu to be displayed. A text field allows the user to name the agent. While the pull down menu (see Figure 4) confirms the agent's identity and allows the user to: save the agent, make a copy of it (both automatically give it high fitness), delete it and close the menu. Naming an agent makes it easier for the user to track the agent he has evolved using "top ten" and "Hall of Fame" web pages, An agent "saved" by the user is stored in its cookie and appended to the global population. Once in the global population, the agent can be down loaded by other users and so distributed across the world. Cloning an agent causes an additional copy of the agent to be stored in the local population. This will often require the deletion of another, low fitness, agent. These user initiated actions exert selection pressure on the local and global populations.

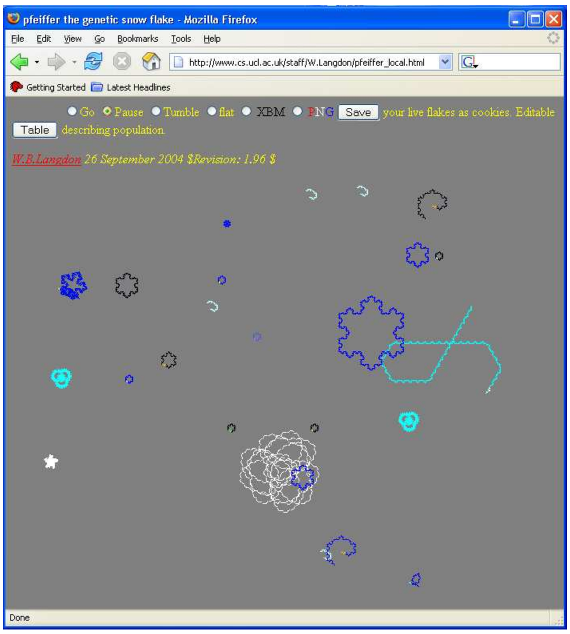


Figure 3: View of evolutionary arena as seen by user

In addition to deciding life and death, the user can influence which agents mate. Using the mouse, an agent can be picked up and moved into the path of another agent. As with saving and cloning, moving an agent implies the user prefers it and it is given high

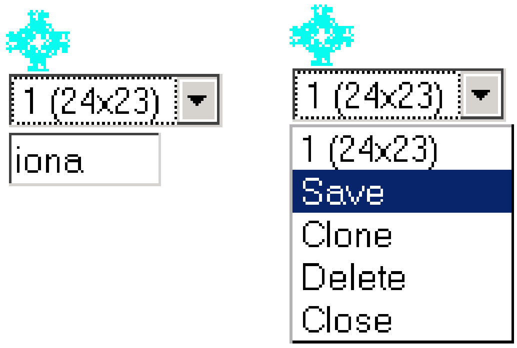


Figure 4: Example menu. Left hand, allows user to change agent's name. Right, pull down menu, allows user to save, copy or delete agent.

fitness, making it very likely to mate with the next mature agent it meets.

2.3 Generating the Phenotype

The system is able to display the results of arbitrary L-systems. In the original system (and even today in some browser) this is beyond Javascript. Therefore it was necessary to generate the graphics on a server and down load them into the user's browser (see Figure 5). In this mode of operation, each new seed is passed to the server. It is interpreted as a Lindenmayer grammar. This generates a series of drawing instructions, which are immediately obeyed. The resulting picture is compressed and converted to .GIF format and passed back to the user's browser for display. Because of the data compression, this takes only a few seconds. However the delay could cause problems due to the agent's genotype and phenotype becoming out of step Langdon (2004a). Therefore the new version of Pfeiffer processes L-systems and graphics generation in the user's browser. However both systems are active (for compatibility with less able browsers).

2.4 Genetic Representation

Each agent seed is a variable length linear text string. The default seed grows into the Koch snow flake . The default seed is the 56 character string `v=60&str=F++F++F & it=2 & sc =5 & rules=('F','F-F++F-F')` (this can be replaced by the user).

The string is split by & characters into parameters. They are are processed left to right. Thus if any parameter is repeated, the second "gene" is "dominant".

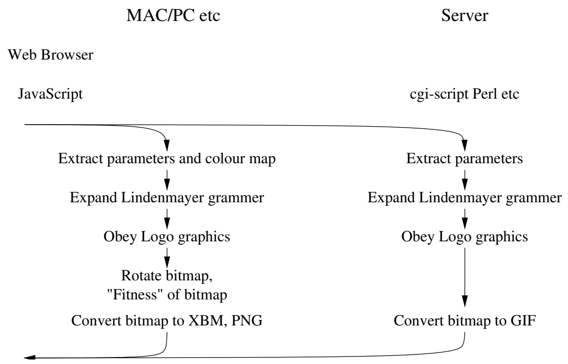


Figure 5: Mapping genotype to phenotype. In this development process the genotype (the L-system plus associated parameters) is converted to a graphic. The chromosome may either be passed to our server, interpreted and a .GIF file returned (right hand side) or interpreted locally (left). The local version avoids Internet delays, allows colour, and 3-D effects but is less portable.

Five parameters are recognised. They are `v` (angle), `str` (start string of grammar), `it` (depth of recursive expansion), `sc` (side, in units of 0.2 pixels) and `rules` (grammar replacement rules). Each substring formed by splitting the seed at the & is further split by =. If the first part of the substring exactly matches one of the parameter names then its value is set to the text between the first and second (if any) =. If a parameter is missing, its default is used. The defaults come from the Koch snow flake, they are `v`=60, `str`=`F++F++F`, `it`=2, `sc`=5 and `rules`=`('F', 'F-F++F-F')`. When `rules` is parsed characters such as `(` and `)` are removed. In our Koch example this means the single substitution rule is: F⇒`'F-F++F-F'`. The use of the defaults is effectively the same as if the default text where inserted at the start of every seed (albeit protected from genetic operators).

Once parameters have been decoded the L-system is interpreted. First the start string `srt` is expanded `it` times. At each step every character which matches the left hand symbol of a rule is replaced by the corresponding right hand side. Note any letter can potentially match a rule, not just those used by the turtle graphics, allowing indirect rules. The expansion yields a potentially very long string. To avoid infinite or very long recursions, time outs are applied.

The string is interpreted as a series of "turtle" drawing instructions. Except for 3-D instructions, predefined graphics and increasing the line width, all of the turtle instructions given in Pruskinkiewicz and Lindenmayer (1990) are supported. The graphic is displayed to the user moving across the main screen, see Figure 3. The local version, also supports a “tumbling” effect in the animation. On the server .GIF format is used. Some browsers (e.g. Mozilla) support colour PNG format files, whilst others (e.g. Microsoft) only support monochrome XBM format.

2.5 Example

Suppose the initial seed (chromosome), sent to the server, is the 59 characters `v=60 & str=F++F++F3t5F+r c sc=5 & rules = ('F','F-F++F-F')`. After removing spaces we are left with 51 characters This splits (at &) into two parameters `v`=60 and `str=F++F++F3t5F+rcsc=5`. Defaults are given for missing parameter values (`it` and `sc`), while `rules` is fixed. So the grammar is `v`=60, `str=F++F++F3t5F+rcsc`, `it`=2, `sc`=5 and `rules=('F','F-F++F-F')`. Note how the original value of `sc` had been corrupted but is restored by the default. The start string is expanded by the server twice (`it`=2) to give the final image.

Iteration		Size	Expansion	Line segments
0		3×3	`F++F++F3t5F+rcsc`	4
1		6×7	`F-F++F-F++F-F++F-F++F-F+ +F-F3t5F-F++F-F+rcsc`	16
2		15×17	`F-F++F-F-F-F++F-F++F-...`	64

3 Global Usage of Pfeiffer

From December 2001 to July 2004, excluding the author, 44,605 L-systems have been evolved and interpreted (3,522 locally and the rest and down loaded) by 1077 user sessions from 384 different sites. Usage of the system has been highly non-uniform. The length of individual sessions is approximately log-normal. However activity also varies in time reflecting the system’s development, particularly support for additional browsers (Mozilla, Netscape 6 and Microsoft, July 2003) and local interpretation and graphics generation (Jan 2004) (see Figures 7 and 8).

All the phenotypes created during a two month trial period are given in Langdon (2004b).

4 Discussion

As pointed out in (Hemberg et al. 2001, page 165) traditional interactive evolution often runs into the problem of human fatigue, i.e. people being unable to continue acting as the “fitness function” after a few

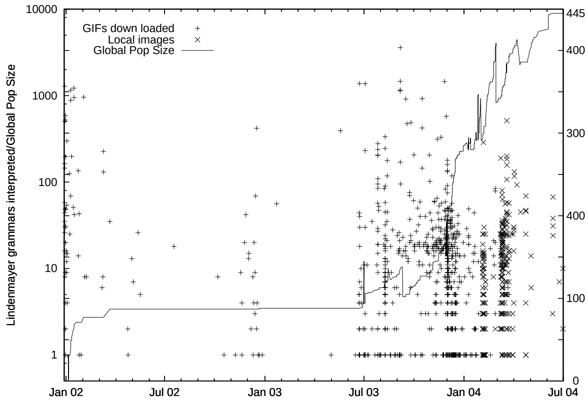


Figure 8: Usage of Pfeiffer. Crosses refer to creation of agents by individual users (note log scale), whilst the line shows the evolution of the global population (linear scale).

minutes. This normally severely limits both population size and number of generations. For example, in the approximately 250 papers described by Takagi (2001), typically populations contain only 9 or 16 individuals and no more than 10–20 generations are used. I.e. typically interactive evolutionary computation (IEC) runs have up to only 100 to 300 fitness evaluations. In contrast, the global population of Pfeiffer has been grown from about 100 to 514 today (January 2005) and 46,000 images have been presented to ≈ 600 people. Pfeiffer continues to attract users after more than three years of operation.

It appears Takagi (2001) most interactive evolutionary computation tries to make the user “optimise” and so tries to minimise varation between people. In contrast Pfeiffer tries to encourage variation, to produce many appealing patterns rather than a single optimal one. Again most IEC computer graphics (although by no means all) are static, while here they are animated.

The simple text string representation is certainly highly robust and flexible. Its compactness makes global distributed on line operation feasible.

The L-system (with supplied defaults and a robust interpreter) allows a huge degree of redundancy to evolve. The “fitness landscape” clearly contains a huge degree of “neutrality” Babajide et al. (1997); Shipman et al. (2000); Smith et al. (2002); Yu (2003) and evolution is using it.

L-systems readily allow evolution of many plane figures (but are not general purpose). Many new fractal like patterns have been readily evolved using them.

First parent

```
v =l-72&strF+'+F+4+F+F&&st F+2& 'c s 5tetulF+ =-F Fe ,- F&F(Fl+&=
```

Second parent

```
v =l=72&&strF '+F+4+F+F&&st F+2& 'c s 5tetulF+ =-F Fe,,- F&F(Fl++=
```

Offspring, replaces first parent

```
v =l-72&strF+'+F+4+F+F&&st F+2& 'c s | 5tetulF+ =|-F Fe ,- F&F(Fl+&=
```

Figure 6: Example crossover. Length of first parent 67, first cut point at 37, remove 10 characters, insert 11 characters. 68 characters in offspring.

Figure 7: Usage of Pfeiffer up to April 2004. Red lines connect each user's country to the central server. Heaviest use has been from UK, USA and Canada, but users have also come from the far and middle east, India, Europe, Latin American and South Africa.

5 Future: Breeding "Intelligent" agents

Our agents are very limited. We feel they need to be able to evolve to react to their environment. They need to be able to evolve to predict their environment. Of course this makes requirements of both the agent and the environment. Also, perhaps crucially, each agent needs to be able to effect the environment, and predict what those effects will do for it (and for others). While L-systems have been mainly used (as we have done here) to create static structures, they can describe networks. Those networks could contain sensory, processing and active elements Hornby and Pollack (2002) and/or use cultural evolution–mental simulation, imitation and knowledge-based operators–such as used by the vehicles of Gabora (1995). Gruau (1994) describes another indirect approach to evolving artificial neural networks (ANNs). While Stanley and Miikkulainen (2003) surveys developmental evolution in computer science.

There is a strand of thought in which intelligence came from a co-evolutionary struggle between members of the same species Ridley (1993). If true, can intelligence arise in isolated agents? Or are interacting/communicating agents needed?

A problem with simulated worlds has been hosting sufficient complexity so as to be challenging but still allowing agents to be able make predictions about what will happen next and what will happen to me or to others if I do this. The Internet hosts tides of data. This data is not random. It ought to be possible to harness it to give a suitable virtual environment.

We have fallen for the usual trap of constructing a two dimensional world (on the computer screen). However is there any hope of evolving artificial life (and thereby artificial intelligence) in two dimensions? Obviously three dimensions are sufficient but computer simulations offer many dimensions (N≫3).

6 Conclusions

Lindenmayer grammars can be used as the basis for a distributed interactive evolutionary system and produce interesting fractal like patterns. Many new patterns have been evolved Langdon (2004b), some exploiting the L-system to produce some regularities and re-use of motifs. It is feasible to represent individual agent's genetic material (seed/chromosome) with a variable length text string without defined fixed semantic fields and using crossover at the character level. The representation allows a huge degree of redundancy to evolve. The "fitness landscape" clearly contains a huge degree of "neutrality" and evolution is using it. This loose representation allows the location etc. (as well as the meaning) of the L-system to evolve. Gene duplication, translocation and other genetic operations could be supported by such a general representation.

In terms of harvesting spare CPU cycles, the project confirms it can be done using Javascript and user's web browser. The project does hint at some successes. World wide distributed evolution is clearly feasible. Perhaps more importantly one can recruit users (which are much more valuable than their CPUs) to assist in guided evolution. Finally animated tools are an attractive way to spread interest in artificial evolution, intelligence and life.

Acknowledgements

I would like to thank anonymous reviewers and Liane Gabora. I would like to thank Birger Nielsen for use of his perl Lindenmayer interpreter `http://www.246.dk/lsystems.html`, David Corney, Emma Byrne, Song Hu, Joao Oliveira, Mel Slater, Jorge Tavares, Ernesto Costa, Francisco Pereira, Sara Silva, Maarten Keijzer Jano van Hemert and several anonymous referees. Internet sites containing javascript source code and documentation have been very useful. Special thanks to `http://www.altan.hr/snow`, `http://devedge.netscape.com`, `http://www.mozilla.org`, `http://javascriptsource.com`, Roger E. Critchlow Jr. `pnglets`, Jukka Korpela `forms/extraspace.html`, Dan Steinman `mouseevents.html`, Peter-Paul Koch `js/events_compinfo.html`, Phil Reeve, and David Blackledge for `AnalogClock.html`.

Finally I would like to thank the users of Pfeiffer.

References

1 Aderonke Babajide, Ivo L. Hofacker, Manfred J. Sippl, and Peter F. Stadler. Neutral networks in protein space, a computational study based on knowledge-based potentials of mean force. *Folding & Design*, 2:261–269, 20 August 1997. ISSN 1359-0278.

2 Liane Gabora. Meme and variations: A computational model of cultural evolution. In Lynn Nadel and Daniel L. Stein, editors, *1993 Lectures in Complex Systems*, pages 471–485. Addison Wesley, 1995. URL `http://cogprints.org/531/00/mav.htm`.

3 Frederic Gruau. *Neural Network Synthesis using Cellular Encoding and the Genetic Algorithm.* PhD thesis, Laboratoire de l'Informatique du Parallilisme, Ecole Normale Supirieure de Lyon, France, 1994. URL `ftp://ftp.ens-lyon.fr/pub/LIP/Rapports/PhD/PhD1994/PhD1994-01-E.ps.Z`.

4 Martin Hemberg, Una-May O'Reilly, and Peter Nordin. GENR8 - A design tool for surface generation. In Erik D. Goodman, editor, *2001 Genetic and Evolutionary Computation Conference Late Breaking Papers*, pages 160–167, San Francisco, California, USA, 9-11 July 2001. URL `http://www.ai.mit.edu/projects/emergentDesign/genr8/lateGecco.pdf`.

5 Gregory S. Hornby and Jordan B. Pollack. Creating high-level components with a generative representation for body-brain evolution. *Artificial Life*, 8(3):223–246, 2002. ISSN 1064-5462. URL `http://www.demo.cs.brandeis.edu/papers/hornby_alife02.pdf`.

6 W. B. Langdon. Global distributed evolution of L-systems fractals. In Maarten Keijzer, Una-May O'Reilly, Simon M. Lucas, Ernesto Costa, and Terence Soule, editors, *Genetic Programming,* *Proceedings of EuroGP'2004*, volume 3003 of *LNCS*, pages 349–358, Coimbra, Portugal, 5-7 April 2004a. Springer-Verlag. URL `http://www.springerlink.com/openurl.asp?genre=article&issn=0302-9743&volume=3003&spage=349`.

7 W. B. Langdon. Lindenmayer systems fractals evolved by Pfeiffer 10 September – 9 November 2003. Research Note RN/04/13, University College, London, 19 July 2004b. URL `ftp://cs.ucl.ac.uk/genetic/papers/rn0413.pdf`.

8 Przemyslaw Pruskinkiewicz and Aristid Lindenmayer. *The Algorithmic Beauty of Plants*. Springer-Verlag, 1990.

9 Matt Ridley. *The Red Queen, Sex and the Evolution of Human Nature*. Penquin, 1993. ISBN 0140167722.

10 Rob Shipman, Mark Shackleton, and Inman Harvey. The use of neutral genotype-phenotype mappings for improved evolutionary search. *BT Technology Journal*, 18(4):103–111, October 2000. ISSN 1358-3948. URL `ftp://ftp.cogs.susx.ac.uk/pub/users/inmanh/bt.pdf`.

11 Karl Sims. Artificial evolution for computer graphics. *ACM Computer Graphics*, 25(4):319–328, July 1991. URL `http://web.genarts.com/karl/papers/siggraph91.html`. SIGGRAPH '91 Proceedings.

12 Tom Smith, Phil Husbands, Paul Layzell, and Michael O'Shea. Fitness landscapes and evolvability. *Evolutionary Computation*, 10(1):1–34, Spring 2002. URL `http://www.cogs.susx.ac.uk/users/toms/Papers/Smith_EvolutionaryComputation2002.pdf`.

13 Kenneth O. Stanley and Risto Miikkulainen. A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130, 2003. URL `http://nn.cs.utexas.edu/downloads/papers/stanley.alife03.pdf`.

14 Hideyuki Takagi. Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation. *Proceedings of the IEEE*, 89(9):1275–1296, September 2001. ISSN 0018-9219. Invited Paper.

15 Tina Yu. Neutral evolution in evolutionary computation. In James Foster, editor, *GECCO–2003 Genetic and Evolutionary Computation Conference: Tutorial Program*, pages 627–643, Chicago, 13 July 2003.

Operationalization of Meme Selection Criteria: Methodologies to Empirically Test Memetic Predictions

Klaas Chielens
Ecco,Vrije Universiteit Brussel
Pleinlaan 2, 1050 Brussels, Belgium
Klaas.chielens@vub.ac.be

Francis Heylighen
Ecco,Vrije Universiteit Brussel
Pleinlaan 2, 1050 Brussels, Belgium
fheyligh@vub.ac.be

Abstract

This paper reviews a number of recent approaches to put memetics to the test of quantitative measurability. The focus is on the selection criteria for the spreading of memes put forward by Heylighen (1997), which include utility, novelty, simplicity, coherence, authority and proselytism. The general hypothesis is that memes scoring higher on these criteria will survive longer and be more prevalent than others. This can be tested by checking which story elements best survive a chain of person-to-person transmissions ("Chinese whispers" game), by simulating the cognitive and social processes that determine this differential survival and spread, and by correlating the score on the selection criteria with the actual frequency with which a meme is encountered. In a pilot study using an Internet survey, this method was applied specifically to virus hoaxes, which can be seen as paradigmatic examples of clearly delimited, self-reproducing messages.

1 Introduction

In 1976 Dawkins coined the term 'meme' to denote the cultural equivalent of the biological gene, i.e. an information pattern that is being copied from person to person. Examples of memes are jokes, ideas, traditions, rumors, fashions and chain letters. Each of these information systems spreads by means of communication from one to several carriers. Thus, a successful meme can be compared to a cultural virus that "infects" a growing group of hosts. Over the past decade, an increasing number of publications has been devoted to memetics (e.g. Blackmore, 2000 & Aunger, 2001), proposing explanations for phenomena from viral marketing to consciousness and religion.

However, the memetic approach has been criticized by many authors (Aunger 2001). Two major shortcomings can be pointed out: 1) it is hard to define what exactly a meme is; 2) the theoretical statements of memetics are as yet too vague to be empirically verifiable or falsifiable (Edmonds, 2002). The present paper proposes a broad methodology to address these problems. We will argue that: a) a memetic perspective can suggest concrete and non-trivial predictions; b) given a suitable memetic unit of investigation, these predictions can be tested empirically. This should establish a firm operational footing for memetics, allowing a comparison of the strengths and weaknesses of different models, and thus transforming memetics from a collection of suggestive hypotheses into a true scientific discipline.

2 Meme Selection Criteria

The core idea of memetics is that the popularity or success of a meme is determined by natural selection. At any moment, several memes are in competition for the attention of potential hosts and only those memes will spread that are well-adapted to the socio-cultural environment formed by these hosts and the network of their interactions; the others will become extinct. This leads to the generic prediction that "fitter" (i.e. better adapted) memes will become more widespread than less fit ones. To operationalize this as yet very abstract (and to some degree tautological) idea, concrete selection criteria need to be formulated that specify the degree to which a meme is adapted to its environment.

Several authors have tried to formulate principles that govern the spread of information. For example, Dawkins (1976), generalizing from the characteristics of biological evolution, listed the following three characteristics for any successful replicator, and thus for a meme: copying-fidelity, fecundity (number of copies made per time unit), and longevity (duration that any copy will survive). Working from a viral marketing perspective, Godin (2002) introduced the concept of the velocity with which 'Idea Viruses' spread from person to person. The social psychologists Schaller, Conway & Tanchuk (2002) focused on the communicability of a cultural trait. However, these characterizations of memetic fitness remain very broad and vague: what is it that makes a meme more communicable, fecund, or faster in spreading? They therefore offer little guidance in making non-trivial predictions.

Other authors have started listing more concrete and detailed criteria that together determine the fitness of a meme. For example, Castelfranchi's criteria (2001) focus on the social and cultural mechanisms of cultural transmission. A different list of criteria (Heylighen, 1997, 1998) focuses on the ways memes adapt to their hosts. In this work, four general criteria families are distinguished: objective, subjective, inter-subjective and meme-centered, depending on whether the selection depends on outside, objective reality, the individual subject or host of the meme, the process of transmission between subjects, or the internal properties of the meme itself. Heylighen (1998) proposes a four-stage model for memetic replication: 1) assimilation of a meme by a host; 2) retention within the host's memory; 3) expression by the host through behavior, language or some other medium; 4) transmission of the expression to one or more other hosts. At each stage there is selection, in the sense that some memes will be successfully assimilated, retained, expressed or transmitted, while others will not. A fit meme must pass all stages. The different selection criteria are typically active at different stages of this replication process.

The following is a selection of the most important criteria of this model[Heylighen, 1997, 1998], that can be easily operationalized:

• utility (the meme contains useful or valuable information)
• novelty (the meme is sufficiently different from already known memes)
• coherence (the meme is consistent with the knowledge that the hosts already have)
• simplicity (since complex memes difficult to process, less important details tend to be left out)
• formality (the less context or background communicating hosts share, the more important it is to express the meme explicitly)
• expressivity (the meme is easily expressible in the available languages or media)
• authority (the source is recognized as being trustworthy)
• conformity (the majority of hosts agree on the meme)
• proselytism (the meme explicitly incites its hosts to spread it further)

The first four of these are subjective and therefore depend on the host: what is useful or novel for one person may not be so for another one. The next four are intersubjective: they depend on the relations and forms of communication between hosts, and thus on the structure of the socio-cultural system. The last one, proselytism, is an example of a meme-centered criterion, that depends only on the meme itself. Simple examples of such self-promoting memes are viral sentences that contain a copy instruction, such as ‘Copy me’ or ‘say me’ (Hofstadter, 1996).

The general prediction that can be derived from this model is that, all other things being equal, if one meme scores higher on one of these criteria than another meme, it will also be fitter, and therefore spread more far and wide. For example, of two otherwise equivalent injunctions the one that is backed up by an authority (such as the pope), or by the majority of the population is likely to make more converts than the one that is not; the one that is novel will attract more attention and therefore spread faster; the one that fits in with people's existing ideas is more likely to be understood and believed and therefore to be memorized and expressed, etc. Moreover, the more criteria a meme fulfils the greater its overall fitness. Thus, the criteria, if valid, would provide us with a set of guidelines for how to recognize and design successful memes.

3 Methodologies for testing the selection criteria

3.1 Creating a memetic transmission chain

Different paradigms exist to study the spreading of memes. Perhaps the most direct, interactive one is the old game of "telephone" or "Chinese whispers", in which one person tells a story to another one, who then tells what (s)he remembers of it to the next person in line, who passes it on to the next one, and so on. At the end of the transmission chain, the final version is compared to the original story. To the amusement of the participants, the differences generally make the end story almost unrecognizable from the begin story.

From a memetic perspective, the different elements of such a story can be seen as individual memes. Some of these memes will be fitter, in the sense that they survive the many omissions and variations during the consecutive transmission better than others. Thus, the results of such a game may show what distinguishes good memes from poor ones.

An elegant example of this approach can be found in the psychological experiments of Lyons & Kashima (2001, 2003). In their game, the first participant read a made-up story about a non-existent tribe, the Jamayans. This participant 1 would retell the story to participant 2, who would retell it to 3, and 3 to 4, who told the final version to the experimenters. Before the experiment started, all participants had received background information about what kind of people the Jamayans were supposed to be, and what opinion the other participants had about that. The story consisted of consecutive elements (e.g. "a Jamayan boy encounters a bear", "he climbs in a tree", "he throws a branch at the bear", etc.). Some of these elements fit with the background knowledge (e.g. climbing in a tree is consistent with the Jamayans being fearful), others did not (e.g. throwing a rock is inconsistent with Jamayans being peaceful).

After several such experiments under varying conditions, a statistical analysis of the story elements that remained at the end of the game found a number of systematic effects that appear to confirm four of the above criteria: 1) coherence: elements inconsistent with the background information were more likely to be left out; 2) novelty: elements that the participants assumed were already known by the others were more likely to be left out; 3) simplicity: details or embellishments that did not affect the story line tended to be left out; 4) conformity: when the participants were told that the majority of them believed that the Jamayans were, e.g., peaceful, they were more likely to leave out elements inconsistent with this fact than if they thought that this was only a minority opinion.

3.2 Simulating meme evolution

A second paradigm for quantitative memetic investigation is simulation. There have been many agent-based simulations of how cultural replicators can spread through a population (e.g. Best, 1997), of which the first one to explicitly speak about memes may well be Gabora (1995). However, the agents and the memes used in these simulations are generally too simple to be used as models for the higher cognitive, emotional and social dynamics that govern meme transmission among humans. One of the only selection criterion to emerge (i.e. without being imposed by the programmer) from such simulations is conformity: the more agents already host a meme, the higher the probability that the other agents will be infected as well (cf. Boyd & Richerson, 1985).

Van Overwalle, Heylighen & Heath have started to investigate more realistic models in which agents do not just copy a message (with or without errors), but actively "reinterpret" messages, based on their own subjective experience with other agents and messages. To achieve this, agents are represented by simple neural networks that learn from experience. A message then corresponds to a pattern of activation over the nodes in such a network, and communication to the spread of that activation from agent to agent via variable inter-agent connections. The strength of the connection between two agents represents the degree of trust of the one in the information received from the other. This trust is learned on the basis of the degree to which information from that agent is confirmed by own knowledge and other sources.

This approach may allow the selection criteria to be derived from the dynamics of such a distributed connectionist network, rather than have them posited to some degree ad hoc. A preliminary simulation (Van Overwalle, Heylighen & Heath, 2004) indeed suggests that this can be achieved. For example, the reinforcement of inter-agent links through the increase of trust builds authority for the sending agents, and tells them which information the receiving agents are likely to already know and agree with, making it less important for them to transmit detailed, explicit reports (novelty and formality). Moreover, spread of activation along existing connections will automatically attenuate inconsistent (coherence) or complex (simplicity) signals, while amplifying signals that are confirmed by many different sources (conformity) or that activate in-built rewards or punishments (utility). As a first test, this simulation (Van Overwalle et al., 2004) has been able to replicate the most important quantitative results from the aforementioned study of Lyons & Kashima (2001) concerning the probability with which inconsistent or novel story elements are replicated in their "Chinese whispers" game.

3.3 Analyzing existing meme frequencies

A different paradigm for memetic investigations is the collection of existing memes (e.g. urban legends), together with an estimate of their success (e.g. the actual frequency with which a given legend is encountered on the web, or the likeliness that a person is to pass on the story to someone else). The study can then look for correlations between actual or apparent success rates and different criteria to test in how far high scores on the criteria predict memetic fitness.

Heath, Bell & Sternberg (2001) used this method to investigate a number of properties that fall under the general heading of "utility". Utility is a very broad category that includes any estimate of the importance or value of the information contained in a meme. Some of these estimates will be made rationally, e.g. by considering the plausibility of a meme; others will be made more intuitively or emotionally, e.g. by reacting with pleasure to an implied opportunity or fear to an implied danger. From the emotional components of this value judgment, Heath et al. focused on disgust because this is a relatively simple emotion whose strength is easy to measure. When comparing different urban legends that contained an element of disgust (e.g. the story of a man who discovers a dead rat in the cola bottle he has just been drinking from), they found that the more disgusting variations typically were more likely to be spread than the less disgusting ones. The same applied to plausibility, thus confirming two components of a broader utility criterion.

4 A pilot study of virus hoaxes

4.1 Introduction

A shortcoming of the previous studies is that they used rather vague and variable memetic units: "story elements", "traits" or "patterns of activation". As such they do not satisfy Dawkins' requirement of copying-fidelity or the general criticism that memes lack a clear definition and are difficult to analyze.

Through the use of virus-hoaxes as memetic units (cf. Gorden, Ford & Wells) it was possible to eliminate these problems as the easy task of forwarding an email message makes it possible to have a nearly 100% copying-fidelity compared to other memetic spreading principles such as manual copying or oral communication.

The hypotheses of this thesis was that it would be possible to distinguish several selection criteria and correlate their scoring values to their degree of spreading, thus isolating the factors that are most important for their spreading. Thus it would be possible to give a ranking of the criteria for this specific kind of meme. It is important to note that the outcome of the research does not give a ranking of the importance of these criteria for all memes. Virus hoaxes are a particular kind of meme and it is our conviction that there will be a different importance of the criteria for different memes.

4.2 Virus hoaxes as paradigmatic memes

Virus hoaxes have been described as being examples of memes by various authors (eg. Gorden, Ford, Wells). They are email messages warning the recipients for a non-existent computer virus, and urging them to forward this warning to as many other people as possible. As such, a virus hoax is an illustration of a self-replicating message, that parasitizes the attention and computational resources of its recipients in order to maximally multiply itself. The continual expansion of electronic communication points us at the possible dangers of these virus hoaxes, which are threefold:

1) Virus hoaxes often propose methods of “protection” that are actually harmful (such as erasing essential program files).

2) They can create panic among naïve computer users by making them falsely believe that their computer is showing symptoms of a virus.

3) They produce economic damage by making their readers focus on the hoax instead of other activities, which results in a loss of time, energy, bandwidth and other resources.

Thus the study of how virus hoaxes spread is not only scientifically interesting, but it has direct social and economic applications. Moreover, these parasitic email messages are clearly delimited, normally undergo replication without variation, and, being pieces of text, are easy to analyze.

To test this memetics hypothesis, the statistical correlation between the score of a hoax on one of the criteria and an estimate of the degree of spreading of this hoax can be determined. It is important to make sure that enough different hoaxes are analyzed in order to obtain statistical significance. To be able to measure the degree of spreading (and thus the success) of a hoax, it is necessary to determine the exact content of the hoax text. Hoaxes are available in a number of specialized databases maintained by different organizations, such as Symantec or McAfee, on the internet. By comparing the different sources it is not only possible to find the most prevalent form but also to compare the strength of different mutations of the hoax. This could be used to recreate the evolutionary path that the hoax has followed, making a taxonomy of its different mutations (Bennett 2003).

Given the canonical form of a common variation, two or three distinguishing strings in the hoax’s text can be found that determine a unique "signature" of that text. Entering these signature strings in a search engine such as Google or AltaVista will not only find documents that contain this signature, but tell us how often these strings appear together on the internet, both on webpages or in newsgroups. This determines the number of copies of the hoax that still reside on the net.

4.3 The survey

To test this hypothesis, a small pilot study was performed in which 6 hoaxes were scored on 6 criteria by 195 participants (Chielens, 2003).This study was not only able to look at the importance of the selection criteria as proposed in the hypothesis but also presented an opportunity to examine the feasibility of this methodology for realizing quantitative results in a memetic study.

As this particular topic is closely linked to the Internet, an online survey was chosen to collect the data. One of the advantages of online surveys over live interviews is that there is less risk of answers being biased by social expectations, as participants may remain anonymous. The participants were volunteers from the student body of the Brussels Free University, ranging from freshmen to senior students. For the majority of them it can be assumed that English was not their first language.

Use of the computer display made it possible to represent the hoax as it would appear in a participant's mailbox, including the capitalization and grammatical or spelling errors. Moreover, as there is no time-pressure in a computer based survey, the participants can read and re-read the questions and the hoaxes as needed. To avoid a bias caused by the order of the hoaxes, three different surveys were created, each with the same criteria questions, but listing the hoaxes in different orders, participants were automatically and at random directed to one of the three surveys.

4.4 Choosing the Criteria

The scoring of the selection criteria can happen in two ways: objective and subjective. Certain criteria can be measured objectively by applying linguistic techniques directly on the hoax text. Simplicity, for example, can be measured with the aid of Flesch Kincaid or Gunning-Fog readability tests, or the average sentence or word length. Other criteria can only be measured subjectively, by holding a survey in which participants are asked to indicate how strongly a hoax satisfies a certain criterion. To obtain a statistically significant score, the same hoax can be evaluated by a large number of people, after which the scores are averaged. As an extra controlling factor, the same criteria can also be scored by a group of experts.

The criteria that were chosen to be included in the survey needed to be easily understood by the average participant. From the list above, the following criteria were selected: novelty, simplicity, utility, authority and proselytism. In the introduction to the survey each of these criteria was described so as to clarify its meaning. The short descriptions of the criteria were repeated with every question in the survey, as were the values (on a five-point scale) that could be entered for the criterion. For example, simplicity was tested with the following question: “How easy is it to understand this message? Is it hard to grasp or is it pretty clear and simple? (1: Very Hard / 5: Very Easy)”

The criterion of novelty was renamed to originality, in order to avoid a confusion with the idea that the hoax should be objectively ‘new’. Authority probed how far the presumable source of the information (e.g. "This dangerous virus was first announced by IBM and Microsoft") appeared trustworthy. Utility was split up into a negative component, danger, and a positive one, benefit, since these hoaxes always warn of the great danger that may befall the ignorant recipient of a virus, but more rarely also mention the positive measures that can be taken to protect against the virus. Another reason for this split is that negative information normally produces a stronger mental reaction than positive information, a phenomenon called "negativity bias" (Ito et al., 1998). The criterion of proselytism (called "replication pressure" in the survey) is a particularly salient characteristic of virus hoaxes, which typically urge recipients to pass on the warning to all their friends and acquaintances.

4.5 Results

After the participants had scored each of the criteria on a scale from one to five, the average scores were calculated, and correlated with the frequency with which the hoaxes appeared on the web or on newsgroups. One of the strongest correlations was found with the novelty criterion. This fits in with Godin's idea of the "filled vacuum" (2002): a meme can diffuse most easily in a niche where no similar memes are present yet. Specifically for hoaxes, a possible explanation for this correlation is that when a new type of hoax appears, it is not immediately recognized as a fake, whereas a hoax similar to older hoaxes will be found out more quickly. Another strong correlation was found for the criterion of benefit. Proposing a solution to a potential danger may help the hoax to spread as it gives the recipient a feeling of control, while it can indirectly confirm the false threat, as when the recipient carries out the hoax's instructions for tackling the problem and finds that, indeed, the specified file exists on his or her hard drive. Hoaxes that carried a warning with a possible ‘solution’ were indeed considered to have a higher benefit rating than hoaxes which only carried a warning.

The other correlations were too weak to be significant. This is probably due to the lack of data, as it is difficult to find reliable correlations when there are only 6 elements to compare.

However, another plausible explanation for the lack of correlation may be that the hoaxes used were by definition rather successful, since they otherwise wouldn't have appeared in hoax databases. This would mean that they were already close to the optimal score for the most critical criteria, so that a significant further increase in the score would be too much of a good thing, damaging the hoax's credibility. For example, the warning that a virus will erase your hard disk and damage your computer is already frightening enough; adding that it moreover may make you blind and put your house on fire would make the hoax lose its credibility. Similarly, it is likely that a too high proselytism score will not lead to a higher replication rate but to a ridicule of the hoax. A hoax that consists merely of ‘please pass me on’ phrases will not be passed on due to the lack of content, because people simply do not take it seriously (Hofstadter, 1996). A similar effect was found by Heath et al. (2002) in their investigation of disgusting urban legends: for the most successful legends, they found that it was impossible to create a more disgusting version, and the only plausible variations scored lower in disgust.

If most hoaxes in the sample would cluster around the peak value for a criterion, this would erase any clear correlation. To tackle this problem, further research would either need to use a more fine-grained statistical method than correlation coefficients to determine the relation between frequency and criterion scores, or artificially vary the score of a hoax to see whether it would lose in virulence, as Heath et al. (2002) did with some of their urban legends. An explanation for the fact that benefit and novelty still produced good correlations may be that these are less critical properties for virus hoaxes, unlike danger or proselytism, so that a typical hoax still has "room for improvement" on these dimensions.

5 Final Conclusion

Probably the most serious criticism of memetics is that it has not as yet produced any empirically verifiable predictions (Edmonds, 2002). Reviewing a number of partial and preliminary studies, using data about real memes or simulations of the social and psychological processes that govern their transmission, this paper has shown how memetic theories can be operationalized. This allows us to produce to a number of concrete, non-trivial and testable predictions, with immediate applications in domains such as viral marketing, the spread of rumors, or of parasitic email messages. It is our hope that this general approach will provide inspiration for other researchers to build more realistic and sophisticated memetic models, and to gather the detailed empirical evidence that will be necessary to convince other scientists of the value of the memetic perspective.

References

Aunger, R., (Ed.) (2001). Darwinizing Culture: The Status of Memetics As aScience Oxford/New York, Oxford University Press.

Bennet, Charles H. & Li, Ming & Ma, Bin. (2003) Chain Letters & Evolutionary Histories. Scientific American Jun. 2003, 64-69.

Best, M. (1997). Models for Interacting Populations of Memes. Journal of Memetics - Evolutionary Models of Information Transmission, 1. Retrieved October 31, 2004, from http://jom-emit.cfpm.org/1997/vol1/best_ml.html

Blackmore, S. (2000). The Meme Machine Oxford/New York, Oxford University Press (Original work published 1999)

Boyd, R., & Richerson, P. J. (1985). Culture and the Evolutionary Proces Chicago, Chicago University Press.

Castelfranchi, C. (2001). Towards a Cognitive Memetics: Socio-Cognitive Mechanisms for Memes Selection and Spreading. Journal of Memetics - Evolutionary Models of Information Transmission. Retrieved October 31, 2004, from http://jom-emit.cfpm.org/2001/vol5/castelfranchi_c.html

Chielens, K.. The Viral Aspects of Language: A Quantitative Research of Memetic Selection Criteria (unpublished thesis, Vrije Universiteit Brussel, 2003) http://memetics.chielens.net/master/Thesis.pdf

Dawkins, R. (1989). The Selfish Gene (2nd ed.) Oxford/New York, Oxford University Press (Original work published 1976)

Edmonds, B. (2002). Three Challenges for the Survival of Memetics. Journal of Memetics - Evolutionary Models of Information Transmission. Retrieved October 31, 2004, from http://jom-emit.cfpm.org/2002/vol6/edmonds_b_letter.html

Gabora, L. (1993). Meme and variations:: A computer model of cultural evolution. In . L. Nadel, & D. Stein (Ed.), Lectures in Complex Systems New York, Addison-Wesley.

Godin, S. (2000). Unleashing the Ideavirus London, Simon & Schuster.

Gordon S., Ford R. & Wells J. Hoaxes and Hypes. 7th Virus Bulletin International Conference in San Francisco, California, October 1-3, 1997. Retrieved November, 2005, from http://www.research.ibm.com/antivirus/SciPapers/Gordon/HH.html

Heath C., Bell C., & Sternberg E. (2001). Emotional Selection in Memes: The CAse of Urban Legends. Journal of Personality and Social Psychology, 81, 1028-1041.

Heylighen, F. (1997). Objective, subjective and intersubjective selectors of knowledge. Evolution and Cognition, 3, 63-67.

Heylighen, F. (1998). What makes a meme succesful?: Selection Criteria for Cultural Evolution.. In , Proc. 16th Int. Congress on Cybernetcis Namur, Association Internat. de Cybernétique.

Hofstadter, D. R. (1985). Metamagical Themas: Questing for the Essence of Mind and Pattern New York, Basic Books.

Ito T. A., Larsen J. T., & Cacioppo J. T. (1998). Negative information weighs more heavily on the Brain: The negativity bias in evaluative categorizations. Journal of Personality and Social Psychology, 887-900.

Lyons A., & Kashima Y. (2001). The reproduction of culture: communication processes tend to maintain cultural stereotypes. Social Cognition, 19, 372-394.

Lyons A., & Kashima Y. (2003). How are stereotypes maintained through communication?: The influence of Stereotype scharedness. Journal of Personality and Social Psychology, 989-1005.

Ming, L., Bin, M., & Bennet C. (2003). Chain Letters & Evolutionary Histories. Scientific American, 76-81.

Schaller M., Conway L. G., & Tanchuk T. L. (2002). Selective pressures on the once and future contents of ethnic stereotypes: effects of the 'communicability' of traits. Journal of Personality and Social Psychology, 861-877.

Van Overwalle, F., Heylighen, F. & Heath, M. (2004). From Communication between individuals to collective beliefs. Retrieved October 31, 2004, from http://pcp.vub.ac.be/Papers/SiennaSimulation.pdf

Simulation models for biological and cultural evolution

Agner Fog
Aalborg University Copenhagen / IKT-EIT
Lautrupvang 15, 2750 Ballerup, Denmark
agner@agner.org

Abstract

This article discusses the applicability of simulation models to cases of biological and cultural evolution. Two examples are mentioned where genetic simulation models give more reliable results than analytical models. Possible applications of genetic models to memetic evolution are sought, but not found. The differences between genetic and memetic evolution are so fundamental that the two processes cannot be described by the same mathematical models. Cultural evolution is a very complex phenomenon involving many factors that cannot be studied appropriately within the discipline of memetics alone. Some of these factors are best described in terms of quantitative variables rather than discrete information units. An example of a complex interdisciplinary model involving such factors is presented. Several conclusions are drawn regarding the construction of models of cultural systems.

1 Statistical models of Darwinian evolution

I will start this discussion with a statistical model of Darwinian competition. Assume that animals of a particular species are competing for a limited source of food. Individuals die one by one until there is enough food for the remaining animals. There are different variants or mutants of this species with different chances of finding food and hence different probabilities of surviving and reproducing. This selection process is repeated each generation. The process whereby individuals die one by one is analogous to the well-known statistical experiment of picking coloured balls from an urn without replacement. Each colour represents one variant of the species. If all variants have equal probabilities of dying, then the distribution of deaths is a (multivariate) hypergeometric distribution. The Darwinian model assumes, however, that different variants can have different fitness. This corresponds to an urn model where balls of different colour have different probabilities of being picked. Such a model can be envisaged by assuming that balls of different colours have different size or weight. The probability distribution of the balls that we pick from the urn is the little-known Wallenius' noncentral hypergeometric distribution:

$$\mathrm{f}(x_1, x_2, \ldots, x_c) = \left[\prod_{i=1}^{c} \binom{m_i}{x_i} \right] \int_0^1 \prod_{i=1}^{c} \left(1 - t^{\omega_i / D}\right)^{x_i} \mathrm{d}t \, ,$$

$$D = \sum_{i=1}^{c} \omega_i (m_i - x_i) \, ,$$

where c is the number of different colours, x_i is the number of balls of colour i picked, m_i is the initial number of balls of colour i in the urn, and ω_i is the odds for colour i (Chesson, 1976). The surprising complexity of this probability distribution comes from the fact that every time a ball is picked from the urn, the probabilities of the remaining balls are changed. Methods for calculating and sampling from this probability distribution will be published elsewhere (Fog, 2004a). Wallenius' distribution is useful for simulating genetic evolution in cases where there is a Darwinian competition for a limited resource (Manly, 1985).

Statistical models of evolution are useful in cases where genetic drift plays an important role. It is now widely accepted that biological evolution often goes through stages of punctuated equilibria (Gould and Eldredge, 1977). This phenomenon may be explained by assuming that certain steps in the evolution have very low probability, and that these rare steps are interspersed by intermediate steps of higher probability.

Assume, for example, that a particular species can gain a significant increase in fitness if the genome is mutated at two different loci A and B, but that the fitness is decreased if either A or B, but not both, is mutated. This situation can be illustrated by a fitness landscape with two peaks, see fig. 1.

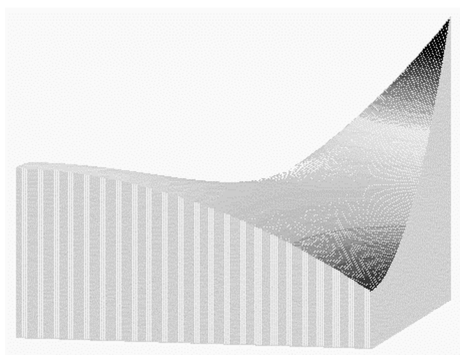


Figure 1: Fitness landscape with two peaks (Fog, 2000).

The x and y axes represent gene frequencies at locus A and B, respectively. The z axis represents mean fitness. The left peak represents the fitness of the wild type, and the higher peak to the right represents the fitness of the possible AB mutant. (Both mutant genes are recessive in the example of fig. 1). The probability that both loci mutate in the same individual is negligible. It is therefore difficult for the population to cross the valley in the fitness landscape and shift to the higher peak. A Monte Carlo simulation shows that the peak shift may occur in highly viscous populations if an unfit A-only or B-only hybrid spreads by genetic drift in an isolated area. The metaphor of a fitness landscape has led some scholars to believe that the most likely trajectory from the low peak to the high peak follows the direct path through the saddle point of the fitness valley, but the simulation shows that this is not the case. The trajectory almost always goes through the corner where the fitness has a minimum (Fog 2000).

This example attests to the value of statistical simulation studies. Another example is a study of group selection. Previous theoretical studies of genetic group selection have been based on mathematical analysis. Many dubious assumptions and approximations have been necessary in order to make the model mathematically tractable (Boorman and Levitt 1980). The results of this analysis have been contradicted by observations of natural examples of group selection in social insects, naked mole rat, and social scrimps (Sherman, Jarvis and Alexander, 1991; Duffy, Morrison and Rios, 2000). A preliminary simulation study including genetic drift is more in accordance with the observations (Fog, 2000).

2 Are these models applicable to memetics?

The above discussion indicates that genetic drift plays an important role in biological evolution and that the simulation of statistical models may be required for analyzing such phenomena. In particular, genetic drift plays a crucial role in certain types of peak-shifts necessary to pass the probability-barriers between punctuated equilibria. Similar barriers certainly do exist in memetic evolution. The crossing of such a barrier is known as a paradigm shift in the evolution of science (Kuhn 1962; Fog 1999). But there is a fundamental difference between the blind variation of genetic evolution and the intelligent problem solving activities of scientific evolution. While the fitness valley in figure 1 is difficult to cross by random genetic drift, a similar fitness valley in memetic evolution can easily be crossed by intelligent planning. Changing two genes is difficult when the intermediate step is unfit. But changing two memes is easy when the advantageous result can be anticipated. The simulation shows that a hypothetical step in biological evolution requiring more than two loci to mutate before the fitness advantage is reaped, is almost impossible due to the probability barrier. But in science and technology it is quite common to see evolutionary steps involving the change of many memes at the same time. Fitness barriers have been much studied in artificial systems of evolutionary algorithms where various techniques for crossing fitness barriers and entropy barriers have been developed (Oates and Corne 2001, Nimwegen and Crutchfield 2001). These techniques do not necessarily have analogies in biological evolution.

The consequence of these differences between biological and memetic evolution is that random drift is more important in genetic evolution than in memetic evolution. While models of random drift may have formal validity to certain cases of memetic systems, such models may obscure rather than clarify the study of memetic phenomena because the sociologically interesting effects lie elsewhere. Fitness barriers and random drift have more relevance to systems of evolutionary computation and artificial life (Nimwegen and Crutchfield 2000). The observation that population viscosity is necessary for the crossing of certain types of fitness barriers to be possible in biological evolution may give rise to interesting experiments in systems of artificial evolution.

3 The search for analogies

Mathematical models of memetics are typically inspired by genetics, and some theorists believe that the application of genetics methods to memetics is unproblematic (Kendal and Laland, 2000). So let's see how far the analogy goes. The model of coloured balls in an urn, which led to Wallenius' noncentral hypergeometric distribution, assumes that individuals compete for a limited resource and the losers die. The selection of memes is rarely based on the survival or death of their hosts, so we must look at the survival or death of the memes themselves. Memes may compete for a limited resource, namely hosts. But while genes competing for the same locus are mutually exclusive, this is not necessarily the case for memes. An individual may acquire the taste for a new kind of art or music without losing his or her fondness of previous pieces of art and music.

The closest we get to mutually exclusive memes is in the area of religion. People rarely confess to more than one creed at the same time. So in this sense, religious memes may be competing for the same host. But the urn model is still not appropriate. The urn model indicates that there can be no more survivors of a particular variant than there are balls of that colour in the urn at the beginning of the experiment. But the number of persons that can be converted to a particular set of religious beliefs is not limited by how many copies of the memes we have in the beginning, only by the number of potential hosts.

If we want to find a memetic system that has such a limitation, we may look at democratic elections. The number of candidates for each party may correspond to the number of balls of a particular colour in the urn experiment. In theory, we cannot elect more representatives for a particular party than there are candidates on the ballot. But the urn model would imply that voters cast their votes one by one and that candidates are removed from the ballot one by one, as they are elected, so that subsequent voters are more likely to elect a candidate from a different party. The urn model focuses on the possible limitation of available candidates for each party, but this is usually a rather uninteresting phenomenon. What is most interesting for a sociological or memetic study is the formation of voter preferences, and this is a process that mostly takes place before the voters enter the polling station. We may therefore conclude that it is difficult to find a memetic system where the formula of Darwinian competition can be applied as an exact analogy.

An analogous artificial system would comprise an evolutionary algorithm with a survival operator without replacement and where survivors are picked one by one in direct competition. Wallenius' noncentral hypergeometric distribution must be replaced by Fisher's noncentral hypergeometric distribution if survivors are picked simultaneously or independently (Fog 2004a).

Differences between memetic and genetic evolution are well known (Fog 1999):

- there is not one universal information unit in culture
- acquired traits can be inherited in memetic systems
- memetic inheritance can be both vertical and horizontal
- hosts can acquire new memes many times through their lifetime
- new memes do not necessarily replace any old memes
- innovations may be goal-directed rather than blind
- probability barriers can be overcome by intelligent planning

Most of the published mathematical models for memetics are heavily inspired by genetics (e.g. Lynch, 1998). The theorists may have constructed the models first, and then defined an imaginary cultural system that this model applies to. These more or less realistic cultural models often belittle the abovementioned differences between genetic and memetic evolution by defining mutually exclusive memes or strict cultural dichotomies; by giving more weight to vertical than to horizontal transmission; by assuming high degrees of cultural isolation between social groups; and by paying little or no attention to the factors that shape individual preferences. Such models may have relevance to ancient societies where social isolation was common, but they are less useful to the study of modern societies with their efficient means of communication.

If we want to understand the cultural dynamics of modern societies, it is more fruitful to first define the phenomenon we want to study, identify the most important factors that influence said phenomenon, and then build a model that fits our knowledge of these factors.

4 A complex example of cultural selection

Having rejected the strict analogy with genetics, we may now take a fresh look at the selection processes that guide cultural change. Living in a modern democracy, it seems obvious to start with an analysis of the democratic process that controls social developments. Elections are obviously determined by voter preferences. Voter preferences are influenced mainly by the political, cultural and other information that we receive through the news media. The contents of the media are selected by editors and journalists. The editorial staff is hired and controlled by media owners, who are forced to make an economically competitive product if they want their business to survive the fierce competition on the media market (except for the ever fewer non-commercial media). What makes a media product economically competitive is the kind of messages that attract the attention of the largest possible audience (sex, violence, gossip, etc.)... Look how quickly the model gets complex and how easily we get away from the original discipline of memetics! We have to look into psychological theories of which topics attract attention; economical theories of media competition; cognitive theories of the media effects on voters; etc. The selection of political memes is controlled by a lot of factors that are outside the discipline of memetics. Models that ignore factors outside the paradigm of memetics cannot provide a full understanding of the cultural dynamics. An interdisciplinary model is therefore needed. An attempt to construct a model of causal mechanisms that influence the democratic process in a modern society has led to the model in figure 2. This model will be discussed in detail elsewhere (Fog 2004b)

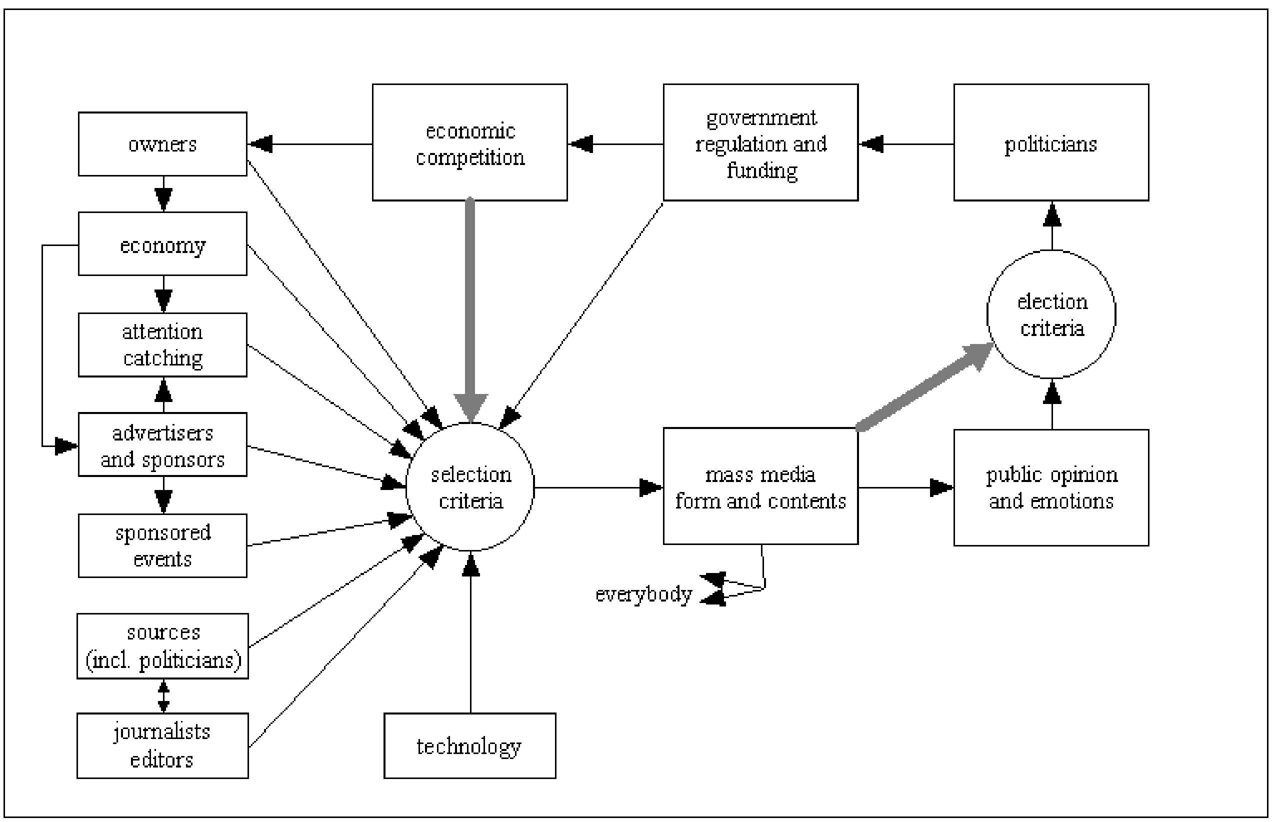


Figure 2: Interdisciplinary model showing the role of the mass media in a democratic society. The thick arrows indicate meta-factors that influence the weight of other factors: The mass media influence the criteria by which election candidates are evaluated by the process of agenda setting. The degree of economic competition between mass media determines whether the contents is determined mainly by the preferences of the staff or by what is economically most profitable (Fog 2004b).

While the election of candidates with different political ideas can be described as a memetic selection, there are other factors in this model that are less suited for the discipline of memetics. The effect known as agenda setting influences which issues people regard as most important and hence the criteria by which they evaluate the political candidates. The prioritisation of issues on the political agenda is a quantitative phenomenon which is more appropriately described by a quantitative model than by a model of discrete information units. Quantitative models become even more important when we want to analyze the economic competition between mass media. The preference of TV viewers for one kind of programs over another can make one TV station grow and competing stations shrink. The TV station that grows can afford to make even better programs, which contributes further to its popularity in a positive feedback process. Viewer ratings and economic turnover are quantitative measures that cannot be described in terms of discrete information units. The dynamics are better modelled by differential equations. This is nevertheless a selection process and indeed an important one if we want to study the evolution of a modern democracy.

This example should clarify my point: What different development processes have in common is selection, not discrete information units. Whether the term *evolutionary* should be applied to processes that involve automatic selection, but no discrete information units, is a matter of discussion. But the model in figure 2, taken as a whole, is indisputably evolutionary since the selection of quantitative variables is part of the mechanism that selects political memes.

The model described here is so complex that it is practically impossible to know all parameters in the model exactly. Unfortunately, this degree of complexity is probably necessary for a realistic study of cultural dynamics. A simulation of this model may not give realistic results when important parameters are not known, but the model may still be useful for determining connections qualitatively or for a sensitivity analysis.

5 Conclusions

- Simulation studies may give more accurate results than mathematical analysis
- Mathematical models of genetic evolution cannot be applied analogously to memetic evolution
- Random drift plays a larger role in genetic evolution than in memetic evolution
- What automatic development processes have in common is selection, not discrete information units
- Selection of quantitative variables and feedback processes may be important for modelling social and cultural developments
- There is no universal model for cultural evolution. Models have to be constructed on a case by case basis

References

Scott A. Boorman and P. R. Levitt. *The Genetics of Altruism*. New York: Academic Press, 1980.

Jean Chesson. A non-central multivariate hypergeometric distribution arising from biased sampling with application to selective predation. *Journal of Applied Probability*, 13(4):795-797, 1976.

J. Emmett Duffy, C. L. Morrison and R. Rios. Multiple Origins of Eusociality among Sponge-Dwelling Shrimps (Synalpheus). *Evolution*, 54(2):503-516, 2000.

Agner Fog. *Cultural Selection*. Dordrecht: Kluwer, 1999.

Agner Fog. Simulation of Evolution in Structured Populations: The Open Source Software Package Altruist. *Biotech Software & Internet Report*, 1(5):226-229, 2000.

Agner Fog. *Calculation methods for Wallenius' noncentral hypergeometric distribution*. Working paper, 2004a. www.agner.org/random/theory

Agner Fog. *The supposed and the real role of mass media in modern democracy*. Working paper, 2004b. www.agner.org/cultsel

Stephen J. Gould and N. Eldredge. Punctuated equilibria: The tempo and mode of evolution reconsidered. *Paleobiology*, 3:115-151, 1977.

Jeremy R. Kendal and K. N. Laland. Mathematical Models of Memetics. *Journal of Memetics*, 4, 2000.

Thomas S. Kuhn. *The Structure of Scientific Revolutions*. University of Chicago press, 1962.

Aaron Lynch. Units, Events and Dynamics in Memetic Evolution. *Journal of Memetics*, 2, 1998.

Bryan F. J. Manly. *The Statistics of Natural Selection on Animal Populations*. London: Chapman and Hall, 1985.

Erik van Nimwegen and J. P. Crutchfield. Metastable Evolutionary Dynamics: Crossing Fitness Barriers or Escaping via Neutral Paths? *Bulletin of Mathematical Biology*, 62(5):799-848, 2000.

Erik van Nimwegen and J. P. Crutchfield. Optimizing Epochal Evolutionary Search: Population-Size Dependent Theory. *Machine Learning,* 45:77-114, 2001.

M. J. Oates and D. Corne. Overcoming Fitness Barriers in Multi-Modal Search Spaces. In W. N. Martin and W. M. Spears, eds. *Foundations of Genetic Algorithms, 6.* San Francisco. 2001, pp. 5-26.

P. W. Sherman, J. U. M. Jarvis and R. D. Alexander. Eds. *The Biology of the Naked Mole-Rat*. Princeton University Press, 1991.

A Dynamic Network Flow Theoretical Model of Information Diffusion

Peter T. Breznay

*University of Wisconsin - Green Bay
Information and Computing Sciences
2420 Nicolet Drive
Green Bay, WI 54311
breznayp@uwgb.edu

Abstract

In this paper we present a new conceptual framework to model diffusion processes in networks. Our approach is to exploit dynamic network flow theory to model the dynamics of time-dependent diffusion processes in networked systems. In contrast to traditional network flow theory that emphasizes the optimization of network flows in the presence of capacity constraints, our objective is to build quantitative models of time-dependent network flow evolution. We derive systems of coupled differential equations whose solutions describe network flow dynamics, and apply the theory to study frequently occuring network process classes. The theory developed in this article is applicable to numerous problem domains where phenomena can be modeled by dynamically changing network flows, such as problems of logistical systems, transportation and traffic flow analysis, the study of information flow in communication systems and social networks, processes of information, innovation and meme diffusion, memetics, marketing theory and other fields.

1 Introduction

The study of information, in particular innnovation diffusion has been the subject of the work of many researchers (Valente, 1995), (Buskens and Yamaguchi, 1999). Several qualitative and quantitative methods have been used to model diffusion processes in natural and artificial networks, including logistical systems, traffic and transportation scheduling, communication systems analysis, and the study of social networks. Recently, the advent of memetics placed new emphasis on diffusion research, investigating how memes spread through populations (Best, 1997), (Edmonds, 1998), (Blackmore, 1999), (Lynch, 1999), (L Bull and Blackmore, 2001), (Lynch, 1998).

Many researchers rely on qualitative and empirical methods to study information and innovation diffusion processes, such as (D A Cliff and Versey, 1981), (Anderson and Jay, 1985), (Rogers, 1995), (Abrahamson, 1991), (Friedkin, 1991).

Of the quantitative approaches taken by researchers, two methodologies have dominated in the past few decades. One type of methods is based on the use of biologically motivated models, inspired by the methods of epidemiology, genetics and virology (May, 1973), (Dawkins, 1976), (Smith, 1982), (Edelstein-Keshet, 1988), (Feldman and Karlin, 1989), (M A Nowak, 2000). The other type of quantitative analysis makes use of statistical theories, creating probabilistic models of network communication flow and information diffusion, using the theory of stochastic processes, most importantly finite Markov chain methods (Bartholomew, 1967), (Bailey, 1970), (Kemeny and Snell, 1969).

The approach taken in this paper presents a novel conceptual framework for the study of information diffusion in networks, based on dynamic network flow theory. Although there exists an extensive body of work on network flow theory (Ford and Fulkerson, 1973), (Lovetskii and Melamed, 1987), (Aronson, 1989), (Magnanti and Orlin, 1993), (W B Powell and Odoni, 1995), (Kotnyek, 2003), the focus of these efforts is network flow optimization in the presence of capacity constraints (Hajek and Ogier, 1984), (Fleischer and Tardos, 1998), (Hoppe and Tardos, 2000), (Fleischer, 2001a), (Fleischer, 2001b) (Ferreira and Jarry, 2003), (Skutella, 2003). In contrast, in this paper we present a network flow model that allows us to derive the particular differential equations that govern diffusion processes in networks. We apply the theory to some important types of networks, and explicitly solve the diffusion equations for these network classes.

2 Static Unconstrained Network Flows

In this section we present the basic definitions and some properties of static unconstrained network flows. The main difference from traditional network flow theory is our use of undirected graphs, and the emphasis on the descriptive modeling of unconstrained network flows, as opposed to flow optimization under capacity limitations. In particular, in the version of the theory presented in this article, we do not consider edge capacity constraints. In traditional network theory we could say that the capacity of each edge is $+\infty$.

Let $G = (V, \mathcal{E})$ be a connected, undirected, simple graph. Here $V = \{x_1, \ldots x_n\}$ is the set of vertices, $n = |V|$ is the number of vertices, $\mathcal{E}$ is the edge set of G. Since G is undirected, $(x_i, x_j) \in \mathcal{E}$ if and only if $(x_j, x_i) \in \mathcal{E}$, and since G is simple, it has no loop edges: $(x_i, x_i) \notin \mathcal{E}\ \forall x_i \in V$; G has no multiple edges.

Definition 1 An $f : V^2 \to \mathbb{R}$ function is a *static network flow* on G if for all $1 \leq i, j \leq n$, $(x_i, x_j) \notin \mathcal{E}$ implies $f(x_i, x_j) = 0$.

The above definition is equivalent with the definition of dynamic network flows in the existing network flow literature, with the technical difference that traditionally the flow function is defined on the *edge* set of a *directed* graph. We can obtain the traditional concept by doubling every edge of G (one directed edge from x_i to x_j and one vice versa) and assigning $f(x_i, x_j)$ to the directed edge from x_i to x_j (and $f(x_j, x_i)$ to the directed edge from x_j to x_i). However, for our purposes the above definition, in particular the use of undirected graphs, is better suited. Our emphasis will be on describing the evolution of network flows by calculating the dynamic behavior of source strength functions of vertices (nodes). We interpret $f(x_i, x_j)$ as the *flow* from x_i to x_j (where $x_i, x_j \in V$ are two vertices of G).

In the following we will use the simpler term ”network flow” for static network flows. Note that by virtue of G being a simple graph, $f(x_i, x_i) = 0$ for all $1 \leq i \leq n$.

Our approach is motivated by many examples of network flows. The transfer of money between banks, the flow of data on a digital network between computers, the exchange of technical data between research institutions can all be represented by networks flows due to the fact that in these cases the unit of exchange is naturally defined. For example, money transfer can be measured in dollars, data flow between computers in bits, information exchange in collaborative research in specific units, such as descriptions of individually sequenced genes, when mapping a genome (such as the human genome) is a distributed collaborative process among several research centres.

However, more abstract flows can also be conceptualized. In the exchange of ideas, such as in the case of people talking on the phone, we can measure the number of words exchanged for a more concrete flow, but the number of ideas communicated as well, in a more abstract sense. Similarly, in memetic theory, the number of memes spreading over a network of people can also be construed as a network flow. Of course, actually constructing such abstract flows assumes advances in representational power, such as advances in meme mapping that in the future might allow precise representations of such abstract constructs as memes.

Definition 2 Let $x_i, x_j \in V$ be two vertices of G. The *net flow* from x_i to x_j is the function $n : V^2 \to \mathbb{R}$, defined by

$$n(x_i, x_j) = f(x_i, x_j) - f(x_j, x_i). \tag{1}$$

Proposition 1 If $(x_i, x_j) \notin \mathcal{E}$ then $n(x_i, x_j) = 0$.

Proof. $(x_i, x_j) \notin \mathcal{E}$ implies that both terms in (1) are 0. □

Proposition 2 For all $x_i, x_j \in V$

$$n(x_i, x_j) + n(x_j, x_i) = 0. \tag{2}$$

Proof. Immediately follows from (1). □

Definition 3 The *source strength* of $x_i \in V$ in the flow f is defined by

$$s(x_i) = \sum_{\substack{j \\ (x_i, x_j) \in \mathcal{E}}} n(x_i, x_j). \tag{3}$$

Proposition 3 For all $x_i \in V$

$$s(x_i) = \sum_{j=1}^{n} n(x_i, x_j). \tag{4}$$

Proof. Immediately follows from Proposition 1. $\square$

By Definition 3, $s(x_i)$ is the total net flow from x_i to its neighbours, i.e. x_i's net contribution to the network flow.

Definition 4 If $x_i \in V$ is a graph vertex then

if $s(x_i) > 0$ then x_i is a *source*,
if $s(x_i) = 0$ then x_i is a *transfer point*,
if $s(x_i) < 0$ then x_i is a *sink*.

These definitions comply with their usual meanings in traditional (static) network flow theory.

In the following we derive a theorem that connects the generation of flow within a subnetwork to the net flow through the boundary of the subnetwork.

Let $U \subseteq V$ and $G(U)$ be the induced subgraph by U. We consider $G(U)$ a *subnetwork*. It has edge set $\mathcal{E}(U)$ such that if $(x_i, x_j) \in U$, then $(x_i, x_j) \in \mathcal{E}$ implies $(x_i, x_j) \in \mathcal{E}(U)$. Clearly, there are two kinds of points in the subnetwork $G(U)$. If $x_i \in U$, then exactly one of the following two cases applies.

The *first* case for $x_i \in U$ is

$$(x_i, x_j) \in \mathcal{E} \Rightarrow x_j \in U. \tag{5}$$

In this case (x_i, x_j) is an *inner edge* of $G(U)$ (i.e. both endpoints of the edge are in U). We call such a point x_i an *interior point* of U. By definition, all neighbours of an interior point of a subnetwork are nodes of the subnetwork.

The *second* option for $x_i \in U$ is

$$\exists\ 1 \le j \le n \text{ such that } (x_i, x_j) \in \mathcal{E} \text{ but } x_j \notin U. \tag{6}$$

In this case (x_i, x_j) is a *boundary edge* of $G(U)$, and x_i a *boundary point* of U.

We will use the notation $I(U)$ for the set of interior points of U, and $B(U)$ for the set of boundary points of U.

As a result of (5) and (6), we have

$$U = I(U) \bigcup B(U) \tag{7}$$

and

$$I(U) \bigcap B(U) = \emptyset. \tag{8}$$

In other words, the node set of a subnetwork is the disjoint union of its interior and boundary points.

It is possible that $I(U) = \emptyset$, in which case $U = B(U)$ (all points are boundary). On the other hand $B(U) = \emptyset$ implies that either $U = \emptyset$ or $U = V$ (a non-trivial subnetwork always contains at least one boundary point, because G is connected).

In a completely disconnected subnetwork (no internal edges) $\mathcal{E}(U) = \emptyset$, which implies $I(U) = \emptyset$. However, it is possible that $\mathcal{E}(U) \neq \emptyset$ (there are internal edges), but $I(U) = \emptyset$. In this case every node of $G(U)$ is connected to at least one vertex outside $G(U)$, making every node of $G(U)$ a boundary point.

The following important theorem relates the total source strength within a subnetwork to the flow through the boundary of the subnetwork.

Theorem 1 (Divergence Theorem) If $U \subseteq V$ then

$$\sum_{\substack{i \\ x_i \in U}} s(x_i) = \sum_{\substack{i,j \\ x_i \in B(U) \\ x_j \notin U}} f(x_i, x_j) - \sum_{\substack{i,j \\ x_i \in B(U) \\ x_j \notin U}} f(x_j, x_i) \tag{9}$$

Proof. By Definition 3 we get $\sum_{\substack{i \\ x_i \in U}} s(x_i) = \sum_{\substack{i \\ x_i \in U}} \sum_{\substack{i,j \\ (x_i,x_j) \in \mathcal{E}}} n(x_i, x_j)$. The second sum in this last double sum runs through all neighbours of x_i. An internal edge $(x_i, x_j) \in \mathcal{E}(U)$ contributes to this sum at both endpoints. At x_i the edge contributes $n(x_i, x_j)$ to the flow, while at x_j the edge adds $n(x_j, x_i)$, thus the total contribution of an internal edge is 0 by (2). Therefore, to evaluate the above double sum, it is sufficient to to add the contributions of the boundary edges of $G(U)$. But these edges are exactly the edges with one endpoint belonging to $B(U)$, and the other belonging to the complement of U, i.e. $V \backslash U$. Consequently, $\sum_{\substack{i \\ x_i \in U}} s(x_i) = \sum_{\substack{i \\ x_i \in B(U)}} \sum_{\substack{i,j \\ (x_i,x_j) \in \mathcal{E} \\ x_j \notin U}} n(x_i, x_j) = \sum_{\substack{i \\ x_i \in B(U)}} \sum_{\substack{i,j \\ (x_i,x_j) \in \mathcal{E} \\ x_j \notin U}} [f(x_i, x_j) - f(x_j, x_i)] = \sum_{\substack{i,j \\ x_i \in B(U) \\ x_j \notin U}} f(x_i, x_j) - \sum_{\substack{i,j \\ x_i \in B(U) \\ x_j \notin U}} f(x_j, x_i)$. $\square$

We call the quantities $\mathbf{O}(f, U) = \sum_{\substack{i,j \\ x_i \in B(U) \\ x_j \notin U}} f(x_i, x_j)$ and $\mathbf{I}(f, U) =$

$$\sum_{\substack{i,j \\ x_i \in B(U) \\ x_j \notin U}} f(x_j, x_i)$$ the *outgoing* and the *incoming* flows, respectively, *through the boundary* of the subnetwork $G(U)$. Theorem 1 states that the total source generated within a subnetwork equals the difference between the outgoing and incoming flows through the boundary of the subnetwork.

3 The Dynamic Equations of Network Processes

In this section we derive the governing equations of dynamic (time-dependent) network processes.

Definition 5 A *network process* u on the graph G is defined as a function

$$u : \mathbb{R} \times V \to \mathbb{R}.$$

A network process describes the dynamic change of flow quantities present at the network nodes. These quantities can represent the amount of any entity under investigation, such as liquid flow in a pipeline system, amount of commodities in a transportation logistics system, but also the amount of information (e.g. measured by the number of bits, or by the number of messages), or the number of memes circulating in a social network, viewed from a memetics point of view. Our goal is to find the differential equations that govern the evolution of network processes as a function of time.

Sometimes it is useful to restrict the definition of a network process to $u : \mathbb{R}_+ \times V \to \mathbb{R}$, with $\mathbb{R}_+ = [0, +\infty)$, to explicitly indicate that time starts at time $t = 0$.

Definition 6 A *dynamic network flow* f on the graph G is defined as a function $f : \mathbb{R} \times V^2 \to \mathbb{R}$ such that for all $x_i, x_j \in V$

$$\text{if } (x_i, x_j) \notin \mathcal{E} \text{ then } f(t, x_i, x_j) = 0$$

holds.

This means that for any $t \in \mathbb{R}$ the function $f(t, ., .) : V^2 \to \mathbb{R}$ is a (static) network flow. In the traditional network flow literature it is common to use directed graphs and assign flow values to the edges.

Next we associate a dynamic network flow with a network process.

Definition 7 Let u be a network process on G, and $\phi : \mathbb{R}^2 \to \mathbb{R}$ a given function, that we call the *conductivity function*. We call the *generated network flow of u under ϕ* the function $d_{u,\phi} : \mathbb{R} \times V^2 \to \mathbb{R}$ defined by

$$d_{u,\phi}(t, x_i, x_j) = \begin{cases} \phi(u(t, x_i), u(t, x_j)) & \text{if } (x_i, x_j) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases}$$

By its definition, the function $d_{u,\phi}$ is a dynamic network flow in the sense of Definition 6.

Definition 7 relates a dynamic network flow to a network process u under the assumption that the flow amounts along each edge are determined exclusively by the amounts at the endpoints, and depend on the level of *conductivity* of the link, determined by the function ϕ. A very simple case is the scenario when the flow is proportional to the **difference** of the amounts of u at the endpoints, and only the vertex with the higher amount emits flow to the vertex with the lower amount. In this case the conductivity function would be defined as

$$\phi(x, y) = c \cdot max\{x - y, 0\}, \tag{10}$$

where c is a proportionality constant (conductivity parameter).

Another important type of conductivity function defines a *periodically changing* network traffic. Such a conductivity function can describe among others a seasonally changing transportation process, such the delivery of agricultural products between countries in the northern and southern hemisphere, or a fluctuating interest in communication depending on the time of the day - communication taking place mostly during the daytime. A simple model of a periodic conductivity function with period 4 is shown in Figure 1.

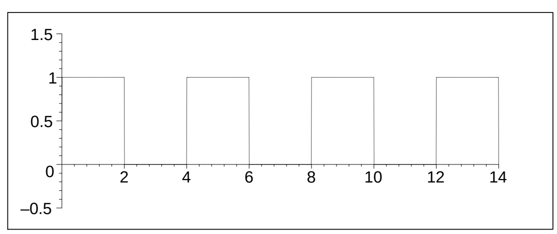


Figure 1: Periodic Impulse Function.

A further, frequently occurring, class of conductivity functions represent *delay* effects in reaction to network events. A typical delay function with delay

$d = 3$ is shown in Figure 2. Under the influence of such a conductivity function, a node of the network will delay participating in the network flow until after the delay period (in this example the delay will span the first 3 time units).

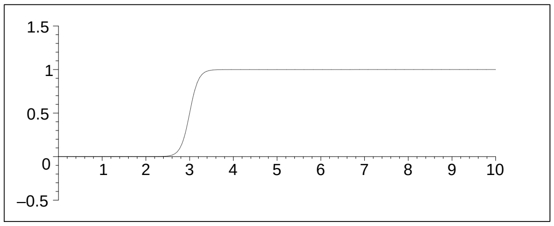


Figure 2: Delay Function, $d = 3$.

Now we derive the equations of diffusion on networks. Our basic modeling assumption is that we assume that the amount of flow at time t on a given edge (x_i, x_j) is determined by the amounts of u at the two endpoints of the edge, i.e. by the quantities $u(t, x_i)$ and $u(t, x_j)$, in addition to possible time dependence (as represented, for example, by periodic impulse or time delay functions) . In particular, the dependence of the flow is quantified by the conductivity function ϕ so that at time t the amount of flow over from x_i to x_j over the edge (x_i, x_j) is $\phi(u(t, x_i), u(t, x_j))$.

Theorem 2 (Diffusion Equation) Let $u : \mathbb{R} \times V \to \mathbb{R}$ be a network process on G. If all edges of G are loss and gain free, u satisfies the following equation at every vertex $x_i \in V$:

$$\frac{du}{dt}(t, x_i) = \sum_{\substack{j \\ (x_i, x_j) \in \mathcal{E}}} [\phi(u(t, x_j), u(t, x_i)) - \phi(u(t, x_i), u(t, x_j))]. \tag{11}$$

Proof. Consider a subnetwork $G(U)$ of G, where $U \subseteq V$. Since at time t the amount of the quantity u at vertex x_i is $u(t, x_i)$, the total amount stored at the nodes of $G(U)$ is

$$Q(t) = \sum_{\substack{i \\ x_i \in U}} u(t, x_i).$$

Assume that $u(t, x_i)$ is a differentiable (and sufficiently smooth) function of the time variable t. Then by the above equation

$$\frac{dQ}{dt} = \sum_{\substack{i \\ x_i \in U}} \frac{du(t, x_i)}{dt}. \tag{12}$$

On the other hand, since $\frac{dQ}{dt}$ measures the change of the total amount of u contained in the subnetwork $G(U)$, it must equal the difference of the net flow through the boundary of $G(U)$, in other words the negative of the source strength, since we assume that the edges are loss- and gain-free. By Theorem 1

$$\frac{dQ}{dt} = \sum_{\substack{i,\, j \\ x_i \in B(U) \\ x_j \notin U}} \phi(u(t, x_j), u(t, x_i)) - \sum_{\substack{i,\, j \\ x_i \in B(U) \\ x_j \notin U}} \phi(u(t, x_i), u(t, x_j)) = \sum_{\substack{i,\, j \\ x_i \in B(U) \\ x_j \notin U}} [\phi(u(t, x_j), u(t, x_i)) - \phi(u(t, x_i), u(t, x_j))]. \tag{13}$$

Let us consider the case $U = \{x_i\}$. Then (12) has the form

$$\frac{dQ}{dt} = \frac{du(t, x_i)}{dt}. \tag{14}$$

and in this casae $B(U) = \{x_i\} = U$. As result, (13) now can be written as

$$\frac{dQ}{dt} = \sum_{\substack{j \\ (x_i, x_j) \in \mathcal{E}}} [\phi(u(t, x_j), u(t, x_i)) - \phi(u(t, x_i), u(t, x_j))]. \tag{15}$$

By comparing (14) and (15) we obtain the required equation. □

Equation (11) can be used to analyze the evolution of dynamic network processes. For example, in the simple scenario of linear uni-directional conductivity (10), by considering the cases $u(t, x_i) > u(t, x_j)$, $u(t, x_i) < u(t, x_j)$ and $u(t, x_i) = u(t, x_j)$ separately, we eventually get

$$\phi(u(t, x_j), u(t, x_i)) - \phi(u(t, x_i), u(t, x_j)) = \begin{cases} c \cdot [u(t, x_j) - u(t, x_i)] & \text{if} \quad (x_i, x_j) \in \mathcal{E}, \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

In this case the diffusion equations of the system are

$$\frac{du}{dt}(t, x_i) = \sum_{\substack{j \\ (x_i, x_j) \in \mathcal{E}}} c \cdot [u(t, x_j) - u(t, x_i)]. \tag{17}$$

In the following we will call a network process a *transport process* if it obeys Theorem 1. The main characteristic of a transport process is its *conservation law*: the quantity present at a node is reduced by the amount of flow leaving the node, and increased by the incoming flow amount. As a result, the total quantity participating in the process remains constant. As the name indicates, transport processes describe traffic- and transportation-type flows, where a quantity (such as a number of vehicles) leaving a point is no longer there. The transfer of money between financial institutions is also a transport process.

In contrast, we call a network process a *replication process* if a quantity flowing from a vertex does not reduce the quantity available at the vertex. Such a process can be thought of as replication: the outgoing flow is immediately replaced at the source. Equivalently, we can think of the source as sending copies (or replicas) of its quantities to its neighbours, while retaining the originals. Information flow, and communication in general, such as sending e-mails or talking on the phone, are examples of replication process. Note however, that information flow can be a transport process under certain conditions. For example, sending original documents in case when photocopies are not acceptable, or transporting criminal evidence material is, in essence, information flow (the information about the existence of some document or material is transferred), but the information is no longer available at the source once it is sent.

We can establish the governing equations of replication processes in analogy of Theorem 2.

Theorem 3 (Replication Equation) Let $u : \mathbb{R} \times V \to \mathbb{R}$ be a network process on G. If u is a replication process and all edges of G are loss and gain free then u satisfies the following equation at every vertex $x_i \in V$:

$$\frac{du}{dt}(t, x_i) = \sum_{\substack{j \\ (x_i, x_j) \in \mathcal{E}}} \phi(u(t, x_j), u(t, x_i)). \tag{18}$$

Proof. Since u is a replication process, (13) changes to

$$\frac{dQ}{dt} = \sum_{\substack{i, j \\ x_i \in B(U) \\ x_j \notin U}} \phi(u(t, x_j), u(t, x_i)).$$

□

4 Application: Transport and Replication Process Dynamics

As applications of Theorems 2 and 3, we find explicit solutions for the case of linear, periodic and delayed network processes on various networks.

Uni-directional linear processes are defined by conductivity functions of the form (10). In this case the diffusion equations take the form (17) for transport processes and its equivalent for replication processes. The result in general is a system of coupled first order linear differential equations, with the number of equations equaling the number of vertices. The actual equations are determined by the graph G.

One of the simplest cases is the case when $G = K_n$, the *complete graph* or *clique* of n vertices. In this case, by (17) we can write the diffusion equations, using the notation $x_i(t) = u(t, i)$, where the vertices are labeled 1 through n, as

$$x_i'(t) = c \cdot \left(\sum_{\substack{j = 1 \\ j \neq i}}^{n} x_j(t) - (n-1) \cdot x_i(t) \right) \tag{19}$$

The general solution of (19) for $n = 6$ is

$$\begin{aligned}
x_1(t) &= c_1 \cdot e^{-6ct} + c_6, \\
x_2(t) &= c_2 \cdot e^{-6ct} + c_6, \\
x_3(t) &= c_3 \cdot e^{-6ct} + c_6, \\
x_4(t) &= c_4 \cdot e^{-6ct} + c_6, \\
x_5(t) &= c_5 \cdot e^{-6ct} + c_6, \\
x_6(t) &= (c_6 - c_1 - c_2 - c_3 - c_4 - c_5) \cdot e^{-6ct}
\end{aligned} \tag{20}$$

Similar solutions apply for the n-node case. Figures 3 and 4 show the solutions of (19) for the case $c = 0.1$ with initial values $x_1(0) = 30$, $x_2(0) = 26$, $x_3(0) = 22$, $x_4(0) = 18$, $x_5(0) = 14$ and $x_6(0) = 10$.

In general, regular networks such as cliques, cycles, hypercubes etc. result in symmetric sets of equations, and tend to have simple analytic solutions, similar to (20). As a second example, in the case of $G = C_n$, the *cycle* on n vertices, (17) can be written as

$$\begin{aligned}
x_1'(t) &= c \cdot (x_2 + x_n - 2 \cdot x_1(t)) \\
x_i'(t) &= c \cdot (x_{i+1} + x_{i-1} - 2 \cdot x_i(t)) \\
x_n'(t) &= c \cdot (x_1 + x_{n-1} - 2 \cdot x_n(t))
\end{aligned} \tag{21}$$

where $i = 2 \ldots n-1$.

The behaviour of the solution of (21) for $n = 6$ with the same initial conditions as above are shown

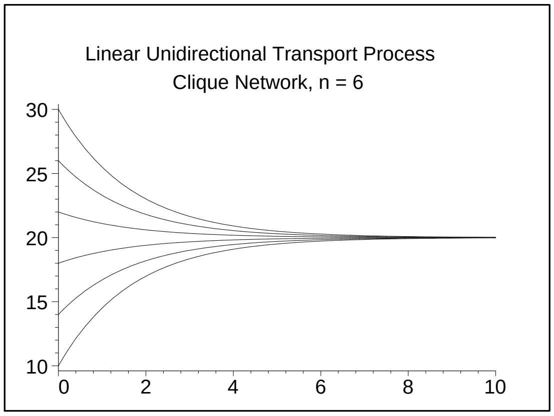


Figure 3: Solution for the clique network, $c = 0.1$.

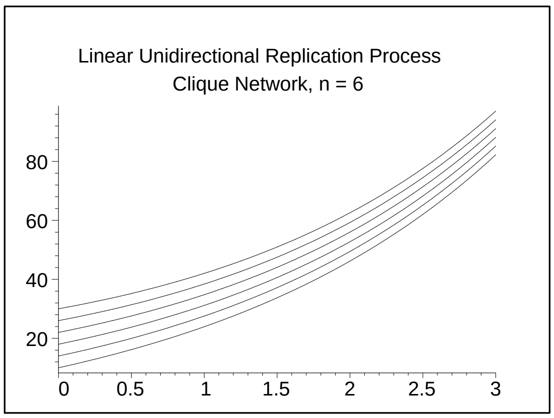


Figure 4: Solution for the clique network, $c = 0.1$.

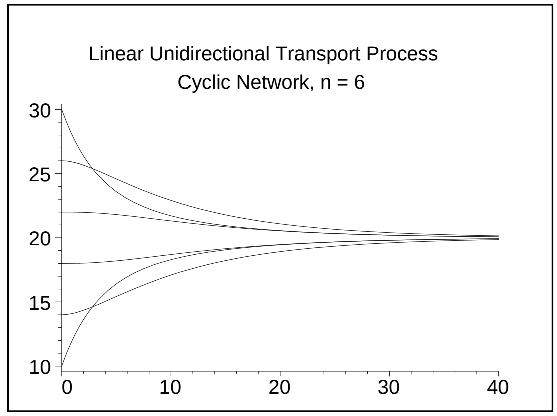


Figure 5: Solution for the cyclic network, $c = 0.1$.

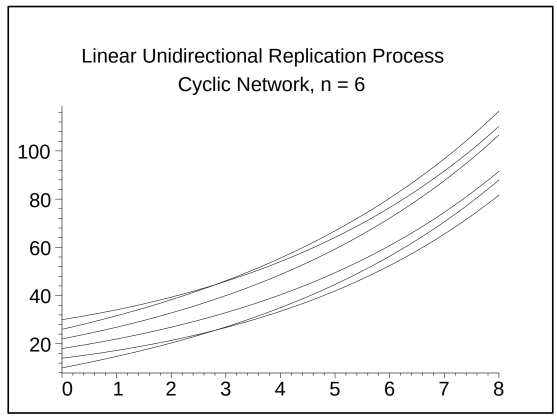


Figure 6: Solution for the cyclic network, $c = 0.1$.

in Figure 5 and Figure 6. These figures clearly show the general characteristic of transport processes to rapidly convert to an equilibrium (while conserving total quantity). Linear replication processes exhibit exponential growth.

While analytical solutions are easily obtained for clique and cyclic networks, removing just one link, e.g. from K_6, results in an asymmetrical set of equations with a highly complex analytical solution. For asymmetrical networks, even for moderate sized ones, numerical methods of solving the resulting systems of ordinary differential equations provide the best alternative to obtain the dynamic characteristics of the analyzed network processes and network flows.

Periodic processes are governed by conductivity functions similar to the one in Figure 1. The solutions of the resulting systems of equations for K_6 are shown in Figures 7 and 8.

The effect of varying *delays* are illustrated in Figures 9 and 10, this time for the cyclic network C_6. The delays are introduced by conductivity functions similar to the one shown in Figure 2.

Finally we apply the theory to study the behaviour of transport and replication processes on Granovetter and Watts-Strogatz networks. Granovetter (Granovetter, 1973) studied the effect of weak ties in networks with clustering. Figures 11 and 12 show the dynamics of flows in Granovetter-type networks. The solutions tend to cluster, reflecting the nature of the network.

Watts-Strogatz networks are highly clustered networks that are made "small-world" by random long-range links (Watts and Strogatz, 1998). Figures 13 and 14 model transportation and replication processes on this type of networks.

Our modelling approach has certain limitations. In assuming differentiability in the time-dependence of network processes, the method applies only to the

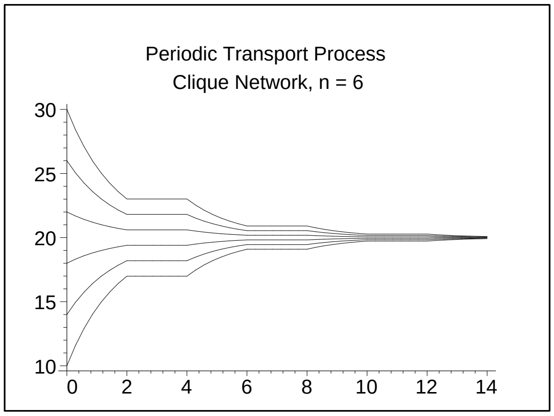


Figure 7: Transport process on the clique network under periodic impulse.

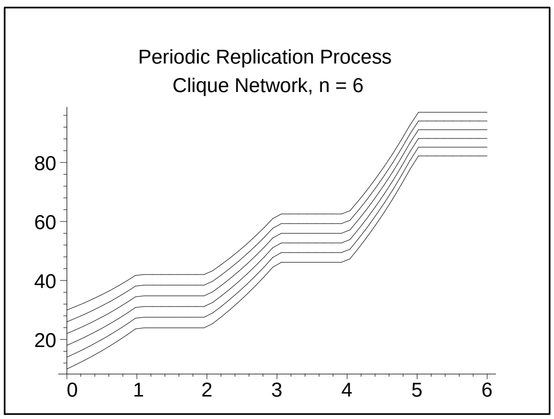


Figure 8: Replication process on the clique network under periodic impulse.

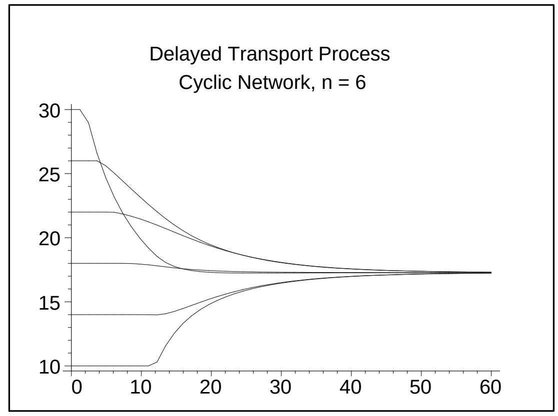


Figure 9: Delayed Transport process on the cyclic network.

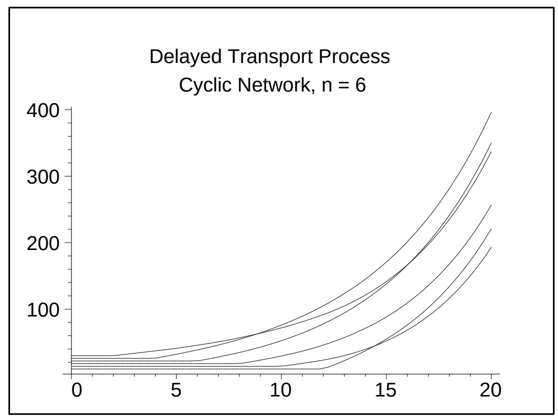


Figure 10: Delayed Replication process on the cyclic network.

dynamics of continuous phenomena. Many discrete quantities, however, are easily approximated by continuous curve fitting and discretisation of the solutions if needed.

5 Future Research

The model described in the previous sections represents a fairly simple scenario of information and commodity diffusion on a network. We are planning to investigate multiple generalizations and extensions of the theory. In particular, the model described here deals with only unconstrained flows, does not consider multicommodity flows and does not incorporate edge loss or gain (as generalized network flows do). Also, edge delay is not considered explicitly, providing ample material for further research.

The aspect of capacity constraints is probably the easiest to handle, since conductivity functions can be set up to represent capacities. We also investigated relatively simple conductivity models. One of our main objectives is to develop more complex conductivity models that are useful in studying diffusion processes in realistic networks, such as communication systems and social structures.

6 Conclusions

In this paper we presented a conceptual framework based on dynamic network flow theory to study diffusion processes on networks. We established a divergence theorem that characterizes diffusion dynamics, and formulated a method to derive the differential equations that govern these processes. We used the theory to quantitatively describe the diffusion dynamics of a linear unidirectional, periodic and delayed processes on some basic network topologies.

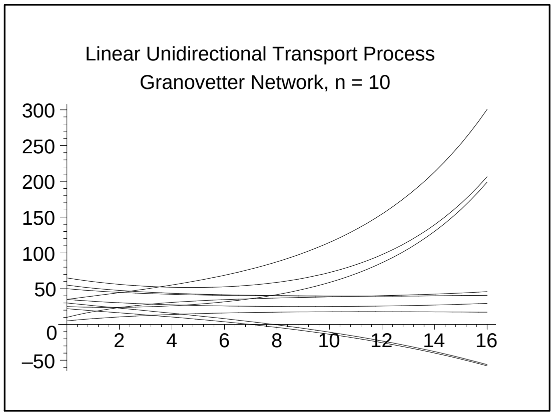


Figure 11: Transport process on the Granovetter network.

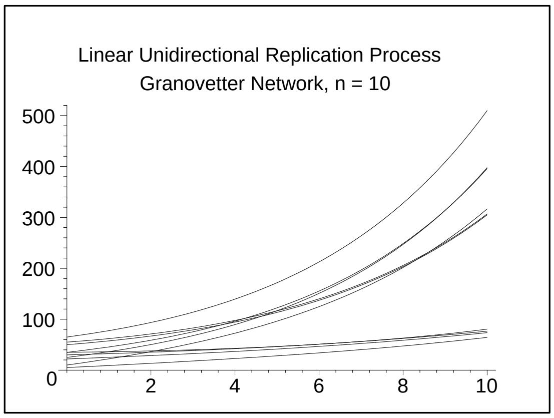


Figure 12: Replication process on the Granovetter network.

Acknowledgements

I wish to express a special thank to Professor Meir Russ for the many insights and valuable discussions on innovation diffusion and memetic theory.

References

E Abrahamson. Managerial fads and fashions: The diffusion and rejection of innovations. *Acad. of Management Review*, 16:586–612, 1991.

J G Anderson and S Jay. Diffusion of medical technology: Social network analysis and policy research. *Sociological Quaterly*, 26:49–64, 1985.

J E Aronson. A survey of dynamic network flows. *Annals of Oper. Res.*, 20:1–66, 1989.

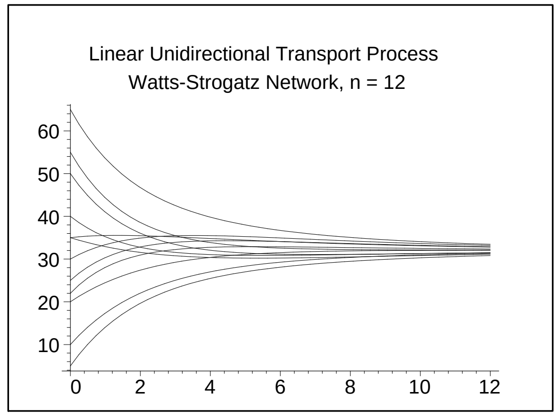


Figure 13: Transport process on the Watts-Strogatz network.

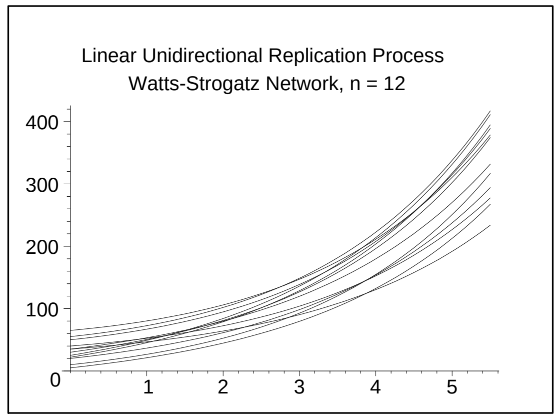


Figure 14: Replication process on the Watts-Strogatz network.

N T J Bailey. *The Elements of Stochastic Processes*. J Wiley & Son, New York, 1970.

D J Bartholomew. *Stochastic Models for Social Processes*. John Wiley & Son, London, 1967.

M I Best. Models for interacting populations of memes: Competition and niche behavior. *Journal of Memetics*, 1, 1997.

S Blackmore. *The Meme Machine*. Oxford Uni. Press, New York, 1999.

V Buskens and K Yamaguchi. A new model for information diffusion in heterogeneous social networks. *Sociological Methodology*, 1:281–325, 1999.

P Haggett J K Ord D A Cliff and G R Versey. *Spatial Diffusion: An Historical Geography of Epidemics in an Island Community*. Cambridge Uni. Press, New York, 1981.

R Dawkins. *The Selfish Gene*. Oxford Uni. Press, Oxford, 1976.

L Edelstein-Keshet. *Mathematical Models in Biology*. McGraw-Hill, New York, 1988.

B Edmonds. On modeling in memetics. *J. of Memetics*, 2, 1998.

M W Feldman and S Karlin. *Mathematical Evolutionary Theory*. Princeton Uni. Press, Princeton, NJ, 1989.

B Bui Xuan A Ferreira and A Jarry. Computing shortest, fatest and foremost journeys in dynamic networks. *International J. of Foundations of Comp. Sci.*, 14:267–285, 2003.

L Fleischer. Faster algorithms for the quickest transshipment problem. *SIAM J. on Optimization*, 12: 18–35, 2001a.

L Fleischer. Universally maximum flow with piecewise-constant capacities. *Networks*, 38:115–125, 2001b.

L Fleischer and E Tardos. Efficient continuous-time dynamic network flow algorithms. *Oper. Res. Letters*, 23:71–80, 1998.

L R Ford and D R Fulkerson. *Flows in Networks*. Princeton Univ. Press, Princeton, NJ, 1973.

N E Friedkin. Theoretical foundations for centrality measures. *Amer. J. of Sociology*, 96:1478–1504, 1991.

M S Granovetter. The stength of weak ties. *American Journal of Sociology*, 78:1360–1380, 1973.

B Hajek and R G Ogier. Optimal dynamic routing in communication networks with continuous traffic. *Networks*, 14:457–487, 1984.

B Hoppe and E Tardos. The quickest transshipment problem. *Mathematics of Oper. Res.*, 25:36–62, 2000.

J G Kemeny and J L Snell. *Finite Markov Chains*. D. Van Nostrand, Princeton, NJ, 1969.

B Kotnyek. An annotated overview of dynamic network flows. *Tech. Report Nr.4936, INRIA, Paris*, pages 1–28, 2003.

W Holland L Bull and S Blackmore. On meme-gene coevolution. *Artificial Life*, 6:227–235, 2001.

S E Lovetskii and I I Melamed. Dynamic flows in networks. *Automation and Remote Control*, 48:1417–1434, 1987.

A Lynch. Units, events and dynamics in memetic evolution. *J. of Memetics*, 1, 1998.

A Lynch. Thought contagion: How beliefs spread through society. *J. of Artificial Societies and Social Simulation*, 2, 1999.

R M May M A Nowak. *Virus Dynamics*. Oxford University Press, Oxford, 2000.

R K Ahuja T K Magnanti and J B Orlin. *Network Flows*. Prentice Hall, New Jersey, 1993.

R M May. *Stability and Complexity in Model Ecosystems*. Princeton Uni. Press, Princeton, NJ, 1973.

E M Rogers. *Diffusion of Innovations (4th Ed.)*. Free Press, New York, 1995.

L Fleischer M Skutella. Minimum cost flows over time without intermediate storage. *Proc. 35th ACM-SIAM Symposium on Discrete Algorithms*, 2003.

J Maynard Smith. *Evolutionary Genetics*. Oxford Uni. Press, Oxford, 1982.

T W Valente. *Network Models of the Diffusion of Innovations*. Hampton Press, Creskill, NJ, 1995.

P Jaillet W B Powell and A Odoni. Stochastic and dynamic networks and routing. *Handbooks in Oper. Res. and Management Sci.*, 8, 1995.

D J Watts and S H Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

Stigmergetic Communication for Cooperative Agent Routing in Virtual Environments

Steffen Priesterjahn
Dept. of Computer Science
University of Paderborn
33098 Paderborn, Germany
spriesterjahn@upb.de

Andreas Goebels
Dept. of Computer Science
University of Paderborn
33098 Paderborn, Germany
agoebels@upb.de

Alexander Weimer
Dept. of Computer Science
University of Paderborn
33098 Paderborn, Germany
aweimer@upb.de

Abstract

Coordinating multiple agents in real or simulated environments has become more and more important in recent artificial intelligence research. In this paper we present the concept of using the environment itself to coordinate the behaviour of a multi agent system (stigmergy) in a three-dimensional virtual environment instead of direct communication. We propose a system to avoid dangerous areas in a virtual map using different information propagation algorithms on extended waypoint graphs which work with global and local information availability. We will present these systems according to their advantages and disadvantages and compare their performance in static and dynamic settings.

1 Introduction

Due to their robustness and scalability swarm systems and multi agent systems have become very important for artificial intelligence research. Thereby, communication and information propagation has become a central aspect of artificial intelligence design. One of the most interesting aspects of swarm algorithms is how information is given from one agent to another.

A swarm normally exists of many individual agents with few, simple abilities which are able to solve hard problems through cooperation (Bonabeau et al., 1999; Kennedy and Eberhart, 2001). In the case of swarm systems it is often stated that the whole is more than the sum of its parts. Since there exists no central controlling instance, the scalability of a swarm system is very high. In addition, every agent can take over the job of another one, so such a system is very robust and fault-tolerant. Finally, swarms can adapt very fast to changing requirements.

For cooperation some kind of communication between the agents is needed. Since a lot of swarm systems are inspired by social insects, many artificial swarms use the environment for information interchange. Speaking in biological terms, information is left in the environment, for example in the form of pheromones which can be detected by the other agents. There is normally no direct communication between the agents. This concept is often referred to as stigmergy. (Bonabeau et al., 1999)

When looking at three-dimensional virtual environments, the question is how to store this environmental information or pheromone distribution. In this paper we propose the idea of using waypoint graphs, which are commonly used for navigational purposes in three-dimensional environments, to hold this information. We will present an application in which pheromone information is used to indicate dangerous areas in a map. For this application we will introduce two propagation methods; one which uses global and another one which uses only individual knowledge.

This paper is structured as follows. In section 2 we give an overview of related research to the topics presented in this paper. Then section 3 gives basic information about waypoint systems. In section 4 we describe the danger adaptive waypoint system, which is the main issue of this paper. There we present two information propagation methods. One which works with global information availability and one in which the agents themselves are responsible for the propagation of the information and every agent only knows parts of the global danger distribution. Finally, we will compare both strategies and their performance in static and dynamic settings in section 5.

2 Related Work

As this paper combines several aspects of computer science, namely swarm systems and multi agent systems as well as navigation in three-dimensional environments, several areas of related work exist.

Concerning swarm algorithms and approaches, much of the basics and possibilities have been researched by Bonabeau et al. (1999). They proposed the so called "ant algorithms" which are based on the behaviour of social insects like ants or termites. The biological foundation of this theory can be found in Grassé (1959) and Deneubourg et al. (1990). A well known result of Bonabeau's work are the "ant colony optimisation" (ACO) algorithms which simulate the foraging behaviour of ants to obtain shortest paths in a graph. These algorithms are successfully used for package routing in the internet. Our approach is closely related to the ideas behind the ACO algorithm as we also try to compute optimal paths. However, the usage of pheromones is somewhat different since ACO uses attracting pheromones while we are dealing with repelling ones. Another important work in swarm intelligence has been published by Kennedy and Eberhart (2001). They propagated the so called "particle swarm optimisation" algorithm which has been successfully applied to many hard problems.

Using waypoint systems or landmarks for agent navigation is very common for virtual environments and three-dimensional computer games (Lidén, 2001). Waypoints are special points in the three-dimensional space. They will be connected by edges, if it is possible to move directly from one waypoint to the other. For more detailed information about waypoints please see section 3.2. Another field in which waypoint graphs are used is robot navigation (Iyengar et al., 1985; Turchan and Wong, 1985). However, since exact positioning is often problematic for robots, the potential fields approach for navigation as proposed by Khatib (1986) and Arkin (1987) is mostly utilized in this field. In this approach attracting and repelling areas are used to navigate in an environment.

The research for agent behaviour in virtual environments plays an important role in artificial intelligence research. There are several approaches for modelling synthetic characters in virtual environments. Some of them use extensive planning, e.g. Hawes (2002); Kaminka et al. (2002); Nareyek (1998, 2002), while others apply learning approaches like reinforcement learning (Nason and Laird, 2004), neural nets (Bauckhauge et al., 2003; Thurau et al., 2003) or evolutionary algorithms (Bakkes et al., 2004). For character modelling the BDI (Belief-Desire-Intention) approach has shown some promising results (Norling, 2003). Much of this research has been done on single agents. Yet, in recent research the team aspect of multiple agents has become more important. Of this new research, much has been done in the field of robot soccer. There the researchers mostly use planning mechanisms, e.g. Tambe (1997), and reinforcement learning, e.g. Riedmiller et al. (2001), to gain team behaviour. Since we use swarm principles, our approach is different to the approaches mentioned above. It is somewhat related to BDI modelling, because we model attraction and repulsion with pheromone trails, but our agents also affect each other by spilling this pheromone onto the map. Our approach can also be compared to using potential fields, which are often used in the field of robot soccer, but mostly only for pure navigation. To our knowledge the combination of stigmergetic effects and waypoint systems is new.

3 Basics

3.1 The Artificial Environment

We use the Quake3-engine for our experiments. Quake3 (1999), often abbreviated as Q3, is a famous first person shooter computer game, which includes a powerful graphics engine and offers simple physics. The Q3 interface is open source and widely supported by a huge community. For placing the waypoints (see section 3.2) we built a waypoint editor within the Q3-engine (figure 1). There the waypoints are placed by hand, while edges can be placed automatically or by hand. We are also working on automatic map exploration and waypoint placement, but discussing this would go beyond the scope of this paper.

Figure 1: The waypoint system of a Q3-map

Looking at the "Capture The Flag" (CTF) game mode of many of today's computer games one can see that intelligent path finding for the agent teams is a major problem. In fact experienced players would say that choosing the path to the opponent flag is crucial for a good team strategy. In most computer games however the artificial characters just take the shortest or some random route. Therefore we chose a modification of the CTF game to determine what can be gained by using stigmergy to communicate routing information with each other. In the CTF game two teams fight against each other and try to steal the enemy's flag and to bring it to their own base. We modified the game so that simply hitting the opponent results in teleporting it back into its base. In the following we will call this process to be "marked".

3.2 Waypoint Systems

We begin this section by defining a standard waypoint system as it is used in many virtual, three-dimensional navigation applications.

Definition 1 (waypoint system, waypoint, edge)
A waypoint system *is a pair* (W, E)*, where* $W = \{w_1, ..., w_n\}$ $(n \in \mathbb{N}_0)$ *is a set of waypoints and* $E = \{e_1, ..., e_m\}$ $(m \in \mathbb{N}_0)$ *is a set of edges.* Waypoints $w \in \mathbb{R}^3$ *are defined as points in three-dimensional space. An* edge $e \in E$ *connects two waypoints and is therefore defined as a pair of two waypoints* $e = (w_1, w_2)$ *whereas* $w_1, w_2 \in W$ *and* $w_1 \neq w_2$.

So, a waypoint system is basically a directed graph in three-dimensional space with fixed positions for the nodes. Additional information is commonly added to the waypoints and (not quite as commonly) to the edges. For example a waypoint can mark a special item or it can hold special information about a trigger, e.g. a button which is positioned close to it. In computer games waypoints generally hold additional strategic information, e.g. whether it is a good spot to cover or a good position to wait and attack. Some examples how waypoints are used in computer games can be seen in (Lidén, 2001). In most applications, edges don't hold more information than their length and maybe a reachability value, e.g. whether you have to walk, jump or crawl to reach the next waypoint. The length of an edge $e = (w_1, w_2)$ is calculated by $\mathrm{dist}(w_1, w_2)$, where dist denotes the euclidean distance between two points in $\mathbb{R}^3$.

Another important property of many waypoint systems is that every waypoint can be reached from every other waypoint. This is due to the layout of the map and the automatic or manual placement of the waypoints. It should be also noted that normally most of the directly connected waypoints are connected in both directions.

The additional data discussed above is mostly not adaptive. This means that its features won't change according to the current game situation. This is because of the additional running time such a routing algorithm would take, since it would have to be reevaluated whenever the agent reaches a waypoint. In the game industry AI routines are not allowed to consume too much calculating time. However, some interesting results could be obtained when using adaptive waypoint or edge features.

4 The Danger Adaptative System

4.1 Basic Idea

We think that for intelligent routing in our environment it would be beneficial to avoid dangerous areas in the map and to take an alternate route. This is the main goal of our danger adaptive system. The principle idea of this system is that whenever an agent is marked it leaves some amount of a danger or fear pheromone on the place it has just been hit. Every time an agent "smells" this pheromone it tries to avoid such places. The pheromone strength decreases over time so the agents will not avoid this part of the map forever. Thereby, a system of avoiding dangerous spots for some amount of time is established. Since we already have the waypoint system at our disposal it is reasonable to use it to hold the pheromone information.

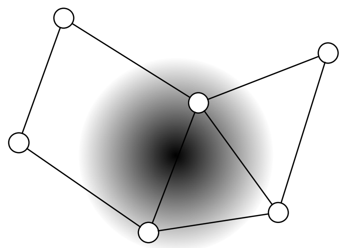


Figure 2: A pheromone spot on a waypoint system

A decision must be made of whether to store this information in the waypoints or in the edges. We chose to use the edges for holding this data, because our waypoints are not very dense (see figure 1) and because a waypoint only represents some spot in the map, while an edge represents the area between two waypoints. Automatically generated waypoints, however, tend to be more dense with much shorter edges. A reconsideration of our decision could be necessary in this case.

However, we are dealing with handmade waypoints in this paper. Therefore we extend the standard waypoint system as follows. The edges hold an additional value called *danger level* which indicates its dangerousness. These danger levels decrease over time by a given *half life*. The propagation of the danger pheromone is parametrized by the *propagation range*. How this propagation range is used depends on the pheromone propagation algorithm. A propagation range of 1 corresponds to approximately 1 metre in the simulated world. A formal definition of the danger adaptive waypoint system is given below.

Definition 2 (danger adaptive waypoint system) *A* danger adaptive waypoint system (DAWS) *is a 4-tuple* $\mathcal{W} = (W, E, h, r)$*, where* W *is a set of waypoints and* E *is a set of edges.* $h \in \mathbb{R}_{>0}$ *is called the* half life *of* $\mathcal{W}$ *and* $r \in \mathbb{R}_{\geq 0}$ *is called the* propagation range *of* $\mathcal{W}$*. For a* DAWS *an edge* $e \in E$ *is defined as a 3-tuple* $e = (w_1, w_2, d)$*, with* $w_1, w_2 \in W$*,* $w_1 \neq w_2$ *and* $d \in \mathbb{R}_{\geq 1}$*.* d *is called the* danger level *of edge* e*.*

The danger level is applied as a length modifier to its edge by computing the weight of an edge $e = (w_1, w_2, d)$ as $d \cdot \mathrm{dist}(w_1, w_2)$. So, an edge with danger level 2 appears twice as long as it really is. We consider an edge with its danger level to be our principle information piece.

The decrease of the pheromone strengths or danger levels, is handled by the following function

$$\begin{aligned} d_{new} &= \begin{cases} d^*, & \text{if } d^* \geq 1 \\ 1, & \text{if } d^* < 1 \end{cases} \\ d^* &= d_{old} \cdot e^{-\frac{\ln 2}{h}\Delta t} \end{aligned}$$

where h is the *half life* of the considered DAWS and Δt is the time since the last update. d_{new} and d_{old} are the new and old danger levels respectively. This function is oriented on similar natural decaying processes and is depicted in figure 3.

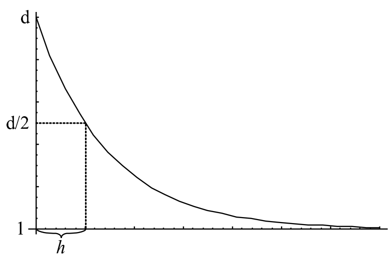


Figure 3: Decaying function of the danger pheromone

The question is now how to propagate the danger levels through the waypoint system. There are several possibilities to do this. Two concepts are presented in the following sections.

4.2 Global Danger Accessibility

In this section we are examining the propagation of the danger values by the waypoint system itself. This means that whenever an agent reaches a waypoint it is asking the waypoint system which way it should take. So the main part of the intelligence is implemented into the waypoint system, while the agents themselves have only very few abilities, namely walking from one waypoint to another and finding a first waypoint to go to at the start.

The algorithm for determining the danger levels is rather simple. Given a DAWS (W, E, h, r), an agent transmits its last position $p \in \mathbb{R}^3$ to the waypoint system whenever it is marked. Then, this position is given to every edge $e = (w_1, w_2, d_{old}) \in E$ for which the new danger level d_{new} is calculated by

$$\begin{aligned} d_{new} &= d_{old} + \frac{d_1 + d_2}{2}, \text{ with} \\ d_1 &= r - \mathrm{dist}(p, w_1) \text{ and} \\ d_2 &= r - \mathrm{dist}(p, w_2). \end{aligned}$$

This is a simple decreasing linear function depending on the distance to p. The agent is determining its path by using Dijkstra's algorithm for calculating shortest paths in the weighted graph. Since the danger levels are always decreasing until they reach 1.0 again, the optimal path has to be recalculated every time an agent reaches a waypoint. For better readability we call the agents which use this "global information accessibility" strategy *g-agents*.

Since simply always taking the safest route would result for all agents in taking the same path, in a real game an agent chooses randomly from the three shortest paths. Yet, the agent takes the safest route with the highest probability. Though, we did not activate this alternate path selection in our experiments, because it would distort the results.

Another solution for obtaining different paths and individual behaviour would be to use a personal DAWS for every agent. So every agent updates only its own DAWS at the points it has been marked. However, using this strategy would result in no information interchange between the agents, since every agent only acts to its own belief. A danger propagation algorithm which uses both ideas of global information availability and own agent beliefs will be presented in the following section.

4.3 Danger Propagation by the Agents

Thinking of a more natural approach to the agent behaviour we developed a system in which the agents itself are responsible for the propagation of the danger/fear pheromone. To achieve this, every agent has a personal DAWS in addition to the global DAWS in which the real danger state is stored. So every agent uses its own DAWS to determine its path and only updates it with the danger information from the global DAWS it came in touch with.

In detail the algorithm works as follows. Whenever an agent is marked, it spills some pheromone onto the edge it just used. It will do the same with the reverse edge of the current edge, if it exists. This means that it adds some amount of danger level on the real edge in the global DAWS and on the edge in its own DAWS. No other edges are affected. For the determination of the new danger level of the current edge the same algorithm as in section 4.2 is used with the difference that it is only applied to this edge.

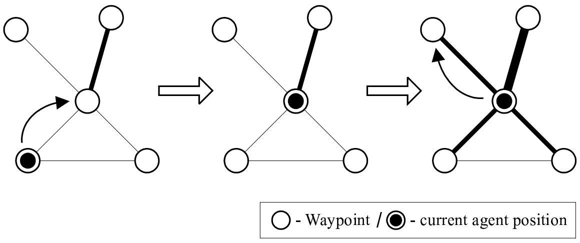


Figure 4: Propagation of the danger level by an agent. The thickness of an edge indicates its danger level.

As it was stated above, the propagation of the danger levels is done by the agents themselves. When an agent arrives at a waypoint (figure 4, left), it looks at all outgoing edges of this waypoint and sums up their danger level (figure 4, center). If the danger is high enough, the agent will spill additional danger pheromone on all considered edges (figure 4, right). This could be interpreted as the agent becoming afraid because of the danger it senses. By doing this, the danger can be propagated over the waypoint system, whereas it decreases with its distance to the originating edge. Formally the algorithm can be described as follows. $\{e_1, ..., e_k\}$ $(k \in \mathbb{N})$ are the outgoing edges of the current waypoint. e^{-1} denotes the, maybe not existing, reverse edge of edge e and $danger(e)$ denotes its danger level.

```
 1:  newDanger := 0;
 2:  for i := 1 to k do
 3:      newDanger += danger(e_i)-1;
 4:  end;
 5:  newDanger = r * newDanger / 8k;
 6:  for i := 1 to k do begin
 7:      danger(e_i) += newDanger;
 8:      if e_i^-1 exists then
 9:          danger(e_i^-1) += newDanger;
10:  end;
```

The standard danger level of an edge is 1. So, 1 has to be subtracted in line 3, because the danger level should not change when there is no danger. The agents use Dijkstra's algorithm on their personal DAWS to determine their paths. Because their choice depends on their personal belief of the danger distribution, every agent can make its own decisions. This means that they will take different routes. This also means that an agent has to walk over a dangerous edge by itself to see that it is dangerous there.

As a result every agent has a different view of the danger level distribution. Only the agent who has last seen a dangerous spot knows the real danger level value of this place. The others only know the danger values they have personally seen some time ago. Since they are expecting the danger level to drop by the decaying function, they belief the dangerous area is safer as it really is. This is because the last agent who has been there has raised the danger levels again. So the agents only have dated information for most of the edges and every agent has up-to-date knowledge of only some edges. Furthermore this up-to-date knowledge is different for every agent.

The propagation strength of the danger level depends on the number of agents which came in touch with the dangerous edges. Though, after an agent has learned that an edge is dangerous, it will not use it again for a period of time. So, the danger level of an edge will not grow higher, after all agents have learned that it is dangerous, as long as there exists an alternate path. However, because of the decay of the danger level the agents will use the edge again when its danger value has decreased enough.

As we did for the g-agents above, we call the agents which use this "local information accessibility" strategy *l-agents*.

5 Results

5.1 The Test Setting

For the first testing of our algorithms we built a simple test map with a waypoint system as illustrated in figure 5[1], to obtain reproducible results. There we have three different paths leading from the blue flag

[1] In the real test setting the number of waypoint was significantly larger than in this figure. (73 Waypoints, 83 Edges)

(b) to the red flag (r) and back. The middle path has been chosen to be the longest, to test whether the system will be able to converge to this path, if the other two paths appear to be dangerous. This was primarily important for the g-agents, because in this method all edges will be affected, if an agent is marked.

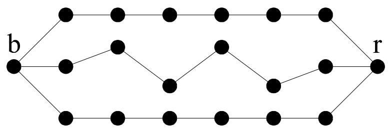


Figure 5: Waypoint configuration for testing

5.2 Static Scenario

We chose the following scenario to test our propagation algorithms. There were three agents in the blue team which all tried to get to the red flag and to bring it to the blue flag. We wanted to have an unsafe, an almost safe and a safe path. So, if an agent took the upper or the lower path, it would be marked at the middle of the map and brought back to the blue flag with a probability of 2/3 or 1/3 respectively. The agents would never be marked, if they took the middle path. Therefore, the middle path had to be a bit longer than the other paths, because otherwise the agents would have always taken it, without ever getting to the unsafe areas of the map. For better reproducibility the stochastic path selection by the g-agents was turned off. So they would only take the optimal path according to the current danger situation. We used a pheromone half life of 20 seconds and a danger propagation range of 1 for this experiment. A run is defined as the attempt to go from one flag to the other. In this setting the following results were obtained by our propagation algorithms.

Table 1: Results for static marking

strategy	runs	marks	ratio
g-agents	2267	121	5.3 %
l-agents	2226	184	8.3 %
random path selection	-	-	33.3 %[2]

Both strategies performed significantly better than simply choosing a random path. Interestingly the l-agents performed much better than we expected. Since every l-agent had to sense the danger for itself and since there were three agents, in the worst case the l-agent would perform three times worse than the g-agents. However, in this setting this factor was only 1.6. This is surprising if you take into account that the structure of the waypoint system is not ideal for the l-agents, because of the long unbranched paths in which the danger is only propagated edge by edge.

Concerning the overall behaviour of the agents they behaved as it could be expected. In both experiments the agents first took the shortest path and went for the middle path after some agents were marked on the outer routes. After some time one or more agents tried the outer paths again, only to see that they were still dangerous.

5.3 Dynamic Scenario

In a second set of experiments we dynamically changed the marking probabilities. As in the static experiment above, one path had a marking probability of 2/3, another one had a probability of 1/3 and the remaining path was safe. The selection of the dangerous paths did change every five minutes. We used different half lifes (10 s and 20 s) and propagation ranges (1 and 2) to see how these parameters influence the performence of the strategies.

Table 2: Results for dynamic marking

#	strategy	hl[1]	pr[2]	runs	marks	ratio
1	g-agents	10 s	1	11024	873	7.9 %
2	g-agents	10 s	2	11319	667	5.8 %
3	g-agents	20 s	1	8604	484	5.6 %
4	g-agents	20 s	2	12287	431	3.5 %
5	l-agents	10 s	1	9204	1125	12.2 %
6	l-agents	10 s	2	14642	4692	32.0 %
7	l-agents	20 s	1	11871	3907	32.9 %
8	l-agents	20 s	2	n.a.	n.a.	n.a.
9	random	-	-	-	-	33.3 %

In table 2 experiment 3 shows that the g-agents were almost not affected by the change of the scenario. Comparing the results obtained for a half life of 10 seconds and a propagation range of 1, the l-agents performed 1.54 times worse than the g-agents. This fits to the factor 1.6, which we obtained in the static scenario. However, in experiments 6 and 7 the l-agents performed much worse. In these cases the paths seemed almost equally safe to the l-agents, because the danger levels did not decay fast enough. So they ended up using the random path strategy. In experiment 8 the danger levels even built up to infinity, because of the much to slow decay. This shows that the behaviour of the l-agents depends highly on the used parameters.

[2] The value was calculated by assuming a uniformly distributed path selection. So, we get a probability of $\frac{1}{3} \cdot \frac{2}{3} + \frac{1}{3} \cdot \frac{1}{3} = \frac{1}{3}$.

[1] half life

[2] propagation range

The results for the g-agents show that they performed good in all experiments. A reduction of the half life resulted in an increase of the marking ratio. Using a danger propagation of 2 instead of 1 for the g-agents resulted in the same behaviour as if we doubled the half life.

5.4 Large Map Scenario

To validate our results for larger maps, we developed a scenario in which a much more detailed map was used. The waypoint system of this map had 340 waypoints and 939 edges. We used a quadratic dangerous area in which every agent would be marked with a probability of 3/4. So, it was possible for them to remain unmarked, though it was more probable to be marked. The position of the dangerous area was randomly shifted every 30 seconds by at most one 100 units (approximately 1 metre in the simulated world). So the area did move fast enough to be dynamic, but also slow enough to let the agents adapt to it. It should be noted that a slower danger area movement would have been beneficial for our agents, but also less realistic for the game.

For comparison we used a randomized strategy. In this strategy the agent randomly selected a path which would shorten its way to its target. This path selection was done at every waypoint. The results which were obtained in these experiments are shown in table 3.

Table 3: Results for the large map scenario

#	strategy	hl	pr	runs	marks	ratio
1	g-agents	10	1	11334	1036	9.1 %
2	g-agents	10	2	11080	774	7.0 %
3	g-agents	20	1	16387	882	5.4 %
4	g-agents	20	2	10686	361	3.3 %
5	l-agents	10	1	12328	2053	16.7 %
6	l-agents	10	2	13819	2180	15.8 %
7	l-agents	20	1	11885	1383	11.6 %
8	l-agents	20	2	17885	1927	10.8 %
9	random	-	-	8072	2478	30.7 %

Again the danger adaptive strategies performed much better than the random strategy. As in the experiments above the g-agents performed better than the l-agents. However, this time the l-agents had no problems with longer decay. This shows that this problem rarely occurs on larger maps, because there are much more alternative routes. Interestingly the change of the propagation range had almost no effect on the performance of the l-agents. A comparison of both strategies is shown in table 4.

In comparison 1 both strategies differ in a factor of 1.8, which is slightly higher than in the results obtained in section 5.3. Comparisons 2 and 3 show a factor of 2.1 and 2.2 respectively. In the last comparison we got the best results for both algorithms, but the factor has risen to 3.3. This is caused by the change of the propagation range, which has almost no effect on the l-agents, but results in a strong improvement of the g-agents.

Table 4: Comparison of the strategies

#	hl	pr	global	local	factor
1	10 s	1	9.1 %	16.7 %	1.8
2	10 s	2	7.0 %	15.8 %	2.2
3	20 s	1	5.4 %	11.6 %	2.1
4	20 s	2	3.3 %	10.8 %	3.3

These results continue the picture we have got from the prior experiments. It is obvious that in such a difficult task a strategy which uses only local information can not perform as good as when using global information. Yet, under some parameters the local strategy gets very close. Though, it is difficult to find these parameters.

6 Conclusion

We have presented a system which uses indirect information interchange to coordinate multiple agents to avoid dangerous areas in a three-dimensional, virtual environment. When comparing local and global information accessibility, the global strategy was advantageous as expected. When using the right parameters, the local strategy also performed very well. However, for some parameters the performance of the l-agents worsened in relation to the g-agents. Therefore, the local strategy was not as robust as we hoped. An intelligent method for choosing the parameters could increase the robustness of this strategy.

In conclusion, it seems to be possible to use only local information to obtain a good danger avoidance strategy. This is encouraging because the behaviour of the l-agents appears more natural and less "algorithmic" than the behaviour of the g-agents and would therefore be a candidate to model human behaviour. This final observation however is very subjective.

In the future we will try to better understand the parameter dependency of the agent behaviour, especially of the l-agents. Our aim is a method to automatically determine good parameters and to work with situation dependent parameters. Therefore, we will perform further experiments, especially with different numbers of agents, to validate our findings and to figure out more exactly how the parameters relate to the agent behaviour. Further studies will also be made about the information propagation through the waypoint system and between the agents and whether other information propagation methods for a local strategy would be advantageous.

References

R.C. Arkin. Motor Schema-Based Mobile Robot Navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 264–271, 1987.

S. Bakkes, P. Spronck, and E. Postma. TEAM: The Team-Oriented Evolutionary Adaptability Mechanism. In *Proceedings of the ICEC*, pages 273–282, 2004.

C. Bauckhauge, C. Thurau, and G. Sagerer. Learning Human-like Opponent Behaviour for Interactive Computer Games. In *Pattern Recognition*, Lecturenotes in Computer Science 2781, pages 148–155. Springer, 2003.

E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence - From Natural to Artificial Systems*. Oxford University Press, 1999.

J.L. Deneubourg, S. Aron, S. Goss, and J.M. Pasteels. The Self-Organizing Exploratory Pattern of the Argentine Ant. *Insect Behaviour*, 3:159–168, 1990.

P.P. Grassé. La Reconstruction du nid et les Coordinates Inter-Individuellez chez Bellicositermes Natalensis et Cubitermes sp. La théorie de la Stigmergie: Essai d'interpretation du Compartement des Termites Constructeurs. *Insectes Sociaux*, 6: 41–80, 1959.

N. Hawes. An Anytime Planning Agent For Computer Game Worlds. In *Proceedings of the Workshop on Agents in Computer Games at The 3rd International Conference on Computers and Games*, pages 1–14, 2002.

S.S. Iyengar et al. Learned Navigation Edges for a Robot in unexplored Terrain. In *IEEE Proceedings of the 2nd Conference on AI Application*, pages 148–155, 1985.

G.A. Kaminka, J. Go, and T.D. Vu. Context-dependent joint-decision arbitration for computer games, 2002.

J. Kennedy and R.C. Eberhart. *Swarm Intelligence*. Morgan Kaufman, 2001.

O. Khatib. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *International Journal of Robotic Research*, 5:90–98, 1986.

L. Lidén. Using Nodes to Develop Strategies For Combat with Multiple Enemies. In *Artificial Intelligence and Interactive Entertainment: Papers from the 2001 AAAI Spring Symposium*, pages 59–63, 2001.

A. Nareyek. A Planning Model for Agents in Dynamic and Uncertain Real-Time Environments. In *Proceedings of the Workshop on Integrating Planning, Scheduling and Execution in Dynamic and Uncertain Environments at the Fourth International Conference on Artificial Intelligence Planning Systems*, pages 7–14. AAAI Press, 1998.

A. Nareyek. Constraint-Based Agents - An Architecture for Constraint-Based Modeling and Local-Search-Based Reasoning for Planning and Scheduling in Open and Dynamic Worlds. In *Künstliche Intelligenz 2*, pages 51–53, 2002.

S. Nason and J.E. Laird. Soar-RL: Integrating Reinforcement Learning with Soar. In *International Conference on Cognitive Modelling*, 2004.

E. Norling. Capturing the Quake Play: Using a BDI Agent to Model Human Behaviour. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1080–1081, 2003.

M. Riedmiller, A. Merke, D. Meier, A. Hoffmann, A. Sinner, O. Thate, and R. Ehrmann. Karlsruhe Brainstormers - a reinforcement learning approach to robotic soccer. In *RoboCup 2000: Robot Soccer World Cup IV, LNCS*. Springer, 2001.

M. Tambe. Towards flexible teamwork. *Journel of Artificial Intelligence Research*, 7:83–124, 1997.

C. Thurau, C. Bauckhauge, and G. Sagerer. Combining Self Organizing Maps and Multilayer Perceptrons to Learn Bot-Behavior for a Commercial Game. In *Proceedings of the GAME-ON'03 Conference*, pages 119–123, 2003.

M.P. Turchan and A.K.C. Wong. Low Level Learning for a Robot: Environmental Model Acquisition. In *IEEE Proceedings of the 2nd Conference on AI Application*, pages 156–161, 1985.

Towards the Emergent Memetic Control of a Module Robot

Alexander Weimer
Dept. of Computer Science
University of Paderborn
Paderborn, Germany
aweimer@upb.de

Steffen Priesterjahn
Dept. of Computer Science
University of Paderborn
Paderborn, Germany
spriesterjahn@upb.de

Andreas Goebels
Dept. of Computer Science
University of Paderborn
Paderborn, Germany
agoebels@upb.de

Abstract

This paper contains the description of a self-reconfigurable module robot and a proposal for an emergent distributed control of such robots. The proposed control system is inspired by biological archetypes as well as by research in the field of peer-to-peer systems. The main design paradigm for this control system is robustness in relation to communication failures, incomplete information, fluctuations in the number of modules and defects of single modules. This robustness is achieved by the use of information pieces (memes) and communication to build and maintain the structure of the robot.

1 Introduction

Modular self-reconfigurable robots are composed of large[1] numbers of identical modules. These single modules are autonomous electro-mechanical devices equipped with own sensors and actors. They have the ability to communicate with other modules in their neighborhood.

This architecture allows these robots to change their shape according to the requirements of a specific task. For example when moving in a impassable environment it is possible for a module robot to shape new constructional elements (e.g. additional legs or manipulators) to move itself more efficiently, and to reshape them if these components are not necessary anymore. An example for this type of application is PolyBot described by Duff et al. (2001).

The ability to change the own shape according to the environment and the requirements of a specific task, combined with the potentiality to distribute the control over all modules without a central instance makes this architecture very flexible and robust.

The idea of self-reconfigurable robotics is not new, but most research in this field has been done in the last 10-15 years. This was caused by the rapid development of new hardware prototypes for this kind of robots (Støy, 2004b). The Research in this area can be divided into three fields which deal with the following reconfiguration classes: mobile reconfiguration, substrate reconfiguration and reconfiguration of closed chains (Casal and Yim, 1999).

Our research focuses on substrate reconfiguration.

[1] hundreds to billions

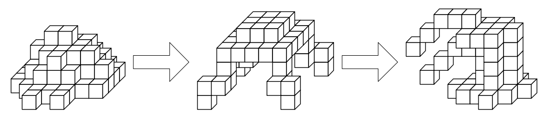


Figure 1: An example for self-reconfiguration

A widely-used model for module robots in this field is the Proteo-Robot introduced by Yim et al. (2001). This robotic system was designed to be capable of approximating arbitrary three-dimensional shapes. The Proteo-Model is used to design new control algorithms (Bojinov et al., 2000) as well as a model for other proposals which deal with the problems of dynamically changing the own shape. There are many interesting attempts to solve the self-reconfiguration problem beginning with the usage of cellular automata (CA) (Støy, 2004a) up to biologically inspired concepts like gradients (Støy, 2004a; Nagpal, 2001) or directed growth (Støy and Nagpal, 2004).

In most cases the researchers decided to store the complete control mechanism in each module to increase the robustness of the whole system. Our proposal is to divide the control mechanism in parts and make it distributable over the modules. This provides us the opportunity to change our control strategy and the plan of the desired shape on the fly. In this manner we can maximize flexibility and gain robustness by handling problems like dynamic adaptation of new shapes and new global behaviors as well as adaptation to dynamic fluctuations of modules caused by the replacement or loss of modules. In this way we can also reduce the hardware requirements for the single modules.

To construct a modular control mechanism we propose to design it as a memetic system where the memes are parts of the global control mechanism. In this way the behavior of the whole robot is a result of interactions of single modules. Each of the modules is controled by the interaction of memes in this module. To import and export memes the modules communicate with their neighbors (see section 3).

This paper is divided into five sections. In section 2 we describe the properties of the robotic system we simulated to check our control algorithms. In the following section 3 we give a description of the memetic system we used to build and maintain an arbitrary shape with our module robot and which is used as a distributed control system for the robot. Finally in sections 4 and 5 we discuss the results observed in our simulations and give an outlook on our future work.

2 Definition of the Module Robot

This section contains basic definitions and assumptions which are necessary for the analysis of the distributed control algorithms of our module robot. The robot defined in this section is a modification of the Proteo-Robot from Yim et al. (2001). These modifications were done to simplify the shape descriptions of the robot and are documented and motivated at appropriate spots.

2.1 The Robot

The following descriptions define our module robot:

- A *module* is an independent and autarkic electromechanical device with actors, sensors, memory, computing power and with the ability to move.
- In contrast to the original Proteo modules which have the shape of rhombic dodekahedrones (RD)[2] our modules have the shape of a cube. The Proteo module shape has better properties for the mobility of the modules and their packing in space against what our module shape is better qualified to simplify the descriptions for the building plans of the robot shapes.
- Modules are placed in a *grid space* where each cell can be described as a position in $\mathbb{Z}^3$. Each cell can either be empty or occupied by a single module.
- The *physical address* of a module can be uniquely described by its position in $\mathbb{Z}^3$.
- The *orientation* of a module in space doesn't matter because it is symmetric.
- As a result of our module placement and the shape of our robot we gain the following *neighborhoods* for each module (see figure 2):
  - 6 *face adjacent neighbors*,
  - 18 *edge adjacent neighbors* and
  - 26 *vertex adjacent neighbors*.
- A module possesses *sensors* which are able to detect if a face adjacent cell is free or occupied.
- A module will be able to *move* into one of its face adjacent cells, if it stays connected to an edge adjacent neighbor which is not moving.

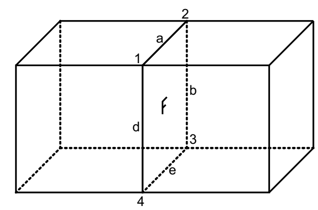


Figure 2: Two modules adjacent in one face (F), four edges (a-e) and four vertices (1-4)

In most control strategies for modular robots communication between modules is necessary. In the following we describe the requirements of the hardware of the modules for this purpose.

For the communication between single modules in a module robot we use the following terms:

- Two modules are *connected*, as soon as they are edge adjacent.
- Communication between two modules is established if they are connected.
- The *communication* is processed by the edges of a module. In doing so the module uses each of its 12 edges as a separate *communication channel*. Each edge possesses a *receive buffer* and a *send buffer*. The receive buffer is large enough[3] to store all received messages up to their procession. The send buffer is large enough to store all outgoing messages up to their transmission over the channel. The necessary size of the buffers depends highly on the control strategy of the robot.
- Communication channels are used asynchronously by the participating communication partners. The channel uses the Carrier Sense Multiple Access with Collision Detection protocol[4] whereas single edges work autonomously. Each edge sends all messages in its send buffer to the channel and writes all received messages to its receive buffer.

[2] RD is a 12 sided dual uniform polyhedron

[3] *Large enough* includes the theoretical eventuality of infinite size for the buffers.

[4] CSMA/CD

Having the definition of the module characteristics we can now describe a module robot which consists of a large set of modules and the shapes it can assume: A *module robot* consists of a connected set of identical modules in a grid space. A *configuration* of a module robot is the set of cells in the grid space occupied by the robot. A module robot is able to change its configuration by a series of *module movements*.

In the process of changing the shape (*self-reconfiguration*) of the module robot we can identify three interesting problems which are handled very differently:

Definition 2.1 (Control Problems)
Given an arbitrary, edge connected initial configuration I *and a* final configuration F *which is the desired shape for the module robot, also the description of the edge connected* current configuration C*, the current* internal states *of the modules* $S = \{s_0, s_1, ..., s_n\}$*, where* s_i *is the internal state of the module* i*, and the current* sensor information *of the modules* $P = \{p_0, p_1, ..., p_n\}$*, where* p_i *is the information perceived by the module* i*. We define the following problems:*

- *The* reconfiguration problem *of a module robot is defined as the search for a sequence of module movements which changes the configuration of the robot from* I *to* F*. This problem is also called the* planning problem.
- *The* global control problem *is the search for the next move for some module by using the current configuration* C*, the final configuration* F*, the internal states of the modules* S *and the sensor information of the modules* P*.*
- *For the* local control problem *the sensor information of a module* p_i*, an arbitrary piece of the building plan* $H \subseteq F$ *and the internal state of this module* s_i *are given. The solution of this problem is a sequence of actions. An action can be a movement or the transmission of a message.*

To maximize the robustness of the module robot we have to avoid central instances and bottlenecks. So, the maximal robustness is reached if all modules decide and act independently from the other modules. This independence of the modules focuses our interest on the local control problem.

Using only local information for the decisions leads to several problems. Two of the basic problems are *instability* and *local minima* (Yim et al., 2001). A stability problem occurs when the local controllers are not able to support global stability and to recognize when the global goal is reached. Local minima names situations where the robot is trapped in a non final configuration.

In both situations the first problem is to recognize the problematic situation, the second is how to deal with it. There exist several proposals to solve this problems or even to avoid them. We will discuss these solution attempts in section 4.

2.2 Control Model

The generic distributed control model of a module robot is based on the following assumptions:

- All modules have identical controllers.
- Each controller possesses enough memory to store parts of the final configuration and a set of states for the storage of information about the current configuration of the robot.
- The final configuration is available in the aggregate memory of the robot, eventually distributed over several modules. The final configuration is represented as a building plan as presented in section 3.1.1.
- Connected modules are able to exchange parts of the final configuration and current state information.
- Each module decides on the basis of local information wether it wants to move or not. If it moves it also has to decide which direction it has to take.
- If the movement of a module is not processed correctly the controller of this module will be informed. The reasons not to execute movements can be environmental restrictions, defects of other modules or malfunctions of the own actors.

Based on these assumptions of the control model of a module it is possible to define a generic control model. After the activation of a module its controller cyclically passes through the following phases:

1. *read messages phase*: In this phase the controller reads all received messages from the receive buffers and preprocesses them for further usage.
2. *decision phase*: In this phase the controller unit of the module decides, based on the sensor information, the received messages and the internal state of the module, which actions the module has to perform. An action is a movement or the act of sending a message.
3. *send messages phase*: In this phase the controller writes all messages it has to dispatch into the send buffers.
4. *move phase*: In this phase the module executes a movement. If the execution fails the module is informed about this.

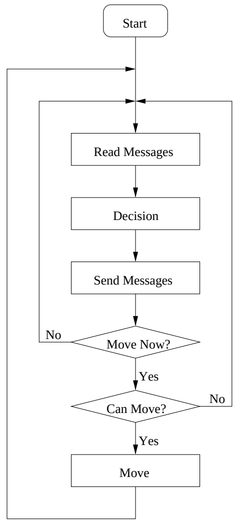


Figure 3: Flow chart of the controller of a module

By putting the strategies for the self-reconfiguration of the module robot into the *decision phase* of each module controller we avoid central instances in the module robot and gain a high degree of robustness.

3 Memetic Control

In this section we describe the memetic control mechanism for our robot. To do this we first have to clarify how we understand memes. In the last 30 years many publications dealt with memes as a concept (e.g. Dawkins (1976), Dawkins (1982), Brodie (1996), Lynch (1996), Blackmore (1999)), and each publication contains its own interpretations of this concept or refers many other possibilities. There is only one principle as a basic fundament of all definitions: *Memes are basic units of imitation*. This definition is a part from Dawkins original definition of memes in (Dawkins, 1976). So we see memes as information that affects the behavior of its host and is portable to other hosts.

There are two types of memes necessary to realize a control system based on imitation:

1. Memes to represent the states of the modules and the robot (declarative knowledge).
2. Memes for the manipulation of information, decision making and planning (procedural knowledge).

Both types of memes are mentioned by Dawkins in the foreword to (Blackmore, 1999).

In our design we have split the control strategy into subsystems. By copying these subsystems from one module to another we realized the imitation of behavior. We call this type of memes *procedural memes*. Information about the state of the robot, internal states of modules like relative positions and orientations, the desired shape for the robot and other declarative knowledge are stored as *declarative memes* in our system. By making local communication possible we ensure that memes can be distributed over the whole robot.

In the following two subsections we describe all declarative and procedural memes we use and the way how they interact with each other.

3.1 Declarative Memes

This section contains the description of the information pieces we use to maintain and to change the shape of the module robot. We divide our concepts into two ontologies which are called *building plan* and *gradients*.

3.1.1 Representation of the Building Plan

Here we describe the *building plan*, a representation of a final configuration of a module robot. The building plan consists of four components: *initial cuboid plan*, *initial position*, *cuboid plans* and *position unifications* described in the following definition:

Definition 3.1 (Building Plan)
A cuboid plan *is a 4-tuple* $Q = (id, X, Y, Z)$*, where* $id \in \mathbb{N}_0$ *is an unique identifier of the cuboid and* X, Y, Z *is the expansion of the cuboid in* $\mathbb{Z}^3$*. A* position *within a cuboid plan can be described as a 4-tuple* $pos = (id, x, y, z)$*, where* $id \in \mathbb{N}_0$ *is the identifier of the appropriate cuboid, and* $0 \leq x < X$*,* $0 \leq y < Y$*,* $0 \leq z < Z$*,* $x, y, z \in \mathbb{N}_0$ *are Cartesian coordinates of the position within this cuboid. A* position unification *is a 3-tuple* $P = (id, pos_0, pos_1)$*, which describes the positions (*pos_0 *and* pos_1*) of one module in two different cuboids, and* $id \in \mathbb{N}_0$ *is a unique identifier. Position unifications are used to describe the relative positions of cuboid plans to each other. With these premises we define a* building plan *as a 4-tuple* $B = (I_{plan}, I_{pos}, QP, PU)$ *with:*

- I_{plan} *is the* initial cuboid plan*. The* id *of this cuboid plan is* 0*.*
- I_{pos} *is the* initial position *within the initial cuboid plan. The module on this position is responsible for the global orientation of the whole robot in the Cartesian space.*
- QP *is a finite, non empty set of cuboid plans* Q*.*
- PU *is a finite set of position unifications* P *within the building plan.*

Each tuple of the definition is used as a single meme. It is assumed that each meme of the building plan is stored in an arbitrary module of the module robot at the beginning of the reconfiguration.

3.1.2 Gradients

Gradients are used in many control strategies as a tool for the maintenance and creation of structures. The idea of gradients in modular robotics is borrowed from biology and mimics the use of chemicals for control processes in living organisms. In our case we define gradients as follows:

Definition 3.2 (Gradient)
Let $GM = (id, type, dir, s, prio, val)$ *be a* gradient meme *where* $id \in \mathbb{N}_0$ *is an identifier of the gradient,* $type \in \mathbb{N}_0$ *is the type of the gradient,* $dir \in \mathbb{Z}^3$ *is the direction the gradient is received from,* $s \in \mathbb{Z}^3$ *is the relative position of the source of the gradient,* $prio \in \mathbb{Z}$ *is the priority of the gradient and* $val \in \mathbb{Z}$ *is the value of the gradient. Then* G_{gid} *is a* gradient *consisting of all gradient memes with the* id *equal to* gid*.*

Gradient memes are created, propagated and erased according to the control strategy of each module.

3.2 Procedural Memes

To demonstrate the capability of a memetic control in a module robot we have chosen a simple hillclimbing heuristic for maintenance and construction of a shape and have segmented the algorithm into independent subsystems. Our algorithm is very similar to the heuristic used by Støy (2004a). The behavior systems (*Building Plan Memory System*, *Helper Gradient System* and *Motion Control System*) and the complete building plan are necessary to construct and maintain the shape described by the building plan.

This information does not has to be stored in each module, it can be arbitrarily distributed over the whole module robot. The fundament for the stable behavior of the robot is the assumption that the *Behavior Control System* and the *Orientation System* are active in each module of the robot.

3.2.1 Behavior Control System

The Behavior Control System is the fundament for a stable memetic control. This meme enables modules to imitate behavior of their neighbors. Whenever a module percepts an unknown behavior by a neighbor it imitates the responsible behavioral rule immediately by copying the accordant procedural meme from this neighbor.

This works in the following way: Once activated this meme scans all perceptions made by the module including the received messages. If the module receives a message from a neighbor module which is not understandable, in other words a message from an ontology it doesn't possess, the module asks this neighbor for the behavior system that is able to deal with this ontology. After receiving the requested meme it integrates the received subsystem into its own controller and activates the new behavior.

3.2.2 Orientation System

This subsystem has to guarantee the availability of the common shared orientation among all modules of a module robot. It is necessary for the determination of the own position within the robot for each module. We have developed a system based only on local information propagation for this purpose. Exact description of this system will be published separately.

3.2.3 Building Plan Memory System

The Building Plan Memory System is used to maintain the building plan. So it acts as memory for all declarative memes which are involved in the description of the desired shape and all declarative memes concerning the actual relative position of the module in this shape. After a module movement this system also computes the new position of the module in the robot by copying the position of a neighbor and adjusting it by the direction of this neighbor. This behavior can be interpreted as the imitation and modification of a declarative meme.

This subsystem realizes a shared memory for the building plan using peer-to-peer technology where peers are single modules of the robot. It works very simple: Whenever the controller perceives an absence of a piece of the building plan it secures oneself the respective component of the plan. Otherwise, this system is also responsible for crowding out pieces of the plan which are used no longer, whereas the completeness of the whole building plan in the whole module robot has to be assured.

3.2.4 Gradient Propagation System

This subsystem is responsible for the propagation of any kind of gradients in the module robot. It consists of a database for gradients and behavioral rules for maintaining the database and communication with the neighbors of the module. The operating mode is quite simple: The subsystem scans all messages received by the module and searches for messages from the gradient ontology. If such a message is found, it will retrieve the gradient meme GM_{new} from the message and modify it by increasing its value (val), adjusting the direction (dir) to the direction the message is received from and recalculating the source (s) of the gradient. Afterwards it checks its own database on gradient memes with the same id as GM_{new}. If no gradients are found, it will be stored in the database. If a gradient GM_{old} is found in the database, the controller will compare the values of both gradients. If the value of GM_{old} is smaller, it will ignore GM_{new} otherwise it will replace GM_{old} by GM_{new}. If the meme with the identifier id is modified in the database the module informs all its neighbors by sending a message with the new meme.

3.2.5 Helper Gradient System

This subsystem perceives defects in the structure of the robot and repairs them by sending free modules to defective spots. This behavior is realized with *helper gradients*. According to its own position, available pieces of the building plan[5] and last sensor information the module checks if all neighbors are present. If neighbors are missing, the module will create a helper gradient for each of them with the source (s) on the position of the missing neighbor and the type ($type$) corresponding to ”helper gradients”. Afterwards the module broadcasts the helper gradients as the content of a message.

This system also periodically deletes old gradient memes of the $type$ ”helper gradients” from the module's gradient memory described in 3.2.4.

3.2.6 Motion Control System

This subsystem is responsible for changing the shape of the robot. This is achieved by controlling the movement of single modules.

This subsystem works in the following way: The controller checks if the position is correct according to the building plan then it will stay at the actual position else it will climb a helper gradient[6].

After each successful movement the controller flushes the memories of all subsystems of the module associated with the old position. This includes all sensor buffers, the complete gradient memory and its own positioning information[7].

3.3 Meme Interaction

The Gradient Propagation System is responsible for the storage of the gradient memes. This makes this system essential to all procedural memes, which use gradients. In our case the Helper Gradient System and the Motion Control System are using this system in different ways.

The Helper Gradient System creates new gradients to indicate failures in the desired shape, which is stored in the Building Plan Memory System. The Gradient Propagation System propagates them through the robot and the Motion Control System reacts to this failure memes by moving free modules to the corresponding locations to maintain the structure.

All procedural memes except the Behavior Control System are dependant on the Orientation System to determine the relative position of their host module to the neighbors of this module.

4 Observations

Simulations of our system have shown interesting results. We gained a high level of robustness and flexibility, but we payed for this advantage with a loss of performance.

To test our control system we have chosen a simple self-reconfiguration problem: Build a line of modules from a connected random initial configuration. The theoretical results of the runtime of the control algorithm are the same as those from Støy (2004a) and Yim et al. (2001), because of the strong similarity of the algorithms. But we have different results in the number of messages send by modules. We need around ten times more messages as Støy (2004a) to complete the self-reconfiguration task.

One reason for the higher communication amount is the peer-to-peer distribution of the building plan. This memory management also has the theoretical property to slow down the building heuristic by a factor equal to the number of modules in each step. However in most tests the delays were very short because of the locality of the queried information. We suppose another reason for the higher messaging amount in a tradeoff between the robustness of the control system and the reduction of the communication between the modules. Our system enhances the

[5]stored in the Building Plan Memory System

[6]*Climbing a helper gradient* stands for selecting a helper gradient (e.g. the helper gradient with the smallest value) and moving to a position in which the value of the selected gradient is not higher than at the old position.

[7]A module can compute its own position by asking its neighbor for its position and by adjusting the received position with the direction of this neighbor

robustness by sending redundant messages through different communication channels.

Using gradients enables each module of the robot to perceive whether the robot achieves the final configuration or even the necessity to maintain a damaged or not complete configuration. Gradients are also helpful for the detection of local minima because they are perceived as instabilities by the robot. For example in the strategy described in section 3.2 the structure described in the building plan is not complete as long as a module receives messages with helper gradient memes. This can be caused by two possible reasons, namely the structure is still under construction or the robot is captured in a non-final configuration.

The solution for the local minima problem on substrate reconfiguration is difficult. There are two ways to deal with it: avoidance of local minima or detection and correction. Støy (2004a) proposed to avoid this problem by choosing only those shapes for the robot where there is no possibility for such configurations. If we want to choose arbitrary shapes and avoid local minima we will have to build up the structure successive e.g. by directed growth (Støy and Nagpal, 2004). But defects on the structures may force us to correct such situations. This approach is still object to research.

5 Conclusion

In our work we combined ideas from modular robotics, distributed control, peer-to-peer networks and memetics to improve the robustness of the control of modular robots. We have proposed an architecture for module robots based on the definition of the Proteo-Robot put forward by Yim et al. (2001). We have also presented a modular, robust, distributed memetic control system for this robot based on the spreading of the information over the whole robot. Furthermore, by using this system we are able to mimic most control algorithms developed for the Proteo-Model and similar robotic systems.

Finally, our system represents a promising and new combination of existing ideas from different research fields. There is a lot of future work to do in this field. For example we have to analyze the consequences of the limited communication range of the modules on the dynamic availability of required information. Based on these results we have to improve the shared memory of the module robot. We also have to analyze the consequences of different approaches to the representation of the building plan. However, our main research intention is the dynamic adaption of shapes according to the environment and to the tasks of a module robot. We suppose that the greatest advantage of the memetic approach lies in this dynamic adaptation.

References

Susan Blackmore. *The Meme Machine*. Oxford University Press, 1999.

Hristo Bojinov, Anarcha Casal, and Tad Hogg. Emergent Structures in Modular Self-Reconfigurable Robots. 2000. IEEE International Conference on Robotics and Automation.

Richard Brodie. *Virus of the Mind: the new science of the meme*. Integral Press, 1996.

Anarcha Casal and Mark Yim. Self-Reconfiguration Planning for a Class of Modular Robots. In *Proc. of SPIE*, volume 3839, 1999.

Richard Dawkins. *The Selfisch Gene*. Oxford University Press, 1976.

Richard Dawkins. *The Extended Phenotype: The Gene as the Unit of Selection*. W.H.Freeman and Company, 1982.

David G. Duff, MarkYim, and Kimon Roufas. Evolution of PolyBot: A Modular Reconfigurable Robot. In *Proc. of the Harmonic Drive Intl. Symposium*, 2001.

Aaron Lynch. *Thought Contagion: How Belief Spreads Through Society*. BasicBooks, 1996.

Radhika Nagpal. *Programmable Self-Assembly: Constructing Global Shape using Biologically-inspired Local Interactions and Origami Mathematics*. PhD thesis, 2001.

Kaspar Støy. Controlling Self-Reconfiguration using Cellular Automata and Gradients. pages 693–702, 2004a. 8th international conference on intelligent autonomous systems (IAS-8).

Kaspar Støy and Radhika Nagpal. Self-Reconfiguration Using Directed Growth. 2004. 7th International Symposium on Distributed Autonomous Robotic Systems (DARS 2004).

Kasper Støy. *Emergent Control of Self-Reconfigurable Robots*. PhD thesis, 2004b.

Mark Yim, Ying Zhang, John Lamping, and Eric Mao. Distributed Control for 3D Metamorphosis. *Autonomous Robots*, 10:41–56, 2001.

Emerging Artificial Societies

in the

Socially Inspired Computing Joint Symposium

Theme Preface
Emerging artificial societies

Human societies are self-organising and probably emerged in parallel with the evolution of language and development of cultural artefacts. This symposium took as its topics:

- current research about the processes and preconditions for the emergence of human societies
- experiments on artificial (i.e. computational) societies designed to shed light on generic processes of the emergence of societies
- the application of anthropological and sociological knowledge to the design of emergent societies of artificial agents
- research on self-organising societies of embedded computational agents
- architectures for computational agents capable of inhabiting such societies

By 'society', we mean here a collection of interacting (human or computational) agents that share an external symbolic system (e.g. a 'language' and cultural symbols) and which possesses social structure (e.g. normatively enforced and shared rules of behaviour). Thus, contributions which consider for example the evolution of language; the development and imposition of norms; the emergence of patterned activity and their recognition by agents; and the design of socially responsive agents will be welcomed.

This Symposium is inspired by and will be led by the EU Framework 6 project, New Ties (New and Emergent World models Through Individual, Evolutionary, and Social learning). This project began in September 2004 and one of the objectives of the Symposium is to encourage the development of a community of scholars, beyond the project itself, interested in these questions.

The New Ties project is concerned with emergence and complexity in socially-inspired artificial systems. It is studying large systems consisting of an environment and an inhabitant population. The main goal of the project is to realize an evolving artificial society capable of exploring the environment and developing its own image of this environment and the society through cooperation and interaction. The project will set up environments that are sufficiently complex and demanding that communication and cooperation are necessary to adapt to the given tasks.

The population’s capacity to develop advanced skills bottom-up consists of individual learning, evolutionary learning, and social learning. One of the main innovations of this project is social learning interpreted as passing knowledge explicitly via an evolved language to others in the same and subsequent generations. This has a synergetic effect on the learning processes and enables the society to rapidly develop an "understanding" of the world collectively. If the learning process stabilises, the collective must have formed an appropriate world map. Then the project will 'probe' the agents to learn how they perceive the environment, including themselves, and what skills and procedures they have developed to adapt successfully. This could yield new knowledge and surprising perspectives about the environment and the survival task. The project represents a significant scale-up beyond the state-of-the-art in two dimensions: the inner complexity of inhabitants and the size of the population. To achieve and explore highly complex organisms and behaviours, very large populations will be studied. This will make the system at the macro level complex enough to allow significant behaviours (cultures etc.) to emerge in separate parts of the system and to interact. To enable this we will set up a large distributed peer-to-peer computing infrastructure, and a shared platform to allow very large scale experiments.

Programme Committee for the meeting consisted of:
- Eiben, Gusz (Free University Amsterdam)
- Gilbert, Nigel (University of Surrey)
- Lorincz, Andras (Budapest)
- Paechter, Ben (Napier University, Edinburgh)
- Vogt, Paul (Tilburg University, University of Edinburgh)

Environment design for emerging artificial societies

Nigel Gilbert Stephan Schuster
Matthijs den Besten Lu Yang
Centre for Research on Social Simulation
University of Surrey, Guildford GU2 7XH, UK
n.gilbert@soc.surrey.ac.uk

Abstract

The NewTies project is developing a system in which societies of agents are expected to develop autonomously as a result of individual, population and social learning. These societies are expected to be able to solve the environmental challenges that they are set by acting collectively. The challenges are intended to be analogous to those faced by early, simple, small-scale human societies. Some issues in the construction of a virtual environment for the system are described and it is argued that multi-agent social simulation has so far tended to neglect the importance of environment design.

1 Introduction

The goal of social simulation is to develop models that shed some light on the functioning of human societies. The advantages of a simulation approach to understanding human societies include the requirement to express theories in complete and unambiguous terms; the opportunity to derive the implications of proposed social mechanisms; and the possibility of performing experiments on the simulated society (Gilbert, 2005). As a result of these advantages, there has been a rapid growth in the popularity of social simulation over the last decade (Gilbert & Troitzsch, 2005).

There are two main current approaches to the construction of simulation models of society. One approach starts with data observed or collected from a human society and tries to find a model that reproduces the observations. This approach, which generally yields results that are complex but can be compared directly with the observed data, has been labelled KIDS (Keep It Descriptive) (Edmonds & Moss, 2004). The other approach, named KISS (Keep It Simple) (Axelrod, 1997), begins by attempting to simplify the putative social phenomena to its essence and models only an abstract version of the society. The model tends to be easier to explore and understand, but validation against human societies is much harder.

This paper, like the NewTies project of which it is part[1], takes a third approach. We aim to see whether an artificial society can 'construct itself' with only the bare minimum of experimenter provided rules or theory. We take our inspiration partly from work on the evolution of language, which has shown that, given a capacity to learn, artificial agents are capable of developing a simple 'language' with which to communicate (see Cangelosi & Parisi (2002) for an overview). Initially agents utter only random noise with no information content, but through repeated interactions, some of which are rewarded, the agents gradually develop a shared lexicon ('a consensus on a set of distinctions' Hutchins & Hazlehurst, 1995:161).

If agents can develop a lexicon from 'nothing', could they also develop a shared culture? This is the hypothesis underlying the NewTies project. Taken strictly, the answer must be 'yes', since language and thus a lexicon *is* an important part of human culture. But we wish to see whether agents can develop culture in a wider sense, as a set of shared behaviours and understandings of the society and environment in which they live. This culture, like the shared lexicon, must be developed collaboratively by the agents from 'nothing'. This means that we give the agents the ability to learn, but do not

[1] New and Emergent World models Through Individual, Evolutionary, and Social learning (NEW TIES), http://www.newties.org

direct them about what to learn, and initialise them with a bare minimum of knowledge about their worlds. However, the agents are given a rich and extensive simulated environment in which they have to learn how to survive.

The next section of the paper reviews the types of learning available to agents. The following two sections introduce the environment and the challenges that the agents face. The fifth section describes the proposed interface between the environment and an agent. Agents perceive their surroundings and act in their world through this interface. The paper concludes by emphasising the importance of the design of the environment for social simulation and suggesting that this aspect has too often been neglected.

2 Learning

The agents are constructed to be able to learn in three ways:

a. Individual learning through trial and error.

Agents act according to their genetic predispositions, overlaid with random variations. Some actions are more effective than others. Those actions that succeeded in the past are remembered and the agent is then more likely to repeat those actions than the others.

b. Population learning through reproduction and selection

Agents with predispositions to carry out effective actions more frequently are more capable and are therefore more likely to reproduce, transferring a version of their genetic material to their offspring. Thus the population of agents as a whole will tend to become more successful over the course of many generations.

c. Social learning

Neither individual nor population learning require any communication between agents. However, these types of learning could be the means by which the agents begin to develop a language for communication. If they do so, they can start to use a more direct and effective mode of learning: that of one agent teaching another.

3 Environmental challenges

If agents are to learn, they must have some motivation to do so. In the NewTies project, that motivation is ultimately that of their survival. Agents are placed in a environment which they find individually and collectively challenging. Unless they master survival in this environment they will 'die'. This notion is operationalised by constructing environments in which there is a limited amount of 'food' to provide agents with energy and requiring the agents to maintain at least a minimum energy level. At first, agents have to act on their own, since they have not yet learnt to act collectively. Those that manage to collect sufficient food from the environment may survive long enough to breed, while those that are less successful are more likely to 'starve'. The environment thus imposes a strong selection pressure on the agents. Eventually, the agents may discover how to communicate and then be able to engage in collective action. This is likely to be more effective than individual acts in obtaining food from the environment.

The fine detail of the environmental challenge is an extremely important factor in the agents' development. If obtaining food is too easy, the agents will not need to learn much, and will probably not do so. If the environment is too unfriendly, all the agents will die of starvation before they have had a chance to learn anything. Secondly, if the environment requires agents to engage in activities which they are not able to carry out, the agents will surely fail, since they are only able to increase their knowledge through learning, but not their repertoire of basic actions. For example, humans have learned to fly, not by growing wings, but by learning how to build aircraft. Thirdly, the long-term objective of the research is to understand human societies better. The environment must set challenges that are analogous to those faced by humans if there is to be even the possibility of reading across from the simulation to human development.

We have designed four environmental challenges, each based on a well studied aspect of human society. In the descriptions below, the human system is first summarized, the challenge stated in terms of the simulated environment, and the observable outcome that might be expected is specified.

3.1 The Kula Ring

A complex system of visits and exchanges among the Trobriand Islanders of the western Pacific was first described by Bronislaw Malinowski (1922 [1978]). Necklaces were exchanged in one direction among the residents of a chain of islands and armbands exchanged in the opposite direction (hence the notion of a ring). These exchanges did not primarily serve an economic function but created a network of social obligations among peoples which could be depended upon at various times in an individual's life. In particular, the social network seems to have been the basis for economic relationships such as trading food for pottery.

The challenge parameters:

Food is distributed in spatial patches and the amount of food in a patch varies over time. The overall quantity is more than enough to feed the population, but there may be short-term local shortages. These can be alleviated by trading or by theft. Trade is

less costly in energy, but requires the prior development of mutual trust by the traders.
Expected outcome:
The establishment of a 'gift-exchange' system in which not only food but also tokens are exchanged.

3.2 Herders in a semi-arid area
Nomadic herding is another human solution for dealing with variable and uncertain shortages. Herders and their cattle move to where food is available, leaving exhausted areas until the grass has re-grown. This requires herders to find ways of managing common pool resources (the grass) so that no individual herder overgrazes the grass. The human solution involves well developed status hierarchies and no private property.
The challenge parameters:
Food is randomly distributed with the mean level of food just sufficient to support the population. The rate of food growth varies randomly over time. Food is perishable. Some food must be left uneaten on each patch since subsequent growth is proportional to amount of food left uneaten.
Expected outcome:
Agents leave uneaten food when they move away, even if they leave hungry.

3.3 Central place theory
Walter Christaller developed Central Place theory in 1933 (King, 1985) to explain the size and spacing of cities that specialize in selling goods and services.
The theory consists of two basic concepts:

- threshold -- the minimum market needed to bring a firm or city selling goods and services into existence and to keep it in business
- range -- the average maximum distance people will travel to purchase goods and services

The theory predicts that settlement size will follow the rank size rule. It works well for human settlements.
The challenge parameters:
The distribution of types of food is such that agents need to trade food with other agents. The food types vary in their transportability. Agents can move to find the best location to maximise their income from trade.
Expected outcome:
Agents settle into spatial clusters separated by relatively empty areas. The size of the clusters is power law distributed.

3.4 Branding
When producers produce and consumers consume complex goods (i.e. ones with a large number of distinct attributes), and there are a large number of producers and consumers, search problems occur. Producers find it hard to locate consumers that desire goods having the precise set of attributes that a producer is selling, and consumers find it hard to identify producers with the desired goods. One 'solution' to the problem each side faces is for producers to brand their range of goods (targeting them at a subset of consumers) and for consumers to use the brand as the major preference criterion. Similar processes may help to account for prejudice and discrimination among human populations.
The challenge parameters:
Agents have characteristic sensible attributes ('tags'). Agents seek to locate other agents with a similar or identical set of tags (through movement and communication), but this search is expensive. Agents are able to create additional tags (the brand) by collecting tokens and carrying them around.
Expected outcome:
Agents either generate one additional tag or specially distinguish an existing tag and this becomes a linguistic category that labels agents and leads to differences in behaviour towards those agents that are labelled and those that are not.

4 The virtual environment

An environment that offers these challenges to agents must be sufficiently rich in features to allow each challenge to be constructed, but also no more complicated than necessary. Any features beyond the minimum required would slow down the simulation and, crucially, make the agents' task of learning how to manage in the environment more difficult, because they would need to learn to disregard irrelevant features.

The environment we have designed consists of a very large simulated flat surface over which the agents are able to move. The surface is divided into small patches or 'locations'; an agent or other object is of a size that it occupies exactly one location. A virtual clock counts 'time steps', used primarily to synchronise the agents' actions. To remain in accord with the real world, agents do not have direct access to their location on the surface, nor to the time. They are, however, able to detect geographical features ('places') and the relative position of the 'sun', an object which slowly traverses the surface, crossing it once per simulated day (there is no night – the sun is always visible). Places are bounded areas of the landscape which differ from the rest of the surface in having a varied, but lesser degree of roughness, making it easier for agents to move within places than in the wilderness outside places.

On the landscape are a number of objects as well as the agents: tokens, plants, and paving stones. Tokens are distinguishable, moveable objects, some of which can be used as tools to speed up the production of food, but most of which have no intrinsic function, but can be employed by agents as location markers, symbols of value ('money'), or for ritual purposes.

Plants are the source of food. They are annuals, living for one year. At the beginning of the year, eating them gives agents little energy, but as the year progresses, they ripen and become better food. In the 'autumn', their energy value decreases again, and is entirely lost at the end of the year when they die. However, before they die, they produce two seeds, one at the parent plant's location and one in an adjacent location. If a seed is the only one in the location, it grows, but if there are more than one, only one will survive. If a plant is picked by an agent, it starts decomposing and will lose all its goodness if not consumed or replanted within a few days.

Agents lose energy (the rate depending on the roughness of the location) when they move over the landscape. The effort required to move can be reduced by building roads. Roads are constructed from paving stones laid end to end.

With these simple ingredients, we can construct scenarios corresponding to each of the challenges. For example, the Trobriand Islands can be represented as places, with the rest of the surface (having a very high value of roughness) representing the sea. The varied availability of food among the Islands (and the seasonal availability of crops) can be represented by arranging the plants in the places. The agents can learn to use tokens as symbolic gifts. Economic trading between islands could involve exchanges of food and of token tools. The other challenges could be modelled by constructing 'scenarios' in similar ways. For example, the 'branding' challenge would involve agents trading many similar but not identical tokens between themselves, with search being costly (i.e. the roads are rough).

5 Agent interface

To survive in this environment, agents need to be able to perceive the landscape and the objects in it, and also need to be able to act on objects and other agents. Moreover, it is expected that experiments will be carried out using a variety of agent designs, possibly including agents constructed outside the NewTies project, and so a simple and precisely specified interface between the agents and the environment is desirable.

At each time step, every agent is given a slice of computational resource. During this step, it must complete two phases in sequence: a *perceive* phase and an *act* phase. During the perceive phase, an agent is given the following information about the environment:

a. a list of the attributes (type, characteristics, colour, heading, and weight) of each object located within a segment defined by the direction in which the agent is facing, plus or minus 45°. The information returned about each object also includes its distance and direction from the agent and, if the object is an agent, its age and sex. These data do *not* include any direct indicator of the objects' identities; the agents have to infer these from the objects' attributes..
b. A list of the places in which the agent is located (places can overlap, so there may be more than one).
c. The agent's current energy level.
d. A list of the attributes of all the objects that the agent is currently carrying.
e. The roughness at the current location.
f. The result of the action performed in the Act phase of the previous time step, if any.
g. A list of messages that other agents have sent during the preceding Act phase.

The agent is able to process this information as it wishes, and can then carry out one action, chosen from the following:

- **Move**: The agent moves from its present location to an adjacent location in its forward direction.
- **Turn left / turn right**: the agent rotates in the indicated direction by 45 degrees.
- **Pick up object**: The agent acquires the object. The object remains with the agent until the agent puts it down or eats it (if the object is food).
- **Put down object**: The agent puts the object down at the current location.
- **Give object**: The agent transfers an object in its possession to another agent. The receiving agent must be in an adjacent location.
- **Take object**: The agent takes an object from another agent. The donating Agent must be in an adjacent location.
- **Build/improve road**: The agent builds (if there is no road already) or improves (i.e. reduces the roughness of) the road at the current location.
- **Talk to agent**: The recipient agent must be 'visible' to the speaker (An agent cannot talk to another agent while facing away from that Agent, but the hearer does not have to be facing the speaker). A character string emitted by the speaker is conveyed to the listener. The effect is that both the

listener and the speaker are given the character string during the next Perceive phase.

- **Shout:** A character string emitted by the shouter is conveyed to all agents within a short distance (including the shouter itself) during the next Perceive phase.
- **Hit**: The agent chooses, first, the amount of energy to expend on the blow, which must be less than the current energy level of the Agent, and, second, which agent will be the victim (the victim must be in an adjacent location). Both the aggressor agent and the victim lose energy proportional to the ratio of the weights of the aggressor and the victim. If the victim's weight decreases to zero or less as a result of the violence, the victim dies.
- **Eat food**: The agent must already be carrying the food (see Pick up object). The energy of the food is added to the agent's energy and the food 'disappears'.

The information given to agents about their environment is intended to reflect the information which would be available to a human. Particular care is taken not to give agents information which would not be accessible to people. For example, the identity of other agents is not provided, only some descriptive characteristics through which agents may be recognised. However, there is no guarantee that all agents will necessarily have a unique set of these characteristics. Also, in a small group, only a subset of the characteristics may in fact be needed to distinguish agents. Utterances are labelled by the system, not with the identity of the speaker, but with its characteristics for the same reason. Speakers hear their own utterances reflected back to them, again because this is the experience of humans, who are able to monitor their own speech.

Initially, agents will have no common lexicon and therefore no understanding of what other agents say to them; we expect, in the light of studies on the evolution of language, that in time the agents will develop a shared vocabulary and ultimately a shared idea of grammar (see Vogt & Divina (2005) for details on language evolution in NewTies). However, because of the design of the agents and the environment, it is not necessary or even likely that this vocabulary will be entirely composed of utterances (i.e. 'words'). Because talking is just one of the actions available to agents, it would be expected that some actions other than talking will come to take on meaning for the agents – in the same way as human gestures, for example, can substitute for or even be preferred to speech for conveying some meanings. This is in contrast to current studies of the evolution of language, which have generally taken a more purely linguistic approach to interaction.

Although the list of possible actions may seem long, it is intended to be the minimum set that would enable the challenges to be met by the agents while yielding social behaviour comparable to that of human societies. For instance, the actions 'give object' and 'take object' are required in order to make trade a possibility. Without these actions, the only way to transfer an object from one agent to another would be for one agent to put the object down and another subsequently to pick it up . However, there would be no way for the first agent to guarantee that the second agent is the recipient, and thus directed personal transfers (required for trade) would be difficult or very risky. The justification for the 'hit' action (aside from the fact that violence is an endemic feature of human societies) is that without violence, private property cannot be preserved. An agent wanting an object in the possession of another could simply remove it and the owner would have no recourse if there were no possibility of violence. To match the human situation, an aggressor will only be effective if it is stronger (i.e.. heavier) than the victim, so we can expect weak (light) individuals to be subject to theft which they cannot resist, at least until a protective social system evolves.

In this environment, agents have only one overriding 'motivation': to obtain sufficient food to survive[2]. Human requirements are of course more complex, involving not just a reasonably balanced diet, but also warmth and water, but we are assuming that 'food' is an adequate abstraction for these more complex needs.

It is intrinsic to the implementation of population learning that agents are born, reproduce and so pass on their genotype, and die. New agents result from the coupling of a male and a female agent (hence agents need to have a gender) and are born in an adjacent location to their parents. Parents have no predisposition to attend to their offspring, but because they are nearby, are likely to interact with them more than with other agents. Parental care of offspring is likely to be selected for since neglected children will find survival even more difficult than their parents (since they have had no opportunity for individual learning). To enable adults to identify children, one of the characteristic features of agents, perceptible by other agents, is their age.

6 Conclusions

We have outlined a design for an environment which can be tuned in ways that are expected to promote the emergence of agent social behaviour to

[2] There is no need for the agents to have this motive 'hard-wired' by the experimenter; agents that are not so motivated, or that are motivated to gather food, but are not effective in doing so, simply die from starvation.

solve environmental challenges analogous to those that human societies have been able to overcome.

If such behaviour does arise, the simulation could serve as an invaluable test bed for examining a wide range of social theories. Its great advantage is that while one cannot experiment on human societies, one can on artificial societies. It will be possible, for example, to determine the conditions under which particular social phenomena emerge and survive in a way undreamt of by social theorists who can observe only a small number of human societies as cases on which to test their ideas. Even these few societies have been subject to an unknown amount of cross-fertilisation (for example, it is believed that the practice of agriculture was only discovered in two or three places in the world's history; all other agriculture was learned by copying these early innovations (Smith, 1995)).

Nevertheless, there must be some caveats about making too close a link between the simulation and human societies. On the one hand, the simulated agents are lacking many of the qualities of humans, and we do not know to what extent the differences between humans and the agents are important for the generation of analogous social phenomena (for example, we noted above that the simulation does not treat 'warmth' as a distinct need for the agents, although in cold climates it is for humans).

On the other hand, what we observe in human societies is one outcome from an unknown number of other possibilities. For example, it is has been pointed out that, although most simple societies engage in some form of trade with other communities, the Kula Ring is unique. No other society has ever been discovered in which there is a two-way flow of symbolic goods. It follows that if the agent society does not generate an institution resembling the Kula Ring, this may simply be because an alternative institution has evolved, as it did in the great majority of human societies faced with similar challenges. This is of course a question that can be explored using the simulation: the experiment can be repeated many times to see whether a Kula phenomenon ever appears.

In contrast to most social simulation research, we have been almost exclusively concerned in this paper with the design of the environment; what in the environment is perceived by the agents; and the actions that the agents can take on the environment. The 'internal' design of the agents has been given little attention because it is entirely generic: agents are required to have:

- a means of generating actions as a function of their past history and current perceptions (but the form of this (phenotype) function is not of direct interest other than to the extent that it is affected by the agent's genotype),
- a genotype which, through some reproduction process, is able to generate copies with variation, and
- an algorithm for categorising objects and associating them with actions (including uttered 'words').

The details of *how* these internal processes work is little consequence for the simulations proposed here (which is not to say that these processes are trivial or easy to design). Their only important features is that they should be effective and efficient. Perhaps the fact that the agents can be black boxes, and yet the simulation can be interesting, should not be surprising, for this is the case with human societies also. We have only the flimsiest understanding of how humans 'work', yet both our social scientific and our everyday understanding of how societies work is increasingly sophisticated.

Acknowledgements

The NewTies project is partially supported by the European Commission Framework 6 Future and Emerging Technologies under contract 003752. Opinions are the authors' own and do not represent those of the Commission. The ideas in this paper have greatly benefited from interaction with other partners in the project (Vereniging voor Christelijk Wetenschappelijk Onderwijs; Eötvös Loránd University; Napier University; and Stichting Katholieke Universiteit Brabant).

References

Axelrod, R. (1997). Advancing the art of simulation in the social sciences. *Complexity, 3*, 16-22.

Cangelosi, A. & Parisi, D. (Eds.) (2002) *Simulating the evolution of language*. London: Springer-Verlag.

Edmonds, B., & Moss, S. (2004). From kiss to kids: An 'anti-simplistic' modelling approach. from http://cfpm.org/%7Ebruce/kiss2kids/kiss2kids.pdf

Gilbert, N. (2005). Agent-based social simulation: Dealing with complexity. from http://www.complexityscience.org/NoE/ABSS-dealing%20with%20complexity-1-1.pdf

Gilbert, N., & Troitzsch, K. G. (2005). *Simulation for the social scientist* (Second ed.). Milton Keynes: Open University Press.

Hutchins, E., & Hazlehurst, B. (1995). How to invent a lexicon: The development of shared symbols in interaction. In G. N. Gilbert &

R. Conte (Eds.), *Artificial societies*. London: UCL Press.

King, L. J. (1985). *Central place theory*: Sage.

Malinowski, B. (1922 [1978]). *Argonauts of the western pacific*. London: Routledge and Kegan Paul.

Smith, B. (1995). *The emergence of agriculture*. New York: Scientific American Library.

Vogt, P. & Divina, F. (2005) Language evolution in large populations of autonomous agents: issues in scaling. Proceedings of AISB 2005

Peer-to-peer networks for scalable grid landscapes in social agent simulations

B.G.W. Craenen*
B.Craenen@napier.ac.uk
B. Paechter*
B.Paechter@napier.ac.uk
*Centre for Emergent Computing
Napier University
10 Colinton Road
EH10 5DT, Edinburgh

Abstract

Recently, peer-to-peer networks have been proposed as the underlying architecture of large scale distributed social agent simulations. A number of problems arise when grid landscapes are used to represent the landscape in these simulations, primarily because, in a peer-to-peer network, the landscape has to be handled collectively by the nodes of the network. Two basic agent actions are identified as central to these problems: look and move. A solution to these problems is provided in which the network maintains a move-buffer and a look-index. Both solutions are configurable by the user of the simulation and provide a trade-off between the scalability of the system, the consistency of the information stored in the system, and the efficiency of the system.

1 Introduction

The size of the world that can be handled efficiently by a social agent simulation run on a single computer is restricted by the resources available on that computer. By combining the resources of several computers in a computer network, the efficient size of the world can be increased. These simulations are called distributed simulations. The architecture of the computer network that underlies a distributed simulation imposes restrictions on the efficiency of the simulation. In the NewTies[1] project, of which this paper is part, we propose to use a peer-to-peer (P2P) network as the underlying architecture for the simulation, since they impose fewer restrictions on the efficient size of the simulation. Social agent simulations commonly use a grid to represent the landscape on which the agents live. When an agent simulation is implemented on a single computer, a grid landscape is both straightforward and efficient to implement. In a distributed simulation this is not necessarily the case. Since a (pure) peer-to-peer network cannot have a central server, it has to partition the landscape so that it can be handled collectively by the nodes in the network. Because the peer nodes may differ in local configuration, processing speed, network bandwidth, and storage capacity, the size of the partitions can vary greatly, and can even dynamically change over time when new nodes become available and other nodes disappear. In this paper we discuss a solution to the problems that arise when a grid landscape has to be maintained in a peer-to-peer distributed social agent simulation.

This paper is organised as follows: section 2 discusses peer-to-peer networks in more detail. In section 3 we discuss how using a grid landscape in a peer-to-peer distributed social agent simulation necessitates the partitioning of the landscape over the nodes of the network and how this relates to the scalability of the network, the consistency of the information stored in the network, and the efficiency of the simulation. Two basic agent actions are identified as central to these issues: the move- and the look-action. In section 4 we discuss how the use of a *move-buffer* solves the problems imposed by the move-action. In section 5 we discuss how the use of a *look-index* solves the problems imposed by the look-action. The conclusions that can be draw from this paper are summarised in section 6.

[1] New and Emergent World-models Through Individual, Evolutionary, and Social Learning (NEW TIES), `http://www.newties.org`

2 Peer-to-peer networks

A peer-to-peer computer network is any network that does not rely on dedicated servers for communication but instead uses direct communication between clients (peers). A pure peer-to-peer network does not have the notion of clients or servers, but only peer nodes with equal functionality that simultaneously function as both clients and servers to the other nodes of the network.

The peer-to-peer network architecture differs from the client-server model in that in a client-server network, communication is usually relayed by the server, while in a peer-to-peer network, direct communication between the peers is the norm. A typical example for client-server communication is email, where the email is transmitted to the server for delivery, transmitted to the destination between servers, and is fetched later by the receiving client. In a client-server network, direct transmission from a client to another client is often impossible. Figure 1 shows a graphical representation of a simple client-server network with one server and four clients.

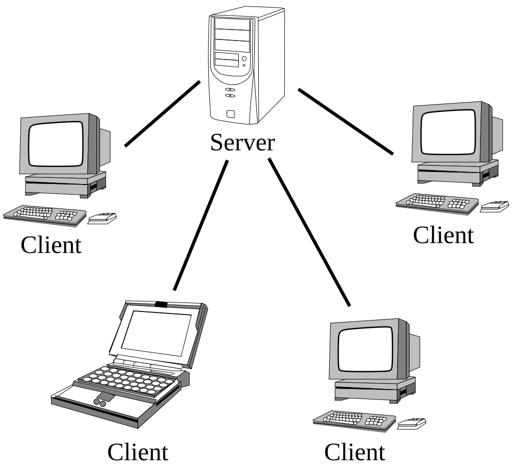


Figure 1: A client-server network

In a peer-to-peer network, any node is able to initiate or complete any supported transaction with any other node. Peer nodes may differ in local configuration, processing speed, network bandwidth, and storage quantity. A graphical representation of a simple peer-to-peer network is shown in figure 2.

There are two important properties of a peer-to-peer network that are of relevance here: the bandwidth of all peers can be fully used, and the network is able to maintain scalability when the number of nodes increases. This means that when the number of nodes in the network increases, the total available

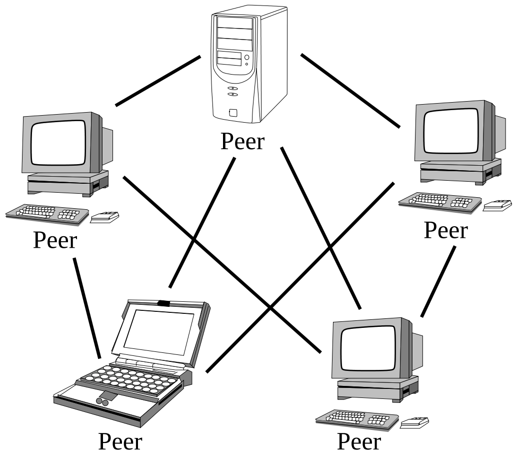


Figure 2: A peer-to-peer network

bandwidth also increases. In a client-server network, all clients have to share the (limited) bandwidth of the server, so that having a larger number of clients actually means slower data transfer.

Technically, a pure peer-to-peer application must implement only peer protocols that do not recognise the concepts of "server" and "client". Such pure peer applications and networks are rare. Most networks and applications described as peer-to-peer actually contain or rely on some non-peer elements, such as the Domain Name System (DNS) of the Internet, which translates the IP-addresses of all computers connected to it into a human-readable hostname. Also, real-world peer-to-peer applications often use multiple protocols and act as client, server, and peer simultaneously, or over time. Many peer-to-peer systems use stronger peers (super-peers or super-nodes) as servers to which client-peers are connected in a star like fashion. The inherent scalability of peer-to-peer networks and their ability to collectively use all available bandwidth has attracted a great deal of attention to their application from computer science research. The advantages of peer-to-peer networks make them an interesting alternative underlying architecture for large scale distributed social agent simulations. The complexity of an social agent simulation increases with the scale of the simulation and for the simulation of some problems, large scale simulations are required. The resource need of these simulations surpasses the resources available from a normal single computer and either the use of a super-computer or distributing the simulation over a computer network has to be considered. The acquisition and/or use of a super-computer, however, is expensive, whereas distributing the simulation over a computer network allows for the use of a range of relatively cheap computers, from resource rich desktop computers to the lowly PDA. Peer-to-peer networks allow for simulations of a scale relative to the number of nodes in the network but do not require the use of fast and/or bandwidth rich servers. In fact, peer-to-peer networks have been used to utilise unused resources on computers all over the Internet, much like in the SETI@home project (see Korpela et al. (2001) and Anderson et al. (2002) for more information). Although peer-to-peer networks are well-suited to handle large scale distributed simulations, they also pose some problems of the own.

For one, all distributed simulations must assume at least some level of unreliability in the availability of the resources in the network. Computer networks, however they are organised, consist of a collection of computers, and computers can become unavailable, can be removed from the network by the user, or can even fail. Although in modern computers and computer networks the failure-rate is small, if the number of computers in the network is large enough or the duration of the simulation long enough, some resource- or information-loss has to be expected. Several mechanisms have been proposed to limit the amount of uncertainty to which the simulation is exposed to, but all these have an adverse effect on the efficiency of the system. In general, we have to assume that a certain level of unreliability in the simulation is acceptable. This is summarised as the *unreliability assumption*.

3 Scalable grid landscapes and P2P networks

A grid landscape is a set of locations connected to each other so that together they form a grid pattern. A grid landscape is a convenient abstraction of the real-world and it is easy to implement and handle by a simulation run on a single computer. When a social agent simulation is distributed over a peer-to-peer computer network however, a grid landscape introduces a number of problems that are not apparent in a single computer implementation. Most, if not all, of these problems arise from the fact that the landscape has to be partitioned over the nodes of the peer-to-peer network. Partitioning is necessary since in a peer-to-peer network a server to handle this information centrally is not allowed: the landscape has to be handled collectively by the peer nodes of the network. In practise, partitioning the landscape means that each node in the peer-to-peer network is assigned a collection of locations that it will handle. Although it is convenient to think of these collections as clusters in the landscape, in practice, this might not be the most efficient assignment. In fact, in this paper we make no assumption about how the landscape is partitioned over the nodes of the network but instead we simply state that there exist several efficient methods for partitioning the landscape. These methods are often *self-organising* in order to maintain (at least an approximation of) the most efficient partitioning of the landscape (see Clarke et al. (2001) for more information). This means that they change the partitioning of the landscape dynamically during the run of the simulation in order, for example, to reflect changes in the network. This implies that the collection of locations handled by a single node in the network changes over time. This paper focusses on the problem of how to maintain scalability and consistency during the run of the simulation without adversely affecting the efficiency of the simulation too much.

3.1 Efficiency

The efficiency of a social agent simulation implemented on a peer-to-peer network depends to a large extent on the efficiency with which the underlying network can supply information that is needed. Environment information is requested frequently and requests — or queries — for landscape information have to be handled efficiency. In peer-to-peer terms, this means that the peer-to-peer network has to handle landscape information *discovery* efficiently. Query efficiency is measured by measuring the response-time, that is, the amount of time it takes between an issue of a query and the return of the requested information. In order to reduce query response-times, peer-to-peer networks use *information indexing*, a technique borrowed from database management systems (see Dabek et al. (2001) for more information on how information indexing is used with Chord). Indexing implies the generation and maintenance of redundant (meta) information, in order to more quickly locate pieces of information stored in the system. In peer-to-peer networks, indexing is used for two reasons: increasing network efficiency and increasing information discovery efficiency. An example of the first kind of indexing is maintaining a node-address index in order to speed-up direct communication between nodes. An example of the second kind of indexing is the maintenance of an index about certain kinds of information stored in the network in order to speed-up queries about this information. A balance has to be struck between the cost of maintaining the index and the speed-up it allows. For more information on how to improve data access in peer-to-peer systems and the use of indexing see Aberer et al. (2002).

3.2 Scalability

The scalability of the system indicates the capability of a system to increase performance under an increased load when resources are added. Scalability is a highly significant issue in databases, routers, networking, etc. A system whose performance improves proportionally to the amount of resources added to it, is said to be a scalable system. For a peer-to-peer network to be scalable the performance of the overall network has to increase proportionally to the combined resources of the nodes that are added to the network. Depending on the algorithm, peer-to-peer networks are considered to be inherently scalable, although they can be less so when the overhead imposed by the network, outweighs the resource addition to the network. In order to maintain a scalable peer-to-peer network, it is imperative that the overhead needed for maintaining the network is also scalable.

All computer networks, however they are structured, impose some amount of overhead to keep them functioning (efficiently). An example of the overhead imposed for maintaining a peer-to-peer network is the information needed to set up the direct communication between nodes (see Rowstron and Druschel (2003) for an example of how this can be done). Indexing the location of the nodes in the network is a common technique for maintaining this information efficiently (see previous section for more information about efficiency and indexing). As the number of nodes of a network increase however, there exists the possibility that the overhead of the system becomes so extensive, that to simply maintain it will take up all the newly added resources of an added node. Therefore, to allow for truly large peer-to-peer networks, the amount of overhead required to maintain them has to be minimised.

3.3 Consistency

For any simulation to provide reliable results, the consistency of the information needed to run the simulation is of paramount importance. The *unreliability assumption* already states that some inconsistency has to be tolerated when a simulation is distributed over a computer network. A peer-to-peer network itself, however, can create inconsistencies in the information stored on it. In order to explain this we have to look at how a peer-to-peer network stores information. We have already explained that a peer-to-peer network has to distribute or partition the information stored on it over its nodes. What we have not explained is how this is done. Information stored on a peer-to-peer network will normally propagate through the network until such time as the partitioning technique used determines where it will be stored (see Druschel and Rowstron (2001) and Jelasity et al. (2002) for more information). This propagation of information is done both as a load-balancing technique, i.e. making sure that the information "load" of each node is proportional to the resources available by that node, and as a way of determining if the newly available information is inconsistent with information already stored in the network. Load-balancing in a peer-to-peer network is done as part of the partitioning technique.

In a peer-to-peer distributed social agent simulation, in order to increase the scalability of the system, information about the landscape should be maintained as sparsely as possible. Only landscape locations that are needed should be maintained and the locations themselves should be created only when they are needed. In a peer-to-peer distributed social agent simulation, landscape information inconsistencies can occur whenever a new location of the landscape is created. This is because it is possible to create a location on one node that has already been created (or is created at the same time) on another. The underlying peer-to-peer network has to be able to handle these kinds of inconsistencies, for example, by keeping the earlier created location and dispensing with the newly created location. However, it has to first be aware that these inconsistencies exist, hence the need to propagate information about location creation through the network.

Propagation of information through a peer-to-peer network and the subsequent handling of any possible inconsistencies takes time. In some cases, the simulation itself is robust to some level of inconsistency of information, reducing the need handle the inconsistencies quickly. In other cases, inconsistent information can have serious consequences, placing more stringent requirements on the capabilities of the peer-to-peer network. In general however, the ability of the peer-to-peer network to handle inconsistencies incurs some overhead on the network and a balance between the efficiency and scalability of the system on the one hand and the ability to handle inconsistencies has to be found.

Just as with the *unreliability assumption*, an *inconsistency assumption* can be formulated, that is, a distributed system in which the information stored on it is totally consistent can only be created at great cost to the efficiency and/or scalability of the system. As such, any distributed system assumes that a certain level of inconsistency of information is inevitable and that the system has to deal with those inconsistencies accordingly. The unreliability and the inconsistency assumptions also have consequences for experiments that can be run on a distributed system, in that experiments in which completely consistent information is required cannot be run.

3.4 The move- and look-action

The delicate balance that has to be struck between scalability, consistency and efficiency in a peer-to-peer distributed social agent simulation involves two basic actions that an agent in a simulation can take: move and look. Most other actions, at least from an information requirement perspective, can be seen as variations on these two actions and they can be handled analogously to them.

The move-action allows an agent to move from one location in the landscape to another. In order to limit the amount of landscape information needed by the simulation, and thereby increasing the scalability of the system, the landscape should be maintained as sparsely as possible: only the locations needed are created and only the locations already created are stored. The move-action is important for the peer-to-peer network because when an agent moves to a new location — one that has not been created yet — the simulation has to create one for it. When two nodes in the peer-to-peer network need to create the same location at the same time, this can lead to inconsistency in the network. Handling the inconsistency at this time is complex and can include backtracking to an earlier state of the system or even the removal of an agent or other movable object from the simulation. We solve this problem through the use of a buffer-zone of locations around agents and other movable objects.

The look-action provides the agent with the perceptual information it needs to decide on the actions it is going to undertake. A look-action can be a conscious action for the agent to take but social agent simulations in which this information is given without a agent having to explicitly do a look-action exist as well. A look-action results in a number of information queries on the locations that the agent can see. Commonly, the number of locations visible to an agent is determined by a *look-distance* parameter and a *look-direction* of the agent. Together, they define a *look-arc* of locations visible to the agent. Since agents look often and sometimes can look far, the action produces a large amount of queries that all have to be handled efficiently. A query in a peer-to-peer network is handled by propagating it through the network. Each node with relevant information then participates towards resolving the query. In a peer-to-peer network without complete information, a query that can not be resolved takes a long time to fail, since it has to propagate throughout the whole network. In small peer-to-peer networks, the response-time, and thus the efficiency of the system, will be reasonable, but with the addition of more nodes, the system can become more and more inefficient. This straightforward implementation obviously is not scalable to a large peer-to-peer network. In this paper, we propose an alternative implementation involving predictive indexing of locations, so that the queries resulting from a look-action can be handled efficiently by the system without adversely affecting its scalability.

4 The move-buffer

The move-buffer should be seen as a buffer-zone around the locations in the landscape that are or have been used by the agents in the simulation. The buffer-zone is used as a means to ensure that consistency of information is maintained during the run of the simulation. Instead of creating locations at the moment when an agent wants to move to them, we create a buffer-zone of locations around all the locations that the agents or other movable objects have used before or are using now, so that in the case that the agents want to move into the buffer-zone, the locations are already created and the information about these locations is consistent throughout the network. The information about the locations in the buffer-zone can be propagated through the network so that the peer-to-peer network can handle any inconsistencies before an agent moves onto them.

An example of how an inconsistency can be avoided by using buffer-zones is when two agents on two landscape islands, handled by two nodes in the network, start moving toward each other. Each time one of the agents moves onto a location in the buffer-zone, the buffer-zone is extended in such a way that a certain, user-defined, distance around the agents is covered by the buffer-zone. At some point, the buffers of both nodes will include some of the same locations. At this point, an inconsistency can occur when two nodes try to create the same location at the same time. This inconsistency can be resolved by choosing one node to handle the location, and copying over information to the other node.

However, while the inconsistency is being resolved, the agents may keep moving. The buffer-zone has to cover enough locations around the agents so that the network has enough time to handle any inconsistencies that can occur before the locations in the buffer-zone are needed by the simulation.

Figure 3 shows an example of how the move-buffer is used. The small solid circles, like the one at coordinate $\langle D, 5\rangle$, indicate a location in the landscape that has been created and used by agents in the simulation. The small dashed circles, like the one at coordinate $\langle D, 6\rangle$, indicate a location in the move-buffer. The two X-s at coordinates $\langle E, 4\rangle$ and $\langle J, 4\rangle$ in the top diagram and $\langle E, 4\rangle$ and $\langle I, 4\rangle$ in the bottom diagram indicate two agents, agent X on the left and agent Y on the right. The simulation is partitioned over two nodes at the line in the middle of the two diagrams. The large thin circle indicates the buffer-zone around the agents. In the bottom diagram, agent Y has moved from coordinate $\langle J, 4\rangle$ to coordinate $\langle I, 4\rangle$. The buffer-zone is extended with the locations at $\langle I, 2\rangle$, $\langle I, 6\rangle$, $\langle H, 3\rangle$, $\langle H, 5\rangle$, and $\langle G, 4\rangle$. However, the location at coordinate $\langle G, 4\rangle$ is already in the buffer-zone of node 1, so there is a possibility that its creation has caused an inconsistency. This inconsistency has to be handled by the network, for example, by copying the information about location $\langle G, 4\rangle$ stored on node 1 over the information stored on node 2. Because the buffer-zone has a radius of two locations, the time allowed to the network for handling the information inconsistency is the same as the time it takes for the agents to make two moves towards each other.

The distance covered by the move-buffer is configurable for each experiment but should take two things into account: the size of the peer-to-peer network, and the level of inconsistency that the simulation can tolerate. The larger the network, the longer it takes for landscape information to propagate through it. When a inconsistency occurs, the time needed to propagate this message back also has to be taken into account. Some simulations are more robust to inconsistencies than others. In some, the landscape information does not have to be exact all the time. These simulations do not require an extensive buffer-zone. Other simulations however do require nearly exact knowledge about the environment, and then, the distance that their buffer-zone should cover, needs to be quite large. Since the move-buffer is part of the overhead of the system, the radius of the move-buffer has an effect on the scalability of the simulation. A larger buffer-zone means more information has to be propagated through, and stored on, the network and

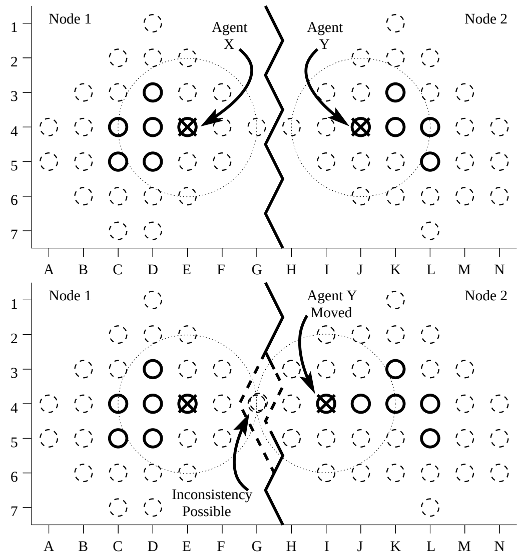


Figure 3: Move-buffer example

so more resources are needed to maintain this information. This becomes clear when we consider an extreme case, that is, when all possible locations in the landscape are in the buffer-zone at initialisation of the simulation. This will cause instantiation of the whole landscape, which depending on the maximum size of the landscape, could make the resource requirements of the network so large that it becomes impractical.

The distance that the buffer-zone covers therefore, should be configurable by the user of the simulation, because only the user can make an assessment of what kind of network is available, and what amount of inconsistency is acceptable. An interesting research question is how to assess and/or maintain the distance that the buffer-zone covers without user input.

5 The look-index

With the look-action, the efficiency of the system is the more important issue. When a peer-to-peer network has to handle a query it is resolved by propagating it through the network so that each node can provide information in order to resolve it. When the peer-to-peer network is large, it is possible that such a query will take a long time to resolve, thereby lowering the response-time of the system and thus its efficiency. When the query can not be resolved at all, for example when the necessary information to resolve it is not available in the network, the query has to propagate through the whole network and back, before its failure can be reported. The worst-case response-time of a query for information in a peer-to-peer network is therefore proportional to the size of the network.

We assume that the queries resulting from a look-action occur often during the run of the simulation. As the landscape is stored as sparsely as possible, there will be (possibly large) portions of the landscape that have not been created yet. The agents in the simulation, however, might still want to look at these portions of the landscape, and since this landscape information is unavailable in the network, we must assume that a (possibly large) number of queries will consequently fail. As a result, the efficiency of a simulation will be adversely affected. In an effort to increase the efficiency of the peer-to-peer network, we propose to index information about the landscape. If a large enough portion of the landscape is indexed, all the information needed for a query should be available in the *look-index*.

In the look-index, we index the locations that the agents might want to look at. This implies that some level of prediction of where the agents might want to look is possible. The index is generated by adding either an empty entry to the index for locations that have not yet been created or used, or the address of the node where the location is handled if it has been created or used. Instead of propagating information about the location through the network, we propagate only the newly created index entries through the network. The node in the network that handles the location then sends back its address to the node that sent the index entry. When an agent does a look-action, before any queries are sent through the network, the simulation first inspects the index. If the location corresponds to an empty index entry in the index, the network knows that the location has not been created yet and no query is issued for that location. If the location corresponds to a non-empty index entry, the network can use the node-address to set-up a direct network connection to retrieve the information about the location efficiently.

The look-index is maintained so that the information that a query can request is complete, in order to make sure that no query can fail. The size of the look-index is a trade-off between the efficiency of the simulation and its scalability. This is best explained by looking at the two extremes of the extent of the look-index. On the one hand, the look-index can extend over the whole possible landscape. This will provide complete information about the whole landscape but also means that the whole landscape needs to be indexed on every node of the simulation. Maintaining an index of the whole landscape on every node is clearly not a scalable solution, especially when large landscapes are used. The other extreme is to maintain no index at all. Although this certainly is a scalable solution, the efficiency of the simulation will suffer as a large portion of the queries issued to the network will probably fail and will take a long time do so.

The ideal extent of the index lies somewhere in between these two extremes and must be set by the user of the simulation, as only the user is able to estimate how much incomplete information is allowable and how scalable the network has to be. As a rule of thumb, however, we suggest that the look-index extends a few locations farther that the distance that the agents can look. The extra extent beyond the look-distance allows the peer-to-peer network some time to propagate the index entries through the network and also so that when two entries about the same location are created in separate parts of the network, the network can validate that the two index entries do not contain conflicting information.

6 Conclusions

In this paper we identified that when a grid landscape is used as a landscape representation in a peer-to-peer distributed social agent simulation, the scalability, information consistency, and efficiency of the simulation can be adversely affected when a naive implementation is used. Based on two basic agent actions — move and look — we demonstrate that a trade-off between these properties is possible when grid locations in the landscape are included in a buffer-zone, in the case of the move-action, or indexed, in the case of the look-action. The extents of the buffer-zone and the index should be set by the user of the simulation, since only the user can have full knowledge of the size of the network and the amount of inconsistency that is acceptable. Future research includes developing methods to estimate both parameters, either by providing an estimation at initialisation or by adjusting the parameters during the simulation itself.

Acknowledgements

The NewTies project is partially supported by the European Commission Framework 6 Future and Emerging Technologies programme under contract 003752. Opinions are the authors' own and do not represent those of the Commission. The ideas in this paper have greatly benefited from interaction with the partners in the project (Vereniging voor Christelijk Wetenschappelijk Onderwijs; Eötvös Loránd University;

Napier University; University of Surrey and Stichting Katholieke Universiteit Brabant).

References

Karl Aberer, Manfred Hauswirth, Magdalena Punceva, and Roman Schmidt. Improving data access in p2p systems. *IEEE Internet Computing*, 6(1), Jan/Feb 2002.

D. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D Werthimer. Seti@home: an experiment in public-resource computing. *Communications of the ACM*, 45(11):56–61, 2002.

I. Clarke, O. Sandbarg, B. Wiley, and T.W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *Lecture Notes on Computer Science*. Springer: New York, 2001.

F. Dabek, E. Brunskill, F. Kaashoek, D. Karger, R. Morris, I. Stoica, and H. Balakrishnan. Building peer-to-peer systems with chord, a distributed lookup service. In *Proceedings of the 8th Workshop on Hot Topics in Operating Systems (HOTOS-VIII)*, 2001.

Peter Druschel and Antony Rowstron. Past: Persistent and anonymous storage in a peer-to-peer networking environment. In *Proceedings of the 8th IEEE Workshop on Hot Topics in Operating Systems (HotOS VIII)*, pages 65–70. Schloss Elmau, Germany, May 2001.

M. Jelasity, M. Preuß, and B Paechter. Maintaining connectivity in a scalable and robust distributed environment. In *Proceedings of the IEEE International Symposium on Cluster Computing and the Grid*, page 389, 2002.

E. Korpela, D. Werthimer, D. Anderson, J. Cobb, and M Lebofsky. Seti@home — massively distributed computing for seti. *IEEE Computational Science and Engineering*, 3(1):78–83, Jan/Feb 2001.

Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. *Lecture Notes in Computer Science*, 2218(2001):329, Jun 2003.

New Ties Agent

A.R. Griffioen, M.C. Schut and A.E. Eiben
*Vrije Universiteit
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
argriffi@few.vu.nl, schut@few.vu.nl, gusz@cs.vu.nl

Á. Bontovics, Gy. Hévízi and A. Lőrincz
†Eötvös Loránd University
Pázmány Péter sétány 1/C, Budapest, Hungary, H-1117
bontovic@inf.elte.hu, gyorgy.hevizi@elte.hu, andras.lorincz@elte.hu

Abstract

In this paper we discuss issues relating to modelling autonomous agents. The agents are situated in a realistic environment where they have to survive for extended periods of time by means of learning. We present among others the architecture of the agent and propose an algorithmic solution to integrate evolutionary and lifetime learning.

1 Introduction

The New Ties project is concerned with emergence and complexity in socially-inspired artificial systems. We will study large systems consisting of an environment and an inhabitant population. One of the project goals is to realize an evolving artificial society capable of exploring the environment and developing its own image of this environment and the society through cooperation and interaction. This will set the stage for the long-term goal of the project which is to learn how to design agents that are able to adapt autonomously to, and then operate effectively in, environments whose features are not known in advance.

The emerging artificial society is a result from the capacity of an agent and the environment challenging this capacity. Notice that other agents are also part of the environment. Certain agent behaviours or cultural patterns will not emerge, if on the one hand the environment is not challenging enough or if on the other hand the starting or basic capacities of an agent are too limited. An example of the latter is that apes can reach some level of symbolic thinking after intensive training, but cannot reach the conceptual level of thought of a human being. In this paper we assume that the environment is challenging enough and focus on the agents structure.

The model of the New Ties agent is loosely based on Homo Sapiens and has some of her or his body (motor system, maximum lifetime, fertility rate) and brain capacities or features. Brain features can be carried by (parameterised) decision procedures, represented in a symbolic, neural, or any other appropriate form. Brain parameters can be part of the agents' genetic makeup that do not change during lifetime, e.g., an agent's tendency to seek mates, that can be used as a parameter within the decision procedure selecting the next action. Alternatively a parameter can be part of the agents' brain state that is changing in its lifetime, e.g., the weights of a neural net classifying other agents as friends or enemies.

Another feature of an agent brain is the learning mechanisms. The learning mechanisms are central to the "emergence engine" of New Ties, because they determine how the agent features can result in more complex (behavioral) patterns. Note that the resulting agent behaviour can be very complex, because the learning mechanisms themselves can undergo changes during the simulation. We distinguish individual, evolutionary and social learning.

Individual learning is performed by the agents independently, although possibly involving other agents. When an agent encounters a situation it does not only (re)act, but also processes the data corresponding to the situation including its own (re)action in order to improve its performance in similar situations in the future. The acquired knowledge - improved skill- will become part of the agent's "personality", it will have an impact on its behaviour. From the knowledge transfer point of view, the learning agent is a sink. Individually learned knowledge remains with the agent that acquired it, it is not passed to its offspring and in the absence of social learning it is not transferred to other fellow agents either.

Physical and mental attributes that belong to the individuals' genome are inheritable. These attributes also influence the agents' behaviour but do not change during its lifetime in a non-Lamarckian system. They undergo variation (mutation and recombination) and selection, hence they are subject to *evolutionary learning*. The learned knowledge here is in the form of superior values and value combinations for the given genes. Learning takes place at population level, good genomes are contained in well-performing individuals that obtain more offspring thus changing the allele distribution. Knowledge is transferred vertically here, down along the line of successive generations.

Social learning is approached from a new angle in our proposal. From the New Ties perspective *social learning* is interpreted as sharing the knowledge that inhabitants learn individually by explicitly "telling" it to each other, thereby collectively developing knowledge that covers different situations they are encountering. This amounts to horizontal knowledge transfer.

From a machine learning perspective we see individuals as collecting data from a given situation and learning from these data. This is individual learning. Thereafter they share the knowledge (model) they generate, but not the detailed description (data) belonging to the learning situation. They can pass the knowledge explicitly by telling each other.

The main challenge for New Ties is to generate the different emergent behaviors, including language, and social behaviors by means of evolutionary and individual learning. Thus we need to address the following issues:

- How to bridge events to be learned and remembered over time? How to store emerged knowledge? (*Problem of time*)
- How to separate complex societal learning, including the emergence of language from individual learning? (*Problem of separation*)
- How to design a learning model for an agent that enables communication? (*Problem of communication*)
- On the architectural level, how to design a consistent architecture for these distinct learning types? (*Problem of consistency*)

In this paper we describe our unified approach to three types of learning methods, The basic design of the New Ties agent is an extension of decision trees, which was made to fit evolutionary algorithms (EA) as well as reinforcement learning (RL), and which is general enough to incorporate tunable function approximators (Haykin, 1999). For details on evolutionary, individual learning and emergence of language, see, e.g, Eiben and Smith (2003), Sutton and Barto (1998), and Cangelosi and Parisi (2002), respectively.

The next section (Section 2) describes the agent in detail. The novel algorithmic construct of the agent is discussed in Section 3. We finish the paper with a discussion.

2 Agent description

In this section, the architectural structure of the agent is provided. We describe the agent architecture after the Wooldridge and Jennings (1995) notion of agent. We discuss each property or feature of the notion of an agent for the New Ties agent and give thereafter additional features of the New Ties agent. First, we will give a short description about the simulated world of the agent.

The world is based on a grid with locations, roads, plants, tokens, and agents. Locations have a 'placeness' feature, that indicates for example a mountain. Roads have a roughness feature that can be changed by the agents. Plants are the (only) energy source for the agents. Tokens are rare objects that have no (built-in) function. It is, however, expected that the agents will assign to them some function like a trading function. Agents can collect tokens and store them in a bag that each agent carries with it. Every object in the world has attributes like shape and colour. Another attribute of an object is 'characteristics'. This attribute increases the probability that the object together with the other attributes can be uniquely identified, although there is always a small probability that some objects may be similar. For more details on the environment we refer to Gilbert et al. (2005b)

The Wooldridge and Jennings (1995) notion of an agent has the following four features:

- autonomy: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state. New Ties agents possess a controller that determines their actions.
- social ability: agents interact with other agents (and possibly humans) via some kind of agent-communication language.
- reactivity: agents perceive their environment, (which may be the physical world, a simulated

world, a user via a graphical user interface, a collection of other agents, the Internet, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it.

- pro-activeness: agents do not simply act in response to their environment, they are able to exhibit goal-directed behaviour by taking the initiative.

The New Ties agent possesses *autonomy*, because its decision process is not determined by any user, but determined by the agent's controller. The "controller" denotes that entity (procedure, algorithm, program, etc.) that chooses the next action of an agent. This choice is influenced by many factors, the controller has many input parameters. We will address these parameters in more detail at the end of this section.

The New Ties agent has a *social ability*, because it can communicate with other agents by means of talking (one-to-one communication) and shouting (one-to-many communication). This communication can change the behaviour of an agent. Initially, agents do not have a common communication language, but it is expected that populations of agents will develop a (common) language during a simulation. Agents do have the possibility, however, to communicate with elementary signals that correspond to body language such as humans may have used when no language was available.

The New Ties agent is *reactive*, because it can perceive its environment. It can see objects located within a segment defined by the agent's heading, plus or minus 45° and the distance. Objects behind coloured (non-transparent) objects are hidden from the agent's gaze and are therefore not included in the list. The information provided for each object is a list of

- its shape, age, characteristics, colour, heading, and weight,
- if the object is another agent, the agent's sex and contents of its bag, and
- the distance and direction (to the nearest 45°) from the agent to the object.

Note, that the agents cannot identify objects directly. They can only see bundles of object features that they have to learn to classify. Agents do not only have to learn to classify between categories - for example between food and agents, but also within categories, i.e. different types of foods or agents.

An agent can hear all messages agents have sent using the Talk or Shout actions within its hearing range, which is a circle with radius r.

Other information that an agent receives about its environment, is

- a list of the Places in which the current location of the Agent is found,
- the Agent's current energy level,
- the roughness at the current location,

Agents cannot perceive everything in the environment such as the energy levels of the other agents. Gilbert et al. (2005a) give more details about what an agent can perceive and receive.

The New Ties agent is *pro-active*, because it is constantly engaged in goal-directed behaviour. It looks for food, a mate, and is trying to stay alive. It can achieve its goals by means of actions. There are motor actions that allow it to move in the environment, to pick-up or put down objects (given that the objects are not too heavy), to hit something, to build roads, to eat, and to mate. An action related to mating is the 'insert agent' action of which delivering a baby is the human or vertebrate equivalent.

In addition to the four features of the Wooldridge and Jennings (1995) notion of an agent, agents have a lifetime, a gender, genetic features, and lifetime learning properties.

The *lifetime* of an agent consists of three periods: a childhood period, an adult period, and a period of old age. Each period has its own properties. In childhood an agent tends to follow one of its parents. As an adult an agent can reproduce. When an agent reaches old age this ability is lost

The *gender* of an agent plays a role in reproduction: only agents of opposite sex can have children. The choice of a partner is not random, but based on the social network of an agent, which it has built during its life (for more details on social networks in New Ties see Vogt and Divina (2005)).

The *genetic features* of an agent may be passed on with reproduction. A child's gene is created from the parents' genes by recombination and mutation. The properties of an agent that we encode in the genome are summarized in Table 1 below. Note that all values can be scaled except the values of 2 and 3. These have values matching our intuition on humans and years.

The New Ties agent has *lifetime learning* features that we have named individual and social learning. Social learning is the explicit passing of knowledge, i.e. telling each other, in a horizontal way. This type of social learning starts with language development. Information is transferred by 'telling' something to (an-)other agent(s). Though the information may not

Table 1: The genome of an agent

Genome of the agent nr	Name	Type	Value	Additional comments
1	Metabolism	real	[0,1]	Determines how how much food is converted into energy.
2	OnsetTimeAdulthood	int	[15,25]	Reaching the adult age the agent becomes fertile
3	OnsetTimeElderdom	int	[55,65]	Reaching elderdom the agent loses fertility.
4	Socialness	real	[0,1]	This determines the degree to which an agent wants to interact with other agents.
5	FollowBehaviour	real	[0,1]	This determines the degree to which an agent wants to follow its parents
6	MaxVisionDistance	real	[0,1]	Determines how far an agent can see.
7	InitialSpeed	real	[0,1]	Initial distance an agent can walk per time step
8	InitialStrength	real	[0,1]	Initial weight an agent can lift.
9	MaxShoutDistance	real	[0,1]	The maximal reach of an agent's shout

be immediately informative, because mutual agreement about language utterances has not yet developed, it is valuable for other agents, as it is being used to build the language knowledge base.

The language learning mechanism consists of several sub-mechanisms. One of them is the mechanism to build a social network. Another sub-mechanism is the discrimination game that is used for the categorisation of the incoming feature bundles into object categories. For more details about the language development mechanism we refer to Vogt and Divina (2005).

Individual learning is the knowledge acquired in every situation that an agent (re-)acts and processes data corresponding to that situation - including its own (re-)actions in order to improve performance in similar situations in the future. Individual learning changes the behaviour of the agent, which means that the agent gives a different action or sequence of actions in a similar situation or with a similar stimulus. Learning may take place over a longer time as a stimulus is not necessarily in the previous time step, but may have happened some time steps ago.

Individual learning is changing the the controller of the agent to optimize its behaviour. Many parameters influence the decisions of the controller. We distinguish three types of parameters influencing the decision process, namely: direct information, information from lifetime learning, and genetic information.

Direct information is information from the last time step coming from the environment or from the controller as feedback. Within direct information different types of information can be distinguished. We depict in Figure 1 the different types of information streams in an agent. The short-term-memory (STM) holds all direct information. It consists of information listed in the reactivity feature of the agent such as visual and auditory information. This information goes to the controller. Note, that visual information is first transformed by the categorisation process, before it goes into the controller.

Lifetime learning and genetic information are part of the controller. If the controller is a rule-based system, lifetime and evolutionary learning determine the conditions in the rules. They optimize the learning system by changing the conditions. We will use the reinforcement learning mechanism for individual learning.

In the next section, we will explain how all learning mechanisms, including social learning, can be integrated if we use a tree structure as a representation for the controller.

3 The algorithmic construct: Decision Q-trees

The algorithmic construct is designed to meet the challenges defined in the Introduction section. To this end, we have extended the concept of decision trees. This novel concept is called Decision Q-tree (DQT). It is depicted in Fig. 2.

We shall use the following notation: $\mathcal{A}$, $\mathcal{S}$, $\mathcal{P}$, and $\mathcal{R}$ denote the set of possible actions, the set of states, the set of decision making rules, and the set of possible rewards, respectively. Possible rewards are assumed to be bounded real numbers.

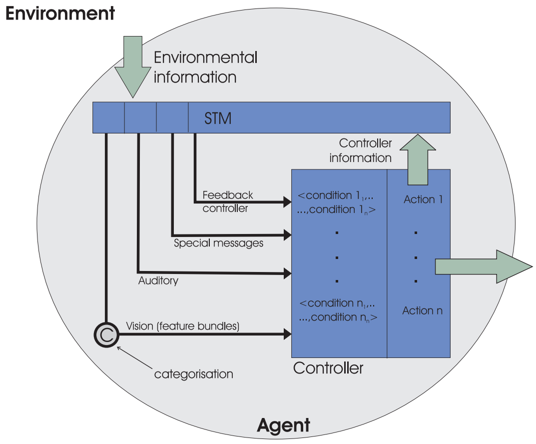


Figure 1: Agent information streams

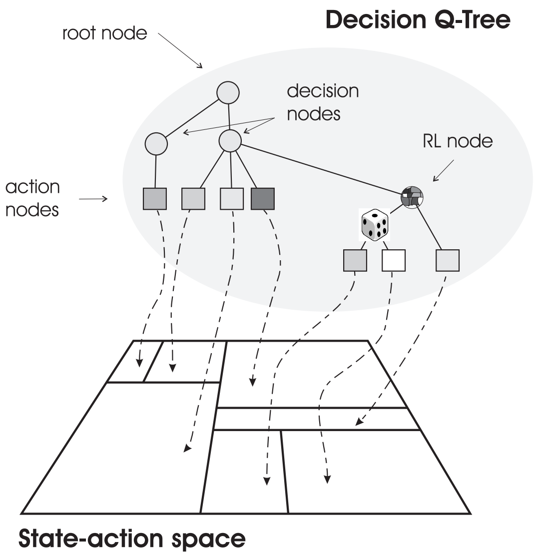


Figure 2: **Decision Q-tree.** DQTs are decision trees with the following features: (1) nodes can denote rules, indices of states, and actions, (2) decisions can be stochastic (denoted by the dice), (3) nodes are assumed to have values (denoted by the gray levels of the nodes). Special cases of DQTs among others are: decision trees, action-state value tables of reinforcement learning, Bayesian decision networks. DQTs have large compression power in state-action mapping and suit evolutionary algorithms. DQT partitions of the state-action space can be overlapping.

3.1 Definitions

A *DQT is a value table* with value entries belonging to the nodes of a decision tree. Graphically, DTQ is a tree made of N nodes n_i $(i = 1, \dots N)$ and directed edges[1] $e_{ij} \in \mathcal{E}$ $(i \neq j = 1, \dots N)$ connecting the nodes: e_{ij} points from node n_j to node n_i. To each node $n \in \mathcal{N}$, which is not a leaf, there belongs a set of decision making rules $(\mathcal{P}_n)$. Decision making rules of a node in the DQT can be deterministic, e.g., *if-then rules*, or *tables*, can be *stochastic rules*, or can be assumed to have *network structures*, such as Bayesian networks. If the rule set of a node has k different cases, then there are k outgoing edges of that node. Every node on the tree is a Q-node in our formulation, i.e., to every one there belongs a set of states (this rendering could be implicit) and a set of actions, the actions of the leaves belonging to that node. Nevertheless, we distinguish Q-nodes from RL nodes. The distinctive feature of an RL node is that it links every action to a single state, and not a state set. Besides, a RL node can be stochastic, i.e., more than one action may belong to that node. If the RL node represents a single state in state space, then we call it a true RL node. Each leaf of a DQT contains a single action, which can also be the nil action. Nodes of the graph have values denoted by gray scale in Fig. 2. Values of the nodes are computed by means of the rewards as follows:

Let us denote the actual tree by $\mathcal{T}$. The value $Q_n^{\mathcal{T}}(t+1)$ of node $n \in \mathcal{N}$ after the $(t+1)^{st}$ encounter of that node is the average of the cumulated discounted returns $R_n^{\mathcal{T}}$ experienced after all encounters:

$$Q_n^{\mathcal{T}}(t+1) = \frac{t}{t+1} Q_n^{\mathcal{T}}(t) + \frac{1}{t+1} R^{\mathcal{T}}(t+1) \quad (1)$$

$$R_n^{\mathcal{T}}(t+1) = r_{t+1}^{(1)} + \gamma r_{t+1}^{(2)} + \gamma^2 r_{t+1}^{(3)} + \dots \quad (2)$$

where $r_{t+1}^{(1)}, r_{t+1}^{(2)}, \dots$ denote the immediate rewards received in temporal order after the $(t+1)^{st}$ encounter of node $n \in \mathcal{N}$. $0 < \gamma \leq 1$ is the discount factor.

3.2 Relation to other methods

Fitness value of an individual (i.e., an agent, or a tree) $f_{\mathcal{T}}$ is the cumulated returns belonging to the root node of tree $\mathcal{T}$ possibly with no discounting:

$$f_{\mathcal{T}} = R_{root\,node}^{\mathcal{T}}$$

for $\gamma = 1$. This is the value that corresponds to the fitness concept of GA.

Policy of the agent: In reinforcement learning, policy is the mapping of states to actions. In turn, the

[1] When the tree is hierarchically depicted, then the direction of the edges is unambiguous and it will not be shown.

policy is the equivalent of the decision process represented by the tree structure.

Action value function ($Q : \mathcal{S} \rightarrow \mathrm{R}$) for a given policy in RL: Our definitions were made to meet the definition of the action value function of RL. The values of the leaves contain the values of the action-value function for policy represented by the tree $\mathcal{T}$, provided that each leaf belongs to a single state. Note that our definition is a generalization of the action value function of RL: here a value belonging to a node of the tree is the occurrence-frequency weighted evaluation of state-action pairs, for the states the agent encountered and for the actions executed by that node or the children of that node. In turn, values of nodes of the DQT evaluate state-action sets and thus generalize the action-value function of RL.

RL and Holland's Learning Classifier Systems (Holland et al., 2000) have been contrasted in the literature (Kovacs, 2001). The main difference is in the way these methods use and modify the policy, the state-action mapping. We unify these distinct algorithms via our DQT architecture. The smoothness of this unification underlines the similarities, instead of the differences of these two paradigms. We expect to make use of the strengths of both algorithms, such as the firm mathematical basis and polynomial time convergence properties of RL and the compressed representations searched for and offered by GA.

3.3 Relevance from the point of view of the challenges

Here we would like to recall the problems of time, separation, distribution and consistencey, which were mentioned in the first part of this paper. DQTs are made to fit RL, incorporate the concept of fitness and are attractive from the point of view of all types of learning, including evolutionary learning, individual learning and social learning, thus providing a possible solution to the consistence issue.

We have chosen to represent the "brain" of the agents like this, because every positive feature of trees can be adapted to this model and it is not limiting. Trees are evolvable structures in the sense that there are well-established solutions for mutating and recombining tree-based genotypes (for more information about genetic programming see Banzhaf et al. (1998)). Therefore, our representation is not preventing evolution. *Evolutionary learning* makes use of the fitness value and utilizes common variation operators in genetic programming, adjusted when needed.

Individual learning occurs through exploration, i.e., the application of new actions, the evaluation of

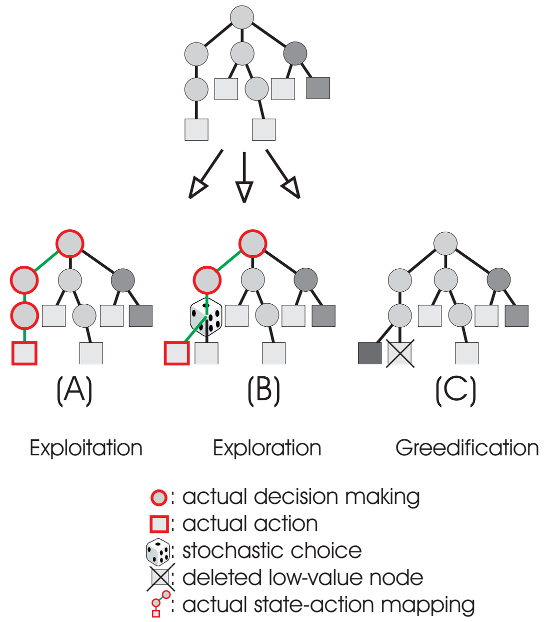


Figure 3: (A): exploitation using DQT, (B): exploration, and (C): "greedification" of a DQT

these new actions and the "greedification" of the policy (here the tree), i.e., the deletion of actions that have low values. These operations are depicted in Fig. 3. Fine tuning the values of the tree nodes is part of the individual learning process, thus it can be separated from social learning and language (Problem of separation).

Thus we get a seamless integration of RL and evolutionary algorithms by Decision Q-Trees.

Language games (Vogt and Divina, 2005) can be covered by generalizing the concept of the nodes: one or more nodes of the DQT can include a neural network, e.g., a Bayesian network. Then this architecture meets the requirements posed by language games. What we gain from our approach is that the agent can continuously evaluate the behavioral – individual – relevance of the learned language, a fine detection method during the emergence of language.

Communication between individuals can be modeled by DQTs as follows: queries about state-action mappings to particular states or sets of states can be queried and these mappings can be communicated between the agents. This way experiences of different agents can be collected and tried in similar or different environments. It is implicitly assumed that language has already been established up to some extent. Thus, our model formulates the problem of the emergence and the development of language as follows:

1. Create joint/similar representation about states and actions.

2. Develop the algorithms for querying and exchanging series of events together with their values.

These efforts aim to develop language for passing relevant experiences in a compressed form.

Note that DQT is an economical representation of the action-value function, the possibly enormous look-up tables when all states are listed. Further compression algorithms for decision trees exist (Carvalho and Freitas, 2004) and can be applied here. Taken everything into consideration, we expect that this novel DQT paradigm can meet the challenges.

4 Discussion

The New Ties-project is not restricted to the specific types of proposed individual and evolutionary learning algorithms. This paper is the start of investigating individual, evolutionary and social learning as described in this paper. We will investigate how the strengths of each individual algorithm and their interaction effects can be utilized to build controllers for autonomous robots that can survive and explore unknown environments. The ideas and resulting algorithms can be applied to robots like swarm-bot robots (see for example Mondada et al. (2002)) to make them more adaptable. The algorithms can also be used in soft-bots that autonomously explore databases and collect relevant data, e.g. finding in a gene database interaction effects between genes and gene sequences.

This work may also be relevant for social and cognitive sciences if the New Ties system architecture and parameter settings resemble sufficiently the systems to be investigated. For example, we can address questions such as whether the agents' capacities and learning algorithms are sufficient for the emergence of social patterns like cooperation and trading.

As we are just at the start of our investigations there are many possible agent designs. Future experiments will judge whether and guide the design of the agent architecture and the combination of individual, evolutionary and social learning are technically and scientifically worthwhile and fruitful. Our challenge will be to design state-of-the-art autonomous agents that are able to overcome the learning challenges and possible answering scientific questions.

Acknowledgements

Careful reading of the manuscript by Katalin Lázár is gratefully acknowledged. The NewTies project is partially supported by the European Commission Framework 6 Future and Emerging Technologies programme under contract 003752. Opinions are the authors' own and do not represent those of the Commission. The ideas in this paper have greatly benefited from interaction with the partners in the project (University of Surrey; Napier University; and Stichting Katholieke Universiteit Brabant).

References

W. Banzhaf, P. Nordin, R.E. Keller, and F.D. Francone. *Genetic Programming: An Introduction*. Morgan Kaufmann, San Francisco, 1998.

A. Cangelosi and D. Parisi, editors. *Simulating the evolution of language*. Springer, London, 2002.

D. R. Carvalho and A. A. Freitas. A hybrid decision tree/genetic algorithm method for data mining. *Inf. Sci.*, 163(1-3):13–35, 2004. ISSN 0020-0255.

A.E. Eiben and J.E. Smith. *Introduction to Evolutionary Computing*. Springer, London, 2003.

N. Gilbert, S. Schuster, and M. den Besten. Notes towards the newties environment. Technical report, Centre for Research on Social Stimulation, University of Surrey, 2005a.

N. Gilbert, S. Schuster, M. den Besten, and L. Yang. Environment design for emerging artificial societies. *Proceedings of AISB 2005*, 2005b.

S. Haykin. *Neural networks: A comprehensive foundation*. Prentice-Hall, Upper Saddle River, NJ, 2nd edition edition, 1999.

John H. Holland, Lashon B. Booker, Marco Colombetti, Marco Dorigo, David E. Goldberg, Stephanie Forrest, Rick L. Riolo, Robert E. Smith, Pier Luca Lanzi, Wolfgang Stolzmann, and Stewart W. Wilson. What is a learning classifier system? In *Learning Classifier Systems, From Foundations to Applications*, pages 3–32. Springer-Verlag, 2000. ISBN 3-540-67729-1.

T. Kovacs. Two views of classifier systems. In *Fourth International Workshop on Learning Classifier Systems - IWLCS-2001*, pages 367–371, San Francisco, California, USA, 7 2001.

F. Mondada, G. C. Pettinaro, I. Kwee, A. Guignard, L. Gambardella, D. Floreano, S. Nolfi, J.-L. Deneubourg, and M. Dorigo. SWARM-BOT: A swarm of autonomous mobile robots with self-assembling capabilities. In C.K. Hemelrijk and E. Bonabeau, editors, *Proceedings of the International Workshop on Self-organisation and Evolution of Social Behaviour*, pages 307–312, Monte Verità, Ascona, Switzerland, September 8-13, 2002. University of Zurich.

R. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, 1998.

P. Vogt and F. Divina. Language evolution in large populations of autonomous agents: issues in scaling. *Proceedings of AISB 2005*, 2005.

M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *In Knowledge Engineering Review*, 10(2):1–62, 1995.

Language evolution in large populations of autonomous agents: issues in scaling

Paul Vogt[⋆†]
[⋆]Language Evolution and Computation Research Unit, University of Edinburgh
40 George Square, Edinburgh, EH8 9EU, Scotland.
paulv@ling.ed.ac.uk

Federico Divina[†]
[†]Induction of Linguistic Knowledge / Computational Linguistics, Tilburg University
P.O. Box 90153, 5000 LE Tilburg, The Netherlands
divina@cs.vu.nl

Abstract

In this paper we discuss issues relating to modelling language evolution in large populations of autonomous agents that are situated in a realistic environment where they have to evolve and learn means to survive for extended periods of time. As we intend to build such a model in relation to the recently started New Ties project, we identify three major problems that are expected for such a model. The paper proposes some solutions and discusses future directions.

1 Introduction

Language evolution is a *hot* topic in today's sciences; especially in the field of computational modelling, see, e.g., (Cangelosi and Parisi, 2002; Kirby, 2002) for overviews. Typically, the computer models studied are simple, clear and provide useful insights into the origins and evolution of language. However, language is a complex phenomenon and this paper provides an outlook toward more complex models of language evolution.

The computational studies that have been proposed and studied so far have been very useful in investigating particular questions raised by theorists and empiricists in related disciplines, e.g., (De Boer, 2000) and sometimes these studies even have developed new hypotheses (Steels, 1998; Kirby and Hurford, 2002).[1] One limitation of today's state-of-the-art, however, is that most studies only focus on one, or possibly a few aspects of language evolution. This, in itself, is not problematic, but the models that are used to study these aspects typically discard all (or at least many) other aspects in their models, most notably those aspects that have some additional form of complexity with it.

For instance, the studies presented in Vogt (2000) have investigated how a population of physical robots could develop a shared communication system that was perceptually grounded in their environment. However, the population in these studies was of size 2, the agents only communicated about 4 objects that were always present in a context, there was no population turnover, there was no grammatical structure in the communication system and there was no ecological function for the language. These studies have gradually been increased in terms of, e.g., larger population sizes and the number of objects – though without perceptual grounding (Vogt and Coumans, 2003), or evolving simple grammars – though still with small populations of size 6 (Vogt, 2005).

These issues, however, are not really points of critique, but merely an observation of the state-of-the-art. Refraining from complex models is very useful and justifiable. For instance, increasing the number of aspects that one includes in his studies will increases the complexity of one's models in terms of degrees of freedom of, e.g., learning, interactions, analysis and – very important – computational power. So, looking at one – or few – aspects of language evolution has many advantages and allows one to investigate structurally what happens inside his models. However, the limiting complexity can have a pitfall for our studies.

[1]Note that these and other studies on which we base our arguments are selected for their high influence in the field. The critiques (or comments) made on these studies apply to all other modelling studies published so far. It should also be noted that although the critiques given are negative, this does not mean that we do not appreciate, like or even adhere to the studies discussed.

For instance, the assumption of using a population of size 2 (cf. Kirby, 2001) or ignoring generational turnover (cf. Steels, 2004) can have a huge impact on the qualitative results of the experiments (Smith and Hurford, 2003; Vogt, 2005). (Note that the studies that discovered such flaws themselves ignore other aspects that, undoubtedly, will lead to qualitatively different results as they too are limited in their set up.)

To what extent then, do we need to complicate our models in order to become more realistic and achieve results that are more likely to be alike real language evolution? The most perfect model of real human language evolution would be the result of reconstructing the real thing. This, however, is not what we want – even if we could do it. However, we should attempt to build models that are beyond our current level of complexity to allow testing hypotheses in large scale simulations that take into account more degrees of freedom in order to become more realistic with respect to the current models. Our aim with the recently started *New Ties* project[2] is to implement a benchmark simulation that allows a level of complexity far beyond the current state-of-the-art.

In the next section, we will briefly introduce the New Ties project and address some problems we think we will encounter. We will discuss how we think we can tackle some of these problems in Section 3. Finally, Section 4 concludes.

2 Identifying the problems

The New Ties project aims at setting up a large scale multi-agent simulation in which the population is to learn and evolve a social culture and individual capabilities that enables them to (co-)operate viably in their environment.[3] The environment will be modelled loosely after the Sugarscape environment (Epstein and Axtell, 1996), which will have a spatial grid, different food sources, tokens, different types of terrain and a large population of agents (Gilbert et al., 2005). We assume that the agents have capacities that will loosely reflect the capacities of early homo sapiens. The agents, which are genetically specified, are supposed to develop a repertoire of behaviours that allow them to survive for extended periods of time. The aim is to have these behaviours develop through individual adaptation, cultural and genetic evolution. The environment will be constrained in such a way that the most efficient way to survive is to develop co-operation. We allow the agents to evolve language such that they can improve on co-operation.

Although eventually the aim is to have the population evolve a drive and means to evolve language, we will start by assuming that they have this drive and means. This leaves us with the non-trivial problem of having the agents develop a shared communication system. Before identifying some of the problems, it is important to realise that each agent starts its lifetime without any knowledge about the world, so it has no representations of meaning and language. It is also important to mention that each agent acts autonomously; there is no form of telepathy or central control regarding the behaviour of agents. We have identified three major problems we have to deal with in New Ties:

1. At each moment in time we aim to deal with a population of around 1,000 agents or more. No known experiment in language evolution has had such a large population size. It is expected that having all agents interact with all other agents leads to an unrealistic scenario and requires a huge number of interactions to arrive at a shared language. However, the agents are distributed spatially across the environment and we do not expect them to travel fast, so the likelihood they will meet every other agent during a lifetime is expected to be low. Nevertheless, we do want them to mix to some extent, but we also believe that learning language in small communities is both realistic and more efficient. So the problem is, how do we control communication?

2. There are a relatively large number of different objects (food items, tokens, agents, roads, places etc.), which are perceived by agents through a fixed, but relatively large number of feature channels. In addition, there are many actions that can be performed. How do we allow agents to categorise the different objects/actions such that they become sufficiently similar to allow language learning (cf. Smith, 2003), and such that these categories are not predefined (i.e. there is typically no one-to-one relationship between object and category)?

3. The contexts in which communication take place are acquired by the agents autonomously. As a result, they may differ from one individual to another (see Fig. 2). In addition, the languages of two individuals may differ, for instance because one of the individuals is still a 'child'. In brief: if a speaker communicates about one object in its context, how will the hearer infer its reference?

[2] See http://www.newties.org.

[3] In order to deal with the computational complexity of such a large scale simulation, 50 computers will be connected through the Internet in a peer-to-peer fashion.

And, how do the agents infer the effectiveness of the interaction? These problems are loosely related to what is known as the *Theory of Mind* (ToM).

The next section will present some directions we propose as solutions to this problem.

3 Proposed solutions

3.1 Large populations

In order to deal with large populations, we decided not to treat it as a problem. Instead, we regard it as an asset with which we can learn about how different dialects and languages may evolve. Nevertheless, we do not want each agent in the population to communicate with all other agents, as we believe this will give us huge convergence problems. In addition, we do not want each agent to communicate unrestrictedly with another agent, as this may lead to unlimited chat sessions among agents who happen to be near to each other.

When an agent S sees another agent A in its visual field, it will evaluate, for each object o_i in the context, the function:[4]

$$f(A, o_i) = \nu_1 \cdot SB(A) + \nu_2 \cdot strA(o_i) + T_0 \quad (1)$$

where ν_1 and ν_2 are weights, $SB(A)$ is the *social bond* of S with A, $strA(o_i)$ is the *attention strength* of object o_i, and T_0 is a *talkativeness* parameter.

In order to favour communication with close kin and 'friends', we introduce a *social bond* variable $SB(A)$, which is based on the social network an agent constructs (Fig. 1). $SB(A)$ is a function that is proportional to the number of interactions between two agents (it is assumed that agents can recognise each other) and the effectiveness of such interactions (cf. Gong et al., 2004). The relation between parents and offspring will be treated separately. It is assumed that kinship innately promotes $SB(A)$ and may be regulated genetically.

The attention strength $strA(o_i)$ is based on a (possibly large) range of aspects occupying an agent with respect to one of the objects o_i in the agent's context. For instance, if the agent is hungry, has no food, but sees that another agent carries a food item F, $strA(F)$ gets a high value. The function is part of the agent's ToM and is defined in Section 3.3, Eq. (2).

[4]For the current presentation we are only discussing objects to communicate about. Actions will be treated similarly, but are set aside for the time being.

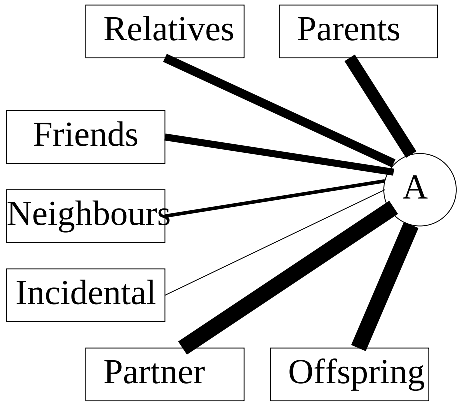


Figure 1: An illustration of an agent's (A) social network. The thickness of the lines indicate the *social bond* $SB(A')$ of A with another agent A'. See the text for more details.

The *talkativeness* T_0 is a bias of the agent to communicate. This bias may be genetically specified, but may also be based on learning that communication is useful.

The agent determines which object in the context yields the highest attention strength and the result of $f(A, o_i)$ will be forwarded to an *action decision* mechanism that evaluates which action the agent will take. In this action decision mechanism, the action *communicate* will compete with other possible actions, such as *move-forward* or *eat*. If the agent now communicates about o_i, $f(A, o_i)$ will temporarily remain low for o_i afterwards in order to prevent unrestricted communication.

Given these mechanisms, we expect that there will emerge a self-regulating communication drive, which has a bias to communicate in small communities, but does not exclude communication outside such communities.

3.2 Categorising objects

Categorisation of objects will be based on the *discrimination game* model (Steels, 1996a) and implemented using a form of 1-nearest neighbourhood classification (Cover and Hart, 1967). The aim of the discrimination game is to categorise the object such that it is distinctive from other objects that are in the agent's context. This need not require that the category is a perfect exemplar of the object. Each object has a number of perceptual features, e.g., shape, colour, weight, location. Objects of a different type

object	f_1	f_2	f_3	f_4	f_5
A_1	0.3	0.7	0.2	0.4	0.1
A_2	0.8	0.2	0.2	0.4	0.5
T_1	0.0	0.0	0.4	0.3	0.3
T_2	0.0	0.0	0.4	0.2	0.5
T_3	0.1	0.5	0.4	0.3	0.3
F_1	0.0	0.0	0.8	0.7	0.8
F_2	0.9	0.3	0.8	0.9	0.8
F_3	0.0	0.0	0.8	0.8	0.6

Table 1: An example context with objects having 5 features f_i. The objects include 2 agents (A_i), 3 tokens (T_i) and 3 food sources (F_i). The features can be perceived as direction (f_1), distance (f_2), shapes (f_3), colours (f_4) individual characteristics (f_4).

may have exactly the same features when they are on the same location. Some objects of the same type, e.g., agents, have the same features in some dimensions, but differ in others to allow identifying individuals.

Table 1 shows an example context containing 8 objects, each perceived with 5 features. Each object has a direction and distance with respect to the perceiving agent (features f_1 and f_2). The objects T_1, T_2, F_1 and F_3 have $f_1 = f_2 = 0.0$ indicating they are carried by the agent in a 'bag'. All objects of the same type have the same shape (f_3) and often the same colour (f_4). Although for most objects colours are fixed, the colour of food sources (F_i) change over time, indicating the freshness and nutrition of the food. Although individual characteristics (f_5) may be the same for different individuals of the same type, they are typically distinct; this is more so the case for agents (A_i) than for tokens (T_i) or food sources. Across different types, similar individual characteristics can serve as a perceptual basis for analogies.

Each object is categorised by finding for each feature the nearest *categorical feature* $c_{i,j}$, which when combined forms a category (cf. Vogt, 2005). Here i refers to the feature dimension, and j is the j-th categorical feature in that dimension. Suppose an agent's repertoire of categories (or *ontology*) includes categorical features $c_{3,1} = 0.2$, $c_{3,2} = 0.5$, $c_{4,1} = 0.3$, $c_{4,2} = 0.4$, $c_{4,3} = 0.7$ and $c_{4,4} = 0.85$. Then objects A_1 and A_2 are mapped onto categorical features $c_{3,1}$ and $c_{4,2}$, and the agent can form the category $\mathbf{c}_1 = (c_{3,1}, c_{4,2})$. In principle, all possible combinations of categorical features can be used as a category, so categories $\mathbf{c}_2 = (c_{3,1})$ and $\mathbf{c}_{4,2} = (c_{4,2})$ are also valid categories. In order to prevent a combinatorial explosion of the search space for categories, we are designing heuristics to prevent searching all possible combinations, such as looking for distinctive categories of the lowest dimension, or by taking combinations that form groups of objects.

Similar to the categorisation of A_1 and $A_2, T_1,\ T_2$ and T_3 are categorised using categorical features $c_{3,2}$ and $c_{4,1}$; the food source F_1 has categorical features $c_{3,2}$ and $c_{4,3}$; and F_2 and F_3 are categorised using $c_{3,2}$ and $c_{4,4}$. As mentioned, the aim of the discrimination game is to find categories that distinguish an object (or group of objects) from the rest of the context. In this example, only F_1 has distinctive categories. When trying to categorise F_3, for example, the discrimination game fails, and the ontology has to be expanded (recall that initially, each agent's ontology is empty). This is done by taking the features of F_3 as exemplars for new categorical features, yielding $c_{3,3} = 0.8$ and $c_{4,5} = 0.8$. Of course when additionally considering all different feature dimensions, the agent may have had categorical features that would distinguish each object from another.

In the language that will be constructed, agents map categories to words. The agent can use a combination of categories to distinguish the object it wants to communicate, thus forming a multiple word utterance. We intend to use this possibility as a means to develop a simple grammar.

3.3 Theory of mind and language games

Probably the biggest problem that this project has to deal with is what we loosely call the *Theory of Mind* (ToM). When a speaker communicates something to another agent, the hearer has to infer what the speaker refers to. When the language is well developed, this may not need to be problematic, but when the communication system of an agent is undeveloped or when the agents speak a different language, this is arguably one of the biggest problems in science. Nevertheless, humans seem to deal with this problem of referential indeterminacy relatively easy. It is commonly accepted that humans have developed (either phylogenetically or ontogenetically) ToM, which relates to the ability to form theories about other individual's intentions (Bloom, 2000).

Although eventually we intend to evolve some aspects of ToM in New Ties, we shall begin by implementing them directly. The ToM will become an integral part of the language games we will develop. The language game, based on (Steels, 1996a), implements the interaction between two (or more) individuals as illustrated in Table 2. In essence, the agents start by perceiving the context of the game and categorise the objects they see using the discrimination game (DG)

t	speaker	hearer
n	-perceive context -categorisation/DG -focus attention -produce utterance -update lexicon1 -send message	
$n+1$		-receive message -perceive context -categorisation/DG -focus attention -interpret utterance -update lexicon1 -respond
$n+2$	-evaluate effect -respond	
$n+3$		-evaluate effect -respond
$n+4$	-update lexicon2	-update lexicon2

Table 2: An example scheme for playing a language game between a speaker and hearer. The game may take up to 5 time steps t. See the text for details.

as explained above.

That the contexts of agents typically differ is illustrated in Fig. 2. The context of agent A_1 contains 4 of the 5 food items of type `Food1`, agent A_2, the contents of A_2's bag (2 more food items of `Food1`) and the contents of its own bag (1 `Token`, 1 `Food1` and 1 `Food2`).[5] The context of agent A_2 contains 2 `Tokens`, 2 `Food1` and agent A_1 from the visual field, the contents of A_1's bag and the contents of its own bag. Due to the physical nature of the environment, we can (and will) not make sure that the contexts of different agents are the same. However, we can introduce aspects of ToM that give the agents cues what the other can see and what the other's intentions are. This will be part of the *focus attention* mechanism. In this mechanism we will assume an attention strength $strA(o_i)$ for object o_i, which is calculated using a formula such as:

$$\begin{aligned} strA(o_i) = \quad & w_1 P_{A'}(o_i) + w_2 V_{A'}(o_i) + w_3 N(o_i) + \\ & w_4 I_S(o_i) + w_5 I_{A'} + \ldots \end{aligned} \tag{2}$$

where w_j are weights and the other arguments are functions that estimate certain aspects of both agents' intentions and knowledge of the current situation. $P_{A'}(o_i)$ is the normalised frequency with which the other agent A' has communicated about object o_i in the presence of the evaluating agent – the self S. $V(A')(o_i)$ is a function that estimates the likelihood that o_i is in the visual field of A'. $N(o_i)$ is a novelty (or *salience*) detector, indicating how novel o_i is in the context of S. We assume $N(o_i) = 1$ if o_i first enters the context or when it is shown by another agent; after which it decays following $N(o_i) = e^{-\beta t}$, with β a positive constant and t the time period that o_i is in the context. If an agent explicitly shows an object, the object will also get a high novelty value. $I_S(o_i)$ is a function that calculates the drive of S to communicate about this object, based on its internal states. For instance, if S is hungry, it has a large drive to communicate about food items. Finally, $I_{A'}(o_i)$ is a function that estimates the drive of A' to communicate about object o_i.

The speaker of the language game will use $strA(o_i)$ to select the topic it wants to talk about. If any of the strengths is below a certain threshold, or – at least – lower than any other action to take, then the language game will not proceed. If, however, for some object o_i the value $strA(o_i)$ exceeds any other object attention strength or action value, the language game will proceed with the utterance production of the speaker.

The agents construct and maintain a lexicon in their

[5] Note that obscured objects are not perceived by the agents.

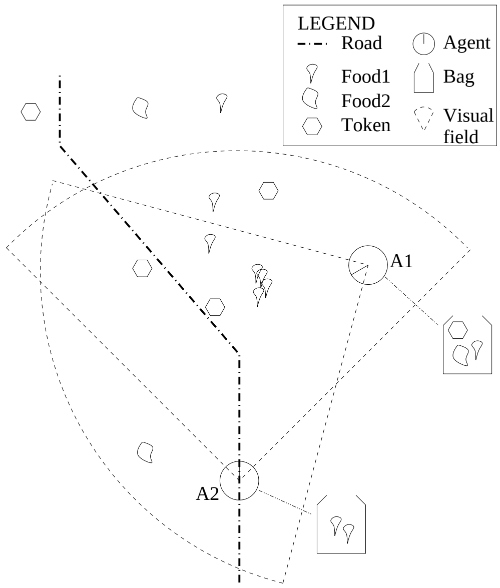


Figure 2: An example situation of a language game. An agent's context is defined by those objects that in its visual field, the contents of the bag of any agent that is in their visual field, including their own. Objects that are obscured by another object are not visible, nor are objects outside the visual field.

	m_1	$\dots$	m_N
w_1	P_{11}	$\dots$	P_{1N}
$\vdots$	$\vdots$	$\vdots$	$\vdots$
w_M	P_{M1}	$\dots$	P_{MN}

	m_1	$\dots$	m_N
w_1	σ_{11}	$\dots$	σ_{1N}
$\vdots$	$\vdots$	$\vdots$	$\vdots$
w_M	σ_{M1}	$\dots$	σ_{MN}

Table 3: Two association matrices constructed and maintained as part of an agent's lexicon. The upper matrix (lexicon1) associates m_j with w_i using conditional *a posteriori* probabilities P_{ij}. The lower matrix (lexicon2) associates categories (or meanings) m_j with words w_i using an association score σ_{ji}.

memories, which is represented by two association matrices as illustrated in Table 3. One of the matrices(referred to as lexicon1 in Table 2) keeps an *a posteriori probability* P_{ij}, which is based on the usage frequencies of associations. The other matrix (lexicon2) keeps an *association score* σ_{ij}, which indicates the effectiveness of an association based on past experiences. The reason for this twofold maintenance is that studies have revealed that when strong attentional cues (such as joint attention or corrective feedback, discussed shortly) guide learning, the lexical acquisition is much faster than when such cues are absent (Vogt and Coumans, 2003). The games using strong attentional cues work fast, because the update mechanism reinforces the score σ_{ij} more strongly than the update of usage based probabilities P_{ij}, which – in turn – work more effectively when the strong attentional cues are absent.

The probabilities are conditional probabilities, i.e.

$$P_{ij} = P(m_j|w_i) = \frac{u_{ij}}{\sum_j u_{ij}} \quad (3)$$

where u_{ij} is the co-occurrence frequency of meaning m_j and word w_i. This usage frequency is updated each time a word co-occurs with a meaning that is either the topic (in case of the speaker) or that is in the context (in case of the hearer). The update is referred to in Table 2 as 'update lexicon1'. If this principle would be the only mechanism, the learning is achieved across different situations based on the covariance in word-meaning pairs (Vogt and Smith, 2005).

The association score σ_{ij} is updated following:

$$\sigma_{ij} = \eta\sigma + (1-\eta)X \quad (4)$$

where η is a learning parameter (typically $\eta = 0.9$), $X = 1$ if the association is used successfully in the language game, and $X = 0$ if the association is used wrongly in the language game, or – in case of a successful language game – if the association is competing with the used association (i.e. same word, different meaning; or same meaning, different word). The latter implements lateral inhibition. The update of association scores is referred to in Table 2 as 'update lexicon2'.

Given these two matrices, the speaker, when trying to produce an utterance, calculates an association strength $strL(\alpha_{ij})$ for each association α_{ij} of a word w_i with meaning m_j, where the meaning is the category that the speaker wants to communicate. This is done using Eq. (5).

$$strL(\alpha_{ij}) = \sigma_{ij} + (1-\sigma_{ij})P_{ij} \quad (5)$$

This formula neatly couples the two variables. When σ_{ij} is high, the influence of P_{ij} is low, and when σ_{ij} is low, P_{ij} will have more influence. This implements a bias toward basing a choice on known effectiveness vs. estimated probabilities. The speaker will select the association that has the highest strength and utter its word. If no association can be found, e.g., because the lexicon is still empty, the speaker may invent a new word and adds the association to its lexicon.

When the hearer receives an utterance, after it perceives a context and categorises its objects using the DG, it will estimate the attention strength of objects in the context using Eq. (2). Then it calculates for each association of which the word matches the utterance and the meaning matches one of the categorised objects using Eq. (5). The hearer then interprets the utterance using the following equation:

$$\rho_{ij} = \omega_L \cdot strL(\alpha_{ij}) + \omega_A \cdot strA(o_i) \quad (6)$$

where ω_L and ω_A are weights. This equation is based on the model presented in Gong et al. (2004).

Based on its choice, the hearer will respond with some action, which still needs to be specified. An example response could be that the hearer will give the speaker food. The speaker will then (time step $n+2$ in Table 2) evaluate the effect of the language game. If this is what the speaker intended, it can signal the effect to the hearer as response.[6] In turn, the hearer will evaluate this signal and – if necessary – respond as well. If this finishes the language game, the agents

[6] Here we assume that agents can notice if an action has a positive or negative effect.

can update lexicon2 using Eq. (4) with $X = 1$ for the used association and $X = 0$ for competing ones if the game is successful. If the game had noticeably failed, then lexicon2 is updated with $X = 0$ for the used association.

There are many reasons why the language game may fail. For instance, the hearer could not interpret the utterance, or its response does not match the speaker's intention. In the first case, the hearer can signal a failure as response. In the latter case, the speaker can signal a failure. In both cases, the game will need to be repaired in order to allow significant learning.

For now, we will assume that the initiative to repair the game lies with the speaker. For example, the speaker can ignore the failure when the hearer was not the direct addressee, or when the social bond is low and the speaker wishes to proceed with another action. The speaker can also decide to do one of the following things in order to provide the hearer with additional cues about which object is the reference of the game:

- show an object from the bag;
- point to an object in the context by looking in its direction;
- show an action;
- go to the object;
- ...

Using these cues, the hearer tries to reinterpret the utterance with a strong additional bias to the shown object, and the game is re-evaluated. We will implement a mechanism to prevent this continuing forever; for instance by allowing only one or two reinterpretations.

If the hearer did not have an interpretable association of the utterance in the first place, it will adopt the utterance and add a new word-meaning association to its lexicon. The initial value of $\sigma_{new,j}$ will be based on existing associations with word w_j – if any – and the attention strength of object o_{new} according to

$$\sigma_{new,j} = k \cdot (1 - \max_i(\sigma_{i,j})) \cdot strA(o_{new}) \quad (7)$$

where we assume that $\sigma_{new,j}$ relates to meaning m_{new} that is a distinctive category of object o_{new}. (Note that there may be more than one such association.) The association(s) will be added to lexicon1 with an initial usage of $u_{new,j} = 1$.

To summarise, we intend to extend the familiar language game model in order to include aspects of ToM. The language game is largely based on the *guessing game*, which uses corrective feedback to guide meaning inference, and a game that uses cross-situational statistical learning (Vogt and Smith, 2005). The cues as formalised in Eqs. (2) – (7), together with the repair mechanisms, are the core mechanisms of the ToM. Initially we intend to hard-wire the ToM into the New Ties project, but at some stage we wish to evolve this – for instance by evolving the various weights of Eqs. (2) and (6).

4 Conclusions

In this paper we identify three major problems regarding modelling language evolution in large populations of autonomous agents, such as proposed in the New Ties project. The problems and proposed solutions can be summarised as follows:

1. How can we control communication in large populations? We intend to treat this as a minor problem by limiting communication based on the spatial location of agents and the social networks they develop. In addition, to provide well structured learning environments for the young agents, we will treat close kinship relations as an extra drive to communicate.

2. How can we categorise a large number of objects such that they are learnable in language? To solve this problem, we propose a model based on Steels' discrimination games (Steels, 1996b) where perceptual features are categorised following the implementation of Vogt (2005). To deal with overlapping classes of objects we intend to develop heuristics that group categorical features that are similar across different objects.

3. How do we deal with issues relating to the Theory of Mind? This problem is identified as the hardest problem. In order to deal with it, we propose to design mechanisms that allow an individual to use perceptual and interpretational information to provide cues concerning the objects that the other agent is likely to communicate about. These mechanisms will be integrated in the familiar language game models used earlier in Vogt and Coumans (2003) similar to the way proposed by Gong et al. (2004). In addition, social strategies are proposed in order to repair failures in language games.

We are currently investigating stability conditions of social networks in relation to the scaling of populations. In addition, we are implementing a simple version of the ToM to prove the principle.

We believe that, although the problems we identified are hard, we can scale up models of language evolution successfully much in the way we discussed. If we succeed, we expect that this experiment will provide an exciting benchmark for many large scale experiments regarding the evolution of language.

Acknowledgements

The NewTies project is partially supported by an EC FET grant under contract 003752.

References

P. Bloom. *How Children Learn the Meanings of Words*. The MIT Press, Cambridge, MA. and London, UK., 2000.

A. Cangelosi and D. Parisi, editors. *Simulating the Evolution of Language*. Springer, London, 2002.

T. M. Cover and P. E. Hart. Nearest neighbour pattern classification. *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, 13:21–27, 1967.

B. De Boer. Self-organization in vowel systems. *Journal of Phonetics*, 28:441–465, 2000.

J. M. Epstein and R. Axtell. *Growing artificial societies: social science from the bottom up*. MIT Press, Cambridge, MA., 1996.

N. Gilbert, S. Schuster, M. den Besten, and L. Yang. Environment design for emerging artificial societies. In *this issue*, 2005.

T. Gong, J. Ke, J. W. Minett, and W. S-Y. Wang. A computational framework to simulate the coevolution of language and social structure. In Jordan Pollack, Mark Bedau, Phil Husbands, Takashi Ikegami, and Richard A. Watson, editors, *Artificial Life IX Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems*, pages 214–219. The MIT Press, 2004.

S. Kirby. Spontaneous evolution of linguistic structure: an iterated learning model of the emergence of regularity and irregularity. *IEEE Transactions on Evolutionary Computation*, 5(2):102–110, 2001.

S. Kirby. Natural language from artificial life. *Artificial Life*, 8(3), 2002.

S. Kirby and J. R. Hurford. The emergence of linguistic structure: An overview of the iterated learning model. In A. Cangelosi and D. Parisi, editors, *Simulating the Evolution of Language*, pages 121–148, London, 2002. Springer.

A. D. M. Smith. Intelligent meaning creation in a clumpy world helps communication. *Artificial Life*, 9(2):559–574, 2003.

K. Smith and J. R. Hurford. Language evolution in populations: extending the iterated learning model. In W. Banzhaf, T. Christaller, J. Ziegler, P. Dittrich, and J. T. Kim, editors, *Advances in Artificial Life: Proceedings of the 7^{th} European Conference on Artificial Life*, Lecture Notes in Computer Science: Lecture Notes in Artifical Intelligence, pages 507–516. Springer-Verlag, Heidelberg, 2003.

L. Steels. Emergent adaptive lexicons. In P. Maes, editor, *From Animals to Animats 4: Proceedings of the Fourth International Conference On Simulating Adaptive Behavior*, Cambridge Ma., 1996a. The MIT Press.

L. Steels. Perceptually grounded meaning creation. In M. Tokoro, editor, *Proceedings of the International Conference on Multi-Agent Systems*, Menlo Park Ca., 1996b. AAAI Press.

L. Steels. The synthetic modeling of language origins. *Evolution of Communication*, 1(1):1–34, 1998.

L. Steels. Constructivist development of grounded construction grammars. In W. Daelemans, editor, *Proceedings Annual Meeting of Association for Computational Linguistics*, 2004.

P. Vogt. Bootstrapping grounded symbols by minimal autonomous robots. *Evolution of Communication*, 4(1):89–118, 2000.

P. Vogt. The emergence of compositional structures in perceptually grounded language games. *Artificial Intelligence*, 2005. In press.

P. Vogt and H. Coumans. Investigating social interaction strategies for bootstrapping lexicon development. *Journal for Artificial Societies and Social Simulation*, 6(1), 2003. http://jasss.soc.surrey.ac.uk.

P. Vogt and A. D. M. Smith. Learning colour words is slow: a cross-situational learning account. *Behavioral and Brain Sciences*, 2005. In press.

Environmental risk

Peter Andras[*]
[*]School of Computing Science
University of Newcastle
peter.andras@ncl.ac.uk

John Lazarus[†]
[†]School of Biology
University of Newcastle
j.lazarus@ncl.ac.uk

Gilbert Roberts[†]
[†]School of Biology
University of Newcastle
gilbert.roberts@ncl.ac.uk

Steven J Lynden[*]
[*]School of Computing Science
University of Newcastle
s.j.lynden@ncl.ac.uk

Abstract

Environmental risk is an important factor that influences social evolution in natural and artificial environments. Here we analyse environmental risk and we define objective, subjective and effective environmental risk, which are important aspects of general environmental risk. We show theoretically and by analysing simulation data that subjective risk is larger than objective risk and that effective risk is smaller than subjective risk, when cooperation is present. We believe that the proposed conceptualisation of environmental risk can help in understanding its effects on social evolution and also in designing of artificial social environments.

1 Introduction

Risk is present in natural environments of living organisms in the form of the variance of outcomes of events or scenarios involving the organism (e.g., the variance of distribution of resources)[1]. Environmental risk plays an important role in determining social relationships between individuals and in particular in the determination of the level of cooperation between individuals (e.g., Andras et al., 2003; Andras and Lazarus, 2004; Callaway et al., 2002; Hewstone et al. 2002; Seghers, 1974). For example, bacteria living in the presence of antibiotics are more likely to form interaction clusters in form of biofilms (Drenkard and Ausubel, 2002), and mole rats living in arid conditions form larger communities than mole rats living in mesic environments (Spinks et al., 2000). Simulation studies also show that higher environmental risk induces more cooperation in societies of selfish agents (Andras et al., 2003).

To understand the role of environmental risk in influencing social interactions and in particular cooperation (e.g., Fehr and Fischbacher, 2003; Fehr and Rockenbach, 2003) we need to understand how environmental risk is structured in the context of social interactions between selfish agents living in a risky environment. We use an agent society simulation approach (Andras et al., 2003) to study the effects of environmental risk. In our simulated world the agents play prisoners' dilemma (Axelrod and Hamilton, 1981) type games in order to generate new resources, using their existing resources. We chose the prisoner's dilemma scenario for interactions between agents, because this is a commonly used scenario in theoretical works about cooperation (e.g., Axelrod and Hamilton, 1981; Milinski et al., 2002; Nowak and Sigmund, 1998; Riolo et al., 2001; Roberts and Sherratt, 1998) and allows the comparison of our results with other works on similar topics. Environmental risk is represented as the variance of the generated new resources.

In our view environmental risk is structured as objective risk, subjective risk, and effective risk. The objective risk is the variance of the resource distribution in the environment, the subjective risk is the perceived variance of the resources, and the effective risk is the variance of the resources gained

[1] We note that risk in general can be conceptualized either in terms of variance of outcomes (e.g., Degeorge et al., 2004; Grundke, 2004) or as likelihood of undesired outcomes (e.g., Kobbeltvedt and Brun, 2004; Lundborg and Lindgren, 2004). Here we adopt the first approach.

by the agents living in the environment. We show in this paper, how these aspects of environmental risk are calculated and we analyse the relationship between them. We use an agent society simulation to validate experimentally our theoretical results about the relationships between the three aspects of environmental risk.

The rest of the paper is structured as follows. In Section 2 we discuss the relationship between objective and subjective risk. Section 3 presents the relationship between the subjective and effective risk. In Section 4 the experimental analysis is discussed. The conclusions are presented in Section 5.

2 Objective and subjective risk

Without restricting the level of generality, we consider environmental risk in terms of variance of resource distributions. Let us denote by $D(R)$ the distribution of resources in an individual's environment, and let σ^2 be the variance of this distribution. The objective environmental risk is σ^2 the variance of the distribution of resources in the environment.

Individuals cannot perceive the full distribution of the resources. The reason for this is that they might be selective (e.g., ignoring too small amounts of resources), or they may be limited in their ability to exploit resources (e.g., limited use of too large quantities of expiring resources). We capture this situation by choosing a sufficiently general scenario, that of the selective ignorance of low amounts of resources caused by the harshness of the environment (e.g., cold, arid or dangerous environment). The environment being harsher makes it more costly for individuals to explore resources raising the minimum amount of acceptable resource (i.e., the amount at which the gained benefit equals the exploration costs). For example it has been observed in case of rodent foraging that individuals shift to more profitable foods under conditions of high predation risk (Hay and Fuller, 1981; Bowers, 1988).

Resources are considered acceptable by individuals only if $R > R_m$, where R_m is the minimum amount of acceptable resource, which is the amount of resources at which the expected cost of exploring the resource is equal to the benefits gained from exploring the resource. This means that the part of the resource distribution corresponding to resource values $R \le R_m$ is ignored by individuals living in this environment, subjectively equating this part of the distribution with a Dirac δ distribution centred in 0 multiplied by the probability $P(R \le R_m) = \int_0^{R_m} D(R)dR$. We assume that the harshness of the environment is not excessive, in the sense that more than half of the full resource distribution is perceived by the individuals, i.e., $\int_{R_m}^{+\infty} D(R)dR > \frac{1}{2}$.

The subjective environmental risk is the variance of the perceived resource distribution. The perceived resource distribution is the following distribution:

$$D_s(R) = \left(\int_0^{R_m} D(R)dR \right) \cdot \delta_0(R) + \begin{cases} D(R), & R > R_m \\ 0, & R \le R_m \end{cases} \quad (1)$$

Calculating the variance of the perceived environmental resource distribution we get:

$$\sigma_s^2 = \int_0^{+\infty} R^2 D_s(R)dR - \overline{R}_s^2 \quad (2)$$

where $\overline{R}_s$ is the mean of the subjective resource distribution.

To compare the subjective and objective environmental risk we calculate the difference between the variances of subjective and objective resource distributions. After calculations we get that:

$$\sigma_s^2 - \sigma^2 = \int_0^{R_m} R \cdot (\overline{R}_s + \overline{R} - R) \cdot D(R)dR \quad (3)$$

where $\overline{R}$ is the mean of the objective resource distribution.

If $\int_{R_m}^{+\infty} D(R)dR > \frac{1}{2}$ it can be proven that

$$2 \cdot \overline{R}_s > R_m \tag{4}$$

We also have that

$$\overline{R} = \int_0^{+\infty} R \cdot D(R) dR > \int_{R_m}^{+\infty} R \cdot D(R) dR = \overline{R}_s \tag{5}$$

Equations (4) and (5) imply that

$$\overline{R}_s + \overline{R} - R > 0 \tag{6}$$

if $R \in [0, R_m]$. So, combining equations (3) and (6) we deduce that

$$\sigma_s^2 - \sigma^2 = \int_0^{+\infty} R \cdot (\overline{R}_s + \overline{R} - R) \cdot D(R) dR > 0 \tag{7}$$

showing that the subjective environmental risk is higher than the objective environmental risk.

Our analysis shows that we need to differentiate between objective and subjective environmental risks. The objective risk is the variance of the whole resource distribution within the environment, while the subjective risk is the variance of the modified resource distribution perceived by the individuals living in the environment. We have shown here that the subjective risk is larger than objective risk.

3 Subjective and effective risk

Experimental evidence shows that biological organisms living in risky environments cooperate more than those living in less risky environments (Callaway et al., 2002; De Bono et al., 2002; Dunbar, 1988; Drenkard and Ausubel, 2002; Farr, 1975; Goody, 1991; Hewstone et al., 2002; Hogg, 1992; Seghers, 1974; Spinks et al., 2000). Earlier analysis of simulation of agent societies confirmed the biological evidence, showing that surviving agent populations in higher risk environments are characterized by higher level of cooperation than surviving populations in lower risk environments (Andras et al., 2003).

Cooperation between individuals essentially means sharing of the risks associated with their individual living. In the context of the simplified scenario described in the previous section, individuals search for resources and by cooperation they share their finds. By sharing they reduce the variance of the distribution of the resources available for them.

Putting it more formally, let us suppose that c is the proportion of those individuals which cooperate, and $1-c$ is the proportion of individuals which do not cooperate (i.e., compete or cheat). This means that the proportion of those who benefit from cooperation is c^2, of those who cheat is $c \cdot (1-c)$, of those who are cheated is $c \cdot (1-c)$, and of those who do not enter in interaction with others is $(1-c)^2$.

Those individuals who cheat gain extra resources without sharing, those who are cheated lose resources without gaining from sharing. The simplest way to model these effects on their perceived resource distribution is to consider that the resource distribution for cheaters is shifted to the right, and for those who are cheated is shifted to the left. This means that we get the following formulas for the mean and variance of the resource distribution:

$$\begin{aligned} \overline{R}_e = \int_{R_m}^{+\infty} [& c^2 \cdot RD_{coop}(R) + \\ & c \cdot (1-c) \cdot RD(R+r) + \\ & c \cdot (1-c) \cdot RD(R-r) + \\ & (1-c)^2 \cdot RD(R)] dR = \\ & c^2 \cdot \overline{R}_{coop} + c(1-c) \cdot \overline{R}_{cheat} + \\ & c(1-c) \cdot \overline{R}_{suck} + (1-c)^2 \cdot \overline{R}_{no} \end{aligned} \tag{8}$$

where the subscript 'coop' refers to those agents who benefit from cooperation, 'cheat' refers to agents that cheat, 'suck' refers to agents that are cheated, and 'no' refers to agents which do not participate in interactions;

$$\sigma_e^2 = \int_{R_m}^{+\infty} [c^2 \cdot R^2 D_{coop}(R) + c \cdot (1-c) \cdot R^2 D(R+r) + c \cdot (1-c) \cdot R^2 D(R-r) + (1-c)^2 \cdot R^2 D(R)] dR - \overline{R}_e^2 = c^2 \cdot \sigma_{coop}^2 + c \cdot (1-c) \cdot \sigma_{cheat}^2 + c \cdot (1-c) \cdot \sigma_{suck}^2 + (1-c)^2 \cdot \sigma_{no}^2 - \lambda \tag{9}$$

where

$$\lambda = \overline{R}_e^2 - \Big(c^2 \cdot \overline{R}_{coop}^2 + c(1-c) \cdot \overline{R}_{cheat}^2 + c(1-c) \cdot \overline{R}_{suck}^2 + (1-c)^2 \cdot \overline{R}_{no}^2 \Big) > 0 \tag{10}$$

and σ_{coop}^2 is the variance of the resource distribution for those who benefit from cooperation, σ_{cheat}^2 is the variance of the resource distribution for those who cheat, σ_{suck}^2 is the variance of the resource distribution for those who are cheated, and σ_{no}^2 is the variance of the resource distribution for those who do not participate in interaction with others.

In accordance with the above suppositions the effect on the variance of the cheater and cheated resource distributions is equivalent to shifting the cut-off point R_m to the right for those who are cheated, and to the left for the cheaters. Consequently, the perceived resource variance for cheaters will be lower than the subjective variance defined in the previous section, and the perceived variance for those who are cheated will be larger than the subjective variance. To make things simple let us suppose that

$$\sigma_{cheat}^2 = \sigma_s^2 - \sigma_0 \tag{11}$$

and

$$\sigma_{suck}^2 = \sigma_s^2 + \sigma_0 \tag{12}$$

In case of individuals who do not enter in interactions their perceived resource variance is exactly the subjective variance. For individuals who participate in cooperation the perceived resource variance is

$$\sigma_{coop}^2 = \frac{1}{2} \sigma_s^2 \tag{13}$$

Considering equations (10) – (13) we calculate the effective resource variance for the population of individuals using equation (14).

Equation (14) shows that the effective environmental risk measured for the whole population is smaller than the subjective environmental risk if there is some level of cooperation within the society of individuals.

$$\sigma_e^2 = c^2 \cdot \frac{1}{2} \sigma_s^2 + c \cdot (1-c) \cdot (\sigma_s^2 - \sigma_0) + c \cdot (1-c) \cdot (\sigma_s^2 + \sigma_0) + (1-c)^2 \cdot \sigma_s^2 - \lambda = (1 - \frac{1}{2} c^2) \cdot \sigma_s^2 - \lambda \tag{14}$$

We have shown in this section that a third important aspect of environmental risk is the effective risk, which is measured as the effective variance of the resource distribution measured for the whole population of individuals constituting a social group or society. Our analysis shows that the effective environmental risk is lower than the subjective environmental risk if there is some level of cooperation within the population.

4 Experimental analysis

We built an agent-based simulation to study environmental risk and its effects on the level of cooperation. Our agents own resources (R) that they spend on living costs and use to generate new resources for the future. The agents live in a two-dimensional world, each having a position (x, y) and change location by random movements, i.e., $(x_{new}, y_{new}) = (x, y) + (\xi_x, \xi_y)$, where ξ_x, ξ_y are small random numbers. The agents have an inclination toward cooperation or competition, expressed as p the probability of cooperation. If $p < 0.5$ they are more likely to compete than to cooperate. They select their behaviour for each interaction in a probabilistic manner biased by their inclination. This is done by choosing a random number q from a uniform distribution over $[0,1]$; if $q < p$ they cooperate, otherwise they compete.

Objective environmental risk was implemented as the variance of the payoffs arising from the cooperative / competitive interactions (the new amounts of resources after the interaction are given by the payoffs). In each time unit, each agent randomly chooses an interaction partner from its neighbourhood and the partners decide whether to cooperate or compete. The payoffs for the agents are determined by sampling a random variable X_R that has normal distribution $N(\overline{X}, \sigma_X)$. The mean value $\overline{X}$ is determined by the amount of invested resources according to the payoff table shown in Table 1. The values in the pay-off table are such that the table satisfies the conditions of Prisoner's Dilemma games (Axelrod and Hamilton, 1981). The variance σ_X is the objective environmental risk. Environmental harshness is modelled by a cut-off value R_m. Resource amounts below the cut-off value are equated to zero. Varying the value of R_m allows us to investigate how objective, subjective and effective environmental risk relate to each other in the context of agent society with agents that may cooperate.

The agents produce offspring at the end of their lifetime who inherit their parent's behavioural inclination with some small random change (i.e., $p_{offspring} = p_{parent} + \xi$, $\xi \in [-\varepsilon, \varepsilon]$, and ε is a small number, e.g., $\varepsilon = 0.025$). The number of offspring depends on the amount of resources of the agent according to the equation

$$n = \alpha \cdot \frac{R - (\overline{R} - \beta \cdot \sigma_R)}{\sigma_R} + n_0 \tag{15}$$

where $\overline{R}$ is the mean and σ_R is the variance of the resources in the population of the agents, and α, β, n_0 are parameters. The offspring share

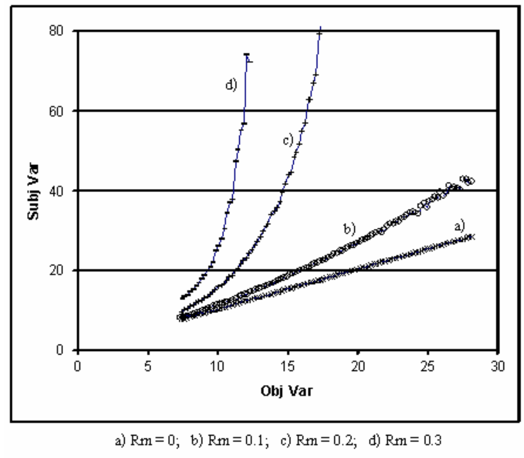


Figure 1: The relationship between the objective and subjective variances for different resource amount cut-off points.

equally the resources of their parent. The offspring start their life from their parent's last location with minor random changes, implying that the offspring of each agent will be closely packed at the beginning. The cluster of offspring diffuses with time, as the offspring make their random movements. The generation of offspring guarantees the evolutionary change in the population of our agents. Successful agents produce many offspring, while unsuccessful agents produce few or no offspring. The success of the agents (i.e., the amount of resources that they accumulate) depends on their willingness to cooperate and on the riskiness of their environment. Many offspring of successful agents and few or no offspring of unsuccessful agents guarantee that the population of agents evolves a mix of behaviours that fit the environment and produces optimal conditions for individual agents of the population.

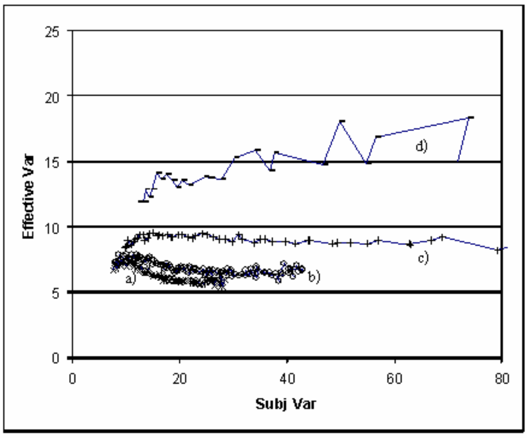
We simulated the evolution of 20 agent populations at four different cut-off levels representing environments with different harshness but with the same objective risk (i.e. we ran 20 simulations for each level of environmental harshness). Each population started with around 1500 individuals and the simulation ran for 1000 time units, the agents' mean lifetime being 60 time units. The inclination toward cooperation of the agents was set randomly according to a uniform distribution over [0,1]. We calculated for each simulation, for each time round the objective variance of the generated resources (i.e., objective risk), the subjective variance of the generated resources considering the amounts of resources that could be generated without cooperation and applying the cut-off at the set level of R_m (i.e., subjective risk), and the effective variance of resources considering the effects of the cooperation games played by the agents and also the cut-off at the level of R_m (i.e., effective risk).

Based on the earlier theoretical analysis we expect that the subjective variance should be above the objective variance, and that the effective variance should be below the subjective variance. Figure 1 presents the relationship between the measured objective and subjective variances of the resources for four levels of cut-off resource amounts (R_m =0, 0.1, 0.2, 0.3). Figure 2 shows the relationship between the measured subjective and effective variances of the resources.

The results confirm our expectations and show that indeed it is important to consider the three identified aspects of environmental risk in the context of analysis of such risk on the evolution of social structures, and in particular on the evolution of cooperation. The results also point to the need for a more detailed investigation of the relationship between the level of cooperation and the difference between the subjective and effective risk.

Table 1. The pay-off matrix for the cooperation / competition game. Entries indicate the payoffs to the row player followed by the column player. R1 and R2 are the amount of resources of the row and column player respectively, and $\Delta = \left[f(R_1 + R_2) - f(R_1) - f(R_2) \right]_+$ (i.e., it takes only the positive values of the expression in brackets and it is zero if the value of the expression is negative). The function f is a diminishing return function, and R1 and R2 are typically in the range where $2f(x) \le f(2x)$, and $0 < \alpha < 1$.

$\overline{X}$	Cooperate	Compete
Cooperate	$f(R_1)+\frac{\Delta}{2}$, $f(R_2)+\frac{\Delta}{2}$	$\alpha \cdot f(R_1)$, $f(R_2)+\Delta$
Compete	$f(R_1)+\Delta$, $\alpha \cdot f(R_2)$	$f(R_1)$, $f(R_2)$



a) Rm = 0; b) Rm = 0.1; c) Rm = 0.2; d) Rm = 0.3

Figure 2: The relationship between the subjective and effective variances for different resource amount cut-off points.

5 Conclusions

We analysed three aspects of environmental risk in the context of cooperation in communities of selfish individuals. The objective risk is the variance of the resource distribution within the environment; the subjective risk is the perceived variance of the resource distribution; while the effective risk is the variance of the distribution of effective resource amounts available for individuals within a population. We have shown on theoretical grounds that the subjective risk is higher than the objective risk, and that the effective risk is lower than the subjective risk if there is some level of cooperation within the environment.

We validated our theoretical results by analysing a series of simulations of simple agent worlds, in which populations of agents evolve. The simulation data confirmed that indeed the subjective risk is larger than the objective risk, and the effective risk is smaller than the subjective risk.

Our analysis highlights the importance of environmental risk for the evolution of social interactions, and clarifies the key aspects of environmental risk in this context. We believe that experimental biological data analysed in sufficient detail will confirm our predictions about the three aspects of environmental risk based on theoretical grounds and simulation data analysis.

Our work also has implications in the context of designing artificial agent worlds. In this respect our results point to the importance of considering the effects of environmental risk for the development of such agent worlds. In particular, our work highlights the importance of appropriate tuning of risk perception and of the effects of cooperation on the effective risk, which might contribute significantly to the achievement of the desired level of mixture of cooperative and competitive behaviour within the artificial agent world.

References

P. Andras and J. Lazarus. Cooperation, Risk and the Evolution of Teamwork. In: *Teamwork: Multi-Professional Perspectives* (ed. N. Gold), pp.56-77. New York: Palgrave Macmillan, , 2004.

P. Andras, G. Roberts, and J. Lazarus. Environmental risk, cooperation and communication complexity. In *Adaptive Agents and Multi-Agent Systems*, (eds E. Alonso, D. Kudenko, and D. Kazakov) pp. 49-65. Berlin: Springer-Verlag, 2003.

R. Axelrod and W.D. Hamilton. The evolution of cooperation. *Science*, 211:1390-1396, 1981.

M.A. Bowers. Seed removal experiments on desert rodents: the microhabitat by moonlight effect. *Journal of Mammalogy,* 69:201-204, 1988

R.M. Callaway, *et al.* Positive interactions among alpine plants increase with stress. *Nature*, 417:844-847, 2002

M. De Bono, D.M. Tobin, M.W. Davis, L. Avery L., and C.I. Bargmann. Social feeding in *Caenorhabditis elegans* is induced by neurons that detect aversive stimuli. *Nature*, 419:899-903, 2002.

F. Degeorge, B. Moselle, and R. Zeckhauser. The ecology of risk taking. *Journal of Risk and Uncertainty*, 28:195-215, 2004.

E. Drenkard and F.M. Ausubel. *Pseudomonas* biofilm formation and antibiotic resistance are linked to phenotypic variation. *Nature*, 416:740-743, 2002.

J.A. Farr. The role of predation in the evolution of social behavior of natural populations of the guppy. *Poecilia reticulata (Pisces: Poeciliidae). Evolution*, 29:151–158, 1975.

E. Fehr and U. Fischbacher. The nature of human altruism. *Nature*, 425:785-791, 2003.

E. Fehr and B. Rockenbach. Detrimental effects of sanctions on human altruism. *Nature*, 422:137-140, 2003.

E. Goody. The learning of prosocial behaviour in small-scale egalitarian societies: an anthropological view. In *Cooperation and Prosocial Behaviour* (eds R.A. Hinde and J. Groebel) pp. 106-128. Cambridge: Cambridge University Press, 1991.

P. Grundke. Integrating interest rate risk in credit portfolio models. *Journal of Risk Finance*, 5:6-15, 2004.

M.E. Hay and P.J. Fuller. Seed escape from heteromyid rodents: the importance of microhabitat and seed preference. *Ecology*, 62:1395-1399, 1981.

M. Hewstone, M. Rubin, and H. Willis. Intergroup bias. *Annual Review of Psychology*, 53:575–604, 2002.

M.A. Hogg. *The social psychology of group cohesiveness. From attraction to social identity.* New York: Harvester Wheatsheaf, 1992.

T. Kobbeltvedt and W. Brun. Measuring and modelling risk in a naturalistic setting. *Journal of Risk Research*, 7:789-810, 2004.

P. Lundborg and B. Lindgren. Do they know what they are doing? Risk perceptions and smoking behaviour among Swedish teenagers. *Journal of Risk and Uncertainty*, 28:261-286, 2004.

M. Milinski, D. Semmann, and H.J. Krambeck. Reputation helps solve the ‘tragedy of the commons’. *Nature*, 415:424-426, 2002.

M.A. Nowak and K. Sigmund. Evolution of indirect reciprocity by image scoring. *Nature*, 393:573-577, 1998.

R. Riolo, M.D. Cohen, and R. Axelrod. Evolution of cooperation without reciprocity. *Nature*, 414:441-443, 2002.

G. Roberts and T.N. Sherratt. Development of cooperative relationships through increasing investment. *Nature*, 394:175-179, 1998.

B.H. Seghers. Schooling behaviour in the guppy (*Poecilia reticulata*): an evolutionary response to predation. *Evolution*, 28:486-489, 1974.

A.C. Spinks, J.U.M. Jarvis, N.C. Bennett. Comparative patterns of philopatry and dispersal in two common mole-rat populations: implications for the evolution of mole-rat sociality. *Journal of Animal Ecology*, 69:224-234, 2000.

The Emergence of Symbiotic Groups Resulting From Skill-Differentiation and Tags

Bruce Edmonds
Centre for Policy Modelling
Manchester Metropolitan University
bruce@edmonds.name

Abstract

The paper presents a evolutionary simulation where the presence of ‘tags’ and an inbuilt specialisation in terms of skills result in the development of ‘symbiotic’ sharing within groups of individuals with similar tags. It is shown that the greater the number of possible sharing occasions there are the higher the population that is able to be sustained using the same level of resources. The ‘life-cycle’ of a particular cluster of tag-groups is illustrated showing: the establishment of sharing; a focusing-in of the cluster; the exploitation of the group by a particular skill-group and the waning of the group. This simulation differs from other tag-based models in that is does not rely on either the forced donation of resources to individuals with the same tag and where the tolerance mechanism plays a significant part. These ‘symbiotic’ groups could provide the structure necessary for the true emergence of artificial societies, supporting the division of labour found in human societies.

1. Introduction

Sometimes when one is good at a certain activity one is necessarily not so good at others. That is to say that there can exist trade-offs between different abilities. In biological terms this might be the result of complex physical limitations – for example, if a species has a physique suitable for running very fast over small distances, this might limit the amount of fat its members can store to allow survival in lean times. In sociological cases this sort of trade-off might result from the amount of time that is necessary to acquire a certain skill – for example, one may not have time to learn to become a skilled musician and a skilled painter. Thus in an ecology one might have a variety of species, each of which is well adapted to exploit a different aspect of a particular environment. Similarly in our society one observes that people do not develop the same profession/skills but that there seems to be a spontaneous differentiation, so that in any locality many different skills possessed by different people are available.

When this sort of complementary differentiation is present, a special kind of cooperation is possible, namely that where individuals with skills contribute to the others any excess in what they produce/gather. In biology, when this relation has evolved into a stable relationship, this is called “symbiosis”. This sort of complementarity is very advanced in human societies; people are encouraged to specialise in terms of the skills in which they most excel, resulting in a huge range of careers and skills whose products are shared and traded in elaborate ways.

However there is a problem as to how such complementary sharing could arise in an evolutionary setting. The problem is this: from the point of view of an individual it is always better (at least in the short term) not to share the results of one’s labours but to accept those shared by others (this corresponds to the so-called “prisoner’s dilemma” (Axelrod 1984)). Thus, at any moment, those that are not sharing should do better than those who share, and hence produce more offspring. Thus it is difficult to see how groups of cooperatively sharing individuals could arise or be maintained – the so-called “tragedy of the commons” (Hardin 1968).

“Tags” are observable cues that can be used to recognise types of individuals (Holland 1993). They do not have any significant bearing on the abilities or behaviour of the individual. One can imagine a room full of people who do not know each other but are randomly given to wear different coloured badges, but who are able to exchange the badge for another if they wish. Although these badges are initially arbitrary it may (in the absence of other significant socially observable cues) allow the people to self-organise. Thus the colours may come to acquire a significance – the significance would emerge from the social processes in the room.

There has now been a sequence of models which show that the presence of such tags can enable the evolution of a population of temporary cooperating 'groups' of individuals with similar tags, even when there is a possibility of being invaded by selfish individuals who do not share or cooperate (Hales 2000, Riolo et al. 2001, Hales 2001). Basically what happens is this: a small "seed" collection of cooperative individuals with similar tags arises somehow; these out-perform the others due to their efficient cooperation and hence are grow in numbers by evolution; eventually defectors arise in the group (or invades from outside); now these defectors do even better than the others in that group and hence is preferentially reproduce until they comes to dominate that group; now the group does not do so well compared to other cooperative groups because there is little or no sharing and so the group dies. Thus what one observes is a continual rising and falling of different groups, so that in the population as a whole a relatively high level of cooperation/sharing is maintained. Clearly this depends on the facility with which new cooperative seed groups can arise compared to the speed with which established cooperative groups are infected and destroyed. This is closely linked to the rates of tag mutation compared to the rate of infection (Hales 2004).

This paper seeks to establish how tags can facilitate the development of (temporary) groups of complementary individuals in an evolutionary setting where individuals are not equipped with great cognitive abilities (to support contracts or elaborate foresight for example) and where individuals are not in any way forced to cooperate. This is important because this sort of process may allow the emergence some of the basic group infrastructure that, in turn, may facilitate the development of more sophisticated societies within an evolutionary setting. Thus the techniques and results in this paper can be seen as another step towards the full emergence of an artificial society.

2. Model Setup

The main assumptions that drive this model is that there are a number of different kinds of 'nutrition' (or 'product') which different individuals are specialised in gathering (or producing). However, although each individual only gathers one kind of resource they all require some of *all* the kinds of resource in order to survive or reproduce. Thus in order to survive and reproduce individuals have to be given resources by other individuals that have them, otherwise they 'starve' and die.

Each individual has the following attributes: its special skill; a tag value; a tolerance value; and the amount of resources it has of the various kinds. The skill determines which kind of nutrition it can harvest from the environment. The tag value is an arbitrary real value in [0, 1], as is the tolerance value. The resources are a record of the amounts of each kind of nutrition they have. The tag value is the only thing that is observable by other individuals.

There is no physical space in the model, only a (one-dimensional) social 'space' determined by the similarity (or otherwise) of the individual's tags. Thus one can imagine that the model represents one location or niche which they all inhabit. Each time period each individual: gathers its share of the resource it is specialised in and adds this to its store; is randomly 'paired' with a number of other individuals – if the difference in tag values is strictly less than its tolerance value and it has an excess in any of its resource stores it gives then a share of its resource; all individuals are 'taxed' a certain amount from all stores to represent consumption; finally individuals survive, die or reproduce depending upon the state of their stores. Resources degrade on transfer – thus the value of resources received in a donation event is only 0.95 of what is given.

Each time period there is a probability that the tag and/or tolerance values are mutated by the addition of Gaussian random noise. Also a small number of new individuals are continually added to the population (arriving from elsewhere). At creation,

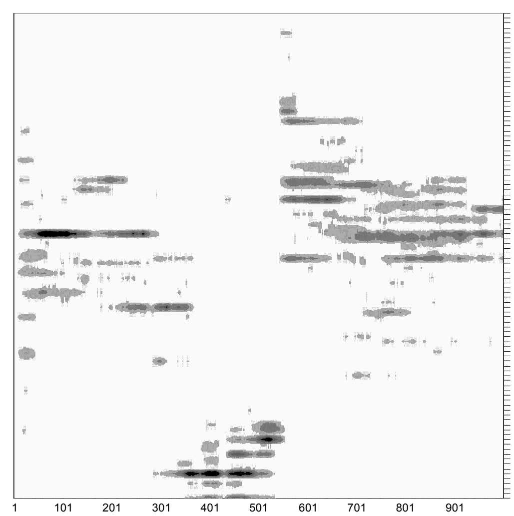


Figure 1. Tag Groups in a Single Run of the Simulation: the vertical axis are the tag values from 0 to 1; the horizontal axis is time; the shade indicates the number of individuals with that tag at that time (white<5; 5=light grey<20; 20=medium grey< 55; 55=dark grey<100; 100=black)

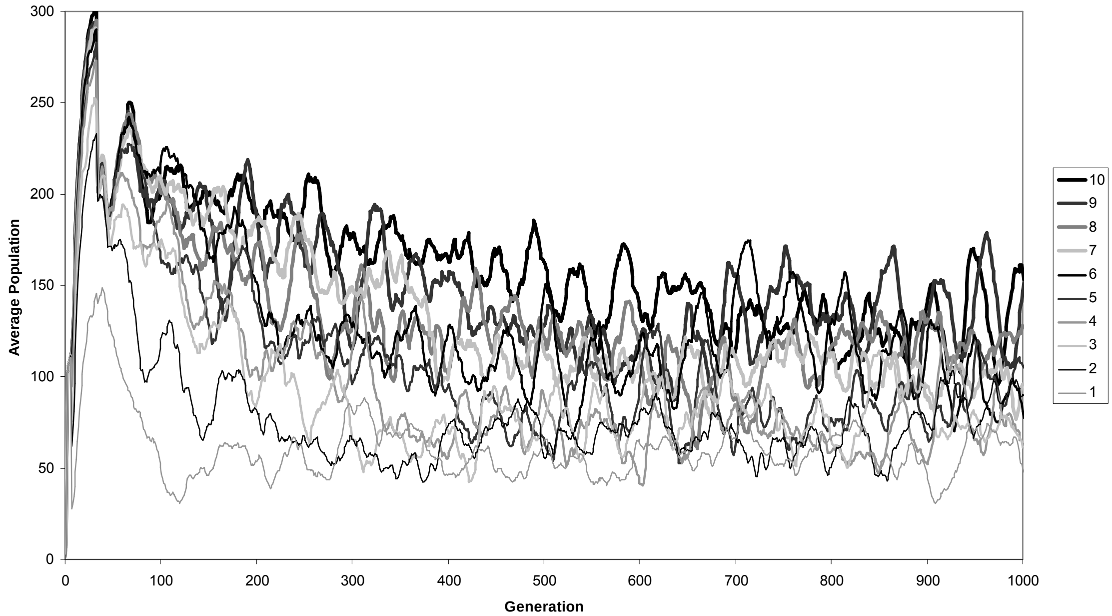


Figure 2. (average) size of population against time, with different numbers of pairings, from 1 (thinner, lighter lines, towards bottom) to 10 pairings (thicker, darker lines, towards the top)

individuals are given some (low level of) initial resources. If an individual reaches a certain age or one of its resources falls to zero that individual dies. If an individual has a certain minimum in *all* of its resource stores it will reproduce (once in that cycle) – the initial stores in the offspring are taken from those of the parent. Individual's only donate if their resource level reaches a minimum, which is higher than the minimum level necessary for reproduction. Thus individuals continually appear (arrive or are born), donate, consume resources, (possibly) reproduce, and die (of starvation or old age). The population level is thus variable – determined by the available resources and the efficiency of the sharing.

3. General Results

The simulation here has the same basic dynamics as other tag models (Hales 2000, Hales 2001, Hales 2004). That is: (1) a group of cooperating individuals happens to form, these have a higher fitness than the rest of the population (composed mostly of defectors) so they preferentially reproduce; (2) leading to a larger group of cooperators; (3) eventually a defector appears in the group who, since it gets all the benefit of the cooperators in the group but none of the costs, reproduces faster than others in the group and so; (4) eventually dominates the group so and hence kills it, since now it does not outperform other, cooperating groups. However, before the group dies another cooperating group has probably started and the process goes on. Thus although, eventually all cooperating groups are doomed, if enough new groups are continually coming into being then the overall level of cooperation across the whole population can be maintained. However this cycle causes oscillations in the population. AS we shall see, in this model, this cycle occurs within a longer cycle concerning the rise and fall of symbiotic sharing.

The rising and falling of tag groups is illustrated in Figure 1. At the start a number of tag groups form but one gains initial dominance. This dominant cluster of groups then loses out to another cluster

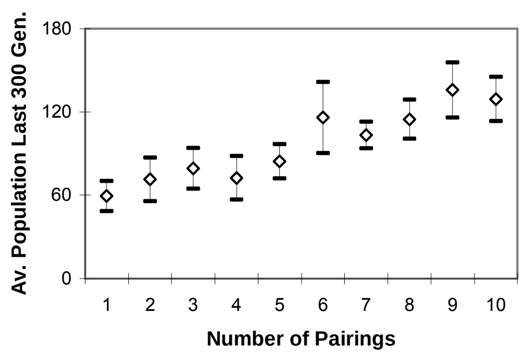


Figure 3. Average Population Size over last 300 generations against Number of Pairings (diamond is the average, bars are one standard deviation each way)

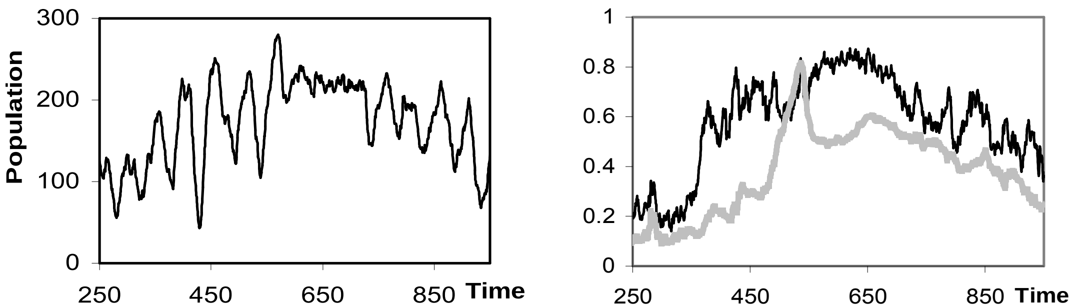


Figure 4. (left) the size of the population against time; (right, thin black) the donation rate (from 0 to 1); (right thicker grey) the average tolerance (from 0 to 1). All against time: cycles 250 to 950;

between generation 200 and 300. This is, in turn, supplanted by the cluster at the bottom between generation 300 and 400. Interestingly this group at the bottom seems to have seeded other groups near to it, but the whole cluster fails around generation 550, allowing a new cluster of groups to arise (towards the top).

Figure 2 shows the population levels over runs with different levels of pairing (from 1 to 10). Each line is the average of 6 runs. One can see that the greater the number of pairings the greater the population that can be sustained with the same input resources. A summary of the over-time averages for the last 300 cycles (when the initial effects have worn off) are shown in Figure 3. This is because with a higher rate of pairing there is a better chance of being paired with an individual which has a similar tag, allowing the evolution of more directed giving.

Since the population is directly related to the birth rate (which depends on individuals being given resources in nutrition types they can not collect) and early death by starvation (which occurs when they are not given all the kinds they need) this indicates that effective sharing is occurring.

4. Case Study: The Life Cycle of a Particular Symbiotic Group

To show the resource-sharing, tag-group mechanism in detail I examine the development and demise of symbiosis in a set of groups.

The example I examine is the period between cycle 250 and 950 in a particular run (with the default parameters given in the Appendix). This period covers a cycle from low population level up to a high level and back down again (left hand panel of Figure 4). During this period the donation rate rises to a peak before falling down again (dark line, right hand panel of Figure 4) as does the tolerance level (grey line, right hand panel of Figure 4).

Figure 5 shows a series of four ‘snapshots’ of the population distribution at cycles: 550, 650, 750 and 950. These show the distribution of tag values for each of the four skill types. By cycle 550 (top left panel of Figure 5) there has developed a scattering of tag peaks in the different skill areas, which share resources due to the high tolerances that exist at this point. By cycle 650 (top right panel of Figure 5) the group has ‘contracted’ to a tighter bunch of skill clusters with lower tolerances; by this stage one of the skills dominates the others in terms of numbers. By cycle 750 (bottom left panel of Figure 5) the sharing has become one-sided with one skill group exploiting the others, this gradually contracts to the situation at cycle 950 where this dominant group has contracted to increasingly lower tolerances. After this, these clusters die out and a new set of related skill groups arise. This is a slightly simplified account because within this ‘life-cycle’ there are subcycles of groups with same skill rising and fading.

5. Related Work

There are a number of models showing how tags can facilitate the emergence of cooperation between groups of individuals with similar tags. This model is different in: (a) no individuals are *forced* (by the model design) to cooperate with individuals with identical tags; (b) the tolerance mechanisms whereby the *range* of difference which is tolerated within groups is necessary and active; and (c) there is no ‘magic’ increase in the value of donated resources from donor to recipient.

The model presented here follows that of Riolo et al. (2001), in that it uses for a tag the intensity of a single continuous variable. Tag comparisons are thus a simple matter of taking the absolute difference in tag values. This eases the display (and hence analysis) of the distributions of values that result, also in many tag models, whether one uses a continuous space of tag values, or a sufficiently large binary space seems not to make significant difference to the results. However as (Roberts and Sherratt 2002, Edmonds and Hales 2003b) showed this model relies upon the fact that individuals are forced to donate to others with an identical tag, and that the tolerance does not play any significant part (contrary to the interpretation in: Sigmund and Nowak 2001).

Takahashi (2000) (and the tag-based variants discussed in (Edmonds and Hales 2003a)) concerned themselves with a model of generalised exchange where resource sharing resulted, but these outcomes depend on the fact that the value of a donation to a recipient is greater than the cost to a donor. That is to say that every donation has the result of increasing the resources available to the model. There is a possible (but rather forced) interpretation of this, that somehow the resource is more useful to the recipient than the donor, which could be for a variety of reasons (e.g. it was excess to the donor's needs), but this increase in value occurs regardless of the. The model in this paper can be seen as an attempt to provide a coherent story behind the difference in value, by specifying different resource needs.

6. Towards the Emergence of Complex Artificial Societies

What has been described above shows how, in a certain setting, cooperative groups of individuals with similar tags can come into being, persist for a while and dwindle out of existence. This provides some of the 'group infrastructure' for more complex social structure to develop. However in order for this to occur more is needed. Essentially the groups need to be able to spawn new groups with characteristics that are similar to that of the original group, before they are infected with defectors. If, in addition to this process, the characteristics that are transmitted from group to group were potentially complex, then all the conditions necessary for the evolution of groups would be present. Presumably those groups that were more successful at resisting infection by defectors and at 'seeding' new groups with similar characteristics as itself would (under suitable conditions) would be more successful at seeding new groups, thus allowing for a continual process of selection and reproduction of groups that are identifiable entities in their own right (identifiable via the tags). Although evolution continues to

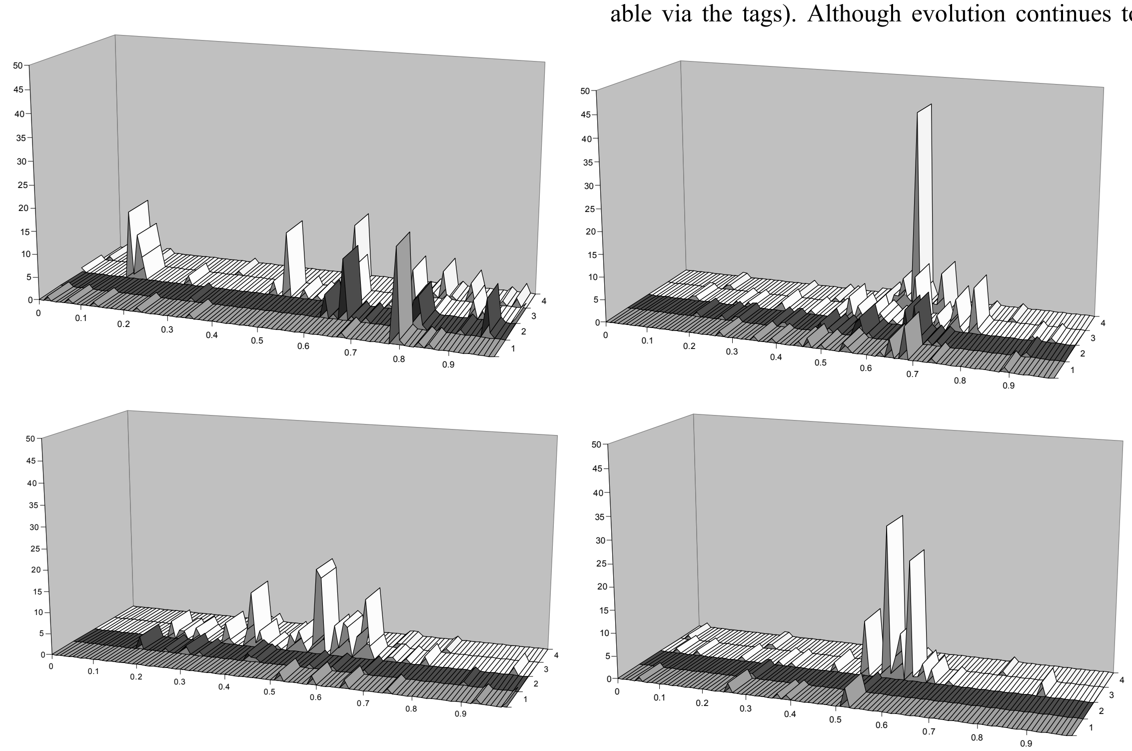


Figure 5. Life cycle of a symbiotic group, vertical axes are number of individuals (0-50); horizontal axes the tag values (0-1); different layers are different skills (1-4): (top left, time 550) high tolerance and broad sharing; (top right, time 650) low tolerance, a tighter group, and sharing; (bottom left, time 750) medium tolerance, one skill type exploiting others; (bottom right, time 950) tolerance has reduced as result of continuing exploitation.

act at the individual level, the fitness of each individual depends crucially upon the state of the group it is a member of, so if it were also the case that seeded groups had the characteristics of the groups they were seeded from (carried by the individuals who migrated out of the original group) then it would be meaningful to talk of 'group selection'[1].

Such a process would accord with the social intelligence hypothesis (Kummer et al. 1997) and that group cultures are highly adapted to the immediate environment they inhabit (Reader 1988). The social intelligence hypothesis posits that the success of our species results more as a result of our social abilities rather than our intellectual abilities. In particular it includes a sophisticated ability to imitate others, so that skills suitable in a certain environment can spread throughout a population. This suggests that our survival may have depended upon the fact that we have socially adapted as groups to inhabit a large variety of different ecological niches, such as the Tundra and the Kalahari. The cultures developed in different groups and passed down culturally within and throughout those groups are responsible for their members ability to survive and reproduce. This model can be seen as a step forward to capturing the development of such cultural plasticity[2].

Acknowledgements

Thanks to David Hales, who has done the lion's share of the work on tags, for numerous discussions on this and everything else. Also to the many others with whom I have discussed these including the participants of MABS, M2M, ESOA and ESSA.

References

Axelrod, R. The Evolution of Cooperation, Basic Books, New York, 1984.

Edmonds, B. and Hales, D. Computational Simulation as Theoretical Experiment, CPM report 03-106, MMU, Manchester, M1 3GH, UK. (http://cfpm.org/cpmrep106.html). 2003a.

Edmonds, B. and Hales, D. Replication, Replication and Replication - Some Hard Lessons from Model Alignment. Journal of Artificial Societies and Social Simulation 6(4). http://jasss.soc.surrey.ac.uk/6/4/11, 2003b.

Hales, D. Cooperation without Space or Memory: Tags, Groups and the Prisoner's Dilemma. In Moss & Davidsson, (eds.) Multi-Agent-Based Simulation. LNAI 1979:157-166. Springer. Berlin, 2000.

Hales, D. Understanding Tag Systems by Comparing Tag Models. Second Model-to-Model Workshop (M2M2) at the Second European Social Simulation Association Conference (ESSA'04) Valladolid, Spain 16-19th Sept 2004.

Hales, D. Tag Based Co-operation in Artificial Societies. PhD Thesis, Department of Computer Science, University of Essex, UK, 2001.

Hardin, G. The Tragedy of the Commons, Science, 162:1243-1248, 1969.

Holland, J. The Effect of Labels (Tags) on Social Interactions. SFI Working Paper 93-10-064. Santa Fe Institute, Santa Fe, NM. http://www.santafe.edu/sfi/publications/wplist/1993, 1993.

Kummer, H., Daston, L., Gigerenzer, G. & Silk, J. The social intelligence hypothesis. In Weingart et. al (eds.), Human by Nature: between biology and the social sciences. Hillsdale, NJ: Lawrence Erlbaum Associates, 157-179, 1997.

Margulis, L. and Sagan, D. Microcosmos: Four Billion Years of Evolution from Our Microbial Ancestors. Summit Books, New York, 1986.

Reader, J. Man on Earth. London: Collins, 1988.

Riolo, R. L., Cohen, M. D. and Axelrod, R. Evolution of cooperation without reciprocity. Nature, 411:441-443, 2001.

Roberts, G. and Sherratt, T. Does similarity breed cooperation? Nature 418, 499-500, 2002.

Sigmund, K. and Nowak, M. A. Evolution - Tides of tolerance. Nature 414:403, 2001.

Takahashi, N. The Emergence of Generalized Exchange". American Journal of Sociology 105:1105-1134, 2000.

Barkow, J. H., Cosmides, L. & Tooby, J. (Eds.), *The adapted mind: evolutionary psychology and the generation of culture*. Oxford University Press, 1992.

Wilson, D.S. & Sober, E. Reintroducing group selection to the human behavioural sciences. Behavioral and Brain Sciences 17 (4): 585-654, 1994.

[1] A review of the literature on group selection is (Wilson and Sober 1994)

[2] For background on these ideas see (Barklow et al. 1992).

Appendix – More about the model

Sources

The model structure of tags and tolerances in a [0,1] range comes from (Riolo et al. 2000). The motivation for improving on this model came from (Edmonds and Hales 2003a, 2003b). The nutrition structure that was added on was suggested by reading texts on evolution and symbiosis, including (Margulis and Sagan 1986).

Structure

There are fixed number of nutrition types and corresponding skills for gathering that type.

There is a variable population of individuals, each of which is characterised by the following characteristics: a tag value (a real from [0,1]); a tolerance value (a real from [0,1]); a skill type (an integer); for each nutrition type: a reservoir holding an amount of that resource (a real), and an age (an integer).

Resource Flow

Resources are broadly conserved within each nutrition type. It enters via distribution and leaves via dissipation, waste and with the death of individuals.

They principally enter the model via the direct distribution of units in the form of the different nutrition types. These are randomly distributed to these four kinds, then all those individuals who posses the appropriate skill to gather that resource kind, equally share that resource.

Also new individuals (the initial population, the 2 new individuals that enter the population each time, and the progeny of individuals that reproduce) are given a fixed amount in each reservoir (*intialFood*). In the case of reproduction these amounts are subtracted from the corresponding reservoirs of the parent.

Each individual is now randomly paired with a fixed number (*numPairings*) of other individuals. In each pairing event an amount of the resource may be transferred from giver to recipient, if some conditions are satisfied. These conditions are: (1) The recipient must be one of those randomly chosen that time; (2) the difference in tag values must be strictly less than the tolerance of the giver; and (3) the giver must have more than a set amount (*foodOfTypeAboveWhichIsExtra*) in the corresponding reservoir. Each donation *donationCost* is subtracted from the giver but only *donationBenefit* given to the recipient. The excess in the reservoir is shared equally among all recipients who qualify.

The individuals' reservoirs can only store up to a fixed maximum (*maxResevoir*). Above that resources are simply lost.

Each unit of time, a 'life tax' is subtracted from each reservoir of each individual.

If an individual has accumulated more than a fixed amount (*foodOfTypeNecessaryForReproduction*) in all of their reservoirs then the reproduce. The resources in the offspring are subtracted from the parent.

If an individual has less than a fixed amount (*foodOfTypeBelowWhichTagDies*) in any reservoir then it dies, also if it has reached its maximum age

```
Generate individuals, giving them all initialFood in each resource and each ar
independent randomly chosen skill, tag and tollerance
For each generation
        Add maxNumNew new individuals (with random tags)
        Units of resource are randomly distributed amoung nutrition types – individuals
        with a skill equally share in that type
        For each individual, D
                For each pairing from 1 to numPairings
                        Randomly choose another individual without replacement,O
                        For each resource type, R
                                If
                                        D has more of R than foodOfTypeAboveWhichIsExtra
                                        and the absolute difference between D's and O's tag
                                        is strictly less than D's tollerance
                                Then
                                        Subtract donationCost in R from D
                                        Add donationBenefit in R to O
                        Next resource type
                Next pairing
        Next individual
        For each individual
                subtract foodUsageRate from each resource
                        If any resource < foodOfTypeBelowWhichTagDies then it dies
                If
                        all resources > foodOfTypeNecessaryForReproduction
                then
                        replicate indidivual (with possible mutation), subtracting new
                        progeny's resources from parent
        Next individual
Next generation
```

Figure 6. An outline of the simulation algorithm

(*maxTagAge*). Resources of those that die are lost.

Principle Dynamics

Once going mutation can occur in tag values, tolerance values or skills. Apart from such mutation the main changes are in the number of individuals with particular skills and tags which results from their success in getting enough nutrition of all types so as (a) not to prematurely die and (2) to reproduce. The fact that all individuals will eventually die (if only because they reach the maximum age) makes the whole system more dynamic.

Initialisation

Each individual was given a random skill, a random tag value, a random tolerance, a zero age and *initialFood* in each of its reserves.

Algorithm

The algorithm is outlined in Figure 6.

Intended interpretation

There are two interpretations for this simulation: one biological and one social. In the biological interpretation: the individuals have the skills genetically encoded and a variety of nutritional needs; these individuals also have a genetically determined ability to recognise those with a similar tag to themselves and donate some of their nutritional excess; the simulation shows how in such a situation symbiotic relationships might evolve. In the social interpretation: individuals choose to develop one of a set of skills and may share the results of applying these skills with others if they judge them similar enough to themselves (in terms of socially observable signals); skills may change and be passed on to their offspring; the simulation shows how cooperative groups based on sharing by individuals with complementary skills might occur.

Details Necessary for the Simulation to Run but not thought Important

Exactly how the population is initialised is not thought to be important, in trial runs with (for example) zero tolerances, cooperative groups did eventually form and the processes described above take place, but this took longer to "get going" (assuming sufficient mutation rates).

Default parameter values

```
initialPopSize = 100
maxTime = 1000
maxRun = 1
numPairings a value from: {1, 2, …, 10}
probMutVal = 0.1
sdMut = 0.1
maxNumNew = 2
donationCost = 1
donationBenefit = 0.95
numFood = 350
numSkillBits = 2
numNutritionBits = 2
maxTagAge = 30
maxStartAge = 0
intialFood = 1
foodOfTypeNecessaryForReproduction = 4
foodOfTypeBelowWhichTagDies = 0
foodOfTypeAboveWhichIsExtra = 5
foodUsageRate = 0.25
maxResevoir = 20
maxTol = 1
```

Description of variations

The variations explored in this paper are limited to those of differing extents of pairing for possible donations. Other investigations have varied other parameters, including the number of different nutrition types (and associated skill types), the level of reserves that individuals can carry, level of nutrition put into the system, and the levels of mutation.

Verification of Code

This model was reimplemented in Java by David Hales. Although we got similar results we did not finish 'docking' these models to check it 100% (as in Edmonds and Hales 2003b). However the results were broadly compatible with what we have found in other tag-based models.

Software Environment and Code

This model was implemented in SDML version 4.1. This is freely available for academic use (sdml.cfpm.org) but is currently unsupported. The code can be obtained from the author by email, but takes at least a basic knowledge of SDML to run.

Behavioural Norms and the Evolution of Societies

Stephen M. Younger
Los Alamos National Laboratory
Los Alamos, NM 87545 USA
smyounger@lanl.gov

Abstract

Behavioural norms play an important role in the evolution of human and artificial societies. Using positive and negative reciprocity (sharing and stealing) as sample normative and non-normative behaviours, discrete agent simulations were used to examine the effects of several models for the transmission of normative character across generations of agents in societies with varying levels of tolerance to transgression. We found that when reputation was a factor in mate selection, a form of individual sanction, population survival was sensitive to tolerance for non-normative behaviour. The survival probability was highest for very low and very high levels of tolerance, but decreased in between. This effect occurred in all of the inheritance models considered, including one that allowed normative character to be modified by the experience of the agent. Group sanctions, represented by ostracism, enabled a normative population to survive with much higher tolerance than was possible with only individual sanctions. We propose that the relation of tolerance to population stability may be one reason for the remarkable uniformity of behavioural norms among simple societies.

1 Introduction

All societies possess a set of behavioural norms that guide the actions of their members. A key question in evolutionary social dynamics is whether there is some *preferred* set of norms that optimize the probability that a given type of agent will prosper in a given environment. Boehm (1999) has observed that the normative systems of simple egalitarian societies are remarkably similar the world over, even though there was little opportunity for contact between them. One reason for this similarity might be that significant deviations from accepted practice would not produce a sustainable society. To investigate this issue, we model a population of agents practicing positive and negative reciprocity, i.e. sharing and stealing. We assume that reputation is enhanced by sharing and decreased by stealing and we use reputation as the determinant for mate selection. What tolerance to theft will result in a stable society and how will this vary for different models for the transmission and development of agent character?

Reciprocity has been studied within an artificial society by Jaffe (2002), Castelfranchi et al (1998), and Younger (2003) and has been extensively studied in human and primate societies as described by, for example, Gowdy (1998). The concept of tolerance to theft, introduced by Blurton Jones (1984), has been investigated by several anthropologists including, for example, Bliege Bird and Bird (1997). This paper presents a summary of work described more extensively in Younger (2004) and Younger (2004a).

2 Methodology

We modeled an isolated population by placing 100 agents on a 20x20 square grid containing five fixed sources of food. The population was divided equally between male and female agents and between agents who shared food and those who stole food.

Each of the five food sources had an initial allocation of 100 points and was replenished at a rate of 20 points per timestep. Since each agent required one food point per timestep, this meant that an average population of 100 agents could be sustained. Agents had the ability to sense food sources and other agents from a distance of five squares in each direction. This finite sensing range prevented them from seeing the entire landscape in a single view and required them to move about to search for a food source that was in supply. When an agent sensed a food source it recorded the location and the amount of food present.

The agent's need for food was increased by one unit per timestep and was decreased by the amount of food consumed at a food source. If this need exceeded 200 units then the agent died and was removed from the population. Agents lived a maximum of 4000 timesteps, at which point they died of old age. To avoid having a large number of agents simultaneously die of old age, the initial population was evenly divided in age over the interval 0 – 2000 time units. At the beginning of the simulation the agents were randomly scattered across the landscape. Simulations lasted 40,000 timesteps, or ten agent lifetimes, so the initial conditions were quickly changed by agent movement and by the creation of new agents via procreation. Results presented below represent averages taken over twenty individual runs.

The sequence of agent decisions was as follows: If the need for food was greater than 100 points, 50% of the maximum, then the agent tried to find and consume food. If the need for food was less than 100 points then the agents explored the landscape, noting which food sources had food for future use. If it encountered another agent, it shared or stole depending on its character and whether it or the other agent was carrying food.

In attempting to satisfy its hunger, an agent first consumed any food that it was carrying. If its need for food was still above 100 points, and if it was collocated with a food source, it consumed up to the amount required to reduce its hunger to zero and, if there was some left over, it collected up to 100 units to take along on its travels. If the agent was not carrying any food and it was not at a food source, it searched its memory for the nearest food source at which food was present. It then moved one square in the direction of that food source. If the agent did not know the location of any food source that was in supply, it set out in a random direction in hopes of finding one.

When two agents occupied the same location, and when one of them was carrying food, there was an opportunity for sharing or stealing. If the carrying agent was a sharing agent, it shared its food equally among the other agents occupying that square. If a stealing agent occupied the same square as an agent carrying food, then the stealing agent stole all of that food. Sharing agents did not steal and stealing agents did not share.

An interaction matrix, imx, tallied the history of interactions between the agents. When agent j shared with agent k, the amount of food shared was added to matrix element imx(k,j). When agent m stole from agent n, the amount stolen was subtracted from imx(n,m). The non-symmetric interaction matrix constituted a form of absolute normative reputation for the agents. When two agents were collocated, they communicated their knowledge of normative reputations by averaging their interaction matrix elements for all of the other agents. In this manner agents would obtain information on the normative character of other agents without having personal experience of those other agents.

Female agents chose a mate upon reaching the minimum reproductive age of 1000 time units. They selected the unmarried male with whom they had the highest interaction matrix element. A female could refuse to mate with a male or a male could refuse the offer of mating if the negative of the interaction matrix element linking them to the potential mate was greater than a tolerance level, which was the same for all members of the population. Since sharing added to interaction matrix elements and theft reduced them, a positive tolerance allowed agents who had stolen to secure a mate. A negative tolerance required a potential mate to have shared in order to find a mate. In this way the tolerance to theft was linked to the reproductive strategy of the population. Mating was monogamous and the female was required to collocate with her husband. If either mate died, the survivor was free to choose another mate.

Mates aged between 1000 and 3000 timesteps could produce offspring. The probability of conception was 0.004 per timestep, chosen to allow a population to survive but not overpopulate the landscape given the limited food resources and the finite lifetime of the agents. Offspring appeared immediately, with no gestation period.

We examined four models, given in Table 1, for the transmission of behavioural characteristics between generations. In the Fixed Distribution model a new agent was assigned a behavioural character (sharing or stealing) so as to maintain a fixed percentage of sharing agents in the population. In the Matrilineal Inheritance model an agent was assigned the normative character of its mother. This was meant to model the effect of early childhood nurture as well as any genetic influence on willingness to abide by norms.

To study the effect of deviations from matrilineal inheritance, we introduced a model where 5% of the agents had the opposite character of the mother. Finally, to approximate the effect of experience on character, we modeled a population described by a parameter A_{sh} where $0 < A_{sh} < 1$ and where agents with $A_{sh} <0.5$ stole and agents with $A_{sh} > 0.5$ shared. The value of A_{sh} for a new agent was set to the average of its parents. Each sharing event caused the A_{sh} of the recipient to increase by 0.1 and each theft causes the A_{sh} of the victim to decrease by 0.1. Hence, experience could change a sharing agent into a stealing agent or vice versa.

Table 1: Models used to study inheritance and modification of normative character

Model	Description
Fixed Distribution	Character of offspring assigned so as to ensure fixed distribution of sharing and stealing agents
Matrilineal Inheritance	Character of offspring set to the character of its mother
Matrilineal Inheritance + 5% Noise	Character of offspring set to character of its mother except in 5% of cases, in which it is set opposite to that of its mother
Genetic + Experience	Character represented by variable A_{sh} where $A_{sh} < 0.5$ implies a thief and $A_{sh} >0.5$ implies a sharer. Character of offspring set by averaging A_{sh} values of parents. Sharing adds 0.1 to A_{sh} of recipient of sharing and subtracts 0.1 from A_{sh} of victim of theft.

3 Results

Table 2 shows the results of the Fixed Distribution model; it gives the percentage of the 20 runs for each parameter set that had a non-zero population at the end of the run, i.e. after 10 lifetimes of simulation time.

For a population of 90% sharing agents the population survived in more than half of the runs when tolerance was equal to - 4, i.e. it was possible to *require* sharing of a potential mate. For larger concentrations of stealing agents, tolerance needed to be greater than zero for the population to survive. Thus for 50% stealing agents, tolerance had to be greater than 16 for the population to survive in more than half of the runs and for 90% stealing agents tolerance had to be more than about 80. A greater tolerance to theft was required for the higher rate of transgressions committed by the larger stealing subpopulation.

Table 2: Percentage of runs with a population that survived until the end of the run for various fractions of sharing vs. stealing agents and various values of tolerance to theft. Blank entries were not simulated.

Tolerance	90% Sharing	50% Sharing	10% Sharing
-16	0		
-8	10		
-4	60		
-2	90		
0	90		
8		0	
12		10	
16		60	
20		80	
32		100	0
64			20
96			75
128			85
160			95

Now let us look at simulations where the character of offspring was set to that of their mother. The results for the inheritance models are shown in Figure 1 and indicate that the population was likely to survive when tolerance was either low or high – in between there was a significant probability of population collapse. When tolerance was low, individual agents were strict in their selection of mates. Those agents who had a reputation for theft were effectively excluded from the mating pool and hence could not pass along their “theft gene” to the next generation. An all-sharing population was the result. Conversely, when tolerance was very high the reputation of the potential mate was effectively irrelevant to mate selection, an “anything goes” type of society. Here the short-term benefit of theft carried with it no long term disadvantage in security a mate and a completely stealing population was the result. For intermediate values of tolerance, a detailed analysis of the simulation revealed that the subpopulation of sharing agents disappeared early in the run since it could not sustain itself under frequent thefts. However, once the sharing subpopulation was eliminated, the thieves had a difficult time finding mates amongst their own kind since frequent thefts degraded their reputation so as to exceed the tolerance level for mate selection. In effect, the long term *disadvantage* of theft dominated the dynamics of the stealing population.

When 5% noise was introduced in the inheritance of normative character, there was a constant replenishment of the sharing population by the random assignment of sharing character to 5% of the agents. This small number of sharing agents was enough to enable some within the largely stealing population to find a mate and hence improve the overall survival probability.

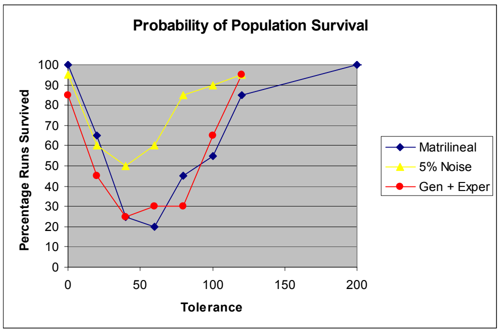


Figure 1: Probability for survival of the population until the end of the run for three models of the inheritance and modification of normative character

The Genetic + Experience model had a survival curve almost identical to that of the straight matrilineal inheritance model, despite its inclusion of an admixture of paternal character and its modification of character based on the experience of the individual. It was the distribution of sharing and stealing agents in the population that was most important rather than the means by which that distribution was established.

So far, only individual sanctions – refusal to mate - have been applied to non-normative agents. Collective sanctions are another means for a society to prosper in the face of occasional transgressions. We studied a form of collective sanction by dividing the initial population of agents into two equal groups with newborn agents assigned to their mother's group. In the Baseline Scenario, agents remained in their initially assigned group for their entire lives. In the Ostracism Scenario, agents were ejected from their group if the average interaction matrix element connecting them to the other members of the group exceeded the tolerance level. They then had the opportunity to apply to the second group for membership but if they were similarly rejected from that group then they became "ostracized". Agents in groups did not share with ostracized agents nor did they choose them as mates. Figure 2 shows the probability that a population would survive in each case and demonstrates that ostracism shifted the minimum in the survival probability to much greater degrees of tolerance when offenders were removed from frequent contact with the group population. At very high values of tolerance, there were no penalties to theft in securing a mate and the probability of survival increased.

Analysis of the runs indicated that ostracism decreased the rate of interaction between sharing and stealing agents and hence reduced the damage done by the latter on the former.

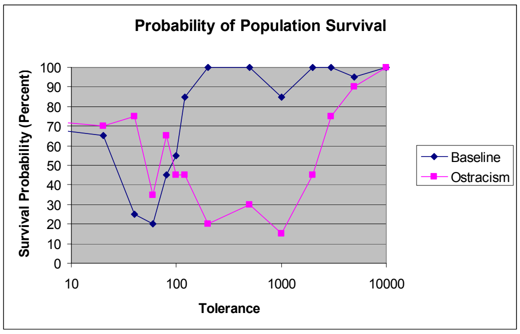


Figure 2: Probability of survival for the Baseline (no ostracism) and Ostracism Scenarios.

4 Discussion

Our results suggest that normative character, the means by which it is transmitted to the next generation, and the tolerance of transgressions within the population, are important factors in determining long-term survival when normative character is a factor in mate selection. A population consisting largely of sharing agents will benefit from a greater equity in the distribution of finite resources. A population of largely stealing agents is also sustainable since theft is another means of resource distribution within the population. (This ignores, of course, the effect of theft on social cohesion or other non-material values within the society.)

While the probability of survival for the population differed among the models used to transmit normative character to the next generation, the qualitative conclusion that moderate levels of tolerance led to increased risk of population collapse was observed in every case. This suggests that it is the normative character of the population, rather than the means by which it was produced, that is most important in promoting social stability in the artificial society.

Numerous studies of the evolution of normative behaviour have been performed for agents playing the prisoner's dilemma and similar games. See, for example, the discussion in Ridley (1996). In his "ecological" tournaments, Axelrod (1984) allowed agents practicing two or more fixed strategies to compete against one another for dominance. In successive generations, more successful strategies were given to more agents and less successful strategies were given to fewer agents, or eliminated entirely. This contrasts with "evolutionary" competitions in which the strategies themselves were changed by means of systematic and/or random "genetic" changes. Most of our simulations were "ecological" in that only one of two fixed sets of behavioural rules – sharing or stealing - could be inherited. Agents who lived longer had more opportunity to mate and hence propagate their normative character. Our "genetic + experience" model does allow some modification of character based on experience, but it is still based on past interactions rather than calculated expectations of future behaviour.

While the benefits of sharing and the communication of normative reputation have been studied in several contexts (e.g. Castelfranchi et al, 1998 and Jaffe, 2002) most previous simulations have focused on the economic benefits of improved access to food rather than social implications such as access to mates or the development of mutual obligation and social cohesion within the society. A particular challenge in the latter is the difficulty of defining appropriate parameters for monitoring non-material factors in social performance.

There are several interesting applications of these results to real and artificial societies. It is notable that sharing is nearly ubiquitous among egalitarian societies living in environments as diverse as the Kalahari, the Australian outback, Polynesia, and the Artic. Also, nearly all egalitarian societies exhibit strong individual norms with low tolerance to transgression in preference to collective sanctions such as ostracism. Indeed, I have been unable to find any account of a culture that persisted over several generations with a high tolerance to transgression. Is there a reason why cultures in such different environments exhibited a high prevalence toward positive reciprocity and the practice of strong individual sanctions? Might one reason be because significant deviations from these practices would not produce a sustainable society? Perhaps these peoples recognized this fact and designed their society accordingly or perhaps we only see the results of societies that lived long enough to leave remains. It would be interesting to see if similar phenomena occurred in a self-directed artificial society that created its own norms based on observations of what behaviour was most successful in satisfying agent needs. One might find that sustainable societies fall into a relatively narrow band of behaviours and that others persist for a limited time, only to collapse later.

We used a rule based model to describe the effects of sharing and stealing on population survival under various conditions of tolerance. This has the advantage of enabling cause-effect relationships to be established between rules and results. An alternate approach would be to emphasize learning by the agents, either by the calculation of some type of utility function or by giving the agents the ability to learn as a result of experience and observation. Future work might investigate other factors that contribute to agent reputation including kinship, physical prowess, charisma, etc. Artificial societies could serve as an interesting testing ground for alternate models of "nature" vs. "nurture" in determining normative character.

References

R. Axelrod. *The Evolution of Cooperation*. New York: Basic Books, 1984.

R.L. Bliege Bird and D.W. Bird. Delayed Reciprocity and Tolerated Theft: The Behavioural Ecology of Food-Sharing Strategies. *Current Anthropology* 38: 49-78, 1997.

N.G. Blurton Jones. A Selfish Origin for Human Food Sharing: Tolerated Theft. *Ethology and Sociobiology* 5: 1-3, 1984.

C. Boehm *Hierarchy in the Forest: The Evolution of Egalitarian Behavior* Cambridge: Harvard University Press, 1999.

C. Castelfranchi, R. Conte, and M Paolucci. Normative Reputation and the Costs of Compliance, *Journal of Artificial Societies and Social Simulation,* 1(3), 1998.

J. Gowdy. *Limited Wants, Unlimited Means: A Reader on Hunter-Gatherer Economics and the Environment*. Washington DC: Island Press, 1998.

K. Jaffe. An Economic Analysis of Altruism: Who Benefits from Altruistic Acts? *Journal of Artificial Societies and Social Simulation* 5(3), 2002.

M. Ridley. *The Origins of Virtue*. New York: Penguin Books, 1996.

S. Younger. Discrete Agent Simulation of the Effect of Simple Social Structures on the Benefits of Resource Sharing. *Journal of Artificial Societies and Social Simulation,* 6(3), 2003.

S. Younger. Reciprocity, Tolerated Theft, and Reproduction Strategies in Foraging Societies. Los Alamos National Laboratory Report LA-UR-04-7119, 2004.

S. Younger. Reciprocity, Sanctions, and the Development of Mutual Obligation in Egalitarian Societies. Los Alamos National Laboratory Report LA-UR-04-7118, 2004a.

Finding Cooperative (tag-based) Stereotypes with Semi-Automated Searching

David Hales
University of Bologna
Mura Anteo Zamboni 7, Bologna, Italy
dave@davidhales.com

Abstract

An Artificial Society model in which cooperation between agents is mediated by stereotypes is presented. The stereotypes evolve memetically via "cultural learning". The model specifies a large number of exogenous parameters. Two methods of exploring this very large parameter space are applied in the search for regions that produce high levels of agent cooperation. A region is located in which a novel cooperation process is identified based on the formation of agent "tag" groups.

1 Introduction

Firstly, we introduce the motivation for the StereoLab artificial society (section 2), then we outline the computational implementation which specifies a number of exogenous parameters (section 3). We apply two methods to explore the parameter space of the mode to locate regions in the space producing high levels of cooperation between agents (section 4). We identify a region in which a novel form of cooperation forming process (based on group formation) is found. This process appears to harness a mimetically driven emergent stereotyping mechanism. We consider both the novel cooperation mechanism and the methodology of semi-automatically searching the parameter space as of interest to those interested in emerging complex socio-cognitive structure in artificial societies.

2 Should You Trust a Hippie?

In human societies people often have to interact cooperatively with others who they have never met before and therefore have no specific knowledge of. In those situations how does an individual select an appropriate behaviour? Specifically, in an economic transaction where trust is involved, when should a stranger be trusted? When should a stranger be cheated? Consider the following scenario:

"Imagine you are driving across country for a family vacation when your car overheats. You have the car towed to a service station that has a repair shop. The mechanic says you need an expensive new radiator. It is a hot and humid August day, the kids are cranky, and you are in no mood to pay to have your car towed to another shop for a second opinion. You have no assurance that the mechanic is telling the truth or will charge a fair price or do proper work. What should you do? Meanwhile, the mechanic is equally worried that an out-of-town motorist may skip out on a bad check." (Macy & Skvoretz, 1998).

In this scenario, both you and the mechanic will benefit if a fair deal can be struck. However, how can either party trust the other not to cheat? What knowledge can you both draw on to make a decision? One mechanism for coping is to make use of "social cues". Both you and the mechanic assess the situation, observe each other and draw on socially or individually gained knowledge to come to a decision on how to act. If a "similar" mechanic in the past did a poor job and overcharged then you might be tempted to write a bad cheque since "this guy looks like a cowboy mechanic". Conversely, if the mechanic observes that you are wearing a caftan and have long hair he may conclude you are a "no-good hippie" who is simply not to be trusted. He may overcharge for poor work or worse may refuse to help you. The mechanic may have never met a "no good hippie" in person before but those he socially interacts with have told him anecdotes of bad deeds. He has been told to watch out for people like this. The point is that individuals may judge others based on personal experience or socially learned beliefs and socially learned beliefs may or may not have some relationship to some real experience, they could simply be myths of uncertain origin and veracity.

2.1 Cues, Stereotypes and Social Distance

Stereotypes are defined here narrowly as knowledge that associates sets of attributes with sets of individuals based purely on observable characteristics (social cues, cultural markers or tags). It is assumed that stereotypes are constructed, maintained and evolved through social interactions between individuals over time. It is also assumed that different individuals may posses different (even conflicting) stereotypes and that the processes that generate them are due to the need for cognitive efficiency and the selection of social strategies based on very limited information. The social psychological literature refers to this characterisation of stereotyping as the "information processing error" explanation (Oakes, P. et al, 1994). This is opposed to the "kernel of truth" position that proposes stereotypes are based (at least in part) on true group differences embedded in the structure of society. However, it can be argued that the "structural differences" from which stereotypes may be generated may themselves be the result of processes involving stereotyping (among other cognitive and social processes) and hence are reflexively[1] related rather than simply reflectively related or false. Social cues in the form of dress, accent, physical characteristics etc. may be used by individuals to make comparisons of "social distance" between themselves and others. It is well documented that individuals often prefer to associate with others who are similar to themselves (Tajfel et al, 1971). Social cues therefore may often be used as mechanisms to enforce forms of social exclusion (either economically or culturally) by creating in-groups and out-groups from populations of individuals. Some social cues (or tags) may be easily changed via social influence (e.g. dress or accent) but others are hard to change or disguise (e.g. sex or racial characteristics). So, two kinds of cues may be delineated: fixed traits and culturally learned traits. Either or both of these kinds of cues may be used in processes of social distance estimation. Extreme examples of such practices manifest themselves in communities such as the American Amish (Hostetler 1998). But less extreme forms of social and economic exclusiveness permeate most societies, often involving sets of overlapping, emerging and dissolving groupings. Numerous social psychological studies (Oakes et al, 1994; Leyens et al., 1994) find that individuals within groups are highly oriented towards their own group both in terms of actively harmonising their beliefs and behaving in a more altruistic way towards in-group members (Kramer & Brewer, 1984) and adapting stereotyped and negative attitudes towards out-group members (so called "in-group bias").

2.2 Salient Features

From the above discussion and example scenario some salient features may be outlined:

- Agents learn socially and individually.
- Interact with strangers is often required.
- Mutually beneficial interaction between strangers may require trust.
- Agents may evaluate strangers with reference to observable social cues.
- The social cues may be culturally learned and propagated.
- Agents often prefer to interact with those holding similar cues.

The StereoLab artificial society attempts to minimally capture these salient features. The Prisoner's Dilemma game (see section 3.1 below) models trust-based interactions and social learning and the propagation of cues and stereotypes is modelled via a minimal "memetic" process. Exclusion practices based on cues are captured by the biasing of game and cultural interactions based on tags. Tags may be fixed or change via cultural interaction. In the following section we describe the computational model in detail.

3 The StereoLab Artificial Society

The aim of the StereoLab design is to capture, in a highly abstracted form, the salient features outlined in section 2.2 above. Throughout the design of the society, important assumptions have been parameterised. Firstly we introduce how a certain kind of interaction based on trust can be modelled as the Prisoner's Dilemma game.

3.1 Trust as a Game

The Prisoner's Dilemma (PD) game models a common social dilemma in which two players interact by selecting one of two choices: Either to "cooperate" (C) or "defect" (D). From the four possible outcomes of the game, payoffs are distributed to the individuals. A reward payoff (PR) and a punishment payoff (PP) are given for mutual cooperation and mutual defection respectively. However, when individuals select different moves, differential payoffs of temptation (PT) and sucker (PS) are awarded to the defector and the cooperator respectively. Assuming that neither player can know in advance which move the other will make and wishes to maximise its own payoff, the dilemma is evident in the ranking of payoffs: PT > PR > PP > PS and the constraint that 2PR > PT + PS. Although both players would prefer PT, only one can attain it. No player wants PS. No matter what the other player does, by selecting a D move a player ensures he gets either a better or equal payoff to his partner. In this sense a D move can't be bettered since playing D ensures that the defector can not be suckered.

The selection of a cooperative strategy by a player in the PD can be seen as a form of trust. The player exposes itself to exploitation by defection

[1] By "reflexively" related, I mean that the stereotyping process affects the very groupings that are represented by stereotypes.

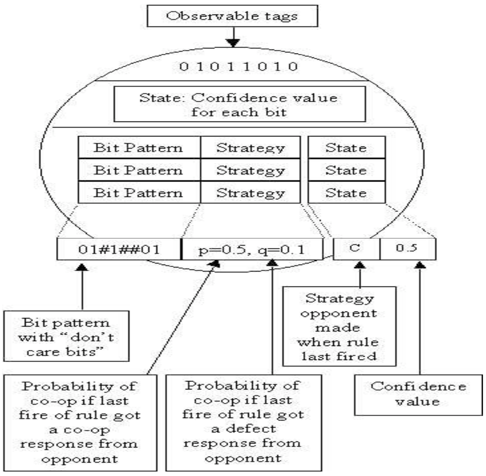


Figure 1: An agent in the StereoLab. An agent consists of a set of tag bits (observable by other agents) and a set of rules (stereotypes) mapping bit patterns to game strategies. Each tag and each rule has an associated confidence value.

from the other player. Trust in this context represents some action that exposes the player to exploitation by another player when no binding agreement or contract is imposed. Trust, here, is seen as an interpretation placed on the action of an agent not a cognitive state of an agent. The StereoLab models economic interactions using pair-wise singe-shot PD game-interactions between agents (players).

3.2 Social Cues as Tags

Labels or tags are defined as observable attributes attached to agents (Axelrod, 1980; Holland, 1998; Riolo, 1997). In a binary string representation of a tag, each bit can be interpreted as representing the presence or absence of some observable characteristic. The definition of tags used by Holland (1993) specifies that they are fixed and unchanging intra-generationally but evolve inter-generationally. The interpretation here, therefore, is one of physically observable properties linked to genetic material. The role of tags as methods of increasing cooperation in Iterated PD games has been discussed by Holland (1992; 1998) and more recently Riolo (1997) and Cohen et al. (1999). In these latter studies, experimentation with computational models demonstrates empirically that tags can increase cooperation in the iterated PD game. However, tags have been used to represent cultural attributes that can be copied intra-generationally between agents in order to abstractly capture a form of cultural group formation (Axelrod 1995; Epstein & Axtell 1996). The interpretation in these cases is one of cultural characteristics gained through cultural interactions (e.g. style of dress, social demeanour etc.), which dynamically form identifiable cultural groups. Tags in the StereoLab may be both unchanging and fixed (the interpretation being of unchanging physical characteristics) or culturally learnable and mutable (the interpretation being of cultural traits such as style of dress).

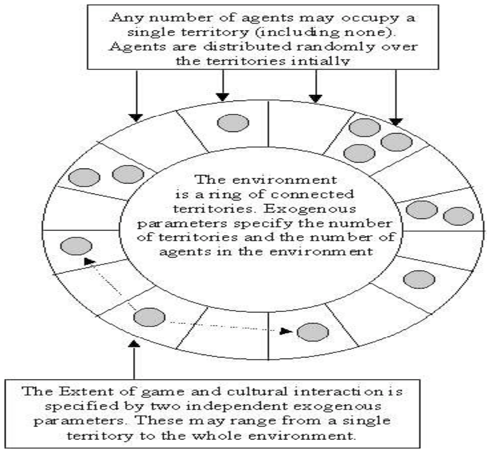


Figure 2. The StereoLab interaction environment. Agents inhabit a ring of connected territories. Each territory may contain any number of agents (including none). An agent culturally and game-interacts over some proportion of territories specified by exogenously set parameters.

3.3 Agents

Individuals are represented as simulated agents displaying tags represented as binary bit strings (social cues). Agents encounter each other dyadically and play a single round of the PD game that may be thought of as an economic interaction requiring trust. Agents store a set of rules that map tag patterns to PD strategies (stereotypes). Figure 1 shows a schematic diagram of a StereoLab agent. Cultural interaction between agents also occurs dyadically and involves the propagation of tags and rules (treated as memes). Agents inhabit a one-dimensional ring comprising a set of independent territories that may contain any number of agents including none (see figure 2).

Agents comprise a set of observable tags (bit strings), a set of behavioural rules and some state (memory) associated with each rule. The number of bits and rules stored are specified by exogenous parameters. Some proportion of the tag bits (specified by an exogenous parameter) and all rules are treated as memes. This means that they can be communicated and mutated. For each meme held the agent maintains a "confidence value" [0..1] which indicates how "psychologically attached" the agent is to the meme. Cultural interactions and periodic satisfaction tests affect confidence values. A proportion of the tag an agent holds may be fixed. The fixed bits never change. The proportion of fixed bits is specified by an exogenously defined parameter (BF). Figure 1 shows a schematic diagram of an agent. In the following sections the components of the agent are described.

3.3.1 Tags and Rules

In order to implement "stereotyping", agents have the ability to generalise over observable tags using their behavioural rules. A simple form of pattern matching achieves this. Agents store some fixed number of rules that map patterns of observable tags to strategy representations

The tag pattern is a string of the same length as the tag bit string but may comprise digits of zero (0), one (1) and "don't care" (#). A "don't care" digit matches both zero and one digits. This mechanism allows for generalisation. A tag pattern containing all "don't care" (#) digits, would match all possible tags. Since agents in certain circumstance may mutate the tag pattern this allows for generalisation and specialisation of stereotypes to take place. That is, rules may be widened or narrowed in their applicability. The number of rules an agent can hold is specified by an exogenously defined parameter (M). M is the same for all agents within a given society.

3.3.2 Strategies and Mutation

Strategies are represented as pairs (p,q) of real values in the range [0..1] as used in (Riolo, 1997; Nowak & Sigmund 1992). The (p) value represents the probability that the agent will cooperate given that the opponent cooperated on the last application of the rule. The (q) value represents the probability that the agent will cooperate give that the opponent defected on the last application of the rule. Therefore, for each rule an agent has an associated memory storing either C or D that indicates the move made by the opponent when the rule was last used. Initially these memories are set randomly. The (p,q) strategy representation is stochastic with a memory of one. It captures many variations of reciprocity and provocability: (1,0) represents tit-for-tat-like reciprocity (Axelrod 1980), (0,0) represents pure defection and (1,1) represents pure cooperation. Consequently, though agents actually play single round games, these are played by the agents as on-going games of the Iterated Prisoner's Dilemma (IPD) as if all agents in the category specified by the tag pattern in the rule were a single agent.

Given this arrangement it is possible for an agent to play tit-for-tat against a whole group of other agents as specified by the tag pattern associated with the strategy. This captures the notion that an agent may punish an agent within a stereotyped group for something that another agent from that same group did in the past. We should note that intuitively it appears that such a process would make cooperation very hard to achieve.

Agents start with a set of randomly generated memes (tags and rules). Any fixed tag bits are also randomly initialised. Agents can only change their memes by mutation or by accepting a meme from another agent via communication. After a satisfaction test (see below) agents examine each of their memes to determine if mutation should take place. The susceptibility of a rule to mutate is inversely proportional to the associated "confidence" value. Since the LHS of a rule (pattern label) is a bit string (perhaps including "don't care" symbols), mutation takes the form of changing with probability MT (where MT is an exogenously defined parameter) each digit from it's current value to one of the other two values with equal probability. When a specific bit value (0 or 1) is replaced by a "don't care" (#) digit then the rule is generalised. Conversely when a "#" is replaced by a "0" or "1" the rule is specialised. On the RHS of the rule, the (p,q) strategy representation, mutation takes the form of changing, with probability MT, the values of each variable by some +ve or -ve value in the range [-MS..+MS]. MS is an exogenously defined parameter. Final values of p or q which are > 1 or < 0 are reset to 1 and 0 respectively. After either mutation or communication changes a rule the confidence associated with the rule is set to a random value. Here the notion of "cultural innovation" is minimally captured. An agent will tend to mutate a rule (stereotype) if its confidence in that rule is low.

3.3.3 Cultural Interaction

Each rule and (non-fixed) tag bit held by an agent is viewed as a meme. The tag bits can be considered as "surface memes" or "social cues" visible to other agents. The rules can be considered as "hidden memes" which are not visible to others. Both are communicated (i.e. propagated from agent to agent) in the same manner. Two agents are selected for cultural interaction using the selection method described previously. Given two agents have been selected, one becomes the sender, the other the receiver (decided by a fair coin toss). Each meme held by the sender is proposed to the receiver with a probability of PM (this is an exogenous parameter, 0 indicates no meme propagation, 1 indicates all memes are proposed). The fundamental mechanisms of meme spread are those of:

- Replication: the sender replicates a meme to the receiver overwriting an existing meme.
- Reinforcement: the receiver already possesses the meme proposed by the sender and this results in an increase in confidence associated with that meme by the receiver.

- Repelling: the receiver is likely to reject an attempted replication when the associated confidence value of the meme to be overwritten is high.

In order to implement such mechanisms each agent must possess the ability to classify its memes into one of three types with respect to the proposed meme: a) Identical memes - which can be reinforced; b) Contradictory memes - which need to be removed if the new meme is accepted; c) Other memes - which are neither identical nor contradictory. The tag bits are naturally either identical or contradictory (the bits match or they do not). Rules (stereotypes) are deemed to be identical if both the pattern and the strategy match exactly and contradictory if the patterns match exactly but the strategies don't. In this latter situation the rules are considered contradictory because they would both fire for an identical set of opponents but give different strategies to apply. The process of meme propagation involves the steps shown in figure 3.

3.3.4 Game-interaction

Game-interaction involves the pairing of two agents for a game of the one-shot PD. Two agents are selected for game-interaction with relevant tag and spatial biasing mechanisms as previously described. Each agent decides whether to cooperate or defect in the following way (see figure 4):

```
                         Meme propogation:

the sender proposes a meme to the receiver
IF the receiver holds an "identical" meme THEN
        receiver increases associated confidence (reinforcement)
ELSE
        IF the receiver holds a "contradictory" meme THEN
                attempt replication - replacing the contradictory meme
        ELSE
                receiver selects a held meme at random
                attempt replication - replacing the chosen meme
        ENDIF
ENDIF

                        Attempt replication:

The receiver draws a random number [0..1]
IF draw > confidence associated with the meme to replace THEN
        receiver overwrites meme with the senders meme (replication)
ELSE
        receiver keeps its meme intact (repel)
ENDIF
```

Figure 3. The steps involved in the propagation of a meme from one agent to another. During cultural interaction between two agents the sender propagates each of its memes with PM probability.

- Each agent reads the other's tag string.
- Using this tag each agent searches its set of rules.
- Each rule with a LHS tag pattern that matches the tag is marked as "active".
- Each "active" rule is assigned a score based on the number of actual bits (1 or 0) that match (specific rules are therefore scored higher than general rules).
- The rule with the highest score is "fired" and the appropriate action performed as dictated by both the strategy represented on the RHS of the rule and the associated memory.
- If more than one rule has the same highest score (i.e. there is a tie) then the rule with the highest confidence is used. If more than one rule has the same highest confidence then a random selection is made between them. There will always be at least one "active" rule since each agent is forced to maintain a default rule - that being, all "don't care" states on the LHS.

3.3.5 Satisfaction Tests and Confidence Values

Confidence values are changed during cultural interaction and periodically through the application of an all-or-nothing satisfaction test. If an agent is satisfied then all of its confidence values are increased by some factor, otherwise all values are reduced by some factor. An agent is said to be "satisfied" if its average payoff from game-interactions is above some threshold (T) since the last satisfaction test. An agent performs a satisfaction test with some probability (P) after each game-interaction. Both T and P are exogenous parameters. Such a scheme implements a crude form of reinforcement learning: if an agent is satisfied it increases the confidence of all memes (by a factor of CI) otherwise confidence is reduced (by a factor of CR). Both CI and CR are exogenously defined parameters. Since the outcome of each game-interaction results in an instant payoff it would not be difficult to accumulate payoffs against the rules that generated them. In this way, confidence values could be differentially updated. However, it is one of the assumptions of the StereoLab society that agents are highly bounded in their reasoning and that they don't know which individual memes are responsible for satisfactory outcomes (Simon 1990).

3.3.6 Tag and Spatial Biasing

Both spatial and tag biasing may be employed during the selection of partners for both game and cultural interaction types. Tag biasing consists in rejecting a potential interaction partner based on the number of differing bits between two tags - tag distance. Exogenous bias parameters specify the extent of biasing for both game (BG) and cultural (BC) interaction. They indicate the maximum tag distance allowable before an interaction rejection is triggered. The total number of rejections allowed in succession by a single agent before interaction is forced is also specified as exogenously defined parameters (TG, TC).

Agents also limit their interactions to a subset of the population who are spatially close (within their "interaction window"). The justification for this is that cultural and economic interactions are often localised spatially within real societies. Agents inhabit a one dimensional space (see figure 2). Each end of the line is joined to form a ring topology. Along the line are a finite number of locations or "territories". The number of territories is specified by an exogenous parameter (S). Each territory can hold any number of agents. Agents are distributed along the line initially at random from a uniform distribution. Both game (VG) and cultural interaction (VC) are mediated by independent "interaction window" size parameters. The largest interaction window specifies that agents in all territories are reachable from any other, the smallest indicates that only agents within the same territory can interact. This spatial arrangement allows for different cultural and game mixing methods to be implemented. From pure random mixing (when VG and VC are at a maximum) to highly restricted or niche mixing (when VG and VC are at a minimum). This parameterisation allows for a large set of different localisation types to be explored minimally in one dimension.

Both game and cultural interaction is dyadic. Each kind of interaction is implemented separately: the same pair of agents do not culturally and game-interact at the same time. Selection of a pair of agents for either kind of interaction follows the same pattern. Firstly an agent is selected from the population at random, then an interaction partner is selected at random from within the appropriate interaction window (implementing the spatial bias). Then tag bits are compared and the interaction partner is rejected if the tag bias constraint is not met. If interaction was rejected another interaction partner is selected. This re-selection is continued until an appropriate interaction partner is found, or until the maximum number of rejections is reached after which interaction is forced with the next randomly chosen partner.

3.3.7 Rule Consistency & Redundancy

A cultural interaction event is defined such that it cannot result in either contradiction or redundancy within an agent rule set. This does not mean that more than one rule from the rule set of an agent cannot match the tag pattern of a single agent. This is resolved via specificity, then confidence, then ultimately a random choice. "Contradictory" and "identical" rules are not allowed to coexist within a single agent rule set. The LHS of each rule must be unique. If a mutation event causes two LHS' to become identical, it is reversed.

3.4 The Time Unit Cycle

In a given time unit the following events occur: with probability FG two agents game-interact; with probability FC two agents culturally interact; with probability FM one agent moves spatially. Movement involves a randomly selected agent moving to a randomly selected location. A single cycle of the system is defined as the number of time units required until 10N game-interactions have occurred, where N is the number of agents in the society (an exogenously defined parameter). FG, FC, FM and N are exogenously defined parameters.

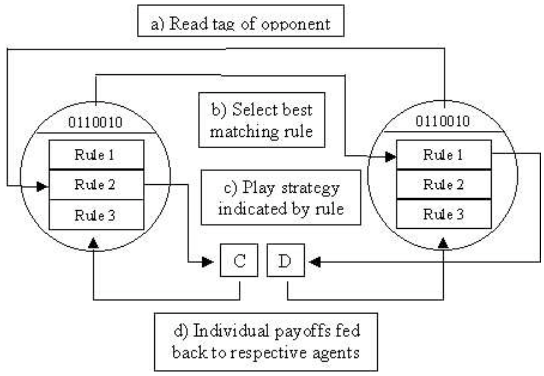


Figure 4. Game-interaction in the StereoLab. Game-interactions take place between selected pairs of agents. Each agent reads the others tag bits and fires a matching rule producing a particular game move (C or D).

3.5 Summary of the Parameters

A summary of the exogenous parameters used by the StereoLab is given in table 1 below. The range column indicates the range from which values can be selected. Parameters with a single value in the range column indicate that they were fixed at the stated value for the work presented here. The satisfaction threshold T, the probability of a satisfaction test P and the Prisoner's Dilemma payoffs PT, PR, PP and PS are all fixed. The fixing of the PD payoffs ensures they follow the PD constraints

Note that the satisfaction threshold T and the probability of satisfaction testing P are fixed such that a game-interaction only produces satisfaction if an agent receives a temptation (PT) or reward (PR) payoff. Several of the parameters were fixed or limited in range for practical reasons. For example, N and S are fixed since large numbers of agents would significantly increase the time taken to execute a simulation run and very sparse distributions of agents in the environment (which would result from large values of S or small values of N) would limit the application of spatial biasing. Large values for B (the number of tag bits) also significantly increases execution time and small values (below two) would not allow for the proper functioning of the tag processes previously described. The minimum value for FG is set to 0.1 rather than zero since some level of game-interaction is required in order to obtain meaningful results.

	Description	Rng
B	Number of bits in tag string	4..8
M	No. stereotypes agent stores (mem. size)	2..10
S	Number of locations in environment	101
N	Number of agents in the society	101
T	Satisfaction threshold	3
PM	Probability of meme propagation	0..1
P	Probability of satisfaction test	1
MT	Mutation rate	0..1
CI	Factor by which to increase confidence	0..1
CR	Factor by which to decrease confidence	0..1
MS	Mutation size for strategy parts	0..1
FG	Prob. of game-interaction in a time unit	0..1
FC	Prob. of cultural interaction in time unit	0..1
FM	Prob. of rand. agent movement in time unit	0..1
BF	Proportion of tag bits that are fixed	0..1
BG	Req. prop of tag shared for game-interac.	0..1
BC	Req. prop. of tag shared for cult.-Interac..	0..1
TG	No. refusals before forced game-interac.	1..10
TC	No. refusals before forced cultural-interac.	1..10
VC	Size of cultural interaction window	0..1
VG	Size of game-interaction window	0..1
PP	The P payoff from the PD matrix	1
PT	The T payoff from the PD matrix	5
PR	The R payoff from the PD matrix	3
PS	The S payoff from the PD matrix	0
PP	The P payoff from the PD matrix	1
PT	The T payoff from the PD matrix	5
PR	The R payoff from the PD matrix	3
PS	The S payoff from the PD matrix	0

Table 1. The parameters that characterise the StereoLab artificial society

3.6 What Kind of Society Has Been Proposed?

A moment's reflection on the fixed parameters and the nature of the agents indicates that since the agents are satisficers rather than optimisers, and since the satisfaction threshold T = PR (the reward payoff from a PD interaction), the dilemma of the PD is partially resolved. That is, if all agents choose to cooperate then all will be satisfied. The assumption expressed here is that for all StereoLab societies a state of total satisfaction through complete cooperation is possible. To put this in a more anthropomorphic way: each agent is happy to sustain a convention of cooperation if all other game-interaction partners encountered also cooperate. This assumption intuitively makes cooperation appear more likely. However, this can be contrasted with the previous assumption that agents may retaliate against others that are categorised within the same stereotype as a previous agent that was not cooperative - they make a generalisation. This generalisation means that agents subjectively stereotyped as members of the same group are treated as if they were a single individual. Taking both of these aspects into account the StereoLab consists of agents who (quite reasonably) are prepared to cooperate if all others do so but (perhaps less reasonably) may retaliate against any stereotyped group member when some member of that group does not cooperate. Such agent behaviour is very reasonable if the stereotyped groups are viewed as single agents rather than some subjectively categorised grouping.

4 Searching for Cooperation

We wish to find regions in the StereoLab parameter space where cooperation is high. The parameter space was quantised into discrete increments. For integer parameters the increment was set to 1. For parameters in the range [0..1] the increment was set to 0.1. This produces a discrete space, or grid, containing 5 x 9 x 10^2 x 11^{13} = 1.55 x 10^{17} possible unique points. In order to locate regions of high cooperation two methods were used. Firstly, decision tree induction (Quinlan, 1993) was used over a large random sample of points taken from the whole space (section 4.1). Secondly, k-means cluster analysis (Spath, 1980) was used with points found via hill-climbing in the space (see section 4.2).

4.1 Random Sampling and Tree Induction

The C4.5 classification algorithm (Qunlan, 1993) induces decision trees from a sample over a space of parameters (attributes) for a given categorical variable (in this case some category of observable phenomena of the simulation runs). The algorithm works by recursively splitting the parameter space along the dimension which produces the highest "information gain" (based on information theory) over the sample. Figure 6a shows schematically the application of C4.5 to a random sample producing class homogenous regions.

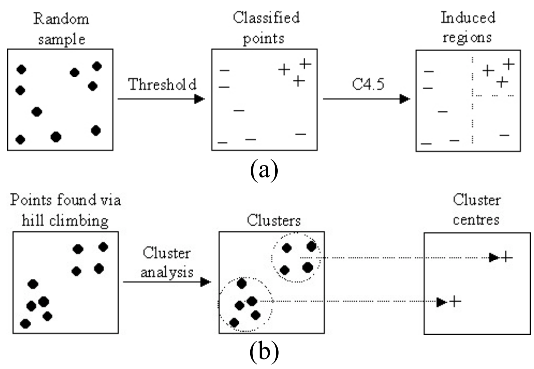


Figure 6. (a) Points sampled randomly from the parameter space can be processed to induce a set of category homogenous regions. (b) Cluster analysis applied to a sample of points found via hill-climbing may be used to find clusters.

The C4.5 decision tree induction algorithm was used to induce regions in the space as follows: Firstly the parameter space was randomly sampled (approximately 10,000 points) in order to gain an empirical measure for the prevalence of cooperation within the space. Each point in the space represents a simulation run terminated after 100 cycles. One cycle is equivalent to 10N game-interactions (where N is the number of agents). The number of agents is fixed at 101[2] for all experiments detailed here. Consequently a single simulation run is terminated after 101,000 game-interactions. This means the sample is a synthesis of approximately 1.01×10^9 individual game-interactions.

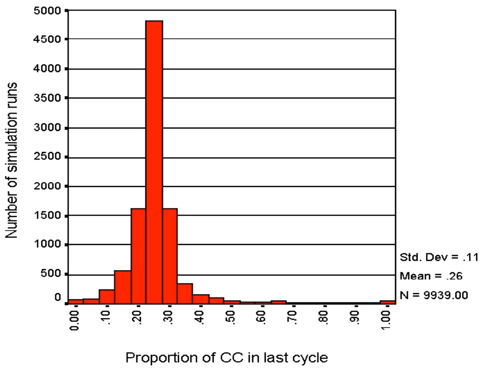


Figure 7. Frequency of cooperation over the whole parameter space. A random sample of approximately 10^3 points

Game-interaction between agents involves pairs of agents playing single-round games of the PD. The amount of cooperation (CC) for a run is calculated as the proportion of mutually cooperative (i.e. when both agents cooperate) game-interactions over the last cycle. If agents selected strategies randomly, 25% of game-interactions would be mutually cooperative. Figure 7 shows a frequency distribution histogram of the CC measures for the random sample. As can be seen the majority of runs produced low levels of cooperation between agents (not much more than would be achieved if games were selected at random). However, the distribution shows some small number of runs producing high amounts of cooperation. For the purposes of analysis the top 10% of runs (based on level of cooperation) where classified as "high cooperation", the rest as "low cooperation". The C4.5 algorithm was applied to all the points from the sample and several regions werinduced[3]. The two "best" regions (based on the number of high cooperation points contained within them) are given below:

$$MT > 0,\ CR > 0,\ VG = 0,\ FM <= 0.1$$

This region contained 150 points of which 80% were "high cooperation". The parameter ranges indicate that:

- Meme mutation is non-zero
- Agents reduce confidence in their memes if they are not satisfied
- Game-interaction is limited to a single territory
- The frequency of agent movement between territories is low.

Inspection of individual runs indicates that agents have only a small set of game-interaction partners. This makes the search space for coordinated game-interactions small and so it is more likely that agents will find a cooperative convention. Agents are repeatedly meeting the same small number of agents and hence "learn" to find a mutually satisfactory set of behaviours.

$$MT>0,\ CR>0,\ VG >0,\ PM >0.4,\ FG<=0.1,\ FC>0.1$$

This region contained 284 points of which 44% were "high cooperation". The parameter ranges indicate that:

- Meme mutation is non-zero
- Agents reduce confidence in their memes if they are not satisfied
- Game-interaction is NOT limited to a single territory
- Cultural interaction events are, at least, one order of magnitude more frequent than game-interaction events.

A high frequency of cultural interaction between games gives the agents more scope to adapt and hence coordinate their game-interactions. Agents are exchanging many memes between game-interactions and therefore "learn" to find a mutually satisfactory set of behaviours.

4.2 Hill-Climbing and Cluster Analysis

In order to apply k-means cluster analysis, a set of points were located in the space[4] that produced the maximum possible cooperation over the final (100th) cycle of the simulation run (see Figure 6b). Maximum cooperation indicates that all games played in the final cycle produced mutual cooperation. 39 such points were found by hill-climbing from 100 randomly chosen locations for 100 steps. This means that 10,000 individual runs were executed (an identical computational effort to that used for the random sample above). The points were normalised into unit space and clustered into 5 clusters using k-means clustering (where the objec-

---

[2] The value of 101 agents was selected to be equal to the number of territories in the environment. The number of territories was set to an odd number so that an agent within a given territory had a balanced number of neighbouring territories in either direction around the ring into which game and cultural interaction windows may extend.

[3] A "weight" of 100 was used with the C4.5 algorithm. This means that the C4.5 was constrained to induce only regions with a minimum of 100 points within them.

[4] The space used was increased over that used with C4.5. The real valued parameters were quantised with the finer increment of 0.01 (instead of 0.1) and the range of TG and TC was [1..200].

tive function was Euclidean distance from cluster centriods). Various numbers of clusters were tried but beyond 5 the objective function did not decrease significantly.

As before, cooperation is high when game-interaction is limited to single territory (a single cluster was found to identify this region in the space). Cooperation was also high when BG and TG were high and BF was low (three distinct clusters had these values). In these clusters the biasing of game-interaction towards those sharing similar tag bits is high and the low value for BF indicates that the majority of tag bits are culturally learned. However, what process could produce high cooperation from such biasing?

5. Conclusion

Based on observation of individual runs from the clusters identified above the following is a hypothesis as to why tag biasing produces high cooperation:

- Tags combined with biasing create "game-interaction groups" sharing the same tags
- Cultural learning can change tags
- Hence agents "move" between groups
- Unsatisfied agents change tags, hence groups
- Groups satisfying their members (via cooperation) tend to stabilise and recruit
- Groups that do not satisfy tend to dissipate
- Hence cooperation is promoted

It is argued that this process may be viewed as a novel from of "cultural group selection", where the "groups" are not extended in physical space but in "tag space", this result bears some comparison with more abstract models of a biological rather than cultural nature (Hales, 2000; Riolo, 1997).

Two distinct forms of game-interaction localisation (or "parochialism") promote cooperation (among other mechanisms). These take the form of spatial parochialism - agents only playing games with others sharing the same territory and cultural parochialism - agents only playing games with others sharing the same or very similar tags. The C4.5 algorithm combined with random sampling identified the spatial mechanism as did k-means cluster analysis combined with hill-climbing. The cultural mechanism was only found by the clustering method but this was applied to an extended space (so this is not a true comparison of techniques).

References

[1] Armstrong, A. and Durfee, E. (1998). Mixing and Memory: Emergent Computation in an Information Marketplace. In The Proceedings of the 3rd International Conference in Multi-Agent Systems (ICMAS'98). IEEE Computer Society Press, Los Alamitos, CA.

[2] Axelrod, R. (1980). The Evolution of Cooperation. Basic Books, New York.

[3] Axelrod, R. (1995). The Convergence and Stability of Cultures: Local Convergence and Global Polarization. SFI Working Paper 95-03-028. Santa Fe Institute, Santa Fe, N.M.

[4] Cohen, M., Riolo, L., Axelrod, R. (1999). The Emergence of Social Organization in the Prisoner's Dilemma: How Context-Preservation and other Factors Promote Cooperation. SFI Working Paper 99-01-002. Santa Fe Institute, Santa Fe, N.M. Available at <http://www.santafe.edu/sfi/publications/>

[5] Epstein, J. and Axtell, R. (1996). Growing Artificial Societies: Social Science from the Bottom Up. The MIT Press, Cambridge MA.

[6] Hales, D. (2000) Cooperation without Space or Memory: Tags, Groups and the Prisoner's Dilemma. In Moss, S., Davidsson, P. (Eds.) Multi-Agent-Based Simulation. Lecture Notes in Artificial Intelligence 1979. Berlin: Springer-Verlag.

[7] Holland, J. (1992). Adaptation in Natural and Artificial Systems. The MIT Press, Cambridge MA.

[8] Holland, J. (1993). The Effect of Labels (Tags) on Social Interactions. SFI Working Paper 93-10-064. Santa Fe Institute, Santa Fe, NM.

[9] Holland, J. (1998). Emergence - From Chaos to Order. Perseus Books, Cambridge, MA.

[10] Hostetler, J. (1963). Amish Society. Johns Hopkins Press, Baltimore.

[11] Kramer, R. and Brewer, M. (1984). Effects of Group Identity on Resource Use in a Simulated Commons Dilemma. Journal of Personality and Social Psychology, 46(5):1044-1057.

[12] Leyens, J. et al. (1994). Stereotypes and Social Cognition. Sage, London.

[13] Macy, M. and Skvoretz, J. (1998). Trust and Cooperation Between Strangers. American Sociological Review, 63:638-660.

[14] Nowak, M. and Sigmund, K. (1992). Tit for tat in heterogeneous populations. Nature, 355:250-253.

[15] Oakes, P. et al. (1994). Stereotyping and Social Reality. Blackwell, Oxford.

[16] Quinlan, J. R. (1993). C4.5: Programs For Machine Learning. Morgan Kaufmann, San Mateo, California.

[17] Riolo, R. (1997). The Effects of Tag-Mediated Selection of Partners in Evolving Populations Playing the Iterated Prisoner's Dilemma. SFI Working Paper 97-02-016. Santa Fe Institute, Santa Fe, NM. Available at: <http://www.santafe.edu/sfi/publications/>

[18] Simon, H. A. (1990). A Mechanism for Social Selection and Successful Altruism. Science, 250:1665-1668.

[19] Spath, H, (1980). Cluster Analysis Algorithms for Data Reduction and Classification of Objects. Ellis Horwood, Chichester.

[20] Tajfel et al. (1971). Social Categorization and Intergroup behaviour. European Journal of Social Psychology, 1:149-177.

Engineering with Social Metaphors

in the

Socially Inspired Computing Joint Symposium

DELIS BISON AGENTLINK

Theme Preface

Engineering with Social Metaphors

INTRODUCTION

It appears that the requirement to engineer ever more decentralised, self-organising and large computer systems is pushing engineers to start asking what appear to be sociological kinds of questions e.g: How can trust be established and maintained between autonomous sub-units? How can systems maintain their functions when old sub-units leave and new sub-units arrive? How can functions emerge without a priori planning or centralised control?

These kinds of questions are no longer of purely theoretical interest - they are issues that engineers need to addressed now. Often, for want of any alternatives, `old-style' solutions are applied in which various degrees of centralised planning and control are applied producing brittle and poorly scaling systems. There is a technological bottleneck here that needs to be addressed. Ideas from biology have already been successfully applied to some such problems. This theme day of the 'Socially Inspired Computing' Joint Symposium aims to focus on work which contributes to doing the same with ideas and metaphors originating in social phenomena.

Social systems are complex self-organising and self-regulating systems that emerge certain kinds of properties that would appear to be very useful if they could be instantiated in computer systems. For example, the emergence and maintenance of roles, institutions, power-relations, exchange and trust systems are very much. Historically, these kinds of issues have been studied by the social sciences based on theoretical speculation or on observation of existing human societies. More recently, the emerging discipline of computational social science has begun to formalise concepts about social mechanisms algorithmically - i.e. using (often agent-based) simulation. It would appear that there is a great potential for cross fertilisation between researchers trying to solve difficult engineering problems and those producing computational models of complex social phenomena. We hope to encourage this process by the exchange of relevant ideas, techniques and problems.

The eight presentations comprising the day include two invited talks from researchers (Márk Jalasity and Giovanna Di Marzo) radically new self-organising and emergent methods to systems engineering. Additionally, we have contributions presenting socially inspired and potentially useful techniques and work employing `participatory modelling techniques' in which humans form part of the `loop'.

PROGRAMME COMMITTEE (we thank)

* Rafael Bordini (Durham, UK)
* Francois Bousquet (CIRAD, France)
* Joanna Bryson (Bath, UK)
* Edmund Chattoe (Oxford, UK)
* Rosaria Conte (CNRS, Rome, Italy)
* Owen Densmore (Backspaces.net, NM, USA)
* Alexis Drogoul (Lip6, France)
* Guido Fioretti (Bologna, Italy)
* Nigel Gilbert (Surrey, UK)
* Pierre Glize (IRIT, France)
* Márk Jelasity (Bologna, Italy)
* Catholijn Jonker (UVA, Netherlands)
* Giovanna Di Marzo (Geneva, Switzerland)
* Edoardo Mollena (Bologna, Italy)
* Alberto Montresor (Bologna, Italy)
* Mario Paolucci (CNRS, Rome, Italy)
* Paolo Petta (Ã-FAI, Austria)
* Scott Moss, (MMU, UK)
* Juliette Rouchier (CNRS, France)
* Mihaela Ulieru (Calgary, Canada)
* Adam Wierzbicki (Polish-Japanese Inst. of I.T., Poland)
* Franco Zambonelli (Modena, Italy)

SPONSORING PROJECTS AND ORGANISATIONS (we acknowledge support from):

AgentLink III: Co-ordination Action for Agent Based Computing (IST-funded). http://agentlink.org
DELIS: Dynamically Evolving Large Scale Information Systems (IST-001907). http://delis.upb.org
BISON: Biologically-Inspired techniques for Self-Organisation in dynamic Networks (IST-2001-38923). http://www.cs.unibo.it/bison/

ORGANISATION

David Hales (University of Bologna, Italy)
Bruce Edmonds (Manchester Metropolitan University, UK)

Engineering Emergence through Gossip

Márk Jelasity[⋆]
[⋆]University of Bologna
Mura Anteo Zamboni 7, 40127 Bologna, Italy
jelasity@cs.unibo.it
and RGAI, MTA-SZTE Szeged, Hungary

Abstract

Gossip is one of the most usual social activities. The result of gossip is that new and interesting information spreads over a social network not unlike diseases during an epidemic, or computer worms over the Internet. We will argue here that the core "idea" of gossip, that is, periodic information exchange among members of a group over a network that connects them, and a subsequent update of the knowledge of the group members based on the information they exchange, is a powerful abstraction that can be applied for solving a wide range of problems in distributed computing. The applications include—apart from the most natural one: information dissemination—gathering global knowledge about distributed systems and organizing the group members into several structures, such as ordering, clustering or other arbitrary topologies.

1 Introduction

Gossip is one of the most usual social activities. The result of gossip is that new and interesting information spreads over a social network very efficiently, not unlike diseases during an epidemic, or computer worms over the Internet.

The characteristics of information spreading through gossip are quite remarkable. Considering that participants only talk to their acquaintances and relatives, and they make strictly local and private decisions about what to gossip, and how to interpret the received information, it is quite impressive how efficient the process is. This fact has not been left unnoticed in the distributed algorithms community: in fact, the application of gossip to spread information over various distributed systems is commonplace, see e.g. Eugster et al. (2004).

However, the basic "protocol" underlying gossiping holds a much more general potential than merely information spreading. If we distill the basic components, we can realize that we have a complex social network that connects people and the "algorithm" which is run by all people is essentially periodic communication with some neighbors in this network. During such a communication, a person selects information to be shared with the neighbor, and receives information from the neighbor. After the reception of information, everyone updates their knowledge.

This scheme can be easily translated into the language of distributed systems, where the participants are processes or network nodes, and the social network becomes a physical or virtual (overlay) computer network. The skeleton of the gossip scheme is shown in Figure 1. Note that this scheme is rather similar to a cellular automaton, only more general in that the connection topology can be arbitrary, and it can even change over time. Furthermore, the nodes can execute rather complex algorithms to update their states. The components of the scheme are the following:

do once in each T time units at a random time
  p = selectPeer()
  send state to p
  receive state$_p$ from p
  state = update(state$_p$)

(a) active thread

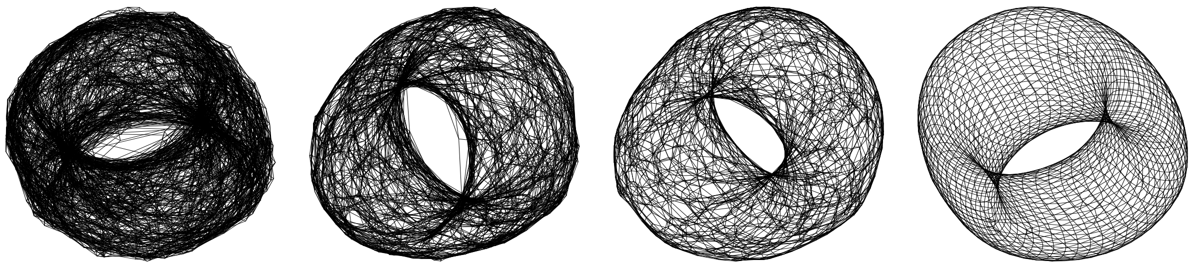
do forever
  receive state$_p$ from p
  send state to p
  state = update(state$_p$)

(b) passive thread

Figure 1: The generic protocol scheme run on each network node.

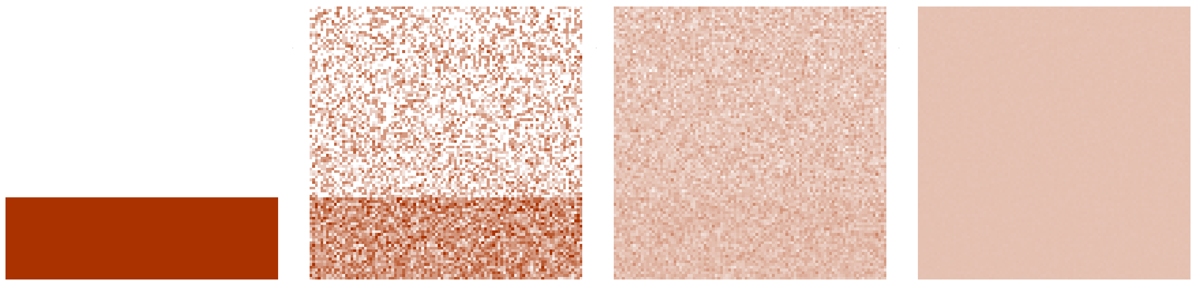
state is defined by the application domain (for example, a number, a set of documents, a set of neighbors, known information items, etc)

selectPeer() defines the way the peer is selected



after 3 cycles after 5 cycles after 8 cycles after 15 cycles

Figure 2: T-Man is run starting with a random network. A torus is evolved within a few cycles. The example shown is a 1000 node graph, but experiments show that convergence time is logarithmic in network size. A cycle is T/2 time units, that is, each node communicates once on average during a cycle.



initial state after 1 cycle after 3 cycles after 8 cycles

Figure 3: A network of 10 000 nodes is shown, each pixel representing a node in a 100x100 bitmap. The intensity of each pixel represents the numeric value held by a node. The underlying random overlay network is not shown. Convergence speed is similar irrespective of initial configuration and network size. A cycle is T/2 time units, that is, each node communicates once on average during a cycle.

(random, biased towards geographic proximity or high bandwidth, etc)

update() is the key function: the local rule that results in global behavior. Analogous to the update rule of cellular automata, but more general operating on arbitrary structures (states)

2 Examples

In this section we briefly outline two examples to illustrate the generality of the gossiping scheme.

2.1 Construction of Structures

Over a set of nodes connected to the Internet, one can define a so called overlay topology based on a "who-knows-whom" relation. That is, although any node can potentially communicate with any other node, to actually communicate they have to know the address of the peer node. The set of addresses known by each node define a virtual, or overlay, network.

Overlay networks have recently received increasing attention, because they are very useful in supporting distributed protocols. Applications include routing information, and clustering and sorting the nodes according to some attributes to facilitate search.

The gossip scheme is useful also to evolve such overlay topologies in a completely decentralized way, very quickly. All we need to assume is that the nodes are able to rank any set of other nodes according to preference of selecting them as neighbors. The components are implemented as follows:

state is a set of peer addresses: the partial view. The views of the nodes define the overlay topology.

selectPeer() is biased towards nodes that are "closer" according to the actual target topology to be evolved, using the preference of the nodes.

update(a,b) generates a new partial view from the two partial views a and b. It keeps those addresses from the union of a and b that are "closest" in the target topology, again, based on the preference ranking.

Figure 2 illustrates the protocol when it is used to construct a torus. The protocol has been studied by Jelasity and Babaoglu (2004), where it was shown that it is rather independent of the characteristics of the topology that we would like to generate. The cases of the ring, torus and a binary tree were shown to converge at virtually the same, logarithmic speed in the network size.

2.2 Data Aggregation

The problem of data aggregation is to provide all the nodes with global information about the distributed system in which they participate. Examples include the average or maximal value of some attribute, such as storage capacity, available bandwidth, or temperature (in sensor networks), the size of the system (number of nodes), or the variance of some attribute. Aggregation plays an important part in monitoring and control applications.

The gossip scheme offers a possibility to implement a simple but very robust and efficient averaging scheme that follows a diffusion-like dynamics. The components of the scheme have to be implemented the following way:

state is a number, representing any attribute, like temperature, free storage, available bandwidth, etc.

selectPeer() is random from the entire system, assuming an underlying random network. There are protocols that can provide this random network, such as NEWSCAST that is based on the gossiping scheme itself, see Jelasity et al. (2004).

update(a,b) defines the aggregate function to be calculated. Some examples are maximum (or minimum), where update(a, b) =max(a, b) (or min(a, b)), or any mean of the form $f^{-1}((f(x_1)+...+f(x_n))/n)$ that covers among others average ($f(x) = x$), quadratic ($f(x) = x^2$), harmonic ($f(x) = 1/x$) and geometric ($f(x) = \ln x$) means. In this case update(a,b)= $f^{-1}((f(a) + f(b))/2)$.

Figure 3 illustrates the speed at which all the nodes converge to the average value. It has been shown that the protocol is very fast and extremely robust, see Jelasity et al.; Jelasity and Montresor (2004); Montresor et al. (2004).

3 Conclusions

It has been shown that the basic scheme underlying gossiping can be efficiently used to implement very different fully distributed functions, in a controllable, robust, simple and relatively well understood way. This means that this scheme represents a way of engineering emergent properties of systems such as structure of the connectivity network, and calculation of global information.

In fact this approach can be even incorporated into a component architecture, in which a large set of services are provided by overlay networks participating in a gossip protocol, see Babaoglu et al. (2004). In this framework, overlay networks (random and structured, as shown above) are constructed and maintained, which in turn support other higher level functions such as load balancing, information dissemination, search, and aggregation.

Finally, we note that since these ideas seem to be useful and powerful in computer science engineering, the reverse question becomes also interesting: isn't it possible that real gossip also works in much richer ways than usually assumed? For example, gossip itself can change the social network it uses for spreading, and other ways of feedback are possible, like learning about certain pieces of information might change our preferences and our gossip behavior, the set of people we talk to, which in turn changes our sources of information, and so on. This dynamics might be responsible for complex emergent social phenomena.

References

Ozalp Babaoglu, Márk Jelasity, and Alberto Montresor. Grassroots approach to self-management in large-scale distributed systems. In *Pre-Proceedings of the Workshop on Unconventional Programming Paradigms (UPP'04)*, pages 165–172, Le Mont Saint-Michel, France, September 2004. invited paper, Springer LNCS volume pending.

Patrick Th. Eugster, Rachid Guerraoui, Anne-Marie Kermarrec, and Laurent Massoulié. Epidemic information dissemination in distributed systems. *IEEE Computer*, 37(5):60–67, May 2004.

Márk Jelasity and Ozalp Babaoglu. T-Man: Fast gossip-based construction of large-scale overlay topologies. Technical Report UBLCS-2004-7, University of Bologna, Department of Computer Science, Bologna, Italy, May

2004. `http://www.cs.unibo.it/pub/TR/UBLCS/2004/2004-07.pdf`.

Márk Jelasity, Rachid Guerraoui, Anne-Marie Kermarrec, and Maarten van Steen. The peer sampling service: Experimental evaluation of unstructured gossip-based implementations. In Hans-Arno Jacobsen, editor, *Middleware 2004*, volume 3231 of *Lecture Notes in Computer Science*, pages 79–98. Springer-Verlag, 2004. ISBN 3-540-23428-4.

Márk Jelasity and Alberto Montresor. Epidemic-style proactive aggregation in large overlay networks. In *Proceedings of The 24th International Conference on Distributed Computing Systems (ICDCS 2004)*, pages 102–109, Tokyo, Japan, 2004. IEEE Computer Society.

Márk Jelasity, Alberto Montresor, and Ozalp Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Transactions on Computer Systems*. to appear.

Alberto Montresor, Márk Jelasity, and Ozalp Babaoglu. Robust aggregation protocols for large-scale overlay networks. In *Proceedings of The 2004 International Conference on Dependable Systems and Networks (DSN)*, pages 19–28, Florence, Italy, 2004. IEEE Computer Society.

Using Localised 'Gossip' to Structure Distributed Learning

Bruce Edmonds
Centre for Policy Modelling,
Manchester Metropolitan University
http://bruce.edmonds.name
bruce@edmonds.name

Abstract

The idea of a "memetic" spread of solutions through a human culture in parallel to their development is applied as a distributed approach to learning. Local parts of a problem are associated with a set of overlapping localities in a space and solutions are then evolved in those localities. Good solutions are not only crossed with others to search for better solutions but also they propagate across the areas of the problem space where they are relatively successful. Thus the whole population co-evolves solutions with the domains in which they are found to work. This approach is compared to the equivalent global evolutionary computation approach with respect to predicting the occurrence of heart disease in the Cleveland data set. It outperforms a global approach, but the space of attributes within which this evolutionary process occurs can greatly effect the efficiency of the technique.

1. Introduction

The idea here is to apply the idea of "gossip", that is locally distributed messages, to facilitate an evolutionary algorithm. In this approach it is the whole population of 'solutions' that learns how to solve a problem – the population is not just a vehicle for evolving the 'best' global solution. Thus, although the proposed approach can be interpreted as the adaptive propagation of solutions (or "memes") within a population spread across different local conditions, it has an application as a truly distributed evolutionary learning algorithm.

2. The Idea and the Technique

The idea of this technique is that there is a space in which the potential solutions or memes are distributed. Each "location" or "region" of the space is associated with a different part of a problem domain. At each location in the space there is a local competition and evolution of these memes or solutions. Thus the algorithm as a whole attempts to learn what solutions work best for the part of the problem it is associated with. Solutions that are locally successful propagate to neighbouring (overlapping) locations where it has to compete with the other solutions there. If there is an accessible global solution it will eventually propagate to all locations, whilst solutions which have a more limited scope will only successfully propagate to those problem areas where they work well. At the end of the process, it may well be that there is no single solution that globally dominates, but different solutions may be found to work better in different parts of the problem domain. If a global solution is required then this can be constructed by analysing the whole population that develops. That is, by finding the best solution in each location and forming a composite solution from these.

This is analogous to how human societies have developed different ways of exploiting the environment in the different geographical niches in which it has spread (Reader 1990). This variety of methods does not stop regions learning from their neighbours where this is found to be useful. Thus some techniques (such as the use of fire) have spread to all parts of the globe, whilst others (such as hunting with harpoons) are only found in particular areas.

Thus the technique, at its most basic, consists of two phases: a development stage followed by an analysis phase. In the development phase there must be a population of solutions spread across different locations forming a series of small overlapping localities, such that each locality can be associated with a different sub-problem (or sub-domain of a problem). Repeatedly, in each locality, the solutions are evaluated on the associated sub-problem or sub-domain and solutions selected and replicated in that locality. The localities must overlap so that solutions that are successful in one locality can spread through neighbouring localities, and potentially the whole space.

The analysis phase takes the resulting population of solutions and analyses it in order to extract useful information. This might involve identifying the best solution in each locality and combining them together to form a complex composite solution.

This technique, as with all techniques, has advantages, and disadvantages – this can be seen as a consequence of the "No Free Lunch" theorems (Wolpert and Macready 1997). On the plus side it: uses to the maximum extent the information about the problem encoded in the whole population of solutions and not just that in the single best solution; the technique is only evaluated locally which is computationally efficient; complex compound (total) solutions can be evolved with relatively small genes, and it is eminently suitable for massively parallel execution (each locality on a separate processor with no need for global communication). Disadvantages include the need for an analysis stage after the development phase, and that the way the chosen problem space can effect its effectiveness.

Let me illustrate the approach with a curve fitting example. One is trying to evolve the curve that best fits a given set of points. In a global approach (Figure 1), the solutions attempt to fit all the points simultaneously and hence are evaluated across the whole domain each time.

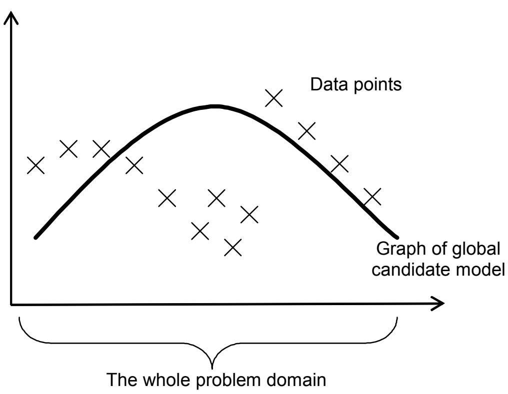


Figure 1. Graph fitting example – trying to fit some points with a single curve

In the distributed approach propounded in this paper, the domain is divided up into a number of different (overlapping) neighbourhoods and the solutions evolved and evaluated only at those localities. This is illustrated in Figure 2. A global solution one would have to construct it "piecemeal" from the best fitting curves in each locality – one could see this as a sort of evolutionary version of local regression (Cleveland and Devlin 1988).

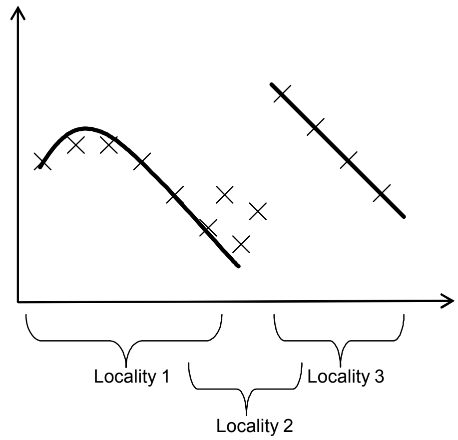


Figure 2. Graph fitting example - fitting the points with two different solutions in two localities

3. Model Setup

The working of this algorithm is illustrated in Figure 3. If you imagine that every point is a person who may be invaded by nearby memes, the one that is best for that person (in their situation) is selected (or mixed from the best). This repeatedly happens allowing the gradual spread of solutions across the space by (mostly) local propagations and crossings.

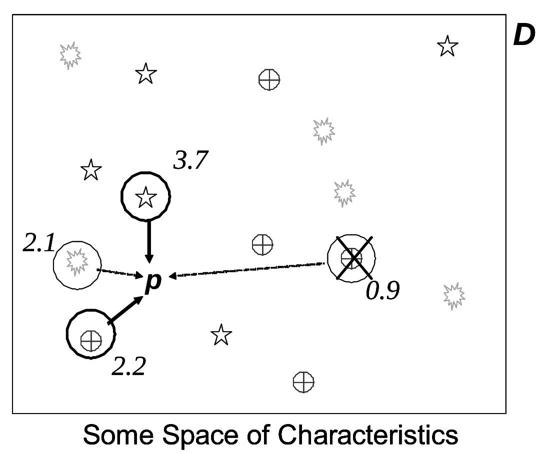


Figure 3. An Illustration of the working of the development phase. The problem space (*D*) is scattered with different solutions (the shapes); each instant: a random point in the space (*D*) is chosen (*p*); some solutions nearby are selected (circled); they are evaluated at *p* giving the finesses (numbers); the fittest are selected (bold circles) and crossed (or propagated); the result placed at the point, the worst eliminated (the cross).

An outline for the algorithm is as follows:

```
Initialise space with a random set of genes
```

```
Repeat
        For geneNum from 1 to popSize
                Randomly select a locality
                randomly select from locality
                        a set of sample genes
                evaluate set in the locality
                chose two best from set
                if randomNum < probCrossover
                then cross two best -> newInd
                else best -> newInd
        Next geneNum
        New population composed of newInds
Until finished
```

In this case the problem was predicting the outcomes of heart disease in a set of data from Patients in Cleveland. There were four possible outcomes: 0, 1, 2, 3, 4 to be predicted on the basis of 13 other attributes, all numeric or coded as numbers.

The approach was based on Genetic Programming (Koza 1992, 1994). Each gene was composed of 5 numeric expressions (one for each possible outcomes), coded as trees. Possible functions in these trees include basic arithmetic and comparison operations. The leaves include a selection of constants and the values of the 13 attributes. Evaluation is done given a set of values for the 13 "predictive" attributes by evaluating the 5 functions – the greatest value indicating which outcome is indicated. When two genes are crossed, there is a probability that each corresponding tree will be crossed.

4. The Data Set/Test Problem

The Data Set that the technique was tested upon was those concerning heart disease in Cleveland, US available at the ML repository of problems. This was chosen a random from those available. The data I used consisted of 281 examples of 14 numeric attributes, including one predicted value coded: 0, 1, 2, 3 or 4 depending on the actual outcome. The problem is to predict the outcome given the other 13 values of the characteristics. Attributes referred to in the paper are 1, 2, 4 and 5 which stand for the *age*, *sex*, resting blood pressure in mm Hg on admission to the hospital (*trestbps*), and serum cholesterol in mg/dl respectively (*chol*). Thus the spaces I tried for the space of solutions were {*age*, *sex*} and {*trestbps*, *chol*} – these selections were pretty arbitrary, simply based on a quick inspection of the values and not based in any way upon knowledge of what the important factors are. More details about the data set can be found in appendix 1.

5. Results

Three sets of runs were done. The first was a standard GP algorithm "*Global*" (12 runs); the second using the local algorithm above with the context space being defined by attributes 1 and 2 "*Local (1, 2)*" (12 runs); the second using the local algorithm above with the context space being defined by attributes 4 and 5 "*Local (4, 5)*" (12 runs). All solutions in all runs use all of the 13 attributes.

Comparing the different algorithms is not entirely straightforward. The purpose of the GP algorithm (*Global*), is to evolve the best global solution. Thus its effective error is the best solution measured by that solution's average error over the whole problem. The purpose of the algorithm proposed here is to evolve local solutions. Thus its effective error is its average error of the best local solutions when evaluated over the their local spaces. Also the local algorithm involves orders of magnitude less computational time per generation for the same population size, so comparing the effective error rate per generation would be misleading. The overwhelming overhead in this (and all) evolutionary algorithms is the time taken to evaluate each solution. To give the Global runs more of a chance each time a solution is evaluated it does so against a random sample of only 10% of the total population (though in the statistics below the error is a truly global evaluation). With respect to the effective error against the number of evaluations the performance of the Global approach was even worse when each (new) solution was evaluated against the whole problem rather than just a sample, since although the effective error achieved was slightly better, the number of evaluations this took was roughly 10 times greater. Thus I calculate each run's effective error against the number of evaluations it takes. It is this comparison which is shown in Figure 4. The dense, flat regions at the end of the Local sets is the analysis stage where the generality of discovered solutions occurs. This is included in the graph below because this stage is a necessary overhead in the local approach proposed.

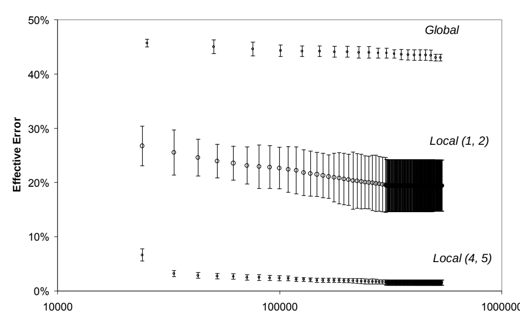


Figure 4. A comparison of effective error rates in the three sets of runs against the number of evaluations it takes (circles are averages, with the error bars indicating one standard deviation adjusted for sample size)

As you can see in Figure 4, The two *Local* runs significantly out-perform the *Global* runs. That is, for the same computational expense the average local errors of the locally best solutions in the *Local* runs are significantly less than the average global error of the single best solution in the *Global* runs. But what is also interesting in these results is the difference that the chosen problem space has on the effectiveness of the algorithm. The Local algorithm did *much* better when done using the space defined by attributes 4 and 5 than using the space defined by attributes 1 and 2.

Figure 5 and Figure 6 show the average effective error and the average spread of the *Local (1, 2)* and *Local (4, 5)* runs respectively. Here the spread is the number of localities that a solution occupies in the space. Thus an average spread of 2 would mean that there were twice as many solutions in the space as unique genes. In these figures the development and analysis phases are clearly shown. In the development phase there is a low average spread as new (unique) solutions are continually being generated, but the appearance of new solutions makes the gradual decrease in error possible. In the analysis phase there are no new solutions being generated but only local propagation of solutions, so that they 'fill out' the areas of the space that they perform best in, so the effective error rate is flat. In this phase the spread increases as the best solutions occupy more than one location.

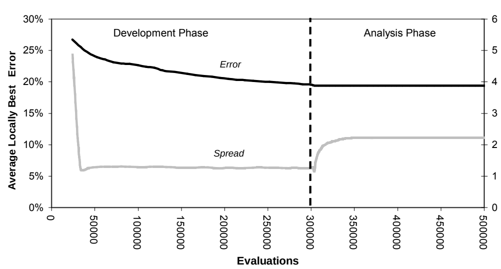


Figure 5. The average (over 12 runs) of the effective error rate and gene spread for *Local (1, 2)*

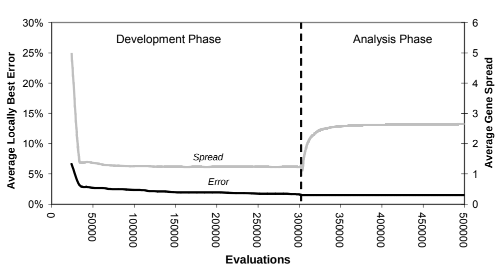


Figure 6. The average (over 12 runs) of the effective error rate and gene spread for *Local (4, 5)*

In Figure 5 and Figure 6 one can see that not only did the Local(4, 5) runs have a far lower effective error than the Local(1, 2) runs but also that they ended up with a slightly higher average spread. That means that the Local(4, 5) runs achieved (on average) a greater level of generality than the Local(1, 2) – there was no trade-off between error and generality between these two, the later was better in both respects.

Figure 7 and Figure 8 are illustrations of the sort of local spread of solutions that have occurred by the end of the analysis phase. In these only the more numerous solutions are shown so that their 'domains' are easily distinguishable.

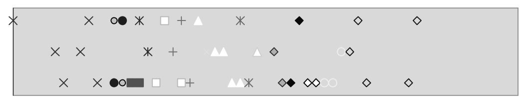


Figure 7. The attribute distribution of the more numerous best genes (those with at least 2 occurrences) in the run with the smallest effective error for *Local (1, 2)*

Figure 7 shows the positions in the problem space determined by the attributes of age (horizontal axis) and sex (vertical axis – only male top, both middle, only female bottom) of all the best solutions (in the run with the best effective error) that occurred more than once.

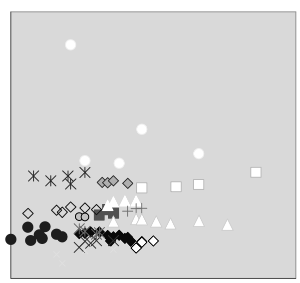


Figure 8. The attribute distribution of the more numerous best genes (those with at least 4 occurrences) in the run with the smallest effective error for *Local (4, 5)*

Figure 8 shows the positions in the problem space determined by the attributes of resting blood pressure (trestbps - horizontal axis) and serum cholesterol (chol - vertical axis) of all the best solutions (in the run with the best effective error of 1.07%) that occurred more than thrice. Here we see pronounced clustering by in both dimensions, but perhaps more by chol than trestbps. It may be the facts that: both dimensions allowing pronounced clustering; the greater number of localities that; and the greater connectivity in the space that resulted in *Local (4, 5)* being more effective than *Local (1, 2)*.

6. Discussion

Although *Local (1, 2)* did better in terms of effective error than the other runs and better than some previous ML attempts (see Appendix 1), this is not an entirely fair comparison because they are aiming at different sorts of outcomes. Clearly by simply approximating the original table of data one would obtain a zero level of error using an entry-by-entry level of locality. However, as one can see from Figure 7 and Figure 8, at least *some* level of generality above an entry-by-entry level has been achieved. There is presumably (for each problem) some sort of three-way trade-off between: the generality of the solution one obtains; the efficiency of the distributed search; and the level of effective error. Presumably by adjusting the parameters in the local approach one can obtain different levels of generality and explore this trade-off (something I have not done). This might be exploited by a gradual increase in the level of locality as the process progresses – rather like the “cooling” regime in simulated annealing.

Clearly, if the greatest “compression” encoded in a single solution and the computational cost of the algorithm is all that concerns one, then this approach is probably not the best. What this approach offers is a computationally feasible way of discovering relevant information about partial solutions *and* their domains within a solution space. It utilises the information in the *whole* population to give richer information than can be obtained from a single best global solution. Further it does this in an adaptive way which does not require the user to know in advance how the problem space should be decomposed, although it does require some knowledge of what might form a good attribute space.

7. Related Work

The algorithm was originally published as (Edmonds 2001) but applied and interpreted in a different way to that here. There it was developed as a step towards solving the problem of learning appropriate cognitive contexts arising from the analysis of the roots of context in (Edmonds 1999).

The model has a close relation to that of “demes” in evolutionary programming (Tanese 1987). There the space of solutions is split into a series of islands (where the evolution occurs), there being allowed a slow rate of migration between islands. This technique acts to preserve a greater level of variety in the total population of solutions than would be the case if they were all evolved together. However in that technique the solutions in each island are evaluated globally against the whole problem space. It is particularly closely related to diffusible cooperative co-evolutionary genetic algorithms (DCCGA) in (Wiegand 1999). In CCGA (Potter and de Jong 1994, Potter 1997) the population is divided up into subpopulations, each of which is evolved to solve a designated sub-problem of the whole. Spears (1994), identified the separate sub-populations using “tags” allowing some drift between sub-populations using a low rate of mutation in these tags. Wiegand (1999) combines these techniques so that some diffusion between populations is added to CCGA resulting in DCCGA. However, in DCCGA: the separate solutions in each population are still evaluated with respect to the whole problem (along with other solutions to other sub-problems); the sub-populations are determined in advance by the programmer; and there is no space to structure the diffusion of solutions with respect to the relation between sub-problems.

It also is related to clustering algorithms, in that it divides up the domain into those where particular solutions can dominate. However unlike those which cluster using assumptions about the characteristics of data, this approach co-evolves the solutions with the clusters, allowing the discovery of clusters with respect to discovered solutions.

This model has an obvious ecological interpretation (e.g. Wright 1932, Vose and Liepins 1991). The localities in the problem space can be seen as the various niches which the different species (the different solutions) compete to occupy. Successful species will tend to spread out over the areas in which they are competitive. After a while mutation will cause speciation among the most populous species in any set of localities and these (closely related) species will then start to compete. This process is described in (Edmonds 2001). Just as in nature, areas which are particularly difficult may have few species whilst other environments may have many fit species.

8. Conclusion

All search techniques exploit some trade-off or other. This technique trades in the generality of a single solution in return for a more efficient algorithm and information about the problem *structure*. Instead of a uniform, single solution one gets a composite solution by analysing the resulting whole population. Although the space within which problems will evolve can greatly effect the quality of the solution that results, one does not have to explicitly divide up this space into specific sub-problems, but areas that are solvable using the same local solution co-evolve with the content of the solutions.

Acknowledgements

Thanks to the participants of the International Conference on Modelling and Using Context, for discussions on these and related ideas.

References

Aha, D. and Kibler, D. Instance-based prediction of heart-disease presence with the Cleveland database. Technical Report, University of California, Irvine, Department of Information and Computer Science, Number ICS-TR-88-07, p. 8, March 1988.

Blake, C.L. & Merz, C.J. (1998). UCI Repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science. www.ics.uci.edu/~mlearn/MLRepository.html

Cleveland, W. S. and Devlin, S. J. (1988) Locally-weighted regression: An approach to regression analysis by local fitting, J. Am. Statist. Assoc. 83 596 - 610.

Culberson, J. On the Futility of Blind Search: An Algorithmic View of 'No Free Lunch', Evolutionary Computation 6 (1998): 109–27.

Detrano, R. et al. International application of a new probability algorithm for the diagnosis of coronary artery disease. American Journal of Cardiology 64:304-310, 1989.

Edmonds, B. The Pragmatic Roots of Context. in Bouquet, P. et al. (eds.) Modeling and Using Contexts: Proceedings of the 2nd International and Interdisciplinary Conference, Springer-Verlag, Lecture Notes in Artificial Intelligence, 1688:119-134, 1999.

Edmonds, B. Learning Appropriate Contexts. in Akman, V. et al. (eds.) Modelling and Using Context: Proceedings of the 3rd International and Interdisciplinary Conference, Springer-Verlag, Lecture Notes in Artificial Intelligence, 2116:143-155, 2001.

Gennari, J. H. Langley, P. and Fisher, D. Models of incremental concept formation, Artificial Intelligence, 40:11-61.

Kaufman, L. and Rousseeuw, P. J. (1990) Finding groups in data: and introduction to cluster analysis, John-Wiley & Sons.

Koza, J. R. Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA, USA, 1992.

Koza, J. R. Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press, Cambridge, MA, USA, 1994.

Potter, M. A. The Design and Analysis of a Computational Model of Cooperative Coevolution (Doctoral dissertation, George Mason University). (1997)

Potter, M. A. and De Jong, K. A Cooperative Coevolutionary Approach to Function Optimization. In: The 3rd Parallel Problem Solving from Nature (PPSN), Proceedings of the Conference. Springer-Verlag, New York (1994) 249-257

Reader, J. (1990) Man on Earth. Penguin Books.

Spears, W. "Simple subpopulation schemes". In: Proceedings of the 3rd Conference on Evolutionary Programming. World Scientific (1994) 297-307.

Tanese, R. (1987) Parallel genetic algorithms for a hypercube. In Grefebstette, J. J. (ed.), Genetic Algorithms and their Applications: Proceedings of the 2nd International conference on genetic algorithms, Hillsdale, NJ: Lawrence Erlbaum, 177-183.

Vose, M. D. and Liepins (1991) Wright's shifting balance theory: an experimental study. Science 253:1015-1018.

Wiegrad, R. (1999) Applying Diffusion to a Cooperative Coevolutionary Model. Parallel Problem Solving from Nature PPSN V 5th International Conference, Springer, LNAI ??

Wolpert, D. H. and Macready, W. G. No Free Lunch Theorems for Optimization, IEEE Transactions on Evolutionary Computation 1 (1997): 67–82.

Wright, S. (1932) The roles of mutation, inbreeding, crossbreeding and selection in evolution. In Proceedings of the 6th International Congress of Genetics, vl. 1, 356-366.

Appendix 1 – The Data Set

The information given below is culled from the information file that comes with the data set at the Repository of machine learning databases (Blake and Merz 1998). I include it for completeness – I have almost no knowledge of heart disease.

Title

Heart Disease Databases (processed Cleveland subset)

Source Information:

- Creator: V.A. Medical Center, Long Beach and Cleveland Clinic Foundation: Robert Detrano, M.D., Ph.D.
- Donor: David W. Aha (aha@ics.uci.edu) (714) 856-8779
- Date: July, 1988
- Obtainable from: www.ics.uci.edu/~mlearn/MLRepository.html

Past usage

(Detrano et al 1989) achieve approximately a 77% correct classification accuracy (i.e. 23% error) with a logistic-regression-derived discriminant function on similar data sets. (Aha and Kibler) achieved a 77% accuracy with Ntgrowth and 74.8% accuracy with C4, using instance-base prediction of heart-disease presence. (Gennari, Langley and Fisher 1989) achieved a 79.9% accuracy using their CLASSIT conceptual clustering system. The last two were on the same data set as used here.

Summary

The full database contains 76 attributes, but all published experiments refer to using a subset of 14 of them. There are, in fact, four data sets from different parts of the world, but the Cleveland database is the only one that has been used by ML researchers to this date. The "goal" field refers to the presence of heart disease in the patient. It is integer valued from 0 (no presence) to 4. Experiments with the Cleveland database have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0).

There are 303 instances, but this includes fields with missing data. The subset used here were the 281 with complete data.

Attributes

Only 14 used in the processed subset. The hashed number is the attribute number in the complete set.

1. #3 (age): age in years
2. #4 (sex): sex (1 = male; 0 = female)
3. #9 (cp): chest pain type
    - Value 1: typical angina
    - Value 2: atypical angina
    - Value 3: non-anginal pain
    - Value 4: asymptomatic
4. #10 (trestbps): resting blood pressure (in mm Hg on admission to the hospital)
5. #12 (chol): serum cholesterol in mg/dl
6. #16 (fbs): (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
7. #19 (restecg): resting electrocardiograph results
    - Value 0: normal
    - Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)
    - Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
8. #32 (thalach): maximum heart rate achieved
9. #38 (exang): exercise induced angina (1 = yes; 0 = no)
10. #40 (oldpeak): ST depression induced by exercise relative to rest
11. #41 (slope): slope: the slope of the peak exercise ST segment
    - Value 1: upsloping
    - Value 2: flat
    - Value 3: downsloping
12. #44 (ca): number of major vessels (0-3) coloured by fluoroscopy
13. #51 (thal): 3 = normal; 6 = fixed defect; 7 = reversible defect
14. #58 (num) (the predicted attribute): diagnosis

Appendix 2 –More Detailed Model Description

Static Structure

The data limits/determines what informational attributes are available to learn from. In this case there where 13 attributes. From these we chose a subset of these to define the “problem space” (in this case two different pairs). Due to the nature of the problem chosen in this case (a finite data set), there are effectively only finite subset of locations that can be learned about – those for which we have data. The solutions are distributed among this finite set of locations. Here the obvious optimisation of calculating the distances between the locations once at the start was made, thus forming a network defined by the nearest-neighbour relation. Each location is associated with a different subset of the data (those with the same values in terms of the problem space attributes).

Dynamic Structure

What changes as the algorithm progresses are the solutions at each location.

Solution Language

Each solution is composed of 5 numerical functions corresponding to each of the 5 possible outcomes (0, 1, 2, 3, or 4) for attribute 14. The predicted outcome is decided by which of the 5 functions outputs the highest value when evaluated with values for the 13 attributes. The functions are specified as GP tree-structures which are separately interpreted when the solution is evaluated.

The non-terminal nodes of these trees are one of the following.

- IGZ –3 arguments. If the first evaluates to greater than zero, return the result of evaluating the second argument else the third argument.
- MAX, MIN –2 arguments. Returns the maximum or the minimum of the results of the arguments.
- PLUS, MINUS, TIMES –2 arguments. Returns the obvious arithmetic calculation.
- SAFEDIVIDE – 2 arguments. Returns the division unless the divisor is 0 then return 0.

The terminals of the trees are one of the following.

- The value of one of the give attributes: INPUT1, … INPUT13
- One of a set of supplied constants: CONSTANT –1, CONSTANT –0.9, … CONSTANT 0, … CONSTANT 0.9, CONSTANT 1.

Algorithm

The algorithm is outlined in Figure 9 below.

Initialisation

The population was initialised with random trees to the depth specified according to the gene language. There are the specified number of genes at each location.

Important Parameters

There are the following global parameters for all versions and stages of the algorithm.

- Crossover probability (= 1 – propagation probability) [0…1]
- Tournament size [2, 3,…]
- Number of solutions at each location [1, 2, …]
- Size of neighbourhood [1, 2, …]
- Number of generations in development phase [1, 2, …]
- Number of generations in analysis phase [1, 2, …]
- Initial GP tree depth [1, 2, …]
- Number of neighbours [1, 2, …]

In the global run all solutions are at a single location and there is no analysis phase – thus the neighbourhood size, number of neighbours have no effect. In the global version there is also the following parameter.

- Number of samples used in evaluation [1, 2, …]

Variations

As a control the *global* algorithm imitates a standard GP algorithm, by having a single location, selecting solutions probilistically according to their fitness, and evaluating their fitness across a random sample of the whole data set.

The two local sets of runs varied only in the attributes chosen to define the problem space.

```
Randomly generate candidate models and place them randomly about
the domain, D
for each generation
   repeat
      randomly pick a point, P, in D
      pick n models, C, from locality within neighbhdSize of P
      evaluate all in C at P
      pick random number x from [0,1)
      if x < (1 - crossover probability)
         then copy the fittest in C to position P
         else cross two fittest in C, put result at P
   until new population is complete
next generation
```

Figure 9. The skeleton of the algorithm used in this paper

Comparison of Reproduction Schemes in an Artificial Society for Cooperative Gathering

A.E. Eiben G.S. Nitschke M.C. Schut

Artificial Intelligence Section
Department of Computer Science
Vrije Universiteit Amsterdam
De Boelelaan 1081a, 1081 HV Amsterdam, NL
gusz@cs.vu.nl nitschke@cs.vu.nl schut@cs.vu.nl

Abstract

This paper compares reproduction schemes for adaptive behavior in an artificial society, where the collective task of the society is the gathering of resources in an artificial environment. The environment is randomly distributed with varying quantities of different resource types, where different resource types yield different fitness rewards for agents that successfully gather them. Gathering of the more valuable resource types (those yielding higher fitness rewards) requires cooperative behavior of varying degrees (a certain number of agents working collectively). We compared reproduction schemes over three dimensions. The first was a comparison of agents that could reproduce only at the end of their lifetimes (*single reproduction at the end of the agent's lifetime*) and agents that could reproduce several times during their lifetime (*multiple reproduction during lifetime*). The second was a comparison of agents that could reproduce only with agents in adjacent positions and agents that could reproduce with agents at any position in the environment. The third compared different methods for deriving the number of offspring produced and the fitness share given to each offspring, as well as stochastic variants of these methods. Results indicate that the single reproduction at the end of the agent's lifetime scheme afforded the artificial society a higher level of performance in its collective task, according to the evaluation criterion, comparative to artificial societies utilizing the multiple reproductions during lifetime reproduction scheme.

1 Introduction

Our research interest can be best described by the term Emergent Collective Intelligence (ECI)[1]. It is rooted in the artificial society simulations field in that it concerns groups of agents, specifically, collectives, which develop certain properties bottom-up. The applications we envision include engineering tasks.

We are interested in the design of cooperative behaviors in groups of agents, where such cooperative behavior could not be developed or specified *a priori*. The key idea is that a desired group behavior emerges from the interaction of the component agents, where no single agent would be able to accomplish the task individually, the task is predefined, and the environment is unknown. The end goal of such an artificial social system would be the transference of a cooperative behavior design methodology to a physical system (for example: multi-robot) that has a specific and well-defined task in an unexplored environment. For example, we envisage the use of such a methodology in swarm-robotics (Nolfi *et al.* 2003) for the gathering of resources in hazardous locations (for example: the surface of Mars or a deep-sea ocean bed). Hence, associating a concrete task with the artificial social system introduces the engineering or design element. If one can measure how well the given task is performed, we have a natural optimization criterion. Consequently, a well-calibrated system will be one where the evolutionary mechanisms (and probably other adaptive features) are able to generate high quality collective behaviors efficiently.

In this paper we consider the task of collective gathering, where a group of agents need to explore their environment in order to find some resources, mine them and collect them at a central location. The formal objective here can be expressed by the total value of resources gathered together in a given amount of time. The system, the environment, and

[1] http://www.cs.vu.nl/ci/eci

the task will be described in *Section 3: Simulator, Environment and Agents*.

As for the agent collective we use an adaptive artificial social system where our technical research goal is to establish what reproduction mechanisms lead to the best results in terms of the total value of resources gathered. In particular, we investigate:

1. Two reproduction schemes, single reproduction at the end of the agent's lifetime (SREL) and multiple reproduction during an agent's lifetime (MRDL)
2. Two mate selection methods locally restricted mating versus panmictic mating.
3. Two methods for determining the initial fitness of new individuals at birth, and for both methods we applied:
   3a. A deterministic variant
   3b.A stochastic variant

These issues will be discussed in *section 4: Experiments* and *section 5: Analysis and Discussion*.

2 Related Literature

This section presents a brief overview of prevalent results pertaining to the study of emergent cooperative behavior, particularly: cooperative gathering and transport, within simulated swarm-based systems. The term swarm-based systems refer to artificial societies containing potentially thousands of agents. Results reviewed maintain particular reference to research that uses biologically inspired design principles and concepts, such as emergence, evolution and self-organization, as a means of deriving cooperative behavior to accomplish tasks that could not otherwise be individually accomplished.

The study of the synthesis of collective behaviour, particularly the emergence of cooperation, is a research field in which there has been little work done in both simulated (Iba, 1996) and real world (Quinn, 2000) problem domains. Traditionally collective behaviour and multi-agent systems have been studied using a top down classical approach. Such approaches have achieved limited success given that it is extremely difficult to specify the mechanisms for cooperation or collective intelligence in all but the simplest problem domains. The investigation of artificial evolution relating to emergent collective behavior, specifically cooperation, remains a relatively unexplored area of research in the cooperative gathering and transport problem domain.

With relatively few exceptions, and then only in multi-robot systems containing relatively few robots (Mataric, 1992), the majority of research in emergent cooperative behavior is restricted to simulated problem domains given the inherent complexity of applying evolutionary design principles to collective behaviors in groups of real robots (Floreano and Nolfi, 2000). This is especially true in swarm-based systems, which by definition contain thousands of individuals.

Within simulated swarm-based systems there has been a significant concentration of research on the study of emergent behavior in artificial ant colonies (Deneubourg *et al.* 1987). Certain artificial life simulators and applications have popularized studies of swarm-based systems. These include *Swarm* (Daniels 1999), *MANTA* (Drogoul *et al.* 1995), *Tierra* (Ray, 2001), and *Avida* (Adami, 1994).

Drogoul *et al.* (1992a; 1992b), (Drogoul and Ferber, 1992) presented a simulation model of social organization in an ant colony termed: *MANTA* (*M*odel of an *ANT*-hill *A*ctivity), which was designed to explore the contribution of emergent functionality such as division of labor on emergent cooperation. Results elucidated that emergent division of labor improved the efficiency of emergent functionality in the population. Such emergent functionality included cooperative foraging and sorting behavior. The authors concluded that the notion of emergent cooperation remains very unclear, difficult to define, and that many of the behaviors viewed as cooperative emerged as a result of the competitive interaction that occurs between individuals in a constrained environment with limited resources.

As part of the swarm-bots initiative, Nolfi *et al.* (2003) conducted several experiments to address the problem of how a group of simulated robots (s-bots) could coordinate their movements and actions so as to cooperatively move objects in the environment as far as possible within a given period of time. Nolfi *et al.* (2003) conducted a set of experiments designed to facilitate emergent cooperative behavior, where a group of eight s-bots were connected to an object, or connected so as to form a closed structure around an object, and were given the task of moving the object as far as possible in the least amount of time. In the first set of experiments the eight s-bots used what the authors termed the *ant formation*, which connected all s-bots to the object, but there were no links between the s-bots themselves. The result was dependent upon the weight of the object, such that the s-bots cooperatively negotiated to either push or pull the object to their destination. In the second set of experiments, s-bots were assembled so as to form a circular structure around the object. The results were similar to those obtained with the ant-formation, with the exception that the s-bot formation deformed its shape so that some s-bots pushed the object, while other s-bots pulled the object. The mechanism deemed to be primarily responsible for these results was the neural controllers of individual s-bots, which evolved the capability to cooperatively coordinate movement when connected to either each other or the object. That is, each s-bot was inclined to follow the direction that the majority of s-bots followed at a given time.

From this overview of these different research efforts, associable by similar tasks and the general research topic of emergent cooperation, it is obvious that some formalization of mechanisms for the design and analysis of emergent cooperation is needed. Specifically, if emergent cooperative behavior in swarm systems was sufficiently understood, purposeful design of cooperative behavior could be applied to benefit a variety of application domains including telecommunications (Di Caro and Dorigo, 1998), space exploration (Brooks and Flynn, 1998) and multi-robot systems (Mitsumoto *et al.* 1995).

3 Environment and Agents

The experiments presented in this paper were performed with our simulation framework: JAWAS[2]. Using this framework we implemented a particular environment and agents populating this environment.

3.1 Swarm-Scape

Swarm-Scape is a specific swarm-based model implemented within the JAWAS simulation framework. Swarm-scape utilizes an initial population of 1000 agents, placed at random positions on a grid-cell environment with a 50 x 50 resolution. A maximum of 4 agents can occupy any given grid-cell within the environment. Also, a home area spanning 4 x 4 grid-cells is randomly placed somewhere within the environment. This home area is where each agent must deliver resources that it is transporting. The process of mining, transporting, and delivering a resource is termed gathering.

Within the environment there exist three types of resources: gold, iron and stone. It is essential in our design that resources also have a *value* that can differ for different types of resources. In particular, in our present system one stone-unit is worth of 1 abstract unit of value, one iron-unit is worth 2, and one gold-unit is worth 4.

Initially, there is some quantity, defined in terms of resource units, of each resource type. For each grid-cell, a maximum quantity (number of resource units) of each resource is specified, and for all grid-cells the re-grow rates (number of resource units that are replenished per simulation iteration) of each resource is specified. Each of these resources has different properties pertaining to its value and cost to transport for each agent.

[2] JAWAS: Java Artificial Worlds and Agent Societies, can be downloaded from http://www.cs.vu.nl/ci/eci/

In order to mine each resource some degree of cooperative behavior is necessitated. Specifically, to mine a unit of gold (the most valuable resource), 4 agents need to be situated on the same grid-cell. To mine a unit of iron (the medium valued resource), at least 3 agents need to be situated on the same grid-cell. To mine a unit of stone (the least valuable resource), only a single agent needs to be situated on the grid-cell. For the purposes of the experiments described within this paper, the term *cooperation* was defined as the instance when at least two agents, situated on the same grid-cell, simultaneously attempted to mine the same resource unit.

3.2 Task Environment

The task of each agent in the environment is the gathering of the highest possible value of resources during the course of its lifetime. This task was interfaced to the agent collective by using the value of the resources gathered by an agent, where gathered value translates into fitness rewards. In our system, fitness was used as a metaphor of energy: performing actions costs fitness units. Furthermore, fitness also played its conventional role in survivor selection: if an agent's fitness reaches zero, it dies.

The particular method we used to reward agents' performance worked as follows. In an instance when a resource unit is delivered to the home area, the agent is given a fitness reward proportional to the total value of the resource units delivered. Specifically, one gold-unit yields a fitness reward of 20 fitness units, 1 iron-unit yields a fitness reward of 10 fitness units, and 1 stone-unit yields a fitness reward of 5 fitness units. The total fitness reward corresponded to the total value of the resources an agent delivered.

The initial amount of gold, iron and stone in the environment was 250, 500, and 1000 respectively, where the number of resource units that could be on any given grid-cell was unlimited. The re-grow rate for each of the three resources was 1 unit per 3 simulation iterations.

3.3 Swarm Agents

Our agents were based on the classical SugarScape design, adopting most of the SugarScape features (Epstein and Axtell, 1996). An agent was able to detect agents and resources for a number of grid-cells determined by a *sight* property. Specifically, an agent was able to detect the number of agents, and the types of resources, in all grid-cells surrounding its current position for a distance (number of cells) given by *sight*.

Each Swarm-Agent used the following set of heuristics in order to determine the action it takes during any given simulation iteration:

```
IF end of life and SREL active THEN reproduce
IF at home THEN unload resources transported
  IF MRDL active THEN reproduce
IF transporting a resource THEN go home
ELSE IF gold detected THEN move to gold
  ELSE IF iron detected THEN move to iron
    ELSE IF stone detected THEN move to stone
      ELSE move to a random cell
```

For any given simulation iteration, each agent was able to move for a number of grid-cells in any position given by the value set for its *move* property. Both the sight and move properties were initially set to one grid-cell. Also, upon initialization each agent was assigned the maximum time for which it would live, assuming that it did not reach zero fitness before this time. This property termed: *death age* was randomly set for each agent to a value between 40 and 80 upon its initialization.

Each agent in the population followed a set of heuristics directing the agent to move, to mine, and then to transport the most valuable resource it could find in the environment. Once an agent had mined as much of a given resource as it could transport (determined by the resource type and the number of units mined), it would immediately begin transporting the resource units back to the home area. Each agent had several properties dictating restrictions on its behavior.

The *maximum gold mining capacity* property specified the maximum number of gold units that each, of 4 cooperating agents, could mine. For these experiments the maximum gold mining capacity property was set to 5. The *maximum iron mining capacity* property specified the maximum number of iron units that each, of at least 3 cooperating agents, could mine. For these experiments the maximum iron mining capacity property was set to 10. The *maximum stone mining capacity* property specified the maximum number of stone units that each agent could mine. For these experiments the maximum stone mining capacity property was set to 20.The *transport-capacity* property determined the maximum number of units of resources a single agent could transport.

An important property for each agent was its *fitness* (that is: the agent's energy rating). At the beginning of each simulation, fitness was randomly initialized for each agent to a value between 90 and 100. Every action taken by the agent cost some portion of its fitness. Mining of any resource type cost one fitness unit. Every grid-cell of distance that an agent moved cost one fitness unit. An agent's fitness could only be replenished when it delivered a resource unit to the home area of the environment.

The initialization settings for each of these parameters is based the most 'appropriate' settings for the given environment, as ascertained in previous experiments (Vink, 2004).

3.4 Reproduction of Swarm Agents

In our system, agents evolved, that is, they underwent variation and selection where the environment performed selection implicitly. Agents with a high fitness (those that performed their tasks most efficiently) were selected for, where as poorly performing agents with not enough fitness died. Variation of agents was accomplished by recombination of agent genotypes.

The core of reproduction was the reproduction cycle where two parent agents created a number of offspring agents via recombining their own genes for *maximum gold mining capacity, maximum iron mining capacity, maximum stone mining capacity* and *transport-capacity* and passing the average of their values onto their offspring.

In this investigation we compared two temporal schemes for reproduction. In the SREL scheme an agent could only perform one *Single Reproduction act at the End of its Lifetime*. That is, when each agent reached the end of its lifetime it selected m mates (partner agents) and then produced a number of offspring according to the particular reproduction method being used. In the MRDL scheme *Multiple Reproduction* acts are executed *During Lifetime*. Using the MRDL scheme, every agent was able to reproduce when a resource quantity was delivered to the home area. Upon delivery of a resource quantity, the agent would receive an immediate fitness reward, and a reproduction cycle would start. During this cycle the agent would select m partner agents from the environment, and then produce a number of offspring according to the reproduction parameters being used.

The second reproduction feature we studied here concerns the spatial distribution of mates for reproduction: *panmictic* versus *locally restricted* mate selection. Using the locally restricted method, an agent could only reproduce with agents in the adjacent grid-cells. In this case, all agents on the same grid-cell or in adjacent grid-cells were taken into account as mates. Using the panmictic method, an agent could reproduce with any other agent anywhere else in the environment. In this case the number of mates m was a random integer between 0 and 10 drawn with a uniform distribution.

Third, we compared two methods for determining the initial fitness given to offspring agents at birth. For both fitness inheritance methods we used a distribution mechanism where 90 percent of a parent agent's fitness was passed onto and divided among its offspring and we divided the total amount of fitness to be inherited (x) over the number of children (n) equally, that is, giving each offspring agent $y = x/n$ fitness units. The parameters to distinguish the investigated methods were n and y.

Using the first method, n, the number of offspring to be produced was predefined and y was derived for each reproduction act by dividing the actual value of x for the two given parent agents by n. In the second method, the fitness share y was predefined and n was determined as x/y (rounded up). The values we used for our experiments are $n = 5$ for the fixed number of offspring method and $y = 10$ for the fixed offspring fitness method.

For both fitness inheritance methods we applied deterministic and stochastic variants. The deterministic variants simply used outcomes of the calculation (rounded up, when needed). The stochastic variants were the same two methods, though random noise was added to the fitness share (in the case of the first method), or random noise to the number of children produced (in the case of the second method). In the case of the first stochastic variant, the random noise was generated within the range between -1 and +1 by a uniform distribution, and in the case of the second variant, random noise was generated within the range of -5 and +5.

4 Experiments and Results

We designed our experiments along three parameter dimensions and two values for each dimension as outlined in the research objectives:

1. Reproduction scheme: SREL versus MRDL.
2. Mate selection method: panmictic versus locally restricted.
3. Fitness inheritance method: fixed n or fixed y.

This led to 8 different experimental setups, although since we also compared a deterministic and a stochastic variant for the inheritance methods, the total number of different experimental setups was 16. For each of them we performed 50 independent runs (using different random initialization parameters), where one run was executed for 2000 iterations.

4.1 Simulation Monitors

Within each simulation, several experimental monitors are set as objective measures for the performance of the society across multiple generations of agents. The first and second are the *number of agents* and the *average value gathered cooperatively* since it is these that determine the value of resources gathered together in a given amount of time, which is our formal objective. The *average fitness* of the population and the *average distance to home*, which describes the population density, are additional measures illuminating details on the overall behavior of the artificial society.

As presented in section 3, cooperative behavior was evaluated according to the total value of each resource: gold, iron, and stone, gathered by the agent population over the course of a given simulation. Specifically, the measure of cooperative behavior is the total value gathered cooperatively, which includes all resource types gathered by the society over the course of the simulation. Sub-measures of this are: value of gold gathered cooperatively, value of iron gathered cooperatively, and value of stone gathered cooperatively. These measures can be simply monitored via the GUI and saved for off-line analysis later on, but are not reported in the present paper.

4.2 Results

Figures 1 through to 8 present results attained for the objective measures described above with all 16 different setups. The presentation principle we follow is to use a table style arrangement, with four rows and two columns. Here, each row belongs to one of the measures; the two columns correspond to the two reproduction schemes we investigated. A cell in this table contains a graph divided into a right-hand side and a left-hand side histogram, belonging to the two methods for distributing the parents' fitness over the offspring. Within each histogram deterministic and stochastic variants of these methods are further distinguished by their left/right position. Finally, the two colours are used represent the two mate selection methods.

5 Analysis and Discussion

As mentioned in the introduction, our formal objective is to maximize the total value of resources gathered. To this end, the average value gathered collectively and the average number of agents is essential, as their product indicates how well the population performs.

The reproduction scheme turned out to be one the most influential features in our study, that is, the feature with the highest impact on performance. The impact was most prominent on population sizes. Using the multiple reproductions during lifetime scheme (MRDL) the population sizes varied in a range that was around one tenth of population sizes under the single reproduction at the end of lifetime (SREL) scheme. This is remarkable, in that the number of reproduction cycles was much lower when agents are only allowed to mate once in a lifetime. Apparently, it is worthy to "save" fitness for a longer period and create offspring only in a "rich" state. Perusing the average values gathered one could observe that the impact of the reproduction scheme is much less (as presented in figures 2 and 6). Differences are at most of a factor 2 to 3, sometimes in favour of SREL, sometimes not. Concerning the net effects on total value gathered by the whole population[3] the SREL scheme is the clear winner.

Interestingly, the average fitness values were much less sensitive to these reproduction schemes. In 8 out of the 16 experiments average fitness values did not differ significantly for the SREL and MRDL schemes (as illustrated in figures 3 and 7). In the other 8 cases they did differ in about a factor 3 to 5 in favour of the MRDL scheme. The figures on the average distance to home measure, disclose that the MRDL scheme evolved smaller and denser populations.

The investigated options for the mate selection method, panmictic versus locally restricted reproduction, showed no significant differences in performance for our task environment.

For the inheritance method we could make observations quite similar to those about reproduction schemes. The most affected measure was the population size with differences up to a factor 10. Variations in the average value gathered were much less, up to a maximum of factor 2 to 3. Whether the fixed number of offspring (n) or the fixed offspring fitness (y) method worked better depended on the usage of random noise. For instance, using a fixed y in a deterministic way enabled much larger populations than its stochastic counterpart. However, the fixed n method worked much better in the stochastic variant.

6 Conclusions and Future Work

In this paper we presented an artificial society and a particular task the inhabitants of this society needed to accomplish. This task was the gathering (finding, mining, transporting, and delivering) of certain resources. Resources differed in their difficulty to mine, in that they required a different degree of cooperation to be mined. Resources also differed in their value; that is: the rewards an agent would receive upon delivery were different. Resource mining difficulty and value were related: more difficult resources were worth more.

We investigated reproduction mechanisms within this society and found that two features clearly influenced the performance of the agent population. Firstly, the results of our investigation show that the single reproduction at end of lifetime (SREL) scheme yielded a higher total amount that the population gathered comparative to the multiple reproduction during lifetime (MRDL) scheme. The second feature with a high influence was the fitness inheritance method. The best method depended upon the right combination with either a stochastic or deterministic variant. In particular, we found that the stochastic fixed number of children and deterministic fixed offspring fitness outperformed their counterparts.

The overall best combination of the investigated aspects of the reproduction mechanisms within our world was the SREL reproduction scheme with panmictic mate selection and deterministic fixed offspring fitness. This combination yielded twice the performance (total value gathered cooperatively) of the second best combination.

Three future research objectives have been defined based upon the results presented in this paper. The first is to further investigate the mechanisms that lead to the SREL societies attaining a higher performance (value gathered cooperatively) for the given task, though maintaining a comparable fitness to MRDL societies.

The second is to increase the complexity of the agent controllers and evolutionary process, giving agents the capacity to learn during their lifetimes, as well as evolution the capacity to modify genotypes based upon lifetime behaviors (collective or individual). Modifying the evolutionary process such that a greater part of the agent genotype is subject to evolution would also likely yield greater complexity and diversity in emergent behaviors.

The third is to measure the impact of the number of offspring produced upon the given task. Specifically, to investigate if societies that produce many offspring with small fitness shares have superior performance compared to societies that produce few offspring with relatively large fitness shares.

Forthcoming results will be published on different scientific forums; for locating them conveniently one can visit: http://www.cs.vu.nl/ci/eci.

[3] The total value gathered was the average value gathered (figures 3 and 7) multiplied by the average number of agents (figures 1 and 6).

SREL: Single **R**eproduction at **E**nd of **L**ifetime

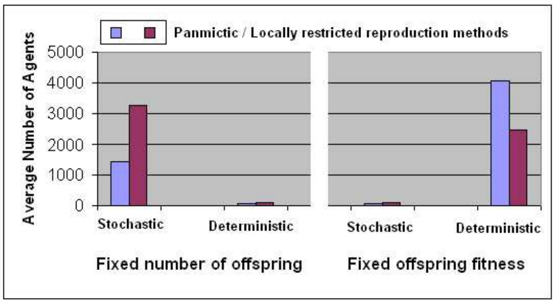


Figure 1: The average number of agents, when using the *SREL* reproduction scheme (Note the scale for the average number of agents in comparisons with figure 5).

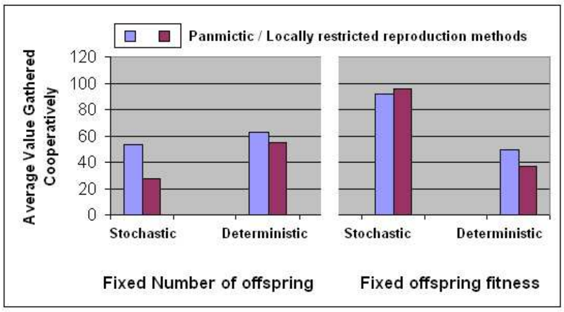


Figure 2: The average resource value gathered cooperatively by the agent population, when using the *SREL* reproduction scheme.

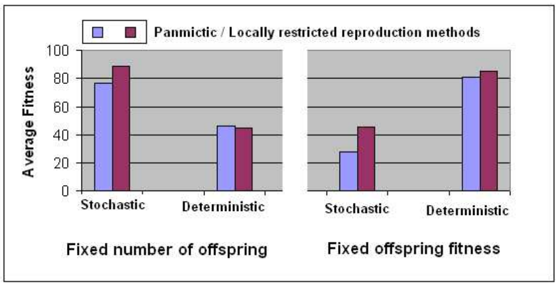


Figure 3: The average fitness of the agent population attained under the *SREL* reproduction scheme.

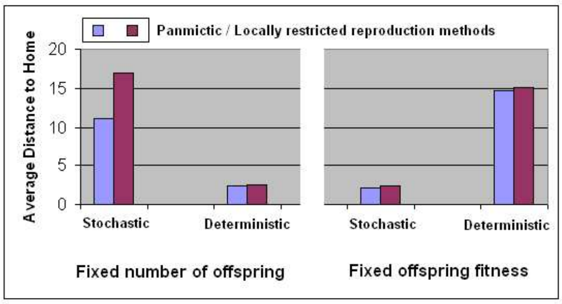


Figure 4: The average distance to home for the agent population, when using the *SREL* reproduction scheme (Note the scale for the average distance to home in comparisons with figure 8).

MRDL: Multiple **R**eproductions **D**uring **L**ifetime

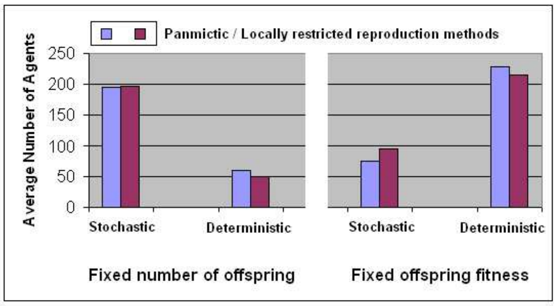


Figure 5: The average number of agents, when using the *MRDL* reproduction scheme (Note the scale for the average number of agents in comparisons with figure 1).

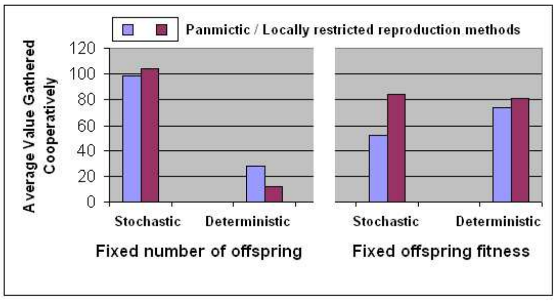


Figure 6: The average resource value gathered cooperatively by the agent population, when using the *MRDL* reproduction scheme.

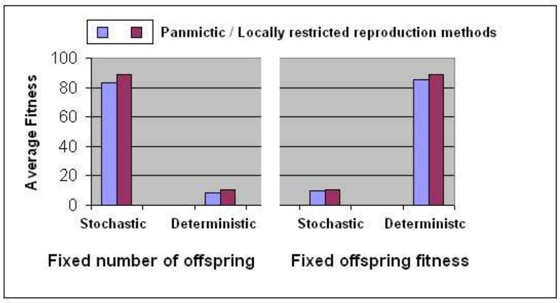


Figure 7: The average fitness of the agent population attained under the *MRDL* reproduction scheme.

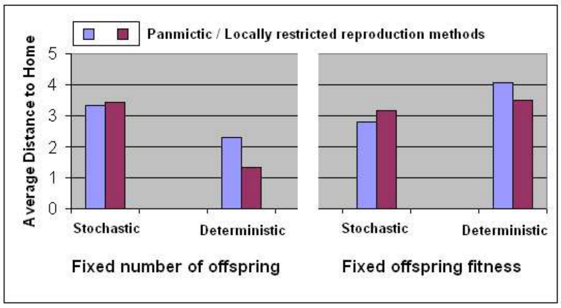


Figure 8: The average distance to home for the agent population, when using the *MRDL* reproduction scheme (Note the scale for the average distance to home in comparisons with figure 4).

Acknowledgements

We would like to thank Nico Vink, Volker Nannen, and Robert Griffioen for their contributions to the JAWAS platform and the inspiring discussions.

References

Adami Chris, Evolutionary Learning in the 2D Artificial Life System Avida, *Technical report*, Kellogg Radiation Lab, Caltech, Pasadena, 1994.

Brooks Rodney and Flynn Anita. Fast, Cheap and Out of Control: A Robot Invasion of the Solar System. *British Interplanetary Society Journal*, 1(1): 478-485, 1989.

Deneubourg Jean-Louis, Goss Simon, Pasteels Jacques, Fresneau Dominique, and Lachaud Jean-Paul. Self-Organization Mechanisms in Ant Societies (II): Learning in Foraging and Division of Labor. *Behavior in Social Insects*, 54(1): 177-196, 1987.

Di Caro Gianni and Dorigo Marco. AntNet: Distributed Stigmergic Control for Communications Networks. *Journal of Artificial Intelligence Research*, 9(1) 317-365, 1998.

Drogoul Alexis, Ferber Jacques, Corbara Bruno, Fresneau Dominique, A behavioral simulation model for the study of emergent social structures, In *Toward a practice of autonomous systems*, Varela Francisco and Bourgine Paul, Eds, Kluwer Academic Publishers, Brussels, pp. 161-170. 1992.

Drogoul Alexis, Corbara Bruno, Fresneau Dominique. Applying Etho-modeling to social organization in ants, In *Biology and Evolution of Social Insects*, Billen Johan, Ed, Leuven University Press, Leuven, pp. 375-383, 1992.

Drogoul Alexis and Ferber Jacques. Using reactive multi-agent systems in simulation and problem solving, In *Distributed Artificial Intelligence: theory and praxis*, Avouris Nicholas and Gasser Les, Eds, Kluwer Academic Publishers, Brussels, pp. 53-80. 1992.

Drogoul Alexis, Corbara Bruno, Lalande Steffen, MANTA: New Experimental Results on the Emergence of (Artificial) Ant Societies, In *Artificial Societies: the computer simulation social life*, Gilbert Nigel and Conte Rosaria, Eds, University College of London Press, London, pp. 190-211, 1995.

Epstein Joshua and Axtell Robert. *Growing Artificial Societies: Social Science From The Bottom Up*, Brookings Institute Press, 1996.

Eiben A.E and Smith Jim. *Introduction to Evolutionary Computing*, Springer, 2003.

Floreano Dario and Nolfi Stefano. *Evolutionary Robotics*. MIT Press, Cambridge, 2000.

Iba Hitoshi. Emergent cooperation for multiple agents using genetic programming. In *Parallel Problem Solving from Nature, PPSN IV.* LNCS 1141, Springer-Verlag, Berlin, pp. 32-41, 1996.

Mataric Maja. Designing emergent behaviors: From local interactions to collective intelligence, In *Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, MIT Press, Cambridge, pp. 432-441, 1992.

Mitsumoto Naoki, Fukuda Toshio, Shimojima Koji, Ogawa Akio. Micro Autonomous Robotic System and Biologically Inspired Swarm Strategy as a Multi Agent Robotic System, In *Proceedings of the IEEE International Conference on Robotics and Automation*, IEEE Press, Tokyo, pp. 2187-2192, 1995.

Nolfi Stefano, Baldassarre Gianluca, and Parisi Domenico. Evolution of collective behavior in a team of physically linked robots, In *Applications of Evolutionary Computing*, Gunther Raidl, Guillot Agnès, Meyer Jean-Arcady, Eds, Springer Verlag, Heidelberg, pp. 581-592, 2003.

Quinn Mathew. Evolving cooperative homogeneous multi-robot teams. In *Proceedings of the International Conference on Intelligent Robots and Systems*, IEEE Press, Takamatsu, pp. 1798-1803, 2000.

Ray Tom. Overview of Tierra, *Technical report*, Advanced Telecommunications Research Institute: Human Information Processing Research Laboratories, Kyoto, 2001.

Vink Nico. Exploring Communication Dynamics in VU-Scape, Technical report, Department of Computer Science, Vrije Universiteit, Amsterdam, The Netherlands, 2004.

A Social Semantic Infrastructure for Decentralised Systems Based on Specification-Carrying Code and Trust

Giovanna Di Marzo Serugendo[⋆]
[⋆]Centre Universitaire d'Informatique
University of Geneva
24, rue Général-Dufour
CH-1211 Geneva 4
Giovanna.Dimarzo@cui.unige.ch

Michel Deriaz[†]
[†]Centre Universitaire d'Informatique
University of Geneva
24, rue Général-Dufour
CH-1211 Geneva 4
Michel.Deriaz@cui.unige.ch

Abstract

Decentralised systems made of autonomous devices and software are gaining more and more interest. These autonomous elements usually do not know each other in advance and act without any central control. They thus form a society of devices and software, and as such need: *basic interaction mechanisms* for understanding each other, and a *social infrastructure* supporting interactions taking place in an uncertain environment. In an effort to go beyond pre-established communication schema and to cope with uncertainty, this paper proposes an interaction mechanism based exclusively: on semantic information expressed using specifications, and on a social infrastructure relying on trust and reputation.

1 Introduction

The growing diffusion of personal devices connected to Internet is promoting the development of pervasive and wireless applications, as well as those that are to be deployed on a Grid or on a P2P network. A key characteristic of these applications is their self-organised and decentralised nature, i.e., they are made of autonomous software entities which do not know each other in advance and act without any central control. These software entities need advanced means of communication: for understanding each other, to gather and share knowledge, information and experience among each other, and to ensure their own security (data integrity, confidentiality, authentication, access control). Therefore, such a technology needs a social infrastructure supporting, in an intertwined way: mutual understanding, knowledge sharing and security support.

This paper proposes to combine a meta-ontology framework with a dynamic trust-based management system, in order to produce a social semantic middleware supporting the diffusion of semantic information among interoperable software.

The proposed infrastructure relies on the notion of Specification-Carrying Code (SCC) as a basis for mutual understanding, acting as a meta-ontology. Each autonomous software entity incorporates more information than its operational behaviour, and publishes more data than its signature. The idea is to provide separately, for each entity, a functional part implementing its behaviour - the traditional program code; and an abstract description of the entity's functional behaviour - a semantical behavioural description under the form of formal specification. In order to cope with the uncertainty about the environment, and peer entities, individual entities maintain as well local trust values about other entities and share trust and reputation information among themselves.

Such an interaction mechanism is useful for large scale systems (world-wide, or with high density), where a centralised control is not possible, and for which a human admnistration must be completed by a self-management of the software. Domains of applications of such an interaction mechanism include P2P, Grid computing systems, as well as emerging domains such as Autonomic Computing, or Ambient Intelligence.

Section 2 presents the principles of the Specification-Carrying Code paradigm and the associated Service Oriented Architecture. Section 3 then explains how trust-based management systems can be combined with SCC in order to produce a social semantic infrastructure supporting autonomous decentralised software. Finally, Section 4 describes some related works.

2 Specification-Carrying Code

At the basis of any social life, we find communication capabilities. Communication is grounded on common understanding of the information that is transmitted along communication media. In the case of social insects, pheromone deposited by ants in their habitat is correctly understood depending on whether it refers to food, or to the nest. In the case of human beings, words of the language refer to well understood concepts. Similarly, societies of devices and software need interactions based on a common understanding, i.e. relying on a common semantics. Current practice usually rely on pre-established common meanings: communication through shared APIs, usually already shared at design time and which are uniquely a syntactic expression of signatures; communication through shared ontologies allowing run-time adequacy but requiring sharing of keywords. We foresee that future programming practice will consist in programming components and "pushing" them into an execution environment which will support their interactions. Therefore, future components will be developed so as to share a minimal design time common understanding.

The idea advocated in this paper is that interactions should be based on a minimal common basis, merely *concepts*. Pragmatically, for articifial entities to understand each other, those concepts have to be expressed in some language. Therefore, the minimial common basis consists in a common *specification language* used for expressing the concepts. Concepts can then be expressed with different words, and with different properties, but equivalent concepts should share equivalent properties. Thus, there is no need to share identical expression of concepts (either through APIs, ontologies, or identical specifications). However, it is necessary to have a run-time tool able to process those specifications and to determine which of them refer to the same concept.

Pushing the idea at its extreme, even different specification languages could be used simultaneously by different entities to communicate provided there exists translators from one language to the other. But this is beyond the scope of this paper.

In practice, in addition to their code, entities carry a specification of the functional (as well as non-functional capabilities) they offer to the community. The specification is expressed using a (possibly formal) specification language, for instance a higher-order logical language defining a theory comprised of: functions, axioms and theorems. The specification acts as a meta-ontology and describes semantically the functional and non-functional behaviour of the entity. We call this paradigm *Specification-Carrying Code (SCC)*. In our current model, a service-oriented architecture supports the paradigm. Before interacting with a service providing entity, a requesting entity may check (through run-time proof checking) some of its own theorem on the submitted theory. Vice-versa, before accepting to deliver a service, a service providing entity may check the correctness of the requesting entity. This allows an entity to interact with another entity only if it can check that the way the other entity intends to work corresponds to what is expected. The important thing to note here is that entities do not share any common API related to the offered/requested service. Indeed, since entities do not know in advance (at design time) with which entities they will interact, the specification language acts as the minimal common basis among the entities. The lack of APIs implies in turn that input/output parameters can only be of very simple types.

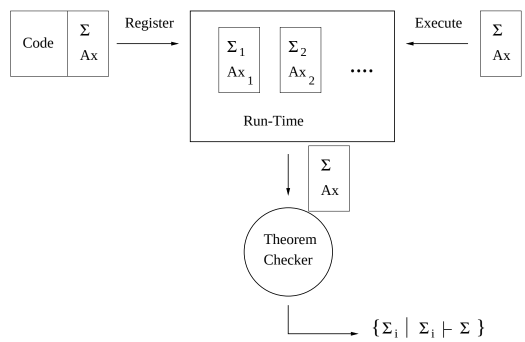


Figure 1: SCC Principle

Figure 1 shows two basic primitives of the SCC paradigm: a service providing entity *registers* its specification to some run-time middleware that stores the specification in some repository. An entity requesting a service specifies this service through a specification, and asks the run-time middleware to *execute* a service corresponding to the specification.

Once it receives an execute request the run-time infrastructure activates a model checker that determines which of the registered services is actually able to satisfy the request (on the basis of its registered specification). The theorem checker establishes the list of all services whose semantics corresponds to the request. Depending on the implementations, the run-time infrastructure may either chose (non-deterministically) one service, activate it and give back the result (if any) to the requesting entity; or pass the information to the requesting entity which will directly contact the service provider. In the first case, the communication is anonymous, while in the second case it is not. Depending on the situations, both cases are valuable.

Depending on the chosen specification language, the specification may vary from a series of keywords together with some input/output parameters description, to a highly expressive formal specification consisting of a signature and additional axioms and theorems characterising the behaviour of the operators specified in the signature. Services matching requests are not necessarily specified in the same textual manner. The theorem checker ensures that they have the same semantics. The more expressive is the specification language, the more it allows to get rid of shared conventions or keywords.

2.1 A Semantic Service-Oriented Architecture

The Specification Carrying Code paradigm is supported by a service-oriented architecture, where autonomous entities register specifications of available services, and request services by the means of specifications. We have realised two different implementations of this service-oriented architecture.

The first implementation has been realised for specifications expressing: signatures of available operators whose parameters are Java primitive types; and quality of service required. Both operators name and quality of service are described using keywords. The resulting environment, a middleware called LuckyJ, allows server programs to deposit a specification of their own behaviour or of a requested behaviour at run-time. In the LuckyJ environment activation of services occurs anonymouly and asynchronously. The service providing entity and the service requesting entity never enter in contact, communication is ensure by the LuckyJ middelware exclusively. The requesting entity is not blocked waiting for a service to be activate. Experiments have been conducted for dynamic evolution of code, where the services can be upgraded during execution without halting or provoking an error in the client program. This is an important feature of decentralised applications since the application transparently self-adapts to new (or updated) services introduced into the environment. The LuckyJ environment only allows the description of basic specification relying on ontology (keywords) shared among all the participating services (Oriol and Di Marzo Serugendo, 2004). Even though LuckyJ allows purely syntactical specifications, it nevertheless proved the viability of the approach under the form of a service-oriented architecture, and its usefulness for dynamic evolution of code.

In order to remove the need for interacting entities to rely on pre-defined keywords, a second implementation of the above architecture has been realised. This architecture allows entities to carry specifications expressed using different kinds of specification language, and is modular enough to allow easy integration of new specification languages (Deriaz and Di Marzo Serugendo, 2004). This architecture supports simple primitives for an entity to register its specifications, or to request a service, and for the environment to execute the corresponding requested code once it has been found.

The current prototype supports specifications written either in Prolog, or as regular expressions. However it cannot check together specifications written in two different languages. In the case of Prolog, the middleware calls SWI Prolog tool to decide about the conformance of two specifications, in the case of regular expressions we have implemented a tool that checks two regular expressions, and is able to transform them into Java code. We foresee the integration of additional specification languages, such as Higher-Order Logic (HOL) and Isabelle theorem checker, JENA, and the Common Simple Logic (CSL).

These languages have different expressive powers: regular expressions are a powerful tool for describing syntactic expressions, and do not support expression of semantical properties. Prolog and HOL are logical languages allowing rich expressivity for describing properties. However, it can rapidly become impracticable to describe usual things such as printing, or complex lists. Therefore, we are investigating languages allowing both logical expressivity and some ontological concepts, such as Jena or CSP.

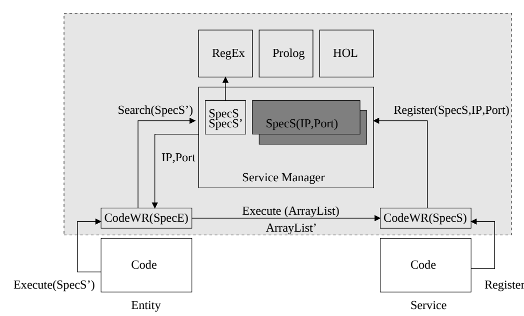


Figure 2: Semantic Service-Oriented Architecture

Figure 2 shows the implemented semantic service-oriented architecture. A *Code* wishing to provide a service or requesting a service is first encapsulated into a wrapper *CodeWR*, which is responsible to handle the specification corresponding to the behaviour of *Code*, and to handle the two basic primitives *Register* and *Execute*. The advantage of using such a wrapper is that with very minor modifications any existing server code can become a specification-carrying code.

A run-time environment, called *Service Manager*, stores specifications of registered services, and activates the corresponding Theorem Checkers once a request has been submitted. In case a service corresponding to the request has been found, the wrapper of the requesting entity then receives the necessary information (IP address and Port number) for contacting directly the service.

The *Code* is not aware that there has been a direct call to a service, the wrapper has transparently managed the whole request. If we consider the wrapper being part of the middleware, communication is anonymous, as in our previous implementation.

Additional information related to programming services and requests can be found in Deriaz and Di Marzo Serugendo (2004).

2.2 Example

2.2.1 Regular Expressions

A specification, is a XML file divided into subsections. Each subsection corresponds to a particular language. Each subsection has to be self- contained: it describes completely a service or a request. A specification file is structured as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<specs>
  <regex active="true">
    ...
  </regex>
  <prolog active="false">
    ...
  </prolog>
</specs>
```

Once it has received an entity request, the service manager tries to match the request specification with the service specification for all languages that are active. In the above example, we see that two languages are defined (regex and prolog) but only one is active (regex). It means that only regular expressions will be taken into consideration. XML allows us to define a different structure for each language. For example in the case of regex, we have four tags: `<name>` which denotes the name of the service, `<params>` which describes the expected parameters, `<result>` which defines the structure of the result, and `<comment>`, which contains optionally additional information.

The following is an example of a sorting *service publication* defined by the regular expression:

```
  <specs>
   <description active="true">
     <content> Sorting Service</content>
  </description>
  <regex active="true">
     <name>(?i)\w*sort\w*</name>
     <params>String\*</params>
     <result>String*</result>
   </regex>
 </specs>
```

The regular expression describing the name `((?i)\w*sort\w*)` accepts all the words that contains the word sort, like quicksort, sorting, or sort. `(?i)` sets the matching case insensitive. The parameters are expressed be the `String\*` regular expression, which means that we expect a list of 0, 1 or more Strings. If we would expect exactly three Strings (for example), we would write String String String. The result tag indicates that this service returns a list of Strings as well. Note that it is of course only a trivial example; the power of regular expressions allows us to express a service name much more precisely.

A service request then is expressed in the following manner:

```
<specs>
  <description active="true">
    <content>Sorting Request</content>
  </description>
  <regex active="true">
    <name>sort</name>
    <params>String*</params>
    <result>String\*</result>
  </regex>
</specs>
```

New tags can be added in the future. Another language can have a completely different structure. These two last points justify the use of such an extensible language as XML.

2.2.2 Prolog

Here is service, able to reverse lists, expressed in Prolog. This service defines first the `append` operator which is necessary to define the reverse operator `rev`. Appending any list `L` to the empty list `[]` returns `L` (line 9). Appending any list `L2` to a non-empty list `[H|T]` (Head and Tail) returns a list with the same head `H` and with `L2` appended to `T` (lines 10, 11).

The `rev` operator is then defined: reversing the empty list, returns the empty list (line 13); and reversing a non-empty list `[H|T]` returns a list `R` obtained by recursively applying `rev` on the tail of the list and appending the head at the end (lines 14, 15).

```
1 <specs>
   <description active="true">
3   <content> Sorting Service</content>
  </description>
5  <regex active="false">
  </regex>
7 <prolog active="true">
    <content>
9    append([],L,L).
     append([H|T],L2,[H|L3]) :-
11       append(T,L2,L3).

13   rev([],[]).
     rev([H|T],R) :-
15       rev(T,RevT), append(RevT,[H],R).
   </prolog>
17 </specs>
```

The specification request simply describes the axioms expected to be satisfied by a reverse operator here called `revlist` (lines 9, 10), as well as the property that reversing two times a list returns the original list (line 11).

```
1  <specs>
   <description active="true">
3    <content> Sorting Service</content>
  </description>
5 <regex active="false">
  </regex>
7 <prolog active="true">
    <content>
9    revlist([],[]), revlist([A|B],R),
     revlist(B,RevB), append(RevB,[A],R),
11   revlist([A|B],R) , revlist(R,[A|B]).
  </prolog>
13 </specs>
```

3 Combining SCC and Trust-Based Systems

Human beings exchange different kinds of *semantical* information for different types of purposes: to understand each other, to share knowledge about someone or something else, to take decisions, to learn more, etc. Despite people share the same understanding regarding information, this information remain local, incomplete and uncertain, leading people to rely on trust to actually take decisions. A common example is provided by the trust put into banking establishments, acting as largely trusted third parties for credit card based interactions.

It is similar for artificial entities that are situated into uncertain environments and that have to interact with unknown entities. Specifications help understanding. However nothing prevents a malicious entity to not follow its specification. In order to fully verify this point, the specification should be accompanied by a proof asserting that the code actually satisfies the specification. Unfortunately, even if a formal proof ensures that the code is not malicious and that it follows its specification, the same code can be, due to bad operational conditions, unable to perform the intended service. Therefore, instead of relying on formal (rigid) proofs, we have preferred to consider a trust-based mechanisms that allows run-time adaptation to peers behaviour.

The model we intend to build thus considers the following two above aspects of human behaviour: (a) communication through semantical information; and (b) ability to take decisions despite uncertainty based on the notion of trust and risk evaluation.

The semantical information is expressed using a specification language conveying the semantical part of the specification. Run-time checked properties assess semantical meaning. Local context information is also provided under this form. This is useful for mobile devices, or mobile code.

As said before, even if properties have been checked, the underlying code can be malicious, or for some reason it cannot follow its specification. Therefore, in addition to the notion of specification (formally describing the basis of interactions), a trust-based model is used for sharing knowledge among entities. This allows run-time adaptation to current behaviour, based on direct observations, and recommendations. The same semantical framework serves for expressing recommendations, or diffusing observations (theories can be dynamically built and modified). In addition, the propagation of properties or theorems integrates well into the trust framework, since sending a theorem is one form of recommendation. Entities exchange information conveying different types of meaning: functionality, non-functional aspects, quality of service, current state; events (recommendations, security attacks, observations), etc.

3.1 Trust-Based Systems

Trust-based systems or reputation systems take their inspiration from human behaviour. Uncertainty and partial knowledge are a key characteristic of the natural world. Despite this uncertainty human beings make choices, take decisions, learn by experience, and adapt their behaviour. We present here two research works from which we will take inspiration to extend our current architecture: an operational model for trust-based control, and a trust calculation algorithm that allows to calculate a global (emergent) reputation from locally maintained trust values.

SECURE Trust System. The European funded SECURE project has established an operational model for trust-based access control. Systems considered by the SECURE project are composed of a set of entities that interact with each other. These entities are autonomous components able to take decisions and initiatives, and are meaningful to trust or distrust. Such entities are called *principals*. Principals are for instance portable digital assistants (PDAs) acting on behalf of a human being, or personal computers, printers, mobile phones, etc. They interact by asking and satisfying services to each other.

In a system based on the human notion of trust (Cahill and al., 2003), principals maintain local *trust values* about other principals. A principal that receives a request for collaboration from another principal, decides or not to actually interact with that principal on the basis of the current trust value it has on that principal for that particular action, and on the risk it may imply of performing it. If the trust value is too low, or the associated risk too high, a principal may reject the request. A PDA requiring an access to a pool of printers, may see its access denied if it is not sufficiently trusted by the printers. For instance, it is known that this PDA sends corrupted files to the printers.

After each interaction, participants update the trust value they have in the partner, based on the evaluated outcome (good or bad) of the interaction. A successful interaction will raise the trust value the principal had in its partner, while an unsuccessful interaction will lower that trust value. Outcomes of interactions are called *direct observations*. After interacting with a printer, a PDA observes the result of the printing. If it is as expected, for instance double-sided, and the document is completely printed, the PDA will adjust the trust value on that particular printer accordingly.

A principal may also ask or receive *recommendations* (in the form of trust values) about other principals. These recommendations are evaluated (they depend on the trust in the recommender), and serve as *indirect observations* for updating current trust values. As for direct observations, recommendations may either raise or lower the current trust value. We call *evidence* both direct and indirect observations. Some PDAs may experience frequent paper jams, on a given printer. They will update (in this case lower) their trust value in that printer, and advertise the others, by sending them their new trust value. The PDA that receives this recommendation will take it into account, and decide if it uses that printer or not (Terzis et al., 2004).

Thus, trust *evolves* with time as a result of evidence, and allows to adapt the behaviour of principals consequently.

EigenTrust. EigenTrust (Kamvar et al., 2003) is a reputation system for P2P networks in which every peer rates the peers from whose they download files. It is an interesting solution to the problem of maintaining in a totally decentralised manner local trust values that globally converge to an emergent reputation value. These values are stored in a local trust vector. Starting from theses local trust values, the distributed EigenTrust algorithm computes a global trust vector, representing the global reputation of each peer. Each peer computes this vector and the authors proved that the computation will always converge to the same global trust vector. Simulations of systems, based on this trust mechanism, show that the number of inauthentic files downloaded by honest peers still significantly decreases even if up to 70% of the peers collude in order to subvert the system.

The idea is that the global reputation of one peer depends on what other peers think about it, according to the successfulness of former transactions, on what friends think about it, on what the friends of friends think about it, and so on; if the chain is long enough, the result of the computation converges to the global trust value.

A set of peers, called score managers, is assigned to each peer. A score manager is responsible to store the global trust value, i.e. the emergent reputation value, of its daughter peer. To determine the score managers of a specific peer, a client peer will apply different distributed hash functions on the peer's identity. All honest score managers of a specific peer will then give the same global trust value.

3.2 Towards a Social Semantic Service Oriented Architecture

In order to incorporate a social layer into our current semantic architecture, we are planning: to extend our current interaction model in order to incorporate trust information; and to adapt the EigenTrust algorithm from file sharing to services requests.

Derived from the SECURE trust-based access model, we describe here trust-based interactions rules grounded on semantic information exchange and global emergent reputation:

- *Request for collaboration and exchange of spec-*

ifications. A principal A receives a request for collaboration from another principal B. A and B exchange their respective capabilities under the form of a specification expressed in the specfication language. They learn each other about their respective provided services.

- *Decision to interact.* Based on the received specification, A and B respectively evaluate if the services provided by the other fulfill its needs (checking of properties expected to be satisfied by the partner).

  The decision then depends on the evaluation of the specification, past direct observations of interactions with B (if any), previously received recommendations about B from other entities, current trust value A has about B, and the risk incurred by the interaction. A may also decide to ask score managers about the reputation of B.

- *Trust Update.* If A decides to interact with B, it will observe the outcome of the interaction, evaluates it (positive or negative), and updates accordingly the local trust value it maintains about B.

- *Reputation Update.* Once local trust values have been updated, the EigenTrust algorithm is then started and the new value of the global reputation is computed.

- *Recommendations.* Besides collaboration requests, A may receive a recommendation from B under the form of specification precising the degree of trust the recommender has on a subject C. Recommendations are evaluated with respect to trust in the recommender, and make the trust A has in the subject C evolve (increase or decrease).

The model defines then a homogeneous framework which serves for expressing and checking semantical information of different kinds: functional behaviour, non-functional behaviour, observations, and recommendations.

3.3 Discussion

Using EigenTrust in our architecture will allow users to ask services only to reputable peers and exclude malicious peers. Starting from the current EigenTrust algorithm, we intend to address the following issues:

Two-ways rating. In its current form EigenTrust allows one-way rating only. In the systems we consider, we need a two-ways rating. Indeed, like in eBay, where both buyer and seller rate each other, we want that service providers and clients rate each other after every transaction. On the client side it is obvious that we want to know which services are reputable and which are malicious ones. On the service side it is also interesting to avoid malicious clients that try to make denial of services attacks or that try to corrupt the service by sending bad parameters.

Privilege good principals. In order to encourage principals to provide good services, we suggest privileging those with a high reputation. In case of a network overload, a reputable service will serve only reputable clients. In fact, the more a principal becomes reputable, the more it will deal with high-trusted peers.

Different trust values. The EigenTrust algorithm defines only one trust value for each peer. The authors claim that a peer that provides good files will also be good in providing trust values for other peers. In the case of our architecture, we prefer to compute different trust value: one for each available service, one for the behaviour of a principal when it acts as a client, and one indicating its reliability for trust computation of other peers.

Reputation Update. The EigenTrust algorithm implies that reputation values are all calculated together, since trust values are all closely linked and dependent of each other. However, we could consider a more flexible algorithm, still inspired by EigenTrust, that as well converges to the global emergent reputation, but not necessarily in one shot. The reputation value would converge slowly but the whole algorithm would not affect the efficiency of the system.

Distributed Architecture. Our SOA architecture is currently centralised. The Service Manager acts as a server that connects client entities with services. It is similar to Napster; clients ask the server for a specific file, and the server respond with the address of the peer that contains it. The main difficulty that we will have to face to obtain a completely decentralised architecture is the problem of peer discovery. Where should a peer connect in order to find a service? In many well-known P2P file sharing systems, like in Kazaa, the peers that have a high-speed connection are automatically designed as super-nodes. A super-node is a peer like another, but which adds a directory service. All other peers connect to the closest super-node in order to locate a specific file. If the super-node does not have it, it transmits the request to another super-node.

In our future distributed architecture, the centralised service manager will disappear. The directory functionality provided by the service manager will become a service like another. Every peer can therefore act as a service manager.

3.4 Example

The following small example shows how a group of computers can share a pool of printers through our envisioned infrastructure. Before interacting with each other computers and printers exchange their respective functional as well as non-functional capabilities, e.g. a printer claims that it is a postscript double-sided printer, and a computer asks to print a PDF file. After having interacted with a printer, the computer stores the observation related to its experience with the printer (works as expected, only one side, no impression at all, etc.). Depending on the outcome of the interaction, or if it has been requested to do so, the computer may want to share its knowledge with some of the other computers. It will then inform the others that the printer is not actually double-sided, but only single sided, or that the printer went out of toner, and is no longer available, or that one of the printers is faulty and has a random behaviour.

This example shows that: printers and computers can exchange information about their respective functional and non-functional behaviour; computers can exchange information among themselves about the printers and other computers state or actual capabilities (independently of their claimed functionality); the shared knowledge allows computers to efficiently use the remaining set of working printers (adaptation, resource management), as well as to correctly inform the user about the nearest well functioning printer. This example shows as well the validity of information. The faulty printer has a random behaviour, this is a long term valid information (information is not very accurate, but not volatile). However, if the printer has been able to print two minutes ago, we can almost be sure that it will be able to print in the next couple of minutes, but not necessarily later (information is accurate but highly volatile). This example raises also the question of the accuracy of a shared information. In the case of the printer, it claims that it can print, but actually it cannot. In the case of a computer, it can claim that the printer is out of order, but it may lie. In both cases, sharing knowledge about printers or other computers helps circumvent the problem, and adapt the individual as well as the collective behaviour to the environment.

4 State of the Art

Specification-Carrying Software. The notion of specification-carrying software is being investigate since several years at the Kestrel institute (Pavlovic, 2000; Anlauff et al., 2002). This idea has been proposed initially for software engineering concerns, essentially for: ensuring correct composition of software and realising correct evolution of software. Algebraic specifications and categorical diagrams are used for expressing the functionality, while coalgebraic transition systems are used to define the operational behaviour of components. The visions of this team include as well run-time generation of code from the specifications. Compared to these works, this paper proposes a “light” version where the behaviour of a component is not fully specified in all its operational details, but sufficiently in order to be used for correct self-assembly of software at run-time.

Meta-Ontologies. Meta-ontologies are algebra allowing definition of type theories, operations, and axioms. From that perspective, category theory (Johnson and Dampney, 2001), higher-order logics that define terms, operators, axioms, and provable or checkable theorems are meta-ontologies.

Current semantic Web services simply use information, expressed or communicated through languages such as RDF or OWL, as linking glue. However, the exchanged information is not yet used to allows full interoperation, or reactive behaviour to the semantics of information. Middleware addressing both semantic issues and intelligent interoperability are currently an open issue.

Trust-Based Management Systems. Trust management systems deal with security policies, credentials and trust relationships (e.g., issuers of credentials). Most trust-based management systems combine a higher-order logic with a proof brought by a requester that is checked at run-time. Those systems are essentially based on delegation, and serve to authenticate and give access control to a requester (Weeks, 2001). Usually the requester brings the proof that a trusted third entity asserts that it is trustable or it can be granted access. Those systems have been designed for static systems, where an untrusted client performs some access control request to some trusted server (Appel and Felten, 1999; Bauer et al., 2001). Similar systems for open distributed environment have also been realised, for instance Li et al. (1999) proposes a delegation logic including negative evidence, and delegation depth, as well as a proof of compliance for both parties involved in an interaction. The PolicyMaker system is a decentralised trust management systems (Balze et al., 1996) based on proof checking of credentials allowing entities to locally decide whether or not to accept credentials (without relying to a centralised certifying authority).

Tag-Based Models. Tags are markings attached to each entity composing the self-organising application (Hales and Edmonds, 2003). These markings comprise certain information on the entity, for example functionality and behaviour, and are observed by the other entities. In this case the interaction would occur on the basis of the observed tag. This is particularly useful if applied to interacting electronic mobile devices that do not know each other in advance. Whenever they enter the same space, for example a space where they can detect each other and observe the tags, they can decide on whether they can or cannot interact.

Smart labels/Smart Tags. Smart tagging systems are already being deployed for carrying or disseminating data in the fields of healthcare, environment, and user's entertainment. For instance, in the framework of data dissemination among fixed nodes, (Beaufour et al., 2002) propose a delivery mechanisms, based on the local exchange of data through smart tags carried by mobile users. Mobile users or mobile devices do not directly exchange smart-tags, they only disseminate data to fixed nodes when they are physically close to each other. Data information vehicled, by smart tags, is expressed as triples indicating the node being the source of the information, the information value, and a time indication corresponding to the information generation. Smart tags maintain, store, and update these information for all visited nodes. A Bluetooth implementation of these Smart Tags has been realised in the framework of a vending machine (Beaufour, 2002). In smart tagging systems, data remain structurally simple, and understandable by human beings, and does not actually serve as a basis for autonomous local decisions.

5 Conclusion

The model proposed here follows the separation into individual capabilities and social organisation mentioned by Minsky (Minsky, 1988). The exchange of functional and non-functional capabilities in our model corresponds to the diffusion of knowledge about the capabilities of individual principals. The use of trust and the exchange of recommendations adds a social layer on top of the interaction mechanism. Typical applications that can benefit from this technology include wireless cellular network routing, ambient intelligence systems (Ducatel et al., 2001), autonomic computing systems (Kephart and Chess, 2003), or access control systems.

In order to experiment this approach with mobile components, we foresee as well to combine our prototype with a positioning system currently deployed in our department.

Acknowledgements

This work is partly supported by the EU funded SECURE project (IST-2001-32486), and by the Swiss NSF grant 200020-105476/1.

References

M. Anlauff, D. Pavlovic, and D. R. Smith. Composition and refinement of evolving specifications. In *Proceedings of Workshop on Evolutionary Formal Software Development*, 2002.

A. W. Appel and E. W. Felten. Proof-carrying authentication. In *6th ACM Conference on Computer and Communications Security*, 1999.

M. Balze, J. Feigenbaum, and J. Lacy. Decentralized trust management. In *IEEE Conference on Security and Privacy*, 1996. URL

L. Bauer, M. A. Schneider, and E. W. Felten. A proof-carrying authorization system. Technical Report TR-638-01, Princeton University Computer Science, 2001.

A. Beaufour. Using Bluetooth-based Smart-Tags for Data Dissemination. In *Pervasive Computing 2002*, 2002.

A. Beaufour, M. Leopold, and P. Bonnet. Smart-tag based data dissemination. In *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'02)*, 2002.

V. Cahill and al. Using trust for secure collaboration in uncertain environments. *IEEE Pervasive Computing Magazine, special issue Dealing with Uncertainty*, 2(3):52–61, 2003.

M. Deriaz and G. Di Marzo Serugendo. Semantic service oriented architecture. Technical report, Centre Universitaire d'Informatique, University of Geneva, Switzerland, 2004.

K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J.-C. Burgelman. Scenarios for Ambient Intelligence in 2010. Technical report, Institute for Prospective Technological Studies, 2001.

D. Hales and B. Edmonds. Evolving Social Rationality for MAS using "Tags". In J. S. Rosenschein, T. Sandholm, M. Wooldridge, and M. Yokoo, editors, *Second International Joint Conference on Autonomous Agents and MultiAgent Systems*, pages 495–503. ACM Press, 2003.

M. Johnson and C. N. G. Dampney. On Category Theory as a (meta) Ontology for Information Systems Research. In *International Conference On Formal Ontology In Information Systems (FOIS'01)*, 2001.

S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *WWW2003, May 20-24, 2003, Budapest, Hungary*, 2003.

J. O. Kephart and D. M. Chess. The Vision of Autonomic Computing. *Computer*, 36(1):41–50, January 2003.

N. Li, J. Feigenbaum, and B. N. Grosof. A logic-based knowledge representation for authorization with delegation. In *12th IEEE Computer Security Foundations Workshop*, 1999.

M. Minsky. *La Société de l'Esprit*. InterEditions, 1988.

M. Oriol and G. Di Marzo Serugendo. A disconnected service architecture for unanticipated run-time evolution of code. *IEE Proceedings-Software, Special Issue on Unanticipated Software Evolution*, 2004.

D. Pavlovic. Towards semantics of self-adaptive software. In *Self-Adaptive Software: First International Workshop*, volume 1936 of *LNCS*, pages 50–65. Springer-Verlag, 2000.

S. Terzis, W. Wagealla, C. English, P. Nixon, and A. McGettrick. Deliverable 2.1: Preliminary trust formation model. Technical report, SECURE Project Deliverable, 2004.

S. Weeks. Understanding trust management systems. In *2001 IEEE Symposium on Security and Privacy*, 2001.

Towards a Simulation Tool for Evaluating Dynamic Reorganization of Agents Societies

Virginia Dignum* Frank Dignum* Vasco Furtado† Adriano Melo†
Liz Sonenberg‡
*Institute of Information and Computing Sciences
Utrecht University, 3508 TB Utrecht, The Netherlands
{virginia, dignum}@cs.uu.nl
†University of Fortaleza
Av Washigton Soares 1321, Fortaleza, CE, Brazil
{vasco@unifor.br, aanmelo@hotmail.com}
‡Department of Information Systems
University of Melbourne, Parkville 3010, Australia
l.sonenberg@unimelb.edu.au

Abstract

Reorganization of the structure of an organization is a crucial issue in multi-agent systems that operate in an open, dynamic environment. Currently, most coordination mechanisms are imposed upon the system at design time, and their modification implies the redesign of the system. However, autonomous agents must be able to evaluate and decide the most appropriate organization given the environment conditions. That is, there is a need for dynamic reorganization of coordination structures. In this paper, we propose a classification of reorganization types which considers two layers of reorganization: behavioral and structural. We further describe how simulations can help to determine whether and how reorganization should take place. Finally we present a simulation scenario that is used to evaluate the different reorganization forms.

1 INTRODUCTION

Establishing an organizational structure that specifies how agents in a system should work together helps the achievement of effective coordination in MAS (Barber and Martin, 2001). An organization-oriented MAS starts from the social dimension of the system, and is described in terms of organizational concepts such as roles (or functions, or positions), groups (or communities), tasks (or activities) and interaction protocols (or dialogue structure), thus on what relates the structure of an organization to the externally observable behavior of its agents.

Environments in which the MAS systems function are not static. Their characteristics can change, ranging from new communication channels to tasks that are no longer useful or are new. In such a changing environment, agents can disappear, be created or they can migrate. The organizational objectives can change, or operational behavior can evolve. Models for MAS must therefore not only cater for adaptive agents (Jennings et al., 1998) but also be able to describe dynamically adapting organizations to changes in the environment. Depending on the type of organization and on the perceived impact of the changes in the environment, adaptation is achieved by behavioral changes at agent level, modification of interaction agreements, or the adoption of a new social structure. Even though in most MAS, reorganizations are currently realized by re-engineering the system (i.e. external assessment and modification of a system), for a MAS to be truly autonomous, mechanisms for dynamic reorganization must be available. The concept of *dynamic adaptation* refers to modifications in structure and behavior of a MAS, such as adding, removing or substituting components, done while the system is running and without bringing it down (Valetto et al., 2001). Dynamic adaptation demands that systems can evaluate their own ”health” (i.e. success and other utility parameters) and take action to preserve or recover it, by performing suitable integration and reconfiguration actions. Reorganization of organizations should therefore allow both for changes of the operational behavior of the organization, such as admission or departure of agents, as well as for changes of the social structure of the society changes, that is, roles, relationships, norms or interactions.

In (Dignum et al., 2004), we discuss different types and motivations for reorganization and the consequences for MAS models of enabling dynamic reorganization at different complexity levels. Not every change in the environment or an agent will lead to an organizational change. But when and who will actually decide upon such a structural change?
When a decision is made to change the organization it should also be decided what and how the organization is changed. Are interaction patterns changed, do we change some roles, some constraints,...? Organizational success is brought about by the organization's ability to bring all its information and assets to bear, and the ability to recognize and take advantage of fleeting opportunities. In this sense, successful reorganization should lead to an increased utility of the system. That is, the reorganized instance should perform better in some sense than the original situation.

From the perspective of the individual agents, their participation in an organization also depends on utility factors. Utility is however appreciated differently from the perspectives of the society and of the agents. On the one hand, the organization will only admit an agent, if the overall utility of the society increases (Glasser and Morignot, 1997). On the other hand, assuming rational agents, the agent will only join an organization if its own utility increases.

In this paper, we will first describe a theoretical framework of the reorganization aspects discussed above. After that we will discuss how simulations can be used to discover some properties of the reorganization process. Finally we describe the first steps in this process, presenting a simulation environment in which reorganization can be studied.

2 SOCIAL ORGANIZATION

Many applications require a set of agents that are individually autonomous (in the sense that each cognitive agent determines its actions based on its own state and the state of the environment, without explicit external command), but corporately structured. As such, there is a growing recognition a combination of structure and autonomy is often necessary. More realistic models for the simulation of organizations should also be based on cognitive agents. In fact, greater cognitive realism in social simulations may make significant differences in terms of organizational performance. Sun and Naveeh (2004) present a study showing that different combinations of social structure and individual cognition level influence organizational performance.

2.1 Organizational utility

One of the main reasons for having organizations, is to achieve stability. Nevertheless, environment changes and natural system evolution (e.g. population changes), require the adaptation of organizational structures. Reorganization is the answer to change in the environment. As reorganization is contrary to stability, the question is then: under which conditions is it better to reorganize, knowing that stability will be (momentarily) diminished, and when to maintain stability, even if that means loss of response success. In order to answer this question, it is necessary to define the *utility* of an organization. Reorganization is therefore desirable if it leads to increased utility of the system. That is, the reorganized instance should perform better in some sense than the original situation.

Given the assumption of agent autonomy, it is also necessary to define agent utility, as each agent should, in principle, be able to determine whether a reorganization results in increased utility for the agent itself. Utility is thus evaluated differently from the perspectives of the society and of the agents.

Society Utility We define the utility of an organization based on organization properties:

- *Interaction success*: how often do interactions result in the desired aim.
- *Role success*: how often do enacting agents realize role goals.
- *Structure success*: how well are global objectives achieved in an organizational structure.

For example, a given combination of structure and population is said to be successful if the overall success of the organization is higher in that situation than for others. Society utility depends also on the cost of the reorganization. That is, any function to measure organization utility must take in account both the success of a given structure, and the cost of any change needed to achieve that structure from the current situation (Glasser and Morignot, 1997).

Agent Utility is different for each agent, taking in account issues such as its own goals, resource production and consumption. Basically, we can assume that rational agents will participate in a society only if, in their own perception, their individual utility increases. Furthermore, different social attitudes will result in different evaluations of individual utility. That is, the utility function of a social agent may take on account some measure of society utility, whereas for a selfish agent only individual concerns matter.

2.2 Organizational Change

Change is a result of observation of the environment. Making sense of a situation begins by identifying relevant patterns and access current response possibilities. Sense-making is however more than sharing information and identifying patterns. It involves the ability to generate options, predict outcomes and understand the effect of particular courses of action. Such sense-making activities require to keep some sort of system history, also across different role enactors. These are capabilities that few software agents are endowed with. Hence, enabling dynamic reorganization has consequences for the capabilities required from the agents involved. Therefore makes sense to, firstly, identify which organization type is most appropriate for a given situation, , secondly, what is then needed to adapt the current organization to the one with the highest utility, and, finally, what is required from the individual agents to enable them to realize the reorganization.

A characteristic of reorganization is *timeliness*, that is adequate response at the appropriate time (not to be confused with speed). This implies the need to assess when and how often, and at which level to change. When change occurs too often and too quickly, the predictability of the system will decrease, but too slow and too late changes result in rigidness of the system. Both situations are usually not desirable. The characteristic to aim at is *resiliency*, that is, flexible but durable and consistent with the (meta) norms and objectives of the organization. An interesting study presented in (Carley et al., 2002), explores the resiliency of organizations by studying their performance when key leaders were removed . Different domains will have different appreciations of timeliness and resiliency. For instance, in rescue operations, timeliness is often directly related to speedy response. That is, a quick, even if sub-optimal, adaptation will be preferred over the optimal solution if that one only arrives after it is too late (e.g the house has already burned down). On the other hand, in institutions (such as an university department), timeliness is often related to consensus. That is, the good time to change is when all parties are conscious of the need to change and agree on the changed model.

3 A TYPOLOGY OF REORGANIZATION

In early work in reorganization, restructuring was only possible in the initialization phase of the system. During the actual problem solving phase, the structure was fixed. Currently, most dynamic approaches to reorganization consider only the behavioral aspects, that is reorganization only affects the current population of agents in the system, both at the social (i.e. interactions and relationships) (Carley and Gasser, 1999), as well as individual level (Hannebauer, 2002). Existing implementations of organizational adaptation include approaches based on load balancing or dynamic task allocation. The later is often the case in organizational self-design in emergent systems that, for example, include composition and decomposition primitives that allow for dynamic variation of the organizational structure (macro-architecture) while the system population (micro-architecture) remains the same (So and Durfee). Another common approach is dynamic participation. In this case, agent interaction with the organization is modelled as the enactment of some roles, and adaptation occurs as agent move in and out of those roles (Dignum, 2004; Glasser and Morignot, 1997; Tambe, 1997). However, few of these systems allow agents to change the problem-solving framework of the system itself (Barber and Martin, 2001).

Based on the above considerations, we identify the following reorganization situations:

Behavioral change: Change at behavior level, that is, organizational structure remains the same, but behavior of agents enacting organizational roles change. Examples are when agents join or leave the society, when they change between existing roles, or when their characteristics change (e,g. more or less consumption or production of some resources). It does not affect future enactments and therefore there is no need for organizational memory.

Structural change: Aims at accommodating long-term changes, such as new situations or objectives. Structural change influences the behavior of the current but also of future society instantiations. Examples of structural change are adding, deleting or modifying structural elements (e.g. roles, dependencies, norms, ontologies, communication primitives) Change at social level implies a need for society level learning. That is, by keeping an organizational memory, the society itself can reflect on the difference between desired and actual behavior and decide on social level changes (roles, norms, etc.).

Another perspective on reorganization, concerns the ways the reorganization decision is taken. Considerable work has been done analyzing the advantages and disadvantages of centralized and distributed problem-solving structures. In centralized situations, decisions are taken by one role in the organization. It corresponds to a master/slave relationship between agents acting at different levels of autonomy. Roles empowered with decision-making authority, *direct* change of other roles. In distributed decision-making situations that (all) roles are collectively responsible for a change decision. Changes are thus achieved by collaboration or consensus. In (Barber and Martin, 2001) three types of decision-making styles are identified, that relate to centralized and distributed decision-making situations:

- *Command-driven*: the agent does not make any decisions on how to pursue its (role) goals, and some other agent has authority over it (child in a hierarchical relation)

- *True consensus*: Agent works as a team member, sharing decision making control equally with other agents. (network relation)

- *Locally autonomous/master*: The agent makes decisions alone and may or not have control over other agents (parent or root in a hierarchical relation).

Related to the above, is work on the application of the military notions of Command, Control and Communications (C3) to MAS focuses on the authority to effect changes at different levels (Tidhar and Sonenberg, 2003). *Command* refers to the authority and responsibility to determine the objectives of the organization and update the social structure of the organization accordingly. *Control* refers to the authority to specify and modify detailed plans for achieving objectives, that is, the authority to modify interactions and behavior. *Communications* refer to sharing information about the environment, the state of the organization, the state of the achievement of objectives, and the state of execution of the plans. Figure 1 depicts the relations between the different perspectives on reorganization.

In directive situations, agents enacting directive roles (or *directors*), must be able to monitor and evaluate the overall behavior of the system, according to some success factors and determine what adaptation is required. The need for communications is reduced

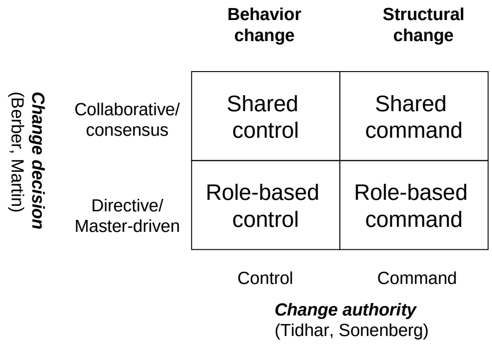


Figure 1: *Dimensions of change.*

as the directive agent forms its decisions independently from the information it receives from others. The director communicates, or otherwise enforces, changes in interaction or behavior to the other agents, but can only assume that the others will in reality realize those changes (because it cannot access internal behavior and motivations leading other agents' actions).

In collaborative situations, all agents need high meta reasoning and communicative capabilities in order to assess changed situation, communicate with others about its observations, and negotiate how the group should adapt to it. The need for communications is high as change decisions can only be achieved by negotiation between all agents, which form their own decisions based on their own evaluation of the environment, possibly benefiting communications with the others.

4 Objectives for Simulation of Reorganization

In the previous sections we have brought forward a number of aspects and ideas that play a role in the reorganization of MAS. In this section we will explain how we use simulations to substantiate the theory.
First of all we have to point out that a theory on reorganization brings together a number of aspects on different levels of the MAS that cannot be studied all in the same simulation. Therefore we have to divide the process in a number of steps, each building on the previous one. The main complicating factor is that we assume that the behavior of an agent in a MAS does not only depend on its own internal state and the state of the environment, but that it also depends on the organizational structure of the MAS in which it operates. Important point is that we cannot assume the organization to be just another part of the environ-

ment, because it cannot be changed in the same way as other parts of the environment by a single agent (we recognize that this is not a very strict distinction, but the important part is that the organization does have a special status when we take into account explicit reorganizations).

The first step in the exploration of the reorganization process is thus to find out exactly what is the influence of the organization form on the behavior of the MAS in a certain environment. In order to make this more precise we have to indicate which are the elements of the organizational form that we consider. Without claiming completeness, we consider the following aspects to be the most important ones:

- The type of goal of the organization. Is it a very simple, unrestrictive goal or a hard to achieve, very limiting goal.
- Which are the roles to be distinguished. I.e. how are the organizational goals divided over roles. In the extreme cases all agents play the same role or all play a different role.
- Related to the previous point is how the roles are instantiated with agents. How many agents play the same role.
- The interaction between the agents playing roles. This concerns both the interaction patterns (communication protocols) as well as role dependencies (does a role have power over resources, task allocation, etc. and can thus steer other roles).

Given a certain environment and agents with fixed capabilities we can use simulations with differently organized MAS to find out which of the organizations performs ”best” in such an environment. In such a way it will be possible to make a match between organizational form and type of environment. The research question here is thus ”Which type of organization structure performs best given a certain environment and organizational objectives?”

The next step in the exploration process is about the actual reorganization itself. In this step we want to find out how an organization should be reorganized from one form to another to best suit an environment that changed (drastically). So, in this step we actually explore the possibilities for reorganization given in the previous section. Aspects that will be important here are how quick an organization can react to a changing environment and how big are the ”costs” of the reorganization. If a certain mechanism takes too much time the MAS might not recover in time to survive. On the other hand, the costs of a reorganization can be so big that it is better to quit the organization and start all over from scratch. The aim of this step is thus to evaluate the different possibilities for changing into a more adequate structure given a change of environment characteristics.

In the previous we assumed that all agents within the organization somehow will know that the environment changed and a certain type of reorganization has to be performed. In the last step we will look at cases where certain agents will discover that the environment changes and the reorganization has to be initiated through communication. This is a very typical scenario for crisis management in which teams of agents have to react to changing circumstances that are detected by one or more members of the team. Especially in this last step we will look at the reasoning and communication capabilities of the agents in the MAS and the influence this has on the reorganization possibilities.

In summary, the three steps in the reorganization simulation process are as follows:

1. Identify the match of organizational structure or behavior to environment characteristics
2. Reorganization of system to adapt to (drastic) changes. Also, evaluate the advantages and disadvantages of structural and behavioral change, role-directed or collaborative.
3. Investigate the communicative requirements to reason about change. Also, evaluate the influence of reasoning with limited knowledge.

5 Initial Simulation Setup

As described in the previous section, the aim of our research is to develop a simulation tool that enables the study of the effects of reorganization strategies on the performance of societies consisting of multiple agents. We are interested in investigating both the properties of systems that exhibit reorganization possibilities and the degree of complexity necessary to build agents that are able to reason about social reorganization. In order to simulate real-life organizations it is first necessary to find out which are the most important parameters and measurements. For this purpose we started with a simple artificial organization in order to keep the complexity in hand and move slowly to more realistic situations. The development of the simulation game, VILLA, follows the three steps described in the previous section, and was further designed to meet the following requirements:

- The system must be simple enough to enable empirical evaluation of the results.
- The system must be complex enough to emulate situations where reorganization really matters.

VILLA simulates a community inhabited by number of Creatures, divided into three groups: the Gatherers, the Hunters, and the Others. The unique goal of the community is to survive. All creatures must eat in order to survive. When creatures don't eat, their health decreases, until health is 0, when they die. Gatherers and Hunters are responsible to keep the food stack supplied. Gatherers and Hunters should eat more than Others to allow for the effort of collect food. Furthermore, the health of Gatherers and Hunters determines how much food they can collect. That is, the healthier a Hunter or Gatherer is the more food it can collect. However, food collection is not always guaranteed and Gatherers or Hunters may only sporadically be successful. The probability of success of Gatherers is higher than that of Hunters. On the other hand, when successful, Hunters can collect more food than Gatherers. Gatherers find food on their own but Hunters must hunt in groups (two or more). Therefore, Hunters must be able to move in order to find other Hunters with whom they can hunt. The hunting capability increases with the size of the group. Other Creatures can be seen as the elderly and children of the society, they only eat and are not in state of contributing to the food collection effort. Formally, a VILLA community can be defined as:

$Villa = \{C, G, H, FS, F0, E, T, m_E, M_E, R\}$, where:

- $C = \{c : c = (\{health, food-intake\}, \{eat\}, \{O(eat|food > 0)\})\}$, are the creatures. The obligation indicates that all creatures must eat if there is food available.
- $G \subseteq C, G = \{g : g = (\{health, food-intake, gather-power, gather-probability\}, \{eat, gather\}, \{t < E, O(g, gather(g, t))\}\}$, is the subset of Gatherers. The obligation indicates that gatherers are obliged to gather food in each run. How much food is gathered is a function of its gather-power and the gather probability.
- $H \subseteq C, H \cap G = \oslash, H = \{h : h = (\{health, food-intake, hunt-power, position\}, \{eat, gather, observe, move\}, \{t < E, O(h, hunt \vee move))\}\}$, is the subset of creatures that can hunt food. The obligation indicates that hunters are obliged either to hunt or to move in each run. How much food is hunted is a function of the number of Hunters in a group, and the combined gather-power and gather probability.
- $FS = (\{food\}, \{\}, \{\})$ is the food stack agent, describing the amount of food available at any moment
- $F_0 \in Int$, is the value of the initial food stack
- $E \in Int$, is the number of runs
- $T \in Int$, is the number of ticks per run
- $m_E \in Int, m_E =< num(C)$, minimal number of creatures at time E
- $M_E \in Int$, maximal amount of food at time
- $R = \{r1, r2, r3, \}$ are the society rules, defined as follows

R1 $\forall c \in C, \forall i \leq E, eat(c, i) \rightarrow food(i) = food(i-1) - food-intake(c)$

R2 $\forall g \in G, \forall i \leq E, gather(g, i) \rightarrow food(i) = food(i - 1) + gather-power(g, t) \times gather-probability(g, t)$

R3 $hunt-group(p) = h_1, , h_n \leftrightarrow \forall h_x, h_y \in p, adjacent-position(h_x, h_y)$

R4 $\forall p : hunt-group, \forall i \leq E, hunt(p, i) \rightarrow food(i) = food(i - 1) + hunters(p) \times ((hunt-power(h, t) \times hunt-probability(h, t))$

R5 $\forall c \in C, (food(i) = 0) \rightarrow eat(c, i)$

R6 $\forall c \in C, noteat(c, i) \rightarrow health(c, i) = health(c, i - 1) - 1$

R7 $\forall c \in C, health(c, i) = 0 \rightarrow dead(c)$

R8 $\forall g \in G, \forall i \leq E, gather-power(g, i) = f(health(g, i))$
(i.e. gather-power is a function of health)

R9 $\forall h \in H, \forall i \leq E, hunt$
$int - power(h, i) = f(health(h, i))$
(i.e. hunt-power is a function of health)

R10 $\forall h \in H, \forall i \leq E, move(h, i) \rightarrow position(h, i) \neq position(h, i - 1)$

R11 $\forall c \in C, dead(c) \rightarrow num(C) = num(C) - 1$

R12 $success(village, R) \rightarrow num(C, R) \geq m_R \wedge food(R) =< M_R$

The VILLA simulation game consists of a fixed number of runs. During each run, Gatherers and Hunters will gather food, and as many Creatures will eat as the food stack allows. Each run consists of a number of 'ticks'. Each agent can use each tick either to act or to reason (not both simultaneously). The objective is to have as many as possible creatures surviving at as low possible cost. [1] We have implemented the VILLA simulation game using the RePast simulation environment (Collier, 2003).

5.1 Simulation without reorganization

The specification above describes the basic simulation setting. In this simple version without reorganization, simulation starts with a fixed number of creatures of the three groups and a initial amount of available food (possibly 0). In each step of the simulation, all Creatures eat, Gatherers and Hunters try to catch some food to replenish the common food stack. Furthermore, Hunters need to move around the field in order to became adjacent to other hunters and therefore be able to hunt. All other agents (Gatherers and Others) either gather food and/or eat in their own block.

Figure 2 shows the initial settings of the simulation. Since Hunters can only start hunting after they have found at least another hunter, it is easy to see that in the first runs of the simulation only Gatherers are able of providing food to the community's stack, while all creatures are eating. Without reorganization, the chances of survival of the community are dependent on the initial food stack and on the probability of Gatherers to find food. In this situation, the community needs 40 units of food per step, and if only the Gatherers are collecting food in average only 18 units are collected per step [2]. This setting is thus an example of a organization structure with low utility given the aim of survival of as many Creatures as possible, By setting up many different possible organizational settings (e.g. varying number of Creatures, Gatherers and Hunters, collect probabilities and collect power, and initial food stack) we can empirically evaluate which organization is more successful given an environment situation.

In the example above, reorganization decisions should lead to the determination that if the food stack decreases below a certain amount, then, for example,

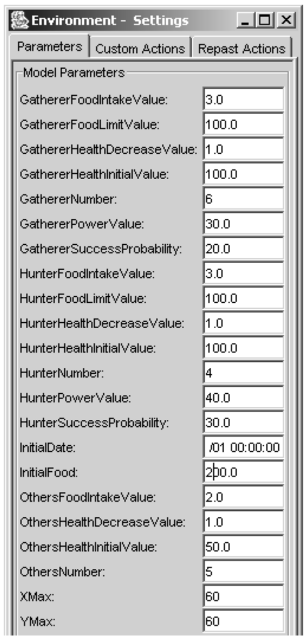


Figure 2: *Initial simulation settings.*

either the Gatherers should be able to gather more food (behavioral reorganization) or some of the Others or of the Hunters should be given Gatherer capabilities in order to increase the number of Gatherers (structural reorganization). In the following section, we describe how we have extended the simulation environment to simulate such reorganization strategies.

5.2 Reorganization Simulation

In the VILLA simulation scenario, the utility of the organization is described by the success of the community to survive. That is, a successful VILLA community is that which makes possible for as many as possible creatures to survive with as high as possible health. In order to be successful, communities must make sure that at any step there is enough food in the common food stack to feed the whole group. This can be influenced in several ways, e.g., either more food is collected (by augmenting the power of collection of Hunters and/or Gatherers, or by having more creatures hunting and/or gathering) or less food in consumed (in which case health still decreases but slower than when there is no food at all) Our objective is to use the reorganization environment described above to implement the 4 reorganization strategies described in section 3, as follows:

1. **VILLA1 - Role-based control**: a new role (the community Head) is introduced that can evaluate the overall utility of the society at any time

[1] In a possible future extension, the success probability of hunt-groups can be made to increase/decrease in function of the individual probabilities (good hunters together have more chance than bad hunters together).

[2] Since hunters must hunt in groups and thus first have to find each other, in the beginning hunters will not be collecting any food.

and decide on behavior alterations for the next run (that is, food-intake and gather-power can be changed, number of creatures, and gatherers remains fixed)

2. **VILLA2 - Role-based command**: the role Head is introduced, as VILLA1, which can decide, based on its evaluation of the society utility, to increase or decrease the number of hunters and/or gatherers, by giving some of the other creatures Hunter or Gatherer capabilities in order to increase the number of Hunters and/or Gatherers.

3. **VILLA3 - Shared control**: Gatherers, Hunters and Others must all be able to evaluate the overall health of the society and communicate their solution to the others. A agreement strategy must be chosen using a (fixed) negotiation strategy (i.e. majority, unanimity). Decisions involve behavior changes (that is, food-intake and gather-power)

4. **VILLA4 - Shared command**: as VILLA3 all roles must be able to evaluate the society utility and achieve by common agreement a reorganization decision, involving structural change, that is, about increase or decrease the number of Hunters and/or Gatherers, by changing the capabilities of other creatures.

The current version of the simulation environment enables the user first to determine the change authority (shared or role-based) and then the focus of the reorganization (behavior or structure). For role-based reorganization strategies a new role is added to the community, that of Head, which is responsible to reason about the performance of the community and implement the required reorganization actions. In shared reorganization strategies, all roles must be extended to incorporate reasoning about the performance and the capabilities to modify the community. The current version of the tool supports the reorganization strategies VILLA1 and VILLA2. The shared reorganization options VILLA3 and VILLA4 are currently under construction.

In the case of a behavior-based reorganization, the user can describe which parameters should trigger the reasoning (e.g health or food stack are below a certain value) and what changes of behavior should be triggered (e.g increase collect power, decrease food intake). The reorganization settings window for this case is depicted in figure 3. In the same way, the user can also determine the triggers and effects (e.g increase/decrease the number of Gatherers or Hunters)

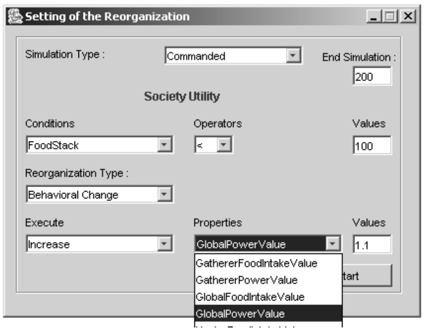


Figure 3: *Parameters for behavior-based reorganization.*

of a structure-based reorganization simulation. The reorganization settings window for this case is described in figure 4.

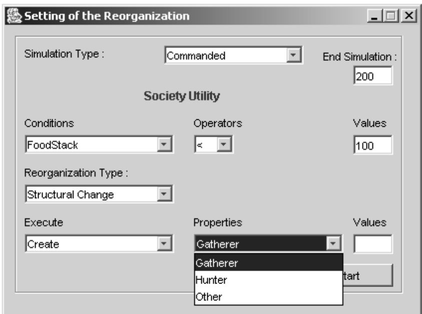


Figure 4: *Parameters for structure-based reorganization.*

In total, the simulation tool will support different reorganization strategies. The following, are a few possibilities we consider for the reasoning capabilities on the different versions. In the case of role-based decision situations (VILLA1, VILLA2), different reasoning strategies are possible, depending on what the Head can observe. If the Head has total information and is endowed of algorithms that determine the optimal organization structure given a certain environment, the Head can achieve optimal decision. However, in more realistic situations, the Head has neither complete information nor complete knowledge. In shared decision situations (VILLA3, VILLA4) all agents have only knowledge about themselves. Different versions will include co-operative agents (comply to change requests from others) or uncooperative ones, or mixed. In structural reorganization strategies, Others can be asked to become Hunters or Gatherers, Hunters can become Gatherers (e.g. if hunt-probability very low, or unable to join a hunt group), and Gatherers can become

Hunters (e.g. if adjacent to a high probability hunt-group).

We are currently implementing the settings to enable the above experimentations. In this work we concentrate on the effects of the reorganization strategy, in terms of effectiveness (how well does the decision achieve its aims), complexity (both of the reasoning process of agents and of the communication needs), and timeliness (how long does it take to reach a reorganization decision).

5.3 Evaluation of the VILLA Environment

We are currently setting up the empirical experimentation that will allow for the rigorous evaluation of the different reorganization strategies described above, and how they compare to the situation where no reorganization occurs. No statistical significant results are as yet available, but we already present a few examples of simulation runs that show the different behaviors related to the reorganization strategy chosen. All examples have a length of 200 runs and start from the same initial settings: 17 creatures (6 Gatherers, 6 Hunters, 5 Others) with 50% initial health; Gatherers have a success probability of 9% with gather power of 20 and Hunters have a success probability of 10% with gather power of 30; total food needed in each run is 58 4 units for each gatherer or hunter and 2 units for others; initial food stack is 500 units. Each Gath-

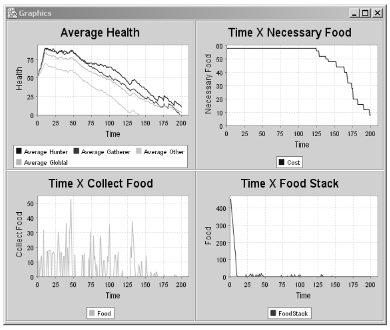


Figure 5: *Simulation with no reorganization.*

erer will collect 2,7 units in average and each Hunter 3 units (increased by the size of the hunting group). While Hunters are not hunting, the average collected food is thus 16.4 units. That is, even if Hunters start hunting, the community will most likely have trouble collecting enough food to keep all creatures healthy and alive. Figure 5 refers to the simulation with no reorganization. As expected, this community was not able to keep all creatures alive, and after consuming the initial food stack, they were hardly able to keep any food reserves. All creatures were dead by the end of the 200 runs.

The first reorganization example concerns a behavioral reorganization. In this case, depicted in figure 6, when the food stack drops below 250 units, the Head will increase the gather power of Gatherers by 1. In this way, the community manages to keep healthy and maintain food reserves. However, because food stack stayed under 250 for many runs, the gather power increased from the initial 20 to almost 100, which can be argued to be not very realistic. Figures 7 and 8 re-

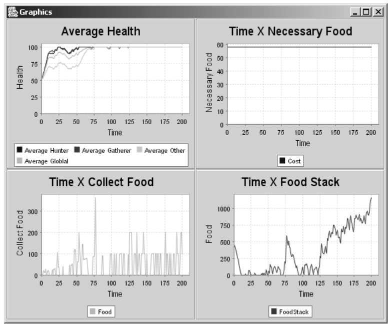


Figure 6: *Behavioral reorganization: Gather power increases by 1 if food stack reaches below 250.*

fer to structural reorganization strategies. In the simulation depicted in figure 7, if the food stack drops below 250 units, then a Gatherer was added (that is, an Other creature was given Gatherer capabilities), while in the simulation depicted in figure 7 a Hunter was added. In both cases food needs per run increase to 68 units, due to the fact that collecting creatures need more food that Others. In the case Gatherers were added, the average collecting power of all 11 gatherers is 30 and therefore not enough to keep the community alive, but still better in average than the case of no reorganization. In the second case, Hunters were added. Because there are more Hunters in the field, the probability they find each other increases, as in the case depicted. Once Hunters start collecting food, because of their larger power and higher collecting probability, in average more food will be collected and as such the community survives. In situations were Hunters do not manage to find each other, the behavior of the simulations tends to resemble the no reorganization case.

6 CONCLUSIONS

Reorganization of the structure of an organization is a crucial issue in multi-agent systems that operate in

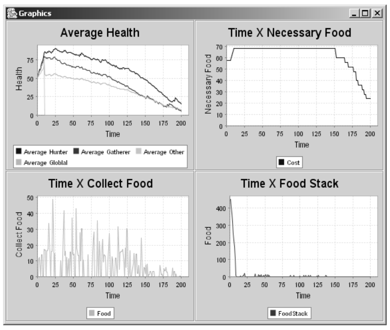


Figure 7: *Structural reorganization: Gatherer added if food stack reaches below 250.*

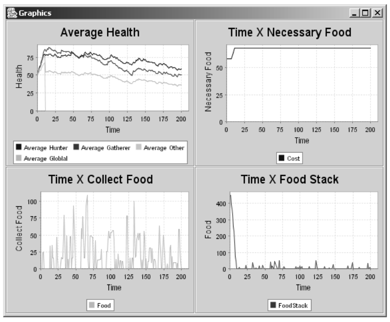


Figure 8: *Structural reorganization: Hunter added if food stack reaches below 250.*

an open, dynamic environment. In this paper, we presented a classification of reorganization types which considers two layers of reorganization: behavioral and structural; and described how simulations can help to determine whether and how reorganization should take place. Finally, we presented current work on the development of a simulation scenario that is used to evaluate the different reorganization forms.

Our current research on the development of a simulation tool for reorganization experimentation will enable to identify conditions and requirements for change, ways to incorporate changes in (running) systems, how to determine when and what change is needed, and how to communicate about changes. We are setting up empirical experimentations to this effect. Another important future research direction (following the simulation work), is the development of conceptual formal models that enable the specification of dynamic reorganization of agent societies.

References

K. S. Barber and C. E. Martin. Dynamic reorganization of decision-making groups. In *Proceedings of the 5th Autonomous Agents*, 2001.

K. Carley and L. Gasser. Computational organization theory. In G. Weiss, editor, *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, pages 299–330. The MIT Press, 1999.

K. Carley, J. Lee, and D. Krackhardt. Destabilizing networks. *Connections*, 24(3):79–92, 2002.

N. Collier. Repast: An extensible framework for agent simulation. In *http:repast.sourceforge.net*, 2003.

V. Dignum. *A Model for Organizational Interaction: based on Agents, founded in Logic*. Utrecht University, 2004. PhD Thesis.

V. Dignum, F. Dignum, and L. Sonenberg. Towards dynamic organization of agent societies. In G. Vouros, editor, *Workshop on Coordination in Emergent Agent Societies, ECAI 2004*, pages 70–78, 2004.

N. Glasser and P. Morignot. The reorganization of societies of autonomous agents. In *MAAMAW*, pages 98–111, 1997.

M. Hannebauer. *Autonomous Dynamic Reconfiguration in Multi-Agent Systems*, volume 2427 of *LNAI*. Springer-Verlag, 2002.

N. R. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. *Journal of Autonomous Agents and Multi-Agent Systems*, 1(1):7–38, 1998.

Y. So and E. Durfee. An organizational self-design model for organizational change.

R. Sun and I. Naveeh. Simulating organizational decision-making usinga cognitively realistic model. *Journal of Artificial Societies and Social Simulation*, 7(3), 2004.

M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, (7):83–124, 1997.

G. Tidhar and L. Sonenberg. Engineering organization-oriented systems. In H. Hexmoor, editor, *Proc. of Workshop on Autonomy, Delegation and Control: From Inter-Agent to Organizations and Institutions*, AAMAS, 2003.

G. Valetto, G. Kaiser, and Gaurav S. Kc. A mobile agent approach to process-based dynamic adaptation of complex software systems. In *8th European Workshop on Software Process Technology*, pages 102–116, 2001.

Two-sides of Emergence in Participatory Simulations

Paul Guyot[⋆]
[⋆]LIP6, boîte 169
Université Pierre et Marie Curie - Paris VI
4, place Jussieu, F-75252 Paris Cedex 05
paul.guyot@lip6.fr

Alexis Drogoul[†]
[†]LIP6, boîte 169
Université Pierre et Marie Curie - Paris VI
4, place Jussieu, F-75252 Paris Cedex 05
alexis.drogoul@lip6.fr

Abstract

Starting from an agent-based model of the coffee market in the state of Veracruz, we conducted participatory simulation experiments where human players were given the roles of reactive agents. The simulations were tuned to favor the apparition of coalitions among coffee producers. In addition to the expected coalitions, we witnessed another kind of emergence: roles were specialized with the apparition of traders among the coffee producers. Drawing from this first-hand experience, we came to consider participatory simulations as a way to create multi-agent systems where humans improve problem solving capabilities of the system.

1 Introduction

In this paper, we describe the emergence of behaviors within a participatory simulation as a way to create multi-agent systems where humans improve problem solving capabilities of the system.

Whether at work on multi-agent simulations or multi-agent systems, the computer scientist specializing in complex systems tries to produce an emergent behavior. Multi-agent simulations can be conceived as an attempt to reproduce an emergent behavior of a target system and can be used by a domain expert to determine the conditions of the emergence of this behavior. Multi-agent systems are fairly complex systems designed by the computer scientist to solve a problem, often using emergent properties of these systems.

The SimCafé experiments, part of a LAFMI[1]-funded project, were conducted in Xalapa, Veracruz, within the Laboratorio Nacional de Informática Avanzada (LANIA). These experiments were participatory simulations inspired by an agent-based approach where the agents' control architectures were performed by human players.

In a first part, we will describe the experiments as a multi-agent simulation approach. We will then interpret it as a distributed problem solver and present the roles that emerged. Finally, we will draw lessons from the participatory approach and explain how emergence in our experiments is different from what can be observed in other approaches.

[1]http://lafmi.imag.fr/

2 SimCafé as a multi-agent simulation

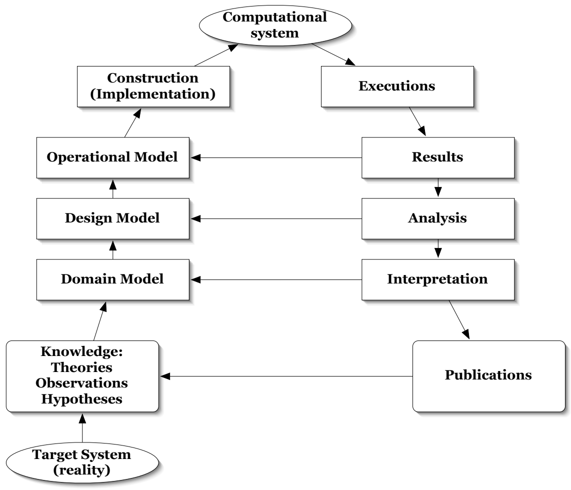


Figure 1: Design of multi-agent simulations (from Vanbergue (2003) and Drogoul et al. (2002))

The design process of the SimCafé experiments was very similar to the design of multi-agent simulations (figure 1). We started from a domain model of coffee production and coffee market. Then we tailored it as a design model for the very purpose of the participatory simulation. Finally, we implemented it as an operational model by building a distributed participatory simulation tool.

2.1 Coffee production in Veracruz

We started from hypotheses and theories about the coffee market of the state of Veracruz. The domain model, elaborated by our Mexican partners from the LANIA, covered both coffee production and coffee market.

Coffee production is a four-step process:

- The fruit, called “el café cereza”, is cropped once a year on coffee trees.
- After picking, the beans are transformed into pergamino in factories called “beneficio húmedo”.
- Then, they are transformed in “café oro” or “verde” in factories called “beneficio seco”.
- Finally, coffee is torrefied

The most critical step, according to local producers we met, is the transformation of the beans into pergamino. It takes three days.

In the state of Veracruz, according to local government data[2], there are 67,500 coffee producers for 3,000 full time jobs. Most of the producers only are part-time tree growers. Owners of beneficios need to buy cereza or pergamino and sell transformed coffee, either pergamino or oro coffee depending on the beneficio they own. Very few producers control the whole process, owning lands with trees, beneficios and torrefying the coffee themselves. Multinational companies such as Nestlé buy the fruits before they are cropped and process them themselves, but a lot of the production of beans is bought from beneficio owners.

Buyers make offers to beneficio owners and they usually have one week to accept and fulfill the offer. During this period, domain experts we worked with thought that the producers could form coalitions in order to fulfill the offer. Assuredly, the offer sometimes exceed the amount of coffee producers currently have. However, while alianzas (cooperative) exist, there is no sound evidence of the existence of other forms of coalitions among producers. For various reasons, producers refuse to talk about any coalition behavior they may have.

2.2 Coalitions in the coffee market

Our Mexican partners defined three types of coalitions that may happen within the coffee market of Veracruz.

[2] http://www.veracruz.gob.mx/

In the first kind of coalition (figure 2), a producer initiates negotiations with other producers. Some of the producers he contacts may also have had received the same offer from the buyer.

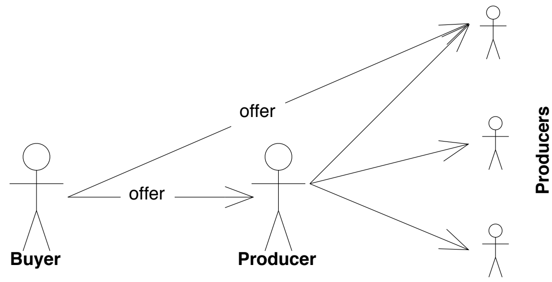


Figure 2: Direct negotiation

Cooperative of producers form the second kind of coalitions (figure 3). Cooperatives (called Sociedad or Alianza) gather producers who share risks, information and benefits. They fulfill offers together.

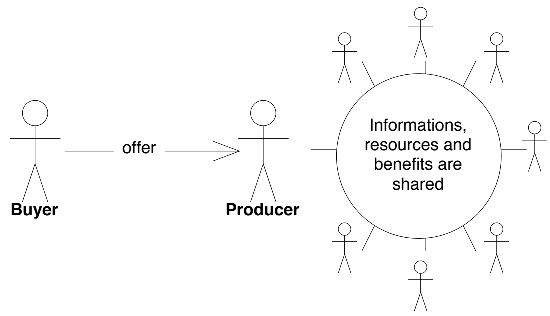


Figure 3: Coalition as a cooperative (Sociedad/Alianza)

The third kind of coalitions determined by our Mexican partners is inspired from the Contract Net protocol (Smith, 1980). Instead of talking directly to other producers, the initiator sends a broadcast offer to many producers who may then accept or reject the offer.

2.3 A model for participatory simulations

The goal of the domain experts was to determine whether coalitions occurred and to validate their model of coalition formation. The domain model needed to be transformed in order to achieve this goal.

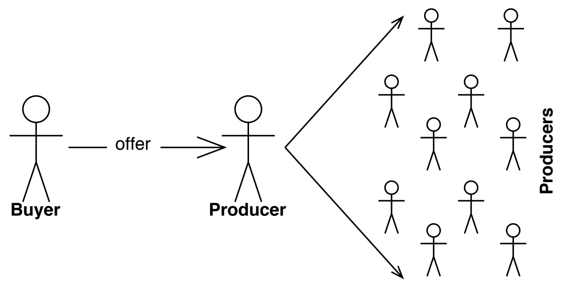


Figure 4: Coalition with a broadcast offer from a producer

Agents were sorted into two classes. Coffee producers in our simulations are beneficio húmedo owners. They can buy cereza at the market price and transform it into pergamino. The other class consists in pergamino buyers.

In the original model, coalitions are always initiated by the coffee buyer. Coffee producers can be described as reactive agents. However, the operational model allowed agents to communicate and to exchange coffee and money without any initial offer from a buyer. We decided to break the operations into smaller primitives and give as much freedom to the players as possible. The model needed to be relaxed in order to validate the hypothesis of the existence of coalitions.

We also had to specify what information agents would have. The coffee buyers are omniscient because their roles, assumed by the animators of the simulations, consist in favoring the apparition of coalitions. Coffee producers, on the other hand, only know the size of other producers' beneficios. The size determine the amount of pergamino that can be produced during a three days period.

Offers of a buyer to one or several producers consisted in a contract for a given quantity at a given price with a given deadline. Because the buyers were played by animators, the offers could not be negotiated. Producers could refuse an offer at any time. To accept an offer, they had to react before the deadline and be able to fulfill it, i.e. they had to own the required amount of pergamino. The first producer to accept an offer won the contract and other producers could not accept it afterwards. Recipients of an offer were aware of who else received it.

2.4 Emergence of coalitions

The experiments consisted in three simulations about an hour and a half long each with a single buyer. During the last experiment, there were two coalitions to satisfy an offer, a third was nearly completed but the offer was accepted by another producer. Table 1 lists the offers that were accepted and fulfilled by producers during this experiment. The first three columns describe the offer (quantity, price and time) and the last two columns describe who won the offer and how it was resolved.

Table 1: Resolutions during the third experiment

Amt	Price	Time	Agent	Resolution
200	15	200	Hector	direct
50	15	40	Abelardo	direct
500	20	200	Hector	coalition (bought 470 from others)
30	10	40	Abelardo	direct
100	15	40	Francisco	direct
25	50	40	Clemente	direct
50	10	40	Benjamin	direct
10	20	40	Daniel	direct (Francisco was preparing a coalition)
120	10	50	Abelardo	direct
800	25	250	Hector	coalition (bought 480)

The time is in hours (of simulation) and the amount in bags. Hector bought pergamino from other players in both cases of coalition. In the first case, he bought them from Francisco (290), Emiliano (80) and Abelardo (100) and in the second case from Francisco (10), Emiliano (160+80) and from Ignacio (130+100).

3 SimCafé as a multi-agent system

While the SimCafé experiments can be viewed as multi-agent simulations trying to reproduce a real target system, the coffee market in the state of Veracruz, it can also be seen as a multi-agent system designed to solve the problem of the fulfillment of buyer offers. Within this frame, we can reinterpret the emergence of coalition as a specialization of roles.

3.1 Distributed Problem Solver

The SimCafé experiments can be considered as a distributed problem solver. As such, the system formed by the players and their interface to the simulation is very similar to a multi-agent system. Traditionally, agents are composed of sensors, effectors and a control architecture. In our case, sensors and effectors consist in the SimCafé interface (Figure 5). Effectors are broken into small primitives within the domain model: agents can send money or pergamino without any counterpart. They can also send messages to other agents. The human participants play the role of the control architectures.

Figure 5: The SimCafé interface

The problem can be solved in a distributed way because offers sent to players can be fulfilled with cooperation among the producers. Producers could accept offers either by producing coffee themselves, provided that the time permitted it, or by buying coffee from other producers or by combining both. A proper choice of the deadline allowed the omniscient buyer to cast offers that could only be solved with cooperation of the producers. Players were not informed of this bias and the first offers actually could be solved directly. Consequently, agents, played by human players, were conducted to form coalitions without being intrinsically designed or required to.

3.2 Emergence of roles

While the emergence of coalitions was not a surprise, a very interesting outcome of the simulation consisted in the analysis of the actual roles of the agents. On the contrary to what the initial model defined, agents were not reactive but pro-active since they were controlled by human players who could communicate and exchange coffee without any initial offer from the buyer.

Players had exactly the same information. The only difference consisted in the size of their beneficios, represented by a little gauge under the house of each player (figure 5). It ranged from 15 for Ignacio, meaning that Ignacio was able to produce 15 bags every three days, to 100 for Francisco.

While cooperative were not expected because there was no risk to share, some players apparently tried to ally in order to fulfill the offer before other players. We witnessed several attempts of alliances. For example, Ignacio and Emiliano tried to ally each other during the last offer. In the end, they both separately sold coffee to Hector who won the offer.

The most striking particular behavior that emerged was Abelardo's. Abelardo is not an important producer because his beneficio is just the second in size with a throughput of 30 bags every three days. Instead, in addition to producing coffee himself, he became a trader. He has been broadcasting several messages to buy and sell large quantities of bags at a given price, and he often found sellers and buyers. During the third experiment, seeing the offer of 800 bags, he sent two messages to all other players saying that, in order to fulfill the current offer, he was selling 200 bags at 22 pesos each and he happened to have actually sold 200 bags to Clemente. While this offer was still running, he even offered to buy 300 bags at 20 each, announcing he would pay after having accepted the offer: "compro 300 costales pago 20 pesos por costal, cheque postfechado" (I buy 300 bags at 20 pesos each, postdated check). With less than 800 pesos, he could not buy such an amount of bags then. Sending money after having fulfilled an offer is possible because the exchange was broken into smaller primitives (send money, send pergamino).

Other less surprising roles included producers of large quantities of coffee who did not try to fulfill the offers but preferred to sell their production to other players and recurring privileged cooperations between some players.

4 Lessons of participatory simulations

Introducing human players in multi-agent based experiments brings several outcomes directly linked to participation itself. The SimCafé experiments belonged to only one of several methodologies of participatory simulations and could be compared with the Multi-Agent System/Role Playing Game (MAS/RPG) methodology.

4.1 Emergence and outcomes of participation

The outcomes of participatory experiments are closely linked to the actual participatory approach used.

In the pedagogical approach, participants are students who are taught the link between individual and collective behaviors. Colella (1998) immersed students in a simulation of virus propagation and asked them to determine the rules of the propagation. This pedagogical tool is actually used to teach students, through role playing activities, the mechanism of the emergence in complex systems (Resnick and Wilensky, 1997).

The negotiation approach aims at helping stakeholders to negotiate. Usually, they are required to explicit their behavior through the participation in the simulation. Sometimes, the roles are exchanged (Etienne, 2003). Emergence in this approach would rather be what Barreteau calls ”social learning” (Barreteau, 2003): observers learn through the learning of players.

The SimCafé experiments belong to the sociological approach, aiming at validating and consolidating models (Guyot, 2003). In this approach, participants are stakeholders and the witnesses of the emergence are domain experts, usually social scientists. Participatory simulations are used as a tool to determine the condition of the emergence. As a matter of fact, this approach belongs to the experimental approach in social sciences (Earley et al., 1990; Chesney and Locke, 1991), especially experimental economics: even if the SimCafé experiments were not led by economists, they could be interpreted as an experience to understand the economic behavior of coffee producers (Castro and Weingarten, 1970).

4.2 Emergence within MAS/RPG

The introduction of participation in multi-agent system design historically lead to what is now called the multi-agent system/role playing game methodology (figure 6). This methodology applies to natural resource management. It consists in first elaborating a multi-agent system to simulate the evolution of the natural resources. This system is then used within a role playing game with the participation of stake holders. Stakeholders are then represented in the multi-agent system. This introduction is not only done from the role playing game experiments, but it can also be done with the help of the stakeholders themselves.

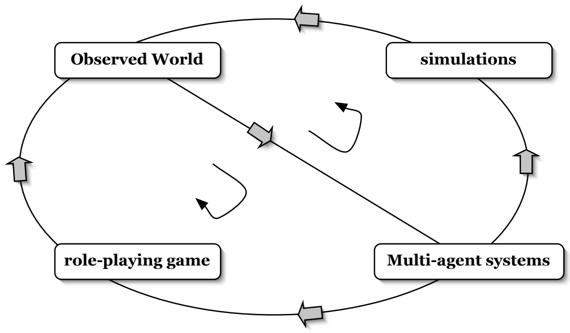


Figure 6: The MAS/RPG Methodology (from Barreteau et al. (2001))

What may emerge in this methodology is an improvement of the multi-agent system with the help of the stakeholders. Understanding the link between reality and what appears on the screen, i.e. the multi-agent system, thanks to role-playing game experiments, stakeholders are able to improve the underlying model of the multi-agent system in order to more closely match reality. This emergence, being only one of the outcomes of this methodology, may or may not be the priority of the domain experts. Some researchers prefer to focus on the negotiation help properties of participatory simulations.

Moreover, the emergence in the MAS/RPG methodology does not include all the features of the specialization of roles observed during the SimCafé experiments. It is rather a participatory design of the multi-agent system. Stakeholders actually help the scientist to adjust his multi-agent system in order to make it more closely match the reality they experiment in their everyday life. Typically, a stakeholder could explain that some behavior of a cell of his land in the multi-agent system could not happen because his land has such and such property that the domain expert ignored.

5 Conclusion

SimCafé was designed as a multi-agent simulation. We started from a domain model and we were able to quite fully follow the traditional multi-agent simulation design process even if what we were building was a participatory simulation instead. The reason is that humans actually take the role of the control architecture. And as expected in these multi-agent simulations, several coalitions emerged.

However, such an interpretation of the experiments are insufficient to understand the two sides of the emergence that occurred during these participatory simulations. To understand why roles emerged, we need to interpret the SimCafé experiments as a multi-agent system, i.e. as a distributed problem solver. Humans, by specializing their roles, tried to improve the capability to fulfill offers of the coffee buyer.

This interpretation intrinsically could not be applied to participatory approaches such as multi-agent systems coupled with role-playing games: in a role-playing game, humans play a pre-defined role. In SimCafé experiments, the problems solving capabilities of the system were improved by the emergence of unexpected roles played by human participants.

Future work include analysis of another experiment, SimBar, based on the El Farol Bar model, where specialized roles apparently didn't emerge. SimCafé also is the first step in the design of multi-agent based participatory simulations (Guyot and Drogoul, 2004) where humans are assisted by semi-autonomous agents.

Acknowledgements

The SimCafé experiments were conducted at the LANIA within a project funded by Le Laboratoire Franco-Mexicain d'Informatique (LAFMI) and directed by Amal El Fallah-Seghrouchni and Christian Lemaître Léon. We also would like to thank the members of the LANIA who accepted to participate in the simulations and who gave us an extremely valuable feedback.

References

Oliver Barreteau. The joint use of role-playing games and models regarding negotiation processes: characterization of associations. *Journal of Artificial Societies and Social Simulation*, 6(2), 2003. URL `http://jasss.soc.surrey.ac.uk/6/2/3.html`.

Oliver Barreteau, François Bousquet, and Jean-Marie Attonaty. Role-playing games for opening the black box of multi-agent systems: method and lessons of its application to senegal river valley irrigated systems. *Journal of Artificial Societies and Social Simulation*, 4(2), 2001. URL `http://jasss.soc.surrey.ac.uk/4/2/5.html`.

Barry Castro and Kenneth Weingarten. Towards experimental economics. *Journal of Political Economy*, 78(3):598–607, 1970.

Amelia A. Chesney and Edwin A. Locke. Relationships among goal difficulty, business strategies, and performance on a complex management simulation task. *Academy of Management Journal*, 34 (2):400–424, 1991.

Vanessa Colella. Participatory simulations: Building collaborative understanding through immersive dynamic modeling. Master's thesis, MIT, 1998. URL `http://xenia.media.mit.edu/~vanessa/part-sims/thesis.html`.

Alexis Drogoul, Thomas Meurisse, and Diane Vanberge. Multi-agent based simulations: Where are the agents? In Jaime Simão Sichman, François Bousquet, and Paul Davidsson, editors, *Multi-Agent-Based Simulation, Third International Workshop, MABS 2002, Bologna, Italy, July 15-16, 2002, Revised Papers*, volume 2581 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2002. ISBN 3-540-00607-9. URL `http://www-poleia.lip6.fr/~drogoul/papers/drogoul.mabs02.pdf`.

P. Christopher Earley, Gregory B. Northcraft, Cynthia Lee, and Terri R. Lituchy. Impact of process and outcome feedback on the relatiojn of goal setting to task performance. *Academy of Management Journal*, 33(1):87–195, 1990.

Michel Etienne. Sylvopast: a multiple target role-playing game to assess negotiation processes in sylvopastoral management planning. *Journal of Artificial Societies and Social Simulation*, 6(2), 2003. URL `http://jasss.soc.surrey.ac.uk/6/2/5.html`.

Paul Guyot. Simulations multi-agents participatives: Vers une méthode de validation et de consolidation des modèles de pratiques collectives. Master's thesis, Université Pierre et Marie Curie, 2003. URL `http://www-poleia.lip6.fr/~guyot/files/2003-dea-simulation.pdf`.

Paul Guyot and Alexis Drogoul. Designing multi-agent based participatory simulations. In Helder Coelho and Bernard Espinasse, editors, *Proceedings of 5th Workshop on Agent Based Simulations*, pages 32–37, Erlangen, San Diego, May 2004. SCS Publishing House. ISBN 3-936150-31-1. URL `http://www-poleia.lip6.fr/~guyot/files/2004-abs-designing.pdf`.

Mitchel Resnick and Uri Wilensky. Diving into complexity: Developing probabilistic decentralized thinking through role-playing activities. *Journal of Learning Sciences*, 7(2), 1997. URL `http://ccl.northwestern.edu/cm/papers/starpeople/`.

R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. In *IEEE Transaction on Computers*, number 12 in C-29, pages 1104–1113, 1980.

Diane Vanbergue. *Conception de simulation multi-agents: application à la simulation des migrations intra-urbaines de la ville de Bogota*. PhD thesis, Université Paris VI, December 2003.

Agent-based participatory simulation activities for the emergence of complex social behaviours

Stefano Cacciaguerra, Matteo Roffilli
Department of Computer Science, University of Bologna
Mura Anteo Zamboni 7, 40127 Bologna, Italy
{scacciag,roffilli}@cs.unibo.it

Abstract

Nowadays, social organizations (at macro-level) can be represented as complex self-organizing systems that emerge from the interaction of complicated social behaviours (at micro-level). Modern multi-agent systems can be employed to explore “artificial societies” by reproducing complicated social behaviours. Unfortunately, promoting interactions only among pre-set behavioural models may limit the capability to explore all possible evolution patterns. In order to tackle this issue, we aim at discovering emergent social behaviours through simulation, allowing human people to participate in the simulation environment, so that the range of possible behaviours is not predetermined. In order to support this new approach, we propose a system architecture that is able to support an endless session level between a software agent and a human player (called participatory framework). In particular, while network faults or human low reactivity do not allow the human being to control his agent, this system architecture adopts a virtual player mechanism (called ghost player) that takes control of the agent driven by the user when he does not. The advanced version of such a ghost player relies on sub-symbolic machine learning techniques for mimicking the strategy of the off-line human being. Preliminary visual results show the effectiveness of our approach.

1 Introduction

Social organizations can be studied at many different levels of abstraction and analysis. Historically, in the analysis of organizational decision-making processes, a common strategy is to reduce a complex social activity to a single constrained optimisation problem that is solved by means of a (macro-level) function. Nowadays, social organizations can be approached as complex self-organizing systems that emerge from the interaction of complicated social behaviours (at micro-level) (Lomi *et al.*, Groningen 2003). Differently from the historical approach, this new one makes it possible to explore the connection between the micro-level behaviour of individuals and the macro-level patterns that emerge from the interaction of many individuals (Lomi *et al.*, Notre Dame 2003). It is possible to effectively describe these behaviours as the actions of agents into an environment, where the agents are the individuals and the environment is the complex self-organizing system. We define an agent as a computer system capable of independent actions in order to satisfy its planned objectives. In particular, in order to describe a complex self-organizing system we need several individuals, while to reproduce it, we need several agents. Along with this consideration, a multi-agent system can be successfully employed, in order to describe self-organizing systems. A multi-agent system is an environment that consists of a number of agents, which interact with one-another. Therefore, it is possible to reproduce social societies into a synthetic environment by creating “artificial societies”. In order to successfully mimic real societies, the multi-agent systems make the agents interact thanks to their ability to cooperate, coordinate, and negotiate. In their current form, multi-agent systems mainly have teaching purposes. For example, a multi-agent system could be used as a computer-based learning environment to teach students of social and economic schools a number of central issues when studying organizational and decision-making processes, and the respective representation of problems (Chen *et al.*, 1993; Colella *et al.*, 1998). These “artificial societies” create a quasi-experimental observation-generation environment where it is possible to conduct tests. Modern multi-agent systems can be employed to explore multiple phenomena from natural to social ones by involving different disciplines: art, biology, chemistry, physics, computer science, earth science, games, mathematics and social sciences.

Well-known modern multi-agent systems are: Swarm (Minar *et al.*, 1996), Repast (Collier *et al.*, 2003), Jas (Sonnessa, 2004), SPADES (Riley, 2003) and Netlogo (Wilensky, 1999). Swarm is a collection of (Objective-C) libraries that promotes the implementation of agent-based models. The Swarm code is Object-Oriented and facilitates the job of simulationists by supporting the incorporation of Swarm objects into their simulation programmes. Its programmes are hierarchical: the top level (called the "observer swarm") creates screen displays and the levels below them. These levels (called the "swarm model") implement the individual agents, schedule their activities, collect information about them and exchange it on the base of an "observer swarm" request. Swarm provides a lot of tutorials that share portions of code in order to facilitate the design of an agent-based model, for example: the management of memory, the maintenance of lists, the scheduling of actions, and so on. Jas and Repast are clones of Swarm originated from the translation of Swarm Objective-C sources into Java. In fact, they provide a (Java) library of objects useful to model, schedule, display and collect data from an agent-based simulation. Again, they allow the visualization of the data obtained from the simulation by means of histograms and sequence graphs. Further, they can show snapshots of the evolution of the simulated complex systems in a 2-dimensional (2D) "movie" format. SPADES (System for Parallel Agent Discrete Event Simulation) is a middleware system for agent-based distributed simulation. SPADES allows the simulationist to define the behaviour of agents (as remote processes) and the rules of the world where they live. This means that, differently from the previous ones, it supports the distributed execution of the agents across multiple operating systems, while at the same time it runs distributed simulations regardless of network or system load while adopting a fair policy. NetLogo is a programmable modelling environment that allows the simulationists to give instructions to several passive (i.e. patches) and active (i.e. turtles) agents all operating at the same time. It also implements a classroom participatory simulation tool (called HubNet). HubNet connects networked computers or handheld devices to the Netlogo environment by helping each user control an agent during a simulation.

Typically (apart from Netlogo), a simulationist can interact with these multi-agent systems only during the configuration phase. This means that a simulationist can only choose the initial conditions and after this phase he simply becomes a spectator of the complex system evolution (simulated). If the estimation of the system variables does not critically affect the soundness of the simulation results, the above approach works right. In other cases, alternative approaches are needed to tackle this problem (ill-posed problem). One of them is called "participatory simulation" (Resnick *et al.*, 1998; Wilensky *et al.*, 1999). It provides a way to expand the capability of interactions with these systems at run time. Hence, during a participatory simulation, each single user can play the role of individual system entities and can see how the behaviour of the system as a whole can emerge from the individual behaviours. These synthetic environments promote the cooperation, coordination, and negotiation among the agents controlled by pre-fixed behavioural models (designed by a simulationist) and those driven by humans, all pursuing their own goals. The emergent behaviour of the model and its relation to the participation of humans can make the dynamics of the simulated system clearer. Therefore, these participatory role-playing activities result useful to understand how complex dynamic systems evolve over the time. This approach is very didactic because it promotes a deeper comprehension of the evolution of the simulated complex system. For example, consider a virtual stock exchange, where each player (investor) can play the role of a virtual buyer or seller who engages in the activities of the resulting share exchange dynamics.

The remaining part of this paper is organized as follows. In Section 2, we illustrate the main limits of the modern multi-agent systems, in general, and of agent-based participatory simulation activities, in particular. In Section 3, we present a new alternative approach that overcomes these limitations by adopting a ghost mimicking software mechanism and a participatory framework. Section 4 shows a set of preliminary results we obtained with a prototypic implementation of our system. Finally, Section 5 concludes our work with some hints for future developments.

2 Limitations of multi-agent systems

One of the main attractions of the above-described simulation environments is the easiness by which it becomes possible to statistically assess the validity of a model. Simulationists can simply explain their idea by writing some lines of code in natural language and then start the simulation. During the evolution they observe the values of some pre-fixed interesting variables and make decisions.

Recent works permit to embed, in real time, results of the simulation by means of a simple but powerful 2-dimensional computer graphics (Repast; Jas; Netlogo). In our previous work (Cacciaguerra *et al.*, Las Vegas 2004), we improve these capabilities with a 3-dimensional (3D) computer graphics highlighting that this improvement allows to tackle a new class of problems from different points of view.
Nevertheless the multi-agent simulations presented up to now share a common feature: they carry out interactions only between pre-set software behavioural models. While this is extremely important for statistical assessments, we argue that it limits the generation of emerging complex behaviours in any simulation. Along with these considerations, we deem that there are two reasons for the limitation.

The first is related to the simplicity of the model assumed. Every model is defined as a hypothetical-deductive assumption related to some personal knowledge of the simulationist. In fact, the simulationist tries to describe his insight about target problem in a way that a deterministic machine can interpret. This approach is very sensitive to the level of accuracy when modelling the target problem. In fact, it results very difficult to accurately describe all the behaviours included in a model because of intrinsic complexity of social interactions. Then, to leave some degree of freedom, stochastic steps are often introduced causing a loss of sharpness in the analysis. In other cases, it is not possible to fully define a behavioural model because of the not-deterministic physical law behind it. Considering these expert design issues, the analysis are often performed only at standard time intervals: at starting point, at running and finally at asymptote. Obviously changing the starting conditions the simulation shows different behaviours, but asymptotically it reaches the same state-condition or the same periodical fluctuation. This approach guarantees the statistical soundness of the simulation results while it limits the capability to explore all possible evolution patterns.

The second reason is related to the bounded computational power. The current software is not able to handle large amounts of interactions in a timely way because of its engineering. In this case, as well as when facing typical problems related to physical simulations, the time constraint cannot be dealt with in a short period by making the experimentation of complex models impossible. In addition, the analysis of physical systems may result easier than the social one because of the rigid constraints and the proven theories behind it.
Hence, it seems to be difficult to implement social simulations that are able to generate new and emergent behaviours. We argue that, by reducing the constraints for the statistical soundness, it is possible to overcome the two limitations (due to both the model accuracy and the time constraint) in an efficient way. To achieve this result, it is necessary for accurate behavioural models to be able to interact together quickly and for a sufficiently long time inside a synthetic environment. In particular, the following is needed:

- A common protocol (i.e. language) to exchange information,
- A high-bandwidth channel for managing communication and
- Large computation power to control behavioural models.

We believe that a cooperative game environment satisfies all the three requirements. A cooperative game is a special kind of game in which many people play together to reach some pre-set goals. The agent-based participatory simulation shows to be one of the best approaches for implementing a cooperative game. It is worth noting that according to our idea the game is the instrument for running a simulation and not the goal of the simulation. One of the main attractions of the transposition of the above problem from a pure software simulation into a cooperative game is that, in the transposed problem, humans can directly interact with the agents inside the synthetic environment by joining the game. Hence, any previous knowledge of the simulation toolkits and programming language is needed, making the simulation methodology widely accessible. Therefore, it becomes possible to use humans as complex and accurate behavioural models for the simulation. In fact, apart from general considerations about Artificial Intelligence (Penrose, 1994), we consider a human being as a very complex social behavioural model. Hence, in defining the objective of the game, we (implicitly) promote the human being to apply his own social model to a pre-fixed task. We argue that this is very similar to the mental process that the simulationist performs when writing a social model for a common simulation toolkit. In addition, humans obviously do not require additional computational power to interact together in a timely way. They also share a priori common language to perform interactions. In fact while a software simulation toolkit offers a hand-made protocol for exchanging information among agents, a game is self-explaining for humans. The 3D visualization (eventually extended with positional 3D audio) is the fastest way to perform interactions among people. In fact, it exploits human natural senses and it is of immediate comprehension. Hence, the cooperative game only demands to create and manage the shared environment to exchange information (that represents the game). In this way, the problem of time constraint is solved too. Further, the cooperative game shows other interesting properties. While solving key problems when running a simulation some questions about experimental design arise.

1. How can we analyse the behaviour of a hand-made behavioural model in such context?
2. Can we assume that providing a large number of participants and a long duration to the simulation will resume the lost statistical soundness?
3. And assuming this is right, how can we find such a large number of people that will play a simulation for an entire week?

3 New approach

In order to tackle these issues, we propose to populate the cooperative game with virtual agents that play together with human players in the same environment. Each virtual agent could be controlled by a software that implements hand-made behavioural models. Further, each human being is represented in the game by his digital avatar that can be fully controlled. Hence, we can think of the avatar as another agent that is driven by the human being instead of a software. In this way, no distinction is made between human beings and software players inside the game context. Hence, with this assumption, from a game perspective it is easy to reach hundreds or thousands or, even, millions of concurrent players.

Further, this trick offers interesting considerations. The first is that it becomes very difficult (if not impossible) to distinguish between software programs and human being-controlled agents (inside the game) using a priory or trivial information. The only way to do that is to analyse the behaviour of an agent for enough time to classify it with some pre-fixed model of knowledge. In other words, a human being should evaluate the strategy (i.e. the pattern of behaviour) of another agent by using his internal definition of what is a strategy. Along with this consideration, we can think to create an agent that makes this classification extremely difficult. Hypothetically, constructing an agent so that no human being can recognize it as a software while playing with it for a long time should be possible. If this *mimic game* is successful, we could safely assert that this software has passed a new version of the Turing test (Turing, 1950). Designing such a software is a hard task and out of the scope of this work. Despite this consideration, promising technologies are emerging.

3.1 The ghost mimicking software

Nevertheless maintaining a high number of human players for a long time is a hard job due to both physiological limits and technical issues. In fact, humans are quickly stressed by intense actions and briefly degrade their mental performances. In addition, depending on the modality of connection to the server where the synthetic environment is accommodated, the play session can be broken by a variety of causes. Managing these exceptions from a network point of view can be a difficult task. In any case, a good simulation environment could admit some degree of freedom in managing the players.

To this aim, we propose a preliminary but powerful adaptive mechanism (see Figure 1). The idea is the following: while a human player is gaming a *ghost software* player is locked to his agent. The ghost software has been previously programmed to run pre-fixed algorithms (a.k.a. behaviourist model) in order to achieve some goals during the game. Further, we deploy an adaptive mechanist able to recognize when the human player stops controlling his agent during the simulation session. Exploiting this service, when the human player breaks the connection the ghost player immediately starts to control the orphan agent avoiding game interruptions and excessive slowdown. If the human player tries to take the control of the agent again, the ghost software returns immediately silent in the background abandoning the control. Hence, this combined mechanism is able to keep the game session of a human player alive during the human rest and the network fault.

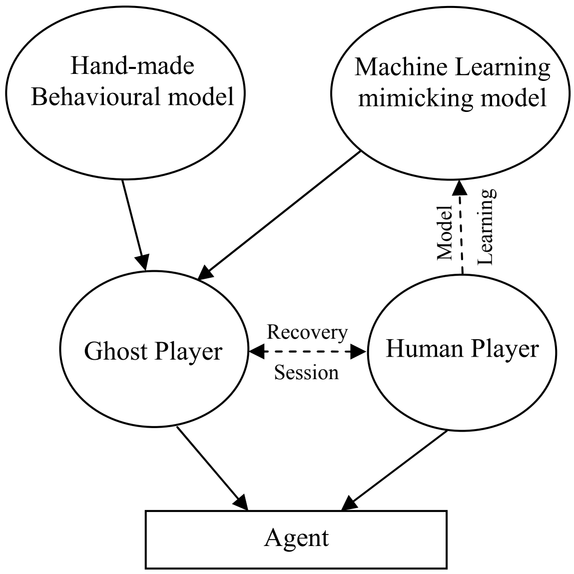


Figure 1: System architecture of our approach

While keeping the game alive this mechanism partially corrupts its coherency. In fact, the ghost player shows a behaviour that is absolutely different from human beings' behaviours. If a lot of ghost players switch on and off intermittently this results in a high degree of unpredictability that potentially transforms the participatory game in a random game where no constructive interactions can be performed. Also a human player that breaks his connection for a short time, and then recovers his session, could find his agent in a situation that destroys his long time strategy.

Considering the previous considerations about the mimic game, we propose to replicate the strategy of the human player by providing the ghost software with mimic capabilities. The ghost software analyses the actions of the agent in background and seeks to fit its own pre-fixed behavioural model to the agent's behaviour. In addition, exploiting Machine Learning (Dietterich, 1997; Mitchell, 1997) technologies, it should be able, starting from an imperfect knowledge (i.e. noise-corrupted estimation of system variables) of the environment, to automatically construct a behavioural model resembling that of a human being. A preliminary mimic methodology could be the following: the ghost software knows the legal actions inside the game and it is programmed to consider only a sequence of *n* moves. Then, it statistically updates the probability of performing the action *y* knowing that *n* actions $x_1,..., x_n$ where previously done.

We are planning to substitute this simple Bayesian statistics in order to reach more accurate fitting and generalization. We are looking for some candidate methodologies gathered from the field of sub-symbolic Machine Learning. In particular we are evaluating Artificial Neural Networks (Bishop, 1995), ε-Machines (Shalizi *et al.*, 2000), and, especially, Support Vector Machines (Vapnik, 1998), which demonstrated good generalization power in hard tasks (Campanini *et al.*, 2004).

It is worth noting that our purpose is not to create an agent that learns to solve a given problem in an unknown environment and in unsupervised manner. This goal had been deeply analysed in the 90's and a bunch of symbolic algorithms were proposed to tackle it. Our aim is to teach an agent to replicate an existing behaviour starting from noise-corrupted knowledge. Thus, it is a sub-symbolic supervised machine-learning task.

3.2 The participatory framework

According to the above-proposed approach, we develop a participatory framework that supports the management of the interaction between humans and agents into any participatory simulation. A user can make decisions (and then can act in the synthetic environment) in place of the behavioural model of an agent. More simply, a user can participate in the evolution of the (remote) simulated complex system. Therefore, this framework implements a connection between the user and the agent where a (ISO/OSI) session level is exploited. The user drives a specific agent by means of a client at application level (according to a client-server model architecture that recalls something similar to the Hubnet tool) that communicates over a network connection to the synthetic environment (see Figure 2). In particular, this mechanism becomes very useful if we are running a participatory simulation over an unreliable network. A typical multi-agent system architecture adopts a fair turn approach to evolve the synthetic environment. This means that each agent must act during each turn (also the NULL move is permitted). Therefore, agents driven by humans must act according to the turn approach too. In addition, the actions coming from the slow remote human player may slow down the whole serialization of the sequences of fair turns. For this reason, the participation of multiple (remote) users can slow down the evolution of the simulated complex system to unacceptable speed. This may be due to two possible reasons: a momentary or permanent interruption of the communication and high-delayed move by the user. In the first case, the momentary interruption is due to network congestion or outages of the communication channel, while the permanent one is due to the client or server disconnection. In the second case, the missed move of the human player is due to his low reactivity, while the unsent move is due to his own free will (i.e. when he leaves the control of the agent to the ghost player to rest). Hence, the goal of this framework is to maintain the speed of the system evolution over a certain time threshold (by supporting the human playability). For this reason, if the human player is not able to participate in all the turns on time, the framework guarantees the correctness of the sequence serialization within pre-fixed time constraints, by imposing to the slow agent to be played by the ghost mimic player.

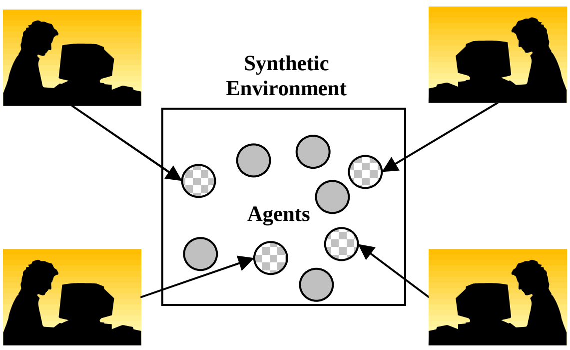


Figure 2: Client-Agent as client-server architecture

In addition this framework manages the communication by means of a "session recovery" mechanism that allows the user to take the control of his agent again, after the interruption of the communication or on his own free will. In this way, the simulationist can exploit a distributed simulation environment that takes advantage of a session level over the standard ISO/OSI stack (see Figure 3).

3.3 Implementation of the participatory framework

We develop a participatory framework that implements a session level over the TCP/IP stack (see the Figure 3). This framework guarantees the correctness of the simulation evolution, and avoids the slowing down of its time performance by accurately managing a session mechanism between the human being and the agent. In particular, the participatory framework consists of a mechanism of session management and a communication management.

The *session management mechanism* guarantees that the human being can participate in the simulation by building his personal session. This means that a human player takes the control of an agent for a simulation run. Therefore, if the human player looses his connection (on purpose or against his own free will) with the agent, his participation in the simulation is guaranteed by the session management mechanism that gives the control to the ghost mimic player. In the near future, if the human player connects his agent again, the mechanism recovers the previously instantiated session by returning the control to the human being.

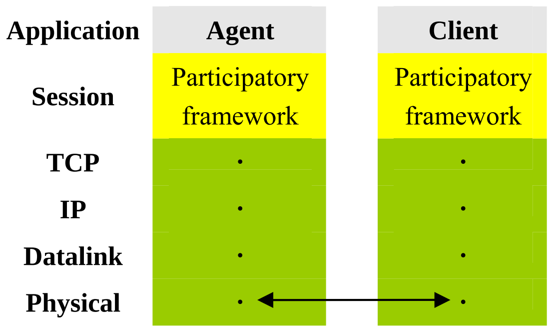


Figure 3: Participatory framework over TCP/IP stack

The *communication management mechanism* consists of an action timeout handler and a TCP timeout handler. The first is used by the communication management mechanism to avoid that a low reactivity from the human player slows down the evolution of the complex systems under a certain threshold. In particular, the action timeout handler controls the responsiveness of the client (on which the human being moves). Therefore the simulationist can set the upper bound (called action timeout) to the responsiveness at a configured time. Obviously, above this bound, the ghost mimic player drives the agent in place of the human being. Instead, the second (TCP timeout handler) is a handler used both by agent and the client. This handler decides if the communication between the agent and the client of the human being is closed, based on statistical calculations related to the previous performance according to the agent responsiveness on one (client) side and human being's responsiveness on the other (agent) side. In particular, from the agent's side, the TCP timeout handler sets the state of a communication as "broken" when a certain number (i.e. maximum consecutive action timeout configured by the simulationist) of consecutively lost interactions occurs. When the state of the communication is considered as "broken", the TCP timeout handler closes it (with a shutdown). Instead, from the client's side, the TCP timeout handler sets the state of a communication as "broken", only after an amount of time (called TCP timeout) has passed without receiving any session acknowledgement from the agent. The agent periodically sends session acknowledgements to the client to confirm his responsiveness and waiting for the next move. After a TCP timeout expiration, the "client" TCP timeout handler closes the communication (with a shutdown). This may be recovered by requesting a connection to his agent (in active way by clicking a button or in passive by adequately setting up the configuration file).

4 Preliminary results

In this section, we want to show the preliminary results that highlight the effectiveness of our novel approach. Along with this consideration, we implement a predator-prey artificial ecosystem as a model for a multi-agent system adopting the innovative components of our prototype. This simple biological model is interesting because it is the base for more complex systems. The predator-prey model randomly positions a variable number of preys and predators in a synthetic environment. Obviously, the preys' goal is to escape, while the predator's is to pursue the prey. Once a predator reaches a prey, it kills it. Otherwise, if a long period of simulated time passes, the predator dies for starvation. In particular, in these preliminary tests, we focus on the escape trajectory of the prey-agent (green ball of Figure 4-6). The Figures 4, 5 and 6 summarize the video clip frames related to different runs of the artificial ecosystem (Cacciaguerra *et al.*, December 2004). These frames represent the output of the predator-prey model executed on our prototype. The red balls report the previous positions of the prey-agent. According to this representation, the set of red balls represents the escape trajectory of the prey. In all the simulation runs, the prey-agent is driven by a human player during the pre-determined time (see inset of Figure 4). After this time, the human player does not send the next moves, leaving the control of the prey-agent to the ghost player (in particular, after a maximum consecutive action timeout, the human beings were disconnected; see the Figure 7). In Figures 4 and 5, the ghost player adopts his mimic capabilities trying to reproduce a pattern of moves (i.e. a strategy) similar to that of the human being. The pattern of moves related to the human being is similar to a stairway. It is clear that the ghost player with mimic capabilities tries to reproduce the same pattern of moves as the human being's. This does not mean that the ghost player duplicates exactly the learned pattern in a periodical manner or in replicated copies. Instead, the ghost player has learned the way in which the human player drives his agent (to escape) and applies this abstract knowledge to mimic his behaviour (called generalization).

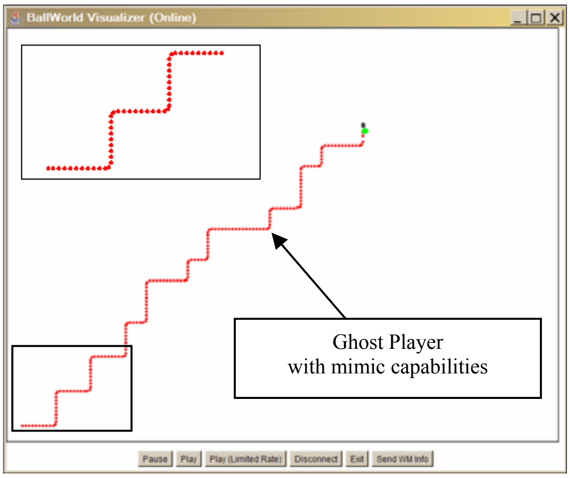


Figure 4: 2D visualization of the escape trajectory of the prey driven by ghost player with mimic capabilities (on Windows XP)

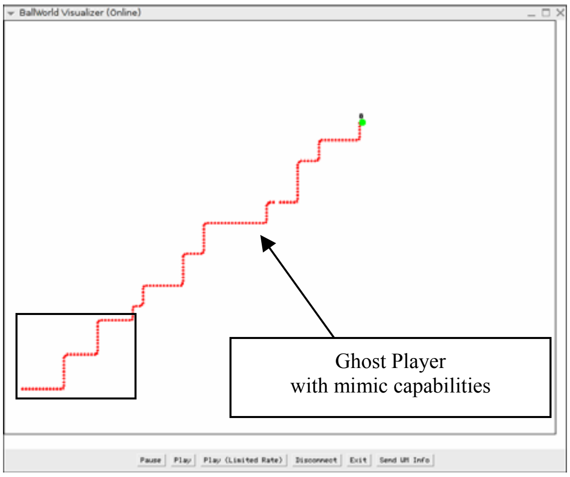


Figure 5: 2D visualization of the escape trajectory of the prey driven by ghost player with mimic capabilities (on Linux)

This becomes clear in Figure 5 where the ghost player, stressed to learn the same sequence of moves (see inset in Figure 4), shows a different but similar behaviour as in Figure 4. Obviously, if we look at Figure 6, where the ghost player was running adopting a non-mimic (i.e. random) algorithm, it is clear that the pattern of moves is very dissimilar.

Finally, Figure 7 shows the responsiveness of the prey-agent during the previously presented simulation run scheme. This graph shows the time spent by the prey-agent to insert his next move into the synthetic environment.

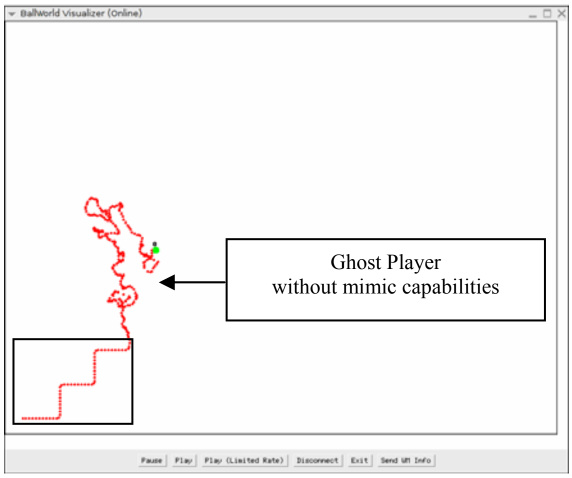


Figure 6: 2D visualization of the escape trajectory of the prey driven by the ghost player without mimic capabilities (on Linux)

In particular, three phases are evident:

I. From zero to 2600 simulated time, the agent is driven by the (remote) human player,
II. From 2601 to 5700 simulated time, the agent is driven by the (local) ghost player because the human being is not playing a move under the action timeout,
III. From 5701 simulated time to the end, the agent is driven by the (local) ghost player because a "maximum consecutive action" timeout has expired.

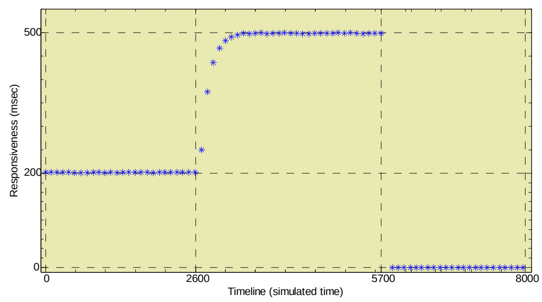


Figure 7: Responsiveness of the prey-agent before and after a user disconnection

5 Conclusions

We have designed and developed a software prototype able to support the execution of agent-based participatory simulation activities to discover the emergence of complex social behaviours. In particular, this prototype supports the participants with an endless session level that allows the human player to disconnect from the synthetic environment while a ghost player takes the control of his agent. A mimicking strategy has been developed to drive the ghost player by means of Machine Learning algorithms. Enabling both our proposed mechanism and framework makes it possible to engage agent-based participatory simulation activities with thousands of players dispersed in the world for a long time. The mimicking mechanism is fundamental to maintain a good level of coherence in the game during network faults and human rest. Preliminary results confirm, by means of visual graphs, the efficacy of our approach. In particular, the movie (in mpeg format) of the simulation run reported in Figure 4 highlights the usefulness of our approach. We are designing our software prototype to pass to a new version of the Turing test using some methodologies gathered from the field of Machine Learning as Artificial Neural Networks, ε-Machines, and Support Vector Machines. Further, we are currently planning a massive experimental campaign to study the performance of our participatory framework. We hope this will demonstrate the emergence of complex social behaviours. In order to achieve these results learning behavioural models through imitation seems to be a key point. Finally, we wish to conclude this work by mentioning that these trained behavioural models may be very effective in other possible application fields such as digital cinema (Regelous, 2004), edutainment (Wilensky *et al.*, 1999), and multiplayer games (Ferretti *et al.*, 2003) where people can leave and come back.

Acknowledgements

Many thanks to Marco Pracucci for his contribution in the implementation of our idea.

References

C. M. Bishop. Neural Networks for Pattern Recognition. *Clarendon Press*, Oxford, 1995.

S. Cacciaguerra, A. Lomi, M. Roccetti and M. Roffilli. A Wireless Software Architecture for Fast 3D Rendering of Agent-Based Multimedia Simulations on Portable Devices. *In proc. of the IEEE CCNC 2004*, Las Vegas, USA, 2004.

S. Cacciaguerra and M. Roffilli. AISB 2005 movie website. http://wwwfn.csr.unibo.it/projects.html (Dec 2004).

R. Campanini, D. Dongiovanni, E. Iampieri, N. Lanconelli, M. Masotti, G. Palermo, A. Riccardi and M. Roffilli. A novel featureless approach to mass detection in digital mammograms based on Support Vector Machines. *In Phys. Med. Biol.* 49:961-975, 2004.

D. Chen and W. Stroup. General Systems Theory: Toward a conceptual framework for science and technology education for all. *Journal for Science Education and Technology*, 1993.

V. Colella, R. Borovoy and M. Resnick. Participatory Simulations: Using Computational Objects to Learn about Dynamic Systems. *In Proc. of the Computer Human Interface (CHI '98) Conference*, Los Angeles, 1998.

N. Collier, T. Howe and M. North. Onward and Upward: The Transition to Repast 2.0, *Proc. of the First Annual North American Association for Computational Social and Organizational Science Conference*, Pittsburgh, USA, 2003.

T. G. Dietterich. Machine Learning Research: Four Current Directions AI Magazine. 18 (4), 97-136, 1997.

S. Ferretti and S. Cacciaguerra. A Design for Networked Multiplayer Games: an Architectural Proposal. *In proc. of the Euromedia'03*, Plymouth, UK, 2003.

A. Lomi and S. Cacciaguerra. Organizational Decision Chemistry on a Lattice. *In proc. of the 7th Annual Swarm Users/Researchers Conference)*, Notre Dame, USA, 2003.

A. Lomi and S. Cacciaguerra. The Emergence of Routines in an Organizational Decision Chemistry. *In proc. of the 1st European Social Simulation Association Conference (ESSA'03)*, Groningen, Netherland, 2003.

N. Minar, R. Burkhart, C. Langton and M. Askenazi. *The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations*, Santa Fe Institute Working Paper, 1996.

T. Mitchell. Machine Learning, McGraw Hill,1997.

R. Penrose. "Shadows of the Mind: A Search for the Missing Science of Consciousness", Oxford Univ Press, 1994.

S. Regelous. Massive Software. http://www.massivesoftware.com (Dec 2004).

M. Resnick and U. Wilensky. Diving into complexity: Developing probabilistic decentralized thinking through role-playing activities. *Journal of the Learning Sciences*, 1998.

P. Riley. SPADES: System for Parallel Agent Discrete Event Simulation. *AI Magazine, 24(2):41–42, 2003.*

C. R. Shalizi and J. P. Crutchfield. Pattern Discovery and Computational Mechanics. *In proc. of the 17th International Conference on Machine Learning*, Santa Fe Institute, 2000.

M. Sonnessa. The JAS (Java Agent-based Simulation) Library. *In proc. of the 7th Annual Swarm Users/Researchers Conference (SwarmFest'03)*, Notre Dame, Indiana, 2003.

A. Turing. Computing machinery and intelligence, Mind, 1950.

V. Vapnik. Statistical Learning Theory, New York, 1998.

U. Wilensky. NetLogo. http://ccl.northwestern.edu/netlogo/. *Center for Connected Learning and Computer-Based Modeling*, Northwestern University. Evanston, IL, 1999.

U. Wilensky and W. Stroup. HubNet. http://ccl.northwestern.edu/netlogo/. *Center for Connected Learning and Computer-Based Modeling*, Northwestern University. Evanston, IL, 1999.

Engineering with Sociological Metaphors: Examples and Prospects

David Hales
*Dept. of Computer Science
University of Bologna, Italy
dave@davidhales.com

Abstract

One way of approaching the engineering of systems with desirable properties is to examine naturally occurring systems that appear to have such properties. One line of work examines biological theories and phenomena. Ideas from the social sciences are less well explored as a possible source of so-called 'self-*' (self-organisation, self-repair, self-management) engineering techniques. We briefly overview some recent work that follows this latter approach and consider some specific prospects for future work.

1 Why Social Science?

Human social systems appear to be scalable, self-repairing and self-regulating and often robust. They spontaneously form, and emerge apparently functional structures, institutions and organisations.

Much social scientific research has been produced concerning why and how social phenomena occur and social science itself has numerous sub-discplines, sub-schools, methodologies and approaches.

We believe that many of the deep engineering problems inherent in the self-* (self-star) approach (Babaoglu, O., Jelasity, M., Montresor, A., van Steen, M., van Moorsel, A., Fetzer, C., Leonardi, S. (in press) can be thought of as sociological questions.

Recently, new computational approaches have been applied to explore the complex processes of emergence that often characterise social phenomena. This approach forces a new kind of rigour on social theory construction and offers the prospective self-* engineer a possible source of ideas to plunder.

2 Computational Social Science

It is only very recently, with the arrival of cheap, fast, desktop computers and social science researchers who know how to program them, that a new area of 'computational social science' has begun to emerge.

There has been an explosion of published work concerning sociologically motivated computational models (Gilbert, N. and Doran J., 1994; Gilbert, N. and Conte, R., 1995; Epstein, J.M. and Axtell, R., 1996; JASSS , 2004). In contrast to early equation-based 'high-level' models, in which there was no space of individual behaviours, much of these models are described as 'agent-based'.

Agent-based modelling in these contexts means a discreet, individual and event-based approach. Individual behaviours of agents (representing people, groups or institutions) are programmed explicitly as a computer program. A population of such agents (or programs) inhabiting a shared environment are then allowed to interact over time and the emergent results and outcomes are observed. It is therefore a prerequisite of such work that agent behaviours must be specified algorithmically.

The emphasis of much computational social science is on the emergent properties of these 'artificial societies'. By experimentation and observation researchers attempt to gain general insights into mechanisms of social emergence and then to relate these to real human societies. Since the outputs produce by algorithms are objective properties, of those algorithms, they can be verified (or more accurately disproved) following a kind of quasi-inductive replication methodology, similar to experimental verification in the natural sciences (Edmonds B. and Hales D., 2003).

It should be noted that the relationship between real social systems and computer models is, and probably always will be, highly controversial — human social systems are so complex, fluid and political (by definition) that debates about what constitutes adequate validation and verification of models rarely converge to agreement. However, these kinds of debates do not need to trouble an engineer looking for new techniques to construct self-* systems.

3 A Brief Note on Game Theory

Some branches of economics, particularly classical game theoretical approaches, formalised their subject matter, analytically, some time ago. This was due, in part, to the advances made by von Neumann and Morgenstern's seminal work (von Neumann, J. and Morgenstern, O., 1944) and early pioneers such as Nash (Nash, J. F. , 1950).

However, due to the focus and strong assumptions of classical game theory — quite proper for the original focus and application of the work — a lot of results are hard to apply to typical self-* scenarios (e.g. noisy, dynamic and with little information concerning the possible behaviour of other units in the system). The classical approach gives analytical proofs of the 'best' way to act in a given situation under the assumption that each actor or agent has complete information and infinite computational resources.

Despite these qualifications, classical game theoretical analysis has many possible areas of application (Binmore, K., 1998) — but we will not concentrate on these here. Also the abstracted scenarios (games) constructed by game theorists to capture certain kinds of social interactions are useful as a basis for evaluating other kinds of modelling techniques (as we shall see later with the Prisoner's Dilemma game).

Interestingly, within economics there are now many researchers using agent-based modelling to concentrate on issues, such as emergence, using agents employing simple heuristics or evolutionary learning algorithms — this area is often termed 'Agent-based Computational Economics' (ACE) (Kirman, A.P., and Vriend, N.J., 2001).

We contrast the 'sociologically inspired' approach we overview in this paper with a classical game theoretic approach — specifically we are more interested in dynamics than equilibrium and in the development of algorithms that can function in noisy environments with incomplete information.

4 Example: BitTorrent and World War I

A general issue explored by much computational sociological work is that of maximising the collective performance of a group while allowing individual agents reasonable levels of autonomy. In many situations there arises a contradiction between these two aspects. This kind of thing happens in human societies all the time, for example, when someone decides to not to pay on a short train ride (free-ride) or evade tax by not declaring income.

One way to stop these anti-social behaviours is to impose draconian measures via centralised government control — ensuring all individuals behave for the common good stopping free-riders. However, this is costly and hard to police and raises other issues such as: who polices the police? In the parlance of distributed systems engineering — the method does not scale well, is sensitive to noise and has a high computational overhead.

In the context of actually deployed massively distributed software systems, Peer-2-Peer (P2P) file sharing applications (such as the KaZaA and eDonkey systems) have similar problems — most users only download files rather than sharing them (Adar, E. and Huberman, B., 2000). This limits the effectiveness of such systems. Even when the P2P client software is coded to force some level of sharing, users may modify and redistribute a hacked client. It has been noted that P2P file sharing is one of the applications in which only a small number of altruists are needed to support a large number of free riders (Adar, E. and Huberman, B., 2000). Consequently it can be argued that this might be why popular P2P applications tend to be limited to only file sharing rather than, say, processor or distributed storage for example.

These sort of cases can be seen as examples of a more fundamental issue: how can one maintain cooperative (socially beneficial) interactions within an open system under the assumption of high individual (person, agent or peer) autonomy. An archetype of this kind of social dilemma has been developed in the form of a minimal game called the Prisoner's Dilemma (PD) game.

In the PD game two players each selected a move from two alternatives and then the game ends and each player receives a score (or pay-off). Figure 1 shows a so-called 'pay-off matrix' for the game. If both choose the 'cooperate' move then both get a 'reward' — the score R. If both select the 'defect' move they are 'punished' — they get the score P. If one player defects and the other cooperates then the defector gets T (the 'temptation' score), the other getting S (the 'sucker' score). When these pay-offs, which are numbers representing some kind of desirable utility (for example, money), obey the following constraints: $T > R > P > S$ and $2R > T + S$ then we say the game represents a Prisoner's Dilemma (PD). When both players cooperate this represents maximising of the collective good but when one player defects and another cooperates this represents a form of free-riding. The defector gains a higher score (the temptation) at the expense of the co-operator (who then becomes the 'sucker').

	Cooperate	Defect
Cooperate	R, R	S, T
Defect	T, S	P, P

Figure 1: A payoff matrix for the two-player single round Prisoner's Dilemma (PD) game. Given $T > R > P > S \wedge 2R > T + S$ the Nash equilibrium is for both players to select Defect but both selecting Cooperate would produce higher social and individual returns. However, if either player selects Cooperate they are exposed to Defection by their opponent — hence the dilemma

A game theoretic analysis drawing on the Nash equilibrium solution concept (as defined by the now famous John Nash (Nash, J. F. , 1950)) captures the intuition that a utility maximising player would always defect in such games because whatever the other player does a higher score is never attained by choosing to cooperate. The Nash Equilibrium (NE) might be a partial explanation for why there is so much free-riding on existing P2P file-sharing systems users are simply behaving to maximise their utility. However, do we have any way to solve this problem without going back to centralised control or closed systems? The NE analysis gives us a good explanation for selfish behaviour but not for altruistic behaviour. As stated earlier, even in P2P file sharing systems there are some altruists (keeping the show on the road).

It has been argued by many researchers from the social and life sciences that human societies produce much more cooperation than a Nash analysis would predict. Consequently, various cooperation promoting mechanisms (often using the PD as their test case) have been proposed by social scientists.

BitTorrent, designed by Bram Cohen (Cohen, B., 2003), employs a strategy popularised in the 1980's by computer simulation tournaments applied to the PD. Researchers were asked to submit programs (agents if you like) that repeatedly played the PD against each other (Axelrod, R., 1984). The result of all these tournaments was that a simple strategy called 'Tit-For-Tat' did remarkably well against the majority of other submitted programs.

Tit-for-tat (TFT) operates in environments where the PD is played repeatedly with the same partners for a number of rounds. The basic strategy is simple: an agent starts by cooperating then in subsequent rounds copies the move made in the previous round by its opponent. This means defectors are punished in the future: the strategy relies on future reciprocity. To put it another way, the "shadow" of future interactions motivates cooperative behaviour in the present. In many populations and scenarios this simple strategy can outperform pure defection in the repeated PD.

In the context of BitTorrent, while a file is being downloaded between peers, each peer maintains a rolling average of the download rate from each of the peers it is connected to. It then tries to match it's uploading rate accordingly. If a peer determines that another is not downloading fast enough then it may 'choke' (stop uploading) to that other. Additionally, peers periodically try new peers randomly by uploading to them testing for better rates (Cohen, B., 2003).

Axelrod used the TFT result to justify sociological hypotheses such as understanding how fraternisation broke out between enemies across the trenches of World War I. Cohen has applied a modified form of TFT to produce a decentralised file sharing system resistant to free-riding, robust against a number of possible exploitative strategies and scalable.

However, TFT has certain limitations and it is not guaranteed to always be the best way of avoiding free-riding strategies, but its simple to implement and performs 'well enough' (currently at least) — BitTorrent traffic currently constitutes a major portion of bandwidth usage on the Internet.

The Tit-For-Tat (TFT) strategy employed by BitTorrent works well when agents exchange many file parts over a period of time (repeat the game interaction many times) but is next to useless if interactions follow a single interaction (such as a single game of the Prisoner's Dilemma). This tends to limit it's use to the sharing of very large files where mutual cooperation can be established.

But how might "strangers" who interact only once come to co-operate? We discuss a recent technique developed from socially motivated computer models in the next section.

5 Example: File Sharing and the 'Old School Tie'

Recent work, drawing on agent-based simulations of cooperative group formation based on 'tags' (surface features representing social labels or cues (Holland, J., 1993)) suggests a novel co-operation mechanism which does not require reciprocal arrangements

(Hales, D., 2000; Riolo, R., Cohen, M. D. & Axelrod, R., 2001). It is based on the idea of a kind of 'cultural group selection' and the well known social psychological phenomena that people tend to favour those believed to be similar to themselves even when this is based on seemingly arbitrary criteria (e.g. wearing the same coloured tie). Like TFT, the mechanism is refreshingly simple. Individuals interact in cliques (subsets of the population sharing the same tags). Periodically, if they find another individual who is getting higher utility than themselves they copy them — changing to their clique and adopting their strategy. Also, periodically, individuals form new cliques and / or randomly change their strategies.

Defectors can do well initially, suckering the co-operators in their clique — but ultimately all the co-operators leave the clique for pastures new — leaving the defectors alone with nobody to free-ride on. Those copying a defector (who does well initially) will also copy their strategy, further reducing the free-riding potential in the clique. So a clique containing any free-riders quickly dissolves but those containing only co-operators grow.

Given an open system of autonomous agents all cliques will eventually be invaded by a free-rider who will exploit and dissolve the clique. However, so long as other new cooperative cliques are being created then co-operation will persists in the population as a whole.

In the sociologically oriented models, cliques are defined as those individuals sharing the same labels and their interpretation is as some kind of socially observable marking attached to individuals. There is no population structure other than the cliques themselves and the population changes over time by employing a population level evolutionary algorithm employing replication and mutation (Hales, D., 2000; Riolo, R., Cohen, M. D. & Axelrod, R., 2001).

In the context of application to P2P systems the clique to which a node belongs is defined by it's immediate neighbourhood. Movement between cliques and copying of strategies follows a process of network 're-wiring' which brings a form of evolutionary process into the network — an Evolutionary Rewiring Algorithm (ERA). Figure 2 gives an example of this simple re-wiring process followed by each node over time.

The adapted tag mechanisms have been shown to be effective in a simulated P2P file-sharing scenario (Hales, D. , 2004) based on that given by Sun Q. & Garcia-Molina, H. (2004). The mechanism demonstrates high scalability with zero scaling cost i.e. it does not take longer to establish cooperation in bigger populations (see figure 3). Although there are outstanding issues to be addressed before the technique can be deployed it offers applications beyond file sharing (such as load sharing or co-operative routing). The ERA algorithm bears some comparison with the SLIC algorithm (Sun Q. & Garcia-Molina, H., 2004) which makes use of incentives. The ERA appears to achieve similar results by producing an emergent incentive structure.

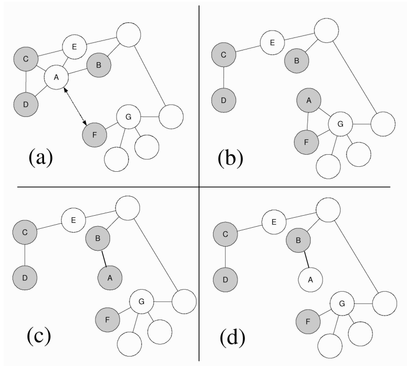


Figure 2: An illustration of 'replication' and 'mutation' as applied in the Evolutionary Rewiring Algorithm (ERA), from Hales, D. (2004). Shading of nodes represents strategy. In (a) the arrowed link represents a comparison of utility between A and F. Assuming F has higher utility then (b) shows the state of the network after A copies Fs links and strategy and links to F. A possible result of applying mutation to As links is shown in (c) and the strategy is mutated in (d).

The tag-based process has been likened to 'old school tie' in-group effects (Sigmund & Nowak, 2001; Hales, D., 2001) that appear to permeate many human societies. It offers a possible explanation for why individuals may behave more altruistically towards perceived in-group members, even if they have never met before — a puzzle for self-interest based social theory. Here we have given an overview of how the same mechanism was adapted and applied within a simulated file-sharing P2P scenario to control free-riding when nodes act selfishly (Hales, D. , 2004).

6 Prospect: Specialisation with 'Foraging Tribes'

Specialisation between individuals is the basis of human society. Agents come to specialise in particular

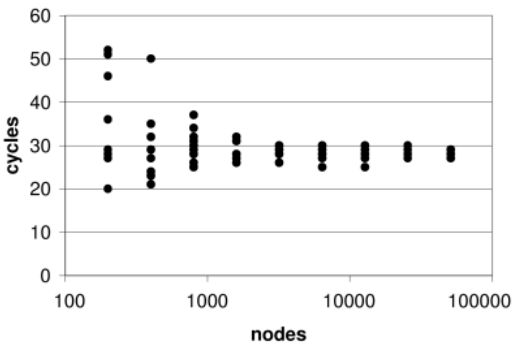


Figure 3: The chart shows the number of cycles required before high file-sharing behaviour is attained. Ten independent runs for each network size are shown. Note that increasing the network size does not increase the time to high performance — from Hales, D. (2004).

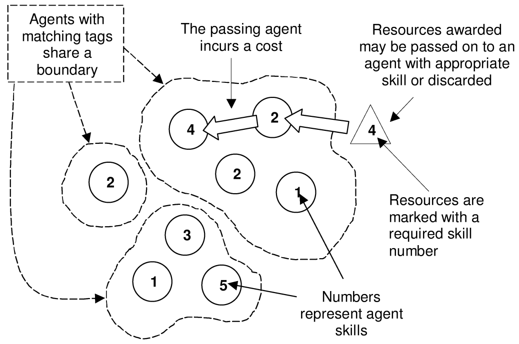


Figure 4: A schematic representation of how resources are passed to an in-group with the required skill at a cost to the passing agent and hence making use of in-group altruism (from Hales, D. (2004)).

tasks and then use methods of exchange or communal ownership to meet the needs of the collective. But how can agents with only local knowledge and simple learning rules come to specialise in this way — particularly if they behave selfishly?

Some models have demonstrated how group processes similar to those discussed previously (i.e. tag-based) can produce internally specialised co-operative groups (Hales, D., 2002, 2004; Spector, L., J. Klein, C. Perry, and M. Feinstein., 2003). Instead of agents evolving behaviours relating to just co-operation or non-co-operation they evolve discreet skill-types in addition to altruistic giving behaviour.

In (Hales, D., 2002, 2004) a resource foraging and harvesting scenario is modelled. Agents forage for resources and then harvest them to gain energy. Different resources require different skills but agents can only posses one skill at a time and are therefore only able to harvest those resources that match their specific skill. An agent may pass a resource it can not harvest to a fellow agent at a cost to itself (an altruistic act) or it may simply ignore such resources (act selfishly). When an agent harvests a resource it attains energy (utility) which can be considered as a form of 'fitness'. Figure 4 gives a schematic of the scenario.

If agents follow a tag-based evolutionary algorithm (similar to that previously described) then they form groups (which can be thought of as cliques or 'tribes') that contain a diversity of skills within them and sharing becomes high.

Figure 5 gives some results from (Hales, D., 2002). The main result worth noting is that donation rates are high even when the cost of giving is high to the donating agent. The cost values given are as a proportion of the the harvest value of a resource (one unit of energy).

As can be seen, even when donation costs half as much as a harvested resource, donation rates are still high if the environment is sufficiently 'resource rich' and a 'smart' method of locating recipients is used (the smart method simply means that agents are able to locate others within their group directly rather than search randomly in the population for them — we do not concern ourselves hear with this issue).

We can envisage prospects for application of this technique to the formation of internally specialised cliques within P2P networks. The skills would become different kinds of services that nodes could offer (e.g. processing, query answering, storage) and resources could represent job requests submitted at nodes. Figure 6 shows a schematic of this.

The process of translation from the abstract sociologically oriented models previously produced (Hales, D., 2002, 2004) to a P2P type application is a non-trivial exercise — for example, the previous exercise of applying 'tag' models of co-operation to P2P file-sharing involved a four stage process in which an abstract model was adapted towards an application domain (Hales, D. , 2004). At each stage a simulation model needed to be extensively explored to ensure that the desirable emergent properties had not been lost.

However, we are given confidence that specialisation can be generated within working systems since recent work, applied to simulated robotics, applying similar techniques based on tags (combined with ge-

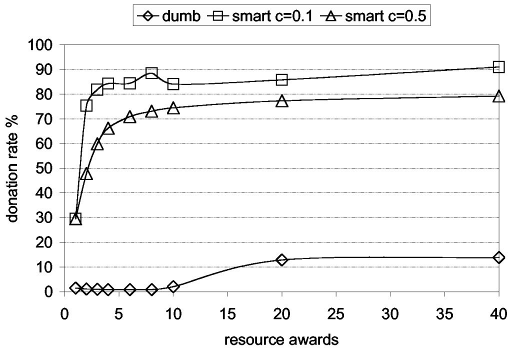


Figure 5: The chart shows averaged results from a number of runs where there are five skills associated with five unique resource types. The x-axis indicates how 'resource rich' the environment is. The y-axis indicates the amount of altruistic donation within groups. The comparison of dumb and smart agents refers to the method of locating a recipient for the donation and the cost indicates the cost to the donating agent (from Hales, D. (2002)).

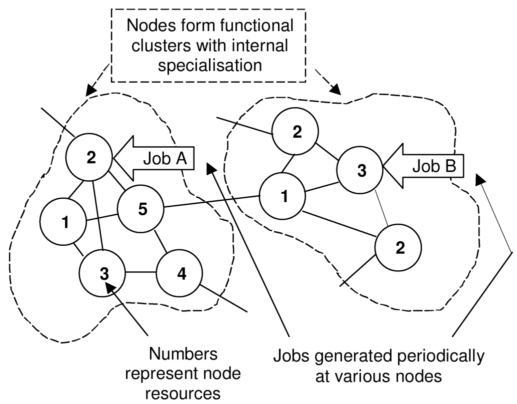


Figure 6: The specialisation mechanism could be applied within a peer-to-peer network. The above schematic shows an example network fragment. Jobs are submitted at nodes and may require services (or resources) from other nodes. Using a similar mechanism to the ERA algorithm described previously, the network could be made to self-organise into functional clusters to satisfy job requests.

netic programming) produced specialised and altruistic behaviour within in-groups (or 'tribes') (Spector, L., J. Klein, C. Perry, and M. Feinstein., 2003).

7 Prospect: Power, Leadership and Hierarchy

A major area of interest to social scientists is the concept of power — what kinds of process can lead to some individuals and groups becoming more powerful than others? Most explanations are tightly related to theories of inequality and economic relationships, hence this is a vast and complex area.

Here we give just a brief very speculative sketch of recent computational work, motivated by sociological questions, that could have significant import into understanding and engineering certain kinds of properties (e.g. in peer-to-peer systems), in which differential power relationships emerge and may, perhaps, be utilised in a functional way.

Interactions in human society are increasing seen as being situated within formal and informal networks (Kirman, A.P., and Vriend, N.J., 2001). These interactions are often modelled using the abstraction of a game capturing interaction possibilities between linked agents (Zimmermann, M.G., Egufluz, V.M. and San Miguel., 2001). When agents have the ability to change their networks based on past experience and some goals or predisposition, then, over time, networks evolve and change.

Interstingly, even if agents start with more-or-less equal endowments and freedom to act, and follow the same rules, vastly unequal outcomes can be produced. This can lead to a situation in which some nodes become objectively more powerful that other nodes through topological location (within the evolved network) and exploitative game interactions over time.

Zimmerman et al found this in their simulations of agents playing a version of the Prisoner's Dilemma on an evolving network (Zimmermann, M.G., Egufluz, V.M. and San Miguel., 2001). Their motivation and interpretation is socio-economic: agents accumulate 'wealth' from the payoffs of playing games with neighbours and make or break connections to neighbours based on a simple satisfaction heuristic (based on a rule discussed in Kirman, A. (1993)).

Figure 7 from Zimmermann, M.G., Egufluz, V.M. and San Miguel. (2001)) shows a an example of an emergent stable hierarchical network structure. Interestingly, it was found that, over time, some nodes accumulate large amounts of 'wealth' (through ex-

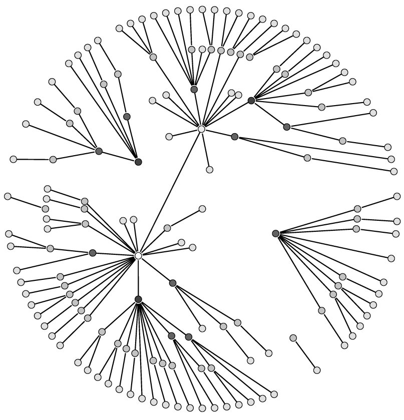


Figure 7: Forms of 'hiearchy', 'leadership' and unequal wealth distribution have been observed to emerge in simulated interaction networks (from Zimmermann, M.G., Egufluz, V.M. and San Miguel. (2001)). Nodes play PD-like games with neighbours and break connections based on a simple satisfaction rule. Hierarchies are produced in which some nodes are more connected and hence can effect the network dramatically by their individual actions — a form of 'topological power'.

ploitative game behaviour) and other nodes become 'leaders' by being at the top of a hierarchy. These unequal topological and wealth distributions emerge from simple self-interested behaviour within the network. Essentially, leaders, through their own actions, can re-arrange significantly the topology of the network — those on the bottom of the hierarchy have little 'topological power'.

The idea of explicitly recognising the possibility of differential power between sub-units in self-* systems and harnessing this is an idea rarely discussed in engineering contexts but could offer new ways to solve difficult co-ordination problems.

Considering P2P applications, one can envisage certain kinds of task in which differential power would be required for efficient operation — e.g. consider two nodes negotiating an exchange on behalf of their 'group' or 'follower' nodes. This might be more efficient than individual nodes having to negotiate with each other every time they wished to interact. Or consider a node reducing intra-group conflict by imposing a central plan of action.

We mention the notion of engineering emergent power structures, briefly and speculatively here, because we consider power to be an under-explored phenomena within evolving information systems. Agents, units or nodes are often assumed to have equal power. It is rare for human societies to possess such egalitarian properties and perhaps many self-* like properties are facilitated by the application of unequal power relationships. We consider this a fascinating area for future work.

8 Conclusion and Summary

Here we have provided some examples and prospects of sociologically inspired approaches to engineering self-* systems. Rather than attempt an extensive overview we have focused on a few encouraging specific results and possible P2P-type applications.

We believe that the computational social science literature can be a potential source of new techniques and ideas for prospective self-* engineer because social phenomena are generally self-organising, robust and scalable — all desirable properties for self-organising information systems.

Computational social science tries to reverse engineer general properties at a fairly abstract level whereas self-* engineers need to apply techniques to specific concrete problem domains. As we have hoped to show, however, it is possible to import useful techniques (see (Hales, D. , 2004) for a case study in applying a technique to realistic domain) from the one approach to the other.

The idea of using social metaphors and approaches for the construction of smart information systems is far from new (Minsky, M., 1988). What is new is that distributed systems engineers are increasing asking sociological questions (even if they are unaware of it!) and social scientists are increasingly turning to algorithmic specification and computer simulation to explore their theories. We hope that advances from both areas can be brought together and used to reinforce each other. Experience so far indicates this not to be an unreasonable hope.

Acknowledgements

This work partially supported by the EU within the 6th Framework Program under contract 001907 (DELIS).

References

Adar, E. and Huberman, B. Free Riding on Gnutella. *First Monday*, 5(10), 2000.

Axelrod, R. *The evolution of cooperation.* N.Y.: Basic Books, 1984.

Babaoglu, O., Jelasity, M., Montresor, A., van Steen, M., van Moorsel, A., Fetzer, C., Leonardi, S. (Eds.) *Self-* Properties in Complex Information Systems* To be published by Springer-Verlag, (in press).

Binmore, K. *Game Theory and the Social Contract. Volume 2: Just Playing*. Cambridge, MA: The MIT Press, 1998.

Cohen, B. Incentives Build Robustness in BitTorrent. Presented at the *1st Workshop on the Economics of Peer-2-Peer Systems*, June 5-6, Berkley, CA, 2003 (Available at: http://www.sims.berkeley.edu/research/conferences/p2pecon/), 2003.

Edmonds B. and Hales D. Replication, replication and replication - some hard lessons from model alignment. *Journal of Artificial Societies and Social Simulation*, (4), 2003.

Epstein, J.M. and Axtell, R. *Growing Artificial Societies: Social Science From The Bottom Up*. London: MIT Press, 1996.

Gilbert, N. and Doran J., (eds.) *Simulating Societies: the Computer Simulation of Social Phenomena.* London: UCL Press, 1994.

Gilbert, N. and Conte, R. (eds.): *Artificial Societies: the Computer Simulation of Social Life.* London: UCL Press, 1995.

Hales, D. Cooperation without Space or Memory: Tags, Groups and the Prisoner's Dilemma. In Moss, S., Davidsson, P. (eds.) *Multi-Agent-Based Simulation. Lecture Notes in Artificial Intelligence 1979*. Berlin: Springer-Verlag, 2000.

Hales, D. Tag Based Cooperation in Artificial Societies. Ph.D. Thesis, Department of Computer Science, University of Essex, UK, 2001.

Hales, D. Evolving Specialisation, Altruism and Group-Level Optimisation Using Tags. In Sichman, J. S., Bousquet, F. Davidsson, P. (eds.) *Multi-Agent-Based Simulation II. Lecture Notes in Artificial Intelligence 2581*. Berlin: Springer-Verlag, 2002.

Hales, D.: Searching for a Soulmate — Searching for Tag-Similar Partners Evolves and Supports Specialization in Groups. In Lindemann, G., Moldt, D. and Paolucci, M., (eds.) *Regulated Agent-Based Social Systems — 1st Intnerational Workshop, Lecture Notes in Artificial Intelligence 2934*. Berlin: Springer-Verlag, 2004.

Hales, D. From selfish nodes to cooperative networks — emergent link based incentives in peer-to-peer networks. In *Proc. of the 4th IEEE International Conference on Peer-to-Peer Computing (P2P2004)*. IEEE Computer Soc. Press, 2004.

Holland, J. The Effect of Labels (Tags) on Social Interactions. Santa Fe Institute Working Paper 93-10-064. Santa Fe, NM, 1993.

JASSS. The Journal of Artificial Societies and Social Simulation (JASSS). Available at: http://jasss.soc.surrey.ac.uk. 2004

Kirman, A. Ants, Rationality and Recruitment. *Quarterly Journal of Economics*, 108, 137156, 1993.

Kirman, A.P., and Vriend, N.J. Evolving Market Structure: An ACE Model of Price Dispersion and Loyalty. *Journal of Economic Dynamics and Control*, 25, Nos. 3/4, 459-502, 2001.

Minsky, M. *Society of Mind*. Simon & Schuster, 1988.

Nash, J. F. Equilibrium Points in N-Person Games, *Proc. Natl. Acad. Sci.* USA 36, 48-49, 1950.

Riolo, R., Cohen, M. D. & Axelrod, R. Cooperation without Reciprocity. *Nature* 414, 441-443, 2001.

Sigmund & Nowak Tides of tolerance. *Nature* 414, 403-405, 2001.

Spector, L., J. Klein, C. Perry, and M. Feinstein. Emergence of Collective Behavior in Evolving Populations of Flying Agents. In E. Cantu-Paz, et al (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2003)*, pp. 6173. Berlin: Springer-Verlag, 2003.

Sun Q. & Garcia-Molina, H. SLIC: A Selfish Link-based Incentive Mechanism for Unstructured Peer-to-Peer Networks. In *Proceedings of the 24th IEEE international Conference on Distributed Systems*. IEEE computer Society, 2004.

von Neumann, J. and Morgenstern, O. Theory of Games and Economic Behavior. Princeton, 1944.

Zimmermann, M.G., Egufluz, V.M. and San Miguel. Cooperation, adaptation and the emergence of leadership. In A. Kirman and J.B. Zimmermann (eds.) *Economics with Heterogeneous Interacting Agents*, pp. 73-86. Berlin: Springer, 2001.