AAMAS 2010
TORONTO

The 9th International Conference on
Autonomous Agents and Multiagent Systems
May 10-14, 2010
Toronto, Canada

Workshop 15

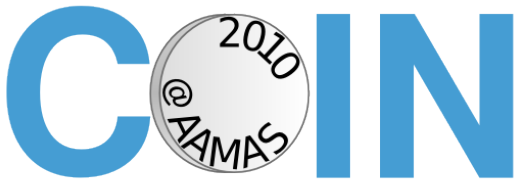
The Ninth International Workshop on Coordination, Organization, Institutions and Norms in Multi-Agent Systems

COIN 2010

Editors:
Wiebe van der Hoek
Gal A. Kaminka
Yves Lespérance
Michael Luck
Sandip Sen

Coordination, Organization, Institutions and Norms in Multi-Agent Systems

@AAMAS2010



9th International Workshop
11 May 2010, Torronto, Canada

Marina De Vos
Jeremy Pitt (Eds.)

Foreword

This collection of papers is the pre-proceedings of the 10th in the series of international workshops COIN: Coordination, Organization, Institutions and Norms in Multi-Agent Systems, held in Toronto on May 11, 2010, as part of the Workshop Programme associated with the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS).

Autonomous and autonomic management of the scale and complexity of contemporary distributed systems requires intelligence; in particular an intelligence that is manifested by individual strategies and/or collective behaviour. In such circumstances, system architects have to consider: the inter-operation of heterogeneously designed, developed or discovered components; inter-connection which cross legal, temporal, or organizational boundaries; the absence of global objects or centralised controllers; the possibility that components will not comply with the given specifications; and embedding in an environment which is likely to change, with possible impact on individual and collective objectives.

The convergence of the requirement for intelligence with these operational constraints demands: coordination: the collective ability of heterogeneous and autonomous components to arrange or synchronise the performance of specified actions in sequential or temporal order; organization: a formal structure supporting or producing intentional forms of coordination; institution: an organization where (inter alia) the performance of designated actions by empowered agents produces conventional outcomes; and norms: standards or patterns of behaviour in an institution established by decree, agreement, emergence, and so on.

The automation and distribution of intelligence is the subject of study in autonomous agents and multi-agent systems; the automation and distribution of intelligence for coordination, organization, institutions and norms is the focus of COIN@AAMAS2010.

The goal of the workshop is to bring together researchers in autonomous agents and multi-agent systems working on the scientific and technological aspects of organizational theory, electronic institutions and computational economies from an organizational or institutional perspective. The papers in this collection address various different mathematical, logical and computational perspectives of, and modelling, animation and simulation techniques for, these types of multi-agent system. However, it is to be noted that, as befits a workshop, some of these papers are in an early state of development, and they are being evaluated on their contribution to stimulate discussion and participation as well as their technical content.

Through various information and opinion exchange mechanisms before, during and after the workshop, we aim to lift these works to a higher state of technical execution, as well as aiming to generate new ideas, consolidate and develop an (already) active community, highlight future challenges and opportunities, and lead/define (at least part of) the future research agenda. Improved papers will be subject to a second round of review and if accepted, will merge with the outcomes of the second COIN workshop in 2010, to produce a final proceedings to be published with Springer.

The Workshop Organisers would like to thank the COIN Steering Committee for entrusting the organisation of the workshop to us; for a successful series it a welcome but serious responsibility. We are then dependent on, and duly thankful to, the COIN@AAMAS2010 Programme Committee for their prompt, timely and diligent reviewing. We'd also like to thank authors for their interest in the workshop, and thank them (in advance) for their participation in what we hope will be a lively, engaging and informative event. Finally, we'd particularly like to thank Kagan Tumer, AAMAS 2010 Workshop Chair, for his continuous friendly support and encouragement throughout the organization process.

March 2010

Marina De Vos
Jeremy Pitt Organisers
COIN@AAMAS2010

Organisation

Workshop Chairs

Marina De Vos	(University of Bath, UK)
Jeremy Pitt	(Imperial College London, UK)

Programme Committee

Alexander Artikis	(National Centre for Scientific Research Demokritos, Greece)
Guido Boella	(University of Torino, Italy)
Olivier Boissier	(ENS Mines Saint-Etienne, France)
Dan Corkill	(University of Massachusetts Amherst, USA)
Antonio Carlos da Rocha Costa	(UCPEL, Brazil)
Stephen Cranefield	(University of Otago, New Zealand)
Virginia Dignum	(Delft University of Technology, The Netherlands)
Nicoletta Fornara	(University of Lugano, Switzerland)
Olivier Gutknecht	(LPDL, France)
Jomi Fred Hubner	(Federal University of Santa Catarina, Brazil)
Fuyuki Ishikawa	(National Institute of Informatics, Japan)
Catholijn Jonker	(Delft University of Technology, The Netherlands)
Christian Lemaitre	(Universidad Autonoma Metropolitana, Mexico)
Maite Lopez-Sanchez	(University of Barcelona, Spain)
Eric Matson	(Purdue, USA)
John-Jules Meyer	(Utrecht University, The Netherlands)
Daniel Moldt	(University of Hamburg, Germany)
Pablo Noriega	(IIIA-CSI, Spain)
Eugenio Oliveira	(Universidade do Porto, Portugal)
Sascha Ossowski	(URJC, Spain)
Julian Padget	(University of Bath, UK)
Eric Platon	(National Institute of Informatics, Japan)
Juan Antonio Rodriguez Aguilar	(IIIA-CSIC, Spain)
Christophe Sibertin-Blanc	(IRIT, France)
Jaime Sichman	(University of Sao Paulo, Brazil)
Catherine Tessier	(ONERA, France)
Luca Tummolini	(ISTC/CNR, Italy)
Leon van der Torre	(University of Luxembourg, Luxembourg)
Wamberto Vasconcelos	(University of Aberdeen, UK)
Javier Vazquez-Salceda	(University Politecnica de Catalunya, Spain)
Harko Verhagen	(Stockholm University, Sweden)
George Vouros	(University of the Aegean, Greece)

COIN Steering Committee

Christian Lemaitre	(Universidad Autonoma Metropolitana, Mexico)
Eric Matson	(Purdue University, USA)
Guido Boella	(University of Torino, Italy)
Jaime Sichman	(University of Sao Paulo, Brazil)
Javier Vazquez Salceda	(Universitat Politecnica de Catalunya, Spain)
Julian Padget	(University of Bath, UK)
Jeremy Pitt	(Imperial College London, UK)
Nicoletta Fornara	(University of Lugano, Switzerland)
Olivier Boissier	(ENS Mines Saint-Etienne, France)
Pablo Noriega	(Artificial Intelligence Research Institute, Spain)
Sascha Ossowski	(Universidad Rey Juan Carlos, Spain)
Virginia Dignum	(Delft University of Technology, The Netherlands)
Wamberto Vasconcelos	(University of Aberdeen, UK)
George Vouros	(University of the Aegean, Greece)

Additional Referees

Matthias Wester-Ebbinghaus
Moser Silva Fagundes
Valerio Genovese
Inacio Guerberoff
Luis Gustavo Nardin
Henrique Lopes Cardoso
Fernando Marcellino
Dmytro Tykhonov
Matteo Vasirani
Serena Villata

Table of Contents

Reputation-based Agreements: From Individual to Collective Opinions *

Roberto Centeno
CETINIA
University Rey Juan Carlos
Madrid, Spain
roberto.centeno@urjc.es

Ramón Hermoso
CETINIA
University Rey Juan Carlos
Madrid, Spain
ramon.hermoso@urjc.es

Viviane Torres da Silva
Universidade Federal
Fluminense (UFF)
Rio de Janeiro, Brazil
viviane.silva@ic.uff.br

ABSTRACT

Reputation mechanisms have been developed during last few years as valid methods to allow agents to better select partners in organisational environments. In most of works presented in the literature, reputation is summarised as a value, typically a number, that represents an opinion sent by an agent to another about a certain third party. In this work, we put forward a novel concept of *reputation-based agreement* in order to support the reputation definition, as well as, some desirable properties about it. We define a reputation service that collects opinions from agents, so creating *agreements* over *situations*. This service will also be in charge of presenting the information by using different *informative mechanisms*. On the other hand, we analyse how to enforce agents to send their opinions to the reputation service by adding *incentive mechanisms*. Finally, two different case studies are presented to exemplify our work.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Intelligent Agents*

General Terms

Theory, Design

Keywords

Agreement, Reputation, Organisation, Trust

1. INTRODUCTION

Reputation mechanisms have been proved to be successful methods to build multi-agent systems where agents' decision-making processes to select partners are crucial for the system functioning [5][6][10]. In models such as in [6][10] the authors focus on letting the agent the duty of requesting opinions, aggregating replies and inferring conclusions from the gathered information. Although reputation gathering process from the agent's point of view is an important issue, in this work we propose a complementary approach that endows organisations with a reputation service that may help agents to make decisions when their own information is scarce.

*The present work has been partially funded by the Spanish Ministry of Education and Science under project TIN2006-14630-C03-02 (FPI grants program) and TIN2009-13839-C03-02 and by the Spanish project "Agreement Technologies" (CONSOLIDER CSD2007-0022, INGENIO 2010)

In this paper we introduce the concept of *reputation-based agreement* as the cornerstone of the reputation service in an organisational multi-agent system. An *agreement* is usually defined as a meeting of minds between two or more parties, about their relative duties and rights regarding current or future performance. Around this concept new paradigms have emerged [1][2] aimed at increasing the reliability and performance of agents in organisations by introducing in such communities these well-known human social mechanisms. With this in mind, we propose a novel approach for the meaning of reputation. From a global point of view, a *reputation-based agreement* is a meeting point on the behaviour of an agent, participating within an organisation, with regard to its reputation. Agreements are evaluated by aggregating opinions sent by participants about the behaviour of agents. Notice that this notion of agreement bases on a passive process instead of on an active one, since agreement is reached without any dialogue among agents, but with the opinions gathered from them. We also define some properties that describe different types of agreements. Besides, information about reached agreements will be provided to agents by using the concept of informative mechanism [3].

The second part of the paper tackles the problem of how to make agents to collaborate sending their opinions to the reputation service in a pro-active manner. We will examine the concept of incentive mechanism [3] as a way of manipulating participants, in order to get more collaboration sending their opinions about different situations they have been involved in.

The paper is organised as follows: Section 2 formalises the reputation service, supported by the idea of reputation-based agreements. In Section 3 we illustrate all concepts introduced by means of a case study. Section 4 puts forward an incentive mechanism that enforces agents to collaborate with the reputation service. Section 5 elaborates a second case study using an incentive mechanism to enforce agents to collaborate sending their opinions to the reputation service. Section 6 discusses some related work and, finally, Section 7 summarises the paper and presents the future work.

2. A SERVICE BASED ON REPUTATION-BASED AGREEMENTS

As we have previously pointed out, the current work faces with the task of formalising a reputation service working on organisational multi-agent systems. It is motivated because the reputation of an agent participating within an organisation varies as consequence of its behaviour with regard to the norms of such a system. That is, the violation

of norms within an organisation affects the reputation of an agent. We adhere the definition of organisation given in [4]. Summarising, an organisation is defined as a tuple $\langle \mathcal{A}g, \mathcal{A}, \mathcal{X}, \phi, x_0, \varphi, \{\mathcal{ON}^{om}, \mathcal{R}^{om}\}\rangle$ where $\mathcal{A}g$ represents the set of agents participating within the organisation; $\mathcal{A}$ is the set of actions agents can perform; $\mathcal{X}$ stands for the environmental states space; ϕ is a function describing how the system evolves as a result of agents actions; x_0 represents the initial state of the system; φ is the agents' capability function describing the actions agents are able to perform in a given state of the environment; $\mathcal{ON}^{om}$ is an organisational mechanism based on organisational norms; and $\mathcal{R}^{om}$ is an organisational mechanism based on roles that defines the positions agents may enact in the organisation (see [4] for more details).

2.1 How Agents Send Their Opinions

During an agent lifetime within an organisation, it is involved in several different *situations*. A situation is defined as a tuple $\langle \mathcal{A}g, \mathcal{R}, \mathcal{A}, T\rangle$, that represents an agent $\mathcal{A}g$, playing the role $\mathcal{R}$, while performing the action $\mathcal{A}$, through a time period T. As detailed in [4], different types of situations can be defined following this definition. For instance, situations in which an agent performs an action, regardless of the role it is playing – $\langle \mathcal{A}g, \_, \mathcal{A}, \mathcal{T}\rangle$ –, or situations in which an agent is playing a role during a time period, regardless the action it performs – $\langle \mathcal{A}g, \mathcal{R}, \_, \mathcal{T}\rangle$. Agents usually evaluate those situations in order to compile reliable information that allows them to predict the result of future situations. The rationale of the current work is that if agents share their knowledge about the situations they are involved in, this information might be useful when other agents have not enough information to select partners to interact with. This problem becomes hard when new participants join an organisation and they do not have not strong opinions yet.

Situations are evaluated from an agent's individual point of view. Thus, an evaluation may reflect the experience of the agent performing the evaluation – direct way – or the opinions provided by third parties about the evaluated situation – indirect way.

At any time, an agent can send its opinion about a particular situation to the reputation service. We call this information *reputation information message*:

Definition 1. *A reputation information message $\mathcal{R}^{info}_{ag_i \in \mathcal{A}g}$ is a tuple, representing an opinion sent by the agent ag_i to the reputation service containing an evaluation about a particular situation:*

$$\mathcal{R}^{info}_{ag_i} = \langle Sit, OpR\rangle,$$

where ag_i stands for the agent which sends the opinion; Sit is the situation being evaluated; and OpR represents the agent's opinion about the behaviour of its partner in the situation (typically a number)[1].

Therefore, an agent, by using this kind of messages, is somehow making public its opinions – evaluations – about different situations: agents, roles, etc.

[1]In our previous work [4], agents exchange each other the reasons (norms that had been violated due to particular facts) of their opinions. However, for the sake of simplicity we have eliminated such reasons, because we do not take into account them in this work.

2.2 Creating Reputation-based Agreements

In this section we intend to face the task of giving a novel approach for the meaning of reputation tackling this concept as a partial agreement about a certain situation. When the reputation service receives reputation information messages from agents, it aggregates them creating what we have called *reputation-based agreements*. That is, the aggregation of all the opinions regarding a particular situation is '*per se*' what a set of agents – as a whole – actually think about the aforesaid situation. Thus, a reputation-based agreement represents the consensus reached in the reputation opinions space sent by a set of agents about a particular situation.

Definition 2. *A reputation-based agreement π for a particular situation, is a tuple:*

$$\pi = \langle Sit, \mathcal{A}g, OpR, t\rangle$$

where:

- Sit is the situation about the agreement is reached;
- $\mathcal{A}g$ is the set of agents that contributed to the agreement;
- OpR represents the opinion rating – whatever its representation is (qualitative, quantitative, etc.) – reached as a consequence of all opinions sent about Sit;
- t stands for the time when the agreement was reached.

Therefore, an agreement means a global opinion that a set of agents have on a certain situation. This agreement, as we put forward in the next section, can be used as a generalist expectation for a situation in which agents have no (or little) previous information about.

As we have claimed, a reputation-based agreement is reached as consequence of the aggregation of all opinions sent about a particular situation. Thus, the reputation service requires a function that is able to aggregate information reputation messages sent by agents. The aim of such a function is to create agreements from reputation opinions that agents send to the service by means of reputation information messages. We formally define the function as follows:

Definition 3. *Let f_π be a function that given all the reputation information messages sent by agents and a particular situation creates a reputation-based agreement for that situation:*

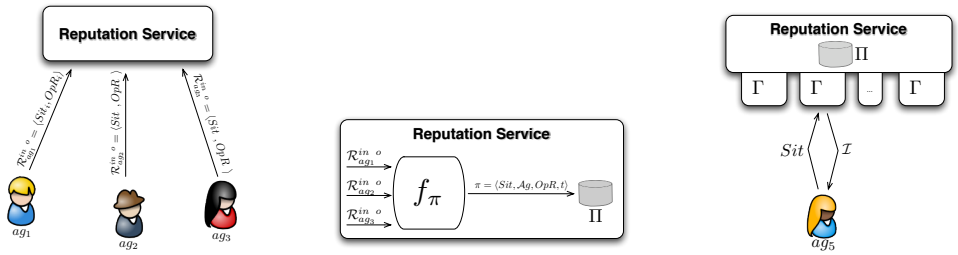
$$f_\pi : |\mathcal{R}^{info}_{ag_i \in \mathcal{A}g}| \times Sit \rightarrow \Pi$$

where:

- $|\mathcal{R}^{info}_{ag_i \in \mathcal{A}g}|$ stands for the set of reputation information messages received by the reputation service;
- Sit is the set of situations;
- Π represents the set of reputation-based agreements.

Some desirable characteristics should be taken into account when a function is designed. Therefore, we propose the following guidelines:

- an agreement about a situation should be updated when new reputation information messages are sent to the reputation service about the same situation;



(a) sending opinions (b) creating agreements (c) giving information

Figure 1: Dynamics of the Reputation Service

- the function should take into account the temporality of the messages received, that is, two opinions should not have the same importance if they were sent in different moments;
- when an agreement about a specific situation is being created, the function should also consider opinions about a more general situation. Let us illustrate it by an example. If a reputation-based agreement is being created about the situation $\langle Harry, seller, selling - shoes, t\rangle$, the function should take into account opinions about situations such as: $\langle Harry, seller, \_, t\rangle$ or $\langle Harry, \_, \_, t\rangle$.

Following these guidelines the reputation service might use any function that is able to aggregate values. It could use a simple function to calculate the average of all opinions, or a more elaborated one that aggregates the opinions by means of complex calculation, i.e., weighting the assessment by taking into account who is giving the information.

2.3 Reputation-based Agreements: Properties

From previous definitions (2 and 3) it is possible to define some desirable properties about reputation-based agreements. These properties should be taken into account when agreements are created and may also provide useful extra information when informing about different issues.

PROPERTY 1. *A reputation-based agreement π is **complete** iff all agents participating in an organisation, at time t, contribute to reach that agreement:*

$$\pi^* \Leftrightarrow \begin{cases} \mathcal{O} = \langle \mathcal{A}g, \mathcal{A}, \mathcal{X}, \phi, x_0, \varphi, \{\mathcal{ON}^{om}, \mathcal{R}^{om}\}\rangle \ \wedge \\ \pi = \langle Sit, \mathcal{A}g', OpR, t\rangle \ \wedge \\ (\mathcal{A}g = \mathcal{A}g') \end{cases}$$

That is, given a time t every participant $ag \in \mathcal{A}g$ in the organisation $\mathcal{O}$ has necessarily sent a reputation information message indicating its opinion about the situation concerning the agreement ($\mathcal{A}g = \mathcal{A}g'$). More complete agreements mean more reliable the information in the system.

PROPERTY 2. *A reputation-based agreement π is α**-consistent** iff the reputation value of π differs, at most, $1-\alpha$ from the reputation value sent by every agent that contributed to reach that agreement:*

$$\pi^\alpha \Leftrightarrow \begin{cases} \pi = \langle Sit, \mathcal{A}g, OpR, t\rangle \ \wedge \\ \forall ag \in \mathcal{A}g \ \ [\forall r \in Rep_{ag}^{info}[(r = \langle Sit_i, OpR_i\rangle) \wedge \\ (Sit_i = Sit) \ \wedge \ (|OpR_i - OpR| \leq 1-\alpha)]] \end{cases}$$

This property represents how agents sending their opinions about a situation agree in a certain extent. Therefore, the higher α is, the more similar the opinions are.

PROPERTY 3. *A reputation-based agreement π is **full** iff it is complete and 1-consistent:*

$$\pi^\phi \Leftrightarrow (\pi^* \ \wedge \ \pi^\alpha \ \wedge \ \alpha = 1)$$

In the case α is 1 all agents have the same opinion about a given situation. This property is very desirable when seeking reputation-based agreements, because the more agents contribute to the agreement, the stronger validity the latter gets. Thus, the likelihood of capturing what is actually happening in the organisation tends to be higher.

PROPERTY 4. *A reputation-based agreement π is $\mathcal{R}$**-consistent** iff all the agents participating in the agreement play the same role in the system:*

$$\pi_{\mathcal{R}} = \langle Sit, \mathcal{A}g, OpR, t\rangle \Leftrightarrow \forall ag \in \mathcal{A}g \ \ play(ag, \mathcal{R})$$

where $\mathcal{R}$ stands for the role the consistency is based on, $\mathcal{A}g$ is the set of agents that contribute to reach the agreement, and $play : \mathcal{A}g \times \mathcal{R} \rightarrow [true, false]$ is a function that returns $true$ if the agent $\mathcal{A}g$ plays the role $\mathcal{R}$.

This property is useful in cases in which a new agent, joining an organisation, wants to know what other agents – that are executing in the organisation and playing the same role – think about a given situation. For instance, someone who is thinking of *buying* something would like to know which are the opinions of those who have previously played the role *buyer*.

PROPERTY 5. *A reputation-based agreement π is $\mathcal{R}$**-complete** iff it is $\mathcal{R}$-consistent and is complete for all the agents that play the role $\mathcal{R}$ at time t:*

$$\pi^*_{\mathcal{R}} = \langle Sit, \mathcal{A}g', OpR, t\rangle \Leftrightarrow$$
$$\begin{cases} \pi_{\mathcal{R}} \wedge \mathcal{O} = \langle \mathcal{A}g, \mathcal{A}, \mathcal{X}, \phi, x_0, \varphi, \{\mathcal{ON}^{om}, \mathcal{R}^{om}\}\rangle \wedge \\ \forall ag \in \mathcal{A}g \ \ (play(ag, \mathcal{R}) \rightarrow ag \in \mathcal{A}g') \end{cases}$$

PROPERTY 6. *A reputation-based agreement π is $\mathcal{R}$**-full** iff it is $\mathcal{R}$-complete and is 1-consistent:*

$$\pi^\phi_{\mathcal{R}} \Leftrightarrow (\pi^*_{\mathcal{R}} \ \wedge \ \pi^\alpha \ \wedge \ \alpha = 1)$$

Although properties 1 and 3 are desirable, they are not achievable in systems that have a significant number of agents, for instance, in electronic marketplaces. However, many systems have those properties, such as closed organisational systems where the number of participants is not huge.

2.4 Providing Information about Reputation-based Agreements

Once we have defined an agreement as a distributed consensus-based expectation for a set of agents on a certain situation, we now describe how the reputation service can present the relevant information about the reached agreements to the agents participating in the organisation. Reputation-based agreements somehow capture the general thinking about a particular situation – the more α-consistent the agreement is the more reliable it is. Thus, information about the agreements reached until that moment may be very useful for agents. In particular, when agents have recently joined the organisation, they do not have any hint about situations in which they might be involved in, so if the reputation service provides information about agreements, agents may improve their utility from the very beginning.

With this in mind, we deal with the problem of how the reputation service may provide such information. To that end, we part from the notion of *informative mechanism* [3]. Those types of mechanisms are in charge of providing some kind of information to agents in order to regulate a multi-agent system. Thus, an *informative mechanism* $\Gamma : \mathcal{S}' \times \mathcal{X}' \rightarrow \mathcal{I}$ is a function that given a partial description of an internal state of an agent (S') and, taking into account the partial view that the service has of the current environmental state ($\mathcal{X}'$), provides certain information ($\mathcal{I}$).

We formally define them as follows:

Definition 4. *An informative mechanism providing information about reputation-based agreements is:*

$$\Gamma_\Pi : Sit \times \mathcal{X}' \rightarrow \mathcal{I}_\Pi$$

where Sit and $\mathcal{X}'$ are already defined and $\mathcal{I}_\Pi$ stands for the information provided by the mechanism by using the set of agreements Π reached over the situation Sit.

We have chosen a very general definition of information in order to cover all possible types of information the reputation service could offer taking into account the reputation-based agreements reached. The information provided may consist of (i) a ranking sorting the best agents for a particular situation, such as $\langle \_, \mathcal{R}, \mathcal{A}, \_ \rangle$, created from the agreements reached for that situation, (ii) a value representing the reputation value for a situation, reached as a consequence of the agreement for that situation, (iii) an information about the properties of the agreement reached for a particular situation, if it is full, complete, etc.

Notice that we consider agents as rational entities capable of choosing which informative mechanisms to ask for information depending of, e.g. its own preferences or basing on past requests.

3. CASE STUDY: PUBS AREA

In this section, we illustrate the proposed model by means of a simple case study. The scenario we use involves five different agents: *Anna*, *John*, *Jessica*, *Albert* and *Harry* participating within an organisation. In this organisation agents can *order* and *delivery* drinks, so the action space of agents is composed of actions such as, *order-1000-drink-a*, *delivery-2000-drink-b*, where a and b represent the type of the drink agents order/deliver. That organisation is created with the aim of getting in touch pubs' owners and providers of drinks. Thus, agents join the organisation playing the roles of *pub* and *provider*, representing a pub's owner and a company provider of drinks, respectively. In our particular example, agents are playing the following roles: *Anna - pub*, *John - pub*, *Jessica - pub*, *Albert - provider* and *Harry - provider*.

In this scenario, agents representing pubs' owners are interested in collaborating by sharing information about providers. The pubs are situated in the same area and they collaborate with each other so as to foster the attraction customers to that area. That is, although they try to maximise their own benefits, one of their goals is to foster the pubs area where they are, even if that entails to exchange information about drink providers.

Therefore, after several interactions among them – performing actions of ordering and delivering different types of drinks – *Anna* decides to make public her opinion about *Albert* and *Harry* as providers. Thus, she uses the reputation information messages to send to the reputation service her opinions, as follows:

$$\mathcal{R}^{info}_{Anna} = \langle \langle Albert, provider, \_, \_ \rangle, 0.2 \rangle$$
$$\mathcal{R}^{info}_{Anna} = \langle \langle Harry, provider, \_, \_ \rangle, 0.9 \rangle$$

This information shows that *Anna* has had bad experiences while she was ordering drinks from *Albert* (0.2)[2] because *Albert* always delivers all drinks later than the agreed date. Otherwise, the second message shows that she has had good experiences with *Harry* (0.9) because *Harry*, for instance, never violates contracts and offers low prices. Similarly, *John* and *Jessica* send their opinions about *Albert* and *Harry* as providers, by using the following messages:

$$\mathcal{R}^{info}_{John} = \langle \langle Albert, provider, \_, \_ \rangle, 0.2 \rangle$$
$$\mathcal{R}^{info}_{John} = \langle \langle Harry, provider, \_, \_ \rangle, 0.8 \rangle$$
$$\mathcal{R}^{info}_{Jessica} = \langle \langle Albert, provider, \_, \_ \rangle, 0.2 \rangle$$

It seems that both *John* and *Jessica* agree that *Albert* is not a reliable provider. However, *Harry* is quite reliable delivering drinks, from *John*'s point of view.

When the reputation service receives this information, it is able to create reputation-based agreements by using a function that aggregates the reputation information messages. Let us suppose that it aggregates the messages by calculating the average of reputation values sent by agents over exactly the same situation[3]:

$$f_\pi(Sit) = \frac{\sum_{i=1}^{n} \mathcal{R}^{info}_{ag_i} = \langle Sit, OpR_i \rangle}{n}$$

Therefore, from the set of messages sent by the agents, so far, the reputation service can create two reputation-based agreements regarding to two different situations:

$$\pi_1 = \langle \langle Albert, provider, \_, \_ \rangle, \{Anna, John, Jessica\}, 0.2, t \rangle$$
$$\pi_2 = \langle \langle Harry, provider, \_, \_ \rangle, \{Anna, John\}, 0.85, t \rangle$$

π_1 represents that there exists an agreement within the organisation regarding to *Albert* as *provider* – regardless the action he performs – is evaluated as 0.2, and such an agreement is reached by the collaboration of *Anna*, *John* and *Jessica*, at time t. Besides, π_2 shows that there exists an agreement in which Harry is evaluated 0.85 – it is calculated as the mean of all opinions sent – as *provider* and that the agreement is reached by *Anna* and *John*, at time t.

In order to provide information about agreements the reputation service offers three different informative mechanisms:

[2] We suppose that reputation values – denoted by OpR – are in the range [0..1]

[3] It could be used whatever other function that is able to aggregate the information received from agents

- $\Gamma^1_\Pi(\langle \mathcal{A}g, \mathcal{R}, \_, \_\rangle)$ given a situation where an agent and a role are specified, it returns meta-information[4] about the agreement reached regarding that situation;
- $\Gamma^2_\Pi(\langle \mathcal{A}g, \mathcal{R}, \_, \_\rangle)$ given a situation where an agent and a role are specified, it returns the reputation-based agreement reached. In particular, it returns the reputation value in the agreement of that situation;
- $\Gamma^3_\Pi(\langle \_, \mathcal{R}, \_, \_\rangle)$ given a situation where a role is specified, it returns a ranking of agents playing that role, sorted by the reputation value they have as consequence of the reputation-based agreements reached until the current time t.

Let us suppose that a new pub is opened in the same area by *Alice*, so she joins the organisation playing the role *pub*. Since the pub is recently open, she needs to order drinks. Thus, she should select a provider of drinks but she does not know any provider yet. One solution could be asking to another agent about a particular provider – distributed reputation mechanism. However, this process could be very costly because she would require many queries sent to different agents to ask about different providers. Another solution is to use informative mechanisms to get information about other participants, in this case about providers. Thus, *Alice* searches for an informative mechanism that provides a ranking of "best" providers[5]. She finds Γ^3_Π that returns a ranking of agreements when it is queried based on a situation and a role. So, *Alice* performs the following query to Γ^3_Π: $\Gamma^3_\Pi(\langle \_, provider, \_, \_\rangle) \Rightarrow \{Harry, Albert\}$ and the informative mechanism returns a ranking of agents, sorted by the reputation values according to all reputation-based agreements reached at that moment, by matching the situation specified in the query with the situation of agreements.

By using this information *Alice* knows that there exists an agreement within the organisation showing that *Harry* is a better provider than *Albert*. But, how good are they? To answer this question *Alice* queries the informative mechanism Γ^2_Π as follows:

$$\Gamma^2_\Pi(\langle Harry, provider, \_, \_\rangle) \Rightarrow 0.85$$
$$\Gamma^2_\Pi(\langle Albert, provider, \_, \_\rangle) \Rightarrow 0.2$$

In that moment, *Alice* is quite sure that *Harry* is much better provider than *Albert* and there exists an agreement, within the organisation, that *Harry*'s reputation delivering drinks is 0.85 and another one that *Albert* as provider is 0.2. However, *Alice* is still doubting about which provider could be the best, because she is wondering how consistent those agreements are. Thus, she queries the informative mechanism that provides meta-information about the agreement reached regarding a given situation. Therefore, she performs the following queries:

$$\Gamma^1_\Pi(\langle Harry, provider, \_, \_\rangle) \Rightarrow \pi^{0.95}$$
$$\Gamma^1_\Pi(\langle Albert, provider, \_, \_\rangle) \Rightarrow \pi^1$$

With this information *Alice* clears all her doubts. She knows that all opinions sent about *Albert* are coincident because the reputation-based agreement reached is 1-consistent (π^1). Besides, she knows that the opinions sent by the agents that have interacted with *Harry* are almost the same since their variability is low (0.95-consistent). With this in mind, *Alice*, finally, selects *Harry* as her provider.

In this domain, the reputation service has been an useful mechanism allowing *Alice* to select a good provider, when she did not have any previous information about providers. The mechanism worked perfectly due to agents participating in the organisation collaborated by sending their opinions to the reputation service. They were motivated to send opinions because of the own nature of the domain – pub's owners wants to create a pubs area to attract customers. Thus, pubs get benefits individually from making public their opinions about providers. But what happens when agents are not motivated by the domain to send opinions to the reputation service? Next section deals with that problem.

4. HOW TO MOTIVATE AGENTS TO SEND THEIR OPINIONS

As we have mentioned before, this section deals with the problem of motivating agents to send their reputation information messages to the system. Thus, we propose to endow the reputation system with an *incentive mechanism* [3] so as to face this task. Following the work in [3], an incentive mechanism is formalised as a function that, given a possibly partial description of an environmental state of a multiagent system, produces changes in the transition probability distribution of the system: $\Upsilon_{inc} : \mathcal{X}' \rightarrow [\mathcal{X} \times \mathcal{A}^{|\mathcal{A}g|} \times \mathcal{X} \rightarrow [0..1]]$, where $\mathcal{X}'$ stands for the partial description of an environmental state; and $\mathcal{X} \times \mathcal{A}^{|\mathcal{A}g|} \times \mathcal{X} \rightarrow [0..1]$ is the transition probability distribution of the system, that describes how the environment ($\mathcal{X}$) evolves as a result of agents' actions ($\mathcal{A}^{|\mathcal{A}g|}$) with certain probability in $[0..1]$. Hence, an incentive mechanism, producing changes in the transition probability distribution of the system, may produce changes in the consequences of agents' actions by introducing rewards and/or penalties. That is, when a mechanism is able to modify the consequences of an action, such a modification might become in a reward or a penalty for the agent. For instance, a mechanism that installs radars over a road, is an incentive mechanism, since the probability of a car – an agent – to get fined (and, thus, the probability to change to a state with less money) is higher if the car passes at prohibited velocity than without the radar. Thus, the mechanism changes the consequences of the action *passing a road at high velocity*.

Any incentive mechanism must tackle the following requirements: *i)* to choose the agents that will be affected by the mechanism; *ii)* to select the actions in which the incentive will be applied; *iii)* to find out, at least one attribute and a possible modification of this, that affects the preferences of each selected agent[6]; and finally, *iv)* to apply the modification of the parameters selected in the step *iii)* as a consequence of the selected actions in the step *ii)*, giving in such a way a reward or a penalty to these agents. Formally, an incentive mechanism is composed of $\Upsilon_{inc} = \langle \mathcal{A}g_{inc}, \bigtriangledown, \omega_{inc} \rangle$, where $\mathcal{A}g_{inc}$ is the set of agents that will be applied for the incentive mechanism, $\bigtriangledown$ stands for the set of actions in which the incentive will be applied and ω_{inc} represents the set of attributes and their selected modification to tune up the preferences of agents $\mathcal{A}g_{inc}$. Each attribute is formalised

[4]Meta-information means the α-consistency of the agreement, i.e., if it is full, complete, etc.

[5]We suppose that informative mechanisms are made available to all participants by the organisation

[6]It means that this attribute affects to the utility function of the agent. Because we assume that such preferences are expressed by means of an utility function.

as a tuple $\langle attribute, value \rangle$.

In our particular case, we are interested in motivating all agents participating in the organisation, because the more information the system has, the more complete the reputation-based agreements will be, and consequently, the more useful the information provided will be as well. Thus, all agents within the organisation will be affected by the incentive mechanism: $\mathcal{A}g_{inc} = \mathcal{A}g$ (requirement *i)*). As we have pointed out, all agents should be motivated to send their opinions to the system. Thus, the action of *sending reputation information messages* to the system has to be affected by the incentive mechanism (requirement *ii)*). Therefore, $\bigtriangledown = \{send(\mathcal{R}_{ag_i}^{info})\}$. In order to find out the attributes that affect agents' preferences – requirement *iii)* – there exist two different alternatives: *a)* discover the attributes by observing the performance of agents, by modifying – the mechanism – attributes randomly, what could be very costly; and *b)* introduce a new attribute in the system, becoming an attribute that influences the agents' preferences. We tend to favour the second option, introducing a new attribute to the organisation: *points*. That is, each agent is assigned with an amount of points when it joins the organisation and the incentive mechanism will modify their amont. Formally: $\omega_{inc} = \langle points_{ag_i}, value \rangle$

Furthermore, agents within an organisation must select partners to interact with, so whatever the domain of the organisation is, such agentes are interested in selecting the best partners. Thus, their utility is influenced by the selection of such partners. So, since the reputation system might provide them useful information to that end, if we associate the new attribute with the action of querying that information, the new attribute somehow becomes in an attribute that influences the agents' preferences. Therefore, the action of querying an informative mechanism has to be affected by the incentive mechanism as well. Formally: $\bigtriangledown = \{send(\mathcal{R}_{ag_i}^{info}), query(\Gamma_\Pi)\}$.

In order to complete requirement *iv)* the mechanism must decide how to modify the attribute introduced – the points each agent has – as a consequence of the actions selected to receive an incentive – to send opinions and to query information. If the incentive mechanism does not exist, agents will be interested in performing the action of querying an informative mechanism – because they might get useful information –, but they will not be interested in performing the action of sending their opinions – because they might lose utility if they make public such opinions. Hence, the mechanism has to get the opposite effect, that is, it should make more attractive the sending of opinions and less attractive the querying for information. In this way, the new attribute becomes an attribute whose modification affects the agents' preferences.

In the case of the action $send(\mathcal{R}_{ag_i}^{info})$, the mechanism should make more attractive the state in which an agent will be, when it performs that action, since they are not interested in performing it. Thus, when an agent $ag_i \in \mathcal{A}g_{inc}$ performs the action $send(\mathcal{R}_{ag_i}^{info})$ the consequence of such an action will be a modification of the value of the attribute $\omega_{inc} = \langle points_{ag_i}, value \rangle$, such that:

$$value = value + (\alpha_1 x + \alpha_2)$$

where x is the number of new reputation-based agreements that will be created with the new opinion sent and α_1, α_2 are parameters to weight the incentive ($\alpha_1 > 0$, $\alpha_2 > 0$).

We are inspired by the market and the law of demand and offer, that is, the price of a service/product is fixed based on the demand and offer this service/product has. Thus, the points an agent gets when it sends an opinion depends on how further the new opinion is. This is measured by calculating the number of reputation-based agreements it creates – parameter x in the equation. It fluctuates between 0 and 1, when an agent sends an opinion about a situation that an agreement was not reached so far, it creates as maximum one new agreement. In such a way, when more new opinions are sent, more points agents get, so consequently, agents will be motivated to send new opinions.

On the other hand, when an agent $ag_i \in \mathcal{A}g_{inc}$ performs the action $query(\Gamma_\Pi)$, the attribute $\omega_{inc} = \langle points_{ag_i}, value \rangle$ is modified as a consequence in the following way:

$$value = value - (\alpha_3 y + \alpha_4)$$

where y stands for the demand the informative mechanism being queried by the agent has. It is calculated by the number of times such a mechanism is queried; and α_3, α_4 are parameters to weight the incentive, such that $\alpha_3 > 0$ and $\alpha_4 > 0$.

Following the simile of the law of offer and demand, the more an informative mechanism is queried, the more points agents lose querying it. Therefore, it supposes that the more an informative mechanism is queried, the more useful is the information it provides. Thus, its price will be risen and consequently, agents will need more points to query. Since the only way to get points is sending opinions, agents are definitely motivated to send information. Hence, a market of points is created giving incentives to agents to share their opinions. It is important to notice that the attribute $points_{ag_i}$ cannot be negative. That is, if the consequence of performing the action $query(\Gamma_\Pi)$ was a negative value the information will not be provided.

In order to solve a deadlock produced when agents join the organisation without any points, the incentive mechanism assigns to agents an amount of "trial" points. It should assign, at least, a minimum quantity of points to make them able to query the informative mechanism, since the available information might be useful for them. Therefore, the attributes will be initialised as follows: $\omega_{inc} = \langle points_{ag_i}, value \rangle$ where $value = \alpha_3 n + \alpha_4$ such that, α_3 and α_4 are the same as the modification of the attribute when agents query an informative mechanism; and n is the number of "free" queries the incentive mechanism assigns when agents join the organisation.

Finally, it is important to remark that those incentives should be published so as to be effective. We suppose that the organisation also publishes them together with the informative mechanisms. In Section 5 we illustrate how the reputation service, coupled with this incentive mechanism, works in domains in which agents are not motivated to share their opinions.

5. CASE STUDY: TASKS SERVERS

In this section we put all together by illustrating the dynamics of the reputation service working with the incentive mechanism introduced in the same system. This scenario involves the same five agents: *Anna*, *John*, *Jessica*, *Albert* and *Harry* participating within another organisation. In this organisation, agents can execute different tasks when another agent requests it. Thus, agents can join the organisation playing two different roles: *servers* and *customers*. The formers are able to execute the tasks that customers request

them. This domain is characterised by the following aspects: *i)* when a tasks server is overload it is not able to execute more tasks; *ii)* each server has different capabilities to execute a task, i.e., the quality of the executed task may be different; and *iii)* when a task is assigned to a server and the quality of the executed task is not good enough, the task could be required to be executed again by another server. With this characteristics agents are not motivated to send their opinions, because if all agents discover the best server, the latter will unrelentingly be overload. On the other hand, if agents do not have any hint about the best servers for each task, they will select the partners – the servers – randomly, and it could imply a loss of utility, because they could need to repeat the request to a different server. Obviously, the reputation service, without the incentive mechanism, will not work in this domain, because agents will not send their opinions and, as a result, reputation-based agreements will not be created. Therefore, the organisation will be endowed with an incentive mechanism set up as follows:

$$\Upsilon_{inc} : \langle \mathcal{A}g_{inc} = \{Anna, John, Jessica, Albert, Harry\}, \\ \bigtriangledown = \{send_{ag_i}(\mathcal{R}^{info}_{ag_i}), query_{ag_i}(\Gamma_\Pi)\}, \\ \omega_{inc} = \{\langle points_{ag_i}, 130\rangle\}\rangle$$

where ag_i stands for an agent in the set $\mathcal{A}g_{inc}$; and the number 130 represents the initial points, calculated by the equation $value = \alpha_3 n + \alpha_4$ with $n = 2$ and α_3, α_4 as we will detail next. In order to modify the value of the attributes, as consequence of the execution of actions in $\bigtriangledown$, the mechanism is configured with the following parameters: $\alpha_1 = 200$ $\alpha_2 = 100$ $\alpha_3 = 10$ $\alpha_4 = 50$.

In addition, agents join the organisation playing the following roles: *Anna - customer*; *John - customer*; *Jessica - customer*; *Albert - server* and *Harry - server*. Within the organisation there are many agents playing the server role as well, but for the sake of simplicity we do not detail them.

After this point, agents start to interact by selecting their partners randomly, because although they have enough points to query informative mechanisms, there are not agreements available yet, since agents have not send their opinions. Since bootstrapping of the incentive mechanism is out of the scope of the paper we decided to assign no points to agents forming the initial state of the scenario. Thus, initial agents will have a value of 0 in the corresponding attribute of points. Otherwise, newcomers will have a value of 130, calculated as we have explained before.

When several interactions have been carried out, *customer* agents are aware of some useful information about *servers*. Once agents run out of points they need to send their opinions, so as to get some points in order to keep on querying. Thus, the following reputation information messages are sent to the service:

$$\mathcal{R}^{info}_{Anna} = \langle\langle Albert, server, \_, \_\rangle, 0.9\rangle \\ \mathcal{R}^{info}_{John} = \langle\langle Albert, server, \_, \_\rangle, 0.9\rangle \\ \mathcal{R}^{info}_{Jessica} = \langle\langle Albert, server, \_, \_\rangle, 0.9\rangle \\ \mathcal{R}^{info}_{John} = \langle\langle Harry, server, \_, \_\rangle, 0.9\rangle \\ \mathcal{R}^{info}_{Anna} = \langle\langle Harry, server, \_, \_\rangle, 0.2\rangle$$

As a consequence of executing those actions, the incentive mechanism modifies the values of the attribute $points_{ag_i}$ by using the equation explained in Section 4, as follows: $\langle points_{Anna}, 400\rangle$, $\langle points_{John}, 400\rangle$ and $\langle points_{Jessica}, 100\rangle$, since Anna and John contributed to create a new agreement when they sent their opinions. However, Jessica did not contribute to any new agreement.

When the reputation service receives those messages, the following reputation-based agreements are created, by using the same function – the average function – that in the case study explained in Section 3:

$$\pi_1 = \langle\langle Albert, server, \_, \_\rangle, \{Anna, John, Jessica\}, 0.9, t_1\rangle \\ \pi_2 = \langle\langle Harry, server, \_, \_\rangle, \{Anna, John\}, 0.55, t_1\rangle$$

These agreements show that all agents playing the role *customer* think the same about *Albert* as server and that Harry is evaluated with 0.55 as *server*, from the point of view of *Anna* and *John*. We suppose that the reputation service has the same informative mechanisms as in the example of Section 3: $\Gamma^1_\Pi(\langle \mathcal{A}g, \mathcal{R}, \_, \_\rangle)$, $\Gamma^2_\Pi(\langle \mathcal{A}g, \mathcal{R}, \_, \_\rangle)$ and $\Gamma^3_\Pi(\langle \_, \mathcal{R}, \_, \_\rangle)$.

At this point, our agents could query any informative mechanism because they have enough points to do it. However, we will focus on a new agent – *Alice* – that joins the organisation at this moment. Since she is a newcomer, she will be assigned 130 points – two "free" queries to an informative mechanism. Then, she performs the following queries:

$$\Gamma^3_\Pi(\langle \_, server, \_, \_\rangle) \Rightarrow \{Albert, Harry\} \\ \Gamma^2_\Pi(\langle Albert, server, \_, \_\rangle) \Rightarrow 0.9$$

After these queries *Alice* knows that the best server, according to the reputation-based agreements reached, so far, is *Albert*, with 0.9 evaluation. Then, the value of the attribute $points_{Alice}$ will be modified to 10, following the equation explained in Section 3. Since she cannot query again due to the lack of points, she will select *Albert* as server. When she performs the interaction with Albert, she wants to get more information about how the agreement about *Albert* is, because she is wondering if such an agreement is not consistent enough, she could have problems if she assigns a different task to *Albert*. Thus, she needs to send her opinion in order to get more points. After that, she gets 310 points (10 that she already has plus 300 that she gets sending an opinion about a situation that does not form part of an agreement yet). Now, she can perform a new query: $\Gamma^1_\Pi(\langle Albert, server, \_, \_\rangle) \Rightarrow \pi^*_{customer}$. With this new information, she knows that *Albert* is the best server and all customers evaluate him exactly with the same reputation ($\pi^*_{customer}$ is *customer-complete*). This information is worthy for *Alice* because she will not need to change of server.

6. RELATED WORK

In this paper we do not propose another reputation mechanism but a reputation service that generates agreements based on collected opinions about the reputations of agents in a given situation. Such service can be used by the centralised part of an hybrid reputation model [13] or by an agent participating in a distributed mechanism that is interested in aggregate the opinions it has received about its behaviour or in aggregate the opinions it has about the behaviour of another agent, for instance.

One of the main advantages of having a centralised reputation service is the feasibility for an individual to know a more consistent reputation about another agent based on numeral experiences. In the case of distributed mechanisms (such as [11][6][10]), the agent itself would need to participate in several interactions with the given agent and also to ask other agents for their experiences with others. In the case of a

centralised mechanism, the agent can easily get information about the reputation showing the behaviour of other agents within the system. In [12], Sabater et al. present a centralised reputation mechanism that is incorporated as a service in Electronic Institutions (EIs). From a global perspective, this work has many similarities with ours, since uses also a reputation service in an organisational environment (EIs). However, the authors do not focus on how to exploit the collected information as agreements that can be presented to agents in different ways.

Regarding the incentive mechanisms existing in literature to drive agents to collaborate by sending their truthfully opinions a well known work is the one by Jurca and Faltings [8]. They use a mechanism of buy/sell information using credits. The main difference is that they use it in a distributed environment, where agents send opinions among them, but not to a central repository as in our case. In [7] the authors present an approach to create rankings able not only to provide the most trustful agents but also a probabilistic evidence of such reputation values. Those rankings are also computed by a centralised system by aggregating the reputations reported by the agents. This approach and the one presented in our paper could be complementary, since that paper focuses on defining the ranking algorithms and ours focuses on describing the mechanism that allows to receive the reputation information and to provide the already evaluated agreements (for instance by using rankings). Another work that could be also complementary to the approach presented here, is the one presented in [9]. They describe the algorithm *NodeRanking* that creates rankings of reputation ratings. Therefore, our reputation service could use this algorithm so as to provide information about the reputation-based agreements reached within the organisation.

7. CONCLUSIONS AND FUTURE WORK

Summarising, this work puts forward a novel approach of reputation-based agreement concept by supporting on a reputation service that creates reputation-based agreements as aggregations of opinions sent by participants within an organisation. Besides, we also define some desirable properties that can be derived and should be taken into account when providing the information they contain. Furthermore, we also propose to use the agreements by utilising the concept of informative mechanisms [3], so providing agents with useful information. On the other hand, we propose an incentive mechanism [3] to deal with the problem of lack of collaboration from agents to send their opinions to the service. Finally, different examples have been analysed so that they illustrate how the reputation service works in two different domains: the former represents a collaborative domain where agents are interested in sharing their opinions, and the later shows a competitive domain in which the reputation service must be coupled with an incentive mechanism to motivate agents to send their opinions.

In future work we plan to experimentally test our approach by implementing a case study presented here, as well as, running several experiments comparing our approach with similar ones. We also intend to investigate new properties about reputation-based agreements to provide agents participating in an organisation with more useful information. Finally, we plan to extend the concept of reputation-based agreement by creating agreements aggregating "similar" situations, so we must go into the concept of similar situations in depth. Moreover, it is our intention to provide agreements based on more abstract (or general) situations. The agreements of a generic situation can be created based on the agreements of its specific situations. In order to do so, an ontology of action/situation could be used to represent the generic situations and the more specific ones.

8. REFERENCES

[1] H. Billhardt, R. Centeno, A. Fernández, R. Hermoso, R. Ortiz, S. Ossowski, J. Pérez, and M. Vasirani. Organisational structures in next-generation distributed systems: Towards a technology of agreement. *Multiagent and Grid Systems: An International Journal*, 2009.

[2] C. Carrascosa and M. Rebollo. Modelling agreement spaces. In *Proc. of WAT@IBERAMIA'08*, pages 79–88, 2008.

[3] R. Centeno, H. Billhardt, R. Hermoso, and S. Ossowski. Organising mas: A formal model based on organisational mechanisms. In *Proc. of SAC'09*, pages 740–746, 2009.

[4] R. Centeno, V. T. da Silva, and R. Hermoso. A reputation model for organisational supply chain formation. In *Proc. of the COIN@AAMAS'09*, pages 33–48, 2009.

[5] C. Dellarocas. *Handbook on Economics and Information Systems*, chapter Reputation Mechanisms. Elsevier, 2005.

[6] T. Huynh, N. Jennings, and N. Shadbolt. Fire: An integrated trust and reputation model for open multi-agent systems. In *Proc. of the ECAI'04*, pages 18–22, 2004.

[7] A. Ignjatovic, N. Foo, and C. Lee. An analytic approach to reputation ranking of participants in online transactions. In *Proc. of the WI-IAT'08*, pages 587–590, 2008.

[8] R. Jurca and B. Faltings. An incentive compatible reputation mechanism. In *Proc. of the AAMAS'03*, pages 1026–1027. ACM, 2003.

[9] J. Pujol, R. Sangüesa, and J. Delgado. Extracting reputation in multi agent systems by means of social network topology. In *Proc. of the AAMAS'02*, pages 467–474. ACM, 2002.

[10] J. Sabater and C. Sierra. Reputation and social network analysis in multi-agent systems. In *Proc. of the AAMAS'02*, pages 475–482. ACM Press, 2002.

[11] J. Sabater and C. Sierra. Review on computational trust and reputation models. *Artificial Intelligence Review*, 24(1):33–60, 2005.

[12] J. Sabater-Mir, I. Pinyol, D. Villatoro, and G. Cuní. Towards hybrid experiments on reputation mechanisms: Bdi agents and humans in electronic institutions. volume 2, pages 299–308, 2007.

[13] V. Silva, R. Hermoso, and R. Centeno. A hybrid reputation model based on the use of organization. In *Coordination, Organizations, Institutions, and Norms in Agent Systems IV*, volume 5428 of *LNAI*. Springer-Verlag, 2009.

Rational Strategies for Autonomous Norm Adoption

N. Criado, E. Argente, V. Botti
Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera s/n. 46022 Valencia (Spain)
email:{ncriado,eargente,vbotti}dsic.upv.es

ABSTRACT

Norms represent an effective tool for achieving coordination and cooperation among members of open systems. However, agents must be able to adopt norms autonomously. The adoption of norms can be explained by both rational and non-rational reasons. Rational motivations consider the influence of norm compliance and violation on agent's goals. In this work the BDI agent architecture has been extended in order to allow agents to adopt norms autonomously. Moreover, the implementation of rational strategies for norm adoption in this architecture is also described.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Intelligent agents

General Terms

Theory

Keywords

Norms, Reasoning, BDI agents, Multi-Context systems

1. INTRODUCTION

Maybe the most promising application of MAS technology is its usage for supporting Open Distributed Systems [13]. They are characterized by the heterogeneity of their participants, their limited trust, possible individual goals in conflict and a high uncertainty [2]. Norms represent an effective tool for achieving coordination and cooperation among members of open systems. However, norms, to become effective, must be dynamically adapted to the environmental changes and be autonomously adopted by agents [7]. Therefore, autonomous agents need strategies for determining when and how adopting norms.

The question of norm adoption has been traditionally discussed by the sociology field. Taking as a basis the work on social norms presented in [9], the norm adoption process can be justified by: i) *Rational* motivations, i.e. a norm can be fulfilled by self-interest motivations (e.g. undesirability of the sanctions, interest in the rewards) or it can be considered as suitable for common interests; (ii) *Non-Rational* reasons, which are related to emotions such as anxiety and shame which maintain social norms, honour and envy among others. Here several strategies for norm adoption based on rational motivations are described. More specifically, this work describes a BDI agent architecture which has been extended in order to allow agents to acquire norms from their environment and consider them in their decision making processes. Moreover, different strategies for a rational adoption of norms have been defined considering the facilities provided by this new normative agent architecture.

This paper is structured as follows: the first section describes the multi-context BDI proposal. Next our extension of the multi-context BDI agent architecture for normative decision making is described. Moreover, several strategies for norm adoption have been proposed. This work has been applied into the m-Water case study, in which an irrigator agent must choose between respecting or not respecting norms employing these different norm adoption strategies. Finally, conclusions and future works are detailed.

2. PRELIMINARIES

A graded BDI agent [5] is defined as a *multi-context* agent architecture [16] formed by (Figure 1 grey units): *mental* units that characterize beliefs (BC), intentions (IC) and desires (DC); and *functional* units for planning (PC) and communication (CC). Following, these units are explained:

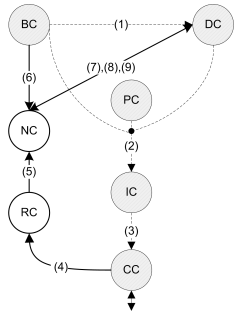


Figure 1: Multi-Context Normative BDI Architecture. Grey contexts and dashed lines (bridge rules) correspond to the basic definition of a BDI agent. The normative extensions are the white contexts and bold lines.

- *Belief Context (BC).* It is formed by propositions belonging to the BC-Logic [5]; i.e. logic propositions such as $(B\,\gamma, \beta_\gamma)$ where $B\,\gamma$ represents a belief of an agent, $\gamma \in \mathcal{L}_{\mathcal{DL}}$ is a dynamic logic [14] proposition and $\beta_\gamma \in [0, 1]$ represents the certainty degree of this belief.

- *Intention Context (IC).* It is formed by propositions belonging to the IC-Logic [5]; i.e. logic propositions such as $(I\,\gamma, \iota_\gamma)$ where $I\,\gamma$ represents an intention of an agent, $\gamma \in \mathcal{L}_{\mathcal{DL}}$ and $\iota_\gamma \in [0,1]$ represents the intention degree ascribed to this intention.
- *Desire Context (DC).* It is formed by propositions belonging to the DC-Logic [5]; i.e. logic propositions such as $(D^*\gamma, \delta^*_\gamma)$ where $D^*\gamma$ represents a desire of an agent; $\gamma \in \mathcal{L}_{\mathcal{DL}}$; $\delta^*_\gamma \in [0,1]$ represents the desirability degree; $* \in \{+,-\}$ represents positive desires and negative desires, respectively. Thus, degrees of positive or negative desires allow setting different levels of preference or rejection.
- *Planner Context (PC).* It allows agents to determine sequence of actions that will be intended according to their desires [5]. Due to space reasons, the process by which agents generate new plans for achieving their goals has not been described here.
- *Communication Context (CC).* It communicates agents with their environment [5].

According to this notation, given a proposition γ: β_γ represents the belief degree assigned to $B\,\gamma$; ι_γ is the intentionality degree of $I\,\gamma$; δ^+_γ is the desirability degree of $D^+\gamma$; and δ^-_γ is the undesirability degree of $D^-\gamma$.

2.1 Bridge Rules

Several bridge rules, which connect both the mental and functional contexts, have been defined in the existing literature in order to determine different types of BDI agents (e.g. strong realism, realism and weak realism agents [15]). Next, only those bridge rules which have an impact on the normative reasoning process are described:

- The *Deriving Concrete Desires* bridge rule (Figure 1 Rule 1) allows abstract desires to be concreted into more realistic ones according to the agent beliefs:

$$DC : (D^*\varphi, \delta^*_\varphi), BC : (B([\alpha]\varphi), \beta_\phi) \Rightarrow DC : (D^*[\alpha]\varphi, f_D(\delta^*_\varphi, \beta_\phi)) \quad (1)$$

  Generic agent desires $(D^*\varphi, \delta^*_\varphi)$ derive more realistic desires $(D^*[\alpha]\varphi, f_D(\delta^*_\varphi, \beta_\phi))$; taking into account the existence of actions that allow them to be reached $(B([\alpha]\varphi), \beta_\phi)$. Thus, the preference degree of the concrete desire relies on the original desirability (δ^*_φ) and the possibility of achieving it by means of action α (β_ϕ). This is calculated by f_D function; its concrete definition is problem dependent. However, in our proposal it is defined as the product of these two values for obtaining the expected satisfaction or disgust value.

- The *Deriving Agent Intentions From Positive Desires* bridge rule (Figure 1 Rule 2) derives the intended formulas of the agent from the set of preferred formulas which are reachable by some existing plan:

$$DC : (D^+[\alpha]\varphi, \delta^+_{[\alpha]\varphi}), DC : (D^+\alpha, \delta^+_\alpha), PC : plan(\Sigma)$$
$$DC : (D^-[\alpha]\psi_1, \delta^-_{[\alpha]\psi_1}), ..., DC : (D^-[\alpha]\psi_n \delta^-_{[\alpha]\psi_n}),$$
$$\alpha \in \Sigma, \delta^+_{[\alpha]\varphi} + \delta^+_\alpha) \geq \sum_{k=1}^{n} \delta^-_{[\alpha]\psi_k} \Rightarrow \quad (2)$$
$$IC : (I[\alpha]\varphi, f_I(\delta^+_{[\alpha]\varphi} + \delta^+_\alpha, \sum_{k=1}^{n} \delta^-_{[\alpha]\psi_k}))$$

  More specifically, those positive desires $(D^+[\alpha]\varphi, \delta^+_{[\alpha]\varphi})$ which can be achieved by an action (α) belonging to a plan $(Plan(\Sigma))$ will generate a new intention $(I[\alpha]\varphi, f_I(\delta^+_{[\alpha]\varphi} + \delta^+_\alpha, \sum_{k=1}^{n} \delta^-_{[\alpha]\psi_k}))$ if the desirability degree of both the proposition $(\delta^+_{[\alpha]\varphi})$ and the action (δ^+_α) is greater than the sum of the negative effects of the action $(\sum_{k=1}^{n} \delta^-_{[\alpha]\psi_k})$. Finally, f_I is a function that combines both positive and negative effects of an action; in this case it has been defined as: $f_I(a,b) = min(max(0, a-b), 1)$

- The *Deriving Actions From Intentions* bridge rule (Figure 1 Rule 3) defines the next action to be performed by the agent $(act(\alpha))$ as the intention which has the maximum degree $(I[\alpha]\varphi, \iota_{max})$:

$$IC : (I[\alpha]\varphi, \iota_{max}) \Rightarrow CC : act(\alpha) \quad (3)$$

3. NORMATIVE BDI ARCHITECTURE

Taking as a reference the graded BDI agent architecture [5], in [8] it has been extended by adding new units and bridge rules in order to allow agents to make decisions according to norms. In particular, two new functional contexts are defined (Figure 1, white units): the *Recognition Context* (RC), which is responsible for the norm identification process; and the *Normative Context* (NC), which allows agents to consider norms in their decision making processes.

Basically, the norm decision process starts when the RC derives a new norm through analysing agent environment. More specifically, the RC allows agents to identify norms. These abstract norms are translated into a set of inference rules which are included into the NC. Then, the NC derives new desires according to the current agent mental state and the inference rules which have been obtained from norms. These new desires may cause intentions to be updated and, as a consequence, normative actions might be carried out.

3.1 Recognition Context (RC)

Norms can be explicitly created by the system designer or a representative agent which has been empowered to define the normative context. In addition, other types of norms such as commitments are created as a result of an interaction among agents. Finally, there are norms, such as social norms, which emerge in a society without being explicitly created by any agent. Whatever process norm creation is, any new norm must be spread in the society in order to be internalized and respected by agents. In this sense, agents are able to recognise norms which control their environment by two different manners [1]: they may be informed about the existing norms or, on the contrary, cognitive agents are capable of inferring norms from observation. Therefore, the RC context receives the environmental facts, both observed and communicated, and identifies the set of norms which control the agent environment. In this proposal, recognised norms are defined as follows:

DEFINITION 1 (NORM). *A norm n is defined as* $n = \langle D, C, A, E, S, R \rangle$ *where:*

- $D \in \{O, F\}$, *is the deontic operator. In this work only obligations (O) and prohibitions (F) which impose constraints on agent behaviours have been considered; whereas permissions have not been considered*

since they are defined as operators that invalidate the activation of obligations or prohibitions;

- *C is a logic formula that represents the normative condition that must be carried out in case of obligations, or that must be avoided in case of prohibition norms;*
- *A, E are wff that determine the norm activation and expiration conditions, respectively;*
- *S, R are expressions which describe the actions (sanctions and rewards) that will be carried out in case of norm violation or fulfilment, respectively.*

Thus, the RC is formed by expressions defined as ($RC\ n$, ρ_n, ρ_S, ρ_R); where n is a first order formula which represents a norm. $\rho_n \in [0, 1]$ is the probability of norm application; i.e. the trust degree of the communicated norm or the observed probability of norm compliance in case of an inferred norm. In this work we assume that agents are informed about norms which control their behaviours by a representative of the normative system. Thus, there is a total reliability on the recognised norm ($\rho_n = 1$). Finally, $\rho_S, \rho_R \in [0, 1]$ are the probability values ascribed to the application of sanctions and rewards, respectively; i.e. the probability of being sanctioned or rewarded by the norm issuer.

3.2 Normative Context (NC)

The NC is inspired by the *commitment* context [10] in which agent behaviour is affected by commitments. However, in our approach norms are not static constraints implemented on agents. On the contrary, agents are able to acquire and accept norms dynamically in an autonomous way. Performance of the NC is: i) mental contexts inject formulas inside the NC; ii) the NC carries out an inference process in order to reason about norms considering the current mental state; and iii) BDI units are modified according to the new mental propositions which have been derived from norms.

The NC is formed by expressions like $NC(\lceil\gamma\rceil)$; where γ is a first-order logic expression which is defined as an inference rule which corresponds to a translated norm from the RC. In particular, these inference rules relate belief propositions with desires. The expression $\lceil\gamma\rceil$ means that γ is embedded in the normative context as a term; i.e. modal logic expressions modeled as first order theories.

The normative context logic consists of the axiom schema K, closure under implication, together with the consistency axiom. Therefore, contradictory norms are allowed; i.e. it is possible to define $NC(\lceil\gamma\rceil) \wedge NC(\lceil\neg\gamma\rceil)$. This fact is interesting for our work since agents are usually controlled by conflicting norms addressed at the different roles played by the agent or there may be a conflict among agent goals and norms. However, contradictory predicates such as $NC(\lceil\gamma\rceil) \wedge \neg NC(\lceil\gamma\rceil)$ are not allowed, i.e. expressions that claim that a certain norm exists and not exits simultaneously.

3.3 Bridge Rules

Normative BDI Agents require the definition of additional bridge rules for allowing norms to be recognised and normative decisions to be taken.

3.3.1 Updating the RC

Agent observations and communications which it perceives from its environment ($input(\beta)$) are included into the recognition context as a new term or theory ($\lceil input(\beta)\rceil$) (see Figure 1 Rule 4):

$$CC : input(\beta) \Rightarrow RC : \lceil input(\beta)\rceil \tag{4}$$

3.3.2 Norm Transformation Rules

Inside the recognition unit new norms are acquired. Those abstract recognised norms ($RC\ n, \rho_n, \rho_S, \rho_R$) are transformed into terms inside the normative context ($NC(\lceil\gamma\rceil)$) (see Figure 1 Rule 5):

$$RC : (RC\ n, \rho_n, \rho_S, \rho_R) \Rightarrow NC : (NC(\lceil\gamma\rceil)) \tag{5}$$

As previously argued, each abstract norm is translated into an inference rule ($\gamma = \varphi \rightarrow \psi$) belonging to the normative context. The definition of this inference rule depends on the concrete deontic type of the norm which is being translated.

- *Obligation Norm.*

$$RC : (RC\ \langle O, C, A, E, S, R\rangle, \rho_n, \rho_S, \rho_R) \Rightarrow NC : NC(\lceil\gamma\rceil)$$

where:

$$\gamma = \varphi \rightarrow \psi$$
$$\varphi = (B\ A, \beta_A) \wedge (B\neg E, \beta_{\neg E})$$
$$\psi = (D^{+}C, f(\theta_{activation}, \theta_{adoption}))$$

If an agent considers that the obligation is currently active ($(B\ A, \beta_A) \wedge (B\neg E, \beta_{\neg E})$) then a new positive desire corresponding to the norm condition is inferred:

$$(D^{+}C, f(\theta_{activation}, \theta_{adoption}))$$

- *Prohibition Norm.*

$$RC : (RC\ \langle F, C, A, E, S, R\rangle, \rho_n, \rho_S, \rho_R) \Rightarrow NC : NC(\lceil\gamma\rceil)$$

where:

$$\gamma = \varphi \rightarrow \psi$$
$$\varphi = (B\ A, \beta_A) \wedge (B\neg E, \beta_{\neg E})$$
$$\psi = (D^{-}C, f(\theta_{activation}, \theta_{adoption}))$$

Similarly to obligation norms, a prohibition related to a condition C is transformed into an inference rule which asserts a negative desire if the norm is active.

The certainty degree related to the norm activation ($\theta_{activation}$), together with the certainty or desirability of norm adoption ($\theta_{adoption}$) are employed by the function f in order to assign a degree to the normative desire. The concrete definition of f is problem dependent. However, in this work it has been implemented as:

$$f(\theta_{activation}, \theta_{adoption}) = \theta_{activation} \times \theta_{adoption}$$

The norm activation ($\theta_{activation}$) is defined as a $f_{activation}$ function that combines the belief degrees related to the norm activation and expiration conditions (β_A and $\beta_{\neg E}$) and the certainty degree of the norm (ρ_n):

$$\theta_{activation} = f_{activation}(\beta_A, \beta_{\neg E}, \rho_n)$$

$$f_{activation}(\beta_A, \beta_E, \rho_n) = \beta_A \times \beta_{\neg E} \times \rho_n$$

If the agent has not any belief related to occurrence of any of these conditions, then the belief degree is defined as zero.

The norm adoption ($\theta_{adoption}$) is defined as a $f_{adoption}$ function that takes as input the positive/negative degrees of the norm condition (δ^*_C), sanction (δ^*_S) and reward (δ^*_R); and the possibilities of being sanctioned and rewarded (ρ_S and ρ_R). With this information, the adoption function will determine if the agent accepts the norm.

$$\theta_{adoption} = f_{adoption}(\delta^*_C, \delta^*_S, \delta^*_R, \rho_S, \rho_R)$$

$$\delta^*_C = (\delta^+_C, \delta^-_C); \delta^*_S = (\delta^+_S, \delta^-_S); \delta^*_R = (\delta^+_R, \delta^-_R)$$

3.3.3 Updating the NC

Besides the definition of bridge rules for connecting the RC with the NC; additional bridge rules are needed in order to allow normative BDI agents to consider norms in their decision making process. More specifically, both agent desires and beliefs (γ) are included into the normative context as first order formulas ($\lceil\gamma\rceil$) in order to determine when a norm is active (Figure 1 Rules 6 and 7):

$$BC : \gamma \Rightarrow NC : NC(\lceil\gamma\rceil) \tag{6}$$

$$DC : \gamma \Rightarrow NC : NC(\lceil\gamma\rceil) \tag{7}$$

3.3.4 Updating the DC: Coherence Maintenance

In addition, after performing the inference process for creating new desires ($\lceil(D^*\ \gamma, \delta)\rceil$) derived from norm application; the normative context must update the DC (Figure 1 Rules 8 and 9). The addition of these propositions into this mental context may cause an inconsistency with the current mental state. Next, the problem of coherence maintenance among desires is faced.

In this proposal of BDI architecture, the maintenance of coherency among desires has been achieved by means of two different schemas (i.e. DC_1 and DC_2) which have been previously defined in [5]. These schemas impose some constraints between the positive and negative desires of a formula and its negation. Next, each schema is explained.

On the one hand, the DC_1 schema avoids having contradictory desires; i.e. to desire ($D^+\gamma, \delta^+_\gamma$) and ($D^+\neg\gamma, \delta^+_{\neg\gamma}$) simultaneously. Thus, this constraint and the corresponding for negative desires impose the next constraint over propositions belonging to the DC:

$$min(\delta^*_\gamma, \delta^*_{\neg\gamma}) = 0$$

where δ^*_γ and $\delta^*_{\neg\gamma}$ are the desirability or undesirability degrees assigned to proposition γ and its negation, respectively.

On the other hand, schema DC_2 imposes a restriction over positive and negative desires for a same goal. In particular, it claims that an agent cannot desire to be in world more than it is tolerated (i.e. not rejected). Therefore, it determines that:

$$\delta^+_\gamma + \delta^-_\gamma \leq 1$$

According to DC_1 and DC_2 schemas, bridge rule for updating the DC with the positive desires derived from norms is defined as follows (Figure 1 Rule 8):

$$\begin{aligned} NC : NC(\lceil(D^+\ \gamma, \delta)\rceil), \delta > \delta_{thres}, \\ DC : (D^-\ \gamma, \delta^-), DC : (D^+\ \gamma, \delta^+) \Rightarrow \\ DC : (D^+\ \gamma, max(\delta, \delta^+)), DC : (D^+\neg\gamma, 0), \\ DC : (D^-\ \gamma, min(\delta^-, 1 - max(\delta, \delta^+))) \end{aligned} \tag{8}$$

Thus, the desire degree assigned to the new proposition γ is defined as the maximum between the new desirability and the previous value ($max(\delta, \delta^+)$). In order to follow DC_1 schema, desirability of $\neg\gamma$ is updated to 0. According to DC_2 schema, the undesirability assigned to γ is updated as the minimum between the previous value of undesirability assigned to γ (δ^-) and its maximum coherent undesirability, which is defined as $1 - max(\delta, \delta^+)$. Moreover, in order to avoid the propagation of insignificant terms, only these new terms whose degree exceeds δ_{thres} will be transformed into mental objects. The definition of this threshold is also problem dependent.

Similarly, bridge rule for updating the DC with negative desires is defined as follows (Figure 1 Rule 9):

$$\begin{aligned} NC : NC(\lceil(D^-\ \gamma, \delta)\rceil), \delta > \delta_{thres}, \\ DC : (D^-\ \gamma, \delta^-), DC : (D^+\ \gamma, \delta^+) \Rightarrow \\ DC : (D^-\ \gamma, max(\delta, \delta^-)), DC : (D^-\neg\gamma, 0), \\ DC : (D^+\ \gamma, min(\delta^+, 1 - max(\delta, \delta^-))) \end{aligned} \tag{9}$$

Along this section, our proposal of Normative Graded BDI architecture has been explained. Through the adoption function different strategies for norm adoption can be implemented. Strategies for norm adoption have been classified into rational and non-rational motivations. The next section illustrates how the former type of norm adoption strategies is implemented in our architecture.

4. RATIONAL STRATEGIES FOR NORM ADOPTION

Several proposals [6, 7, 12] have been made with the aim of defining rational strategies for norm adoption. One of the first works on analysing motivations for norm adoption from an agent perspective was presented in [6]. Here it is claimed that norms not only require a *behaviour* but also a *mental* attitude. According to this work, strategies for norm adoption are classified into: i) *unconditional* adoption, which implies that agents do not have capabilities for considering norms and they always fulfil norms; ii) *instrumental* adoption, which implies a greater level of autonomy since agents adopt norms if they consider them as beneficial to their goals; iii) *cooperative* agents adopt norms whenever they consider them being beneficial for the whole society; and iv) *benevolent* agents fulfil those norms which benefit other agents which they want to favour. In [12] these strategies were revised and extended.

This section illustrates how different adoption strategies can be easily implemented by our proposal of multi-context BDI agent. Mainly, the norm adoption strategy determines the certainty degree assigned to the new mental attributes created by the inference rules inside the normative context. More specificallly, the $f_{adoption}$ function implements the different norm adoption strategies.

4.1 Traditional Strategies for Norm Adoption.

Taking as a reference the classification of strategies for norm adoption described in [12], we propose to implement each strategy as follows:

- *Simple Strategies*, these ones do not consider the effects that compliance with a norm might have on agent's goals. They are classified into:

– Agents which follow the *Automatic* strategy will accept all norms:

$$f_{adoption}(\delta^*_C, \delta^*_S, \delta^*_R, \rho_S, \rho_R) = 1$$

– Agents which follow the *Rebellious* strategy will reject all norms systematically:

$$f_{adoption}(\delta^*_C, \delta^*_S, \delta^*_R, \rho_S, \rho_R) = 0$$

– Agents which follow a *Fearful* strategy will accept those norms which have a sanction. Thus, these agents do not consider whether the sanction is beneficial or detrimental to their goals, but they only consider if the norm has a sanction:

$$f_{adoption}(\delta^*_C, \delta^*_S, \delta^*_R, \rho_S, \rho_R) = \begin{cases} 1 & \text{if } \delta^+_S + \delta^-_S > 0 \\ 0 & \text{otherwise} \end{cases}$$

– Finally, *Greedy* agents adopt all norms whose compliance is rewarded, without considering the utility of the reward:

$$f_{adoption}(\delta^*_C, \delta^*_S, \delta^*_R, \rho_S, \rho_R) = \begin{cases} 1 & \text{if } \delta^+_R + \delta^-_R > 0 \\ 0 & \text{otherwise} \end{cases}$$

- *Motivated Strategies* are more complex strategies which consider the possible effects on goals of both the norm condition and the rewards in the case a norm is fulfilled, and the effects of punishments if it is not. These strategies are based on the *utilitarian* view which defines the utility as the good to be maximized. Thus, the desirability of both norm fulfilment and violation is considered as criteria for norm adoption.

  – An *Egoist* agent will accept only those norms which benefit its goals:

$$f_{adoption}(\delta^*_C, \delta^*_S, \delta^*_R, \rho_S, \rho_R) = \begin{cases} 1 & \text{if } \delta^+_C > 0 \\ 0 & \text{otherwise} \end{cases}$$

  – *Pressure* can be a motivation for norm adoption. More concretely, an agent which follows the *Pressure* strategy will respect all norms whose sanction is more undesired than the norm condition:

$$f_{adoption}(\delta^*_C, \delta^*_S, \delta^*_R, \rho_S, \rho_R) = \begin{cases} 1 & \text{if } \delta^-_S > \delta^-_C \\ 0 & \text{otherwise} \end{cases}$$

  – An *Opportunist* will accept all norms whose reward is more preferred to the negative effects of the norm:

$$f_{adoption}(\delta^*_C, \delta^*_S, \delta^*_R, \rho_S, \rho_R) = \begin{cases} 1 & \text{if } \delta^+_R > \delta^-_C \\ 0 & \text{otherwise} \end{cases}$$

- *Social Strategies.* In the existing literature different social strategies for norm adoption have been defined:

  – *Cooperative* agents accept norms which are considered as beneficial for the whole society. This strategy can be implemented by defining social goals as agent desires.

  – A *benevolent* agent adopts those norms which are desirable for another agent which it wants to favour. In order to implement this strategy, it is necessary to determine if a target agent j would be favoured from norm application; i.e. if target agent j has a positive desire of C. Therefore, any agent should be able to represent other's mental attitudes as beliefs (i.e. nested mental propositions such as $(B\,(D^+_j C), \beta) \in BC$). This represents a problem since desire formulas as $(D^+_j C)$ are many-valued formulae (they have a truth value belonging to [0,1]). Taking as a reference the solution proposed in [5], this problem can be solved by means of the definition of a projection operator (∇) defined as true if the mental proposition has a positive degree. In this sense, $\nabla(D^+_j C)$ would be true when target agent j had a positive desire related to proposition C whose degree was greater than 0. Thus, norms will be adopted if the agent has the belief $(B\,\nabla(D^+_j C), \beta)$, which means that an agent beliefs that another agent j desires occurrence of C with an intensity higher than 0.

4.2 Complex Strategies for Norm Adoption.

Besides these well known strategies for norm adoption, more complex ones can be defined taking advance of the possibilities which the proposed Normative BDI architecture provides. The main idea beyond complex strategies is to consider both positive and negative effects derived from norm fulfillment and violation.

- *Mixed Strategy.* It accepts a norm if the effect of norm compliance is higher than the effect of norm violation. On the one hand, consequences of norm fulfilment are the desirability of both the norm condition (δ^+_C) and the reward (δ^+_R) and the undesirability of the norm sanction (δ^-_S), which will be avoided if the norm is respected. On the other hand, the effects of norm violation are the desirability of the sanction (δ^+_S) and the undesirability of both the norm condition (δ^-_C) and reward (δ^-_R), that will be avoided if the norm is not respected.

$$f_{adoption}(\delta^*_C, \delta^*_S, \delta^*_R, \rho_S, \rho_R) = \begin{cases} 1 & \text{if } \delta^+_R + \delta^+_C + \delta^-_S > \\ & \delta^-_R + \delta^-_C + \delta^+_S \\ 0 & \text{otherwise} \end{cases}$$

- *Mixed Pondered Strategy.* This strategy is very similar to the previous one, since it accepts a norm if the effect of norm compliance is higher than the effect of norm violation. However, both desirability and undesirability of sanctions and rewards are pondered with their observed probabilities (ρ_S and ρ_R) when calculating the effects of the norm.

$$f_{adoption}(\delta^*_C, \delta^*_S, \delta^*_R, \rho_S, \rho_R) = \begin{cases} 1 & \text{if } (\rho_R * \delta^+_R) + \delta^+_C + \\ & (\rho_S * \delta^-_S) > (\rho_R * \delta^-_R) \\ & +\delta^-_C + (\rho_S * \delta^+_S) \\ 0 & \text{otherwise} \end{cases}$$

These are discrete strategies in the sense they determine if the norm will be adopted or not. However, continuous functions can be easily defined by employing the difference between norm compliance and violation effects.

As being illustrated, the proposed Normative BDI agent architecture is general enough for implementing well known strategies for norm adoption as well as more complex strategies or a combination of different strategies. Next, our proposal of normative agent architecture is applied into a case study which illustrates the differences among the norm adoption strategies.

5. THE M-WATER PROBLEM

The management of natural resources is a challenge of significant social relevance. As argued in [3], the use of water in a basin can be seen as a MAS controlled by norms. In this section an example scenario of the m-Water problem is illustrated. According to the Spanish Water Law, the *irrigators* which belong to the same area of a basin can be organized forming *irrigator communities*. These communities act on behalf of their members by defending their rights and interests. However, each community can impose some norms or restrictions to their members.

5.1 Basic Scenario

The *irrigator* agent represents a farmer who wants to obtain high quality vegetables from its plantation:

$$(D^{+}highQuality, 1)$$

In order to achieve its goal of picking high quality vegetables it has two different irrigation possibilities: to irrigate daily or every two days. Thus, there are, at least, two different cultivation plans: one which contains the daily irrigation action and another which performs the action corresponding to the irrigation in alternative days. Logically, it is more possible to obtain a good crop if the land is frequently irrigated:

$$(B\ [dailyIrrigation]highQuality, 0.75)$$
$$(B\ [alternateDaysIrrigation]highQuality, 0.5)$$

Finally, he avoids being fined: $(D^{-}payFine, 0.8)$.

5.1.1 Bridge Rule Application

Realistic Desires. The first step performed by the BDI architecture consists in applying Bridge Rule 1 in order to refine abstract desires into more realistic ones:

$$(D^{+}\ highQuality, 1),$$
$$(B\ [dailyIrrigation]highQuality, 0.75) \Rightarrow$$
$$DC : (D^{+}\ [dailyIrrigation]highQuality, f_D(1, 0.75))$$

$$(D^{+}\ highQuality, 1),$$
$$(B\ [alternateDaysIrrigation]highQuality, 0.5) \Rightarrow$$
$$DC : (D^{+}\ [alternateDaysIrrigation]highQuality, f_D(1, 0.5))$$

Function f_D is implemented as the product of both degrees.

Deriving Intentions. These more specific desires allow the agent to determine which actions will be intended according to the existing plans (Bridge Rule 2):

$$DC : (D^{+}\ [dailyIrrigation]highQuality, 0.75),$$
$$PC : plan(dailyIrrigation), 0.75 > 0 \Rightarrow$$
$$IC : (I[dailyIrrigation]highQuality, 0.75)$$

$$DC : (D^{+}\ [alternateDaysIrrigation]highQuality, 0.5),$$
$$PC : plan(alternateDaysIrrigation), 0.5 > 0 \Rightarrow$$
$$IC : (I[alternateDaysIrrigation]highQuality, 0.5)$$

Action Selection. Finally, the action which is more intended will be performed by the agent (Bridge Rule 3):

$$IC : (I[dailyIrrigation]highQuality, 0.75)$$
$$IC : (I[alternateDaysIrrigation]highQuality, 0.5),$$
$$0.75 > 0.5 \Rightarrow CC : (act(dailyIrrigation))$$

5.2 Normative Decision Making

The *irrigator* agent is affected by norms as a consequence of being a member of an irrigator community. In this example the community forbids agents to irrigate daily if a drought state has been declared in this area. Once the agent becomes a member of the community it is informed by a representative about norms which affect it. Thus, the agent assigns the maximum certainty degree to the recognised norm:

$$(RC\ \langle\langle F, drought, -, dailyIrrigation, payFine,$$
$$candidateGov\rangle, \rho_n, \rho_S, \rho_R)$$
$$\rho_n = 1, \rho_S = 0.25, \rho_R = 1$$

If the *irrigator* respects the norm then it would become a candidate to the *governors board* of the community. However, being a governor implies a lot of responsibilities. Because of this the *irrigator* is not interested on becoming a candidate to the *governors board* $((D^{-}candidateGov, 0.5))$.

In this example, the irrigator agent is not sure if a drought state has been declared. However, according to the meteorological conditions it thinks it is possible that there is a drought situation. Hence it has a belief $(B\ drought, 0.6)$ in order to represent this drought possibility.

5.2.1 Norm Transformation

Once the norm is been recognised by the RC it is transformed into an inference rule inside the NC (Bridge Rule 5):

$$RC : (RC\ \langle\langle F, drought, -, dailyIrrigation, payFine,$$
$$candidateGov\rangle, 1, 0.25, 1) \Rightarrow$$
$$NC : NC(\lceil\varphi \rightarrow \psi\rceil)$$

where:

$$\varphi = (B\ drought, 0.6)$$
$$\psi = (D^{-}dailyIrrigation, f(f_{activation}(0.6, -, 1)),$$
$$f_{adoption}(\delta^{*}_{dailyIrrigation}, \delta^{*}_{payFine}, \delta^{*}_{candidateGov}, 0.25, 1))$$

On the one hand, the norm activation function takes as input the certainty value assigned to the occurrence of the norm activation condition (0.6) and the confidence value assigned to the norm recognition (1). In particular, this agent has implemented the $f_{activation}$ as the product of its not empty parameters:

$$f_{activation}(\delta_A, \delta_{\neg E}, \delta_{nr}) = \delta_A \times \delta_{\neg E} \times \delta_{nr} = 0.6$$

5.2.2 Norm Adoption

Regarding the norm acceptance, different results can be obtained depending on the adoption strategy that has been employed.

$$f_{adoption}(\delta^{*}_C, \delta^{*}_S, \delta^{*}_R, \rho_S, \rho_R)$$

where:

$\delta^{+}_C = \delta^{+}_{dailyIrrigation} = 0; \delta^{-}_C = \delta^{-}_{dailyIrrigation} = 0$ since there is none (positive or negative) desire on the norm condition (i.e. *dailyIrrigation*);

$\delta_S^+ = \delta_{payFine}^+ = 0; \delta_S^- = \delta_{payFine}^- = 0.8$, since the irrigator agent avoids being fined ($(D^- payFine, 0.8)$);

$\delta_R^+ = \delta_{candidateGov}^+ = 0; \delta_R^- = \delta_{candidateGov}^- = 0.5$, as he is not interested on becoming a candidate to governor ($(D^- candidateGov, 0.5)$;

$\rho_S = 0.25$, which implies that there is a low probability of being sanctioned when not following this norm;

$\rho_R = 1$, which implies that the reward action will be always applied when following this norm;

Next, results obtained with each strategy are shown:

- *Automatic Strategy.* In this case the *irrigator* always accepts the norm, thus $f_{adoption} = 1$. Then the f function multiplies the values obtained by the $f_{activation}$ and $f_{adoption}$ functions. Therefore, a new normative desire ($D^- dailyIrrigation, 0.6$) is inserted into the DC (Bridge Rule 8), being $\delta_{threshold} = 0.25$:

$$NC : NC(\lceil (D^- dailyIrrigation, 0.6) \rceil),$$
$$0.6 > 0.25 \Rightarrow DC : (D^- dailyIrrigation, 0.6))$$

  Then the IC is updated through Bridge Rule 2:

$$DC : (D^+[dailyIrrigation] highQuality, 0.9),$$
$$DC : (D^- dailyIrrigation, 0.6),$$
$$PC : plan(dailyIrrigation), 0.9 > 0.6 \Rightarrow$$
$$IC : (I[dailyIrrigation] highQuality, f_I(0.9, 0.6))$$

  Thus, a new intention related to the *dailyIrrigation* action ($I[dailyIrrigation] highQuality, 0.3$) is created. Its intentionality has been reduced since the action has a negative desire. Finally, the intention update implies the modification of the agent behaviour. The intention related to the *alternateDaysIrrigation*, whose degree is 0.5, is the most intended. Thus, the action performed is *alternateDaysIrrigation*, since the *irrigator* has adopted the norms of its irrigation community.

- *Rebellious Strategy.* This strategy implies that the *irrigator* rebuts respecting norms systematically, so then $f_{adoption} = 0$ for any norm. In this case a new intention will be created in the NC ($D^+ alternateDaysIrrigation$, 0). However, this normative desire is not inserted into the DC since its degree does not exceed δ_{thres}. Consequently, the norm is not followed and the agent maintains the daily irrigation.

- *Fearful Strategy.* Since the norm has a sanction which punishes *agents* that do not respect the prohibition, the *irrigator* would follow the social norm ($f_{adoption} = 1$). Consequently, it will adopt an *alternateDaysIrrigation* action, similarly as in the case of an automatic strategy.

- *Greedy Strategy.* This strategy implies following the norm whenever there is a reward. In this case, since $\delta_R^+ + \delta_R^- = 1 > 0$ the *irrigator* agent will adopt the norm and, thus, it will perform the *alternateDaysIrrigation* action.

- *Egoist Strategy.* With this strategy, the *irrigator* will respect the norm only if it benefits its goals ($\delta_C^+ > 0$). In this case $\delta_{dailyIrrigation}^+ = 0$ and then $f_{adoption} = 0$.

- *Pressure Strategy.* This strategy defines that the agent adopts the norm only if its sanction is more undesired than the norm condition ($\delta_S^- > \delta_C^-$). Since $\delta_{payFine}^- = 0.8$ and $\delta_{dailyIrrigation}^- = 0$, then $f_{adoption} = 1$.

- *Opportunistic Strategy.* This strategy defines that the agent will adopt the norm only if its reward is more desired than the undesirability of the norm condition ($\delta_R^+ > \delta_C^-$). $\delta_R^+ = 0$, since the irrigator agent does not desire the reward at all, so then $f_{adoption} = 0$.

- *Mixed Strategy.* With this strategy the agent would consider both positive and negative effects derived from norm respect or norm violation. The norm is adopted if:

$$\delta_R^+ + \delta_C^+ + \delta_S^- > \delta_R^- + \delta_C^- + \delta_S^+$$

  According to the current mental state its value is:

$$0 + 0 + 0.8 > 0.5 + 0 + 0$$

  which is true. Then $f_{adoption} = 1$.

- *Mixed Pondered Strategy.* This strategy is defined as the previous one but the reward and sanction desirability and undesirability are pondered with their possibilities; i.e. the norm is respected if:

$$(\rho_R * \delta_R^+) + \delta_C^+ + (\rho_S * \delta_S^-) > (\rho_R * \delta_R^-) + \delta_C^- + (\rho_S * \delta_S^+)$$

  In this example this formula is:

$$(1 \times 0) + 0 + (0.25 \times 0.8) > (1 \times 0.5) + 0 + (0.25 \times 0)$$

  Then, the comparison is not true and $f_{adoption} = 0$.

Along this section, a case study of an autonomous normative agent has been illustrated. More specifically, it consists of an *irrigator* agent which must choose whether respecting or not respecting norms. In its decision making process it employs different norm adoption strategies which have been defined for the proposed Normative BDI architecture.

6. DISCUSSION

Autonomous normative agents are defined as agents which have explicit knowledge about norms and are able to decide about norm adoption convenience; i.e. they have capabilities for recognizing, representing and accepting norms, and for solving possible conflicts among them [6]. Several proposals have been made in order to define agents provided with some of these capabilities. However, the definition of an agent architecture and the reasoning processes over this architecture which overcome all of the challenges raised by autonomous normative agents stays an open issue.

Regarding recent works on normative reasoning, the BOID architecture [4] represents obligations as mental attributes and analyses the relationship and influence of such obligations on agent beliefs, desires and intentions. This approach is very similar to the work proposed here. However, our approach overlaps the main drawbacks of the BOID proposal in different ways: i) our normative model does not only consider obligation norms but it gives support to constitutive and procedural norms [8]; ii) it employs graded BDI logics for representing mental attitudes, which allows agents to face with uncertain and conflicting mental sates; and iii) it consider norms as dynamic entities that agents should acquire

from their environment. In relation with this last feature, the EMIL-A proposal [1], which has developed a framework for autonomous norm recognition, might be employed for complementing the RC component of our normative BDI architecture. Thus, agents would be able to acquire new norms by observing the behaviour of other agents which are situated in their environments. The main advantage of our proposal with respect to EMIL-A is that our agent architecture allows agents to decide whether adopting or not norms according to their own motivations and interests. On the contrary, EMIL-A agents adopt all recognised norms automatically by deriving new normative goals.

Finally, the normative reasoning problem requires sophisticated techniques in order to allow agents to consider convenience of norm compliance according to their current mental state. In this sense, norms may be inconsistent with the mental state of agents. The *cognitive coherence theory* evaluates the truth of cognitions in relation with a set of cognitions [17]. Its main purpose is the study of associations; i.e. how pieces of information influence each other by imposing a positive or negative constraint over the rest of information. The problem of coherence among agent cognitions has been superficially addressed in this paper and will be object of future work. Regarding more elaborated solutions to the coherence problem, in [11] a formalization of deductive coherence theory has been used as a criterion for rejecting or accepting norms. This work is based on a very simple notion of norm as an unconditional obligation. Moreover, this proposal only considers coherence as the one rational criterion for norm acceptance.

7. CONCLUSIONS

Our proposal of Normative Graded BDI architecture allows agents to acquire new norms from their environment and consider them in their decision making process. The fact that mental attitudes of agents are quantified allows them to reason in open environments which are controlled by norms. In this sense, graded modalities allow agents to represent uncertain knowledge about the current state of the world. Moreover, graded intentions and desires enable agents to make decisions according to their satisfaction criterion. This is specially interesting when designing normative agents whose behaviour can be affected by conflicting norms. Thus, the desirability degrees of desires and intentions allow agents to decide between norm violation or fulfilment according to their priorities.

As been illustrated our proposal of Normative Graded BDI architecture: i) allows the definition of those strategies which had been defined in previous works; ii) the fact that mental attitudes are represented as graded propositions allows agents to consider not only whether norms are beneficial to their goals and motivations but also the intensity in which they will be affected; and iii) it overlaps previous works since it allows the definition of complex strategies which consider both positive and negative effects of norm fulfilment and violation.

As future work, we plan to continue by working on the analysis of non-rational motivations for norm adoption. In this sense, we are working on extending our agent architecture with an emotion model which will allow agents to take into consideration phenomena such as shame, honour, gratitude, etc. when adopting new norms. Finally, we will carry out a set of experimentations on norm adoption in order to determine empirically the role of both rationality and emotions on the norm adherence process.

8. ACKNOWLEDGMENTS

This work was partially supported by the Spanish government under grants CONSOLIDER-INGENIO 2010 CSD2007-00022, TIN2009-13839-C03-01 and TIN2008-04446 and by the FPU grant AP-2007-01256 awarded to N. Criado.

9. REFERENCES

[1] G. Andrighetto, M. Campenní, F. Cecconi, and R. Conte. How agents find out norms: A simulation based model of norm innovation. In *NORMAS*, pages 16–30, 2008.

[2] A. Artikis and J. Pitt. A formal model of open agent societies. In *AGENTS*, pages 192–193. ACM, 2001.

[3] V. Botti, A. Garrido, A. Giret, and P. Noriega. Managing water demand as a regulated open mas. In *MALLOW Workshop on COIN*, page In Press., 2009.

[4] J. Broersen, M. Dastani, J. Hulstijn, Z. Huang, and L. van der Torre. The boid architecture – conflicts between beliefs, obligations, intentions and desires. In *AAMAS*, pages 9–16. ACM Press, 2001.

[5] A. Casali. *On Intentional and Social Agents with Graded Attitudes*. PhD thesis, 2008.

[6] C. Castelfranchi. Prescribed mental attitudes in goal-adoption and norm-adoption. *Artif. Intell. Law*, 7(1):37–50, 1999.

[7] R. Conte, C. Castelfranchi, and F. Dignum. Autonomous norm acceptance. *LNCS*, 1555:99–112, 1999.

[8] N. Criado, E. Argente, and V. Botti. A bdi architecture for normative decision making (extended abstract). In *AAMAS*, page In Press, 2010.

[9] J. Elster. Social norms and economic theory. *Journal of Economic Perspectives*, 3(4):99–117, 1989.

[10] D. Gaertner, P. Noriega, and C. Sierra. Extending the BDI architecture with commitments. In *CCIA*, pages 247–257, 2006.

[11] S. Joseph, C. Sierra, M. Schorlemmer, and P. Dellunde. Deductive coherence and norm adoption. *Logic Journal of the IGPL*, 18:118–156, 2010.

[12] F. López. *Social Powers and Norms: Impact on Agent Behaviour*. PhD thesis, 2003.

[13] M. Luck and P. McBurney. Computing as interaction: Agent and agreement technologies. In *EUMAS*, pages 1–15, 2008.

[14] J. Meyer. Dynamic logic for reasoning about actions and agents. In J. Minker, editor, *Logic-Based Artificial Intelligence*, pages 281–311. Kluwer Academic Publishers, Dordrecht, 2000.

[15] S. Parsons, C. Sierra, and N. R. Jennings. Agents that reason and negotiate by arguing. *JLC: Journal of Logic and Computation*, 8(3):261–292, 1998.

[16] L. Perrussel. Contextual reasoning. In *ECAI*, pages 366–367, 1998.

[17] P. Thagard. *Coherence in Thought and Action*. The MIT Press, Cambridge, Massachusetts, 2000.

Identifying conditional norms in multi-agent societies

Bastin Tony Roy Savarimuthu, Stephen Cranefield, Maryam Purvis and Martin Purvis
Department of Information Science
University of Otago
P O Box 56, Dunedin, New Zealand
((tonyr,scranefield,tehrany,mpurvis)@infoscience.otago.ac.nz)

ABSTRACT

Most works on norms have investigated how norms are regulated using institutional mechanisms which assume that agents know the norms of the society they are situated in. Few research works have focused on how an agent may infer the norms of a society without the norm being explicitly given to the agent. These works do not address how an agent can identify conditional norms. In this paper we describe a mechanism that an agent can use to identify conditional norms which makes use of our previously proposed norm identification framework. Using park littering as an example, we show how conditional norms can be identified. In addition we discuss the experimental results on the dynamic addition, modification and deletion of conditional norms.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*; J.4 [**Computer Applications**]: Social and Behavioral Sciences—*Sociology*

General Terms

Experimentation

Keywords

Norms, Conditional norms, Norm identification, Agents, Societies

1. INTRODUCTION

Most works on norms in normative multi-agent systems have concentrated on how norms regulate behaviour (e.g. [15,18]). These works assume that the agent somehow knows (*a priori*) what the norms of a society are. For example, an agent may have obtained the norm from a leader [9] or through an institution that prescribes what the norms of the society should be [3, 29].

Only a few researchers have dealt with how an agent may infer what the norms of a newly joined society are [4, 22]. Recognizing the norms of a society is beneficial to an agent. This process enables the agent to know what the *normative expectation* of a society is. As the agent joins and leaves different agent societies, this capability is essential for the agent to modify its expectations of behaviour, depending upon the society of which it is a part. As the environment changes, the capability of recognizing a new norm helps an agent to derive new ways of achieving its intended goals. Such a norm identification mechanism can be useful for software agents that need to adapt to a changing environment. In open agent systems, instead of possessing predetermined notions of what the norms are, agents can infer and identify norms through observing patterns of interactions and their consequences. For example, a new agent joining a virtual environment such as Second Life [20] may have to infer norms when joining a society as each society may have different norms. It has been noted that having social norms centrally imposed by the land owners in Second Life is ineffective and there is a need for the establishment of community driven norms [27]. When a community of agents determines what the norm should be, the norm can evolve over time. So, a new agent joining the society should have the ability to recognize the changes to the norms. In our previous work we have proposed a norm identification framework which can be used to identify norms in the society [22, 23]. The norm identification framework takes into account the social learning theory [7] that suggests that new behaviour can be learnt through the observation of punishment and rewards.

This work aims to answer the question of how agents infer conditional norms in a multi-agent society. Conditional norms are defined as norms with conditions[1]. Software agent should not only have the ability to identify norms but also the conditions under which these norms hold.

Identifying conditional norms is important because an agent that has inferred that another agent gets punished when that agent littered when it was 25 metres away from the bin may infer that the condition associated with the norm is the distance of 25 metres. But the actual norm could be that no one should litter within 50 metres from the bin. The utility of the agent can be negatively impacted through a sanction when it violates a norm if it litters 30 metres away from a bin. In this case, the agent does not know the correct condition associated with the norm. Another example of a conditional norm is the tipping norm. In one society an agent may tip 10% of the bill while in another society an agent might be obliged to tip 20% of the bill. In this work we are interested in experimenting with the formation, modification and the removal of conditional norms in the minds of the agents and the impact of the normative conditions on the utility of the agents.

The paper is organized as follows. Section 2 provides a background on norms and how the concept of norms is investigated in the field of normative multi-agent systems (NorMAS). Section 3 provides an overview of our previous work on the norm identification framework. Section 4 describes a mechanism for identifying conditional norms. Section 5 describes the experiments that we have conducted and the results obtained. Section 6 provides a discussion on the work that has been achieved and the issues that can be addressed in the future. Concluding remarks are presented in

[1]We distinguish norms that are not associated with conditions from the ones that have conditions. An example of a norm without a condition is the norm that prohibits anyone from littering a public park (i.e. *prohibit(litter)*). An example of a norm with condition is a norm that prohibits one from littering as long as there is a rubbish bin within x metres from the agent (e.g. *if (distanceFromBin* < 10*) then prohibit(litter)*).

Section 7.

2. BACKGROUND

Due to multi-disciplinary interest in norms, several definitions for norms exist [4]. Elster notes the following about social norms [13]. *"For norms to be social, they must be shared by other people and partly sustained by their approval and disapproval. They are sustained by the feelings of embarrassment,anxiety, guilt and shame that a person suffers at the prospect of violating them. A person obeying a norm may also be propelled by positive emotions like anger and indignation ... social norms have a grip on the mind that is due to the strong emotions they can trigger"*.

Based on the definitions provided by various researchers, we note that the social practices surrounding the notion of a social norm are the following:

- *The normative expectation of a behavioural regularity*: There is a general agreement within the society that a behaviour is *expected* on the part of an agent (or actor) by others in a society, in a given circumstance.
- *A norm enforcement mechanism*: When an agent does not follow a norm, it could be subjected to a sanction. The sanction could include monetary or physical punishment in the real world which can trigger emotions (embarrassment, guilt, etc.) or direct loss of utility (e.g. decrease of its reputation score).
- *A norm spreading mechanism*: Examples of norm spreading mechanisms include the notion of advice from powerful leaders, imitation and learning on the part of an agent.

2.1 Normative multi-agent systems

The definition of normative multi-agent systems given by the researchers involved in the NorMAS 2007 workshop is as follows [8]. *A normative multi-agent system is a multi-agent system organized by means of mechanisms to represent, communicate, distribute, detect, create, modify and enforce norms, and mechanisms to deliberate about norms and detect norm violation and fulfillment.*

Researchers in multi-agent systems have studied how the concept of norms can be applied to artificial agents. Norms are of interest to multi-agent system (MAS) researchers as they help in sustaining social order and increase the predictability of behaviour in the society. Researchers have shown that norms improve cooperation and collaboration [26, 31]. Epstein has shown that norms reduce the amount of computation required to make a decision [14]. However, software agents may tend to deviate from norms due to their autonomy. So, the study of norms has become important to MAS researchers as they can build robust multi-agent systems using the concept of norms and also experiment on how norms may evolve and adapt in response to environmental changes.

Research in normative multi-agent systems can be categorized into two branches. The first branch focuses on normative system architectures, norm representations, norm adherence and the associated punitive or incentive measures. Several architectures have been proposed for normative agents (refer to [19] for an overview). Researchers have used deontic logic to define and represent norms [16, 32]. Several researchers have worked on mechanisms for norm compliance and enforcement [3, 18].

The second branch of research is related to emergence of norms. Researchers have worked on both prescriptive (top-down) and emergent (bottom-up) approaches to norms. In a top-down approach an authoritative leader or a normative advisor prescribes what a norm of the society should be [30]. In the bottom-up approach, the agents come up with a norm through learning mechanisms [25, 26]. Researchers have used sanctioning mechanisms [6] and reputation mechanisms [12] for enforcing norms.

Many research works assume that norms exist in the society and the focus is on how the norms can be regulated in an institutional setting such as electronic institutions [5]. Very few have investigated how an agent comes to know the norms of the society [4, 22]. We have previously proposed an architecture for norm identification [22, 23]. In this work, we extend our earlier work by incorporating the mechanism for identifying conditional norms. Identifying conditional norms is important because the agent can confidently apply the norm if the conditions associated with the norm are known. This will help the agent not to lose utility by preventing it from applying the norm under wrong conditions.

3. OVERVIEW OF THE NORM IDENTIFICATION FRAMEWORK

In this section, we provide an brief overview of the norm identification framework that we have proposed and experimented with in earlier works [22, 23]. An agent employing this architecture follows a four-step process.

- Step 1: An agent actively perceives the events in the environment in which it is situated.
- Step 2: When an agent perceives an event, it stores the event in its belief base.
- Step 3: Based on recognizing signals (i.e. events that are either rewards or a sanctions), the agent stores them in a "special events" base.
- Step 4: If the perceived event is a special event an agent checks if there exists a norm in its *personal norm* (*p-norm*) base or the *group norm* (*g-norm*) base. An agent may possess some p-norms[2] based on its past experience or preference. A *p-norm* may vary across agents, since a society may be made up of agents with different backgrounds and experiences. A *g-norm* is a norm which an agent infers, based on its personnel interactions as well as the interactions it observes in the society. An agent infers g-norms using the norm inference component[3].

When a special event occurs an agent may decide to invoke its norm inference component to identify whether a previously unknown norm may have resulted in the occurrence of the special event. In the context of the park-littering scenario, an agent observing a sanctioning event may invoke its norm inference component to find out what events that had happened in the past (or that had not happened in the past) may have triggered the occurrence of the special event[4]. The invocation of the norm inference component may

[2]A *p-norm* is the personal value of an agent. For example an agent may consider that littering is an action that should be prohibited in a society. This personal value may not be shared by the agents in a society.

[3]The norm inference component of the framework makes use of Candidate Norm Inference (CNI) algorithm. The CNI algorithm uses association rule mining approach to identify sequences of events as candidate norms.

[4]Prohibition norms can be identified by inferring the relevant events that happened in the past. For identifying obligation norms the agent may have to reason about what events that did not happen in the past are the likely reason for a sanction (i.e. not fulfilling an obligation). In this work we have considered prohibition norms.

result in the identification of a *g-norm*, in which case it is added to the *g-norm* base.

An agent, being an autonomous entity, can also decide not to invoke its norm inference component for every occurrence of a special event but may decide to invoke it periodically. When it invokes the norm inference component, it may find a new *g-norm* which it adds to its *g-norm* base. If it does not find a *g-norm*, the agent may change some of its norm inference parameters and repeat the process again in order to find a *g-norm* or may wait to collect more information.

At regular intervals of time an agent re-evaluates the g-norms it currently has, to check whether those norms hold. When it finds that a *g-norm* does not apply (e.g. if it does not find any evidence of sanctions), it deletes the norm from the *g-norm* base.

The internal operational details of the norm inference framework can be found in our previous work [23]. The next section describes how conditional norms are inferred by an agent. The mechanism for identifying conditional norms is built on top of the norm inference framework.

4. IDENTIFYING CONDITIONAL NORMS

In our framework, when a new agent enters a society it will try to identify the norms that currently hold in that society. Once an agent has identified a norm it may want to identify the context and the exact conditions under which the norm holds. For example, the norm in a public park could be not to litter, i.e. *prohibit (litter)*[5]. It could be that the norm prohibits people from dropping litter in the park as long as a rubbish bin is not visible to them (or the rubbish bin is 50 metres away). The context here is the rubbish bin and the condition is the distance from the rubbish bin. When an agent identifies the norm in the first instance through observation, it may not know the exact conditions associated with the norm.

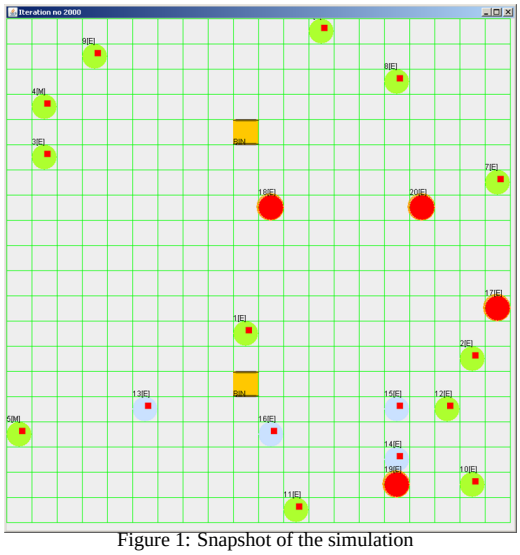


Figure 1: Snapshot of the simulation

[5]An agent infers a prohibition norm (e.g. *prohibit(litter))* by using association rule mining approach where norms are identified by inferring the relevant events that happened in the past [23]. For example an agent may notice that a sanctioning event is always preceded by a littering event. Hence the agent might infer that littering is prohibited in the society.

Let us assume that an agent upon identifying the norm knows the context of the norm. For example, on identifying that littering is prohibited the agent identifies the presence of the bin as the context. The condition associated with the norm is the distance between the agent and the bin[6]. After inferring the norm, the agent will observe the distance between the agents and the bins when they are sanctioned. Based on observed distances, it will form its conditional norm.

Algorithm 1: Pseudocode of an agent to identify a conditional norm

```
1  maxDistanceFromBin = 0, tempDistance = 0;
2  conditionalNormReferralConsidered = true;
3  conditionalNormRecommenders = ∅;
4  foreach norm inference cycle do
5      Obtain Norms list (NL) ; /* By invoking
       Candidate Norm Identification and
       verification algorithm                  */
6      if NL ≠ ∅ then
7          foreach punished agent with the visibility threshold do
8              tempDistance =
               getDistanceFromNearestBin;
9              if tempDistance > maxDistanceFromBin then
10                 maxDistanceFromBin = tempDistance;
11             end
12         end
13         if conditionalNormReferralConsidered then
14             conditionalNormRecommenders =
               getAgentsFromVicinity ;
15             foreach agent ∈ conditionalNormRecommenders
               do
16                 if agent.maxDistanceFromBin >
                   maxDistanceFromBin then
17                     maxDistanceFromBin =
                       agent.maxDistanceFromBin;
18                 end
19             end
20         end
21     end
22 end
```

The pseudocode of an agent to identify the conditional norm of the park is given in Algorithm 1. In each norm inference cycle an agent will first identify a set of norms using the norm identification framework [23]. Let us assume that the agent has identified *prohibit(litter)* as the norm which is stored in its Norms List (NL). For each of the littering agents that were observed to be punished, an agent calculates the distance from the nearest bin to the punished agent using Chebyshev's distance metric [1][7]. The agent can

[6]The condition associated with a norm will be specific to the domain under consideration. In the park littering example, the condition can be either one or two-dimensional. For example, the distance between a littering agent and bin is a single dimensional entity. The littering zone can be modelled as a two dimensional entity if is defined using x and y coordinates (i.e. an agent should not litter within 5 metres from bin's x position and 10 metres from bin's y position). Some researchers have used a two dimensional representation for identifying norm conflicts [17, 28]. In this work we have used the distance metric.

[7]Chebyshev's distance also known as the Chessboard distance is the minimum number of steps required for a King to move from one square of the chessboard to another. In our implementation Chebyshev distance represents the minimum distance between an agent and the nearest bin.

choose to ask for referral from one or more agents from its vicinity threshold regarding the zone in which littering is prohibited (e.g. the von Neumann neighbourhood [2] of a certain radius). If the referrer's recommended distance is greater than distance observed by the agent the agent increases the distance.

We note that the algorithm presented here is specific to the domain model under consideration. A generic algorithm for identifying conditional norms will be desirable.

5. EXPERIMENTS

In this section we firstly describe the experimental set-up in subsection 5.1. In the rest of the sub-sections we describe the experiments that were conducted and the results obtained.

5.1 Experimental set-up

We model agents in our virtual society in a two-dimensional space. This virtual environment can be considered as a communal region such as a park shown in Figure 1. The agents explore and enjoy the park by moving around. Each agent has a visibility threshold. The visibility threshold of the agent is governed by a Chebyshev distance [1] of a certain length. An agent can observe actions of agents and the interactions that happen between two agents within its visibility threshold. There are three types of agents in the simulation. They are learning litterers (LL), non-litterers (NL) and non-littering punishers (NLP). There are four possible types of actions defined in the simulation system: *move, eat, litter* and *punish*. The LL agents can *move, eat* and *litter*. The NL agents can *move* and *eat* while the NLP agents can *move, eat* and *punish*. The agents' movement can be in one of the four directions: up, down, left or right. The agents that are at the edge of the two dimensional space can again re-appear in the opposite side (i.e. a toroidal grid is implemented). The agents are represented as circles using different colours. The LLs are green, the NLs are blue and the NLPs are red. The id and action that an agent currently does appear above the circle. All the agents make use of the norm inference component [23] to infer norms. The red squares that appear within the circles represent the identification of a norm. Rubbish bins in the simulation environment appear in orange.

The simulation parameters that were kept constant for all the experiments are given in Table 1. A sample simulation can be viewed from this link[8].

Table 1: Simulation parameters

Parameters	Values
Grid size	20*20
Total number of agents	20
Number of litterers	12
Number of non-litterers	4
Number of non-littering punishers	4
Visibility threshold	5
Number of rubbish bins	2
Radius of non-littering zone (maxDistanceFromBin)	10
Number of referrals (when used)	1

5.2 Experiment 1 - Conditional norm identification

The objective of the first experiment is to show that agents in a society infer conditional norms using the proposed mechanism. We also compare the rate of norm establishment in the society with the rate of conditional norm establishment in the society.

Figure 2 shows two lines that represent the proportion of agents with norms and the proportion of agents with conditional norms in a society, respectively. It can be seen from Figure 2 that even though the norm has been established in the society[9] in iteration 270, the conditional norm (i.e. the agent should not litter when it is within 10 metres from the bin), is not inferred in the society till iteration number 410. This is because the conditional norm identification process is invoked by an agent after it has found a norm. As the agents interact more and more in the society, they gather more evidence regarding the condition associated with the norm. If the norm does not change, then the correct condition associated with the norm is inferred eventually. When an agent does not infer a norm for certain amount of time or when the norm changes it will remove the norm and its associated conditions from its norms base.

5.3 Experiment 2 - Conditional norm identification with and without referral

An agent can expedite the process of identifying a conditional norm if it asks another agent for its evidence of the normative condition. We call this as the conditional norm referral process. It can be observed from Figure 3 that when the referral is used, the rate of establishment of the conditional norm increases. The agents ask for referral from one other agent in the society[10].

Figure 4 shows the progression of two agents towards the identification of the correct conditional norm (non-littering zone radius of 10) with and without referrals. The progression rates of the two agents are different because of their different paths of travel. If an agent observes more agents on its path, then it has a higher probability of inferring both the norm and the condition associated with the norm. It should be noted that the conditional norm establishment for these two agents improve when the referrals are used.

The two dashed lines in Figure 4 show the radius of the non-littering zone identified by the agents during the simulation. The agent which found the norm first (agent 1, iteration 90) was not the one to find the correct conditional norm first[11]. When agent 1 found the norm in iteration 90, the non-littering zone identified by the agent was 6 metres (shown using an arrow in the Figure). It found the correct conditional norm in iteration 380. Agent 2, albeit finding the norm second (iteration 110, non-littering zone radius of 7 metres), found the correct conditional norm faster (iteration 190). This again is governed by the number of agents an agent gets to observe (i.e. the path of travel).

The two solid lines show the radius of the non-littering zone identified by the agents when referrals are used. It is interesting to note that when the referral mechanism is used, the agent which found the norm first was also the one that found the normative condition first. This is because once the agent finds the norm it can ask the agents in the vicinity for referral instead of waiting for a long amount of time to find out the maximum distance from the bin from which a violation that was punished occurred.

5.4 Experiment 3 - Dynamic conditional norm identification

[8] http://www.youtube.com/watch?v=RODPrloHUa0

[9] We assume that a norm is established in a society if all the agents (100%) have inferred the norm. Researchers have used different criteria ranging from 35% to 100% [21].

[10] When the number of referees increases, the rate of conditional norm establishment increases. This has also been reported many other works in multi-agent systems [10, 33].

[11] The correct conditional norm is the non-littering zone of 10 metres.

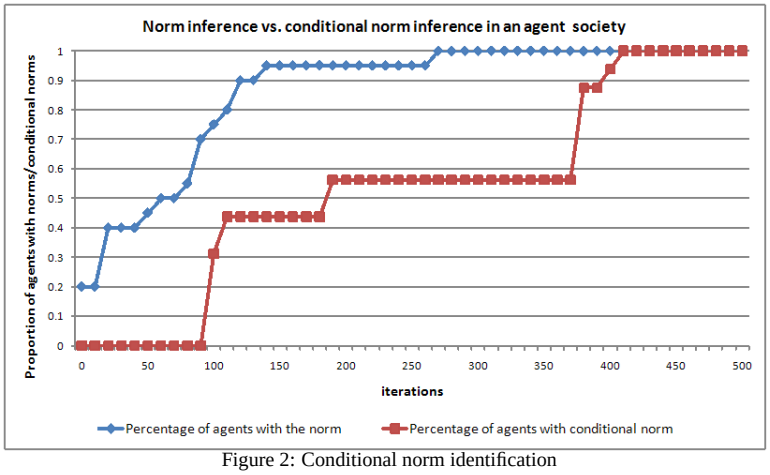


Figure 2: Conditional norm identification

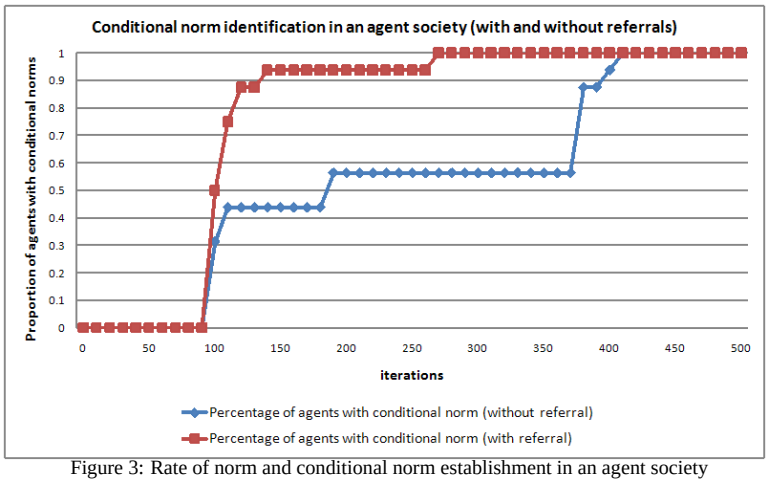


Figure 3: Rate of norm and conditional norm establishment in an agent society

An agent should have the ability to dynamically add newly identified norms and remove norms that do not hold. This experiment demonstrates that conditional norms can be added, removed and modified by an agent dynamically depending upon the environmental conditions. The ability to change norms is important for an adaptive agent so that it can flexibly adopt norms. An agent, on identifying a norm, evaluates whether the norm holds at regular intervals of time. If the norm does not hold, it removes the norm from its norm base. When it removes the norm it also removes the condition associated with the norm.

Figure 5 shows three lines that represent an agent identifying conditions associated with norms when the norm is changing. In this experiment, the punishers do not punish from iterations 1000 to 1250[12]. In this experiment, having identified a norm, an agent checks for the validity of the norm once again after 5 norm inference cycles (norm inference happens once every 10 iterations). If the norm is found again, then the agent does not delete the norm. If the norm is not found, it removes the norm and the conditions from its norm base. When the punishers do not punish, the norm is not inferred. As the norm is not inferred for 50 iterations (5 norm inference cycles from iteration 1010 to 1050), the agent removes the norm and the conditions associated with the norm.

It can be observed from Figure 5 that an agent (without referral) identifies a conditional norm in iteration 60, and the correct conditional norm in iteration 120. The agent loses the conditional norm in iteration 1050 when the norm is not found any more (as there aren't any punishers). It again finds a conditional norm in iteration 1280 and the correct conditional norm in iteration 1670. It can be observed that when the referral is used the conditional norm is identified earlier.

We also varied the radius of the non-littering zone (i.e. the punishment zone for littering). After iteration 1250 all NLP agents punished only those agents that littered within 5 metres from the bin (as opposed to 10 metres which was used in iterations 1 to 1000). It can be observed from Figure 5 that the agent inferred the new normative condition (radius = 5) [13]. Note that the agent

[12]There can be several reasons why punishers may stop punishing. The punishers can move from one society to another or can just stop punishing because their utility has gone below certain threshold.

[13]The simulation video can be found at

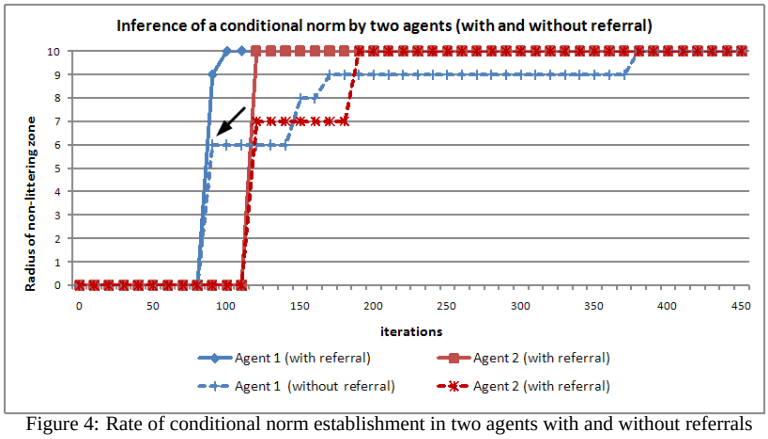


Figure 4: Rate of conditional norm establishment in two agents with and without referrals

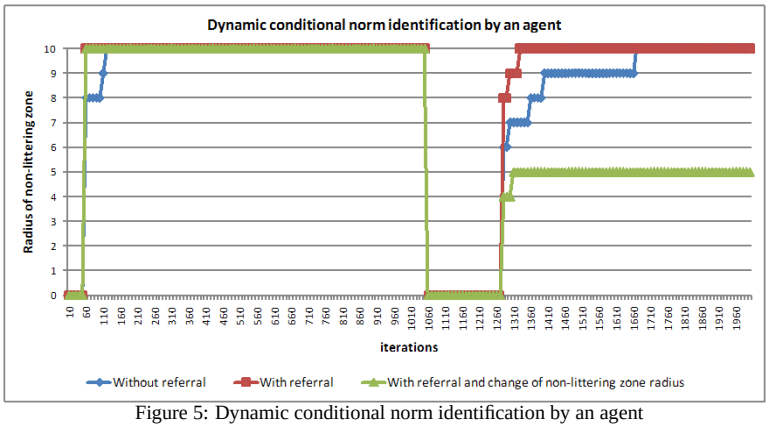


Figure 5: Dynamic conditional norm identification by an agent

has made use of referral in this case.

5.5 Experiment 4 - Comparison of utility of agents with and without conditional norm identification

The objective of this experiment is to compare the utility benefits of an agent when they identify norms with and without conditions. In this experiment, an agent has a utility value which we call the satisfaction level (S) which varies from 0 to 100.

An agent's satisfaction level (S) decreases in the following situations:

- When a litterer is punished, its utility decreases (-1).
- For all agents, littering activity results in the decrease of the utility. This is because each littering activity ruins the "commons" area (-1/number of agents in the society).

An agent's satisfaction level (S) increases (i.e. it gains utility) in the following situation:

- When a litterer litters, it gains utility in a society (+1).

We have experimented with the utility of the agent with and without conditional norm identification. An LL agent is better off by using conditional norm (CN) identification. Once identifying a norm an LL agent may choose to abstain from the action that is being prohibited. In the case of a conditional norm, it learns the exact condition under which it should not violate the norm. By this process, it can improve its utility. It can be observed from Figure 6 that an LL agent's utility increases (greater than 50) when it has identified the conditional norm than just identifying the norm without conditions (less than 50). =

For an NL agent, when it identified a norm (without a condition), the utility initially decreases but then stabilizes to a constant value because when all the agents inferred the norm, there aren't any littering actions in the society. When the NL agent identifies the conditional norm, its utility continues to decrease because whenever an LL agent litters outside the not-to-litter zone, its utility decreases[14]. The same applies to an LL agent, but the utility gains due to littering more than compensates for the utility lost because of other agents polluting the commons area.

The utilities of NLP agents are not discussed here because we assume these agents have other utility functions for punishing (e.g. a leader who wants to promote a smoother functioning of the soci-

http://www.youtube.com/watch?v=bT6CVJ6HU0I

[14] It should be noted that when the utility of an NL agent goes below a certain threshold, it can leave the society, or can become a litterer or become a punisher. This is explored in another work [24]. The objective here is to show that the conditional norm identification has an impact on the utility of the agents.

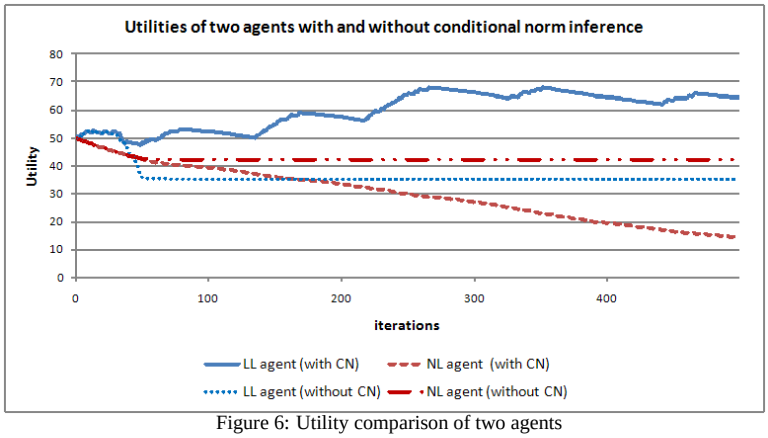


Figure 6: Utility comparison of two agents

ety, or an altruistic agent who does not care about its diminishing utility). We note that if non-altruistic punishers are present in the society, then the cost incurred by the non-altruistic punishers will play a role in the establishment of a norm in the society (see [24]).

6. DISCUSSION

The issue of conditional norm identification has not been dealt with by researchers in the field of normative multi-agent systems. To this end, we have experimented how a conditional norm can be identified by an agent in the context of park-littering. Identifying norms with conditions can be beneficial in several settings. For example, the norm identification architecture can be used to infer norms in Massively Multi-player Online Role Playing Games (MMORPGs) such as World of Warcraft (WoW). Players involved in massively multi-player games perform actions in an environment to achieve a goal. They may play as individuals or in groups. When playing a cooperation game (e.g. players forming groups to slay a dragon), individual players may be able to observe norms. For example, a dragon can only be slayed if two players are within certain distance from the dragon. An agent that identifies this condition (the distance) will be better-off than an agent that just infers the norm of cooperation (i.e. two players are needed to slay a dragon). The mechanism proposed in this paper can be used to identify norms with conditions. This mechanism can also be used in virtual environments such as Second Life to infer conditional norms.

Another application of identifying conditional norms is in the area of e-commerce. For example, in one society, the norm associated with the deadline for payment (i.e. obligations with deadlines as in [11]) may be set to 120 minutes after winning the item. Depending upon what an agent has observed, agents may have subtly different norms (e.g. one agent may notice that *pay* follows *win* after an average of 80 minutes while another may notice this happens after 100 minutes). Both these agents could still infer the norm but the deadlines they had noticed can be different. This may result in an unstable equilibrium with reference to the norms and hence conflict resolution mechanisms should be used to resolve them [17,28].

We note that we have modelled and experimented with a simple domain. The number and type of agents can easily be increased and the normative conditions identified can be richer and more complex depending upon the problem domain. However, we believe the main contribution is the identification of conditions associated with norms. We have also shown how an agent can dynamically add, remove and modify conditions associated with the norms.

7. CONCLUSION

This paper addresses the question of how conditional norms can be identified in an agent society using the norm inference architecture. Identification of conditional norms has been demonstrated in the context of a simple park-littering scenario. The ability of an agent to add, delete and modify a conditional norm has also been demonstrated. It has also been shown that identifying norms with conditions has an impact on the utility of the agents in the society.

8. REFERENCES

[1] Chebyshev distance. http://en.wikipedia.org/wiki/Chebyshev_distance.

[2] von neumann neighbourhood. http://mathworld.wolfram.com/vonNeumannNeighborhood.html.

[3] Huib Aldewereld, Frank Dignum, Andrés García-Camino, Pablo Noriega, Juan A. Rodríguez-Aguilar, and Carles Sierra. Operationalisation of norms for usage in electronic institutions. In *Proceedings of the fifth international joint conference on Autonomous Agents and MultiAgent Systems (AAMAS'06)*, pages 223–225, New York, NY, USA, 2006. ACM Press.

[4] Giulia Andrighetto, Rosaria Conte, Paolo Turrini, and Mario Paolucci. Emergence in the loop: Simulating the two way dynamics of norm innovation. In Guido Boella, Leon van der Torre, and Harko Verhagen, editors, *Normative Multi-agent Systems*, number 07122 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.

[5] Josep Ll. Arcos, Marc Esteva, Pablo Noriega, Juan A. Rodríguez-aguilar, and Carles Sierra. Environment engineering for multiagent systems. *Engineering Applications of Artificial Intelligence*, 18(2):191–204, 2005.

[6] Robert Axelrod. An evolutionary approach to norms. *The American Political Science Review*, 80(4):1095–1111, 1986.

[7] Albert Bandura. *Social Learning Theory*. General Learning Press, 1977.

[8] Guido Boella, Leendert Torre, and Harko Verhagen. Introduction to the special issue on normative multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 17(1):1–10, 2008.

[9] Magnus Boman. Norms in artificial decision making. *Artificial Intelligence and Law*, 7(1):17–35, 1999.

[10] Teddy Candale and Sandip Sen. Effect of referrals on convergence to satisficing distributions. In *Autonomous Agents and Multi-agent Systems*, pages 347–354, 2005.

[11] Henrique Lopes Cardoso and Eugenio Oliveira. Directed deadline obligations in agent-based business contracts. In *Proceedings of the international workshop on Coordination, Organization, Institutions and Norms in agent systems (COIN@AAMAS 2009)*, 2009.

[12] Cristiano Castelfranchi, Rosaria Conte, and Mario Paolucci. Normative reputation and the costs of compliance. *Journal of Artificial Societies and Social Simulation*, 1(3), 1998.

[13] Jon Elster. Social norms and economic theory. *The Journal of Economic Perspectives*, 3(4):99–117, 1989.

[14] Joshua M. Epstein. Learning to be thoughtless: Social norms and individual computation. *Computational Economics*, 18(1):9–24, 2001.

[15] Dorian Gaertner, Andrés García-Camino, Pablo Noriega, Juan A. Rodríguez-aguilar, and Wamberto W. Vasconcelos. Distributed norm management in regulated multi-agent systems. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems (AAMAS'07)*, pages 624–631, Honolulu, Hawaii, 2007. ACM.

[16] Andrew J. I. Jones and Marek Sergot. On the characterisation of law and computer systems: The normative systems perspective. In *Deontic Logic in Computer Science: Normative System Specification*, pages 275–307. John Wiley and Sons, 1993.

[17] Martin J. Kollingbaum, Wamberto Weber Vasconcelos, Andrés García-Camino, and Timothy J. Norman. Managing conflict resolution in norm-regulated environments. In *Engineering Societies in the Agents World (ESAW'07)*, pages 55–71, 2007.

[18] Fabiola López y López. *Social Powers and Norms: Impact on Agent Behaviour*. PhD thesis, Department of Electronics and Computer Science, University of Southampton, United Kingdom, 2003.

[19] Martin Neumann. A classification of normative architectures. In *Proceedings of World Congress on Social Simulation*, 2008.

[20] Michael Rymaszewski, Wagner James Au, Mark Wallace, Catherine Winters, Cory Ondrejka, Benjamin Batstone-Cunningham, and Philip Rosedale. *Second Life: The Official Guide*. SYBEX Inc., Alameda, CA, USA, 2006.

[21] Bastin Tony Roy Savarimuthu and Stephen Cranefield. A categorization of simulation works on norms. In Guido Boella, Pablo Noriega, Gabriella Pigozzi, and Harko Verhagen, editors, *Normative Multi-Agent Systems*, number 09121 in Dagstuhl Seminar Proceedings, pages 43–62, Dagstuhl, Germany, 2009. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany.

[22] Bastin Tony Roy Savarimuthu, Stephen Cranefield, Maryam Purvis, and Martin Purvis. Internal agent architecture for norm identification. In *Proceeding of the international workshop on Coordination, Organization, Institutions and Norms in agent systems (COIN@AAMAS 2009)*, pages 156–172, 2009.

[23] Bastin Tony Roy Savarimuthu, Stephen Cranefield, Maryam A. Purvis, and Martin K. Purvis. Norm identification in multi-agent societies. Discussion Paper 2010/03, Department of Information Science, University of Otago.

[24] Bastin Tony Roy Savarimuthu, Maryam Purvis, Martin K. Purvis, and Stephen Cranefield. Social norm emergence in virtual agent societies. In *DALT*, volume 5397 of *Lecture Notes in Computer Science*, pages 18–28. Springer, 2009.

[25] Sandip Sen and Stephane Airiau. Emergence of norms through social learning. In *Proceedings of Twentieth International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 1507–1512. AAAI Press, 2007.

[26] Yoav Shoham and Moshe Tennenholtz. Emergent conventions in multi-agent systems: Initial experimental results and observations. In *Proceedings of third International Conference on Principles of Knowledge Representation and Reasoning*, pages 225–231, San Mateo, CA, 1992. Morgan Kaufmann.

[27] Phillip Stoup. Athe development and failure of social norms in second life. *Duke Law Journal*, 58(2):311–344, 2008.

[28] Wamberto Weber Vasconcelos, Martin J. Kollingbaum, and Timothy J. Norman. Normative conflict resolution in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 19(2):124–152, 2009.

[29] Javier Vázquez-Salceda. Thesis: The role of norms and electronic institutions in multi-agent systems applied to complex domains. the harmonia framework. *AI Communications*, 16(3):209–212, 2003.

[30] Harko Verhagen. Simulation of the Learning of Norms. *Social Science Computer Review*, 19(3):296–306, 2001.

[31] Adam Walker and Michael Wooldridge. Understanding the emergence of conventions in multi-agent systems. In Victor Lesser, editor, *Proceedings of the First International Conference on Multi–Agent Systems*, pages 384–389, San Francisco, CA, 1995. MIT Press.

[32] R. J. Wieringa and J.-J. Ch. Meyer. Applications of deontic logic in computer science: a concise overview. In *Deontic logic in computer science: Normative system specification*, pages 17–40. John Wiley & Sons, Inc., New York, USA, 1994.

[33] Pinar Yolum and Munindar P. Singh. Emergent properties of referral systems. In *Autonomous Agents and Multi-agent Systems*, pages 592–599, 2003.

A Modeling Language to Model Norms

Viviane Torres da Silva* Christiano Braga Karen Figueiredo

Computer Science Department, Universidade Federal Fluminense (UFF)
Rua Passos da Pátria 156, Bloco E, 24210-240, Niterói, Brazil

{viviane.silva, cbraga, kfigueiredo}@ic.uff.br

ABSTRACT
Norms describe the permissions, prohibitions and obligations of agents in order to regulate their behavior. In this paper we investigate if the properties and characteristics of norms are being considered in MAS modeling languages, methodologies and organization models and if they give support to the checking of conflicts between the norms at design time. In addition, the paper presents the preliminary version of a normative modeling language called NormML that, although being in its infancy, is able to model several of the main properties and characteristics of the norms and to check the conflicts between them at design time.

Categories and Subject Descriptors
I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*

General Terms
Design, Languages, Theory.

Keywords
Norm, Modeling, Validation, Conflict, Metamodel.

1. INTRODUCTION
Norms in multi-agent systems are mechanism used to restrict the behavior of agents by describing the actions that must be performed or states that must be achieved (obligations), actions that can be performed or states that can be achieved (permissions) and actions that cannot be performed or states that cannot be achieved (prohibitions). They represent a way for agents to understand their responsibilities and the responsibilities of the others. Norms are used to cope with the autonomy, different interests and desires of the agents that cohabit the system.

Norms can be defined at design time together with the modeling of the system, or created at runtime by agents that have the power to do so [21]. In this paper we focus on the description of norms at design time. The modeling of norms is an important part of the specification of a system and should be treated as an important task of Multiagent System (MAS) design. The alignment of the norms with the elements that represent the system, such as its entities and the actions that they execute, is a fundamental activity because the norms specification relates such entities, their actions and the period during while the actions are being regulated. Thus, the redesign of the system may affect the specification of the norms and the redesign of the norms may influence the set of elements that represent the system.

Another important issue that must be considered while specifying the norms is the conflicts that may arise between them. A clear example of such conflicts occurs when there is a norm that prohibits an agent to perform a particular action and another that requires the same agent to perform the same action at the same period of time. When norms are defined at design time some of those conflicts can be detected and solved by, for instance, amending the conflicting norms, which might cause the system's redesign (by the inclusion of new actions, actors and roles, for example). By solving at least part of the conflicts at design time, it is possible to reduce the time the agents will spend executing such task at runtime.

Due to the interdependency between the modeling of norms and the modeling of the elements of the system and the importance of finding out conflicts and solving them at design time, it is important that the modeling languages and the notations used by methodologies and organizational models to model MAS make possible the modeling of the norms together with the modeling of the whole system and also provide mechanism for solving the conflicts at design time.

Taking this into account, the main goals of this paper are: (i) to investigate if the elements that compose norms can be modeled by using the MAS modeling languages and notations provided by methodologies and organizational model; (ii) to check if some dynamic characteristics of the norms, such as their creation, cancelation and delegation, can be modeled by them; and (iii) to explore the languages, methodologies and model in order to find out if they give support to the check of conflicts at design time. The paper also aims to present the preliminary version of the normative modeling language called NormML that is able to model several of the main properties and characteristics of the norms and to check the conflicts between them. Although being in its infancy, the language put forwards several desired attributes as presented in the paper.

The paper is organized as follows. In Section 2 we identify some of the main properties and characteristics of a norm. In Section 3

*The present work has been partially funded by the Spanish project "Agreement Technologies" (CONSOLIDER CSD2007-0022, INGENIO 2010), by the Spanish Ministry of Education and Science under project TIN2009-13839-C03-02 and by the Brazilian research councils CNPq under grant 553518/2009-7 and FAPERJ under grant E-26/110.959/2009.

we discuss about the support given by the modeling languages and the notations provided by the methodologies and organizational models analyzed to model such properties and characteristics and to check norm conflicts. Section 4 provides some background material and Section 5 introduces the preliminary version of our normative modeling language and tool used to automatically check for conflicts and query the norms model. Section 6 concludes the paper with final remarks and discusses future work.

2. PROPERTIES AND CHARACTERISTICS OF A NORM

In this section we stress the properties and characteristics of a norm by pointing out the main static aspects of a norm, i.e., the key elements that compose a norm, and its main dynamic aspects, i.e., the aspects related to its creation, cancelation and delegation.

The elements that compose a norm are based on the premise that *norms restrict the behavior of system entities during a period of time and define the sanctions applied when violated or fulfilled*. Such elements were found out after investigate ten specification and implementation languages used to describe and implement norms [1, 8, 13, 14, 17, 20, 21, 22, 30, 33]. The main elements that contribute to define the static aspects of the norms are the following six:

Deontic concept: The deontic logic refers to the logic of requests, commands, rules, laws, moral principles and judgments [23]. In multi-agent systems, such concepts have been used to describe behavior restrictions for the agents in the form of obligations (what the agent must execute), permissions (what the agent can execute) and prohibitions (what the agent cannot execute). Thus, one of the main properties of a norm is the identification of the type of restriction being defined, i.e., the identification of the deontic concepts associated with the norm.

Involved entities: Since norms are always defined to restrict the behavior of entities, the identification of such entities whose behavior is being restricted is fundamental. A norm may regulate the behavior of individuals (i.e., a given agent, or an agent while playing a given role) or the behavior of a group of individuals (i.e., all agents playing a given role, groups of agents, groups of agents playing roles or all agents in the system).

Actions: Since a norm defines restriction over the execution of entities, it is important to clearly represent the action being regulated. Such actions can be communicative ones, typically represented by the sending and receiving of a message, or non-communicative actions (such as to access and modify a resource, to enter in an organization, to move to another environment, etc.). In this paper we have not taken into account norms applied to states.

Activation constraints: The norms have a period during while they are active, i.e., during while their restrictions must be fulfilled. Norms can be activated by one constraint or a set of constraints that can be: the execution of actions, the specification of time intervals (before, after, between), the achievement of systems states or temporal aspects (such as dates), and also the activation / deactivation of another norm and the fulfillment / violation of a norm.

Sanctions: When a norm is violated the entity that has violated this norm may suffer a punishment and when a norm is fulfilled the entity who has followed the norm may receive a reward. Such rewards and punishments are called sanctions and should be described together with the norm specification.

Context: Norms are usually defined in a given context that determines the area of its application. A norm can, for instance, be described in the context of a given environment and should be fulfilled only by the agents executing in the environment or can be defined in the context of an organization and fulfilled only by the agents playing roles in the organization.

The two main dynamic aspects of a norm took into account in our research were the creation and cancelation of the norm. Besides these two aspects, we have also analyzed the delegation of norms to other entities since it is also considered by some languages/methodologies.

Creation: Besides being able to define norms at design time, it is also desired to be able to define the entities that have the power to create norm at runtime and the circumstance under which the norms can be created.

Cancellation: Norms defined at design or runtime can be cancelled during the system execution. In order to specify the constraints of valid cancelations, it is important to identify the entities that have the power to cancel the norms and the circumstances under which the norms are canceled.

Delegation: A norm (specified at design or runtime) applied to certain entity can be delegated by by the entity to another entity at runtime. After the delegation, the behavior of the entity receiving the norm will be restricted by the norm.The definition of a valid delegation includes concepts such as: which entity has the power to delegate the norm, which entities can be the responsible for the fulfillment of the norm and under which circumstances the norm can be delegated.

3. RELATED WORK

Although there are some works, such as the modeling languages AUML [27] and ANote [25] and the methodology MESSAGE [4] that do not support the modeling of norms, there are already many others that make possible the modeling of several properties and characteristics of a norm. From the set of two modeling languages [9, 37], seven methodologies [7, 15, 16, 19, 28, 29, 39] and three organization models [10, 11, 18] analyzed, no one is able to model all the elements described in the previous section. In this section we discuss those modeling languages, methodologies and organization models showing how they employ these elements and represent the concept related to the norms.

Most of modeling languages and methodologies make available the deontic concept permission in order to describe the actions that agents can execute, but only few languages [37], methodologies [39] and organization models [10, 11] provide the three deontic concepts: obligation, permission and prohibition. Methodologies such as [16, 19, 28, 29] and the organization model proposed in [18] do only offer the concepts obligation and permission since they consider that everything that is not permitted is automatically prohibited. In the Secure Tropos methodology [16] the concept of obligation can be represented by

the delegation relationship and the concept of permission by the ownership and trust relationships.

All languages, methodologies and organization models analyzed have a way to describe the entities to which the norm applies. However, none of them provides support to describe all the types of entities identified in Section 2. The majority provide support to describe a norm for a particular role, as in [7, 9, 10, 16, 18, 19, 28, 29, 37, 39] or for an agent playing a role [11, 15]. But only the ones presented in [9, 10, 11, 15, 16, 28, 29, 37] allow the description of norms that apply to a group of individuals. The Secure Tropos methodology also allows the designer to describe the system itself as an entity and to define norms that can be applied to the system as a whole.

All the modeling languages, methodologies and models analyzed provide a way to restrict non-communicative actions, i.e., actions not related to the sending and receiving of messages. In [7, 9, 10, 11, 15, 16, 29, 39] it is also possible to restrict communicative ones. In ROADMAP [19], that is one of the proposed extensions for Gaia [39], the user can only restrict the access to objects, roles and protocols of the system.

The modeling languages and methodologies analyzed present several ways to describe the period during while a norm is active, i.e., to describe the restrictions for their activation and deactivation. In [7, 9, 10, 11, 15, 18, 29, 39] it is possible to express time constraints as dates and in [7, 9, 10, 11, 15, 19, 29, 39] it is possible to describe periods of time as restrictions. In [7, 9, 10, 11, 15, 29, 39] the execution of actions are used to restrict the period during while the norm is active, and in [10, 11] the activation/deactivation of a norm and the fulfillment/violation of another norm can also be used as restrictions. According to [24], the SODA [28] formalism is still being developed so we cannot affirm the types of restrictions that such methodology will support.

A small number of languages and methodologies consider that norms can be violated, and only few of them provide a way for describing sanctions. The AORML language [37] assumes that commitments (or obligations) between entities of the system can be violated, and, as consequence, a sanction should be applied. But the language does not offer a way to describe this sanction. The organizational models OperA [10], MASQ [11] and MOISE+ [18] consider that norms can be violated, and, excluding MOISE+, they have mechanisms to describe sanctions.

The O-MaSE methodology [15] group norms in two kinds of policies: law policies and guidance policies. Only the guidance policies can be violated but there is not a way to define sanctions for such violations. The Gaia and PASSI [7] methodologies express norms as organization rules that cannot be violated, and so there is no need to define a sanction mechanism. No language or methodology allows the description of rewards in case of the fulfillment of a norm.

All languages, methodologies and organizational models define the norms in an organizational context, i.e., the norms are valid for all individuals and groups belonging to this organization. Besides AORML, that offer support to express obligations between two agents (as commitments) in the context of an interaction, methodologies such as [16, 29, 39] and the organizational model [10] also allow the description of norms in such context. Moreover, OperA and Prometheus [29] it is possible to describe a norm in a context that represents the transition of scenes.

Only AORML, OperA, ROADMAP and SODA provide support to model the dynamic aspects of norms, like creation, cancellation and delegation. Unlike other extensions of the Gaia methodology, ROADMAP has mechanisms for describing permissions for a role to modify the norms applied to another role. In Secure Tropos it is only possible to define the delegation of norms between the entities of the system; the other dynamic aspects are not included.

The SODA methodology uses security concepts based on role-based access control design (RBAC) [12] to describe access permissions to resources and services of the system. Thus, the methodology defines special permissions that can be used to specify roles that can create, cancel or modify the norms of the system (in this case, called permission policies).

In addition to the elements presented, another interesting characteristic to be considered when analyzing the modeling languages, methodologies and organizational model is the ability to detect conflicts between the norms of the system at design phase. However, few works discuss this issue.

The AORML language assumes that there is a normative inconsistency when there is at the same time a permission and a prohibition, or a prohibition and an obligation to the same action. It considers that obligations already have a permission embedded, so there is no conflict in this sense. Although the language considers that conflicts can occur, it does not have an automatic mechanism to detect these conflicts.

The Secure Tropos methodology defines eight properties to be used in an automatic verification of conflicts, including the validation of conflicts between the system's obligations and permissions.

The OperA organizational model allows the automatic verification of conflicts between the norms that apply to a given entity. However, such mechanism does not give support to the checking of conflicts between norms applied to different entity types, i.e., between the norms applied to a group and the norms applied to roles or agents themselves.

4. BACKGROUND

In this section we briefly provide background material for the rest of this paper. NormML is a UML-based modeling language for the specification of norms that constraint the behavior of agents in MAS. The choice for UML as metalanguage allows for an easy integration of NormML with UML-based MAS modeling languages such as AUML, AML [9] and MAS-ML [31]. Moreover, metamodel-based validation techniques may be applied to norms specified in NormML. Therefore, Section 4.1 introduces basic notions of models and metamodels, necessary to understand the design of NormML.

Our modeling language was designed with the perception that norm specification in MAS design and security policy specification in RBAC design are closely coupled issues. RBAC security policies specify the *permissions* that a *user* has under a given *role*, while trying to access system *resources*. In MAS we specify the *norms* that regulate the behavior (or *actions*) of an *agent* playing a given *role*. Although we consider security policies

and norms couple issues, norms can be violated since they only define how agents *should* behave.

In Section 4.2 we introduce SecureUML [3], a domain-specific language (DSL) for modeling role based access control policies. It has been applied successfully both in academic projects [3] and industrial ones [6]. The work presented in this paper builds on previous work by the first and second authors. The reason why SecureUML was chosen is because it has a well-defined syntax, given by its metamodel, a formal semantics [2] and is designed specifically for RBAC modeling.

In this paper we want to explore the modeling of norms using RBAC concepts. As a DSL for RBAC, SecureUML provides an elegant way for modeling norms in a precise and effective way.

4.1 Models and Metamodels

A modeling language provides a vocabulary (concepts and relations) for creating *models*. Such vocabulary is described by the *metamodel* of the modeling language which elements formalize the language concepts and their relationships. A metamodel may include invariants that specify additional properties that the models must fulfill as instances of the metamodel. Such invariants specify the *well-formedness* conditions (or *well-formed rules*) of a model with respect to its metamodel and the *consistency* conditions between metamodel concepts.

When UML is chosen as metalanguage, a metamodel is represented by a class diagram and its invariants are written in OCL (Object Constraint Language) [26]. This is the choice followed in this paper.

4.2 SecureUML

SecureUML provides a language for modelling *Roles*, *Permissions*, *Actions*, *Resources*, and *Authorization Constraints*, along with the relationships between permissions and roles, actions and permissions, resources and actions, and constraints and permissions. The actions described in the language can be either *Atomic* or *Composite*. The atomic actions are intended to map directly onto actual operations of the modeled system (delete, update, read, create and execute). The composite actions are used to hierarchically group atomic ones.

SecureUML leaves open what the protected resources are and which actions they offer to clients. ComponentUML [3] is a simple language for modeling component-based systems that provides provides a subset of UML class models: entities can be related by associations and may have attributes and methods. Therefore, *Entity*, *Attribute*, *Method*, *Association* and *AssociationEnd* are the possible protected resources. By using such SecureUML+ComponentUML[1] it is possible, for instance, to specify the permissions a user playing a given role must have to execute a method (or to update an attribute) of a resource. In order to do so, it is necessary to instantiate the metaclasses *User*, *Role*, *Permission*, *ActionExecute*, *Method* (or *ActionUpdate*) and *Attribute*.

[1] The metamodel of SecureUML+ComponentUML (from now referred as SecureUML metamodel) is available at http://www.ic.uff.br/~viviane.silva/normML/secureUML.pdf

5. NormML: A NORTMATIVE MODELING LANGUAGE

As state before, norms are viewed as security policies. While in SecureUML it is possible to define *permissions* a *user* has, i.e., the constraints that a user, in a given role, must fulfill to perform actions over the system resources, in NormML it is possible to define the *norms* an *entity* must obey, i.e., it is possible to describe the set of *actions* an *agent*, a *role* or an *agent playing a role* is *obliged*, *permitted* or *prohibited* to execute conditioned by the execution of other *actions*.

The NormML metamodel extends the SecureUML metamodel with the following basic elements: *Norm*, *Agent* and *AgentAction* (Section 5.1). The NormML metamodel also includes a set of *invariants* that guarantees the well-formedness of a norm and several *operations* that are used to identify conflicts between two given norms (Section 5.2).

5.1 The NormML Metamodel

Figure 1[2] presents the NormML metamodel by highlighting the new classes added to the SecureUML metamodel. A norm corresponds to an instance of the NormML metamodel, i.e., it is defined by instantiating several metaclasses and their relationships from the NormML metamodel. A norm may be either a permission (by instantiating the metaclass *NormPermission)*, a prohibition (by instantiating the metaclass *NormProhibition)* or an obligation (by instantiating the metaclass *NormObligation)*.

A norm may constraint the behavior of *Agents* itself, the behavior of all agents playing a given *Role*, or the behavior of a specific agent it is playing a given role, captured by the *Agent<->Role relationship*.

NormML inherits four resource kinds from SecureUML: *Attribute*, *Method*, *Entity* and *AssociationEnd*. It extends the set of resources with agent and roles' actions represented by the metaclass *AgentAction*. Thus, it is possible to describe norms to control the access to attributes, methods, objects and association ends, and also to control the execution of the actions of agents and roles.

Each resource kind has a set of actions that can be used to control the access to the resource. For instance, attributes are associated with the actions *read*, *update* and *full access (read+update)*. In the case of restrictions applied to actions of agents and roles (*AgentAction* metaclass), the behavior that must be used is the *execution* of the action (*ActionExecute*). Note that *AgentAction* is the resource and *ActionExecute* is the action being used to control or restrict the access to the resource.

[2] Some of SecureUML metaclasses are not presented for readability purposes.

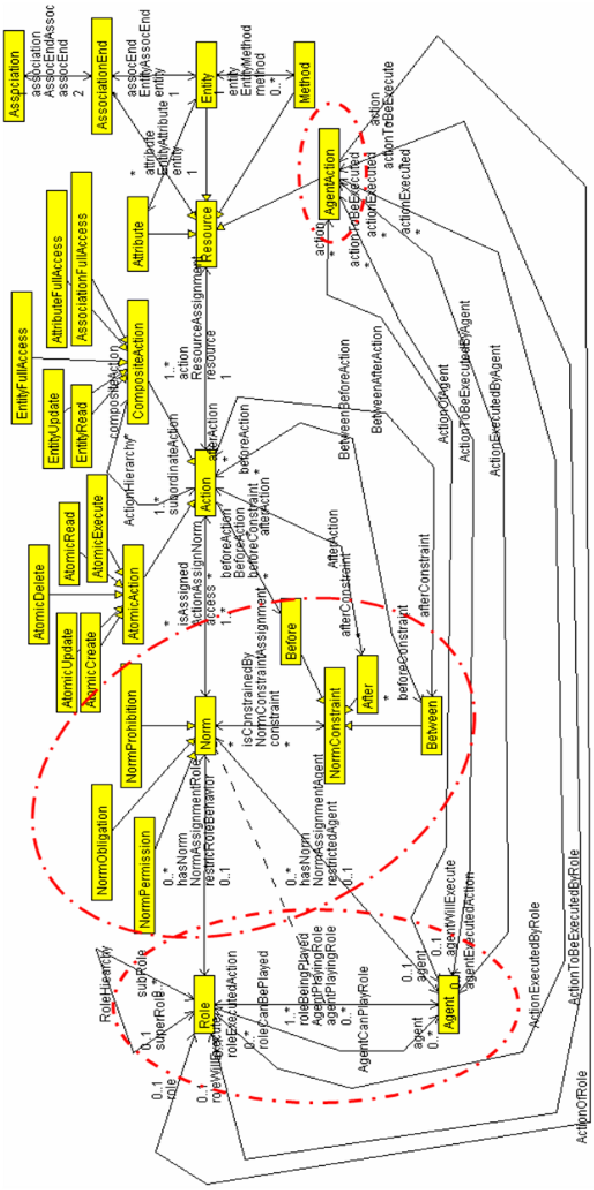


Figure 1. The NormML Metamodel

Furthermore, NormML allows for the specification of the time period that a norm is *active*, which is represented by the metaclass *NormConstrain*. If a norm is conditioned by a *Before* clause, it means that the norm is active before the execution of the action(s) described in the *Before* clause. If a norm is conditioned by an *After* clause, it means that the norm is active only after the execution of the action(s) described in the *After* clause. In the case of a *Between* clause, the norm is only active during the period delimited by two groups of actions.

In order to illustrate the use of NormML to model the norms of a MAS, consider following norms *N1*, *N2* and *N3*. Figure 2, Figure 3 and Figure 4 illustrates the norm diagram of *N1, N2 and N3*.

N1: Seller is *obliged* to give the good to the buyer *after* the given buyer paid for it.

N2: Seller is *permitted* to update the price of a good *before* a buyer pays for it.

N3: Buyer is *prohibited* to return a good he/she has bought.

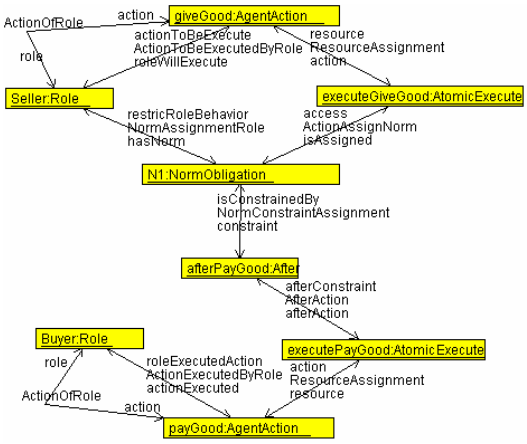


Figure 2. Norm N1 described by using NormML

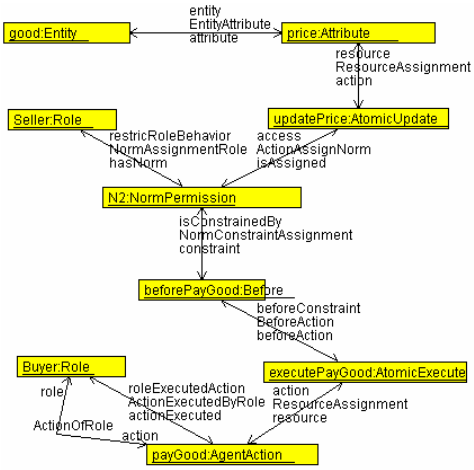


Figure 3. Norm N2 described by using NormML

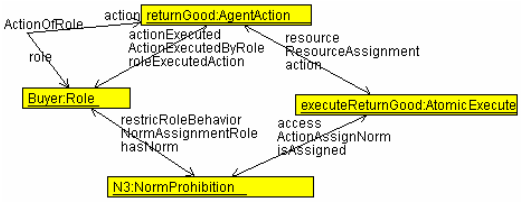


Figure 4. Norm N3 described by using NormML

5.2 Validating the Norms

The process of validating a norm encompasses two steps. First, the norm, as an instance of the NormML metamodel, is checked according to the invariants of the metamodel. The invariants check if the norm is well-formed according to the metamodel

specification. The second step checks if any given two norms are in conflict. The current version of NormML is able to check the invariants and conflicts by using a set of operations described in OCL[3].

5.2.1 Well-formed norms

Not all the norms that can be instantiated from the metamodel are well-formed. Below we describe two examples of well-formed rules of the NormML metamodel. Those were chosen since they represent the two sets of well-formed rules defined: rules that restrict the relationship between the metaclasses already defined in SecureUML and the metaclasses added by NormML and rules that are related to the specification of the norms themselves.

WFR1: The resource AgentAction cannot be constrained by Permission. Although the metaclass *Permission* is defined in the SecureUML metamodel to define the permissions a user has over resources, the resource *AgentAction* can only be used by *Norms* to restrict the actions of an agent. Thus, it should be constrained by *NormPermission*.

WFR2: A norm that regulates the execution of a given action cannot be conditioned by the execution of the same action by the same agent. An agent cannot be obliged, permitted or prohibited to execute an action conditioned to the execution of such action. This rule uses four operations in order to guarantee that the action being regulated by the norm is not in the set of actions of the *Before*, *After* or *Between* constraints.

5.2.2 Checking for Conflicts

After verifying the well-formedness of the norms, it is important to check if there are conflicts between the norms. The following situations are checked:

Permission and Prohibition: One states a permission and another one a prohibition to execute the same action and such norms are active during the same period of time or during periods of time that intersects. The conflict occurs because the agent is permitted and prohibited to execute an action at the same time. Example:

N3a: Buyer is *prohibited* to return a good it has bought.

N3b: Buyer is *permitted* to return a good it has bought *before* using it.

The activation time of N3a and N3b intersects since N3a states an unlimited prohibition. Thus, these norms are in conflict.

Obligation and Prohibition: One norm states an obligation and another one a prohibition over the same action and such norms are active during the same period of time or during periods of time that intersect. The conflict occurs because the agent is obliged and prohibited to execute an action at the same time. Example:

N1a: Seller is *obliged* to give the good to buyer *after* the given buyer paid for it.

N1b: Seller is *prohibited* to give the good to buyer *before* the latter pays for it.

The activation time of N1a and N1b do not intersect. These norms are not in conflict since the seller is not being obliged and prohibited to execute the same action during the same period of time.

Permission and Obligation: One norm states a permission and another one an obligation over the same action and such norms are *not active during the same period of time.* The period during while the agent is obliged to execute the action does not coincide with the period during while the agent is obliged to execute such action. Example:

N2a: Seller is *permitted* to update the price of a good *before* a buyer pays for it.

N2b: Seller is *obliged* to update the price of a good *after* a buyer pays for it.

The activation time of N2a and N2b do not intersect, thus these norms are in conflict.

In addition, we also consider that a conflict can be caused due to the relationship between an agent and the roles it is playing.

- *A norm applied to a role and another one applied to an agent may be in conflict:* A norm applied to a role restricts the behavior of all agents playing such role. Therefore, when searching for conflicts, it is important to check the incompatible norms that are applied to roles and also the ones applied to agents that are able to play such roles. Note that agents can play several roles.

- *A norm applied to a role and another one applied to an agent playing the role may be in conflict:* Since the norm applied to a role regulates the behavior of all agents applying such role, when searching for conflicts, it is important to check the incompatible norms that are applied to roles and to agents playing roles.

- *A norm applied to an agent and another one applied to the agent playing a role may be in conflict:* Since both norms will regulate the behavior of the same agent, when searching for conflicts it is important to check the incompatible norms that are applied to agents and to agents playing roles.

Note that two norms applied to different roles (that are not hierarchically related) are never in conflict even though the same agent can play both roles. Although an agent can play more than one role at the same time, an action is always executed in the context of one role. We understand that an agent must be able to obey each norm separately while playing the roles.

The mechanism used to detect conflicts proposed in our paper is based on the approach presented in [33]. We extend such approach to consider conflicts between norms that state permissions and obligations—the authors in [33] only consider permissions and prohibitions or obligations and prohibitions—and to deal with activation time that is related to the execution of actions—the activation time proposed in [33] is related to values associated with attributes.

5.2.3 The Use of MOVA to Model, Validate and Query the Norms

MOVA (Modeling and Validation group) tool [5] was used as a modeling tool (i) to describe the NormML metamodel, (ii) to create the normative models, (iii) to check the well-formedness of the norms and their conflicts, and also (iv) to inspect the normative models. MOVA allows for the creation of class diagrams, the definition of a set of invariants and operations over such diagrams and checking if object diagrams respect the invariants defined in class diagrams. We used MOVA to define

[3] Some of the well-formed rules of the NormML metamodel and all the OCL operations used to check for conflicts are available in http://www.ic.uff.br/~viviane.silva/normML/normML.pdf

the NormML *metamodel* as a class diagram and to describe the *well-formed rules* of the metamodel as invariants of the class diagram. The *normative models* were then described as object diagrams and checked if they comply with these invariants.

By using MOVA it is also possible to query the object diagram (i.e., to define queries over such models) by using the operations defined in the class diagram. Such mechanism was used not only to *check the conflicts* between the modeled norms by defining operations that investigate the possible conflicts (according to the rules described in Section 5.2.2) but also to *explore the normative models* themselves. Such investigation is fundamental when dealing with large-scale MAS that typically define a large number of norms. After describing hundreds norms it is almost impossible to find out, without the helping of a tool, all the norms applied to a role, for instance.

6. CONCLUSION AND FUTURE WORK

We have presented some of the main properties and characteristics of a norm and discussed how several MAS modeling languages and the notations provide by methodologies and organizational models give support to the modeling of these elements and to the checking of conflicts between norms.

In Section 3, it is clear that none of the analyzed modeling languages and notations give support to the modeling of all elements described in Section 2 together with the checking of conflicts. With this in mind and with the aim of creating a normative modeling language that could be easily integrated with the MAS modeling languages based on UML—since a number of MAS language extends UML—the preliminary version of the NormML was proposed.

NormML is a normative modeling language that builds on RBAC concepts to model the norms of MAS. Our approach follows others that have also used RBAC as a basis such as [32, 35, 36, 38]. The work most similar to ours is SODA since it is also based on RBAC and is able to model norms. However, in SODA it is not possible to model obligations and prohibitions, to define sanctions and, since it is not formalized yet, it is not clear what are the set of possible activation constraints of a norm.

The NormML metamodel was defined according to the normative grammar proposed in [30]. This grammar extends the normative language proposed by Garcia-Camino et al. [13] with the notion of non-dialogical actions proposed by Vazquez-Salceda et al. [34] and with the definition of sanctions and relationships between norms stated by Lopez y Lopez et al. in [21][22]. However, the current version of NormML does not contemplate all the elements defined in the grammar.

By using the current version of NormML it is possible (i) to model permissions, prohibitions and obligations; (ii) to regulate the behavior of individuals; (iii) to define norms that restrict the execution of non-communicative actions; (iv) to define activation constraints based on the execution of actions; and (v) to define norms in the context of organizations. The language also gives support to the checking of conflicts among norms. As stated before in Section 5.2, the rules that check for conflicts are implemented in OCL as operations and queries.

We are in the process of extending the language (i) to define norms that control the behavior of groups of individuals; (ii) to define norms that restrict the execution of communicative actions and the achievement of states; (iii) to define different activation constraints, such as deadlines; (iv) to define sanctions associated with the norms; and (v) to associate the norms with different contexts.

It is also our intension to define a sequence diagram for NormML to describe the sequence of the executed actions. By using such diagram it will be possible to: (i) represent dynamic aspects as the creation, cancellation and delegation of a norm; (ii) define norms in an interaction context; (iii) check conflicts that depend on the sequence of the executed actions; and (iv) identify the norms that are active and the ones that were violated.

7. REFERENCES

[1] Aldewereld, H., Dignum, F., Garcia-Camino, A., Noriega, P., Rodriguez-Aguilar, J. and Sierra, C. 2006. Operationalisation of norms for usage in electronic institutions. In Proc. 5th AAMAS, pp.223-225.

[2] Basin, D., Clavel, M., Doser, J. and Egea, M. 2009. Automated analysis of security-design models. Inf. Software Technology, 51(5), 2009, pp: 815—831.

[3] Basin, D., Doser, J. and Lodderstedt, T. 2006. Model driven security: From UML models to access control infrastructures. ACM Transection on Software Engineering and Methodologies,15(1), pp.39-91.

[4] Caire, G., Coulier, W., Garijo, F., Gomez, J., Pavon, J., Leal, F., Chainho, P., Kearney, P., Stark, J., Evans, R. and Massonet, P. 2002. Agent Oriented Analysis Using Message/UML. In AOSE II, LNCS 2222, pp. 119-135.

[5] Clavel, M., Egea, M., Silva, V. 2007. The MOVA Tool: A Rewriting-Based UML Modeling,Measuring,and Validation Tool. In Proc. of Workshop de Dem. de Herramientas de las Jornadas de Ing. del Software y Banco de Datos,pp.393-394.

[6] Clavel, M., Silva, V., Braga, C. and Egea, M. 2008. Model-Driven Security in Practice: An Industrial Experience. In Proc. 4th European Conf. on MDA, pp: 326-337.

[7] Cossentino, M. 2005. From requirements to code with the PASSI methodology. In Agent-oriented Methods, Idea group, pp. 79-106.

[8] Cranefield, S. 2007. Modelling and Monitoring Social Expectations in Multi-Agent Systems. In COIN II, LNCS 4386, pp. 308-321.

[9] Danc, J. 2008. Formal Specification of AML. Department of Computer Science Faculty of Mathematics, Physics and Informatics Comenius University Formal Specification of AML Master's Thesis Ján Danc Advisor: Mgr. Bratislava.

[10] Dignum, V. 2004. A Model for Organizational Interaction: Based on Agents, Founded in Logic. PhD dissertation, Universiteit Utrecht. SIKS dissertation series 2004-1.

[11] Ferber J., Stratulat T. and Tranier J. 2009. Towards an Integral Approach of Organizations: the MASQ approach in Multi-Agent Systems. In Multi-agent Systems: Semantics and Dynamics of Organizational Models. IGI.

[12] Ferraiolo, D. F., Kuhn, D. R., and Chandramouli, R. 2007. Role-Based Access Control. Artech House Publishers, 2nd Edition.

[13] García-Camino, A., Noriega, P. and Rodríguez-Aguilar, J. 2005. Implementing Norms in Electronic Institutions. In Proc. 4th AAMAS, ACM Press, pp. 667-673.

[14] García-Camino, A., Rodríguez-Aguilar, J., Sierra, C. and Vasconcelos, W. 2006. Norm-Oriented Programming of Electronic Institutions. In Proc. 5th AAMAS, ACM Press, pp. 670-672

[15] Garcia-Ojeda, J., DeLoach, S., Robby, O. and Valenzuela, J. 2008. O-MaSE: A customizable approach to developing multiagent development processes. In AOSE VIII, LNCS 4951, Springer, pp.1-15.

[16] Giorgini, P., Mouratidis, H. and Zannone, N. 2006. Modelling Security and Trust with Secure Tropos. In Integrating Security and Software Engineering: Advances and Future Vision. Idea Group.

[17] Governatori, G. and Rotolo, A. 2004. Defeasible Logic: Agency, Intention and Obligation. In Deontic Logic in Computer Science, 7th Int. Workshop on Deontic Logic in Computer Science, LNAI 3065, Springer, pp. 114-128.

[18] Hübner, J. F., Sichman, J. S. and Olivier, B. 2002. A Model for the Structural, Functional, and Deontic Specification of Organizations in Multiagent Systems. In Proc. 16th SBIA, LNAI 2507.

[19] Juan, T., Pierce, A. and Sterling, L. 2002. ROADMAP: Extending the Gaia Methodology for complex open systems. In Proc. 1st AAMAS, pp. 3-10, ACM Press, Bologna (I).

[20] Lomuscio, A. and Sergot, M. 2004. A formalization of violation, error recovery, and enforcement in the bit transmission problem. Journal of Applied Logic, vol. 2, no 1, pp.93-116.

[21] López y López, F. 2003. Social Power and Norms: Impact on agent behavior. PhD thesis, Univ. of Southampton, Faculty of Engineering and Applied Science, Department of Electronics and Computer Science.

[22] López y López, F., Luck M. and d'Inverno, M. 2002. Constraining autonomy through norms. In Proceedings of the 1st AAMAS, ACM Press, pp. 674-681.

[23] Meyer, J. J. and Wieringa, R. J. 1991. Deontic Logic in Computer Science: Normative System Specification. John Wiley and Sons.

[24] Molesini, A., Denti, E. and Omicini, A. 2009. RBAC-MAS & SODA: Experimenting RBAC in AOSE Engineering Societies in the Agents World IX, LNCS 5485.

[25] Noya, R. C. and Lucena, C. J. P. 2005. The ANote Modeling Language for Agent-Oriented Specification.In Sof. Eng. for Multi-Agent Systems III, LNCS 3390, pp.198-212.

[26] Object Management group, OCL Specification, OMG. Available in http://www.omg.org/docs/ptc/03-10-14.pdf.

[27] Odell, J., Parunak, H., and Bauer, B. 2000. Extending UML for Agents. In Proc. Agent-Oriented Information Systems Workshop at National Conf. of AI, pp. 3-17.

[28] Omicini, A. 2001. SODA: Societies and Infrastructures in the Analysis and Design of Agent-based Systems. In Agent-Oriented Software Engineering, LNCS 1957.

[29] Padgham, L. and Winikoff, M. 2002. Prometheus: A Methodology for Developing Intelligent Agents. In Proc. of Agent-Oriented Soft. Engineering Workshop, pp. 174-185.

[30] Silva, V. 2008. From the Specification to the Implementation of Norms: An Automatic Approach to Generate Rules from Norms to Govern the Behaviour of Agents. In IJAAMAS, Special Issue on Norms in Multi-Agent Systems, vol.17, no.1, Springer-Verlag, pp. 113-155.

[31] Silva, V., Choren R. and Lucena, C. 2008. MAS-ML: A Multi-Agent System Modelling Language. In Int. Journal of Agent-Oriented Soft. Eng., Special Issue on Modeling Languages for Agent Systems, vol.2, no.4, pp. 382-421.

[32] Tinnemeier, N., Dastani, M. and Meyer, J.-J. 2009. Roles and norms for programming agent organizations. In Proc. of 8th AAMAS, pp.10-15.

[33] Vasconcelos, W., Kollingbaum, M. and Norman, T. 2007. Resolving Conflict and Inconsistency in Norm-Regulated Virtual Organizations. In Proc. AAMAS'07.

[34] Vázquez-Salceda, J., Aldewereld, H. and Dignum, F. 2004. Implementing Norms in Multiagent Systems. In Multiagent System Technologies LNAI 3187. Springer, pp. 313-327.

[35] Villata, S. 2009. NorMAS-RE: a Normative Multiagent Approach to Requirements Engineering. Dagstuhl Seminar Proceedings 09121.

[36] Viroli, M., Ricci, A. and Omicini, A. 2005. An organisation infrastructure for Multi-Agent Systems based on Agent Coordination Contexts. In Proc. 4th AAMAS, pp: 1189-1190.

[37] Wagner, G. 2003. The Agent-Object-Relationship Meta-Model: Towards a Unified View of State and Behavior. Information Systems. 28(5), pp. 475–504.

[38] Yamazaki, S., Hiraishi, H. and Mizoguchi, F. 2004. Designing an Agent-Based RBAC System for Dynamic Security Policy. In Proc. 13th Int. Workshops on Enabling Tech.: Infra. for Collaborative Enterprises, pp: 199-204.

[39] Zambonelli, F., Jennings, N. R. and Wooldridge, M. J. 2003. Developing multiagent systems: The Gaia methodology. ACM Transactions on Software Engineering and Methodology, 12(3):417-470.

Towards a Model of Social Coherence in Multi-Agent Organizations

Erick Martínez*
SIAT
Simon Fraser University
Vancouver, Canada
emartinez@sfu.ca

Ivan Kwiatkowski†
ISIMA
Clermont-Ferrand, France
kwiatkow@poste.isima.fr

Philippe Pasquier
SIAT
Simon Fraser University
Vancouver, Canada
pasquier@sfu.ca

ABSTRACT

We propose a *social coherence*-based model and *simulation framework* to study the dynamics of multi-agent organizations. This model rests on the notion of *social commitment* to represent all the agents' explicit inter-dependencies including *roles* and *organizational structures*. A local coherentist approach is used that, along with a *sanction policy*, ensures *social control* in the system and the *emergence* of *social coherence*. We illustrate the model and the simulator with a simple experiment comparing two *sanction policies*.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*intelligent agents, multiagent systems*

General Terms

Algorithms, Experimentation, Theory

Keywords

Social and organizational structure, social commitments, agent reasoning, social control

1. INTRODUCTION AND MOTIVATIONS

Research in the area of Computational Organization Theory [3] and multi-agent systems (MAS) has resulted in a large number of models capturing different aspects of organizational behaviour [18, 19, 1, 6]. This paper presents a model and *simulation framework* to study the social dynamics of multi-agent organizations. The model uses the notion of *social commitment* (defined in Section 2) as the main building block to represent all the inter-dependencies between *social entities*. *Sanction policies* provide *social control* mechanisms (defined in Section 3) to regulate the enforcement of *social commitments*. Our model extends previous work on *cognitive coherence* [15, 16] by showing how the *coherence principle* can drive the emergence of *social behaviour*. In particular, by organizing agent behaviour in a way that makes global *social coherence* (formalized in Section 4) emerge from the local *cognitive coherence* of interacting agents.

We strive to build a simple minimalist model, where *social behaviour* emerges from local behaviour, enabling us to study the social dynamics of multi-agent organizations. This paper advances the state of the art by proposing a unified yet computational and operational view of the social aspects of multi-agent systems. We also present a sample pizza delivery domain (Section 5), and illustrate the use of the model and simulator with a simple experiment (Section 6) to investigate *social control* mechanisms while comparing two *sanction policies*. Then, we discuss our work while relating it to other research (Section 7). Finally, we conclude and discusses future work (Section 8).

*School of Interactive Arts and Technology.

†Institut Supérieur d'Informatique, de Modélisation et de leurs Applications.

2. SOCIAL MODELLING

2.1 Handling Actions

We represent atomic actions as (possibly) parametrized predicate formulas with unique names. We use a discreet instant-based sequential model of time where actions are assumed to be instantaneous. However, each action requires a *preparation time* expressed in time steps.

DEFINITION 1. ***(Primitive or Atomic Action)*** *Given the non-empty set $\mathcal{X}$ of all atomic actions in the system, a* ***primitive action*** *$\alpha \in \mathcal{X}$ is represented as a tuple $\alpha = \langle \alpha(\vec{x}), \Delta_\alpha \rangle$, where:*

- *$\alpha(\vec{x})$ is a predicate formula s.t. $\alpha(\vec{x}) \neq \beta(\vec{x})$, and $\alpha(\vec{x}) = \alpha(\vec{y}) \Rightarrow \vec{x} = \vec{y}$; and*
- *$\Delta_\alpha > 0$ specifies the* preparation time *of action $\alpha(\vec{x})$ measured in time steps.*

In our model, *exogenous* events are treated as actions not necessarily performed by agents in the system. Therefore, in the rest of the paper events and actions are used interchangeably. We model an *exogenous* event as an action recurring within certain period of time.

DEFINITION 2. ***(Exogenous Action)*** *Given the set $\hat{\mathcal{X}}$ of all exogenous actions in the system, an* ***exogenous action*** *$\hat{\alpha} \in \hat{\mathcal{X}}$ is represented as a tuple $\hat{\alpha} = \langle \alpha^{exog}(\vec{x}), \varepsilon \rangle$, where:*

- *$\alpha^{exog}(\vec{x})$ is a predicate formula s.t. $\alpha^{exog}(\vec{x}) \neq \beta^{exog}(\vec{x})$, and $\alpha^{exog}(\vec{x}) = \alpha^{exog}(\vec{y}) \Rightarrow \vec{x} = \vec{y}$; and*
- *$\varepsilon \geq 0$ specifies the maximum period within which the event $\alpha^{exog}(\vec{x})$ will occur once.*

2.2 Social Commitment

This section briefly presents a formal model of social commitment (henceforth abbreviated s-commitment). Concretely, commitments have proven useful to represent all the agent inter-dependencies: social norms, roles, authority relations

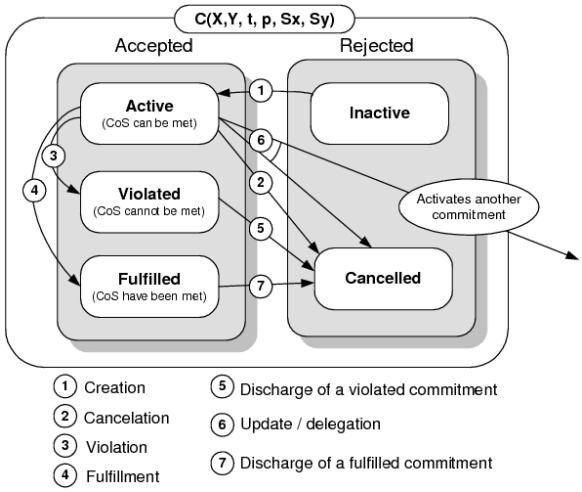


Figure 1: Social commitment finite state transition machine.

and the semantics of agent communication [4, 20]. Conceptually, commitments are oriented responsibilities contracted by a *debtor* towards a *creditor*.[1] One can distinguish *action commitments* from *propositional commitments* [24]. Propositional commitments entail complications and for that reason, following a number of other researchers [4, 8, 7], we will only consider action commitments in the rest of this paper. That is, commitments where a *debtor* is committed towards a *creditor* to bring about the effects of some *atomic action*. We adopt the model of Pasquier et al. [16] in which the dynamics of social commitments is formalized as a finite state machine (FSM). Figure 1 illustrates the different ways s-commitments can be manipulated. Note that update and delegation will not be considered in the rest of this paper.

Definition 3. ***(Action Commitment Schema)*** *Given the non-empty set $\mathcal{SC}$ of action commitments in the system, a particular* ***action commitment schema*** *$\bar{c} \in \mathcal{SC}$ is represented as a* unique *rule of the form:*

$$\bar{c} = \phi \Rightarrow C(x, y, \alpha, t_d, \mathcal{S}_x, \mathcal{S}_y) \quad (1)$$

where:

- *The antecedent ϕ is a formula representing any general trigger condition, i.e. a primitive action, an exogenous action, or any other complex condition*[2]*;*
- *The consequent $C(x, y, \alpha, t_d, \mathcal{S}_x, \mathcal{S}_y)$ is a predicate formula with an arity of 6; representing the fact that the* debtor *enacting role x is committed towards the* creditor *enacting role y to achieve the effects of action α within $t_d > 0$ time steps of the creation time, under the sanctions sets $\mathcal{S}_x$ and $\mathcal{S}_y$, which specify the different sanctions that will be applied to x and y according to the states and transitions applicable to this commitment; and*
- *$\alpha = \langle \alpha(\vec{x}), \Delta_\alpha \rangle$, with $t_d \geq \Delta_\alpha$.*

[1]Social commitments share a great deal with the notion of directed obligation as defined in deontic logic and as also used by some researchers in the context of agent communication.

[2]In this paper we restrict the antecedent formula ϕ to be either a primitive or exogenous event.

Note that, *action commitment schema* (1) can only be valid, if its total duration time t_d is at least as long as the *preparation time* (i.e. Δ_α) of the *atomic action* α. In this paper, we only consider action commitments involving atomic actions. We can look at *action commitment schemes* as *abstract* place holders describing generic oriented responsibilities contracted by a *debtor* towards a *creditor*. Social commitment *schemes* are ultimately instantiated by agents.

Definition 4. ***(Instantiated Action Commitment)*** *Given an* action commitment schema

$$\bar{c} = \phi \Rightarrow C(x, y, \alpha, t_d, \mathcal{S}_x, \mathcal{S}_y) \quad (2)$$

where the trigger condition ϕ is satisfied, we define an ***instantiation*** *of $\bar{c}$ as a* unique grounded *predicate formula with an arity of 7:*

$$c = C(x, y, \alpha, t_s, t_f, \mathcal{S}_x, \mathcal{S}_y) \quad (3)$$

where x and y are debtor and creditor agents, respectively. Formula (3) *results from:*

- *Removing the (satisfied) antecedent ϕ from formula* (2)*;*
- *Removing parameter t_d from the consequent of* (2)*, then adding parameters t_s, t_f where: t_s represents the creation time when ϕ occurs and schema $\bar{c}$ gets* instantiated*; and $t_f = t_s + t_d$;*
- *Instantiating every free variable from formula* (3)*.*

Note that both *schemes* and *instantiated action commitments* must be distinctly identified in the system. *Accepted* action commitments take the form of a grounded predicate formula: $C(x, y, \alpha, t_s, t_f, \mathcal{S}_x, \mathcal{S}_y)$. *Rejected* commitments, meaning that *debtor* x is not committed towards *creditor* y to achieve action α, take the form $\neg C(x, y, \alpha, t_s, t_f, \mathcal{S}_x, \mathcal{S}_y)$.

Our model also accounts for *ordering constraints* between *instantiated* social commitments.

Definition 5. ***(Time Constraint)*** *Given two* distinct instances *of social commitments*

$$\begin{aligned} c_i &= C(x_i, y_i, \alpha, t_{s_i}, t_{f_i}, \mathcal{S}_{x_i}, \mathcal{S}_{y_i}) \\ c_j &= C(x_j, y_j, \alpha, t_{s_j}, t_{f_j}, \mathcal{S}_{x_j}, \mathcal{S}_{y_j}) \end{aligned} \quad (4)$$

where: $t_{f_i} < t_{s_j}$ (resp. $t_{f_j} < t_{s_i}$); we use the ***time (ordering) constraint*** *notation $c_i \prec c_j$ (resp. $c_j \prec c_i$) to represent* disjoint *time intervals between $\{c_i, c_j\}$, where c_i (resp. c_j) temporally precedes c_j (resp. c_i). Otherwise, we use the notation $c_i \preccurlyeq c_j$ (resp. $c_j \preccurlyeq c_i$) to represent a time interval* overlapping constraint *between $\{c_i, c_j\}$.*

Active social commitments raise action expectations, and the enforcement of social commitments can take place through various *social control* mechanisms instead of through assumptions of sincerity and cooperativeness [16]. Social commitments, when modelled with their enforcement mechanism [16], are not necessarily sincere and do not require the agents to be cooperative. From this perspective, social commitments serve to coordinate the agents whether or not they are cooperative and whether or not they are sincere.

2.3 Social entities

In this paper, we only consider three types of social entities: agents, social roles, and social organizations. While numerous refinements are possible, we take a minimalist approach to define these entities. Formally:

Definition 6. ***(Social Entity)*** *Given the non-empty set $\mathbb{D} = Ag \cup Role \cup Org$, a particular* ***social entity*** *d is represented as $d \in \mathbb{D}$ where:*

- *Ag, Role, Org are sets that stand for all the agents, social roles, and organizations respectively;*
- *And* $Ag \cap Role = \emptyset$, $Ag \cap Org = \emptyset$, $Role \cap Org = \emptyset$.

DEFINITION 7. ***(Organization)*** *Given the set Org of all organizations in the system, a particular* ***organization*** $o \in Org$ *is represented as a tuple* $o = \langle A_o, R_o, \rho_o \rangle$ *where:*

- A_o *is the set of agents that belongs to the organization, with* $A_o \neq \emptyset$, $A_o \subseteq Ag$;
- R_o *is the set of roles relevant to the organization, with* $R_o \neq \emptyset$, $R_o \subseteq Role$; *and*
- ρ_o *is a binary relation that assigns to each agent that belongs to the organization, one or several roles from* R_o, *noted* $\rho_o : A_o \longrightarrow R_o^n$ $(1 \leq n \leq |R_o|)$, *s.t.* $\forall ag_i \in A_o$ $\rho_o(ag_i) \neq \emptyset$.

DEFINITION 8. ***(Social Role)*** *Given the set Role of all social roles and the set* $\mathcal{X}$ *of all primitive actions in the system, a particular* ***social role*** $r \in Role$ *is represented as a tuple* $r = \langle \mathcal{X}_r, \mathcal{SC}_r \rangle$ *where:*

- $\mathcal{X}_r$ *is the set of* primitive actions *that define the capabilities of this role, with* $\mathcal{X}_r \neq \emptyset$, $\mathcal{X}_r \subseteq \mathcal{X}$; *and*
- $\mathcal{SC}_r$ *is the set of* s-commitment schemes *specifying the inter-dependencies between this role and every other* debtor *or* creditor, *with* $\mathcal{SC}_r \neq \emptyset$.

DEFINITION 9. ***(Agent)*** *Given the set Ag of all agents in the system, a particular* ***agent*** $ag \in Ag$ *is represented as a tuple* $ag = \langle R_{ag}, \varkappa_{ag} \rangle$ *where:*

- R_{ag} *is the set of roles the agent is assigned to, with* $R_{ag} \neq \emptyset$, $R_{ag} \subseteq Role$; *and*
- $\varkappa_{ag}$ *is a binary relation that assigns a probabilistic* reliability *value to each primitive action* $\alpha_i \in \mathcal{X}_{ag}$ *within the capabilities of agent ag, capturing the probability of agent ag succeeding at performing primitive action* α_i, *noted* $\varkappa_{ag} : \mathcal{X}_{ag} \longrightarrow [0, 1]$, *with* $\mathcal{X}_{ag} = \bigcup \{\mathcal{X}_{r_j} \mid \langle \mathcal{X}_{r_j}, \mathcal{SC}_{r_j} \rangle \in R_{ag}\}$.

Organizations and roles are abstract constructs enacted by actual agents. When representing *instantiated commitments* we use a notation inspired by Carabelea and Boissier [2] to capture the role being enacted by the creditor and debtor agents respectively. So, we can now rewrite Formula (3) as follows:

$$c = C(ag_i : r_x, ag_j : r_y, \alpha, t_s, t_f, \mathcal{S}_{ag_i}, \mathcal{S}_{ag_j}) \tag{5}$$

meaning that agent ag_i enacts role r_x and agent ag_j enacts role r_y.

The *capabilities* of an agent are determined by all the primitive actions which define the capabilities of each *role* the agent is assigned to. For example, besides being a *cook* within organization Ω, agent ag_1 could also play the role of a *volunteer firefighter* within a different organization. In such a case, the individual capabilities of the agent ag_1 will clearly span beyond those determined by the scope of his/her role within organization Ω.

There might be instances where the same agent plays several roles within an organization. There might be other instances where several agents play the same role within an organization. In the latter case, we follow a *fair allocation principle* so that (on average) all agents have a similar chance to enact the same role they were assigned to. In our implementation of the model, the Agent Allocation Manager (AAM) module handles the system-wide allocation of agents. It is actually implemented as a wrapper to the *Mersenne Twister* (MT19937 implementation) pseudo-random number generator, which provides fast generation of high-quality pseudo-random numbers. For each role r_i the AAM keeps track of which agents are available (resp. unavailable). When instantiating a s-commitment, the AAM will randomly pick an agent from the *pool* of available agents enacting role r_i until all agents have been allocated a s-commitment and the *pool* is empty. Then, the AAM 'replenishes' the *pool* by flagging all agents enacting role r_i as available and repeats the same process again.

3. SOCIAL CONTROL MECHANISMS

Theories of *social control* [13, 9] focus on the strategies and techniques that help to regulate agent behaviour, and lead to conformity and compliance with the rules of society (at both the macro and the micro level). In the remainder of this section, we detail the main elements used in the enforcement of social commitments: *sanctions*, which are considered in their general sense of positive or negative incentives.

Most s-commitment-based approaches assume that the agents will respect their social commitments (thus applying regimentation). This assumption is unrealistic since unintended commitment violation is likely to occur and unilateral commitment cancellation as well as commitment modification are desirable. Intuitively, sanctions should meet the following base criteria. Violation and *cancellation* are either associated with (possibly) *negative* sanctions, *fulfilment* is associated with a (possibly) *positive* sanction and *violation* carries either a harsher or similar sanction than *cancellation*.

In previous work [16], we have proposed an ontology of sanction types and punishment policies. Here we will only present the basic mechanism by which the enforcement of s-commitment is ensured in our model of *social coherence*. A *sanction policy* determines the type of sanctions (and their magnitude) that are assigned to social commitments at creation time. For simplicity, we assume that sanctions are not delayed through time and are applied at the time of occurrence as specified in the sanction policy.

DEFINITION 10. ***(Sanction Policy)*** *Given an organization* $o = \langle A_o, R_o, \rho_o \rangle$; *the set* $\mathcal{SC}_o$ *of all social commitment* schemes *related to the organization; and the set* $\mathbb{T}$ *of all the transitions applicable to s-commitments. For every* schema $\bar{c} \in \mathcal{SC}_o$ *of the form*

$$\bar{c} = C(r_x, r_y, \alpha, t_s, t_f, \mathcal{S}_x, \mathcal{S}_y)$$

we specify the sanction sets $\mathcal{S}_x = \{s_x^f, s_x^c, s_x^v\}$, *and* $\mathcal{S}_y = \{s_y^c\}$ *using the following function (where z is the transition consumed in the FSM from Figure 1):*

$$\sigma_{sc}(z) = \begin{cases} s_x^f & \text{if } z = 7, \text{ // discharge of fulfilment} \\ s_{\not x}^{\not v} & \text{if } z = 5, \text{ // discharge of violation,} \\ s_{\not x}^{\not c} & \text{if } z = 2, \text{ // cancellation by debtor,} \\ s_{\not y}^{\not c} & \text{if } z = 2, \text{ // cancellation by creditor} \\ nil & \text{if } z \notin \{2, 5, 7\} \end{cases} \tag{6}$$

where:

- $\sigma_{sc} : \mathbb{T} \longrightarrow [-1, 1]$;
- s_x^f *represents the sanction value applied to* debtor x *when* fulfilling *commitment* c;
- s_x^v *represents the sanction value applied to* debtor x *when* violating *commitment* c;
- s_x^c *represents the sanction value applied to* debtor x *when* cancelling *commitment* c; *and*
- s_y^c *represents the sanction value applied to* creditor y *when* cancelling *commitment* c.

4. SOCIAL COHERENCE

In cognitive sciences and social psychology most cognitive theories appeal to the *coherence principle* which puts coherence as the main organizing mechanism: *the individual is more satisfied with coherence than with incoherence.* In this section, we build on and extend previous work on *cognitive coherence* [15, 16] by showing how to use the *coherence principle* as the driving force that makes *social behaviour* emerge from the local *cognitive coherence* of interacting agents.

4.1 Formal characterization of social coherence

We present a constraint satisfaction based model of social coherence resulting in a symbolic-connexionist hybrid formalism. In our approach, the cognitions of a social entity are represented through the notion of elements (i.e. instantiated s-commitments). We denote $\mathbb{E}$ the set of all elements. *Elements* are divided in two sets: the set $\mathcal{A}$ of *accepted elements* and the set $\mathcal{R}$ of *rejected elements*. We adopt a closed-world assumption which states that *every non-explicitly accepted element is rejected*. Since not all s-commitments are equally modifiable, a *resistance to change* is associated to each element. Formally:

DEFINITION 11. ***(Resistance to Change)*** *We specify the* ***resistance to change*** *of an element (i.e. instantiated s-commitment) through the function:*

$$Res : \mathbb{E} \times \mathbb{T} \longrightarrow \mathbb{R} \equiv -\sigma_{sc}(z) \tag{7}$$

where $\mathbb{E}$ is the set of all elements, $\mathbb{T}$ is the set of all the transitions applicable to social commitments, and $\sigma_{sc}(z)$ $(z \in \mathbb{T})$ is the sanction policy*.*

Note that, we equate the *resistance to change* with the sanctions corresponding to the transitions (i.e. *fulfilment*, *cancellation*, *violation*) of the s-commitment as specified in the *sanction policy* (Formula (6)). The higher the punishment (resp. reward) for cancelling/violating (resp. fulfilling) a s-commitment, the higher (resp. lower) the *resistance to change* will be.

S-commitments can be related or unrelated. When they are related, positive compatibility relations like facilitation and entailment are represented as *positive constraints*. Negative incompatibility relations like mutual exclusion (e.g. critical time overlap), hindering, and disabling are represented as *negative constraints*. We use $\mathcal{C}^+$ (resp. $\mathcal{C}^-$) to denote the set of positive (resp. negative) constraints and $\mathbb{C} = \mathcal{C}^+ \cup \mathcal{C}^-$ to refer to the set of all constraints. For each of these constraints, a weight reflecting the importance degree for the underlying relation is attributed (our constraint generation mechanism is described in Section 4.2). Those weights can be accessed through the function $Weight : \mathbb{C} \longrightarrow \mathbb{R}$. Constraints can be satisfied or not.

DEFINITION 12. ***(Constraint Satisfaction)*** *A positive constraint is satisfied if and only if the two elements that it binds are both accepted or both rejected, noted $Sat^+(x,y) \equiv (x,y) \in \mathcal{C}^+ \wedge [(x \in \mathcal{A} \wedge y \in \mathcal{A}) \vee (x \in \mathcal{R} \wedge y \in \mathcal{R})]$. On the contrary, a negative constraint is satisfied if and only if one of the two elements that it binds is accepted and the other one rejected, noted $Sat^-(x,y) \equiv (x,y) \in \mathcal{C}^- \wedge [(x \in \mathcal{A} \wedge y \in \mathcal{R}) \vee (x \in \mathcal{R} \wedge y \in \mathcal{A})]$. Satisfied constraints within a set of elements $\mathcal{E}$ are accessed through the function:*

$$Sat : \mathcal{E} \subseteq \mathbb{E} \longrightarrow \left\{ \begin{array}{l} (x,y) \mid x,y \in \mathcal{E} \wedge \\ (Sat^+(x,y) \vee Sat^-(x,y)) \end{array} \right\} \tag{8}$$

In that context, two elements are said to be *coherent* (resp. *incoherent*) if and only if they are connected by a relation to which a satisfied (resp. non-satisfied) constraint corresponds. The main interest of this type of modelling is to allow defining a metric of cognitive coherence that permits the reification of the coherence principle in a computational calculus.

Given a partition of elements among $\mathcal{A}$ and $\mathcal{R}$, one can measure the *coherence degree* of a non-empty set of elements $\mathcal{E}$. We use $Con()$ to denote the function that gives the constraints associated with a set of elements $\mathcal{E}$. $Con : \mathcal{E} \subseteq \mathbb{E} \longrightarrow \{(x,y) \mid x,y \in \mathcal{E}, (x,y) \in \mathbb{C}\}$.

DEFINITION 13. ***(Coherence Degree)*** *The* ***coherence degree*** *$\mathcal{C}(\mathcal{E})$, of a non-empty set of elements, $\mathcal{E}$ is obtained by adding the weights of constraints linking elements of $\mathcal{E}$ which are satisfied divided by the total weight of concerned constraints. Formally:*

$$\mathcal{C}(\mathcal{E}) = \frac{\sum_{(x,y) \in Sat(\mathcal{E})} Weight(x,y)}{\sum_{(x,y) \in Con(\mathcal{E})} Weight(x,y)} \tag{9}$$

Note that $\mathcal{C}(\mathcal{E}) \in [0,1]$ since $Sat(\mathcal{E}) \subseteq Con(\mathcal{E})$. The general social coherence problem is then:

DEFINITION 14. ***(Coherence Problem)*** *The general* ***coherence problem*** *is to find a partition of the set of elements $\mathcal{E} \subseteq \mathbb{E}$ (i.e. instantiated s-commitments) into the set of accepted elements $\mathcal{A}$ and the set of rejected elements $\mathcal{R}$, such that, it maximizes the coherence degree $\mathcal{C}(\mathcal{E})$ of the set of elements $\mathcal{E}$.*

The coherence problem is a constraint optimization problem shown to be NP-complete [22]. The state of a social entity can be defined as follows:

DEFINITION 15. ***(Social Entity's State)*** *A* ***social entity's state*** *is characterized by a tuple $W = \langle \mathcal{SC}, \mathcal{C}^+, \mathcal{C}^-, \mathcal{A}, \mathcal{R} \rangle$, where:*

- *$\mathcal{SC}$ is a set of elements that stand for the social entity's agenda, that stores all the social commitments from which the social entity is either the debtor or the creditor;*
- *$\mathcal{C}^+$ (resp. $\mathcal{C}^-$) is a set of non-ordered positive (resp. negative) binary constraints over $\mathcal{SC}$ such that $\forall (x,y) \in \mathcal{C}^+ \cup \mathcal{C}^-$, $x \neq y$;*
- *$\mathcal{A}$ is the set of accepted elements and $\mathcal{R}$ the set of rejected elements and $\mathcal{A} \cap \mathcal{R} = \emptyset$ and $\mathcal{A} \cup \mathcal{R} = \mathcal{SC}$.*

Finally, the overall degree of *social coherence* of an organization can be formally defined as follows:

DEFINITION 16. ***(Organization's Social Coherence)*** *The degree of* ***social coherence*** *of an organization o is calculated over the set of elements $\mathcal{E}_{int} \cup \mathcal{E}_{ext} \subseteq \mathbb{E}$, where:*

- *$\mathcal{E}_{int}$ is the set of instantiated s-commitments where both the* debtor *and the* creditor *are members of organization o; and*
- *$\mathcal{E}_{ext}$ is the set of instantiated s-commitments where either the* debtor *or the* creditor *(XOR) is member of organization o.*

4.2 Constraints generation

Our social coherence model does provide a systematic mechanism for generating the constraints between social commitments. Our approach draws from TÆMS [11], a domain-independent framework for environment centred analysis and design of coordination mechanisms. This very well studied

Table 1: Weights and precedence order between *hard* and *soft* constraints.

Hard Constraints		Soft Constraints	
Disabling	$w = 3$	Hindering	$w = 1$
Overlapping	$w = 2.5$	Facilitating	$w = 1$
Enabling	$w = 2$		

framework, provides a comprehensive taxonomy of elements (i.e. tasks, methods, resources) and their interrelationships for modelling open MAS. We adapted their taxonomy of constraints between tasks and constraint precedence to generate constraints between action commitments, as follows:

1. *Disabling.* Given two distinct instances of social commitments c_i, c_j, involving primitive actions α_i, α_j respectively, such that (i) there is a strict *ordering constraint* $c_i \prec c_j$ (resp. $c_j \prec c_i$); and (ii) the execution of α_i (resp. α_j) *disables* α_j (resp. α_i); we say there is a negative constraint $c_{ij}^- \in \mathcal{C}^-$ between c_i and c_j.
2. *Overlapping (duration).* Given two distinct instances of s-commitments c_i, c_j involving the same *debtor*, such that $c_i \preccurlyeq c_j$; we say there is a negative constraint $c_{ij}^- \in \mathcal{C}^-$ between c_i and c_j.[3]
3. *Enabling.* Given two distinct instances of social commitments c_i, c_j, involving primitive actions α_i, α_j respectively, such that (i) there is a strict *ordering constraint* $c_i \prec c_j$ (resp. $c_j \prec c_i$); and (ii) the execution of α_i (resp. α_j) *enables* α_j (resp. α_i); we say there is a positive constraint $c_{ij}^+ \in \mathcal{C}^+$ between c_i and c_j.
4. *Hindering.* Given two distinct instances of social commitments c_i, c_j, involving primitive actions α_i, α_j respectively, such that (i) there is a strict *ordering constraint* $c_i \prec c_j$ (resp. $c_j \prec c_i$); and (ii) the execution of α_i (resp. α_j) somewhat *diminishes* the way (e.g. cost, duration) α_j (resp. α_i) can get executed; we say there is a negative constraint $c_{ij}^- \in \mathcal{C}^-$ between c_i and c_j.
5. *Facilitating.* Given two distinct instances of social commitments c_i, c_j, involving primitive actions α_i, α_j respectively, such that (i) there is a strict *ordering constraint* $c_i \prec c_j$ (resp. $c_j \prec c_i$); and (ii) the execution of α_i (resp. α_j) somewhat *improves* the way (e.g. cost, duration) α_j (resp. α_i) can get executed; we say there is a positive constraint $c_{ij}^+ \in \mathcal{C}^+$ between c_i and c_j.

We assign weights to *hard* (i.e. *disabling*, *overlapping*, *enabling*) and *soft* (i.e. *facilitating*, *hindering*) constraints to capture the degree of importance of underlying relations between social commitments (Table 1). The constraints are generated automatically at instantiation time based on the constraints between actions (See Example 1, Formula 13). As can be expected *hard* constraints always have a higher precedence than *soft* ones. Note that, *hard* constraints have a strict ordering while *soft* constraints have the same precedence.

4.3 Local search algorithm

Decision theories as well as micro-economical theories define utility as a property of some valuation functions over some states of interest (e.g. consumption bundles, outcome of actions, state of the world). A function is a *utility function* if and only if it reflects the agent's preferences over these states. In our model, according to the afore-mentioned *coherence principle*, *social coherence* is preferred to *incoherence* which allows us to define the following expected utility function:

Algorithm 1 Recursive Local Search Algorithm

Function LocalSearch(W)

Require: $W = \langle \mathcal{SC}, \mathcal{C}^+, \mathcal{C}^-, \mathcal{A}, \mathcal{R} \rangle$; // current agent state
Ensure: List: $Change$; // ordered list of elements to change
  Local:
  Float: $G, Gval, C, Cval$; // expected utility value of best move
  Set: $\mathcal{A}', \mathcal{R}'$;
  Element: y, x;
  State: J; // agent state buffer
1: **for all** $x \in \mathcal{SC}$ **do**
2: **if** $x \in \mathcal{A}$ **then**
3: $\mathcal{A}' \leftarrow \mathcal{A} - \{x\}$; $\mathcal{R}' \leftarrow \mathcal{R} \cup \{x\}$;
4: **end if**
5: $W' \leftarrow \langle \mathcal{SC}, \mathcal{C}^+, \mathcal{C}^-, \mathcal{A}', \mathcal{R}' \rangle$;
  // expected utility of flipping x with transition z
6: $G \leftarrow C(W') - C(W) - Res(x, z)$;
7: $C \leftarrow C(W') - C(W)$; // pure coherence gain
8: **if** $G > Gval$ **then**
9: $J \leftarrow W'$; $y \leftarrow x$; $Gval \leftarrow G$; $Cval \leftarrow C$;
10: **end if**
11: **end for**
12: **if** ($Cval < 0$ and $Gval < 0$) **then**
13: **return** $Change$; // stop when coherence is not raising anymore and the expected utility is not positive
14: **else**
15: $Dialogue(y)$;
16: Update ($Res(y)$); Add (J, $Change$);
17: LocalSearch(J); // recursive call
18: **end if**

DEFINITION 17. ***(Expected Utility Function)*** *The **expected utility** for an agent to attempt to reach the state W'[4] from the state W (which only differs by the change of state of **one** s-commitment X through the consumption of transition Z) is expressed as the difference between the incoherence before and after this change minus the cost of the change (expressed in term of the resistance to change of the modified s-commitment for the given transition, that is in term of sanctions). Formally:*

$$G(W') = C(W') - C(W) - \sum_{X \in \mathbb{E}, Z \in \mathbb{T}} Res(X, Z) \qquad (10)$$

Note that, our expected utility function does not include any probabilities. This reflects the case of equi-probability in which the agent has no information about the probabilities that an actual change of the social commitment will occur. For now, agents do not take into account any uncertainty measures into their coherence calculus. For example, they do not have knowledge of their own *reliability*, nor about others'. Since *sanction policies* provide the *social control* mechanisms to regulate the enforcement of social commitments; Formula (10) explicitly integrates *social control* into the coherence calculus.

At each step of his reasoning, an agent will search for a cognition acceptance state change which maximizes this expected utility. That is, the agent will attempt to change an instantiated social commitment that maximizes the utility value through dialogue. A recursive version of the local search algorithm the agents use to maximize their *social coherence* is presented in Algorithm 1. While this is an approximation algorithm for solving the *coherence problem* (Def. 14), it behaved optimally on tested examples. Since it does not make any backtracking, the worst-case complexity of this algorithm is polynomial: $\mathcal{O}(mn^2)$, where n is the

[3]Note that, we make the assumption that for any agent, two instantiated s-commitments whose time intervals overlap are negatively constrained (i.e., agents do not multi-task).

[4]See Definition 15.

number of elements considered and m the number of constraints that bind them.[5]

Note that, we have no need to encode agents' behaviour as it automatically emerges from the coherence calculus. Although the model provides a computational metric for measuring organizational coherence (Def. 16), the overall behaviour of the system is solely driven by the local behaviour of agents. That is, macro-level social order is a coherence-driven emergent phenomena resulting from the local *cognitive coherence* of interacting agents.

5. EXAMPLE: PIZZA DELIVERY DOMAIN

EXAMPLE 1. *Lets consider a domain involving a pizza delivery organization Ω; social roles $\{r_k$ = cook$\}$, $\{r_{dp}$ = delivery-person$\}$, $\{r_{mt}$ = maintenance- technician$\}$, and $\{r_c$ = customer$\}$; and agents $\{ag_1, ag_2, ag_3, ag_4, ag_5\}$ as follows:*

- *Primitive actions* (Def. 1)*:*

$$\mathcal{X} = \left\{ \begin{array}{l} \alpha_1 = \langle \textbf{order-pizza}(ag_i : r_c, pid), 1\rangle, \\ \alpha_2 = \langle \textbf{cook-pizza}(ag_i : r_k, pid), 7\rangle, \\ \alpha_3 = \langle \textbf{clean-oven}(ag_i : r_k, oid), 5\rangle, \\ \alpha_4 = \langle \textbf{pack-pizza}(ag_i : r_{dp}, pid), 2\rangle, \\ \alpha_5 = \langle \textbf{deliver-pizza}(ag_i : r_{dp}, c, pid), 20\rangle, \\ \alpha_6 = \langle \textbf{pay-order}(ag_i : r_c, ag_j : r_{dp}, price, pid), 1\rangle, \\ \alpha_7 = \langle \textbf{repair-oven}(ag_i : r_{mt}, oid), 30\rangle, \end{array} \right\} \quad (11)$$

- *Exogenous events* (Def. 2)*:*

$$\hat{\mathcal{X}} = \left\{ \begin{array}{l} \hat{\alpha}_8 = \langle \textbf{break-oven}^{exog}(oid), 200\rangle, \\ \hat{\alpha}_9 = \langle \textbf{make-oven-dirty}^{exog}(oid), 100\rangle, \\ \hat{\alpha}_{10} = \langle \textbf{become-hungry}^{exog}, 20\rangle \end{array} \right\} \quad (12)$$

- *Constraints between actions* (Section 4.2)*:*

$$\mathcal{X}_{cons} = \left\{ \begin{array}{l} \textbf{order-pizza } \textit{enables } \textbf{cook-pizza}, \\ \textbf{break-oven}^{exog} \textit{ disables } \textbf{cook-pizza}, \\ \textbf{make-oven-dirty}^{exog} \textit{ hinders } \textbf{cook-pizza}, \\ \textbf{clean-oven } \textit{disables } \textbf{cook-pizza}, \\ \textbf{repair-oven } \textit{disables } \textbf{cook-pizza}, \\ \textbf{cook-pizza } \textit{enables } \textbf{delivery-pizza}, \\ \textbf{delivery-pizza } \textit{enables } \textbf{pay-order}, \end{array} \right\} \quad (13)$$

- *Organization* (Def. 7)*:*

$$\Omega = \left\langle \begin{array}{l} \{ag_1, ag_2, ag_3, ag_4\}, \\ \{r_k, r_{dp}, r_{mt}\}, \\ \{(ag_1, r_k), (ag_2, r_{dp}), \\ \quad (ag_3, r_{dp}), (ag_4, r_{mt})\} \end{array} \right\rangle \quad (14)$$

- *Social roles* (Def. 8)*:*[6]

$$Roles = \left\{ \begin{array}{l} r_k = \langle \{\alpha_2, \alpha_3\}, \{\bar{c_1}, \bar{c_2}, \bar{c_5}, \bar{c_6}\}\rangle, \\ r_{dp} = \langle \{\alpha_4, \alpha_5\}, \{\bar{c_1}, \bar{c_2}, \bar{c_3}, \bar{c_4}, \bar{c_6}\}\rangle, \\ r_{mt} = \langle \{\alpha_7\}, \{\bar{c_4}\}\rangle, \\ r_c = \langle \{\alpha_1, \alpha_6\}, \{\bar{c_3}, \bar{c_4}\}\rangle \end{array} \right\} \quad (15)$$

- *Agents* (Def. 9)*:*

$$Ag = \left\{ \begin{array}{l} ag_1 = \langle \{r_k\}, \{(\alpha_2, 1), (\alpha_3, 1)\}\rangle, \\ ag_2 = \langle \{r_{dp}\}, \{(\alpha_4, 1), (\alpha_5, 1)\}\rangle, \\ ag_3 = \langle \{r_{dp}\}, \{(\alpha_4, 1), (\alpha_5, 1)\}\rangle, \\ ag_4 = \langle \{r_{mt}\}, \{(\alpha_7, 1)\}\rangle, \\ ag_5 = \langle \{r_c\}, \{(\alpha_1, 1), (\alpha_6, 1)\}\rangle \end{array} \right\} \quad (16)$$

- *Social commitment schemes* (Def. 3)*:*

$$\mathcal{SC} = \left\{ \begin{array}{l} \bar{c}_1 = \alpha_1 \Rightarrow C(r_k, r_{dp}, \alpha_2, 8, [0, 0, 0], [0]) \\ \bar{c}_2 = \alpha_2 \Rightarrow C(r_{dp}, r_k, \alpha_4, 3, [0, 0, 0], [0]) \\ \bar{c}_3 = \alpha_4 \Rightarrow C(r_{dp}, r_c, \alpha_5, 21, [0, 0, 0], [0]) \\ \bar{c}_4 = \alpha_5 \Rightarrow C(r_c, r_{dp}, \alpha_6, 2, [0, 0, 0], [0]) \\ \bar{c}_5 = \hat{\alpha_8} \Rightarrow C(r_{mt}, r_k, \alpha_7, 31, [0, 0, 0], [0]) \\ \bar{c}_6 = \hat{\alpha_9} \Rightarrow C(r_k, r_{mt}, \alpha_3, 6, [0, 0, 0], [0]) \end{array} \right\} \quad (17)$$

[5] n coherence calculus (sum over m constraints) for each level and a maximum of n levels to be searched.

[6] Roles $\{r_k = cook\}$, $\{r_{dp} = delivery\text{-}person\}$, and $\{r_{mt} = maintenance\text{-}technician\}$ are part of Ω, but role $\{r_c = customer\}$ is external to the organization.

This example comprises 1 *cook* agent (ag_1), 2 *delivery-person* agents (ag_2, ag_3), 1 *maintenance-technician* agent (ag_4), and 1 *customer* agent (ag_5). Note that, the social commitment schemes in Formula (17) implicitely define the following pizza delivery workflow: **order-pizza** → **cook-pizza** → **pack-pizza** → **deliver-pizza** → **pay-order**; which is initiated when exogenous event **become-hungry**^{exog} occurs, making the *customer* agent perform the action **order-pizza**.

6. INITIAL VALIDATION

A *SC-sim* simulator has been implemented as a Java applet, which provides some flexibility in terms of deployment and facilitates sharing results with the research community. To illustrate the use of the model and the simulator, we introduce a simple experiment involving two *sanctions policies*:

- ***SPol*** $\emptyset$. $\mathcal{S}_d = \{0, 0, 0\}$, and $\mathcal{S}_c = \{0\}$. Debtors receive no rewards. Both debtors and creditors have no penalties. This policy entails no social control; and

- ***SPol 1***. $\mathcal{S}_d = \{0, -1, -1\}$, and $\mathcal{S}_c = \{-1\}$. Debtors receive no rewards and high violation penalties. Both debtors and creditors have high cancellation penalties.

Experiment. We ran the experiment on the pizza delivery domain presented in Example 1. We varied the *periodicity* (Def. 2) of the exogenous event **become-hungry**^{exog} (starting from 80 time steps, down to 40, 20, 10, 5, 2, and 1 time steps). As a result, the *customer* agent starts placing orders more frequently. Note that we assume neither agents, nor actions can fail. We measured the overall *efficiency* (i.e. percentage of s-commitments fulfilled) of the system. For each parametrization, we ran 15 simulations of 750 time steps each and computed the standard sample mean. Figure 2 presents the results.

Observation 1. As expected, the *efficiency* of the organization degraded from nearly optimal as the *frequency* of orders and the corresponding *level of activity* (i.e. number of s-commitments per agent per time step, not shown here) was increased.

Observation 2. We can observe drastic differences between the evaluated policies. These two *sanction policies* had a distinct effect on the performance of the system. Under policy ***SPol 1*** the organization was more *efficient* than without having any *social control* (i.e., ***SPol*** $\emptyset$).

Observation 3. Desirable (sometimes nearly optimal) agent behaviour results from local *coherence maximization*, without explicitly encoding agents behaviour. More importantly, macro-level *social coherence* does emerge from local *coherence maximization*.

Although this paper focuses on presenting the model, we think these experimental results are encouraging as they provide some preliminary validation. Of course, there is still much work to be done in terms of running more experiments, analyzing results and evaluating the scalability of the model. Our work takes on the problem of modelling *desirable and (relatively) predictable emergent social behaviour* from the local actions of the agents [5]. Observation 3, provides some preliminary evidence to support the suitability of our model for running social simulations, where complex *emergent social patterns* can be obtained and reproduced from the dynamics of local interactions among agents. There is

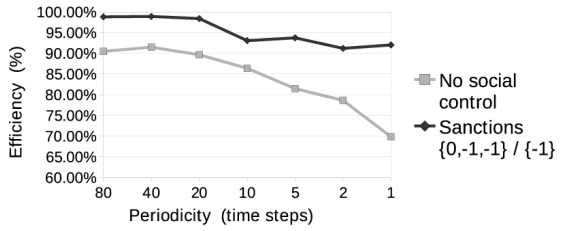


Figure 2: Experimental Results (Efficiency %).

complexity happening that cannot be fully explained analytically, thus justifying an empirical simulation-based approach. Similarly, Observation 2 provides some evidence to support the effectiveness of integrating *social control* mechanisms (for the enforcement of *s-commitments*), into the coherence calculus. Note that, when agents have neither positive, nor negative incentives their local *coherence-driven deliberation* might eventually lead them to unilaterally cancelling, or even violating *social commitments* as there are no consequences. Some authors have suggested [5] that *social cooperation* does not necessarily require an agent's understanding, agreement, nor even awareness. Our proposal aligns with this view, and Observation 2 shows that we are able to re-produce *desirable cooperation-like behaviour*, through the implementation of an appropriate *sanction policy* (e.g., ***SPol 1***).

7. DISCUSSION AND RELATED WORK

There have been several approaches [4, 20, 16, 2] to formalizing *social commitments*. The proposal of Carabelea and Boissier [2] relies on *social commitments* for coordinating agents within the context of organizational interactions. Like us, they do define *social entities* and *organizational structures* entirely based on *social commitments*. However, in our proposal all the dynamics of *social commitments* are captured by a generic state-transition model which is associated with *social control* mechanisms for the enforcement of *social commitments*. In addition, we choose not to explicitly specify *authority relations* between roles. Instead, we capture them as implicitly resulting from *social commitments schema* associated with roles. Thus, we can get a more compact representation without compromising expressiveness.

One other coherence-based framework inspired by early work on cognitive coherence in MAS [15] has been proposed by Joseph et al. [18, 19]. Their framework builds on the BDI model of agency and the coherence theory [21]. Their approach, is also based on a *coherence maximization* model of agent rationality implemented as a constraint satisfaction problem. However, their proposal substantially differs from ours as (i) their main motivation seems to be the study of the interactions between the agent's internal cognitions (BDI) and some social aspects of MAS such as: norm evaluation [19], and the behaviour of institutional agents [18]; and (ii) their approach uses coherence graphs to represent each BDI modality resulting in a more complex model that has not been validated nor implemented and thus does not allow to derive new knowledge. In contrast, we are interested in modelling and evaluating the general dynamics of social systems. We claim that our model not only is more compact and decreases the computational overhead incurred when calculating coherence, but also is expressive enough to represent complex social systems. Although, in this paper, we do not consider *social norms*, we can certainly model them by representing s-commitments from a role towards an organization.[7]

Other organizational approaches to social modelling have been reported in the literature [17, 23]. The former, is a knowledge-based approach to automated organizational design, which enables efficient role selection to match organizational goals, as well as agent-to-role allocation. Like us, they define organizational structures in terms of *agents* enacting *roles* in *organizations*. However, their focus is on designing effective organizations which can change forms depending of varying performance requirements. Instead, our *simulation framework* focuses on evaluating the emergent social dynamics and performance of multi-agent organizations from the local *coherence*-driven interactions among agents. The latter proposal [23], presents an agent-oriented language (endowed with an operational semantics) for developing multi-agent organizations. *Organizations* are defined in terms of *roles*, *norms*, and *sanctions*. Although structurally close from an abstract organizational standpoint; our models also differ as theirs specify *roles* in terms of the same mental attitudes attributed to BDI agents. Instead, we define *roles* in terms of *capabilities* which can be enacted by *agents*. Moreover, the *state* of a *role* makes no reference to mental attitudes.

8. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a simple operational model capturing the dynamics of social systems. Our work advances the state of the art by proposing a unified yet computational view of the social aspects of multi-agent organizations. In previous work [15], we proposed a constraint satisfaction-based model of *cognitive coherence* within the context of agent communication pragmatics. Here, we built on this work and extended it to consider *social coherence*. We introduced the notion of *social coherence* as the main social organizing principle in MAS. Moreover, our model relies on the notion of *social commitment* to represent all the inter-dependencies between *social entities*. Together with the notion of *sanction policy*, *social coherence* reify the notion of *social control*. In our model, *social control* is actually integrated into the coherence calculus (Def. 11). Local *coherence* is the driving force that organizes agents' behaviour and from which *social coherence* emerges. Finally, we illustrated our model and simulator by running a simple experiment to investigate the effects of two *social control* mechanisms (reified by *sanction policies*) on a sample domain.

As future work, we will refine our model using an action language such as *event calculus* [12]. We want to evaluate the benefits of introducing a more comprehensive treatment of *time*, as well as reasoning about *actions*. We also plan to address the issue of handling *complex actions*. Another immediate extension to our model will be the introduction of *uncertainty reasoning* into the *coherence calculus*. For now, agents do not take into account any uncertainty measures. Since both *actions* and *agents* can fail (as reflected by the

[7] An *institution* can be seen as a particular type of organization.

reliability probability value in Def. 9), agents should be able to incorporate these information into their expected utility calculus. Agents with different levels of knowledge should also be modelled, such as: agents with no knowledge, with partial knowledge, or with complete/shared knowledge. Furthermore, various machine learning mechanisms would allow agents to progressively learn these probabilities.

Finally, we want to run more experiments and evaluate the scalability of our model. For instance, we should model social domains with multiple organizations and greater number of agents, where agents can play several roles possibly in different organizations. We also want to investigate how our coherentist approach might be used to evaluate the functionality and behaviour of typical *organizational structures* reported in the literature (e.g., hierarchies, holarchies, societies, federations [10]). Furthermore, since no single organizational design is suitable for all domain applications we want to cross-validate our model by running simulations involving different organizational structures. Last but not least, we want to continue studying the effects of *social control* mechanisms.

9. ACKNOWLEDGMENTS

The authors would like to thank Marek Hatala and the anonymous reviewers for their useful comments.

10. REFERENCES

[1] S. Bandini, S. Manzoni, and G. Vizzari. Agent based modeling and simulation. In Meyers [14], pages 184–197.

[2] C. Carabelea and O. Boissier. Coordinanting Agents in Organizations Using Social Commitments. In *Proceedings of the 1st International Workshop on Coordination and Organisation*, Namur, April 2005.

[3] K. M. Carley and L. Gasser. *Multiagent Systems: a modern approach to distributed Artificial Intelligence*, chapter Computational Organization Theory, pages 299–330. MIT Press, 2001.

[4] C. Castelfranchi. Commitments: from Individual Intentions to Groups and Organizations. In *Proceedings of ICMAS95*, pages 41–48, June 1995.

[5] C. Castelfranchi. Engineering Social Order. In *Proceedings of the First International Workshop on Engineering Societies in the Agents World*, volume 1972, pages 1–18, 2000.

[6] P. Davidsson and H. Verhagen. Social phenomena simulation. In Meyers [14], pages 8375–8379.

[7] R. A. Flores and R. C. Kremer. To Commit or Not to Commit: Modeling Agent Conversations for Action. *Computational Intelligence*, 18(2):120–173, 2002.

[8] N. Fornara and C. Colombetti. Operational Specification of a Commitment-Based Agent Communication Language. In C. Castelfranchi and W. L. Johnson, editors, *Proceeding of the First Autonomous Agents and Multi-Agents Systems Joint Conference (AAMAS'02)*, volume 2, pages 535–543. ACM Press, 2002.

[9] M. Hechter and K. D. Opp. Introduction. *Social Norms*, pages xi–xx, 2001.

[10] B. Horling and V. Lesser. A Survey of Multi-Agent Organizational Paradigms. *The Knowledge Engineering Review*, 19(4):281–316, 2005.

[11] B. Horling, V. Lesser, R. Vincent, T. Wagner, A. Raja, S. Zhang, K. Decker, and A. Garvey. The TAEMS White Paper, 1999.

[12] R. A. Kowalski and M. Sergot. A Logic-Based Calculus of Events. *New Generation Computing*, 4:67–95, 1986.

[13] D. Martindale. *Social Control for the 1980s: A Handbook for Order in a Democratic Society*, chapter The Theory of Social Control, pages 46–58. Greenwood Press, 1978.

[14] R. A. Meyers, editor. *Encyclopedia of Complexity and Systems Science*. Springer, 2009.

[15] P. Pasquier and B. Chaib-draa. The Cognitive Coherence Approach for Agent Communication Pragmatics. In *Proceedings of The Second International Joint Conference on Autonomous Agent and Multi-Agents Sytems (AAMAS'03)*, pages 544–552, Melbourne, 2003.

[16] P. Pasquier, R. A. Flores, and B. Chaib-draa. Modelling Flexible Social Commitments and their Enforcement. In *Proceedings of the Fifth International Workshop Engineering Societies in the Agents World (ESAW'04)*, volume 3451 of *Lecture Notes in Artificial Intelligence*, pages 153–165. Springer-Verlag, 2004.

[17] M. Sims, D. Corkill, and V. Lesser. Automated Organization Design for Multi-agent Systems. *Autonomous Agents and Multi-Agent Systems*, 16(2):151–185, 2008.

[18] J. Sindhu, C. Sierra, and M. Schorlemmer. A Coherence Based Framework for Institutional Agents. In *Proceedings of the Fifth European Workshop on Multi-Agent Systems (EUMAS'07)*, December 2007.

[19] J. Sindhu, C. Sierra, M. Schorlemmer, and P. Delunde. Formalizing Deductive Coherence: An Application to Norm Evaluation. *Logic Journal of the Interest Group in Pure and Applied Logic (IGPL)*, 2008.

[20] M. P. Singh. An Ontology for Commitments in Multiagent Systems: Toward a Unification of Normative Concepts. *Artificial Intelligence and Law*, 7:97–113, 1999.

[21] P. Thagard. *Coherence in Thought and Action*. MIT Press, 2000.

[22] P. Thagard and K. Verbeurgt. Coherence as Constraint Satisfaction. *Cognitive Science*, 22:1–24, 1998.

[23] N. Tinnemeier, M. Dastani, and J.-J. Meyer. Roles and Norms for Programming Agent Organizations. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 121–128, Richland, SC, 2009.

[24] D. N. Walton and E. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. Suny Press, 1995.

Shared Mental Models: A Conceptual Analysis

Catholijn M. Jonker
EEMCS, TU Delft
Delft, The Netherlands
c.m.jonker@tudelft.nl

M. Birna van Riemsdijk
EEMCS, TU Delft
Delft, The Netherlands
m.b.vanriemsdijk@tudelft.nl

Bas Vermeulen
ForceVision
Den Helder, The Netherlands
bas.vermeulen@forcevision.nl

ABSTRACT

The notion of a shared mental model is well known in the literature regarding team work among humans. It has been used to explain team functioning. The idea is that team performance improves if team members have a shared understanding of the task that is to be performed and of the involved team work. We maintain that the notion of shared mental model is not only highly relevant in the context of human teams, but also for teams of agents and for human-agent teams. However, before we can start investigating how to engineer agents on the basis of the notion of shared mental model, we first have to get a better understanding of the notion, which is the aim of this paper. We do this by investigating which concepts are relevant for shared mental models, and modeling how they are related by means of UML. Through this, we obtain a mental model ontology. Then, we formally define the notion of shared mental model and related notions. We illustrate our definitions by means of an example.

Categories and Subject Descriptors

I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Intelligent agents*

General Terms

Design, Theory

Keywords

Shared mental models, human-agent teams

1. INTRODUCTION

The notion of a shared mental model is well known in the literature regarding team work among humans [4, 2, 16, 15]. It has been used to explain team functioning. The idea is that team performance improves if team members have a shared understanding of the task that is to be performed and of the involved team work.

We maintain that shared mental model theory as developed in social psychology, can be used as an inspiration for the development of techniques for improving team work in (human-)agent teams. In recent years, several authors have made similar observations. In particular, in [19] agents are implemented that use a shared mental model of the task to be performed and the current role assignment to proactively communicate the information other agents need. Also, [18] identify "creating shared understanding between human and agent teammates" as the biggest challenge facing developers of human-agent teams. Moreover, [14] identify common ground and mutual predictability as important for effective coordination in human-agent teamwork.

In this paper, we aim to lay the foundations for research on using shared mental model theory as inspiration for the engineering of agents capable of effective teamwork. We believe that when embarking on such an undertaking, it is important to get a better understanding of the notion of shared mental model. In this paper, we do this by investigating which concepts are relevant for shared mental models, and modeling how they are related by means of UML. Through this, we obtain a mental model ontology. Then, we formally define the notion of shared mental model using several related notions. We illustrate our definitions by means of an example.

2. EXPLORATION OF CONCEPTS

This section discusses important concepts related to the notion of shared mental models.

2.1 Working in a Team

An abundance of literature has appeared on working in teams, both in social psychology as well as in the area of multi-agent systems. It is beyond the scope of this paper to provide an overview. Rather, we discuss briefly how work on shared mental models distinguishes aspects of teamwork. Since we are interested in shared mental models, we take their perspective on teamwork for the analyses in this paper. We do not suggest that it is the only (right) way to view teamwork, but it suffices for the purpose of this paper.

An important distinction that has been made in the literature on shared mental models, is the distinction between *task work* and *team work* (see, e.g., [4, 16]). Task work concerns the task or job that the team is to perform, while team work concerns what has to be done only because the task is performed by a team instead of an individual agent. In particular, task work mental models concern the equipment (equipment functioning and likely failures) and the task (task procedures and likely contingencies). Team work mental models concern team interaction (roles and responsibilities of team members, interaction patterns, and information flow), and team members (knowledge, skills, and preferences of teammates).

2.2 Mental Models

In order to be able to interact with the world, humans must have some internal representation of the world. The notion of *mental model* has been introduced to refer to these representations. A mental model can consist of knowledge about a physical system that should be understood or controlled, such as a heat exchanger or an interactive device [8]. The knowledge can concern, e.g., the structure and overall behavior of the system, and the disturbances that act on the system and how these affect the system. Such mental

models allow humans to interact successfully with the system.

Different definitions of mental models have been proposed in the literature (see, e.g., [6] for a discussion in the context of system dynamics). In this paper, we use the following often cited, functional definition as proposed in [17]:

> Mental models are the mechanisms whereby humans are able to generate descriptions of system purpose and form, explanations of system functioning and observed system states, and predictions of future system states.

Central to this definition is that mental models concern a *system* and that they serve the purpose of *describing, explaining, and predicting the behavior of the system*. Another important view of mental models was proposed in [13]. The idea proposed there focuses on the way people reason. It is argued that when people reason, they do not use formal rules of inference but rather think about the possibilities compatible with the premises and with their general knowledge. In this paper, we use the definition of [17] because as we will show, it is closely related to the definition of shared mental model that we discuss in the next section.

2.3 Shared Mental Models

Mental models have not only been used to explain how humans interact with physical systems that they have to understand and control, but they have also been used in the context of team work [4, 16]. There the *system that mental models concern is the team*. The idea is that mental models help team members predict what their teammates are going to do and are going to need, and hence they facilitate coordinating actions between teammates. In this way, mental models help explain team functioning.

Mental models have received a lot of attention in literature regarding team performance. Several studies have shown a positive relation between team performance and similarity between mental models of team members (see, e.g., [2, 16, 15]). That is, it is important for team performance that team members have a shared understanding of the team and the task that is to be performed, i.e., that team members have a *shared mental model*.

The concept of shared mental model is defined in [4] as

> knowledge structures held by members of a team that enable them to form accurate explanations and expectations for the task, and, in turn, coordinate their actions and adapt their behavior to demands of the task and other team members.

Shared mental models thus help *describe, explain and predict the behavior of the team*, which allows team members to coordinate and adapt to changes. In [4], it is argued that shared mental model theory does not imply identical mental models, but "rather, the crucial implication of shared mental model theory is that team members hold compatible mental models that lead to common expectations for the task and team."

In correspondence with the various aspects of teamwork as discussed above, it has been argued that multiple different types of shared mental models are relevant for team performance: shared mental models for task work (equipment model and task model) and for team work (team interaction model and team member model) [4, 16].

In this paper, we are interested in the notion of shared mental model both in humans and in software agents, but at this general level of analysis we do not distinguish between the two. Therefore, from now on we use the term "agent" to refer to either a human or a software agent.

3. MENTAL MODEL ONTOLOGY

We start our analysis of the notion of shared mental model by analyzing the notion of mental model. We do this by investigating the relations between notions that are essential for defining this concept, and provide UML[1] models describing these relations. The UML models thus form a mental model ontology. We have devided the ontology in three figures for reasons of space and clarity of presentation. We have not duplicated all relations between in all diagrams to reduce the complexity of the diagrams. For example, in figure 2 there is no relation between Agent and Team. This relation is modelled in figure 3.

We use UML rather than formal ontology languages such as description logics [1], since it suffices for our purpose. We develop the ontology not for doing sophisticated reasoning or as a design for a multi-agent system, but rather to get a better understanding of the essential concepts that are involved. Also, the developed ontologies are relatively manageable and do not rely on involved concept definitions.

We present the UML models in three steps. First, since the concept of a mental model refers to systems, we discuss the notion of *system*. Then, since shared mental models are important in the context of teams, we show how a *team* can be defined *as a system*. Following that, we introduce the notion of agent into the picture and show how the notions of agent, system, and mental model are related.

In UML classes (concepts) are denoted as rectangles. A number of relations can be defined between concepts. The generalization relation is a relation between two concepts that is denoted like an arrow. This relation represents a relationship between a general class and a more specific class. Every instance of the specific class is also an instance of the general class and inherits all features of the general class. A relationship from a class A to class B with an open diamond at side one of the ends is called a shared aggregate, defined here as a part-whole relation. The end of the association with the diamond is the whole, the other side is the part. Because of the nature of this relationship it cannot be used to form a cycle. A composite aggregation is drawn as an association with a black diamond. The difference with a shared aggregation is that in a composite aggregation, the whole is also responsible for the existence, persistence and destruction of the parts. This means that a part in a composite aggregation can be related to only one whole. Finally, a relationship between two concepts that is represented with a normal line, an association, can be defined. The nature of this relationship is written along the relationship. This can either be done by placing the name of the association in the middle of the line or by placing a role name of a related concept near the concept. The role name specifies the kind of role that the concept plays in the relation. Further, numbers can be placed at the ends of the shared aggregation, composite aggregation and associations. They indicate how many instances of the related concepts can be related in one instance of the relationship.

3.1 System

The previous section shows that the concept of a mental model refers to systems. In this section, we further analyze the notion of system in order to use it to define a team as a system. For this purpose, the basic definition provided by Wikipedia[2] suffices as a point of departure: *A system is a set of interacting or independent entities, real or abstract, forming an integrated whole*. This definition captures the basic ingredients of the notion of system found in the

[1] http://www.omg.org/spec/UML/2.2/

[2] http://en.wikipedia.org/wiki/System

literature (see, e.g., [7]), namely static structures within the system as well as the dynamic interrelations between parts of the system.

Our conceptualization of systems is supported by the UML diagram in Figure 1.

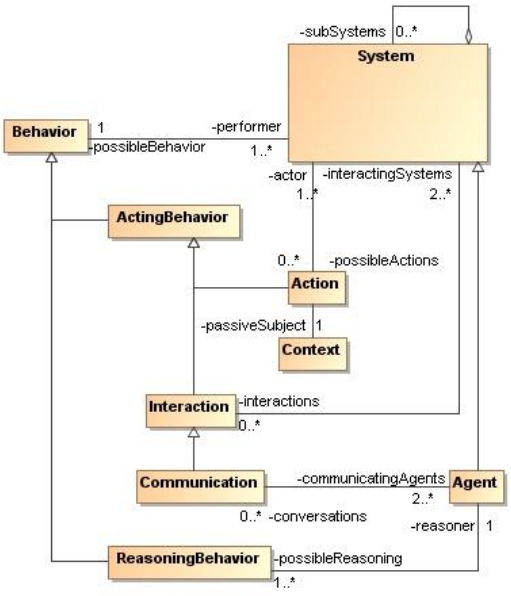


Figure 1: System

The upper-right corner of the diagram depicts that a system may be a composite, i.e., it may be composed of other systems. This modeling choice makes it easier to define in the following section the notion of team as a system. In particular, the compositionality of the concept system in terms of other systems makes the compositionality of mental models straightforward in the next sections. Regarding the definition, this part addresses the sub-phrase that a system is a set of entities.

The system forms an integrated whole, according to the definition. Therefore, the whole shows behavior. As we do not distinguish between natural or designed systems, living or otherwise, we chose behavior to represent the dynamics of the system as a whole. Note that we further distinguish between reasoning behavior and acting behavior. Not all systems will show both forms of behavior. Acting behavior refers to either actions or interactions. An action is a process that affects the environment of the system and/or the composition of the system itself. Interaction is a process with which a sub-system of the system (or the system as a whole) affects another sub-system of the system. Communication is a special form of interaction, in which the effect of the interaction concerns the information state of the other element. Communication is a term we restricted for the information-based interaction between two agents. The term reasoning behavior is also reserved for agents. The concept "context" refers to both the environment of the system as well as the dynamics of the situation the system is in. The system executes its actions in its context.

3.2 Team as a System

The notion of system is central to the definition of mental model. In the context of shared mental models we are especially interested in a certain kind of system, namely a team. According to the definition of system, a team can be viewed as a system: it consists of a set of interacting team members, forming an integrated whole.

As noted above, several aspects are relevant for working in a team. We take as a basis for our model the distinction made in [4, 16]. As noted in Section 2.1, we by no means claim that this is the only suitable definition of a team or that it captures all aspects. We start from this research since it discusses teams in the context of shared mental models. The most important realization for the sequel is that we define a team as a system and that it has as a set of team members that are agents. Other aspects of the team definition can be varied if nessecary.

The following aspects are distinguished: *equipment* and *task* (related to task work), and *team interaction* and *team members* (related to team work). In our model, we include these four aspects of working in a team. However, we divide them not into team work and task work, but rather into *physical components* and *team activity*, where team members and equipment are physical components and task and team interaction are team activities The reason for making this distinction is that we argue that physical components can in turn be viewed as systems themselves, while team activities cannot, as reflected by the link from physical components to system in Figure 2 below. Moreover, we make another refinement and make a distinction between a task and *task execution*. We argue that task execution is a team activity, even though a task might be performed by only one team member. The task itself describes what should be executed. The concept task is also linked to equipment, to express the equipment that should be used for executing the task, and to team member, to describe which team members are responsible for a certain task.

We link this conceptualization of the notion of team to the general notion of system of Figure 1 by defining a team activity as a kind of acting behavior, and more specifically team interaction as a kind of interaction. We see team interaction as interaction induced by executing the team activity. Moreover, by defining that physical components are systems, we can deduce from Figure 1 that they can have interactions with each other. Moreover, by defining a team member as an agent, we can deduce from Figure 1 that team members can have reasoning behavior and that they can communicate.

These considerations are reflected in the UML model below.

3.3 Mental Model

Now that we have conceptualized in some detail the notion of system and of a team as a system, we are ready to zoom in on the notion of mental model.

As noted above, mental models are used by humans, i.e., humans have mental models. However, since in this paper we use the notion of agent as a generalization of human and software agent, here we consider that agents have mental models. Moreover, a mental model concerns a system. The basic structure of how mental models are related to systems and agents is thus that an agent has mental models and a mental model concerns a system.

However, we make several refinements to this basic view. First, we would like to express where a mental model resides, namely in the *mind* of an agent. As such, mental models can be contrasted with *physical models*. In order to do this, we introduce the notion of a *model*, and define that physical models and mental model are kinds of models. Both kinds of models can concern any type of system. A nice feature of this distinction is that it allows us to easily express how the notion of *extended mind* [5] is related. The notion of extended mind is being developed in research on philosophy of mind, and the idea is that some objects in the external environment of an agent, such as a diary to record a schedule of meetings or a shared display, are utilized by the mind in such a way that the

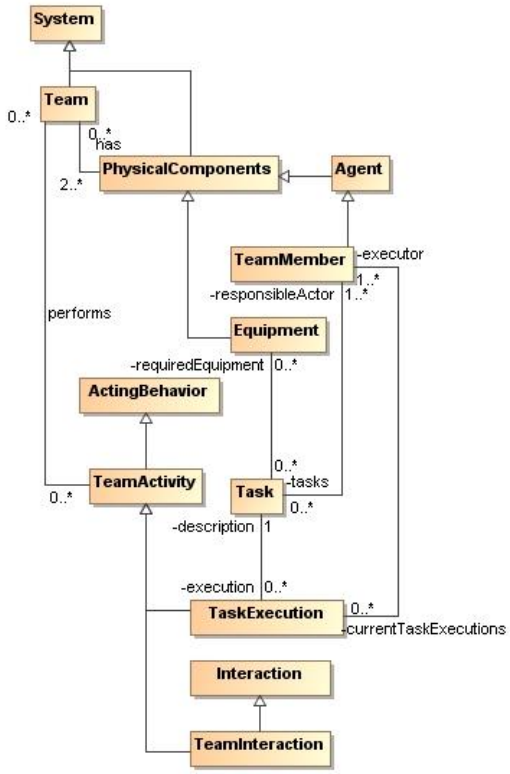


Figure 2: Team

objects can be seen as extensions of the mind itself. The notion is relevant to research on shared mental models because agents in a team may share an extended mind, and through this obtain a shared mental model [2].

Another aspect that we add to the conceptualization, is the notion of *goal* to express that a mental model is used by an agent for a certain purpose, expressed by the goal of the model.

This is captured in the UML model below.

Given this conceptualization, we can express that an agent has a mental model of a team. An agent can have a mental model, since it has a mind and a mind can have mental models. A mental model can concern a team, since a mental model is a model and a model concerns a system, and a team is a kind of system. However, since team interaction is not by itself a system (see previous subsection), our model does not allow to express, for example, that the agent has a team interaction mental model. What our conceptualization does allow to express, is that the team mental model has a part that describes team interaction, since the team mental model concerns a team, and a team has team interaction. According to our model, we thus cannot call this part a mental model. However, we will for the sake of convenience refer to that part as a team interaction model (and similarly for the other parts of a team mental model). This is in line with [4, 16], where the parts of a team mental model are called mental models themselves. We have modelled the relation between team and team member as a normal association instead of by an aggregation because modelling this relation as an aggregation would mean that an agents mind is part of a team, which does not conform to intuition.

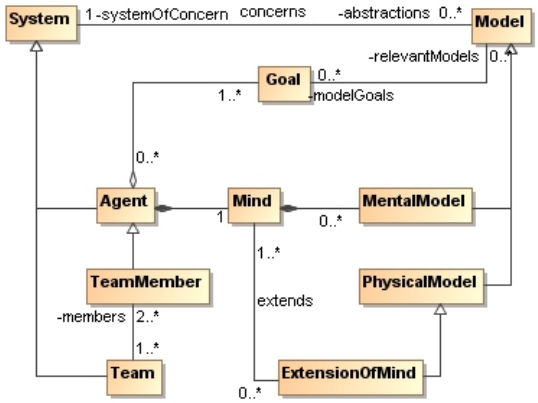


Figure 3: Mental Model

3.4 Accuracy of Models

In research on shared mental models, the relation of both *accuracy*[3] and *similarity* of mental models to team performance has been investigated [15]. As noted in [16], "similarity does not equal quality - and teammates may share a common vision of their situation yet be wrong about the circumstances that they are confronting".

We suggest that the notions of accuracy and similarity not only have different meanings, but play a different role in the conceptualization of shared mental models. That is, the notion of accuracy of a mental model can be defined by comparing the mental model against some standard or "correct" mental model, i.e., it does not (necessarily) involve comparing mental models of team members. The notion of similarity, on the other hand, *does* involve comparing mental models of team members. Although both accuracy and similarity affect team performance [15], we maintain that conceptually, only similarity is to be used for defining the notion of shared mental model. For reasons of space, we therefore discuss accuracy informally, and omit the formalizations. We discuss accuracy and similarity with respect to models in general, rather than to only mental models.

We identify two kinds of accuracy, depending on what one takes to compare the model with. The first is what we call *system accuracy*, which assumes that one has a "bird's eye view" of the system and can see all relevant aspects, including the mental models of agents in the system. In general, this is only of theoretical relevance, since one typically has limited access to the various parts of a system.[4] Another notion of accuracy that is easier to operationalize, is *expert accuracy*. In expert accuracy, the idea is to compare a model to an expert model (see e.g. [15] for an example of how to obtain an expert model). Expert accuracy may be defined as the extent to which the model agrees (see Section 4.2) with the expert model. In research on shared mental models, this is the approach taken to determine accuracy of mental models of team members [15]. That work also describes how this can be operationalized. Viewed in this way, the notion of accuracy has some resemblance to the notion of precision in the field of information retrieval [3].

[3]Here, accuracy is meant in the sense of "freedom from errors", not in the sense of precision/exactness.

[4]In a multi-agent system where one has access to the environment and internal mental states of all agents, one *would* be able to obtain all necessary information.

4. SIMILARITY OF MODELS

As we suggested in the previous section, the essence of the concept of shared mental model is the extent to which agents have *similar* mental models. The word "shared" suggests full similarity, but this is typically not the case. Rather, we propose that *measures* of similarity should be used, which allow the investigation of when models are similar enough for a good team performance, or, in general, good enough for achieving certain goals. We introduce a formal framework in order to be able to express several definitions of notions of similarity. We define sharedness in terms of those notions.

4.1 Formal Framework

The definitions of similarity are based on the concepts and their relations as discussed above. The basic concept that we use in all definitions is *model* (Figure 3). We denote a model typically as M. In this paper, we abstract from the knowledge representation language used for representing the model. Depending on the context, different languages may be chosen. For example, when investigating shared mental models in the context of cognitive agent programming languages (see, e.g., [10]), the knowledge representation language of the respective language can be used.

In order to define to what extent a model is similar to another model, we need to express the content of the model. Rather than considering the entire model, we focus on those aspects of the model that are relevant for the *goal* for which the model is to be used (Figure 3). In order to identify what the model has to say with respect to aspects relevant for the goal, we propose to use *questions* that can be posed to the model. A set of questions is typically denoted by Q. For example, a mental model that is to be used for weather predictions should be able to answer a question such as what the weather will be tomorrow in a certain city. A physical model of our solar system should be able to answer a question such as whether the Earth or Mars is closer to the sun. We write $M \vdash \mathit{answer}(a, q)$ to express that M answers a to question q. As usual, we use $\mid s \mid$ to denote the number of elements of a set s. If the model is represented using a logical knowledge representation language, $\vdash$ can be taken to be the entailment relation of the logic. If this is not the case, $\vdash$ should be interpreted more loosely.

Choosing an appropriate set of questions is critical for obtaining useful measures of similarity. For example, posing questions about the solar system to a model for weather predictions will not be useful for measuring the similarity of the weather prediction model to another such model. Moreover, posing only questions about whether it will rain to a weather prediction model, will not provide a useful measure of the weather model's similarity to another model in predicting the weather in general. A similar issue also arises in research on shared mental models in social psychology. In that work, researchers commonly assess mental models by presenting respondents with a list of concepts and asking them to describe the strength of relationships among the concepts [15, 16]. These concepts are carefully chosen based on, for example, interviews with domain experts. The operationalization of our definitions thus requires methods and techniques to determine the appropriate sets of questions Q for the team tasks, respecting the characteristics of the domain/environment in which the team has to function. The methods and techniques we consider important are those for knowledge engineering and elicitation and should take into account social theories about team building and team performance.

We propose to use questions to identify the content of models because we believe it can be applied naturally to software agents and human agents alike (see the example in the sequel). Asking agents to describe relationships among concepts is more difficult to translate to software agents, unless they are endowed with capabilities for ontological reasoning. Moreover, with some mental flexibility one can use questions both for mental as well as for physical models, as illustrated by the examples provided above.

4.2 Definitions

In the following, let M_1 and M_2 be models of systems S, and let Q be the set of questions identified as relevant for the goal for which M_1 and M_2 are to be used. Let T be a background theory used for interpreting answers. In particular, equivalence is defined with respect to T. For example, the answers "1,00 meter" and "100 centimeter" are equivalent with respect to the usual definitions of units of length.

The first definition of similarity that we provide, is what we call *subject overlap*. Subject overlap provides a measure for the extent to which models provide answers to the set of relevant questions Q. These answers may be different, but at least an answer should be given. We assume that if the answer is not known, no answer is provided. For example, posing a question about the weather in a certain city to a model of the solar system would typically not yield an answer. Also, we assume that answers are individually consistent.

DEFINITION 1 (SUBJECT OVERLAP). *Let the set of questions for which the models provide answers (not necessarily similar answers) be* $\mathit{OverAns}(M_1, M_2, Q) = \{q \in Q \mid \exists a_1, a_2 : M_1 \vdash \mathit{answer}(a_1, q)$ *and* $M_2 \vdash \mathit{answer}(a_2, q)\}$. *Then, we define the level of subject overlap between the model* M_1 *and* M_2 *with respect to set of questions* Q *as* $\mathit{SO}(M_1, M_2, Q) = \mid OverAns(M_1, M_2, Q) \mid / \mid Q \mid$.

Since the literature (see Section 2.3) says that shared mental model theory implies that team members hold compatible mental models, we define a notion of compatibility of models. It is defined as the extent to which models do not provide contradictory answers.

DEFINITION 2 (COMPATIBILITY). *Let the set of questions for which the models provide incompatible answers be* $\mathit{IncompAns}(M_1, M_2, Q) = \{q \in Q \mid \exists a_1, a_2 : M_1 \vdash \mathit{answer}(a_1, q)$ *and* $M_2 \vdash \mathit{answer}(a_2, q)$ *and* $T, a_1, a_2 \vdash \bot\}$. *Then, we define the level of compatibility between the model* M_1 *and* M_2 *with respect to set of questions* Q *as:*
$\mathit{C}(M_1, M_2, Q) = 1 - (\mid IncompAns(M_1, M_2, Q) \mid / \mid Q \mid)$.

Note that our definition of compatibility does not investigate more complex ways in which the so determined set might lead to inconsistencies. Also note that non-overlapping models are maximally compatible. This is due to the fact that we define incompatibility based on inconsistent answers. If the models do not provide answers to the same questions, they cannot contradict, and therefore they are compatible.

Next, we define *agreement* between models, which defines the extent to which models provide *equivalent* answers to questions.

DEFINITION 3 (AGREEMENT). *Let the set of questions for which the models agree be* $\mathit{AgrAns}(M_1, M_2, Q) = \{q \in Q \mid \exists a_1, a_2 : M_1 \vdash \mathit{answer}(a_1, q)$ *and* $M_2 \vdash \mathit{answer}(a_2, q)$ *and* $a_1 \equiv_T a_2\}$. *Then, we define the level of agreement between the model* M_1 *and* M_2 *with respect to set of questions* Q *as:*
$\mathit{A}(M_1, M_2, Q) = \mid AgrAns(M_1, M_2, Q) \mid / \mid Q \mid$.

These measures of similarity are related in the following way.

PROPOSITION 1 (RELATIONS BETWEEN MEASURES). *We always have that* $\mathit{A}(M_1, M_2, Q) \leq \mathit{SO}(M_1, M_2, Q)$. *Moreover, if* $\mathit{SO}(M_1, M_2, Q) = 1$, *we have* $\mathit{A}(M_1, M_2, Q) \leq \mathit{C}(M_1, M_2, Q)$.

PROOF. The first part follows from the fact that $AgrAns(M_1, M_2, Q) \subseteq OverAns(M_1, M_2, Q)$. The second part follows from the fact that if $SO(M_1, M_2, Q) = 1$, all questions are answered by both models. Then we have $AgrAns(M_1, M_2, Q) \subseteq (Q \setminus IncompAns(M_1, M_2, Q))$, using the assumption that answers are consistent. □

Next we define what a shared mental model is in terms of the most important characteristics. The model is a mental model, thus it must be in the mind of an agent. Sharedness is defined with respect to a relevant set of questions Q. Furthermore, we have to indicate by which agents the model is shared. The measure of sharedness is defined in terms of the aspects of similarity as specified above.

DEFINITION 4 (SHARED MENTAL MODEL). *A model M is a mental model that is shared to the extent θ by agents A_1 and A_2 with respect to a set of questions Q iff there is a mental model M_1 of A_1 and M_2 of A_2, both with respect to Q, such that*

1. $SO(M, M_1, Q) = 1$, *and* $SO(M, M_2, Q) = 1$
2. $A(M, M_1, Q) \geq \theta$, *and* $A(M, M_2, Q) \geq \theta$

The definition is easily extendable for handling an arbitrary number n of agents. The definition allows for two important ways to tune it to various situations: varying θ gives a measure of sharedness, varying Q allows to adapt to a specific usage of the model. For example, for some teamwork it is not necessary for every team member to know exactly who does what, as long as each team member knows his own task. This is possible if the amount of interdependencies between sub-tasks is relatively low. For other teamwork in which the tasks are highly interdependent and the dynamics is high, e.g., soccer, it might be fundamental to understand exactly what the others are doing and what you can expect of them. This can also be expressed more precisely by defining expectations and defining sharedness as full agreement of expectations. Making this precise is left for future research.

5. EXAMPLE: BW4T

In this section, we illustrate the concepts defined in the previous sections using an example from the Blocks World for Teams (BW4T) domain [12]. BW4T is an extension of the classic blocks world that is used to research joint activity of heterogeneous teams in a controlled manner. A team of agents have to fill a number of bins with colored blocks that they have to pick up in separate rooms as quickly as a possible. Each bin is to be filled with blocks of a specific color in a specific order. The agents are allowed to communicate with each other but their visual range is limited to the room they are in.To perform this task effectively, the agents have to share a mental model on the order in which tasks are performed, when to communicate, the current task allocation, current location of blocks etc.

The system in our example consists of the whole environment, i.e. the rooms with the blocks, the corridors between the rooms, the bins and the agents. For this system we constructed a set Q of questions regarding, e.g., the current time, the number of blocks per color per room, the required color per position in the three bins, the knowledge about communication requirements, tasking of agents and previous communications. The questions are formulated in such a way that the answer is atomic in the sense that it is not composed of answers to sub-questions.

For example, we formulated questions such as "How many red blocks are there in room 1?". The answer to such a question is a number that can easily be compared to the answer given by another model. Given that there are 12 rooms and 3 colors (white, blue, and red), we formulated 36 questions of the atomic kind for rooms and the number of blocks per color. Similarly, for the three bins, each having three positions, we formulated questions such as "What is the required color at position 1 in bin 1?", leading to 9 questions of this kind. In this way, we constructed 36 + 9 questions that refer to the current state of the environment. Note that over time, the situation changes, because the agents move the blocks around.

Suppose room 1 contains 2 red blocks, 2 white blocks and no blue blocks. Furthermore assume, that agent A, having just arrived in room 1 has been able to observe the blocks in this room, whereas agent B is still en route to room 2 and has no idea about the colors of the blocks in the various rooms as yet. Assume that both agents have an accurate picture of the team task (which color has to go to which position per bin). Taking this set of 45 question Q, then we have that the mental model of agent A, M_A, answers 12 questions out of a total of 45, while M_B, the model of agent B only answers 9 questions. The subject overlap is $SO(M_A, M_B, Q) = 9/45$, and the compatibility is $C(M_A, M_B, Q) = 1$. Also the level of agreement between the models is $A(M_A, M_B, Q) = 9/45$, which in this case equal the subject overlap since the answers do not differ. In order to identify a shared mental model between these agents, we have to restrict the questions to only the part concerning the team task. This model is shared to extent 1. Now, if agent A communicates his findings to agent B, then somewhat later in time the overlap and agreement could grow to 12/45, and the shared mental model would grow when modifying the set of questions accordingly. As the agents walk through the environment, they could achieve the maximum number on measures for these models, as long as they keep informing each other. If this is not done effectively, it may be the case that an agent believes a block to be in a room, while another agent believes it is not there anymore. This would lead to a decreased agreement.

Above, we have considered only questions related to the environment and to the team task, which in this case is also visible in the environment. Of another level are the questions that provide insight into the agents, their tasks, intentions and communication strategies. For this one may, e.g., formulate the following questions: "Under which conditions should agents inform other agents?" which regards what each agent thinks is the common strategy for the team, and per agent the following questions "What is the preferred task order of agent A?", "Which task does agent A have?" , "What is the intention of agent A?". Note that the intention of agents changes over time during the task execution. Finally, we can pose general questions such as "What information was communicated by agent A at time X?", where of course X varies over time, thus leading to an incremental number of questions as the team is at work.

For example, consider that regarding the question "Under which conditions should agents inform other agents?" agent A would answer "An agent communicates when it knows something it knows other agents need to know and everything it intends itself", while B's response would be "An agent communicates when it knows something it knows other agents need to know". The formalizations of these statements could be:

belief(hasTask(Agent,Task)) ∧ belief(requires(Task,Info)) ∧ hasInfo(self,Info) ∧ Agent ≠ self ∧ belief(¬ hasInfo(Agent,Info)) → toBeCommunicatedTo(Info,Agent))

intends(self, X) ∧ belief(¬ hasInfo(Agent,hasTask(self,X))) → toBeCommunicatedTo(hasTask(self,X),Agent)

This implies higher order aspects of the mental models these agents need to have, i.e., a good image of what other agents know about the current situation, knowledge about the tasks and their dependence on information, and information about who has what task. For this example domain, this means that the questions need to be extended: "Which task T does agent A have?", "What information is relevant for task T?", and either object level questions of the form "How many red blocks does agent A believe to be in room 1?" or higher level questions of the form "When can you be sure that an agent knows something?", to which an answer could be *observed(Info, self)* $\vee$ *communicatedBy(Info, Agent)*. Note that the complexity of computing the measures of similarity depends heavily on the complexity of the logic underlying the questions and thus the answers to the questions. The operationalization of testing these measures might require advanced logical theorem proving tools or model checkers.

6. CONCLUSION

In this paper, we have studied the notion of shared mental model, motivated by the idea of taking shared mental model theory as inspiration for the engineering of agents capable of effective teamwork. We have analyzed the notion starting from an analysis of the notion of mental model, and continuing with definitions of similarity of models, leading to a definition of shared mental model. We have illustrated how these definitions can be operationalized using an example in the BW4T domain.

As for future work, there are conceptual as well as engineering challenges. We aim to investigate how theory of mind (agents that have mental models about other agents) fits into this framework. Also, awareness of sharedness may be relevant for effective teamwork and worth investigating. From an engineering perspective, a main challenge for future research is the investigation of mechanisms that lead to a shared mental model that is shared to the extent needed for effective teamwork, which will also depend on the kind of task and environment. Moreover, we will investigate the relation between shared mental models theory as proposed in social psychology, and related (formal) models of agent cooperation in which a notion of sharedness plays a role, such as in joint intentions [11] and shared plans [9].

7. REFERENCES

[1] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider. *The description logic handbook: Theory, implementation, and applications*. Cambridge University Press, 2003.

[2] C. Bolstad and M. Endsley. Shared mental models and shared displays: An empirical evaluation of team performance. *Human Factors and Ergonomics Society Annual Meeting Proceedings*, 43(3):213–217, 1999.

[3] M. Buckland and F. Gey. The relationship between recall and precision. *Journal of the American Society for Information Science*, 45(1):12–19, 1994.

[4] J. A. Cannon-Bowers, E. Salas, and S. Converse. Shared mental models in expert team decision making. In N. J. Castellan, editor, *Individual and group decision making*, pages 221–245. Lawrence Erlbaum Associates, 1993.

[5] A. Clark and D. J. Chalmers. The extended mind. *Analysis*, 58:10–23, 1998.

[6] J. Doyle and D. Ford. Mental models concepts for system dynamics research. *System Dynamics Review*, 14(1):3–29, 1998.

[7] C. Francois. Systemics and cybernetics in a historical perspective. *Systems Research and Behavioral Science*, 16:203–219, 1999.

[8] D. Gentner and A. Stevens. *Mental Models*. Lawrence Erlbaum Associates, New Jersey, 1983.

[9] B. Grosz and S. Kraus. Collaborative plans for complex group action. 86(2):269–357, 1996.

[10] K. V. Hindriks. Programming rational agents in GOAL. In R. H. Bordini, M. Dastani, J. Dix, and A. El Fallah Seghrouchni, editors, *Multi-Agent Programming: Languages, Tools and Applications*. Springer, Berlin, 2009.

[11] N. Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. *Artificial Intelligence Journal*, 74(2), 1995.

[12] M. Johnson, C. Jonker, M. B. van Riemsdijk, P. J. Feltovich, and J. M. Bradshaw. Joint activity testbed: Blocks world for teams (BW4T). In *Proceedings of the Tenth International Workshop on Engineering Societies in the Agents' World (ESAW'09)*, volume 5881 of *LNAI*, pages 254–256. Springer, 2009.

[13] P. N. Johnson-Laird. *Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness*. Cambridge University Press, Cambridge, 1983.

[14] G. Klein, D. D. Woods, J. M. Bradshaw, R. R. Hoffman, and P. J. Feltovich. Ten challenges for making automation a "team player" in joint human-agent activity. *IEEE Intelligent Systems*, 19(6):91–95, 2004.

[15] B. Lim and K. Klein. Team mental models and team performance: A field study of the effects of team mental model similarity and accuracy. *Journal of Organizational Behavior*, 27(4):403, 2006.

[16] E. Mathieu, T. S. Heffner, G. Goodwin, E. Salas, and J. Cannon-Bowers. The influence of shared mental models on team process and performance. *The Journal of Applied Psychology*, 85(2):273–283, 2000.

[17] W. Rouse and N. Morris. On looking into the black box: Prospects and limits in the search for mental models. *Psychological Bulletin*, 100(3):349–363, 1986.

[18] K. Sycara and G. Sukthankar. Literature review of teamwork models. Technical Report CMU-RI-TR-06-50, Carnegie Mellon University, 2006.

[19] J. Yen, X. Fan, S. Sun, T. Hanratty, and J. Dumer. Agents with shared mental models for enhancing team decision makings. *Decision Support Systems*, 41(3):634–653, 2006.

48

Coactive Design

Why Interdependence Must Shape Autonomy

Matthew Johnson[1,2], Jeffrey M. Bradshaw[1] and Paul J. Feltovich[1]
[1]Florida Institute for Human and Machine Cognition (IHMC)
Pensacola, Florida, USA
1-850-202-4476
{mjohnson, jbradshaw and pfeltovich}@ihmc.us

Catholijn M. Jonker[2] and Birna van Riemsdijk[2]
[2]EEMCS, Delft University of Technology
Delft, The Netherlands
+31.15.2781315
c.m.jonker@tudelft.nl and m.b.vanriemsdijk@tudelft.nl

Maarten Sierhuis[2,3]
[3]Carnegie Mellon University
Silicon Valley Campus
Moffett Field, California, USA
+1-650-604-4917
maarten.sierhuis@sv.cmu.edu

ABSTRACT
This paper introduces *Coactive Design* as a new approach to address the increasingly sophisticated roles for people and agents in mixed human-agent systems. The basic premise of *Coactive Design* is that the underlying interdependence of joint activity is a critical design feature. When designing the capabilities that make an agent autonomous, the process should be guided by an understanding of the interdependence in the joint activity. This understanding is then used to shape the operation of agent capabilities and enable appropriate interaction. The success of future human-agent teams hinges not merely on trying to make agents more autonomous, but also in striving to make them more capable of sophisticated interdependence.

Categories and Subject Descriptors
I.2.9 [**Robotics**]: Autonomous vehicles

General Terms
Design, Human Factors, Theory.

Keywords
Coactive, autonomy, interdependence, joint activity.

1. INTRODUCTION
This paper introduces the concept of *coactivity* and presents *Coactive Design* as a novel approach for designing human-agent systems. Throughout the paper we will use the terms "agent" and "robot" interchangeably to mean any artificial actor. Both robot and agent developers continue to pursue much more sophisticated roles for their machines. Some of the envisioned roles include caretaking assistants for the elderly, medical assistants, day care assistants, coworkers in factories and offices, and servants in our homes. Not only are the agents themselves increasing in their capabilities, but also the composition of human-robot systems is growing in scale and heterogeneity. All these requirements showcase the importance of robots transitioning from common roles of today, where they are frequently no more than teleoperated[1] tools, to more sophisticated partners or teammates.

Full teleoperation[1] and full autonomy[2] are often thought of as two extremes on a spectrum. Researchers have been investigating the middle ground between these extremes under various names including mixed-initiative [1], adjustable autonomy [2], collaborative control [3], and sliding autonomy [4]. Each of these approaches attempts to keep the human-agent system operating at a "sweet spot" between the two extremes. As the names suggest, these approaches understand that the ideal is not a fixed location along this spectrum but may need to vary along the spectrum. These approaches and most traditional planning technologies at the foundation of intelligent robotic systems typically take an autonomy-centered approach, focusing on the problems of control and task allocation.

In contrast to these autonomy-centered approaches, *Coactive Design* is a teamwork-centered approach. The concept of teamwork-centered autonomy was addressed by Bradshaw *et al.* [5]. It takes as a beginning the premise that people are working in parallel alongside autonomous systems, and hence adopts the stance that the processes of understanding, problem solving and task execution are necessarily incremental, subject to negotiation, and forever tentative. The basic premise of *Coactive Design* is that, in sophisticated human-agent systems, the underlying interdependence of joint activity is the critical design feature. From this perspective, the design of capabilities to make agents autonomous should be guided by an understanding of the interdependence in the joint activities these agents will undertake. This understanding is then used to shape implementation of agent capabilities, thus enabling appropriate interaction. We no longer look at the problem as simply trying to make agents more autonomous, but, in addition, we strive to make them more capable of being interdependent.

This paper will begin by explaining the different usages of the term *autonomy*. We will discuss several major approaches to human-agent interaction and show how they are mainly autonomy-centered. We will also discuss the ways in which autonomy has been characterized. Next we present the basic premise of Coactive Design and the concept of interdependence itself, which, like autonomy, is a highly nuanced term. We then discuss how Coactive Design fits in relation to prior work and

[1] Teleoperation is manually operating a machine from a distance.

[2] Autonomy will be more fully explained in the following sections, but here it can simply be thought of as "without intervention by other actors."

highlight the new areas of focus. We briefly discuss some preliminary work and then close with a discussion of why this novel approach is important.

2. AUTONOMY

Autonomy has two basic senses in everyday usage. The first sense, self-sufficiency, is about the capability of an entity to take care of itself. Bradshaw [6] refers to this as the descriptive dimension. Similarly, Castelfranchi [7] referred to this as one of the two aspects of "social autonomy" that he called independence. People usually consider robot autonomy in this sense with respect to a particular task. For example, a robot may be able to autonomously navigate in an office environment. The second sense refers to the quality of self-directedness, or freedom from outside control, which Bradshaw calls the prescriptive dimension. Castelfranchi referred to this as autonomy of delegation and considered it another form of "social autonomy." For robots, this usually means freedom from human input or intervention during a particular task.

Castelfranchi argued that both of the preceding senses are reducible to a single aspect and we agree. However, we will continue to keep them separate because it aids our explanation of previous work and because we occasionally need to make the distinction. Castelfranchi also included a third sense of autonomy that he calls "non-social autonomy." This refers to independence from environmental stimuli as opposed to the social autonomy which was about freedom from other agents. Since we do not limit the self-directedness to other agents as Castelfranchi does, we feel it is included in self-directedness.

For this paper, we will use self-sufficiency and self-directedness to distinguish the two senses which often confound discussions on autonomy. Self-sufficiency will be used to express an agent's inherent capability. Self-directedness will be used to express an agent's freedom from outside control or authority.

2.1 How Prior Work is Autonomy-Centered

There have been many approaches to improve human-robot system effectiveness and we will now discuss several of the more prominent approaches. Parts of the discussion below are adapted from [6].

2.1.1 Functional Allocation and Supervisory Control

The concept of automation—which began with the straightforward objective of replacing whenever feasible any task currently performed by a human with a machine that could do the same task better, faster, or cheaper—became one of the first issues to attract the notice of early human factors researchers. These researchers attempted to systematically characterize the general strengths and weaknesses of humans and machines [8]. The resulting discipline of Function Allocation aimed to provide a rational means of determining which system-level functions should be carried out by humans and which by machines. Sheridan proposed Supervisory Control [9], in which a human oversees one or more autonomous systems, allocating tasks. Once control is given to the system, it is ideally expected to complete the tasks without human intervention. Both of these approaches are clearly autonomy-centered, specifically concerned with the self-sufficient aspect of autonomy. The designer's job is to determine what needs to be done and then provide the agent the capability (i.e., self-sufficiency) to do it. Autonomy is therefore shaped by self-sufficiency in this case.

2.1.2 Adjustable/ Adaptive/ Sliding Autonomy

Over time it became plain to researchers that things were not as simple as they first appeared. For example, many functions in complex systems are shared by humans and machines; hence the need to consider synergies and conflicts among the various performers of joint actions. Also, the suitability of a particular human or machine to take on a particular task may vary by time and over different situations; hence the need for methods of function allocation that are dynamic and adaptive. There are many approaches that suggest ways to vary autonomy. Dorais [10] defines adjustable autonomy as "the ability of autonomous systems to operate with dynamically varying levels of independence, intelligence and control." Dias [11] uses the term sliding autonomy, but defines it similarly. Sheridan discusses adaptive automation in which the system must decide which functions to automate and to what extent. We will use adjustable autonomy to refer to them all and take a general meaning of automatically and appropriately adjusting the robot's level of autonomy, in this case the self-directedness aspect, based on the situation. The adjustment may be initiated by the human or by the robot itself. Again, it is clear that these approaches are autonomy-centered, with the focus being on task assignment, control and level of independence. Here the self-directedness shapes the autonomy. One very important concept put forth by these approaches is the importance of adaptivity which will be critical to future systems.

2.1.3 Mixed-Initiative Interaction

Mixed-initiative approaches evolved from a different research community, but share similar ideas and assumptions. Allen defines mixed-initiative as "a flexible interaction strategy, where each agent can contribute to the task what it does best" [1]. Although interaction is used in the description, it is mainly used to negotiate which party does which task. Similarly Myers and Morley describe a framework called "Taskable Reactive Agent Communities (TRAC) [12], which supports the directability of a team of agents by a human supervisor by modifying task guidance." Again directability or task allocation is the central feature. Murphy [13] also uses the term mixed initiative to describe their attention directing system, the goal of which is to get the human to pick up tasks when a robot has a failure. Like all the others, mixed-initiative style approaches are essentially autonomy-centered and frequently focus on task assignment or the authority to act and as such address the self-directedness aspect of autonomy; thus, self-directedness is still the major contributor to shaping the autonomy of the agent. Mixed initiative interaction contributes the valuable insight that joint activity is about interaction and negotiation and that control is not something that is statically assigned, but dynamically shifts as necessary.

2.1.4 Collaborative Control

Collaborative Control is an approach proposed by Fong [3] that uses human-robot dialogue (i.e., queries from the robot and responses, or lack thereof, from the human), as the mechanism for adaptation. As Fong states, "Collaborative control also allows

Cite as: Coactive Design: Why Interdependence Must Shape Autonomy, M. Johnson, J. M. Bradshaw, P. J. Feltovich, C. M. Jonker, B. Riemsdijk and M. Sierhuis, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. XXX-XXX. Copyright © 2010, International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

robots to benefit from human assistance during perception and cognition, and not just planning and command generation" [3]. Collaborative Control is a first step toward Coactive Design, allowing both parties to participate in the same action. Here the interdependence of the navigation task was used to shape the autonomy. The robot was designed to allow the human, who is also involved in the task, to provide assistance in the perceptual and cognitive parts of the task. This assistance is not required, so we are not talking about self-sufficiency, but it is designed for and enabled. Some of the ideas from this approach will be adapted and extended by Coactive Design.

2.2 How Autonomy has been Characterized

A way to gain insight into the focus of approaches in a community is to review how the community categorizes and describes its own work. This also provides a test of our claims that most prior work is autonomy-centered.

Several researchers have worked on describing different levels of autonomy. Yanco [14] made the distinction based on the amount of intervention required. For example, full teleoperation is 100% intervention and 0% automation. Tour guide robots are labeled 100% autonomous and 0% intervention. The assumption in this model is that intervention only occurs when the robot lacks self-sufficiency. However, identifying the percentage of intervention is difficult to quantify particularly when not at the extreme ends of the spectrum. Similarly Sheridan [15] provides a list of levels of autonomy shown in Figure 1.

HIGH	10. The computer decides everything, acts autonomously, ignoring the human.
	9. informs the human only if it, the computer, decides to
	8. informs the human only if asked, or
	7. executes automatically, then necessarily informs the human, and
	6. allows the human a restricted time to veto before automatic execution, or
	5. executes that suggestion if the human approves, or
	4. suggests one alternative
	3. narrows the selection down to a few, or
	2. The computer offers a complete set of decision/action alternatives, or
LOW	1. The computer offers no assistance: human must take all decisions and actions.

Figure 1 Levels of automation [15]

Sheridan's scale is clearly autonomy-centered, as noted by Goodrich and Schultz [16]. Specifically it focused on the self-directed aspect of autonomy. Goodrich and Schultz [16] provided a scale which attempts to focus on interaction instead of autonomy, shown in Figure 2.

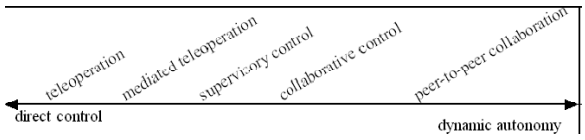


Figure 2 Levels of autonomy with an emphasis on human interaction [16]

Their desire was to capture something more than the previous autonomy-centered views, but it is more of a historical summary. Peer-to-peer collaboration as a term holds promise, but is never clearly defined, as it was a future direction. However, the right most coordinate of Figure 2 is "dynamic autonomy" which sounds a lot like all of the previous autonomy-centered approaches.

Bradshaw describes multiple dimensions of autonomy instead of a single one-dimensional scale of levels [6]. He describes a descriptive dimension and a prescriptive dimension capturing both of the two initial senses of autonomy. He also argues that the measurement of these dimensions should be specific to task and situation.

Castelfranchi suggested dependence as the complement of autonomy [7] and describes dimensions of autonomy in terms of the autonomy/dependence of various capabilities in a standard Procedural Reasoning System (PRS) architecture. These include information, interpretation, know-how, planning, plan discretion, goal dynamics, goal discretion, motivation, reasoning, monitoring, and skill autonomy. Like Bradshaw, Castelfranchi recognizes that autonomy is not a monolithic property, but should be applied to each aspect of the agent. Castelfranchi put it this way: "any needed resource or power within the action-perception loop of an agent define a possible dimension of dependence or autonomy."[7]

3. COACTIVE DESIGN

Coactive Design takes *interdependence* as the central organizing principle among people and agents working together in joint activity. Certainly issues of autonomy are still important, for instance, what does an agent need to do, how well does it need to do it, and how much control do we give it. However, all of these aspects should be guided by an understanding of the interdependence in the joint activity. Then the interdependence can be used to shape the autonomy and enable appropriate interaction. In contrast to autonomous systems designed to take humans out of the loop, we are specifically designing systems to address requirements that allow close and continuous interaction with people.

As we try to design more sophisticated systems, we move along a maturity continuum [17] from dependence to independence to interdependence. The process is a continuum because a small level of independence of agents through autonomy is a prerequisite for interdependence. However, independence is not the supreme achievement [17] in human-human interaction, nor should it be in human-agent systems. Interdependence is much more complex and difficult for both machines and humans. We are no longer dealing with individual *autonomous* actions but with group *participatory* actions [18]. This is a departure from the previous approaches discussed in section 2.1, with the exception of Collaborative Control which began to incorporate all parties into the action (at least in the perceptual and cognitive dimensions). As Clark states, "a person's processes may be very different in individual and joint actions even when they appear identical" [18]. Clark's example is playing a musical solo versus playing a duet. Although the music is the same, the processes involved are very different.

The term *coactive* is specifically chosen to highlight the difference in the Coactive Design approach. There are three meanings [19] associated with coactive:

1) *Joint action*
2) *An impelling or restraining force; a compulsion*
3) *Ecology. Any of the reciprocal actions or effects, such as symbiosis, that can occur in a community.*

Joint action is about each participant being engaged in the same action, or more specifically activity; meaning a process

extended in space and time [18, 20]. Previous work in human-agent interaction focused largely on assigning or allocating tasks to individuals. As we move toward more sophisticated human-machine systems, the activity looks more and more "joint-like." Consider the unmanned aerial vehicle. The first task in development was a standard engineering task to make the vehicle self-sufficient for some tasks (e.g., waypoint following). As the capabilities and robustness increased, the focus shifted to self-directedness (e.g., what am I willing to let this machine do autonomously). Now much of the effort is interdependence focused (e.g., how can I get these vehicles to work effectively as a team with their operators?). It is a natural maturing process and robots and agents are now pushing into new territory.

The compulsion derives from the interdependence inherent in the joint activity. Joint activity means that there is a collective obligation [21] of all parties, even if not currently "assigned" to the task. This obligation includes certain duties and obligations that correlate with good teamwork. These obligations both compel us, for example to provide help, and restrain us, for example not to hog any limited resources. Capturing these obligations is an essential part of Coactive Design and a departure from most previous approaches that do not address the idea of a collective obligation.

The last key feature of Coactive Design is the idea of reciprocal actions. Most previous systems take a unidirectional view. They either focus on automating tasks to offload work from the operator or they focus on enabling the operator to take over a task to make up for poor robot performance or ability. Coactive Design espouses a bidirectional view. For example, if I need to know your status, you must be able to provide status updates. If you can help me make navigation decisions, my navigation algorithm must allow outside guidance. Simply stated; one can only give if the others can take and vice versa. Many of the abilities required for good teamwork required reciprocal abilities from the other team members. In this way Coactive Design focuses on teamwork-centered autonomy. This is another break from the previous work that tended to focus on individualistic autonomy.

Coactive Design is a framework for addressing the more challenging roles for agents (e.g. care taker, medical assistant, coworker, or servant) and human-agent teams, especially heterogeneous teams. These roles have a much higher commitment than other types of interaction, such as passing in a hallway or making a sales transaction with a grocery clerk. The target for Coactive Design is not current teleoperated systems or systems struggling with basic autonomy. We are specifically addressing what a human-agent system would look like if it were to fill one of the roles suggested above. The envisioned roles, if properly performed, have a high level of interdependence that cannot be addressed solely by adjusting who is in control or who is assigned what task—and necessitate a focus on the *coactivity*. In contrast to autonomous systems designed to take humans out of the loop, we are specifically addressing the requirements for close and continuous interaction with people.

Coactive Design has joint activity at its core and joint activity is largely about interdependence. To gain more insight into the aspects of Coactive Design, it is necessary to look more deeply at interdependence.

4. INTERDEPENDENCE

Coordination is foundational to joint activity and is required largely because of interdependencies among activities [22]. Understanding the nature of the interdependencies involved in the coordination is an important part of determining the capability requirements of agents and designing a solution.

4.1 Malone and Crowston

In their interdisciplinary study of coordination Malone and Crowston [23] summarize prior work on coordination in which they drew on Computer Science, Organization Theory, Management Science, Economics, Linguistics and Psychology. They view coordination as managing dependencies. They also characterize the types of dependencies and categorized some of the most common; use of shared resources, simultaneity of processes, producer/consumer relationships, task/subtask roles, task assignment and transfer dependency. Thompson [24] suggested three types of interdependence: pooled, sequential and reciprocal. Pooled interdependence is about each entity contributing a discrete part to the whole and that each is supported by the whole. This is more applicable to large organizations then the smaller teams we are considering. Sequential interdependence maps directly to Malone's producer/consumer category. Reciprocal interdependence is of the bidirectional producer/consumer type. The use of reciprocal here is referring to the cyclical nature of the producer/consumer relation. This is very different from our use of reciprocal which is about the need for complementary capabilities. We believe Malone's categories cover Thompsons and that all of these can be represented by two basic types of dependency: resource and temporal.

4.2 Resource Dependency

Resource dependency can involve a variety of things including a tool, space, the product of a process, or the capability to perform some action. Dependency for resources has received much attention in the literature and is the same as Malone's "shared resource dependency." We will represent an activity A as being resource dependent on resource x as:

$$x \hookrightarrow A$$

There are many ways to formally represent dependence, and we are not attempting to provide another formal specification here. Instead a simplified notation is provided to facilitate this discussion.

4.3 Temporal Dependency

Temporal dependency is the time relation between events or actions. While it is conceivable to view time constraints as a resource as well, this only makes sense when discussing time requirements associated with resources (e.g. I need the hammer for the next five minutes). Hence, it is clearer to keep them separate. For temporal constraints we will need a more detailed definition of the activity involved, so we define an activity A as an activity that spans time from t=0 to t=n such that A is represented by $\{A_0 \ldots A_n\}$ where A_0 is the start of the activity and A_n is the completion of the activity, noting that A_0 can be the same as A_n for actions with negligible duration. Now it is easy to define a serial sequencing temporal dependence such as "A must start after B finishes":

$$B_n \twoheadrightarrow A_0$$

Table I lists a few more examples of temporal dependencies.

Table 1 Types of Temporal Dependency

Types of Temporal Dependency	
$B_n \blacktriangleright\blacktriangleright A_0$	Process A must start after B is complete
$A_0 \blacktriangleright\blacktriangleright B_0$	Process A must start before B starts
$A_0 = B_0$	Processes A must start at the same time as B
$A_n = B_0$	Process A must end when B starts
$A_n = B_n$	Processes A must end at the same time as B
Some A_{0-n} = *Some* B_{0-n}	Process A must temporary intersect B
$A_0 \blacktriangleright\blacktriangleright \mid = B_0$; $B_n \blacktriangleright\blacktriangleright \mid = A_n$	Processes A must be performed concurrently within B

4.4 Malone Revisited

Now we will show how Malone's original listing can be represented as combinations of these two types of dependencies. Assuming B generates x, hence x depends on B ($B \Rightarrow x$), the producer/consumer dependency can be viewed as resource dependence of A on the output of B and a temporal dependence that A must occur after B:

$$B \Rightarrow x \Rightarrow A \textit{ and } B_n \blacktriangleright\blacktriangleright A_0$$

Similarly task/subtask dependence can be viewed as a resource dependence on the subtask B directly and a temporal dependence that requires the subtask to occur within the time span of the task:

$$B \Rightarrow A \textit{ and } A_0 \blacktriangleright\blacktriangleright \mid = B_0 \textit{ and } B_n \blacktriangleright\blacktriangleright \mid = A_n$$

One can generate even more complex time relationships using the types of dependencies discussed in Table I. Task assignment can also be represented this way with the task now being performed by another, thus adding the additional resource dependency of the other agent. Lastly, Malone's transfer dependency is similar to the producer/consumer dependency with the addition of a potentially time dependent exchange of information, which we will call activity C:

$$B \Rightarrow C \Rightarrow A \textit{ and } C_n \blacktriangleright\blacktriangleright A_0 \textit{ and } B_n \blacktriangleright\blacktriangleright C_n$$

In this way, more complex dependencies can be composed from the two basic types; resource and temporal. There are two other types of dependency that we see as critical in Coactive Design that are not captured by Malone's list; soft dependency and monitoring dependency. We will discuss these next.

4.5 Soft Dependency

Dependency can be "hard" meaning that activity A cannot proceed without x, or it can be "soft" meaning that activity A can potentially involve x, but it is not required. For example, in order to enter a room with one door, a robot would have a "hard" dependence on the one door. If the room had two doors, the robot would have a "soft" dependence on both doors. We will represent the "soft" dependency as:

$$x \dashrightarrow A$$

Besides redundant or alternative options, "soft" dependency can also refer to information that is not required, but if provided it could potentially alter the behavior of the recipient. Some examples would be progress appraisals [25]("I'm running late"), warnings ("Watch your step") and unexpected events ("It has started to rain"). While the planning community and others have contributed a large body of work on the standard "hard" dependencies critical to a functioning human-robot system, the "soft" dependencies have received less attention. These types of dependency can lead to richer and more interesting types of interaction than have typically been implemented and are important aspects of Coactive Design.

These typically fall under the compulsion or collective obligation we have discussed. Although not required (hard dependency), understanding them can help shape an agent's autonomy to better support interdependent roles.

4.6 Monitoring Dependency

If there is dependence, either resource or temporal, there is also an implied "monitoring dependency," if joint activity is to be successful. The dependent agent is obligated to monitor the situation appropriately. There are two possible options:

1) Observe the environment (including time or other agents)
2) Wait for a signal or message

If, for example, an agent needs an elevator (resource dependence) the agent can monitor the elevator doors to see when they open. Alternatively, the agent could be notified of availability through signaling (e.g. up arrow light turns on, audible bell, or an elevator operator telling you "going up"). Each option has it challenges but for now we just want to convey that monitoring is an important consideration in Coactive Design. Monitoring dependence also highlights the reciprocal nature of the activity. Not only does the monitoring entity need to monitor, but the monitored entity may need to make certain aspects of its operation transparent.

5. VISUALIZING THE NEW PERSPECTIVE

So how does the coactive design perspective change the way we see the design problem? The first way is to consider joint activity, as explained by Clark [18]. Joint activity highlights the issue of interdependence. We still need to consider autonomy in both its dimensions, but now we must also consider interdependence of the activity, as sketched roughly in Figure 3.

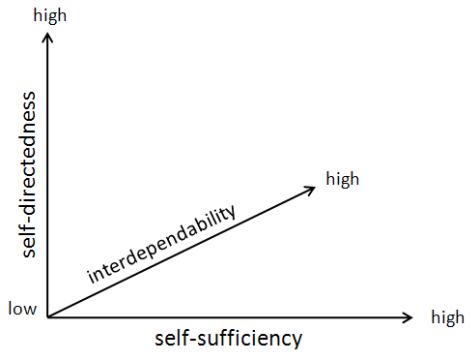


Figure 3 Adding the Third Dimension

The self-sufficiency axis is the capability of an agent to perform a task. Low indicates that the agent is not capable of performing the task without significant help and high indicates that the agent can perform the task reliably without assistance. The self-directedness axis is about freedom from outside control. Low indicates that although possibly capable of performing the task, the agent is not in control. High indicates the agent has the authority over its own

actions, though it does not necessarily imply sufficient competence. The Interdependability axis is about the level at which the agent is able to depend on others or be depended upon by others. This axis is specifically about the capability to be interdependent, not the need or requirement which are captured by the other axes.

We can now map some of the prior work onto this space, shown in Figure 4. First we can talk about where the approaches fall on the new three-dimensional spectrum. The Functional Allocation problem of determining what to automate is really about determining where one wants to be on the self-sufficiency axis. Adjustable autonomy and much of the mixed-initiative approach are about changing along the self-directedness axis. We can also look at how autonomy is characterized in this new model. Yanco's intervention level correlates with the self-sufficiency axis while Sheridan's scale correlates to the self-directed axis. Bradshaw and Castelfranchi address both axes by capturing both aspects of autonomy.

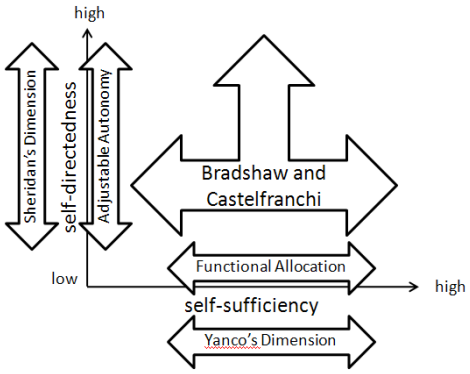


Figure 4 Mapping Prior Work

A degree of self-sufficiency is essential to contribute to any joint activity and self-directedness is a sign of more maturity, but we need to progress toward interdependence to really excel at joint activity.

So far, being autonomy-centered we have stayed within the two autonomy axes. We now push into the third axis that captures interdependability. In additional to the interdependence of joint activity, the Coactive Design perspective includes the reciprocal nature of joint activity addressing for example, the need to match capabilities among participants. This means our interdependence axis needs more than just low and high. It needs to capture the duality of a sender-receiver relationship. In a broad sense, we are talking about the participants in the activity helping one another. This splits the third dimension into two complementary dimensions, as shown in figure 5. We can now map Collaborative Control onto this new perspective. The main feature of Collaborative Control, as discussed earlier, was the ability for the human to provide assistance to the robot at the perceptual and cognitive levels. The wonderful insight of Collaborative Control was that tasks can be done more effectively if done collaboratively. The Coactive Design perspective extends this to allowing for the machine to assist the human providing the complementary side of the axis.

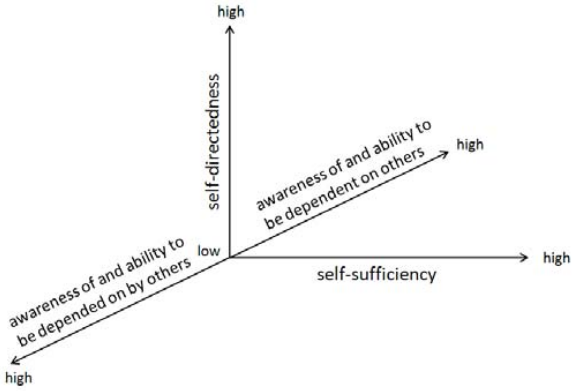


Figure 5 Mapping Collaborative Control

Although we are showing a single set of axes for simplicity, The Coactive Design perspective considers all aspects of an agent's sense-act loop. This is directly in line with Castelfranchi's [7] break down of autonomy based on the components of a PRS system. The take away message is not the support of any particular cognitive model, but instead the concept that there are many aspects to an agent as it performs in a joint activity. Just as Castelfranchi argued that autonomy can occur at any of these "levels" or dimensions, Coactive Design argues that the ability to be *interdependent* exists at each "level" or dimension as well.

Interdependence brings us to the last change in perspective; compulsion. This part of Coactive Design is about focusing on the monitoring and soft dependencies in a joint activity. Good teams distinguish themselves by handling the "soft" dependencies better, improving performance, efficiency, and/or situation awareness. Coactive Design approaches will need to consider "soft" dependencies. Furthermore, identifying areas of dependence in joint activity (both hard and soft) has been stated as a critical part of Coactive Design.

6. SUPPORT FOR COACTIVE DESIGN

We provide supporting evidence for our claims from three sources; a preliminary study of our own, results from other's recent work, and some observations about autonomy, coordination and people.

6.1 PRELIMINARY STUDY

We have begun to investigate the implications of Coactive Design experimentally. We started with a very simple example domain and intend to increase complexity as we progress. Our first domain, Blocks World for Teams (BW4T)[26] (tasks were done in groups), was chosen to be as simple as possible. Similar to the classic AI planning problem of Blocks World, the goal of BW4T is to "stack" colored blocks in order. To keep things simple, the blocks are un-stacked to begin with, so un-stacking is not necessary. The degree of interdependence that is embedded in the task is represented by the complexity of color orderings within the goal stack. The task environment is composed of nine rooms containing a random assortment of boxes and a drop off area for the goal. The environment is hidden from each of the players, except for the contents of the current room. There are basically two tasks in this domain; find a box and deliver a box to the drop

off area. In some simple cases, the task could be done without any coordination, but it is clear that coordination (i.e. the players managing the interdependencies among their moves) is highly beneficial.

Although a simple domain, this example demonstrates the complexity of coordination and interdependence even in the simplest domain. We ran twelve subjects in various team sizes (2, 3, 4, 5, 6, and 8). The teams were all human (i.e. no agents) for this pilot study. The subjects were allowed to talk openly to one another. Although too early to be conclusive, our initial results are interesting and support our claims. As the activity became more interdependent (more complex configurations of blocks as goals), we noted an increase in the number of coordination attempts, as would be expected. We also noted some interesting aspects of the communication. Although only two basic tasks are involved, we observed a wide variety of communications. Of particular interest were the large number of communications that were about *soft dependencies* and *monitoring dependencies*. Progress appraisal was also a common theme. A final observation was that not only the amount of communication changed with the degree of interdependence in the task, but the pattern of communication varied as well.

These initial results come from the first of a sequence of planned experiments of increasing complexity and we cannot make any firm conclusions, but they support the premise of Coactive Design and demonstrate that even in simple tasks, the coordination involved in managing the interdependence can be quite complex.

6.2 Results from Recent Work

There are several examples from recent Human-Robot Interaction work that support our approach. Fong's [3] work demonstrated the support of frail autonomy by making the obstacle avoidance activity a participatory one with matching reciprocal functionality. Stubbs [27] noted that as autonomy increases, transparency became the biggest problem in a remote rover. This is a real world example of how autonomy solves some problems, but at the same time creates new issues that we feel are a direct result of the coactive nature of the task. These examples and our preliminary study highlight the importance of understanding interdependence and using this understanding to shape autonomy.

6.3 The Nature of Autonomy

Autonomy is inherently frailty. Robots, like their creators, will always be imperfect. This underlying truth necessitates human involvement at some level and accentuates the importance of teamwork. Frailty means one will have unexpected events (failures). You cannot overcome failed autonomy with autonomy, but you can with teamwork (e.g. Fong's collaborative control [3]).

Additionally, Christofferson and Woods [28] describe the "substitution myth": the erroneous notion that automation activities simply can be substituted for human activities without otherwise affecting the operation of the system. Even if frailty were not an issue, the "substitution myth" reminds us that autonomy is not removing something, but merely changing the nature of it. Humans cannot simply offload tasks to the robots without incurring some coordination penalty. This is not a problem as long as we keep in mind that autonomy is not an end in itself, but rather a means to supporting productive interaction [16].

6.4 The Nature of Coordination

Once a base level of competence is achieved, coordination of joint activity (teamwork at its simplest form) will take on an ever increasingly important role in the design of a system. This trend was noted by Allen who reported that "the only type of interactions supported by a typical state-of-the-art planning system (namely, adding a new course of action) handled less than 25% of the interactions and that much of the interaction was concerned with maintaining the communication (summarizing and clarifying, for example) or managing the collaboration (discussing the problem solving strategy) [1]." Autonomy centered approaches tend to focus on coordination of content (what they intend to do). Coactive Design also includes coordination of the process (physical and mental systems to carry out the former [18]).

6.5 The Nature of People

As agents move toward greater and greater autonomy, several researchers have expressed concerns. Norman states that "the danger [of intelligent agents] comes when agents start wresting away control, doing things behind your back, making decisions on your behalf, taking actions and, in general, taking over [29]." Simply deciding who is doing what is insufficient, because the human will always need to understand a certain amount of the activity.

Additionally, humans are typically the desired beneficiaries of the fruits of the robot labor. We are the reason for the system and will always want access to the system. Not only do we want access to understand the system, but we also want input to affect it. To paraphrase Kidd [30], it is not that human skill is required, but that human involvement is desired.

7. CONCLUSION

We have introduced Coactive Design as a new approach to address the increasingly sophisticated roles for people and agents in mixed human-agent systems. The basic premise of *Coactive Design* is that, in sophisticated human-agent systems, the underlying interdependence of joint activity is the critical design feature. We have argued that when designing the capabilities that make an agent autonomous, the process should be guided by an understanding of the interdependence in the joint activity. The understanding of interdependence is then used to shape the operation of agent capabilities and enable appropriate interaction. The success of future human-agent teams hinges not merely on trying to make agents more autonomous, but also in striving to make them more capable of sophisticated interdependence.

8. REFERENCES

[1] Allen, J.E., C.I. Guinn, and E. Horvtz, *Mixed-Initiative Interaction.* IEEE Intelligent Systems, 1999. 14(5): p. 14-23.

[2] Kortenkamp, D., *Designing an Architecture for Adjustably Autonomous Robot Teams*, in *Revised Papers from the PRICAI 2000 Workshop Reader, Four Workshops held at PRICAI 2000 on Advances in Artificial Intelligence*. 2001, Springer-Verlag.

[3] Fong, T.W., *Collaborative Control: A Robot-Centric Model for Vehicle Teleoperation.* 2001, Robotics Institute, Carnegie Mellon University: Pittsburgh, PA.

[4] Brookshire, J., S. Singh, and R. Simmons. *Preliminary Results in Sliding Autonomy for Coordinated Teams*. in *Proceedings of The 2004 Spring Symposium Series*. 2004.
[5] Bradshaw, J.M., et al., *Teamwork-centered autonomy for extended human-agent interaction in space applications*, in *In Proceedings of the AAAI Spring Symposium*. 2004, AAAI Press. p. 22-24.
[6] Bradshaw, J.M., et al., *Dimensions of Adjustable Autonomy and Mixed-Initiative Interaction*, in *Agents and Computational Autonomy*. 2004, Springer. p. 17-39.
[7] Castelfranchi, C., *Founding Agents' "Autonomy" on Dependence Theory*, in *ECAI*. 2000. p. 353-357.
[8] Fitts, P.M., *Human engineering for an effective air-navigation and traffic-control system*. 1951, Washington,: National Research Council, Division of Anthropology and Psychology, Committee on Aviation Psychology. xii, 84 p.
[9] Sheridan, T.B., *Telerobotics, automation, and human supervisory control*. 1992, Cambridge, Mass.: MIT Press. xx, 393 p.
[10] Dorais, G. and D. Kortenkamp, *Designing Human-Centered Autonomous Agents*, in *Revised Papers from the PRICAI 2000 Workshop Reader, Four Workshops held at PRICAI 2000 on Advances in Artificial Intelligence*. 2001, Springer-Verlag.
[11] Dias, M.B., et al., *Sliding Autonomy for Peer-To-Peer Human-Robot Teams*. 2008, Robotics Institute: Pittsburgh, PA. Myers, K.L. and D.N. Morley. *Directing Agent Communities: An Initial Framework*. in *Proceedings of the IJCAI Workshop on Autonomy, Delegation, and Control: Interacting with Autonomous Agents*. 2001. Seattle, WA.
[12] Murphy, R., et al. (2000) *Mixed-initiative Control of Multiple Heterogeneous Robots for USAR*.
[13] Yanco, H.A. and J.L. Drury. *A Taxonomy for Human-Robot Interaction*. in *AAAI Fall Symposium on Human-Robot Interaction*. 2002.
[14] Parasuraman, R., T. Sheridan, and C. Wickens, *A model for types and levels of human interaction with automation.* Systems, Man and Cybernetics, Part A, IEEE Transactions on, 2000. 30(3): p. 286-297.
[15] Goodrich, M.A. and A.C. Schultz, *Human-robot interaction: a survey*. Found. Trends Hum.-Comput. Interact., 2007. 1(3): p. 203-275.
[16] Covey, S.R., *The 7 Habits of Highly Effective People*. 1989, New York: Free Press.
[17] Clark, H.H., *Using language*. 1996, Cambridge [England] ; New York: Cambridge University Press. xi, 432 p.
[18] *coaction*, in *http://dictionary.reference.com/browse/coactive*.
[19] Sierhuis, M., *"It's not just goals all the way down" - "It's activities all the way down"* in *Engineering Societies in the Agents World VII, 7th International, Workshop, ESAW 2006*. 2007: Dublin, Ireland.
[20] Diggelen, J.v., et al., *Implementing Collective Obligations in Human-Agent Teams using KAoS Policies*, in *Proceedings of Coordination, Organization, Institutions and Norms in Agent Systems (COIN)*. 2009, Springer.
[21] Feltovich, P.J., et al., *Toward an Ontology of Regulation: Socially-Based Support for Coordination in Human and Machine Joint Activity*, in *Engineering Societies in the Agents World VII* G. O'Hare, et al., Editors. 2007, Springer: Heidelberg, Germany. p. 175-192.
[22] Malone, T.W. and K. Crowston, *The interdisciplinary study of coordination.* ACM Comput. Surv., 1994. 26(1): p. 87-119.
[23] Thompson, J.D., *Organizations in action; social science bases of administrative theory*. 1967, New York,: McGraw-Hill. xi, 192 p.
[24] Feltovich, P.J., et al., *Progress Appraisal as a Challenging Element of Coordination in Human and Machine Joint Activity*, in *Engineering Societies in the Agents World VIII* A. Artikis, et al., Editors. 2008, Springer: Heidelberg, Germany. p. 124-141.
[25] Johnson, M., et al., *Joint Activity Testbed: Blocks World for Teams (BW4T)* in *Engineering Societies in the Agents World X*. 2009.
[26] Stubbs, K., P. Hinds, and D. Wettergreen, *Autonomy and common ground in human-robot interaction: A field study.* IEEE Intelligent Systems, 2007(Special Issue on Interacting with Autonomy): p. 42-50.
[27] Christoffersen, K. and D.D. Woods, *How to Make Automated Systems Team Players.* 2002.
[28] Norman, D.A., *The invisible computer : why good products can fail, the personal computer is so complex, and information appliances are the solution*. 1998, Cambridge, Mass.: MIT Press. xii, 302 p.
[29] Kidd, P.T., *Design of human-centered robotic systems*, in *Human-Robot Interaction*, M. Rahimi and W. Karwowski, Editors. 1992, Taylor & Francis. p. 225-241.

Initial Steps Towards Run-Time Support for Norm-Governed Systems

Visara Urovi[1], Stefano Bromuri[1], Kostas Stathis[1] and Alexander Artikis[2]
[1]Royal Holloway, University of London, Egham Surrey, England
{ visara, stefano, kostas } @ cs.rhul.ac.uk
[2]Institute of Informatics & Telecommunications, NCSR "Demokritos", Greece
a.artikis@iit.demokritos.gr

ABSTRACT

We present a knowledge representation framework with an associated run-time support infrastructure that is able to compute, for the benefit of the members of a norm-governed multi-agent system, the physically possible and permitted actions current at each time, as well as sanctions that should be applied to violations of prohibitions. To offer the envisioned run-time support we use an Event Calculus dialect for efficient temporal reasoning. Both the knowledge representation framework and its associated infrastructure are highly configurable in the sense that they can be appropriately distributed in order to support real-time responses to agent requests. To exemplify the ideas, we apply the infrastructure on a benchmark scenario for multi-agent systems. Through experimental evaluation we also show how distributing our infrastructure can provide run-time support to large-scale multi-agent systems regulated by norms.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Intelligent Agent Environment, Norm Governed Multi-agent System

General Terms

Design, Experimentation, Measurement

Keywords

social interaction, run-time service, GOLEM, event calculus

1. INTRODUCTION

An open multi-agent system [26], such as an electronic market, is often characterized as a computing system where software agents developed by different parties are deployed within an application domain to achieve specific objectives. An important characteristic of this class of applications is that the various parties developing the agents may have competing goals in the application domain and, as a result, agent developers for a specific party will have every interest to hide their agent's internal state from the rest of the agents in the system. Although openness of this kind may encourage many agents to participate in an application, interactions in the system must be regulated so that to convince skeptical agents that the overall specification of the application domain is respected.

Norm-governed multi-agent systems [15, 1] are open multi-agent systems that are regulated according to the normative relations that may exist between member agents, such as permission, obligation, and institutional power [16], including sanctioning mechanisms dealing with violations of prohibitions and non-compliance with obligations. Although knowledge representation frameworks for specifying such relations exist, these frameworks often focus on the expressive power of the formalism proposed and often abstract away from the computational aspects and experimental evaluation. Works studying executable specifications exist but they normally do not provide experimental evaluations of multi-agent system deployment over distributed networks. As a result, the computational behaviour of many representation frameworks for norm-governed multi-agent systems is often studied in isolation, at times theoretically only, and in many occasions experimental evaluation is left unexplored.

The aim of this paper is to use a specific knowledge representation framework to develop an infrastructure for computing at run-time the physically possible actions, permissions, and sanctions, and eventually the obligations, and institutional powers of the members of a norm-governed system. The need for such an infrastructure is motivated by the observation that agents cannot be expected to be capable of computing these normative relations on their own. Practical reasons for this include (a) computational constraints agents may have (e.g. due to lack of CPU cycles, memory, or battery), and (b) incomplete knowledge agents may have about the application state (e.g. due to a partial view of the environment).

Our run-time infrastructure integrates selected versions of the Event Calculus for describing an open multi-agent system as two concurrent and interconnected composite structures that evolve over time: one representing the physical environment of the open multi-agent system and the other representing the social environment. The focus of our knowledge representation framework and its associated run-time infrastructure is to provide real-time responses to agent requests. The novelty of our approach relies on the ability of our framework to provide a distributed implementation of the Event Calculus for Norm Governed Systems. The advantage here is that by distributing a norm-governed application we can efficiently compute the distributed social and the physical states of the system.

The paper is organised as follows. First, we introduce a scenario of a norm-governed multi-agent system. We then use this scenario to describe our run-time infrastructure, the knowledge representation framework and extensions of this framework to support a social state with norms. An experimental evaluation of the approach is presented after that, followed by a comparison with related work. Finally, we

summarize our approach and outline plans for future work.

2. THE OPEN PACKET WORLD

To exemplify the framework and experiment with the proposed infrastructure we will use the *Packet World* [28]. As seen in Fig. 1(a)(i), a set of agents are situated in a rectangular grid (8 x 8 here) consisting of a number of colored packets (squares) and destination points (circles). Agents (a1, a2, a3, and a4 in Fig. 1(a)(i))) move around the grid to pick colored packets which they must deliver in destinations that match a packet's color. As agents can see only part of the grid at any one time (the square around agent a2 represents the perception range of this agent) they often need to collaborate with each other. Collaboration results in agents forming teams to deliver packets and placing flags in locations for letting other agents know that a particular area has been explored and has no packets left. Also, each agent is powered by a battery that discharges as the agent moves in the grid. The battery can be recharged using a battery charger (situated in location (7,8) of Fig. 1(a)(i))). This charger emits a gradient whose value is larger if the agent is far away from the charger and smaller if the agent is closer to the charger.

We are interested here in a variation of Packet World, which we will refer to as *Open Packet World*. This variation differs from the original version as follows. We make the scenario competitive by giving points to agents if they deliver packets to appropriate destinations. Agents are now antagonistic and may be developed by different parties. For instance an agent may try to deceive other agents by placing a flag in an area that has packets. As a result of these extensions we introduce norms. Violation of norms results in sanctions. One type of sanction, in this example, is the the reduction of points of the violating agent. In this paper we focus on permissions and sanctions. Other normative relations, such as institutional power, will be considered in future work.

The Open Packet World (OPW) presents a number of knowledge representation challenges for a norm-governed system. Unlike other practical applications, e.g. electronic markets, it does not require speech acts only but also the simulation of physical actions, which in turn necessitates the representation of physical possibility in the system. Physical possibility requires the representation of a physical environment whose state should be distinct from the state of the social environment. OPW is also convenient from the point of view of experimentation in that we can make the experimental conditions harder by increasing the size of the grid, the number of agents and the number of packets/destinations. In addition, because of the intuitive nature of actions taking place in it, it can be easily visualized.

3. RUN-TIME INFRASTRUCTURE

To experiment with our scenario we use the GOLEM agent platform[1]. GOLEM supports the deployment of *agents* - cognitive entities that can reason about sensory input received from the environment and act upon it, *objects* - resources that lack cognitive ability, and *containers* - virtual spaces containing agents and objects, capturing their ongoing interactions in terms of an *event-based* approach.

[1] http://golem.cs.rhul.ac.uk

A GOLEM container represents a portion of the distributed agent environment and it works as a mediator for the interaction taking place between agents and objects. Events in containers describe what happens in the agent environment as a result of actions being performed by agents. Since interaction is mediated by the containers, the agent has no access to the description of what happens in the environment, however it can observe the state of the container and decide what actions to perform.

In this paper we are not concerned with the implementation of agents. Instead, we are concerned with an implementation of a software framework informing the decision-making of agents by computing, on their behalf, the normative positions current at each time. Whether an agent complies with these positions depends entirely on the implementation of the agent. In general, there is a clear separation between the agent code and the code of our software framework. The code presented in the paper belongs entirely to the proposed software framework. A part of that code — the specification of norms and physical possibility — are application-dependent.

3.1 The Open Packet World in GOLEM

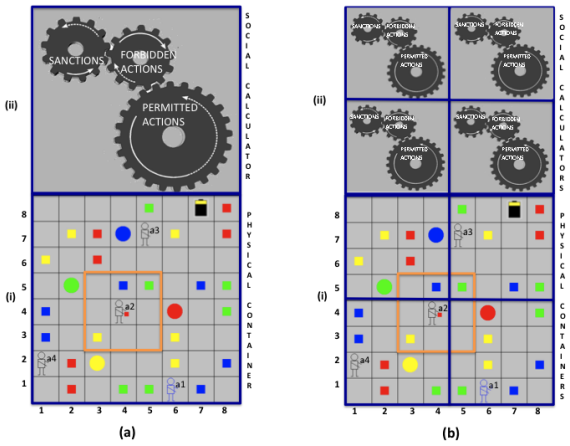


Figure 1: Open Packet-World as a Norm-Governed System

The simplest way to model OPW in GOLEM is shown in Fig. 1(a), where we deploy one container representing the world (see Fig. 1(a)(i)) and extend it in a way that contains a social state representing the normative aspects of the system (see Fig. 1(a)(ii)). Although a single container specification for the original Packet World has been implemented in [2], this container did not have a social state. Here we extend a container with a social state managed by an active object which we call *Social Calculator*. This object computes the agents' permissions and sanctions and publicises this information upon request.

An alternative way to model OPW is to split the physical state of a single container into smaller states that we distribute in different containers. A possible distribution is shown in Fig. 1(b), where we use four 4 x 4 adjacent containers for OPW (see Fig. 1(b)(i)) together with their corre-

sponding Social Calculators (see Fig. 1(b)(ii)). Issues such as distributing the perception range of an agent in different containers (as it is the case with ag2) and moving between containers are already described in [3]. Here we show how containers can use a social state to support a norm-governed system.

3.2 The Physical State of Containers

To represent the state of a GOLEM container we use the object-based notation of C-logic, a formalism that describes objects as complex terms that have a straightforward translation to first-order logic [4]. The complex term below, for example, represents the state of a 2 x 2 packet world with one agent, one packet, one destination and one battery:

```
packet_world:c1[
  address ⇒ "container://one@134.219.7.1:13000",
  type ⇒ open,
  grid ⇒ {square:sq1, square:sq2, square:sq3, square:sq4}
  entities ⇒ {picker:ag1, packet:p1, dest:d1, battery:b1}
]
```

Object instances of this kind belong to classes (e.g. packet_world), are characterized by unique identifiers (e.g. c1), and have attributes (e.g. address). The representation of the 8 x 8 grid of Fig. 1 is similar but larger, i.e. there are more agents, packets, destinations, and squares.

In GOLEM complex instances of objects evolve as a result of events happening in the state of a container. An event happens as a result of entities, such as agents, attempting to act in the environment. For example the assertions:

```
attempt(e14, 100).
do:e14 [actor ⇒ ag1, act ⇒ move, location⇒ sq3].
```

describe an attempt e14 at time 100, containing a physical action made by agent ag1 wishing to move to location sq3. In GOLEM, an attempt becomes an event that happens if the attempt is possible:

```
happens(Event, T) ← attempt(Event, T), possible(Event, T).
```

Happenings of events cause the state of a container C to evolve over time. To query the value Val of an attribute Attr for an entity Id of container C at a specific time T, we will use the definition:

```
solve_at(C, Id, Class, Attr, Val, T) ←
    holds_at(C, container, entity_of, Id, T),
    holds_at(Id, Class, Attr, Val, T).
```

holds_at/5 is defined by the top-level clauses of the *Object Event Calculus* (OEC) [18] and specified as:

```
holds_at(Id, Class, Attr, Val, T)←
  happens(E, Ti), Ti ≤ T,
  initiates(E, Id, Class, Attr, Val),
  not broken(Id, Class, Attr, Val, Ti, T).

broken(Id, Class, Attr, Val, Ti, Tn)←
  happens(E, Tj), Ti < Tj ≤Tn,
  terminates(E, Id, Class, Attr, Val).
```

The above definitions describe how the value Val of an attribute Attr for specific Class instance identified by Id hold at a particular time T, as in the usual Event Calculus [20]. Given an event E, the initiates/5 predicate assigns to the attributes Attr of an object identified by the Id and of class Class a value Val. The terminates/5 predicate has a similar meaning, with the only difference that the event E terminates the value Val of the attribute of an object. The remaining OEC clauses (see [18] for more details). describe how events create instances of C-logic like objects, assign these instances to classes, represent basic hierarchical inheritance where sub-classes inherit attributes from superclasses, destroy complex terms, and terminate single value and multi-valued attributes.

The possible/2 are application dependent rules that specify physical possibility. Bellow, we show an example of how we use the OEC to express a possible/2 rule in OPW:

```
possible(E, T)←
  do:E [actor ⇒ A, act ⇒ move, location⇒ SqB],
  solve_at(this, A, picker, position, SqA, T),
  adjacent(SqA, SqB),
  not occupied(SqB, T).
```

The above rule states that it is possible for an agent to move to a location SqB if the agent is currently in location SqA, SqA is adjacent to SqB, and SqB is not occupied. The keyword **this** is used here to refer to the identifier of the current container.

3.3 Containers with Social State

We extend GOLEM containers with a social state, formalized as a C-logic structure that has a reference to the physical state, and extends this physical state with social attributes to hold information about (a) any current sanctions imposed on any of the agents and (b) how many points agents have collected so far. An example snapshot of a social state for OPW is shown below:

```
packet_world_social_state: s1 [
    physical_state⇒ packet_world:c1,
    sanctions⇒ {sanction:s1 [agent ⇒ a2, ticket ⇒ 5]},
    records⇒ {record:r1[agent ⇒ a1, points ⇒ 35],
              record:r2[agent ⇒ a2, points ⇒ 25]}
]
```

The term above states that agent a2 has been sanctioned with 5 points. We show the records of two agents only to save space. Agent a1 has collected 35 points, while a2 has collected 25 after the sanction is applied. A social state does not contain explicitly the permitted actions. These are defined implicitly in terms of rules. We write:

```
permitted(Event, T)← not forbidden(Event, T).
```

to state that actions specified in events are permitted only if they are not forbidden. Forbidden actions and the evolution of the social state due to these actions are specified in an application dependent manner. A forbidden/2 rule in OPW can be expressed as follows:

```
forbidden(E, T) ←
  do:E[actor ⇒ A, act⇒drop, object⇒flag, location⇒SqA],
  solve_at(this, Id, packet, position, SqB, T),
  adjacent(SqA, SqB).
```

states that it is forbidden for an agent A to drop a flag in location SqA if there are packets nearby.

When a forbidden act has taken place, the Social Calculator raises a violation, which results in a sanction.

```
initiates(E, R, record, points, Points)←
  happens(E,T),
  violation:E[sanction:S [ticket⇒ SanctionPs, agent ⇒ A]],
  solve_at(this, R, record, agent, A, T),
  solve_at(this, R, record, points, OldPoints, T),
  Points = OldPoints - SanctionPs.
```

initiates/5 updates the points of agent A as a consequence of receiving a sanction S at time T. This simple example shows how events happening in the physical environment (e.g. dropping a flag in a location of the grid) affect the social state of the application (e.g. through the initiation of a sanction on the agent that dropped the flag). More complex permissions and sanctions are formalized similarly.

3.4 Distributing a Norm-Governed Application

One important feature of our knowledge representation framework is that we can distribute the state of a norm-governed application into multiple containers in order to support the parallel evaluation of physical and social states. Distributing the system among multiple containers is not a novel architectural idea; however, the proposed architecture — distributed implementation of the Event Calculus supporting norm-governed systems — is, to the best of our knowledge, novel

In particular, GOLEM supports this feature with the *Ambient Event Calculus* (AEC) [3]. The AEC uses the OEC, described earlier, to query C-logic like objects and their attributes that may be situated in different containers, as with the OPW version of Fig. 1(b). Given a container C and a starting Path, we can query a maximum number of neighbors Max, returning a final Path* where an object identifier Id, class Cls, attribute Attr, and value Val hold at time T:

```
neighbouring_at(C, Path, Path*, Max, Id, Cls, Attr, Val, T)←
  Max >= 0,
  locally_at(C, Path, Path*, Id, Cls, Attr, Val, T).

neighbouring_at(C, Path, Path*, Max, Id, Cls, Attr, Val, T)←
  holds_at(C, container, neighbour, N, T),
  not member(N, Path),
  Max* is Max - 1,
  append(Path, [C], New),
  neighbouring_at(N, New, Path*, Max*, Id, Cls, Attr, Val, T).
```

The first clause checks whether the object is in the local state of a container. locally_at/8 checks with holds_at/5 to find the object in the container's state, including sub-containers[2], if any. The second clause looks for neighbors. If a new neighbor N is found, this neighbor is asked the query but in the context of a New path and a new Max*.

We are now in a position to customize our representation for distributing the physical and social state by redefining the solve_at/6. The definition below has the effect of changing all the physical and social rules to work with distributed containers:

```
solve_at(C, Id, Class, Attr, Val, T) ←
    neighbouring_at(C, [], _, 1, Id, Class, Attr, Val, T).
```

The [] list above states that the initial path is empty, the underscore '_', that we are not interested in the resulting path, and the number 1 indicates that we should look at all neighbors whose distance is one step from the current container. In this way, we can query all the neighbors of a container in the OPW of Fig. 1(b).

3.5 Implementation Issues

The AEC is implemented on top of OEC [19] which is an object-oriented optimised version of EC. Clearly EC can be implemented in other programming languages, such as Java and C. We adopted the logic programming approach partly because EC was originally developed as a logic programming language, and partly because of the declarative semantics and concise representation offered by logic programs. An area of future work is to test implementations of EC in other programming languages

The top-level description of OEC is specified below:

```
holds_at(Obj,Attr,Val,T):-
    object(Obj,Attr,Val,start(E)),
    time(E,T1), T1 =< T,
    not (object(Obj,Attr,Val,end(Evstar)),
        time(Evstar,T2), T2>T1, T2 <T).
```

The main difference between this OEC version and the one discussed earlier is that now we add all new properties that are initiated/terminated as object/4 assertions whenever a new event description is added to the container's state. The key is that we store these new properties with time periods denoted by terms such as start(e1) and end(e2). For example, in OPW the assertions below:

```
time(e1, 2).
time(e2, 7).
object(ag1, position, [3,4], start(e1)).
object(ag1, position, [3,4], end(e2)).
object(ag1, position, [4,4], start(e2)).
```

describe how agent a1 moved to position [3,4] at time 2 and changed it to [4,4] at time 7. We know that the periods in the state of a container are either closed or open intervals which persist into the future. A new event such as e2 either starts a new period of time (i.e. start(e2)) for a conclusion or ends a period of time which was started by another event (i.e. end(e2)). The optimization is obtained now because the new event is either related to the attributes of objects or the class membership, so we do not need to check all the events that have happened, as with the previous OEC version. Our implementation also uses indexing on the arguments of object/4 assertions, so that if the first three arguments are specified, the time to retrieve the term is $O(1)$ (which is typically the case with GOLEM queries).

When we distribute the system in many containers we may have a synchronisation problem due to the different timing in different containers. This issue was already addressed in [3] by applying a precise time protocol between sub and super containers.

Another important component of our implementation is that queries to the social and physical environment are executed in parallel. An example of the multithreaded implementation is shown below for how we implement attempts of agents:

```
attempt(E, T):-
    par([exec(possible(E, T), true), exec(permitted(E,T), R)]),
    add(E, R, T).
```

The above program will be called by an agent that wishes to perform an action specified as an event E. The event provides input to two parallel threads, one executing possible(E,T) (which must succeed i.e. return true) and the other executing permitted(E,T) (which must have result R i.e. return true or false). If the event is concluded possible by the first thread, it will be added in the state of the container using add/3; otherwise, the attempt will fail. If the event is concluded possible by the first thread but not permitted (R=false) by the second thread, then the Social Calculator will be triggered by add/3 to produce a sanction in the social state.

4. EXPERIMENTAL EVALUATION

Using our OPW scenario, we conducted a number of experiments to evaluate the performance of the system with different configurations. In particular, we measured the performance with a distributed versus centralised deployment of the system to show how the number of entities, the size of the environment and the distribution affect the performances. In all experiments, we measured the time to compute whether an action is physically possible and whether

[2]Due to lack of space and the fact that sub-containers are not required in our example, we refer the interested reader to [3] for a definition of locally_at/8.

an action is permitted. More specifically, we measured the time taken for possible/2 and forbidden/2 rules against an action performed by an agent in the environment. Then we related this time with the number of events produced.

In the first series of tests, we tested OPW in a centralized GOLEM container deployed in an Intel Centrino Core 2 Duo 2.66GHz with 4GB of RAM. The environment was represented by a 40x40 grid and 100 packets were collected by the agents and released into one of the 8 destinations in the grid. We run the first test with 10 agents, the second test with 30 agents and the third test with 50 agents. In all of the runs, the agent "minds" (reasoning components) were deployed in a separate machine and were remotely connected with their "bodies" (action execution components) deployed in the GOLEM container.

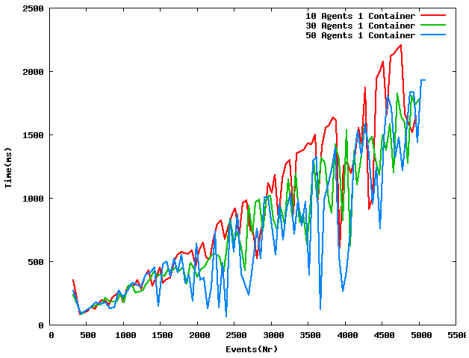


Figure 2: Time to query the physical and the social state of a GOLEM container with 10/30/50 Agents

Fig. 2 shows three linear curves representing the average time to compute a query in a single GOLEM container with respectively 10, 30 and 50 agents. Since the evaluation of the two states is done concurrently, the curves represent the worst case between the social and the physical state.

All the three curves follow a linear behaviour suggesting that the time to query a GOLEM container grows linearly with the number of events produced in the container. The fluctuations in the curves are explained as follows. The high peaks show the worst case where the attempted action was either impossible or not permitted or both. As we check possible and permitted actions in parallel and we wait for both threads to finish the execution, the time shown is the one that took longer between the two. Alternatively, the lowest peaks show the best case where the attempted action was either possible or permitted or both. As before, the one shown is the one that took longer.

We can represent the time T_c to compute the social and physical state for a centralized container with the following equation:

$$T_c = a * E + t0 \text{ with } a \sim Ne/Na$$

where Ne is the number of entities in the system, Na is the number of active entities performing events, E is the number of events in the system and $t0$ is intial time to register the entities in the container. As the number of agents increases, then Na increases, which means that a decreases, which results in better performance. This is due to the fact that the OEC is optimized to deal with events indexed by the identifiers of entities in the agent environment. For example, if we have 10 agents, 5000 events, and assuming that all agents perform the same number of events, each time that we call a solve_at/6 predicate (e.g solve_at(c1, ag1, picker, position, [3,4], 100)), the search for the value of an agent attribute will evaluate a maximum of 500 entries (5000/10), while in the same conditions but with 50 agents, the search will evaluate a maximum of 100 entries (5000/50). Of course, if we consider an increasing number of agents, this also means that they produce more events in less time, but it also means that given the same number of changes applied to the environment, the environment responds better with an increasing number of agents. Thus, the environment as supported by GOLEM scales up better in situations when there are many agents rather than few.

In the second series of experiments we distributed the OPW grid (40x40) first into two containers (20x40) and then into four (20x20) different containers. For the distribution of the containers we used an Intel Centrino Core 2 Duo 2.66GHz with 4GB of RAM and an Intel Centrino Core Duo 1.66Ghz with 1GB of RAM. The agents were deployed between the distributed containers and could move from one machine to another by means of the mobility capabilities offered by GOLEM [3]. Fig. 3 shows what happens when we distribute the environment in multiple containers and use AEC to link these containers.

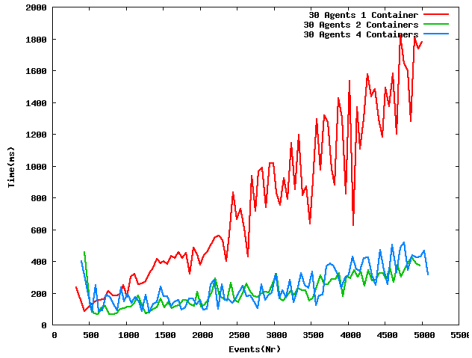


Figure 3: The effects of distribution

As shown in Fig. 3, with a growing number of events if we increase the number of containers, we improve considerably the performance. In Fig. 3 we show that in a system with a small number of events (0-500), it is better to compute the physical and

social state in one container. With a bigger number of events, the experiment shows that we can achieve a big difference in performance if we distribute in two or four containers instead of one container.

In the distributed version the size of the grid managed by a single container becomes smaller and less complex terms (agents, packets and destinations) are registered in a single container. Between 500 to 3500 events, in average, having four or two containers does not make much difference. However, after 3500 events the performance of the application with two containers is better from the performance of the application deployed in four containers. This is due to the fact that with less packets on the grid (most of them after 3500 events have been delivered to the destinations), the agents moving on the grid are more likely to change containers in search for packets. The smaller the grid, the bigger the number of times agents try to move from a container to another. This introduces a distribution cost related to the cost of interactions between containers. For this reason, in the presented experiments there is no improvement when we change from two to four containers.

In general, the time to compute the physical and social state distributed over many containers is defined by the

equation:

$$T_d = \frac{T_c}{d} + i \times c$$

where T_c is the time to compute the same experiment with a centralised container, d is number of containers used in the decentralized version, i is the number of interactions between containers and c is the cost of container interaction. In other words, when we distribute the agent environment in multiple containers, the time to compute the physical and the social state is inversionally proportional to the number of containers, thus improving the performance. However, there is an additional delay to compute the physical and social state which is due to the interactions between the containers.

5. RELATED WORK

There exist several approaches in the literature for executable specifications of norm-governed systems. Consider, for instance, the 'Law-Governed Interaction' (LGI) [23, 22] framework that has been used to regulate distributed systems. The Moses software mechanism [21] is an implementation of LGI that employs regimentation devices monitoring the behaviour of agents, blocking the performance of forbidden actions and enforcing compliance with obligations. Laws in Moses are written in pure Prolog or Java.

A tool for computational support concerning 'e-institutions' [7, 8, 9] is presented in [13]. This is a rule-based system for executing a set of 'normative rules' — expressions which impose obligations or prohibitions on communicative actions — with the aim of providing run-time services, such as the computation of the permissions and obligations of the agents current at each state.

Several action languages and corresponding software tools have also been employed for specifying and executing norm-governed systems. Fox et al. [12], for example, utilised an automated reasoning tool to execute 'organisational rules' formalised in the Situation Calculus [25]. Fardell et al, [10], propose a formalisation of the Event Calculus in XML and aplly it to the representation of contracts to facilitate the automated tracking of the contract state. Commitment protocols [6, 11] have been formalised in, among others, the action language $C+$ [14] and various dialects of the Event Calculus. Moreover, the Causal Calculator implementation of $C+$, and the Discrete Event Calculus reasoner [24] have been employed to execute commitment protocols.

Recently, norm-governed systems specifications have been formalised in semantic web languages [27, 17]; furthermore, various automated reasoning tools have been utilised for executing the specifications.

Our logic programming implementation of the Event Calculus has the following benefits. First, it exhibits a declarative semantics whose advantages, compared to procedural semantics, have been well-documented. Second, the Event Calculus offers a formal representation of the agents' actions and their effects. This is in contrast to semantic web languages that offer limited temporal representation and reasoning. Third, the availability of the full power of logic programming, which is one of the main attractions of employing the Event Calculus as the temporal formalism, allows for the development of very expressive social and physical laws. Fourth, we do not have to know from the outset the domain of each variable. Fifth, the OEC and the AEC versions used here provide an efficient and scalable reasoning mechanism, offering the kind of run-time support that is required for norm-governed multi-agent systems. The last point differentiates our work from approaches offering computational support for norm-governed systems. The last three points differentiate our work from other action language implementations.

6. CONCLUSIONS AND FUTURE WORK

We presented a knowledge representation framework with an associated run-time infrastructure that is able to compute, for the benefit of the members of a norm-governed multi-agent system, physically possible and permitted actions current at each time, as well as sanctions that should be applied to violations of prohibitions. The presented infrastructure is highly configurable in the sense that it can be appropriately distributed to offer run-time support for large-scale norm-governed systems.

There are several directions for further work. First, we are examining various caching mechanisms for the Event Calculus, such as those proposed in [5], in order to further improve the efficiency of temporal reasoning. Second, we aim to perform experiments with larger multi-agent systems in order to determine the extent to which our infrastructure can be used for run-time support. Third, we aim to formalise additional normative relations, such as institutional power.

7. REFERENCES

[1] Alexander Artikis, Marek J. Sergot, and Jeremy V. Pitt. Specifying norm-governed computational societies. *ACM Trans. Comput. Log.*, 10(1), 2009.

[2] Stefano Bromuri and Kostas Stathis. Situating Cognitive Agents in GOLEM. In *Engineering Environment-Mediated Multi-Agent Systems, EEMMAS 2007*, volume 5049/2008 of *Lecture Notes in Computer Science*, pages 115–134. Springer, 2007.

[3] Stefano Bromuri and Kostas Stathis. Distributed Agent Environments in the Ambient Event Calculus. In *DEBS '09: Proceedings of the third international conference on Distributed event-based systems*, New York, NY, USA, 2009. ACM.

[4] W. Chen and D. S. Warren. C-logic of complex objects. In *PODS '89: Proceedings of the eighth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 369–378, New York, NY, USA, 1989. ACM.

[5] Luca Chittaro and Angelo Montanari. Efficient temporal reasoning in the cached event calculus. *Computational Intelligence*, 12:359–382, 1996.

[6] A. Chopra and M. Singh. Contextualizing commitment protocols. In *Proceedings of Conference on Autonous Agents and Multi-Agent Systems (AAMAS)*, pages 1345–1352. ACM, 2006.

[7] M. Esteva, D. de la Cruz, and C. Sierra. Islander: an electronic institutions editor. In C. Castelfranchi and L. Johnson, editors, *Proceedings of the First International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 1045–1052. ACM Press, 2002.

[8] M. Esteva, J. Padget, and C. Sierra. Formalizing a language for institutions and norms. In J.-J. Meyer and M. Tambe, editors, *Intelligent Agents VIII*, LNAI2333, pages 348–366. Springer-Verlag, 2002.

[9] M. Esteva, J. Rodríguez-Aguilar, C. Sierra, and W. Vasconcelos. Verifying norm consistency in electronic institutions. In *Proceedings of the AAAI-04 Workshop on Agent Organizations: Theory and Practice*, pages 8–14, 2004.

[10] Andrew D. H. Farrell, Marek J. Sergot, Mathias SallÃl', and Claudio Bartolini. Using the event calculus for tracking the normative state of contracts. *International Journal of Cooperative Information Systems*, 14:99–129, 2005.

[11] N. Fornara and M. Colombetti. *Multi-Agent Systems: Semantics and Dynamics of Organizational Models*, chapter Formal specification of artificial institutions using the event calculus. IGI Global, 2009.

[12] M. Fox, M. Barbuceanu, M. Grüninger, and J. Lin. An organizational ontology for enterprise modeling. In M. Prietula, K. Carley, and L. Gasser, editors, *Simulating Organizations: Computational Models for*

Institutions and Groups, pages 131–152. AAAI Press/The MIT Press, 1998.

[13] A. García-Camino, P. Noriega, and J. Rodríguez-Aguilar. Implementing norms in electronic institutions. In *Proceedings of the Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 667–673. ACM Press, 2005.

[14] E. Giunchiglia, J. Lee, V. Lifschitz, N. McCain, and H. Turner. Nonmonotonic causal theories. *Artificial Intelligence*, 153(1-2):49–104, 2004.

[15] A. Jones and M. Sergot. On the characterisation of law and computer systems: the normative systems perspective. In *Deontic Logic in Computer Science: Normative System Specification*, pages 275–307. J. Wiley and Sons, 1993.

[16] A. Jones and M. Sergot. A formal characterisation of institutionalised power. *Journal of the IGPL*, 4(3):429–445, 1996.

[17] L. Kagal and T. Finin. Modeling communicative behavior using permissions and obligations. *Journal of Autonomous Agents and Multi-Agent Systems*, 14(2):187–206, 2006.

[18] F. Nihan Kesim and Marek Sergot. A Logic Programming Framework for Modeling Temporal Objects. *IEEE Transactions on Knowledge and Data Engineering*, 8(5):724–741, 1996.

[19] Nihan Kesim and Marek Sergot. Implementing an object-oriented deductive database using temporal reasoning. *J. Database Manage.*, 7(4):21–34, 1996.

[20] Robert Kowalski and Marek Sergot. A logic-based calculus of events. *New Gen. Comput.*, 4(1):67–95, 1986.

[21] N. Minsky. *Law-Governed Interaction (LGI): A Distributed Coordination and Control Mechanism (An Introduction and a Reference Manual)*, 2005. Retrieved October 24, 2008, from `http://www.moses.rutgers.edu/documentation/manual.pdf`.

[22] N. Minsky. Decentralised regulation of distributed systems: Beyond access control, 2008. Submitted for publication. Retrieved October 24, 2008, from `http://www.cs.rutgers.edu/~minsky/papers/IC.pdf`.

[23] N. Minsky and V. Ungureanu. Law-governed interaction: a coordination and control mechanism for heterogeneous distributed systems. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 9(3):273–305, 2000.

[24] E. Mueller. *Commonsense Reasoning*. Morgan Kaufmann, 2006.

[25] J. Pinto and R. Reiter. Temporal reasoning in logic programming: a case for the situation calculus. In D. Warren, editor, *Proceedings of Conference on Logic Programming*, pages 203–221. MIT Press, 1993.

[26] Jeremy Pitt, Abe Mamdani, and Patricia Charlton. The open agent society and its enemies: a position statement and research programme. *Telematics and Informatics*, 18(1):67–87, 2001.

[27] G. Tonti, J. Bradshaw, R. Jeffers, R. Montanari, N. Suri, and A. Uszok. Semantic web languages for policy representation and reasoning: A comparison of KAoS, rei and ponder. In D. Fensel, K. Sycara, and J. Mylopoulos, editors, *Proceedings of Second International Semantic Web Conference*, volume LNCS 2870, pages 419–437. Springer, 2003.

[28] Danny Weyns, Alexander Helleboogh, and Tom Holvoet. The packet-world: A testbed for investigating situated multiagent systems. In *Software Agent-Based Applications, Platforms, and Development Kits*, pages 383–408. Birkhauser Verlag, Basel - Boston - Berlin, 2005.

Norm Adaptation using a Two-Level Multi-Agent System Architecture in a Peer-to-Peer Scenario

Jordi Campos
Universitat de Barcelona
585 Gran Via
08007 Barcelona, Spain
jcampos@maia.ub.es

Maite López-Sánchez
Universitat de Barcelona
585 Gran Via
08007 Barcelona, Spain
maite@maia.ub.es

Marc Esteva
IIIA - CSIC
Campus UAB
08193 Bellaterra, Spain
marc@iiia.csic.es

ABSTRACT

Existing organisational centred multi-agent systems (MAS) regulate agents' activities. Nevertheless, population and/or environmental changes may lead to a poor fulfilment of the system's purposes, and therefore, adapting the whole organisation becomes key. This is even more needed in open MAS, where participants are unknown beforehand, they may change over time, and there are no guarantees about their behaviours nor capabilities. Hence, in this paper we focus on endowing an organisation with self-adaptation capabilities instead of expecting agents to increase their behaviour complexity. We regard this organisational adaptation as an assisting service provided by what we call the *Assistance Layer*. Our abstract Two Level Assisted MAS Architecture (2-LAMA) incorporates such a layer. We empirically evaluate our adaptation mechanism in a P2P scenario by comparing it with the standard BitTorrent protocol. Results provide a performance improvement and show that the cost of introducing an additional layer in charge of system's adaptation is lower than its benefits.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems, Coherence and coordination*

General Terms

Design, Experimentation, Performance

Keywords

Adaptation, Norms, Coordination, Organisation, MAS

1. INTRODUCTION

Developing Multi Agent Systems (MAS) entails the problems of designing a distributed concurrent system plus the difficulties of having flexible and complex interactions among autonomous entities [1]. Organising such systems to regulate agent interactions is a practise that helps to face their complexity [2]. Specially in open MAS, since agents are developed by third-parties, so they may enter or leave the system at any moment and there are no guarantees about their behaviour. To face the derived complexity, some approaches [3, 4] use organisation entities as regulative structures. Such an organisation helps designers to predict/regulate the system evolution within certain bounds. The fact that these structures persist with independence of their participants reinforces their role as first-order entities. Moreover, these approaches usually provide an infrastructure to support the enactment of these entities —to create them, to store their specifications, to check if participants fulfil them, etc. In fact, these approaches provide an organisational framework to agents, which minimises the number of possibilities they have to face. This is because agents can construe other participant's behaviour under a certain context.

As we previously mentioned, an organisational structure helps to regulate MAS. However, certain environmental or population changes may decrease its performance to achieve goals. Thus, adapting such an organisation is an important topic [5, 6, 7, 8], since it can help to obtain the expected outcomes under changing circumstances. This is motivated by the computational organisational theory, which claims that the best organisation designs are domain and context dependent [9]. Adaptation can be seen as a reconfiguration aspect of autonomic computing, where a MAS is able to reconfigure itself [10].

Concerning such an adaptation, we propose to add a *meta-level* in charge of adapting system's organisation instead of expecting agents to increase their behaviour complexity. This is specially relevant when dealing with open MAS, since there is no control over participant's implementation. Hence, we cannot expect agents to be endowed with the necessary mechanisms to adapt the organisation when it is not achieving its goals. We regard this adaptation –together with other possible meta-level functionalities– as an assistance to agents that can be provided by MAS infrastructure. Thus, we call our approach Two Level Assisted MAS Architecture (2-LAMA). In order to avoid centralisation limitations such as fault-tolerance or global information unavailability, we propose a distributed *meta-level* composed of several agents. This paper is focused on 2-LAMA's organisational adaptation capabilities. In particular, it focuses on *norm adaptation* —we assume norms are an organisational regulative structure.

Our approach requires domains with organisations that can be dynamically changed. Besides, it is able to deal with highly dynamic environments and even with domains where there is no direct mapping between goals and the tasks required to achieve them —i.e. it is not possible to derive a set of tasks that achieve a certain goal. As an illustration, we present a Peer-to-Peer sharing network (P2P) as a representative case study. In such a network, computers interact to share some data. Furthermore, their relationships change over time depending on network status and participants. We use this scenario to perform an empiric evaluation and compare our approach with existing BitTorrent protocol [11].

Our general model and its application are described in sections 2 and 3. Further, the adaptation process is detailed in section 4. Next, it is compared with BitTorrent in section 5 and with related work in section 6. Finally, section 7 presents the derived conclusions.

2. GENERAL MODEL

Previous section identifies organisations as useful entities to regulate agents' behaviours and facilitate their coordination. In particular, these entities provide a framework that is useful for agent coordination. Besides, there are MAS infrastructures that provide some organisational-related features as domain independent services. Thus, we regard them as *Coordination Support* services [12] that alleviate agent development. These services also include basic coordination elements such as elemental connectivity or agent communication languages. In brief, all these services are devoted to enact agent coordination. In addition to that, we propose an extra set of services that provides an added value by assisting coordination. We propose to add an *Assistance Layer* on top of a regular system in order to provide such coordination assistance services The main contribution of this paper is the proposal of a distributed pro-active service at the Assistance Layer that adapts organisations depending on the system's evolution.

Before provinding an insight into this organisational adaptation service, we detail how we model an organisational structure itself. Usually, organisation-centred MAS provide services that range from establishing the basis for agent communication through individual messages to providing organisational structures. We denote one of those organisations as: $Org = \langle SocStr, SocConv, Goals \rangle$, its compoments are detailed next. It has a social structure ($SocStr$) consisting of a set of roles (Rol) and their relationships (Rel). In addition, it has some social conventions ($SocConv$) that agents should conform and expect others to conform. They are expressed as interaction protocols ($Prot$) and/or norms ($Norms$). In more detail, protocols define legitimate sequences of actions performed by agents playing certain roles. Whereas norms delimit agent actions by expressing related permissions, prohibitions or obligations. Notice, that in our case study, the only possible actions are message physical exchanges among agents. Finally, it has some goals ($Goals$) that describe the organisation design purpose —they may differ from participant's individual ones. These goals are expressed as a function over the system's observable properties —it may include the reference values they should approach. This way, system performance can be evaluated by using these goals to determine in which degree the system is fulfilling its design objectives.

2.1 Assistance Layer

The Assistance Layer we propose, provides an assistance that may facilitate the enrolment of third-party agents and/or adapt their organisation. This layer provides two main types of services [12]: assisting individual agents to achieve their goals following current social conventions (Agent Assistance); and adapting social conventions to varying circumstances (Organisational Assistance). The former includes services to inform agents about useful information to participate in the MAS (Information service), to provide justifications about the consequences of their actions (Justification service), to suggest alternative plans that conform social conventions

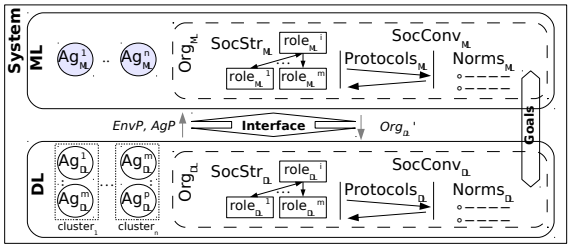


Figure 1: Two Level Assisted MAS Architecture (2-LAMA).

(Advice service) and to estimate the possible consequences of certain actions due to current conventions (Estimation service). The latter, the Organisational Assistance, consists on adapting existing organisations to improve system's performance under varying circumstances. To provide such an adaptation, we propose goal fulfilment as its driving force within the context of a rational world assumption. Hence, the Assistance Layer requires some way (i) to observe system evolution, (ii) to compare it with the organisational goals and (iii) to adapt the organisation trying to improve goal fulfilment. See [12] for further details about all enumerated services.

In order to provide Assistance Layer services, we proposed a Two Level Assisted MAS Architecture (2-LAMA, [13]). The bottom level, we call it domain-level (DL), is composed by agents carrying out domain activities regulated by an organisational structure. On top of it, there is a distributed meta-level (ML) also composed by agents and an organisational structure targeted to provide assistance services to domain-level agents. In between, there is an interface (Int) that communicates both levels as shown in Figure 1. Thus, the whole system can be expressed as: $2LAMA = \langle ML, DL, Int \rangle$[1]. Each level has an organised set of agents so they are respectively defined as $ML = \langle Ag_{ML}, Org_{ML} \rangle$ and $DL = \langle Ag_{DL}, Org_{DL} \rangle$. Using the interface, the meta-level can perceive environment observable properties ($EnvP$, e.g. date or temperature) and agents observable properties (AgP, e.g. colour or position). Specifically, we assume each meta-level agent ($a_{ML} \in Ag_{ML}$) has partial information about them, so it only perceives a subset of $EnvP$ and AgP —in many scenarios global information is not available. In fact, a a_{ML} has partial information about the subset of domain-level agents it assists. We call this subset of agents a *cluster*, which would be grouped according to a domain criterion —e.g. they could be grouped because interactions among them have lower costs than with other agents. However, an assistant can share part of this information with other meta-level agents in order to provide better assistance services.

3. 2-LAMA IN A P2P SCENARIO

Our case study is a Peer-to-Peer sharing network (P2P), where a set of computers connected to the Internet (peers) share some data. We apply our model to this scenario because it is a highly dynamic environment due to the very

[1]In fact, it is possible to nest subsequent meta-levels that update previous level's organisation.

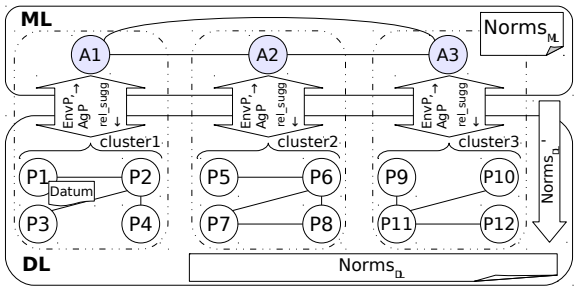


Figure 2: 2-LAMA in the P2P scenario.

Phase	Level	Protocol Messages
initial	Int	`join<hasDatum>`
latency	Int	`get_lat<peers>, lat<peer><measure>`
	DL	`lat_req, lat_rpl`
soc.struct.	Int	`contact<peers>`
handshake	DL	`bitfield<hasDatum>`
share data	DL	`rqst,data,cancel,have,choke,unchoke`
	Int	`complete, has_datum<peer>`
	ML	`all_complete, complete_peer<peer>`
norms	ML	`norm_bw<value>, norm_friends<value>`
	Int	`norm_updated<norm_id><new_def>`

Table 1: Protocol messages grouped into subsequent phases and involved levels —only domain-level (DL), only meta-level (ML) or both (Int).

nature of the Internet communications. We regard the *overlay network*[2] of current contacted peers as its organisational social structure, which is dynamically updated. Finally, this scenario allows the addition of some *norms* to regulate communications. Overall, it lets us apply our organisational and adaptive autonomic approach.

The performance in this scenario is evaluated in terms of time and network consumptions during the sharing process. Thus, we can define as global goals the minimisation of such measures so that the faster the data is obtained and the less network is consumed, the better for the users. Notice, though, that there is a trade-off between time and network usage. Therefore, although a peer can potentially contact any other peer, it usually contacts just a subset in order to consume less network resources —i.e. overlay network.

Real P2P networks are highly complex, so we try to reduce complexity by assuming some simplifications about the protocol and the underlying network. Specially, we assume information is composed of a single piece of data — accordingly, we say a peer is *complete* when it has that single piece. The rest of this section provides the details of the actual scenario and our 2-LAMA approach applied to it.

3.1 Architecture in P2P

We model the P2P scenario as a MAS where computers sharing data are participant agents within the domain-level (Ag_{DL}= $P1 \dots P12$). All of them play a single role Rol_{DL} = $\{peer\}$ within the domain-level organisation (Org_{DL}) — see Figure 2. In addition, we define a type of relationship called *contact* between two agents playing the role peer. Thus, as all agents in domain-level play the role peer, they can establish contact relationships at run-time. These actual relationships form the overlay network mentioned previously. In our model, the meta-level can suggest changes in this net of relationships (rel_sugg) taking into account the system's status. Regarding social conventions, peers use the sharing protocol ($Prot_{DL}$) specified below and two norms $Norm_{DL} = \{normBW_{DL}, normFR_{DL}\}$. First *norm* ($normBW_{DL}$) limits agents' network usage in percentage of its nominal *bandwidth*[3]. This norm can be expressed as: $normBW_{DL}$ ="a peer cannot use more than $\texttt{max}_{\texttt{BW}}$ bandwidth percentage to share data". This way, it prevents peers from massively using their bandwidth to send/receive data to/from all other peers. Second norm ($normFR_{DL}$) limits the number of peers to whom a peer can simultaneously send the data. Analogously to previous norm, we define $normFR_{DL}$ ="a peer cannot simultaneously send the data to more than $\texttt{max}_{\texttt{FR}}$ peers". The last compoment of domain-level's organisation is its goal ($Goals$). This is that all peers –i.e. all computers sharing data– have the data as soon as possible using the minimal network resources. Thus, given some time cost (c_t) and network cost (c_n) metrics, we can define a global goal function that minimises a weighted combination of them: $Goals = min(w_t \cdot c_t + w_n \cdot c_n)$, where ($w_t$,$w_n$) are the corresponding weights that represent the relative importance of each measure.

In order to provide assistance to the domain-level, we add the meta-level on top of it. This *meta-level* also has a single role $Rol_{ML} = \{assistant\}$. Each agent in Ag_{ML}= $A1 \dots A3$ assists a disjoint subset of domain-level agents (cluster$\subset$ Ag_{DL}). It does it so by collecting information about them –about agents or their environment– and adapting their local organisation. Its decisions are based on local information about its associated cluster, aggregated information about other clusters –provided by other assistants— and the norms at their level ($Norm_{ML}$). Some examples of local information are latencies ($EnvP$) or which agents have the data (AgP). Information about other clusters come from other assistants —notice that meta-level agents have their own social structure too. Regarding meta-level norms, we consider one that limits the number of domain-level agents to inform about another domain-level agent having the data. More precisely, when an assistant receives the information that one agent in another cluster has become complete, the number of domain-level agents in its cluster it can inform to is limited. In particular, the norm is expressed as $normHas_{ML}$ ="upon reception of a complete agent (agent $\notin$ cluster) message, inform no more than $\texttt{max}_{\texttt{Has}}$ agents $\in$ cluster". Finally, we assume assistants are located at Internet Service Providers (ISP) and thus related communications are fast.

3.2 Protocol

Our proposed protocol is a simplified version of the widely used BitTorrent [11] protocol. Table 1 lists all its messages,

[2]An overlay network is a network build on top of another one. In the P2P scenario, the base network that connects all peers is the Internet. Then, the network of peers that are really interacting among them is an overlay network on top of the Internet.

[3]The *bandwidth* is the capacity to transfer data over user's network connection. It is expressed as the number of data units that can traverse a communication channel in a time unit. The less is used by the peer, the more is left for other purposes.

which follow the sequence detailed next. At the beginning, a domain-level agent (peer) initiates a handshake phase with another one by sending it a `bitfield <hasDatum>` message. `<hasDatum> = [1/0]` indicates if it has (1) or has not (0) the data —i.e. it is a *complete* or *incomplete* agent. Notice that in current implementation, the data has only a single piece. In turn, the other agent finishes this handshake phase by replaying with another `bitfield <hasDatum>` message to indicate its status. In case one of these agents have the datum and the other lacks it, the later sends a `rqst` (request) message to the former. Then, the former replies with a message containing the datum. On the contrary, if none of the agents have the datum they will not exchange further messages. However, as soon as one agent receives the datum, it will send a `have` message to these other contacted agents to let them know that its status has changed. In such cases, if they still lack of the datum, they will request it. Additionally, an agent may reply to a request with a `choke` message if it is already serving $\texttt{max}_{\texttt{FR}}$ agents —it means this agent is going to ignore any further message. Later on, when a transmission ends, it sends `unchoke` messages to all choked agents, so they can request the datum again. On the other hand, a requester agent is allowed to get data from two sources simultaneously. This is done –for a short time– in order to compare their effective bandwidth so to choose the fastest source (the other one is discarded with a `cancel` message).

Previous messages are related to communication at domain-level. However, there are other messages related to communication at *meta-level* and among levels. Initially, a new domain-level agent sends its `join <hasDatum>` message to the closest assistant —a domain-level agent measures it latency to all assistants and chooses the one having the smallest latency. Then, the assistant asks the agent to measure its latencies with all other agents in its cluster by sending a `get_lat <peers>` message. The agent measures latencies by exchanging `lat_req`/`lat_rpl` messages, and informs back the assistant with a `lat <measure>` message. Once an assistant has all latencies among its domain-level agents ($EnvP$) and knows which ones have the datum, it estimates which would be the best social structure —see [13]. Then it suggests the agent relationships by sending `contact <peers>` messages to all the agents in its cluster.

Additionally, when a domain-level agent receives the datum, it also informs its assistant with a `complete` message. Then, at meta-level this assistant informs other assistants with a `complete_peer <peer>` message. For instance, in Figure 2, when P2 receives the datum, it informs A1, which will inform A2 and A3. Next, contacted assistants spread this information towards their domain-level agents –limited by $\texttt{max}_{\texttt{Has}}$– with a `has_datum <peer>` message —e.g. A2 may inform P6 and P8 that P2 has the datum, if $\texttt{max}_{\texttt{Has}} = 2$. In that moment, informed agents measure their latencies to the new agent and request it, if it is better than any previous source. Finally, when an assistant detects that all domain-level agents in its cluster are complete, it sends an `all_complete` message to other assistants to avoid receiving more `complete_peer` notifications.

Last, the norm adaptation process requires some more messages —see section 4. When an assistant wants to update $normBW_{DL}$, it sends a `norm_bw <value>` message to the rest of assistants. Analogously, it would send a `norm_friends <value>` in case of a $normFR_{DL}$ update. Then, when a new value is finally agreed, each *assistant* informs its the domain-level agents in its cluster with a `norm_updated <norm_id> <new_def>` message.

4. ORGANISATIONAL ADAPTATION

Within our 2-LAMA architecture, the meta-level is able to adapt domain-level's organisation. In particular, we are working on social structure and norm adaptation. The former consists in the meta-level updating domain-level's overlay network as detailed in [13]. The latter is the focus of this paper, and it is described in this section. In brief, norm adaptation proceeds as follows. Initially, assistants collect status information from their cluster domain-level agents but also from other assistants —in a summarised form. Afterwards, they aggregate all this information. Next, they compute their desired values of norm parameters depending on this aggregated information. Finally, they use a voting scheme as a group decision mechanism to choose the actual norm updates before notifying their agents.

The underlying rationale of the norm adaptation process is to align the amount of served data with the amount of received data. Thus, the information collected by each assistant consists of some measures about the agents serving the datum and the ones that lack it. Specifically, they collect the following information:

- `srvBW`: the sum of the nominal bandwidths of the individual channels of the agents that are serving data.
- `rcvBW`: the sum of the nominal bandwidth of the individual channels of the agents that are receiving data.
- `rcvEffBW`: the sum of the effective receiving bandwidth of the agents that are receiving data. It can be smaller than `rcvBW` when only a few data is served or there is network saturation that delays message transport.
- `rcvExpBW`: the expected receiving bandwidth. It is estimated using the nominal one (`rcvBW`) re-scaled by current bandwidth limit ($\texttt{max}_{\texttt{BW}}$). It is computed to be compared with `rcvEffBW`. If effective serving bandwidth is limited by a $\texttt{max}_{\texttt{BW}} < 100$, the reference receiving bandwidth may be lesser than the nominal one (`rcvBW`) —since less data is being injected towards receiving agents.
- `waiting`: the number of agents that do not have the datum and are neither receiving it.

Such information could be collected by each assistant from its agents or by accessing network information. In the former case, assistants would query agents about such information. Thus, this method would require that domain-level agents would report true values —which would be difficult to guarantee in an open MAS. In contrast, we use the latter case, which does not require collaborative agents. In this method, assistants inspect domain-level agent communications to obtain such information by themselves —this requires assistants to have privileges to access network resources, which is acceptable if they are related to ISPs.

Depending on the cost of collecting such information, it may be retrieved continuously or at certain intervals. Also, depending on the cost of applying norm changes, the norm adaptation process may be performed at given intervals. In the current implementation, this process is performed at a

Algorithm 1 Adaptation algorithm used by *assistants*.

```
00 def adapt( srvBW, rcvBW, rcvEffBW,
01              waiting, maxFR, maxBW ):
02   τ = 0.1 ; ϵ = 0.2
03   rcvExpBW = rcvBW * (maxBW / 100)
04
05   // Adapt maxFR -------------
06   case (srvBW<(1-τ)*rcvBW)  : vFR=decr
07
08   case (srvBW>(1+τ)*rcvBW
09                && waiting>ϵ): vFR=incr
10
11   case (srvBW>(1+τ)*rcvBW
12                && waiting<ϵ): vFR=blnk
13
14
15   other /*srvBW ≈  rcvBW */: vFR=same
16
17
18   if(rcvEffBW<(1-τ)*rcvExpBW):vFR=decr
19
20   // Adapt maxBW ------------
21   case (vFR==decr ∧ maxFR==1  : vBW=maxBW/2
22   case (vFR==incr ∧ maxBW<100): vBW=100
23   other                      : vBW=maxBW
24
25   return [ vFR, vBW ]
```

fixed time interval ($\texttt{adapt}_{\texttt{interv}}$) with an average of these measures along it.

In order to compute the desired norms, an assistant weighs the information it has collected from its cluster with the information provided by other assistants. This way it can give more importance to local information. For instance, $\texttt{srvBW} = \texttt{w}_\texttt{L} \cdot \texttt{srvBW}_\texttt{L} + \sum(\texttt{w}_{\texttt{R}_\texttt{i}} \cdot \texttt{srvBW}_{\texttt{R}_\texttt{i}})$, where $\texttt{srvBW}_\texttt{L}$ stands for the local cluster's measure, $\texttt{srvBW}_{\texttt{R}_\texttt{i}}$ stands for the remote ones, $\texttt{w}_\texttt{L}$ stands for the weight of local information, and $\texttt{w}_{\texttt{R}_\texttt{i}}$ stands for the weight of remote one. Moreover, $\texttt{w}_\texttt{L} + \sum \texttt{w}_{\texttt{R}_\texttt{i}} = 1$ and $\nexists \texttt{w}_{\texttt{R}_\texttt{i}}, \texttt{w}_{\texttt{R}_\texttt{i}} > \texttt{w}_\texttt{L}$.

If the local weight is the maximum ($\texttt{w}_\texttt{L} = 1$), then each assistant computes desired norms taking into account only its cluster status. On the contrary, if this weight is the minimum ($\forall_i \texttt{w}_{\texttt{R}_\texttt{i}} = \texttt{w}_\texttt{L}$), then each assistant gives the same importance to local information as to remote one —this is the case in the current implementation. The mid-point is a local weight greater than any remote one ($\forall_i \texttt{w}_{\texttt{R}_\texttt{i}} < \texttt{w}_\texttt{L}$) such as an assistant takes its decisions giving more importance to its local cluster, but taking into account the rest of the system.

With this aggregated information each assistant computes its vote for $\texttt{max}_{\texttt{BW}}$ (vBW) and $\texttt{max}_{\texttt{FR}}$(vFR). In the case of vBW, the vote is the numeric desired value for $\texttt{max}_{\texttt{BW}}$. Whereas in vFR, the vote is an action among incrementing $\texttt{max}_{\texttt{FR}}$ by one (incr), decrementing it by one (decr), keeping the same value (same) or abstaining with a *blank ballot-paper* (blnk) to avoid influencing in new $\texttt{max}_{\texttt{FR}}$ value. They use the process schematised in Algorithm 1 to compute both votes. This algorithm receives the measures we described plus current norm parameter values. Next, in line 2, some constants are initialised to be used as thresholds in comparisons (their values were empirically tested). Then, the expected receiving bandwidth is computed from the nominal one re-scaled by current bandwidth limit (line 3).

The main decision to choose a $normFR_{DL}$ is related to compare the available bandwidth used to serve (srvBW) to the available bandwidth used to receive (rcvBW). If there is a lack of serving bandwidth (line 6), the suggestion is to decrease the number of friends. This way, server *agents* will be simultaneously serving data to fewer agents, and these transmissions will finish sooner. Afterwards, once these other agents get the datum, there will be more data sources in the system and it will take less time to finish the datum distribution. On the other hand, if there is an excess of serving bandwidth and there are still agents waiting for data (lines 8-9) then, the assistant can increase the number of friends in order to serve more agents. There is another situation in which there is also an excess of serving bandwidth but there are no agents waiting for data (lines 11-12). This does not necessarily mean all agents have the datum, but at least the ones lacking it are receiving it from some source. In this case, the assistant uses a blank-ballot paper to let other assistants push for their own interests[4].

Finally, if none of the previous cases is true, it means that the serving bandwidth is similar to the receiving one (line 15) then, the vote is for keeping the same norm. This is because if there is no excess of serving bandwidth, the assistant prefers to vote for the same norm instead of just leaving the decision to the rest of assistants.

Despite previous cases, if there is network saturation in the intermediate channels, it is always better to decrease the number of friends. This will reduce the number of data transmissions. Hence, it will cut back network traffic and hopefully network saturation. In order to estimate if there is network saturation, the assistant checks if the effective receiving bandwidth (rcvEffBW) is smaller than the expected one (rcvExpBW). This is a sign that data packets are delayed by the intermediate network because it is saturated. Consequently, as a solution to saturation, the assistant votes for decreasing $\texttt{max}_{\texttt{FR}}$ (line 18).

Regarding the $normBW_{DL}$, it is only decreased in case it is not possible to further diminish the network usage by decreasing the number of friends —since $\texttt{max}_{\texttt{FR}}$ is already 1. In such a case, the assistant votes for dividing $\texttt{max}_{\texttt{BW}}$ by 2 (line 21). This way, server agents will use less bandwidth, which can help to diminish the network saturation. On the contrary, if the bandwidth is previously limited but there is no network saturation –since the assistant chose to increase $\texttt{max}_{\texttt{FR}}$–, then the bandwidth limit can be established again back to 100% (line 22). For the remaining cases, $\texttt{max}_{\texttt{BW}}$ keeps its value (line 23).

After choosing a convenient value for each norm parameter, an assistant sends its votes (vFR, vBW) to the rest of assistants —see norm_bw and norm_friends messages. Then, when assistants receive all the votes, they compute the actual norm parameters. To conclude, they send to their domain-level agents the new norms using the norm_updated message. Notice that the average may provide the same norm parameters values as before, thus no changes would be performed —in practise, it means no update message would be sent. This situation may occur when opposite options are interesting for the same amount of clusters.

[4]Notice, though, that the weighting method applied to measures may bring an assistant to this case when no agents in its cluster are waiting for data, but there are still waiting agents in other clusters. In such a case, if there is enough serving bandwidth, it is better to let other assistants choose by themselves the norm parameter values.

Regarding norm updates application, once a domain-level agent receives new norms, it tries to fulfil them. Thus, when an agent receives a $normBW_{DL}$, it adapts its sending ratio and when it receives a $normFR_{DL}$ it also tries to fulfil it. This means that if an agent is serving to less friends than the new $\texttt{max}_{\texttt{FR}}$, it will send `unchoke` messages to those agents it has previously choked. This may result in new data requests that it will be able to serve. On the contrary, if it was serving to more friends than the new $\texttt{max}_{\texttt{FR}}$, it will cancel some of those data transmissions and send a `choke` message[5].

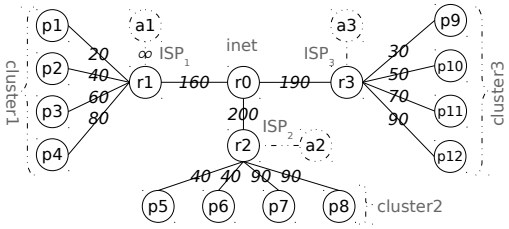


Figure 3: Network topology.

5. EMPIRICAL EVALUATION

In order to test our approach, we have implemented a P2P MAS simulator. This simulator is implemented in Repast Simphony [14] and provides different facilities to execute tests and analyse results. As it simulates both agents and network components, it allows to execute different sharing methods with identical populations and environmental conditions. Thus, we have performed several tests on BitTorrent and 2-LAMA to empirically evaluate the performance of our proposal.

5.1 Sharing methods

In this work, we compare three different approaches. A single-piece version of the BitTorrent protocol (BT) described in [15]. A 2-LAMA approach with social structure adaptation (2L.a) in which assistants update the actual contact relationships among domain-level agents as described in [13]. And a 2-LAMA approach with social structure and the norm adaptation (2L.b) described in this paper.

The BitTorrent implemented protocol (BT) among domain-level agents is very similar to 2-LAMA's since it inspired our approach. In order to make a fair comparison, we adapted BitTorrent to work with a single-piece datum —see [15] for further information. However, it does not have a distributed meta-level but a single agent (*Tracker*) that informs about connected agents. Consequently, agents do not receive any further assistance to share the datum. Instead, they use the algorithms described in [11]. In brief, the main algorithm of an agent having the datum consists in sending `choke` messages to all agents that are interested in it. Then, at certain intervals (`unchoke_interval`), the source agent sends `unchoke` messages to four of the previously choked agents. Next, these agents can request the datum and all of them are served. The selected agents to unchoke are those that were choked most recently. In case two of them were choked at the same time, the one having a larger network bandwidth (`upload_bw`[6]) is selected. In fact, if an agent's interest is older than a defined interval (`aging_period`), its age is ignored and only its agent's `upload_bw` is compared. In addition, in two out of three `unchoke_interval` selection processes, the fourth agent is randomly selected.

Regarding the configuration of our experiments, BitTorrent (BT) uses an `unchoke_interval` of 250 time units (ticks). It is approximately the time required to send four data messages along an average agent link in current topology. Thus, it is the average time that a server agent can invest sending data to four unchoked agents. This is the number of agents that BitTorrent protocol determines that an agent unchokes in an unchoke interval. Accordingly, they use an `aging_period` of 130 ticks to keep the ratio defined by the official protocol. On the other hand, the 2-LAMA experiments (2L.a, 2L.b) have been performed with the following initial norm parameters: $\texttt{maxHas} = \infty$, $\texttt{maxBW} = 100\%$, $\texttt{maxFR} = 3$. These norms lead 2-LAMA approach to a similar initial behaviour as BitTorrent because: $\texttt{maxHas} = \infty$ does not restrict communications among clusters, $\texttt{maxBW} = 100\%$ does not limit agent communication and $\texttt{maxFR} = 3$ is equivalent to the three non-random unchoked agents. This is specially the case because in our current implementation, domain-level agents always fulfil norms[7]. Additionally, for those tests including norm adaptation (2L.b), it has been done at an interval of $\texttt{adapt}_{\texttt{interv}} = 50$ time steps.

5.2 Results

In our experiments, we use a packet switching network model to simulate the transport of messages among agents. Figure 3 shows the network topology we use in our simulations. Notice that, as we are interested in having a different communication capacity for each domain-level agent, we place an *individual link* between each agent ($p1..p12$) and its corresponding Internet Service Provider ($ISP_1..ISP_3$ represented by routers $r1..r3$). In 2-LAMA experiments, each ISP has an associated assistant[8] ($a1..a3$) in charge of its connected domain-level agents. In addition, as we want to model simultaneous network usage by different agents, we place an *aggregated link* among each group of agents –i.e. a cluster, those connected to the same ISP– and the Internet ($r0$). In fact, in BitTorrent experiments, there are no assistants at all but a single tracker linked to this $r0$. Notice that the network topology influences the time required to transmit a message from one agent to another. In particular, this time depends on: message's length, the bandwidths of the traversed links, and the number of simultaneous messages traversing the same links —a link's bandwidth is divided among the messages that traverse it simultaneously. Regarding the former issue, we have used

[5]In the current implementation, an agent does not need to cancel a *friend* if it has already sent more than 75% of the datum to it. This behaviour avoids cancelling data transmissions that will finish really soon.

[6]In a multi-piece scenario, this measure is estimated from previous piece interchanges. However, since in a single-piece implementation no estimation can be performed, its value is taken from the network topology.

[7]Otherwise, we could assume there is an infrastructure mechanism at ISPs that detects and filters out messages that exceed the bandwidth limit (`maxBW`), or the simultaneous data messages limit ($\texttt{maxFR} = 3$).

[8]Our network model includes a quality of service (QoS) feature that gives more priority to messages among assistants or between assistants and domain-level agents. Thus, communications at meta-level and among levels are faster than communications at domain-level.

	time	*cNet*	*nHops*	*nData*	*cLat*	*cML*
BT	933.3	206182	3.4	11	0	0
2L.a	849.7	345060	3.2	40.1	21600	3749.9
2L.b	811.1	316190	3.0	30.7	21600	6596.0

Table 2: Results from BitTorrent (BT), 2-LAMA without norm adaptation (2L.a) and 2-LAMA with norm adaptation (2L.b).

the following message lengths: `piece` messages have 5000 data units, `lat_req` / `lat_rpl` have 150 data units and all the other control messages have a single data unit. Regarding the bandwidths links, Figure 3 shows them as numbers over the edges —we assume upload/download channels are symmetric. Finally, the latter issue, related to simultaneous link usage, is highly dynamic and depends on system's evolution.

We have tested all approaches in the described network topology by varying the agent that initially has the datum. Table 2 shows the results of different evaluation metrics in both approaches: BitTorrent (BT), 2-LAMA with social structure adaptation but no norm adaptation (2L.a) and 2-LAMA with social structure and norm adaptation (2L.b). Figures correspond to the average results for twelve different settings (so that they cover all possible initial datum positions in a single agent).

The evaluation metrics in Table 2 are the following: (1) *time* corresponds to the total time required to spread the datum among all agents; (2) *cNet* is the network cost consumed by all messages —each message cost is computed as its length times the number of links it traverses; (3) *nHops* is the average number of links traversed by each message; (4) *nData* is the total number of sent data messages —they may not be totally transmitted if: a destination agent sends a cancel message to its source or a source stops sending data to fulfil an updated $normFR_{DL}$; (5) *cLat* is the cost of all `lat_req/lat_rpl` messages; (6) *cML* is the cost of all messages related with the meta-level —i.e. all messages sent to or by *assistants*.

If we compare the performance of both approaches (BT and 2-LAMA), we see that our proposal requires less time to share the datum. Notice also, that 2-LAMA with social structure and norm adaptation (2L.b) presents shorter times that the version without norm adaptation (2L.a). In general, having better times in 2-LAMA means that the time invested in communicating with meta-level is less than the benefits of having such an additional level. Even more, we expect larger differences in performance when repeating the data sharing among the same P2P agent community since the information collected by our meta-level –e.g. measured latencies– will be used more than once. In fact, in our current 2-LAMA experiments, from 33 up to 56 ticks –depending on the cluster of agents– are invested in measuring latencies.

In contrast, the network cost (*cNet*) is larger in 2-LAMA, although norm adaptation (2L.b) provides the best performance again. Our proposal requires more communication because it initially measures latencies (*cLat*), it has extra communications due to the meta-level (*cML*), and it sends more data messages (*nData*). Specifically, latency measurements (*cLat*) represent up to a 20% of the network cost increment. This measurements are an initialisation phase that could be omitted in subsequent executions. On the other hand, 2-LAMA agents compare data sources by retrieving some data from them. This increases the number of data messages (*nData*) although most of them are cancelled. We expect to minimise this network consumption when dealing with more than one piece of data, since agents could compare sources depending on previous retrieved pieces. Regarding the number of links traversed by messages (*nHops*), our 2-LAMA approach has more local communications –i.e. intra-cluster– than BT. This is convenient because local messages have lower latencies and costs, since they are usually performed in the same cluster.

Overall, norm adaptation (2L.b) provides the best results despite requiring more assistant communication (*cML*). This stresses the idea that having a meta-level and exploiting its capabilities provides more benefits than the costs it causes.

6. RELATED WORK

Within MAS area, organisation-centred approaches regulate open systems by means of persistent organisations —e.g. Electronic Institutions [3]. Even more, several of these approaches offer mechanisms to update their organisational structures at run-time —e.g. Moise+ [4]. However, most work on adaptation maps organisational goals to tasks and look for agents with capabilities to perform them —e.g. OMACS [5]. Consequently, these approaches cannot deal with scenarios that lack of this goal/task mapping, like our case study. In order to deal with this sort of scenarios, our approach uses norms to influence agent behaviour, instead of delegating tasks. Specifically, our approach uses a norm adaptation mechanism based on social power —see norm taxonomy [16]. In this sense, there are other works that also use the leadership of certain agents (like our assistants) to create/spread norms —e.g. the role model based mechanism [17]. Besides, the most of norm emergence works are agent-centred approaches that depend on participants' implementation and they rarely create/update persistent organisations —e.g. infection-based model [18].

Relating norms and overall system behaviour, is a complex issue that increases its intricacy when there is no control over participant's implementation. In our approach, this task is distributed among a assistant agents which finally reach an agreement about norm updates. Currently, assistants use a voting scheme to agree on actual norms, but they could use some of the other agreement mechanisms present in literature —e.g. using an argumentation protocol [19]. Moreover, currently assistants use an heuristic to take their local decisions, but we are planning to use learning techniques in future work —like in AEI [20].

Regarding our P2P case study, there are some network management perspective approaches that also try to promote local communications but they cannot directly act on network consumption to balance net capacity and traffic —e.g. P4P [21] or ONO [22]. From a MAS angle, there are some works where agents adapt local norms using local information but they cannot reason/act at an organisational level —e.g. P2P normative system [23].

7. CONCLUSIONS

This work proposes an abstract MAS architecture (2-LAMA) to provide *assistance* to its participants. Particularly, this paper regards adapting a MAS organisation to varying cir-

cumstances as a type of assistance. It illustrates this approach in a P2P sharing network scenario, providing in-depth details about the adaptation process.

We endow the system with adaptation capabilities instead of expecting the agents to increase their behaviour complexity. Consequently, we propose to add a distributed *Assistance Layer* to improve system's performance by providing new support services to agents. In particular, in our architecture *meta-level* agents perceive information about MAS participants and environment, and are able to adapt the system's organisation.

Our 2-LAMA approach can be applied to domains with highly dynamic environments and no mapping between tasks and goals. It only requires that an organisation-centred MAS with an alterable organisation can be deployed. Such an organisation may include norms in its regulative structures. Moreover, the MAS can be open to third-party agents. As an illustration of all these issues, we introduce a representative case study based on a Peer-to-Peer sharing network. Additionally, to prove 2-LAMA's feasibility empirically, we have performed some experiments which show that the cost of adding our proposed Assistance Layer is lower than the obtained benefit. Specifically, 2-LAMA approach required less time than the original BitTorrent protocol. Even more, our approach results improved when increasing meta-level adaptation capabilities —i.e. when updating norms in addition to social structure adaptations.

As future work, we plan to confront further issues in open MAS such as how the system should react to agents joining or leaving the MAS anytime, or transgressing its organisational restrictions. In fact, we already have preliminary results about norm violations that show how system re-adapts to counter violation side effects. Besides, we are improving meta-level agents to use learning techniques in order to perform the adaptation process.

Acknowledgements: This work is partially funded by IEA (TIN2006-15662-C02-01), EVE (TIN2009-14702-C02-01 / TIN2009-14702-C02-02) and AT (CONSOLIDER CSD2007-0022) projects, EU-FEDER funds, the Catalan Gov. (Grant 2005-SGR-00093) and M. Esteva's Ramon y Cajal contract.

8. REFERENCES

[1] N. Jennings, K. Sycara, and M. Wooldridge, "A roadmap of agent research and development," *Autonomous Agents and Multi-Agent Systems*, vol. 1, no. 1, pp. 7–38, 1998.

[2] B. Horling and V. Lesser, "A survey of multi-agent organizational paradigms," *Knowl. Eng. Rev.*, vol. 19, no. 4, pp. 281–316, 2004.

[3] M. Esteva, *Electronic Institutions: from specification to development*, ser. IIIA PhD. Vol. 19, 2003.

[4] O. Boissier and B. Gâteau, "Normative multi-agent organizations: Modeling, support and control," in *Normative Multi-agent Systems*, 2007.

[5] S. A. Deloach, W. H. Oyenan, and E. T. Matson, "A capabilities-based model for adaptive organizations," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 1, pp. 13–56, 2008.

[6] R. Kota, N. Gibbins, and N. Jennings, "Decentralised structural adaptation in agent organisations," in *AAMAS Workshop on Organised Adaptation*, 2008.

[7] M. Sims, D. Corkill, and V. Lesser, "Automated Organization Design for Multi-agent Systems," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 2, pp. 151–185, 2008.

[8] C. Zhang, S. Abdallah, and V. Lesser, "MASPA: Multi-Agent Automated Supervisory Policy Adaptation," Tech. Rep. 03, 2008.

[9] K. Carley, "Computational and mathematical organization theory: Perspective and directions," *Computational & Mathematical Organization Theory*, vol. 1, no. 1, pp. 39–56, 1995.

[10] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *IEEE Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[11] BitTorrentInc., "BitTorrent protocol specification," 2001, http://www.bittorrent.org/beps/bep_0003.html.

[12] J. Campos, M. López-Sánchez, and M. Esteva, "Assistance layer, a step forward in Multi-Agent Systems Coordination Support," in *Autonomous Agents and Multiagent Systems*, 2009, pp. 1301–1302.

[13] Jordi Campos and Maite López-Sánchez and Marc Esteva, "Multi-Agent System adaptation in a Peer-to-Peer scenario," in *ACM SAC09 - Agreement Technologies*, 2009, pp. 735–739.

[14] M. North, T. Howe, N. Collier, and J. Vos, "Repast Simphony Runtime System," in *Agent Conference on Generative Social Processes, Models, and Mechanisms*, 2005.

[15] J. Campos, M. López-Sánchez, M. Esteva, A. Novo, and J. Morales, "2-LAMA Architecture vs. BitTorrent Protocol in a Peer-to-Peer Scenario," in *Artificial Intelligence Research and Development - CCIA09*, no. 202. IOS Press, 2009, pp. 197–206.

[16] B. S. S. Cranefield, "A categorization of simulation works on norms," 2009.

[17] B. S. S. Cranefield, M. Purvis, and M. Purvis, "Role model based mechanism for norm emergence in artificial agent societies," *Lecture Notes in Computer Science*, vol. 4870, p. 203, 2008.

[18] N. Salazar-Ramirez, J. A. Rodríguez-Aguilar, and J. L. Arcos, "An infection-based mechanism for self-adaptation in multi-agent complex networks," S. Brueckner, P. Robertson, and U. Bellur, Eds., 2008, pp. 161–170.

[19] A. Artikis, D. Kaponis, and J. Pitt, *Multi-Agent Systems: Semantics and Dynamics of Organisational Models*, 2009, ch. Dynamic Specifications of Norm-Governed Systems.

[20] E. Bou, M. López-Sánchez, and J. A. Rodríguez-Aguilar, *Autonomic Electronic Institutions' Self-Adaptation in Heterogeneous Agent Societies.* Springer, 2009, vol. 5368, pp. 18–35.

[21] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz, "P4P: provider portal for applications," 2008.

[22] D. Choffnes and F. Bustamante, "Taming the torrent: a practical approach to reducing cross-ISP traffic in peer-to-peer systems," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 363–374, 2008.

[23] A. Grizard, L. Vercouter, T. Stratulat, and G. Muller, "A peer-to-peer normative system to achieve social order," *LNCS*, vol. 4386, p. 274, 2007.

Generating New Regulations by Learning from Experience

Jan Koeppen
University of Barcelona
Gran Via 585 08007 Barcelona, Spain
janfrancisco@gmx.net

Maite Lopez-Sanchez
WAI, MAiA, University of Barcelona
Gran Via 585, 08007 Barcelona, Spain
maite_lopez@ub.edu

ABSTRACT

Both human and multi-agent societies are prone to best function with the inclusion of regulations. Human societies have developed jurisprudence as the theory and philosophy of law. Within it, utilitarianism has the view that laws should be crafted so as to produce the best consequences. Following this same objective, we propose an approach to enhance a multi-agent system with a regulatory authority that generates new regulations –norms– based on the outcome of previous experiences. These regulations are learned by applying a machine learning technique (CBR) that uses previous experiences to solve new problems. As a scenario to evaluate this innovative proposal, we use a simplified version of a traffic simulation scenario, where agents move within a road junction. Gathered experiences can then be easily mapped into regular traffic rules that, if followed, happen to be effective in avoiding undesired situations —and promoting desired ones. Thus, we can conclude that our approach can be successfully used to create new regulations for those multi-agent systems that accomplish two general conditions: to be able to continuously gather and evaluate experiences from its regular functioning; and to be characterized in such a way that similar social situations require similar regulations.

Categories and Subject Descriptors

I.2.11 [**Computing Methodologies**]: Distributed Artificial IntelligenceMultiagent systems

General Terms

Algorithms, Experimentation

Keywords

Normative systems, Learning, Agent based simulation, Self-organisation, Norm generation.

1. INTRODUCTION

Regulations have been proven to be useful in both human and multi-agent societies. Human societies use regulations within their legal systems. In fact, they have developed Jurisprudence as the theory and philosophy of law, which tries to obtain a deeper understanding of general issues such as the nature of law, of legal reasoning, or of legal institutions[1]. Within it, Normative Jurisprudence is concerned with normative or evaluative theories of law. It tries to answer questions such as "What is the purpose of law? or What sorts of acts should be subject to punishment?. Normative Jurisprudence has different schools. Among them, Deontology [7] can be described as an ethical theory concerned with duties and rights. On the other hand, Utilitarianism [12] takes the view that the laws should be crafted so as to produce the best consequences. When translating these approaches from human societies to MAS societies, it is obvious that a large number of simplifications have to be taken. Nevertheless, we think that it is still possible to keep and combine their fundamental objectives: to define specific prohibitions, permissions and obligations that promote desired overall system's behaviour for a given MAS society. Thus, the aim of this paper is to define a computational mechanism able to synthesize norms that succeed in the proper regulation of multi-agent societies[2].

We approach this regulation generation problem by learning from the experience of on-going activities within the MAS society. We have chosen Case-Based Reasoning (CBR) as the learning technique to apply. Briefly, CBR solves new problems –i.e., cases– by adapting the solution of similar problems from the knowledge base (which is a compound of solved problems). The selection of this learning technique is somehow inspired in the Anglo-American common law tradition, where judges use legal precedents to make decisions. Hence, using our terminology, we can interpret that judges re*solve* legal *cases* based on the way similar *cases* were previously re*solved*. More specifically, our approach defines a case as a compound of a problem –i.e., a social situation or context– and its associated solution, which in our case corresponds to the regulations that are applied in those contexts. In this manner, the overall learning objective becomes to define cases whose application leads to desired social situations. In CBR, problem description is key, and therefore, we have tested different problem representations that consider global and partial scopes. On the other hand, CBR is a supervised learning method that requires an expert to provide the system with correct problem solutions. Nevertheless, we want to generate best regulations without external knowledge, and thus, CBR cannot be directly applied. Instead, we propose to include an exploratory pseudo-random approach so that CBR becomes unsupervised.

Rather than by individual agents in the society, we assume learning to be performed by an independent regulatory authority within the MAS, able to observe and establish its

[1]Jurisprudence definition extracted from Black's Law Dictionary: http://www.blackslawdictionary.com

[2]We assume goals act as a reference that does not evolve.

norms. Therefore, we are taking an organizational centered perspective over the MAS as opposed to an agent-centered perspective. The underlying rationale is to restrict the focus of our research. An organizational point of view allows to have learning devoted to finding the best regulations for a whole society and to do it while interactions are taking place. On the contrary, taking an individual centered approach –where learning is performed by individual selfish agents– would also require considering additional aspects such as agreement, trust, uncertainty or communication.

The paper is structured as follows: next section introduces related work. Section 3 describes the tested scenario, section 4 details the learning process, and subsequent section 5 presents its empirical evaluation. Finally, some conclusions and future work are drawn in last Section 6.

2. RELATED WORK

Although Artificial Intelligence and Law have been related since a first article from McCarty [11], related research is not usually concerned with machine learning. This is less the case within the MAS area, where some learning techniques have been successfully applied. In fact, Multi-Agent Reinforcement Leaning [4] is quite widely used for individual agent learning. Nevertheless its usage is much more scarce for organizational centered approaches, where an exception is the work by Zhang et al.[19] devoted to improve system's organization. Our work uses CBR as an alternative learning technique, which is also based on system experience, but results in clearer knowledge representations —i.e., cases.

On the other hand, research on norms in multi-agent systems is a quite active area. Just to mention a few works: Boella and van der Torre have done relevant contributions [3] in norm characterization; Campos et al. [5] have proposed norm adaptation methods to specific network scenarios; Artikis et al.[2] have studied the definition of dynamic social conventions (protocols); and Savarimuthu et al. [16] as well as Kota et al. [10] work on norm emergence. Within this area, norm generation has been studied less frequently. Shoham and Tennenholtz [17] focus on norm synthesis by considering a state transition system: they explore the state-space enumeration and state it is NP-complete through a reduction from 3-SAT. Similarly, Hoek et al. [18] synthesize social laws as a model checking problem –again NP-Complete– that requires a complete action-based alternative transition system representation. In our case, CBR has the advantage that, although cases represent the search space, they do not need to be exhaustive, since they can be representatives of a set of similar problems requiring similar solutions. Furthermore, our approach is applied at run-time, being able to generate new norms during the execution of the system (this has the additional advantage of adapting to new situations). An intermediate approach is this of Christelis and Rovatsos [6], that synthesize generalized norms over general state specifications in planning domains. These domains allow for a local search around declarative specifications of states using planning AI methods. From our point of view, CBR allows the application to a wider range of domains, in particular to those where (i) experiences can be continuously gathered and evaluated, and where (ii) similar social situations require similar regulations (i.e., the continuity solution assumption).

Regarding implementation issues, it might be worth mentioning a related work on system monitoring by Modgil et al.[13] which is able to recognize norm compliance; and another one on traffic domain by Dunkel et al. [8] devoted to managing traffic systems. We have also used a simplified traffic scenario to test our innovative approach empirically.

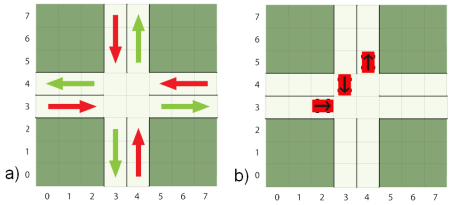


Figure 1: Orthogonal road junction: a) feeder and exit lines, b)traveling cars.

3. TRAFFIC SCENARIO

In order to test our learning approach, we have chosen a simplification of a traffic scenario. It has been developed as a multi-agent based simulation model in Repast [14]. This traffic scenario is an orthogonal two-road junction, where car agents travel along roads towards different destinations. As figure 1 shows, the environment has been discretized by means of a square grid whose cells have the size of a car. Gray (central) cells represent roads and green (corner) cells correspond to their surrounding non-transitable fields. Each road lane has a direction of traffic. Agents can join the road from four different entrance points –i.e., four incoming or feeder lanes (see left side of Figure 1)– and choose the exit point, so they decide the route to follow. Time, measured in ticks, is also discrete. Moreover, cars do have constant speed, so they can only move to adjacent cells in a single tick. Agent possible actions are stop, move forward, turn right, or turn left. Nevertheless, cars just turn in the intersection area and always obey the rules of right side traffic (i.e. they turn right in the first cell of the intersection whereas left turnings require to further traverse the junction and turn on the second cell). Furthermore, car agents also follow the social norms described in section 4 by stopping or moving whenever required.

4. NORM GENERATION THROUGH CASE-BASED REASONING

Multi-agent systems have been enriched with different regulations –norms, constraints, protocols, etc– with the aim of better organizing the society by restricting both individual behaviours and the way interactions are performed. In general, regulated societies build their norms as an implicit common agreement, assuming most of their individuals will respect them. Regulations can come from a norm emergence process or by having a regulatory authority dictating them. Furthermore, they can be created based on previous experiences or by anticipating situations that may appear. Nevertheless, since the number of possible outcomes of complex systems is so large, most societies regulate just those situations that have already occurred so far. This paper focuses on those regulations that can be established based on the experience of the regular functioning of MAS societies. We assume these societies have regulatory authorities that gather experiences in an on-going basis. Inspired in jurispru-

dence used in the Anglo-American common law tradition, we have enriched our MAS with a case based regulatory system. It is is charge of analyzing previous experiences and deciding what (if any) regulations should be applied for specific situation contexts in order to avoid undesired outcomes.

In order to do it, a regulatory authority must be able to first define the goals whose accomplishment guarantees system's performance or its overall desired behaviour. In our traffic scenario, the main goal is to minimize the number of collisions whilst keeping a fluid traffic. This is so because, obviously, if all cars stop, then there will be no collisions at all but cars will not accomplish their individual goals —which most probably will include reaching their destinations. Therefore, we are making an underlying assumption that is that social regulations should guarantee individuals to have enough autonomy so to accomplish their individual goals. Otherwise, punishments should be included to promote norm compliance. In summary, we can somehow interpret that the regulatory authority tries to guarantee basic common agreement about the norms it establishes.

Second, the regulatory authority must have the ability to observe the society in a way that it is able to identify undesired situations —that is, situations where goals are not being accomplished. In our traffic case, both collisions and blockages are main undesired situations.

Afterwards, the regulatory authority should be able to propose regulations that try to prevent undesired situations from being repeated in the future. Prohibitions should be done over those agents' actions that lead to undesired situations. Analogously, obligations can be used to promote desired actions. For example, if we consider our traffic junction, if there is a collision because two cars run on each other, then it is possible to propose a new regulation that prohibits cars to move when they happen to be in the same situation. On the other hand, if no collisions happen when cars traverse the junction it may be useful to create the obligation of keeping moving to prevent blockages. Obviously, deciding which actions should be prohibited or obliged is not a straightforward decision, and that is the reason we introduce automatic learning into the process.

Finally, whenever a new regulation is created and applied on the multi-agent system, the learning process requires the analysis of the consequences of its application. Thus, we need the regulatory authority to observe the society's evolution and to label the experience of applying this new regulation with its subsequent outcome. In this manner, regulatory knowledge is refined in an on-going basis.

The remaining of this section provides further details of our proposed approach. First subsection specifies the architecture of the MAS applied to the traffic scenario, and subsequent subsections detail the learning process.

4.1 Architecture

Following an organizational centered approach, we assume that the multi-agent system in our traffic scenario consists of a set of external agents that interact within a road environment together with a regulatory authority (see Figure 2). External agents play a car role; they are able to observe other car agents and to perform certain actions such as join, traverse, and leave the environment. Regarding the regulatory authority, its aim is to promote fluid car traffic flow with as few as possible collisions amongst traffic participants. This authority is constituted by staff permanent

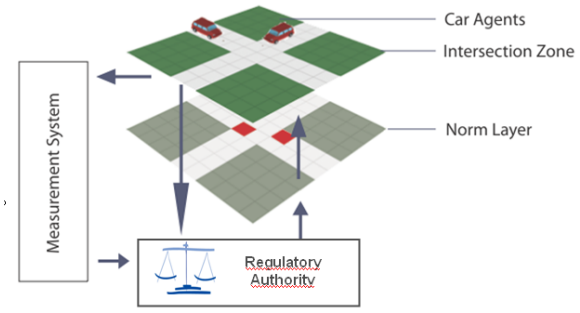


Figure 2: Traffic scenario architecture.

agents that perform regulation tasks. From those agents, we highlight the one in charge of defining current norms –we call it norm agent– and the one conducting the learning process —the CBR agent. Nevertheless, there are other staff agents that provide infrastructure services, such as the ones in the tracking system, in charge of obtaining information from the environment; the scene manager, in charge of runtime details; or the monitoring agents, which provide statistical analysis of the overall system operation.

The norm agent uses the regulatory knowledge from the CBR agent to specify the traffic rules that will be applied in the road environment. As a result, it updates a norm layer that is publicly available for the car agents so they become aware of the norms and can thus follow them. Agents conduct this norm updating process continuously, creating new norms when required or applying previously existing ones. The CBR agent will be the one in charge of taking this last decision. Next subsection details how it is performed.

4.2 Unsupervised CBR cycle

Case-based reasoning is a technique that solves new problems based on past experiences [1]. Experiences are stored in the form of cases, where a case is a description of a problem and its possible solution $Case = \langle probl, sol \rangle$. Cases are stored and maintained in a knowledge base (or case base) for further usage. Briefly, when a new problem is encountered (and thus, it lacks a solution), the CBR process searches for the most similar problem in the case base and adapts its associated solution to solve the current problem. The description of the target problem, together with the provided solution and related information about its performance, constitute a new case that can be in turn stored in the case base. Case performance –i.e., how well the derived solution solved the problem– depends on the continuity of the domain or, in other words, if for the domain it holds that similar problems require similar solutions. This overall process is usually explained in terms of what it is known to be the CBR cycle. It is characterized by four different steps: retrieve, reuse, revise and retain. Before describing them, it is worth mentioning that a case for us is composed of a traffic situation –car distribution–, the regulations –move /not move– that should be applied in such traffic context, and a case performance measure (see subsection 4.3 for further details).

Retrieve: Given a traffic situation description, we first retrieve from our knowledge base the case that is most relevant to solve it. Relevance here is interpreted as similarity,

and thus, we search for a case that describes the most similar traffic situation. More specifically, as we will see in next subsection 4.3 a case is considered to be similar to another if it represents the same number of cars and if these cars are located at rotationally equivalent cells. The retrieved case will include the regulations that were applied for its traffic situation and a score of its application.

Standard CBR systems are considered as supervised learning methods because they assume there is a pre-existing knowledge base, or that at least, a supervisor can provide solutions for new cases to be learned. Nevertheless, we face an unsupervised learning scenario, since we lack the necessary knowledge to determine the proper traffic rules that should be applied for specific situations. Therefore, it can well be the case that the retrieve phase does not provide any case. In fact, we encounter this situation right at the beginning, since we still lack experience. Hence, if no case has been retrieved, we need to somehow generate a new solution by exploring the space of possible solutions, which in our case means to try different combinations of traffic restrictions (norms). In our current implementation, exploration is performed by randomly assigning stopping/moving restrictions to those cells having cars (avoiding empty cells is an heuristic that prunes the search space). Furthermore, since this pseudo-random solution may not be optimal, we extend the cases to include several alternate solutions (generated in the same way) with a performance measure associated to each of them. The number of possible solutions is bounded in order to differentiate a learning phase –when alternate solutions are built– from a subsequent testing phase —when the case is considered to be learned (i.e., closed) and is applied without adding new solutions. Obviously, this limit in the number of explored solutions prevents us from guaranteeing optimal solutions, but they can still be useful to accomplish the goals of our regulatory authority. Powell et al.[15] have a similar approach to unsupervised CBR that uses reinforcement learning.

In the **reuse** phase, the solution of the retrieved case is mapped to the target problem. This may involve adapting the solution as needed to fit the new situation. In our case, since a case may have more than one associated solution, the one having the best performance results is the one chosen. Reuse is done afterwards by translating the traffic rules of the chosen solution to locations in the new solution that may be rotated if the target problem is a rotated version of the retrieved case.

Afterwards, having mapped the previous solution to the target situation, test the new solution and, if necessary, **revise**. In our traffic scenario this means to dictate the traffic norms to car agents (see previous subsection 4.1), and to observe the outcome of their application in the simulation. In current implementation, the regulatory authority checks if goals are fulfilled by observing next[3] simulation step (tick). Then, it updates the performance measure based on the number of resulting collisions and the number of applied prohibition rules: in order to promote fluid traffic, it penalizes over-regulated solutions —i.e., those abusing from preventing the cars from moving. Although system's goals are two-folded –collision avoidance and fluid traffic– they may have different relevance and, therefore, we use a weighted performance updating formula.

[3]Different time intervals could be used depending on the delay of norm application effects.

Finally, the cycle ends with the **retain** phase, that consists on the storage the resulting experience in the knowledge base. In our unsupervised CBR scenario this may lead to three different possibilities: i) If a new case was generated, then it will be stored in the case base; ii) If an existing case was retrieved and a new solution for it was generated, then retain becomes an update of the current case; and iii) if the retrieved case was closed –and thus, no solutions were added– the only required update is the performance measure[4]. This will allow the CBR agent to choose among different traffic rules depending on their application outcome. It is worth noticing that for non-deterministic environments, a desirable regulation may become undesirable further in time and become desirable again under changing circumstances. As we can see, this last step enriches the set of stored experiences, and thus it better prepares the system for future encountered problems as far as they satisfy the underlying premise that similar problems have similar solutions.

4.3 Cases and Norms

As we have already mentioned, a case in CBR is generally understood as the description of a problem and its associated solution: $Case = \langle probl, sol \rangle$ where $prob \in StateSpace$ and $sol \in Norms$. Taking into account our traffic domain, a problem description represents one particular traffic situation whereas the solution corresponds to the traffic rules that should be applied for this particular context. The regulatory authority describes traffic situations in terms of the information it gathers from the system (see section 3) : empty and occupied cells, and the headings of those cars located at occupied cells. Traffic situations can be described by considering a global point of view or a local perspective. A global scope in the representation will imply a large area of the environment and will contain all cars in the environment, no matter their location. On the other hand, a local perspective is focused in a narrower environment area and thus, only those cars near the reference point will be considered. A global scope has the advantage that it represents a complete knowledge but the disadvantage of implying a large search space. Regarding the partial scope, although being smaller in its representation size –and thus, search space–, it may fail in representing some important pieces of knowledge. Therefore, as both approaches present pros and cons, we have modeled them both in our particular traffic scenario (see next evaluation section 5 for a comparison). The remaining of this subsection presents them and the associated solutions (norms) they have within our case representation. In fact, depending on the considered scope, norms will be applied to all involved agents –if global scope– or just to the single agent that is acting as reference in the partial representation.

4.3.1 Global scope

When representing a complete traffic situation, the number of possible distributions of cars in the environment becomes high even if just considering the 7×7 example grid in Figure 1. Nevertheless, some simplifications can be taken. First, by assuming that car agents do have basic driving skills it is possible to reduce the size of the environment grid down to the intersection zone (see left side in Figure 3). These skills correspond to basic capacities such as planning

[4]Additionally, for all three possibilities we also store/update how many times the case has been applied.

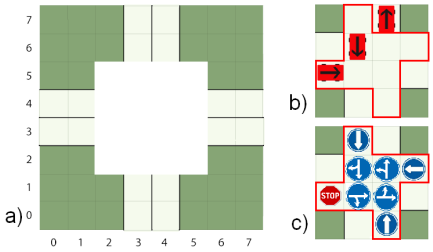


Figure 3: Global scope junction representation: a) initially discarded cells; b) orthogonal shape representing the problem; and c) applied traffic rules.

a path towards a chosen destination, following this route without leaving the proper road lanes or stopping if a car in front of them in the lane brakes suddenly[5]. Thus, traffic in the feeder and exit lanes (see Figure 1) can be discarded without losing any relevant information. Figure 3 shows how, focusing further on the intersection zone, there are still some cells that can be obviated. These cells correspond to both the field area and the exit lanes, which do not interfere in our simplified traffic. In this manner, the final problem representation can be reduced to 8 cells in the junction area.

The state space ($StateSpace$) we are representing consists thus in 8 cells that can be either empty or occupied by one or several cars. Having more than one car in a cell means a collision. Cars in our simulation are removed when colliding, so there is no need to represent this situation (further details can be found in [9]). Furthermore, a car in an occupied cell can have different headings, but due to the traffic flow restrictions, it will only be one for the cells at the junction entrance (the ones of the feeder lanes) or two for the intersection, since two different traffic directions are allowed there. Overall, we have 4 cells with two possible states –i.e., empty or occupied with a fixed heading– and 4 with 3 possible states –empty and occupied with two alternative headings– so that we have $2^4 * 3^4 = 1296$ different possible traffic situations. Finally, we can have situations that represent the same if we apply the appropriated rotation in their representations. Thus, we can further reduce the state space to $1296/4 = 324$ combinations.

Regarding the associated solution ($sol \in Norms$), it represents the same grid area than the problem (see Figure 3 right down) and for each cell, it has a norm that specifies if the car in this location should stop or should keep moving. From a deontic perspective, these traffic rules are represented, respectively, as the obligation of stopping and the prohibition to stop. Thus, the norm agent first considers the solution provided by the CBR agent (see section 4.1) and, afterwards, it applies traffic signs that can be either the stop sign or a direction sign —whose specific direction will correspond to the one of the road cell.

Finally, as we have mentioned, we lack the optimal solution ($sol \in Norms$) for each problem ($prob \in StateSpace$) and thus, the learning algorithm explores different candidate solutions. Thus, a case in our global scope representation

[5]These basic skills may also be modeled as a set of basic norms, but from our point of view regulations should leave some decisions to the agents, whose autonomy can be regulated but should not be overconstrained.

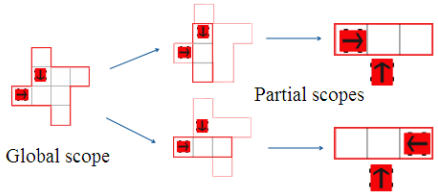


Figure 4: Case global and partial scopes.

corresponds in fact to $Case = \langle probl, \{(sol, score)\}\rangle$, where for each problem we have a set of solution-score pairs, and where a solution is a combination of traffic rules and it is associated to information about their application outcome.

4.3.2 *Partial scope*

As an alternative to use global information, it is also possible to represent situations centered in the point of view of a single agent. Common agent individual perspectives also imply having a limited observation range. Thus, the partial scope reduces the observation area to a subgrid in front of the reference car. Figure 4 illustrates an example that compares the conceptualization of both scopes: for a given global situation at a certain time step we will have as many partial descriptions as involved agents are. Thus following the example in the figure, two different situations –i.e. $prob1, prob2 \in PartialStateSpace$– will be derived. This, in terms of the CBR learning process, means that they will result in two target cases to solve, and therefore, the CBR process will be invoked twice.

As before, problems ($prob \in PartialStateSpace$) are represented by considering empty and occupied cells. The only differences are that their orientation is relative to the reference car and its shape and dimensions, which do not include the cell containing the reference car, are smaller than the global problem representation. Following previous example, we have a rotated 3×1 sub-grid. There, cell states can be 4 (empty, car forward movement, car left turning, and car right turning) for those two cells corresponding to the inner junction area and 2 possible states (empty cell or occupied with a car moving forward) for the single cell in the junction entrance. Obviously, having a sub-grid implies a smaller state space ($|PartialStateSpace| < |StateSpace|$) and thus, the number of possible cases to handle is much smaller ($4^2 * 2 = 32$ in the example). Our implementation allows the definition of different sight range or subgrids – they are treated as masks over the agent's visibility area– so that they can be empirically studied.

Once we have defined a problem ($prob \in PartialStateSpace$), its solution corresponds to the norms that will be applied to the reference agent. In this manner, cases in our traffic scenario will have a predefined set of two possible traffic rules: the obligation to stop ($obl(stop)$) and the obligation to keep moving following the road traffic direction ($proh(stop)$, so that we have a reduced set of norms: Norms=$\{obl(stop), proh(stop)\}$). Cases in this approach will have a predefined set of two possible solutions and their associated outcome measure $Case = \langle probl, \{(obl(stop), scoreStop), (proh(stop), scoreMove)\}$ and the main learning task will be to change the score associated to the performance of the application of both rules ($scoreStop$ and $scoreMove$).

4.3.3 Related metrics

Case retrieve and case update phases in our CBR cycle require the specification of two measures: the distance between two cases and the score of associated solutions.

Both global and local approaches compute case distance by comparing every cell in the area (both compared grids have the same size and shape). Differences between two cells $c_i, c_j \in grid$ are considered to be 1 if their occupancy state is different:

$dist(c_i, c_j) = 1$ if $state(c_i) \neq state(c_j)$, where

$state(c_k) = \{empty, occupied\_forward, occupied\_right\_turn, occupied\_left\_turn\}$ and $c_i, c_j, c_k \in grid$

$distance(grid_1, grid_2) = \sum_{c_i \in grid_1, c_j \in grid_2, i=j} dist(c_i, c_j)$

Thus, for example, if $state(c_i) = occupied\_forward$ and $state(c_j) = empty$, then $dist(c_i, c_j) = 1$ and the same distance results if they are occupied with cars with different headings: $state(c_i) = occupied\_forward$ and $state(c_k) = occupied\_right\_turn$ (then $dist(c_i, c_k) = 1$).

The retrieval phase looks for the most similar case in the knowledge base. In our case, the chosen case will be the one for which, if we apply a proper rotation to the retrieved grid, we get a zero distance result when comparing with the grid representing the target problem. Formally: $retrieved\_case = arg\, distance(rotation(grid, \alpha), target\_grid) = 0$ where $\alpha \in \{0, 90, 180, 279\}$ degrees in our orthogonal environment and $grid$ is the representation of the problem component in the case.

Regarding the scoring computation, we have already said that given a retrieved case with different solutions, the norm agent in the regulatory authority will choose the solution with best application performance. In the global scope, this score update is computed by punishing both the number of collisions (n_col) occurred during the next time step in the simulation; and the number of stop traffic rules ($obl(stop)$) that were applied (n_stop). Both measures are accordingly weighted so that we have:

$$global\_score = previous\_global\_score - (w_{col} \cdot n\_col + w_{stop} \cdot n\_stop)$$

Weight values depend on the priority over goals that the regulatory authority has. Our current implementation considers $w_{col} = 5$ and $w_{stop} = 1$ (i.e., a 1 to 5 ratio in the importance of collisions and traffic jams).

Finally, the computation of the partial scope score has to take into account that partial information may lead to different outcomes when applying the same norms to the same partial problem description. In order to deal with this non-deterministic phenomena, we average current with previous outcomes so to smooth the updating effect. Our testing simulation environment allows the definition of several methods, such as, for example, implementing a sliding window over the experience history.

5. EMPIRICAL EVALUATION

As we have previously mentioned, we have performed an empirical evaluation of our proposal about regulation generation by developing a multi-agent based simulation of a traffic road junction scenario. The simulator has been implemented over Repast simphony [14] so that its runtime environment interface can be used to enhance the user interface of our simulator. Figure 5 shows the user interface: top toolbar includes the standard simulation buttons such as start, step or stop buttons as well as the time (tick) count; left-side area allows the definition of the setup parameters; middle area shows the actual car simulation; and right-side area is devoted to monitor the evolution of this simulation. Thanks to the setup parameters it is possible to customize current simulation characteristics such as the environment grid dimensions; the maximum number of cars to be simultaneously interacting in the environment; or the learning modality (whose values are 0 if no learning is applied, 1 if a global scope is used in the learning process, and 2 if partial scope). With regards to the actual simulation, cars are represented as circles traversing the two intersecting roads. When cars collide they change their colour to red and disappear. Additionally, a square surrounding a car means that a stop traffic rule has been applied in this specific car position —in the figure example, this specific rule prevents the corresponding car from colliding with the car in front of it. Finally, simulation monitoring shows statistical data about those data that can be useful to follow the evolution of the specified simulation mode. Thus, since the screenshot in figure 5 corresponds to the global scope simulation mode, then the statistical data corresponds to: the number of collisions accumulated during a specific time (tick) window (2000 in the figure); how many stopping rules have been applied for this same period; the total number of cases in the knowledge base; and how many solutions have been explored for this amount of cases.

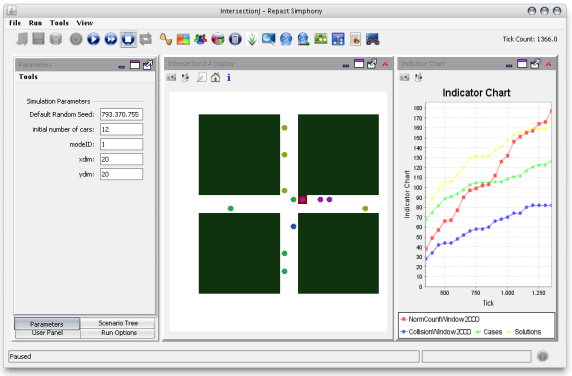


Figure 5: Traffic simulator in Repast.

5.1 Test design

In addition to the development of the simulator it was necessary to conduct a series of experiments in order to evaluate the learning approach. In fact, these experiments were designed sequentially, guided by the results and intuitions gained from previous tests. Our main objective was not to perform an exhaustive search of all possible parameters in the setup process, but a preliminary exploration that gave us some insights about our learning approach. The specific process that we followed can be described in different steps (that are summarized here and detailed in next subsection).

Obviously, we started with the basic simulation mode, in order to asses that cars behave as expected: they drive properly but, since they lack intersection traffic regulations, collisions in the junction area occur with a significant frequency.

Afterwards, we tested the global scope simulation mode. In this case, as next subsection details, we were not able

to avoid collisions completely even after running tests for long periods of time (ticks). This was in part due to the limited exploration capacity but also due to the fact that, given the size of the state space, some rare cases actually happen very scarcely, and so, the system did not have the opportunity to explore enough different solutions. This may not invalidate the global approach for all possible scenarios, but it will certainly limit its performance for those domains with large search spaces.

This led us to try the partial approach with the aim of reducing the search space despite its non-determinism problem. Results there were much more promising, since the system was able to find traffic regulations that generated almost no collisions. In addition, it was able to learn them in much shorter periods of time.

Then, by analyzing the resulting regulations, we got the intuition that they could still be described in a shorter way, and thus, we set up a final experiment with cases described by using the minimum amount of information possible.

5.2 Results

Tests with the global scope were performed along a time interval of three million ticks. During this time, the system had the opportunity to visit the whole state space —or, in other words, all possible situations were reached. Nevertheless, after this long period, some collisions –about 10 collisions in a 20000-tick period– still occurred, so the learning process failed to find the proper set of traffic rules that prevented cars from colliding. The reason is two-fold. Firstly, because, despite having encountered all possible cases, more than 20% of the cases remained open (here, cases were closed after exploring five different possible solutions). In fact, from those open cases, almost 80% just had one or two explored solutions. This means that their traffic situations might occur every 1.5 million or more steps on average, and thus, they correspond to what we refer to as rare cases. Therefore, since they happen very scarcely, the system did not had the opportunity to explore enough different solutions so to learn the best ones. The remaining 80% cases did properly close, and therefore, they were finally assigned a single solution —which corresponds to the one with higher performance score[6]. This leads to the second reason, which is the limited exploration capacity over the set of possible solutions. By analyzing the performance of solutions in closed cases, we could observe that, those cases with two or three involved cars were properly regulated, whereas having four cars in the junction lead to some cases –about 10%– whose chosen solution still generated some problems in the traffic flow. Obviously, having a limited number of chances to explore all possible combinations of traffic rules that can be assigned does not guarantee that the best solution will be found. One may argue that this limit should thus be increased, but it would extend the learning time, where collisions can be generated when applying pseudo-random traffic rules.

Having encountered some limitations with the global approach, a second set of experiments with a partial scope were set-up. The main rationale behind this decision was to reduce the size of the search space despite its intrinsic non-determinism problem. The scope was initially defined to be a 3×1 grid (as in Figure 4), so each car was able to see a range of 3 cells wide in front of him. Figure 6 plots a comparison between global and partial scopes along first 35100 simulation steps. This comparison is performed in terms of three different measures: the number of generated cases (♯cases); the total number of solutions associated to them (♯sol.); and the number of collisions that occurred during a time window of last 2000 ticks (♯col.). As we can see, learning in partial scope is much faster, since the number of cases stabilizes around 30 much before than the global scope, which tends to the 250 cases along the whole time period that is plotted. Having this small number of cases does not affect the number of collisions. On the contrary, they become zero during initial time steps, which is never the case for the global scope approach (see Figure 6).[7]. Obviously, the whole state space was explored [8] and no case could be considered to be rare. Furthermore, since all cases can just have two possible solutions, we do not consider them to be open or closed —although all of them could somehow be considered to be closed. In addition to avoiding collisions, we were interested in analyzing the kind of solutions that were found. This is so because a formal translation of an automatically learned case solution into a standard norm specification may be of great interest for many MAS. Thus we analyzed those traffic situations that had an stopping regulation and observed that most grids had in common that the cell located in the front left side of the car position was occupied by another car heading (relatively) eastwards —that is, in the direction of the cell the reference car is steering towards. Thus, we can conclude that the system had established a "left handside priority" traffic rule. And it was so despite the fact that cars were circulating on their right: in real world traffic systems, driving on one side of the road usually comes along with a priority to other participants approaching from the very same side. Tests were repeated in order to find out if the "right handside priority" was generated. Nevertheless, it was not the case, because the regulations that do not block those cars coming from

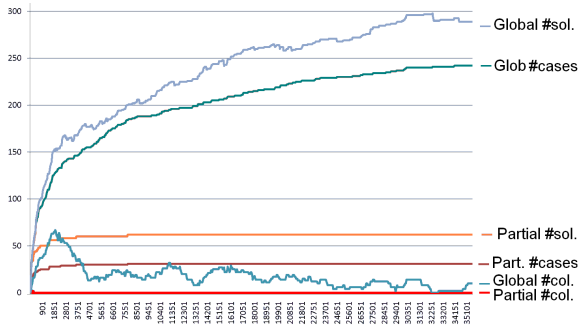


Figure 6: Runtime comparison of global and partial scope learning approaches in terms of the number of: solutions (♯sol.), cases (♯cases), and collisions (♯col.).

[6]From the closed cases, just around 20% were in fact problematic in the sense that required the addition of some stopping rules, the rest corresponded to fluid traffic situations.

[7]CBR learning depends on the order cases are learned. In our case this changes for each new simulation, since the random component on car entrance and route selection may generate traffic situations in different order.

[8]The proportion of problematic cases was very close to the global scope approach.

the left, are in fact promoting a fluid traffic flow within the junction area –similar to roundabout priorities– and thus they got better performances than right handside priorities.

Finally, we wanted to further test if the left handside priority rule was enough to avoid collisions in our traffic simulations. Thus, the last test we did was to repeat partial scope experiments with the minimum range of sight for the reference car: a single cell, the one on its left. Obtained results were really satisfactory, since both the convergence time and the number of collisions was further reduced. From these results, it is possible to argue that the case description in this setting induces the generation of the norm in a straightforward manner, so defining the proper case description may be the underlying problem. Therefore, we do not interpret the positive results obtained with this configuration as the final take-away message. On the contrary, we want to use them as a way that illustrates that learning methods can be used to generate new regulations and that, going a step further, these resulting regulations can be simple enough to be translated into standard traffic rules that can be easily interpreted and followed by external car agents.

6. CONCLUSIONS AND FUTURE WORK

This paper proposes a method to generate new regulations –norms– for multi-agent systems. Specifically, a regulatory authority learns by considering (and exploring) the ones with best application outcome. Learning is based on previous experiences, and corresponds to an unsupervised variation of Case Based Reasoning (CBR). Cases, as defined here, can then be translated to norms, in terms of prohibitions and obligations. We thus claim that this innovative approach can be highly relevant for normative MASs, since, to the best of our knowledge, no general norm generation methods have been established yet.

The paper successfully tests this approach in a simplified traffic scenario. Nevertheless, other scenarios requiring agent coordination –e.g. P2P networks, Robosoccer, etc.– may well benefit from our approach by avoiding (prohibiting) undesired situations –such as network saturation or teammate blocking in previous examples– and promoting (obliging) desired ones. The only requirements[9] are to have monitoring (and evaluating) capabilities as well as continuity in the solution space —i.e., similar social situations require similar regulations. Nevertheless, some undesired situations may appear (e.g, car collisions) as a combination of allowed individual agent actions (e.g., forward driving), thus, norms are required to be more complex than just prohibiting those actions. Context thus becomes necessary. Context, together with its analogy in real Jurisprudence, are the basic rationale of choosing a case representation approach. Nevertheless, we may consider as as future work the application of other learning techniques that cover domains with alternative characterizations. Additionally, we plan to work on norm violation and norm translation issues.

7. REFERENCES

[1] A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Commun.*, 7(1):39–59, 1994.

[9]Obviously, the domain has to be discretisable and a learning phase –where some undesired situations may occur– must be acceptable in that domain. Otherwise, as for the traffic scenario, running simulations may be most adequate.

[2] A. Artikis, D. Kaponis, and J. Pitt. *Multi-Agent Systems: Semantics and Dynamics of Organisational Models*, chapter Dynamic Specifications of Norm-Governed Systems. 2009.

[3] G. Boella and L. van der Torre. Regulative and constitutive norms in normative multiagent systems. *Proceedings of KR'04*, pages 255–265, 2004.

[4] L. Busoniu, R. Babuska, and B. de Schutter. A comprehensive survey of multiagent reinforcement learning. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(2):156–172, 2008.

[5] J. Campos, M. López-Sánchez, and M. Esteva. Multi-Agent System adaptation in a Peer-to-Peer scenario. In *ACM Symposium on Applied Computing - Agreement Technologies Track*, pages 735–739, 2009.

[6] G. Christelis and M. Rovatsos. Automated norm synthesis in an agent-based planning enviroment. In *Autonomous Agents and Multiagent Systems (AAMAS)*, pages 161–168, 2009.

[7] N. A. Davis. *Contemporary deontology*. Singer P. (ed) A companion to ethics (Blackwell), 205-218, 1993.

[8] J. Dunkel, A. Fernandez, R. Ortiz, and S. Ossowski. Event-driven architecture for decision support in traffic management systems. In *IEEE Intelligent Transportation Systems Conf.*, pages 7–13, 2008.

[9] J. F. Koeppen. *Norm Generation in Multi-Agent Systems (master thesis)*. Univ. of Barcelona, 2009.

[10] R. Kota, N. Gibbins, and N. Jennings. Decentralised structural adaptation in agent organisations. AAMAS Workshop Organised Adaptation in MAS, 2008.

[11] T. McCarty. *Reflections on Taxman: An Experiment in Artificial Intelligence and Legal Reasoning*. Harvard Law Review 837–93, 1977.

[12] J. S. Mill. *Utilitarianism*. Parker, Son, and Bourn (London), 1863.

[13] S. Modgil, N. Faci, F. Meneguzzi, N. Oren, S. Miles, and M. Luck. A framework for monitoring agent-based normative systems. In *Autonomous Agents and Multiagent Systems (AAMAS)*, pages 153–160, 2009.

[14] M. North, T. Howe, N. Collier, and J. Vos. Repast Simphony Runtime System. In *Agent Conf. Generative Social Processes, Models, and Mechanisms*, 2005.

[15] J. H. Powell, B. Hauff, and J. D. Hastings. Evaluating the effectiveness of exploration and accumulated experience in automatic case elicitation. In *Case-Based Reasoning Research and Development, LNAI 3620*, 2005.

[16] B. Savarimuthu, S. Cranefield, M. Purvis, and M. Purvis. Role model based mechanism for norm emergence in artificial agent societies. *Lecture Notes in Computer Science*, 4870:203, 2008.

[17] Y. Shoham and M. Tennenholtz. On social laws for artificial agent societies: off-line design. *Journal of Artificial Intelligence*, 73(1-2):231–252, February 1995.

[18] W. van der Hoek, M. Roberts, and M. Wooldridge. Social laws in alternating time: Effectiveness, feasibility, and synthesis. *Synthese*, 1:156, 2007.

[19] C. Zhang, S. Abdallah, and V. Lesser. Integrating organizational control into multi-agent learning. In *Aut. Agents and Multiagent Systems, 757-764*, 2009.

Norm Emergence in Tag-Based Cooperation

Nathan Griffiths
Department of Computer Science
University of Warwick
Coventry, CV4 7AL, UK
nathan@dcs.warwick.ac.uk

Michael Luck
Department of Computer Science
King's College London
London, WC2R 2LS, UK
michael.luck@kcl.ac.uk

ABSTRACT

In multi-agent systems norms are an important influence that can engender cooperation by constraining actions and binding groups together. A key question is how to establish a suitable set of norms in a decentralised population of self-interested agents, especially where individual agents might not adhere to the rules of the system. In this paper we investigate the problem of norm emergence, and the related issue of group recognition, using tag-based cooperation as the interaction model. We explore characteristics that affect the longevity and adoption of norms in tag-based cooperation, and provide an empirical evaluation of existing techniques for supporting cooperation in the presence of cheaters.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence — *Multiagent systems*

General Terms

Experimentation, Algorithms, Reliability

Keywords

Cooperation, Tags, Rewiring, Norm Emergence

1. INTRODUCTION

Multi-agent systems often comprise multiple self-interested agents seeking to achieve tasks that they cannot, or not as easily, achieve alone. In a sense, however, this self-interest suggests that without some other constraining influence, cooperation is unlikely to emerge. Norms provide just one source of such influence on agent behaviour, by constraining actions and binding a group together so that cooperation naturally arises. In this view, one key question is how to establish a suitable set of norms. While formally established institutional rules offer a means of doing this in a centralised fashion, such centralised control is often not possible in large dynamic environments. Indeed, as has been recognised elsewhere [4, 23], *social norms* are not formal, prescriptive, centrally imposed rules, but emerge informally through decentralised agent interactions. In this paper, we explore the nature of such social norms and their impact on group formation through empirical analysis, and examine the impact of *cheating* agents: those that fail to comply with norms but seek to enjoy the benefits of the group.

In seeking to investigate these issues, we adopt the *tag-based* approach taken to the problem of *group recognition*, by Riolo, Cohen and Axelrod, who use observable tags as markings, traits or social cues attached to individuals [17]. Using this approach, Hales and Edmonds have achieved promising results in peer-to-peer settings [11], but these are not resilient when cheaters are introduced, and assume agents have complete control over their links to others. In particular, we need to support cooperation in dynamic environments in the presence of cheaters where individuals have limited control over their connections. Here, tags capture *social norms*: they are recognised by agents who form groups that share a tag (within their tolerance values). The tag can be seen as a norm that is adopted by the agents who share the tag, with the group itself being *governed* by that norm, which binds it together. In this paper, therefore, we examine the problem of supporting cooperation from the perspective of norm emergence, and evaluate the effect of alternative techniques on norm emergence. The key contributions are an evaluation of the characteristics affecting longevity and size of norm-governed groups in tag-based cooperation, and further understanding of mechanisms for coping with cheaters.

The paper begins with an introduction to tag-based cooperation, followed by the specifics of using *context assessment* and *rewiring* to improving group effectiveness in the presence of cheaters. Then, in Section 4, we present an analysis of our experimental findings, and finally we conclude with a discussion of our results and their more general significance.

2. BACKGROUND

It has been widely argued that *norms* provide a valuable mechanism for regulating behaviour in decentralised societies [2, 8, 23]. Through the ongoing behaviour of individuals, norms can emerge that provide coherence and stability, and support cooperation. A common view is that where a group of agents share a particular strategy, behaviour or characteristic, a norm is established [19]. In this paper we investigate factors influencing norm emergence in a population of agents, each of which has a set of neighbours with whom it interacts. This abstract environment reflects the form of many real-world settings, such as ad-hoc communication networks or P2P content sharing. We assume there is no direct reciprocity, and so adopt Riolo, Cohen and Axelrod's tag-based approach, introduced below.

Tag-based cooperation has been considered for many years by biologists and social scientists investigating how cooperative societies of selfish individuals might evolve through the recognition of cultural artefacts or traits [1, 5, 7, 12]. Simple observable traits, or *tags* [13], can be used as cultural artefacts to engender cooperation without relying reciprocity [3,

17, 22]. Existing work on tags, however, has given little consideration to the possibility that some members of the population may be *cheaters* who deviate from the rules of the system, by not cooperating when they should. In this paper, our investigation of norm emergence allows for the possibility of cheaters.

Riolo, Cohen and Axelrod (RCA) propose a tag-based approach to cooperation in which an individual's decision to cooperate is based on whether an arbitrary tag (i.e. observable trait) associated with it, is sufficiently similar to that associated with a potential recipient [17]. The approach is illustrated using a simple *donation scenario* in which each agent acts as a potential donor with a number of randomly selected neighbours. Should an agent opt to donate, it incurs a cost c, and the recipient gains a benefit b (it is assumed that $b > c$), otherwise both receive nothing. Each agent i is initially randomly assigned a tag τ_i and a tolerance threshold T_i with a uniform distribution from $[0, 1]$. An agent A will donate to a potential recipient B if B's tag is within A's tolerance threshold T_A, namely $|\tau_A - \tau_B| \leq T_A$. Agents are selected to act as potential donors in P interaction pairings, after which the population is reproduced proportionally to their relative scores, such that more successful agents produce more offspring. Each offspring is subject to mutation, so that with a small probability a new (random) tag is received or noise added to the tolerance. In relation to norms, the key aspect here is that donation rate is an assessment of the effectiveness of the society and the impact of norms: the greater the effectiveness, the higher the donation rate.

RCA have shown that a high cooperation rate can be achieved with this simple approach. They observe cycles in which a cooperative population is established, which is then invaded by a mutant whose tag is similar (and so receives donations) but has a low tolerance (and so does not donate). Such mutants initially do well, leading to them taking over the population subsequently lowering the overall rate of cooperation, but eventually the mutant tag and tolerance become the most common and cooperation again becomes the norm [17].

Hales and Edmonds (HE) apply RCA's approach in a P2P setting, with two main changes [11]. First, RCA's *learning interpretation* of reproduction is adopted, so that each agent compares itself to another at random and adopts the other's tag and tolerance if the other's score is higher (subject to potential mutations) [17]. Second, HE interpret a tag as being an agent's neighbours in the P2P network, i.e. an agent's links to others. In RCA's work each agent is connected to each other agent, with no corresponding notion of neighbourhood. In HE's model, the process of an agent adopting another's tag is equivalent to dropping all of its own connections, and copying the connections of the other agent (and adding a connection to the other agent itself) [11]. Importantly, in our view, this model reflects the formation of groups based on recognition of tags in group members.

Using simulations, HE have shown this approach to be promising in situations where agents are given free reign to rewire the network and replace all of their connections each reproduction. This rewiring is an *all-or-nothing* operation, in that although an agent can adopt a completely new set of neighbours (*replacing* its existing neighbourhood), it cannot *modify* its existing neighbourhood. Our view is that such extreme rewiring, where the neighbourhood topology might completely change with each new generation, is not practicable in all scenarios. For example, in a communication network this would imply that all existing routes become outdated and need to be re-established, while in a content sharing system an agent would lose all information about the content available in its neighbourhood. In this paper we consider a less extreme situation, in which agents are able to rewire a proportion of their neighbourhood.

Both RCA and HE assume that agents do not deviate from the rules of the system, i.e. they assume no cheaters. A *cheater* is an agent that accepts donations, but will not donate to others, even if the rules of the system dictate that it should. We assume that if a cheater reproduces, then its offspring will also cheat. In this paper we assume that the traits embodied by tags are observable to others, meaning that cheaters cannot falsify their tags. In standard tag-based cooperation, introducing even a small proportion of cheaters into the population causes cooperation to collapse [9].

Norm emergence has been considered in several other settings. For example, norms can emerge in a social dilemma game when individuals are repeatedly randomly paired [20], and in certain settings can emerge simply by individuals learning based on their own individual histories [21]. In this paper, however, our focus is on using the interpretation of a shared tag as representing a norm to further our understanding of tag-based cooperation.

3. IMPROVING GROUP EFFECTIVENESS

In seeking to examine the impact of cheaters on group formation and norm emergence, we consider a population of agents, each of which has its own tag and a set of connections to n neighbours, such that agents can only interact with their neighbours (although for reproduction we consider the population as a whole). We assume that a proportion of agents are cheaters and will not cooperate with others even when their tags are within the tolerance threshold. The *donation scenario* and parameter values used by RCA are adopted, such that benefit $b = 1$ and cost $c = 0.1$ [17]. Each agent i is initially assigned an arbitrary tag τ_i and tolerance T_i with uniform distribution from $[0, 1]$[1]. We investigate norm emergence in relation to RCA's tag-based approach and two techniques that we have previously proposed for improving cooperation in the presence of cheaters: context assessment [9] and rewiring [10].

3.1 Context assessment

Our first technique, originally proposed in [9], enables agents to assess their neighbourhood, or group, in terms of how cooperative they perceive their neighbours to be. The donation decision is modified so that an agent's assessment of its neighbourhood context becomes a factor in the decision to donate. Agents are given a fixed length FIFO memory to record the last l donation behaviours observed for each neighbour. When the neighbour donates, an observation value of $+1$ is recorded, and when it does not -1 is recorded. This memory is fairly sparse, since the number of interactions is small compared to the number of agents, and so the overhead incurred is relatively small (2 bits per

[1]We actually use a lower bound on *tolerance* of -10^{-6} to address Roberts and Sherratt's concerns regarding agents with identical tags being forced to cooperate [18]. This also allows the population to contain non-cooperative agents of the form considered by Masuda and Ohtsuki [15].

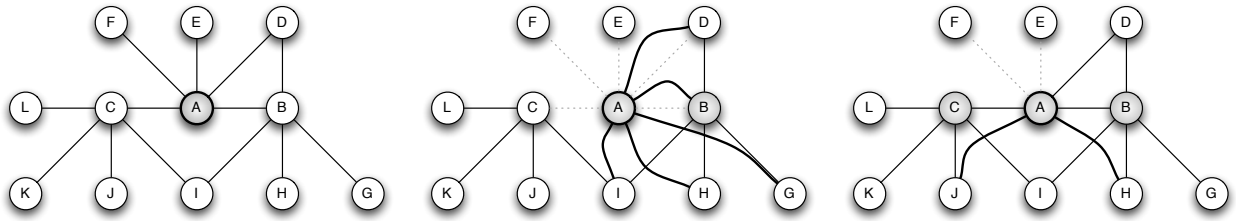


Figure 1: Rewiring showing (a) the original neighbourhood rewired using (b) HE's method and (c) our rewiring approach.

observation for $n \times P$ observations, where n is the number of neighbours and P the number of pairings).

In order to assess its neighbourhood context, an agent considers each of its n neighbours in turn, and determines the contribution to the context assessment c_i of neighbour i, which is simply the proportion of observed interactions in which the neighbour donated, given by:

$$c_i = \begin{cases} \frac{\sum_{j=1}^{l_i} \begin{cases} o_i^j, & \text{if } o_i^j > 0 \\ 0, & \text{otherwise} \end{cases}}{l_i}, & \text{if } l_i > 0 \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where o_i^j represents the j'th observation of neighbour i, and l_i is the number of observations recorded of i's donation behaviour ($l_i < l$). By considering each of its n neighbours, agent A's assessment of its current neighbourhood context C_A is given by:

$$C_A = \frac{\sum_{i=1}^{n} c_i}{n} \quad (2)$$

This context assessment can be used to influence the donation decision. The intuition is that agents 'expect' that by donating they are more likely to receive a future donation from some other (observing) agent, thus binding a group together. However, since the number of interactions is small compared to the number of agents, this is a *weak* notion of indirect reciprocity, and insufficient to support a typical notion of reputation. An agent's donation to another is unlikely to be directly repaid or directly observed by a third party, so there is little direct or indirect reciprocity. Instead, context assessment gives an impression of the donation behaviour in a neighbourhood, indicating the likelihood of receiving future donations. An agent's assessment of its neighbourhood context is incorporated into the model by adapting the decision to donate, such that both tolerance and neighbourhood context are considered. Thus, an agent A will donate to B if:

$$|\tau_A - \tau_B| \leq (1 - \gamma).T_A + \gamma.C_A \quad (3)$$

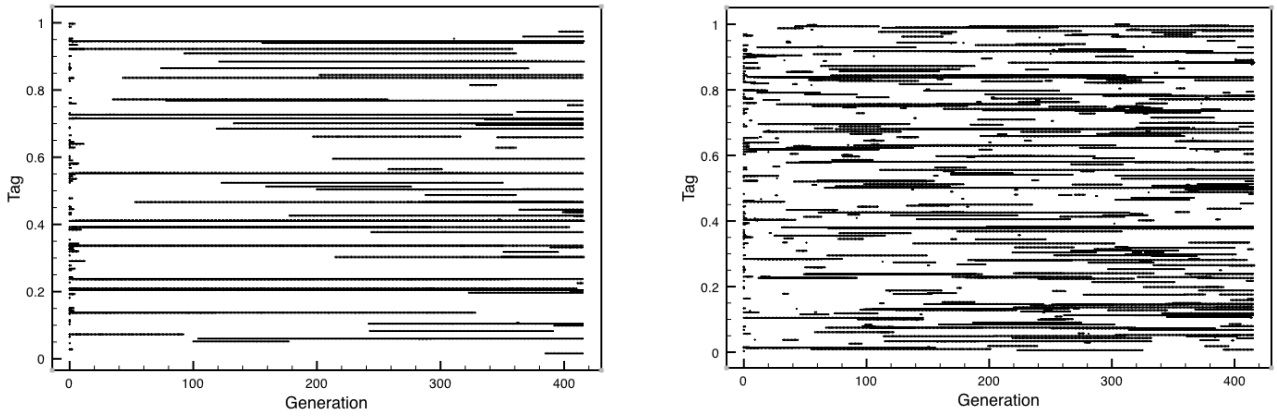
The parameter γ (the *context influence*) allows us to tune the technique. The context influence is in the range $[0, 1]$, with $\gamma = 0$ making the technique identical to RCA's method, while $\gamma = 1$ makes the donation decision determined solely by an agent's assessment of its neighbourhood context. We adopt RCA's learning interpretation of reproduction (as do HE), such that after a fixed number of interaction pairings P an agent compares itself to another, random selected from the population. If the other agent is more successful, then its tag and tolerance are copied (subject to a small probability of mutation), otherwise the tag and tolerance are unchanged.

3.2 Rewiring

Our second technique, proposed in [10], enables agents to *rewire* their network neighbourhoods, such that after reproduction an agent removes a proportion λ (the *rewire proportion*) of connections, and replaces them with connections to new neighbours. This approach is motivated by HE's results, but unlike HE we do not assume that agents can replace *all* of their connections since, as discussed above, this is likely to be impractical in real-world settings. In our mechanism, after reproduction, the $n \times \lambda$ worst neighbours are removed, and the best (non-duplicate) neighbour from each of the agent's $n \times \lambda$ best neighbours are added. The neighbours are considered in descending rank order and, for each, the best non-duplicate neighbour is added. Additional randomly selected neighbours are added if necessary to prevent the neighbourhood shrinking due to duplication (agents have at most one connection to another, and duplicate connections are meaningless).

Connections to remove are determined by ranking each neighbour i using the contribution to the context assessment c_i (defined in Equation 1), with agents having the lowest c_i values being removed. The contribution to the context assessment is also used to determine which connections to add, with an agent asking each of its $n \times \lambda$ best neighbours to recommend their best non-duplicate neighbour. If the c_i values of two or more agents are equal then one is selected arbitrarily. The *rewire proportion* determines the extent to which the network is rewired in each generation. Such rewiring can be thought of as a simplistic reputation mechanism, since agents update their connections based on the experiences and recommendations of others. However, unlike typical reputation mechanisms, the assessment is based on relatively little information, which is not predicated on a notion of (direct or indirect) reciprocity [14, 16].

Figure 1 illustrates the alternative rewiring approaches. Agent A's original neighbourhood is shown in (a). The results of applying HE's rewiring approach is shown in (b) where A drops all of its connections and adopts those of B. Our rewiring approach is illustrated in (c). If A's neighbours in order of preference are B, C, D, E, F, and 2 neighbours are to be replaced, then connections to E and F will be dropped. If B's neighbours, are D, H, I, G, A and C's neighbours are J, K, L, A, I in preference order, then A will add H from B's neighbourhood (D is already in A's neighbourhood and so



(a) low mutation (b) high mutation

Figure 2: Tags with RCA's standard approach using low and high mutation rates.

not added) and J from C's neighbourhood.

4. EXPERIMENTAL ANALYSIS

Using the PeerSim P2P simulator[2], we have built a simulation that allows us to explore norm emergence using RCA's standard approach, context assessment, and rewiring. The quantitative results presented here are averaged over 10 runs using a population of 100 agents, a neighbourhood size of $n = 10$, with 10 pairings per agent per generation ($P = 10$), and a cheater proportion of 30%. Where context assessment is incorporated a *context influence* of $\gamma = 0.5$ is used, and similarly where rewiring is incorporated we use a *rewire proportion* of $\lambda = 0.5$. After reproduction there is a 0.001 probability of mutating the tolerance of each agent by adding Gaussian noise (with mean 0 and standard deviation 0.01), along with a probability of mutating each agent's tag by selecting a new random value. We consider two configurations for mutating tags: a *low mutation* rate of 0.001 and a *high mutation* rate of 0.01. The low mutation rate represents a generally stable population in which mutation is simply a small part of the evolutionary process. Conversely, the high mutation rate represents a more dynamic environment in which there is more significant fluctuation in tags present in the population (this is akin to a small proportion of the agents leaving and joining at the end of each generation).

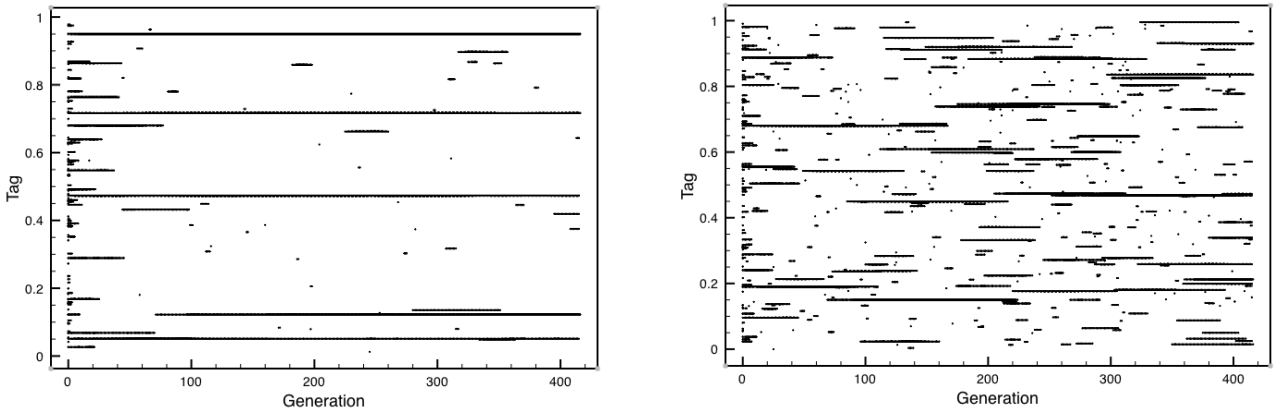
In this section we give an overview of the main findings from our simulations, focusing on two main characteristics. First, we consider the *donation rate* defined as the proportion of interactions resulting in a donation in the final generation of the simulation, averaged across the population. This indicates the effectiveness of the groups that emerge in complying with the norms that establish those groups and govern their maintenance. Second, we consider the *number of unique tags* present in the final generation, which indicates the number of norm-governed groups that have been established. Where a group of agents share a tag (and each others' tags are within their tolerance values) we interpret this as recognising a norm that is then established, since those agents will cooperate by donating to each other (provided that they are not cheaters). The number of unique tags indicates the number of such norm-governed groups that are formed, since each tag value corresponds to a tag group. However, it is important to note that some tags may be adopted only by a single agent in which case there is no norm, and so the number of tags is only an indicative metric.

Where there is a low number of unique tags, so that there are few groups, the average number of agents adopting each tag is high, and the groups are larger in size, with the respective norms being more widely adopted, and a reduced likelihood of a tag belonging to a single individual. Conversely, as the number of unique tags (and therefore groups) increases, the average number of agents having adopted each tag (and hence in each group) reduces, so that the corresponding norms are less widely adopted and there is an increased likelihood of a tag being ascribed to a single agent only. Thus, for lower numbers of unique tags there is more significance in them representing groups of agents having adopted those tags as norms. Note that this is an informal notion of norm establishment, in comparison to other approaches that have a group leader [6], or explict strategies emerging rather than simply shared tags [20].

4.1 Context assessment

The base case for our comparisons is RCA's approach which, with a low mutation rate gives a donation rate of 0.204, and with a high mutation rate gives a donation rate of 0.032. The increased dynamism of the environment, represented by the increased mutation rate, has a catastrophic effect on the donation rate, and in turn on group effectiveness. For a donation to occur, agents must share a tag (within their tolerance values). With a low mutation rate, RCA's approach gives an average of 35.8 tags, each shared by 2.8 agents on average. With a high mutation rate, there are 79.2 tags shared by 1.3 agents on average. Thus, with a high mutation rate a large number of tags are adopted by a single agent, as confirmed by the very low donation rate observed (0.032). In relation to norm emergence, this means that in the low mutation case the resulting norms are on average only adopted by 2.8 agents. In the high mutation case the tag groups do not, on average, correspond to norm emergence since less than two agents adopt each tag. It is not our concern in this paper to attempt to define the num-

[2]`http://peersim.sourceforge.net/`



(a) low mutation (b) high mutation

Figure 3: Tags with context assessment using low and high mutation rates.

ber of agents needed for norm emergence, but clearly there must be at least two agents involved.

Figure 2 shows the evolution of tags in the population over the duration of a sample simulation run for both low and high mutation settings. Each point represents the presence of a tag in a particular generation, and where a tag persists for several generations the points form a line from the generation in which the tag group is created to the generation in which it collapses. Our numerical results are confirmed by Figure 2 which shows that many more tags are present in the high mutation setting than the low setting. This graphical representation also allows us to observe the formation of norm-based groups. In particular, in the low mutation setting norm-governed groups are established and maintained for many generations, while in the high mutation setting many such groups have very brief durations appearing as points or very short lines. (Note that since the number of unique tags is large in Figure 2(b), many of the points do not represent norm establishment, as discussed above.)

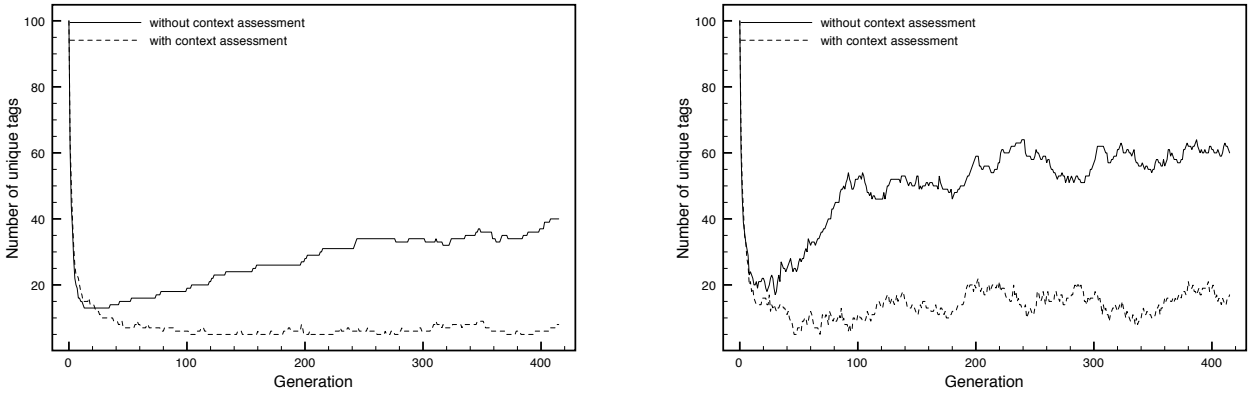
The evolution of tags in the population when using context assessment is shown in Figure 3. Comparing Figures 2 and 3 it is immediately apparent that there are significantly fewer tags present using context assessment than with RCA's approach. On average, context assessment in a low mutation setting results in only 3.7 tags (compared to 35.8 with RCA's approach) and 12.1 tags (compared to 79.2) for a high mutation rate. A donation rate of 0.475 and 0.429 is obtained for the low and high mutation settings respectively (compared to 0.204 and 0.032). The reduction in donation rate in the high mutation setting compared to the low mutation setting is less significant (approximately 10%) than with RCA's approach (where the reduction is approximately 85%). We see this as demonstration that context assessment is more stable in supporting cooperation in dynamic environments than RCA's approach. This tells us that norms resulting from context assessment are more widely adopted than with RCA's approach (by 27 and 8.3 agents on average for the low and high mutation rates respectively). Given that these norms are more widely adopted, we would expect an increase in the group effectiveness (as indicated by donation rate achieved), which is indeed the result we observe.

The evolution of the number of unique tags during a sample simulation run for low and high mutation rates is illustrated in Figure 4. In both settings the number of tags is initially very high and is (approximately) equal to the population size since agents are randomly allocated tags. During the first few generations the number of unique tags drops significantly as agents begin to copy tags from their more successful neighbours. As the simulation progresses the number of unique tags then stabilises. From Figure 4 we can see that in addition to context assessment resulting in significantly fewer tags than RCA's approach, the number of tags also stabilises more quickly. When the mutation rate is high the number of tags increases, as does the extent of the fluctuations over generations (with both approaches having similar fluctuation levels). It is clear that norm-governed groups emerge more quickly using context assessment than with RCA's approach, and on average norms are adopted by many more agents. Due to space constraints we do not discuss the effect of memory length in this paper, but in [10] we have shown that it is not a significant factor.

4.2 Rewiring

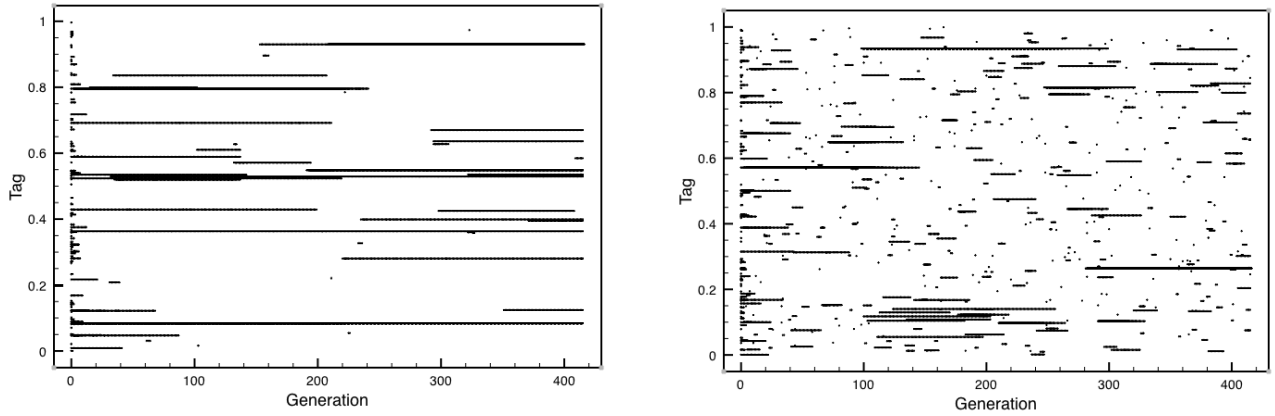
Rewiring gives similar improvements to context assessment, with donation rates of 0.57 and 0.498 for the low and high mutation settings respectively (both of which are higher than with context assessment). The average number of unique tags is higher than for context assessment, with 11.9 and 16.8 for low and high mutation rates, but is significantly lower than with RCA's approach. The norms that are established are adopted by fewer agents using rewiring than with context assessment, 8.4 (rather than 27) and 6.0 (rather than 8.3) for the low and high mutation rates, although by many more agents than with RCA's approach. This result is unexpected, since we intuitively expect that a higher donation rate (and therefore higher group effectiveness) would be achieved when norms are more widely adopted.

The discovery that a higher donation rate can be obtained with smaller groups is potentially very powerful, since in many situations we would like to balance the desire to have widely adopted norms (and few groups) with the desire to ensure that there are several established groups from which agents can select. The widest possible adoption of a norm is where a single norm is adopted by the population in a



(a) low mutation (b) high mutation

Figure 4: Number of unique tags with context assessment using low and high mutation rates.



(a) low mutation (b) high mutation

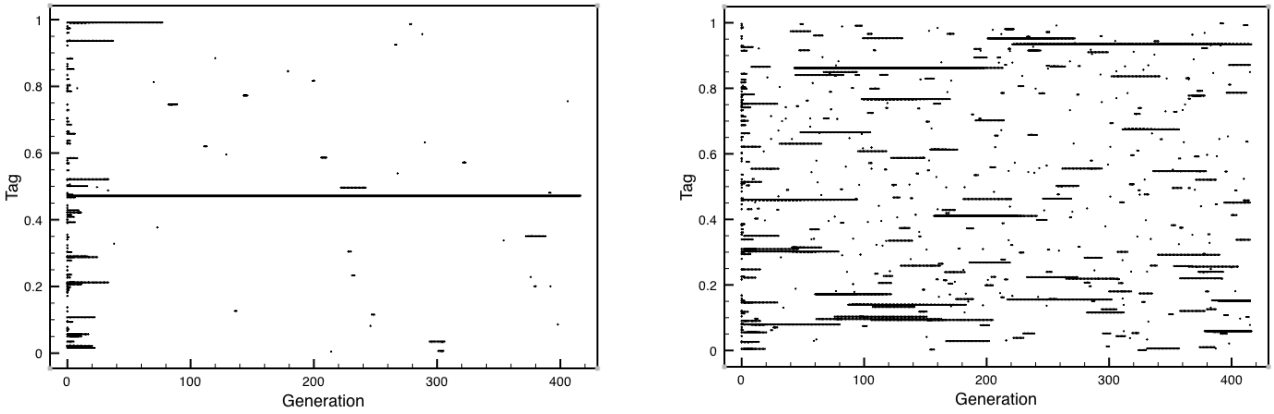
Figure 5: Evolution of tags in a population using rewiring with low and high mutation rates.

single group. Conversely, the largest set of norms is obtained where we have a minimum number of agents per tag for norm emergence, giving groups of that size. As mentioned above, we are not concerned with attempting to define a minimum membership, but clearly more than one agent is required, and so the upper bound of the number of norm-governed groups is half the population size. In the donation scenario the motivation behind balancing the level of adoption and number of norms established is that mutations can cause a norm to collapse at any point, and in such cases we would like agents to be able to adopt an alternative norm by joining another tag group. However, this balance is a general issue in many distributed systems, and corresponds to the general view that fostering competition and avoiding monopolies can be beneficial. The question of how many norms or groups is ideal in a particular setting is an open question, and we see this as a key area of future work.

The evolution of tags in the population using rewiring for sample simulation runs is shown in Figure 5. As is the case with context assessment (Figure 3) there are significantly fewer tags at any point in time than with RCA's approach (Figure 2). As noted above, rewiring results in slightly more tags than context assessment. Figure 5 also allows us to observe that the duration of a given tag group is generally slightly reduced using rewiring in comparison to context assessment, implying that the norms emerging are slightly less long-lived. In the donation scenario this is not a major concern, since there is little cost to changing tag groups, but more generally there may be a higher cost associated with such a change. It is desirable, in general, for norm-governed groups to be of longer duration as we observe with context assessment, but also for there to be a reasonable number of alternative groups as with rewiring.

4.3 Context assessment with rewiring

We have previously shown that combining context assessment and rewiring improves the donation rate [10]. We observe a donation rate of 0.686 and 0.631 for the low and high mutation settings respectively. In terms of norm emergence, a key question is which, if any, of the properties of context assessment (few norms of longer duration) and rewiring (more norms of shorter duration) the combined approach gives. We



(a) low mutation (b) high mutation

Figure 6: Evolution of tags using context assessment and rewiring with low and high mutation rates.

find that the combined approach results in 4.4 and 10.4 tag groups in the low and high mutation setting, corresponding to adoption by 22.7 and 9.6 agents on average respectively. In the low mutation setting this is similar to context assessment, and we would prefer more norms given our desire to have alternative groups established in case of norm collapse. With a high mutation rate, the number of norms is similar to that obtained with rewiring, indicating that there are alternative groups in the event of a norm collapsing. However, the duration of norms and groups is also important in ensuring that alternatives are available.

Figure 6 shows the evolution of tags using the combination of context assessment and rewiring in sample runs, and we can make two important observations. First, the run for the low mutation rate represents a particularly low number of tag groups, with a single dominant group spanning all generations and a small number of short duration groups appearing. A similar situation can occur when using context assessment alone (although the run in Figure 3 shows an example where this does not occur). We would like to avoid this by ensuring that there are alternative established norms (i.e. tag groups) at any point in time, in case of norm collapse. Second, with a high mutation rate many of the groups established persist for only very short durations. Thus, although there are many groups, avoiding the problem from the low mutation case, we would like them to be of longer duration. It seems, therefore, that while combining context assessment with rewiring increases the donation rate, it does not result in groups that have the desired characteristics. In a low mutation rate we would prefer a higher number of norm-governed groups to ensure alternative groups are established in case of collapse, while with a high mutation rate we would prefer more persistence to avoid frequent collapses.

Context assessment appears to have a significant reduction on the number of norm-governed groups established, especially with a low mutation rate. Although this results in an increased donation rate (indeed, when combined with rewiring it gives the highest donation rate) suggesting group effectiveness, it also gives reduced diversity in the population. In many settings this is undesirable, since if there is collapse of cooperation in those groups due to the collapse of the respective norms, there are no alternative established groups for agents to join.

4.4 Summary of results

Our simulations show that both context assessment and rewiring improve group effectiveness (through donation rate) by the formation of groups of agents sharing a particular tag value, which we can interpret as norm establishment. A summary of the quantitative results is shown in Table 1. In both the low and high mutation settings the highest donation rate is achieved when combining context assessment and rewiring. Rewiring gives the second highest rate, followed by context assessment, and finally RCA's approach. In the low mutation setting context assessment, with and without rewiring, results in a low number of norm-governed groups being established. In the high mutation setting the number of groups established is reduced when context assessment is used in comparison to rewiring alone, but this difference is less significant. A visual analysis of the evolution of tags reveals that norm and group duration is potentially an issue, with each of the techniques discussed resulting in several short duration norm-governed groups being established.

5. DISCUSSION AND CONCLUSIONS

Interpreting the formation of groups of agents sharing a tag as the emergence of norms, allows us to view RCA's approach as facilitating norm emergence. Indeed, it is norm emergence that leads to donations, since only when two or more agents share a tag (within their tolerance) will a donation occur. Our previously proposed techniques of context assessment and rewiring to cope with cheaters [9, 10] also facilitate norm establishment, and this interpretation enables us to explore their operation. The norms and groups that are formed are more widely adopted (with fewer norms) than with RCA's approach. Using context assessment increases adoption, especially in the low mutation setting, however as a result there may not be alternative norm-governed groups available in the event of the collapse of an established norm.

An increase in mutation rate leads to less long lived and less widely adopted norms, so that there are more groups, with fewer members, persisting for less generations. Norms also emerge with many tag groups and few agents per tag, but only few such norms are widely adopted and long lived,

Approach	Mutation rate	Average donation rate	Average number of tags	Average size of tag group
RCA's mechanism	0.001	0.204	35.8	2.8
Context assessment	0.001	0.475	3.7	27
Rewiring	0.001	0.57	11.9	8.4
Context assessment and rewiring	0.001	0.686	4.4	22.7
RCA's mechanism	0.01	0.032	79.2	1.3
Context assessment	0.01	0.429	12.1	8.3
Rewiring	0.01	0.498	16.8	6.0
Context assessment and rewiring	0.01	0.631	10.4	9.6

Table 1: Summary of donation rate, number of tags and size of tag group (number of agents per tag).

as seen in Figures 3, 5 and 6, where there are few long duration groups. This is a key area of future investigation, namely to understand the factors that influence the number and duration of norms, and to investigate whether there is an optimal number of groups in a given configuration.

Using tag-based cooperation to investigate norm emergence allows us to observe the effects of various approaches. In particular, it is through norm establishment (i.e. reducing the number of tags) that context assessment and rewiring improve group effectiveness, and this interpretation may inform improvements to tag-based approaches. Since mutation can cause norm collapse it is important to ensure that there are alternative groups. Context assessment (with or without rewiring) is found to reduce the diversity of tags to a low level, and this suggests that further work is needed on coping with cheaters. There is a complex relationship between the evolution of tag groups, the number of tags, and donation rate, and we will investigate this in future work. In particular we aim to develop mechanisms to ensure norm diversity and prolong norm duration. We also aim to investigate the influence of connection topologies and population size to explore our techniques in more realistic settings.

6. REFERENCES

[1] P. D. Allison. The cultural evolution of beneficent norms. *Social Forces*, 71(2):279–301, 1992.

[2] R. Axelrod. An evolutionary approach to norms. *American Political Science Review*, 80(4):1095–1111, 1986.

[3] R. Axelrod, R. A. Hammond, and A. Grafen. Altruism via kin-selection strategies that rely on arbitrary tags with which they coevolve. *Evolution*, 58(8):1833–1838, 2004.

[4] C. Bicchieri. *The Grammar of Society: The Nature and Dynamics of Social Norms*. Cambridge University Press, 2006.

[5] R. Boyd and P. J. Richerson. The evolution of indirect reciprocity. *Social Networks*, 11:213–236, 1989.

[6] J. C. Burguillo-Rial. A memetic framework for describing and simulating spatial prisoner's dilemma with coalition formation. In *Proc. of the 8th Int. Conf. on Autonomous Agents and Multiagent Systems*, pages 441–448, 2009.

[7] R. Dawkins. *The Selfish Gene*. Oxford University Press, 1976.

[8] F. Flentge, D. Polani, and T. Uthmann. Modelling the emergence of possession norms using memes. *J. of Artificial Societies and Social Simulation*, 4(4), 2001.

[9] N. Griffiths. Tags and image scoring for robust cooperation. In *Proc. of the 7th Int. Conf. on Autonomous Agents and Multiagent Systems*, pages 575–582, 2008.

[10] N. Griffiths and M. Luck. Changing neighbours: Improving tag-based cooperation. In *Proc. of the 9th Int. Conf. on Autonomous Agents and Multiagent Systems*, 2010, to appear.

[11] D. Hales and B. Edmonds. Applying a socially inspired technique (tags) to improve cooperation in P2P networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 35(3):385–395, 2005.

[12] W. D. Hamilton. The genetical evolution of social behaviour I&IIi. *J. of Theoretical Biology*, 7(1):1–52, 1964.

[13] J. H. Holland. *Hidden order: How adaptation builds complexity*. Addison-Wesley, 1995.

[14] A. Jøsang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, 2007.

[15] N. Masuda and H. Ohtsuki. Tag-based indirect reciprocity by incomplete social information. *Proc. of the Royal Society B*, 274(1610):689–695, 2007.

[16] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation systems. *Communications of the ACM*, 43(12):45–48, 2000.

[17] R. Riolo, M. Cohen, and R. Axelrod. Evolution of cooperation without reciprocity. *Nature*, 414:441–443, 2001.

[18] G. Roberts and T. N. Sherratt. Does similarity breed cooperation? *Nature*, 418:499–500, 2002.

[19] B. T. R. Savarimuthu, M. Purvis, and M. Purvis. Social norm emergence in virtual agent societies. In *Proc. of the 7th Int. Conf. on Autonomous Agents and Multiagent Systems*, pages 1521–1524, 2008.

[20] S. Sen and S. Airiau. Emergence of norms through social learning. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence*, pages 1507–1512, 2007.

[21] Y. Shoham and M. Tennenholtz. On the emergence of social conventions: modeling, analysis and simulations. *Artificial Intelligence*, 94:139–166, 1997.

[22] A. Traulsen. Mechanisms for similarity based cooperation. *The EPJ B - Condensed Matter and Complex Systems*, 63(3):363–371, 2008.

[23] D. Villatoro and J. Sabater-Mir. Dynamics in the normative group recognition process. In *Proc. of the 11th Congress on Evolutionary Computation*, pages 757–764, 2009.

An Adherence Support Framework for Service Delivery in Customer Life Cycle Management

Kumari Wickramasinghe
Department of General Practice
Monash University,
Melbourne, Australia

Christian Guttmann
Department of General Practice
Monash University,
Melbourne, Australia

Michael Georgeff
Department of General Practice
Monash University,
Melbourne, Australia

Heinz Schmidt
School of Computer Science and Information Technology
RMIT University, Australia

Ian Edward Thomas
Department of General Practice
Monash University,
Melbourne, Australia

ABSTRACT

In customer life cycle management, service providers are expected to deliver services to meet customer objectives in a manner governed by some contract or agreement. When human agents are involved as contract parties (either as customers or service providers), service delivery failures may occur as a result of changes, inconsistencies, or "deficits" in the mental attitudes of these agents (in addition to other possible changes in the service delivery environment). It may be possible to avoid such failures by monitoring the behavior of the contract parties and intervening to ensure adherence to the contractual obligations. The aim of this paper is twofold: (1) to develop a conceptual framework to model how deficits in mental attitudes can affect service delivery; and (2) to propose an adherence support architecture to reduce service delivery failures arising from such deficits. The conceptual framework is based on Bratman's notion of "future-directed intentions" and Castelfranchi's belief-based goal dynamics. The adherence support architecture introduces the notions of precursor events, mental-state recognition processes, and intervention processes and utilizes the Belief-Desire-Intention (BDI) architecture. A multi-agent implementation is carried out for chronic disease management in health care as a proof-of-concept for a complex customer care management system.

Categories and Subject Descriptors

I.2.11 [**Computing methodologies**]: Distributed Artificial Intelligence-Multiagent Systems

General Terms

Algorithms, Contract Service Delivery Management

Keywords

Service Delivery, Human Agents, Mental Attitudes, Mental State Recognition, Adherence Support

1. INTRODUCTION

The aim of the Intelligent Collaborative Care Management (ICCM) Project[1] [17, 9] is to develop a comprehensive architecture for managing the complete life cycle of customer care. In this framework, we consider that a customer is provided a number of possibly interrelated services by various service providers in a manner governed by some agreement or contract. The customer has certain goals or objectives that these services are intended to fulfil and the service providers themselves may have certain objectives in delivering the services to the customer. These services are to be delivered over time and potentially the entire lifetime of the customer.

Often human agents are involved as contract parties (customers and service providers) in CLCM. Humans are goal directed and reason before performing an action or selecting goals to achieve. Human reasoning is a bounded rational process subject to the properties of mental attitudes: beliefs, desires, intentions, plans, emotions etc. In CLCM, changes, inconsistencies, or "deficits" in these mental attitudes may prevent contract parties from performing the required activities, which may then affect:

1. The formation of the contractual arrangements;
2. The management of these contractual arrangements; and
3. The delivery of the services according to the contractual arrangements.

Adherence support mechanisms may assist human agents to maintain and execute contracts as committed. Continuous monitoring and intervention may either reverse the

[1]The work reported here was supported in part by British Telecom (CT1080050530), the Australian Research Council (LP0774944), the Australian Governments Clever Networks program and the Victorian Department of Innovation, Industry and Regional Development, Department of Health, and Multi Media Victoria. We also gratefully acknowledge the contributions and advice from Dr Simon Thompson and Dr Hamid Gharib of British Telecom and Professor Leon Piterman, Dr Kay Jones, Associate Professor Peter Schattner, and Mr Akuh Adaji of the Department of General Practice, Monash University.

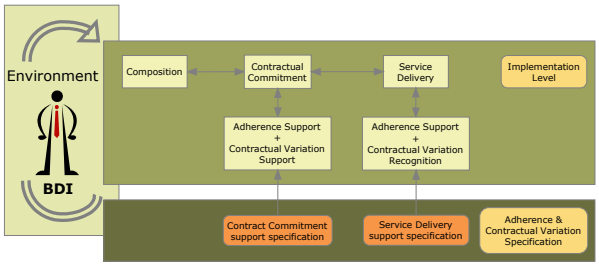


Figure 1: ICCM architecture: adherence and variation support.

deficits or identify reasons to vary the contract. The ICCM architecture with human contract parties and adherence support requirements is shown in Figure 1. The human parties are modelled as BDI agents [5] because this architecture provides formalisms and representation mechanisms of agents' mental attitudes. (Refer [9] for a detailed description of the ICCM architecture). This paper investigates:

- A conceptual framework to model contract violations arising from deficits in mental attitudes; and
- An architecture for adherence support reducing delivery such contract violations.

The conceptual framework is based on Bratman's notion of "future-directed intentions" [5] and Castelfranchi's belief-based goal dynamics [7]. The adherence support architecture with the ongoing monitoring and management processes is based on the following elements:

- precursors: a priori actions, steps or states that may indicate whether the actions of the contract parties are on-track with respect to the contract (e.g., the patient setting an appointment on time to visit a care provider) or likely to go off-track (e.g., very low or very high blood pressure).
- detection strategies for precursors: mechanisms to detect the occurrence or missing occurrence of such precursors at run-time;
- mental state recognition processes: processes to identify possible deficit(s) in mental attitudes which may have led to the (non-)occurrences of precursors; and
- intervention processes: strategies to intervene with one or more of the contract parties to reduce the likelihood of the parties violating their obligations.

As seen in Figure 1, the conceptual framework and the adherence support architecture is common to both the contractual commitment stage[2] and the service delivery stage. In this paper, we consider examples and implementation results from the service delivery stage, in which service providers are expected to deliver services as committed and the customer is expected to perform certain required activities to be engaged in the contract. We focus on contracts made in the health care domain, specifically, chronic disease management [6], as an example of a complex, dynamic domain with human contract parties in which precursor recognition, service delivery failure, and adherence support are important elements.

[2]The contractual commitment stage is additionally based on partially regulated market and negotiation theories and is described elsewhere [9]

Table 1: A sample contract.

Party's name	Party's type	Obligation	Execution due
Harry Brown	Podiatrist	Examine feet	July, Oct, Jan
Mary John	Optometrist	Check eyes	December
Bob Smith	Patient	Walk 1 km	daily
		Take Diamicron	April, July, Oct

The increasing number of patients with chronic diseases and the associated medical care costs motivated us in focusing on this application area: 7 million Australians [1] and 133 million Americans [2] have a chronic medical condition and 60% of all deaths worldwide are due to chronic diseases. It is estimated that improved adherence to "care plans" (contracts among the care team and the patient) could dramatically reduce health care costs and improve patient outcomes. However, these care plans are usually not followed in practice, and few mechanisms are in place for assisting patients or the care team adhere to the care plan and avoid plan "failures" [8]. We expect our research to provide a framework for better understanding these issues and ultimately lead to mechanisms for assisting patients and their care providers better develop and adhere to care plans.

The paper is organized as follows. A framework to model contract violations due to deficits in mental attitudes is described in Section 2. Section 3 presents an adherence support architecture to reduce contract violations due to such deficits. A multi-agent implementation in chronic disease management as a proof-of-concept is discussed in Section 4. Section 5 relates our work to existing research. Concluding remarks and future work are discussed in Section 6.

2. FRAMEWORK TO MODEL CONTRACT VIOLATIONS DUE TO DEFICITS IN MENTAL ATTITUDES

We investigate a conceptual framework to demonstrate how explicit models of the mental attitudes of the human agents affect contract fulfillment. As humans are goal-directed agents, we base our investigation on the role of goals and intentions in human practical reasoning.

Bratman's notion of "future-directed intentions" [5] states that humans frequently decide in advance a plan for the future and then use such a plan as a basis for future actions. We consider the obligations agreed by contract parties as future-directed intentions for such parties (Section 2.1). In addition, the model of belief-based goal dynamics [7] proposes beliefs as the deciding factor for selecting and executing goals: humans commit to obligations based on beliefs that exist at the commitment-time and subsequent execution of the obligations depends on the execution-time beliefs of the agent (Section 2.2). Consolidating these two concepts, we propose three types of deficits in an agent's mental attitudes that may prevent agents from realising contractual obligations (Section 2.3). The contract shown in Table 1 will be used as the working example throughout the paper.

2.1 Future-Directed Intentions

According to Bratman [5], future-directed intentions, which guide the future actions of humans, are two dimensional: (1) volitional, and (2) reasoning centered. The volitional dimension states that if an agent's future-directed intention manages to survive until the time of action, and if the agent senses that the time to perform the action has arrived, and if nothing else interferes with the action, then the agent will execute the action. The reasoning centered dimension considers the roles played by future-directed intentions during the period between the initial formation of intentions and the eventual execution. It involve two reasoning components:
(1) given new information or changes in the agent's expectation, the agent may reconsider future intentions, even though it is expected that the agent settles on future-directed intentions and does not deliberate on them; and
(2) further reasoning from intended ends to the intended means as the agent moves to act on future intentions.

For example, a patient's future-directed intentions (in the form of a "care plan") could include: visit podiatrist in July, visit podiatrist in Oct, $\cdots$, visit optometrist in Dec, walk 1km daily, take 1st repeats of Diamicron in April, $\cdots$ (see Table 1). Subsequent to these commitments, the patient may:
(1) be unaware that the podiatrist appointment is at 10.00am on 1st July 2009 (forgotten or believes to be on another date).
(2) assume his current risk of heart attack is less than that at the time of commitment;
(3) get to know a football match is to be held on an appointment date;
(4) realise he cannot walk more than 0.5 km a day;
(5) discover that Diamicron causes severe side effects;
(6) decide not to take medications that have side effects; or
(7) not know how to set an appointment with the podiatrist or optometrist.

The above examples may result in the patient not complying with his care plan, thereby resulting in a service delivery failure (in this case, of the patient). The relationship between the dimensions of future-directed intentions and deficits in mental attitudes is shown in Table 2 and discussed in Section 2.3.

2.2 Belief-Based Goal Dynamics

In this section, we describe the notion of a contract in terms of belief-based goal dynamics [7] and goal-directed behaviour. In this model, beliefs are proposed as the basis for reasoning about future directed intentions and selecting and executing goals. This model categorizes beliefs into seven categories: motivating, assessment, cost, incompatibility, preference, precondition and means-end and categorises goal processing into four stages: active, pursuable, chosen and executive. These beliefs affect the goal processing in two ways:
1. Belief types intervene on goal processing at different stages, and
2. Each belief type has a unique effect on the goal.

Prior to execution, a goal gets filtered via four processing stages, depending on beliefs:

Activation stage: motivating beliefs $\Rightarrow$ active goals

Table 2: Future-directed intentions and examples.

Dimension	Sub dimension	Important concept	Example (from above)
Volitional dimension		Time has arrived to execute the action	Example 1
		Nothing interferes with the intention	Examples 2, 3, 4 and 5
Reasoning centered dimension	Settling on a certain course of actions	New information	Examples 2, 3, 4 and 5
		Change in what the agent wants	Example 6
	Further reasoning between now and time to execute intention	Reason from intended end to intended means	Example 7

Evaluation stage: assessment beliefs $\Rightarrow$ pursuable goals
Deliberation stage: cost + incompatibility + preference beliefs $\Rightarrow$ chosen goals
Checking stage: precondition + means-end beliefs $\Rightarrow$ executive goals

The obligations shown in Table 1 map to these belief types and goals. In the running example, the obligations: visit podiatrist in July, visit podiatrist in Oct, visit podiatrist in Jan, and so on, are chosen goals (committed goals) for the patient. They are transferred to executive goals based on precondition beliefs and means-end beliefs. For a chosen goal to be part of the checking process, the beliefs on which the goal was deliberated (commitment-time beliefs) have to remain unchanged until the checking stage. Any changes to such beliefs may result in a contract failure. This can be illustrated using the examples mentioned in Section 2.1:
Case (1) corresponds to a missing conditional belief about the appointment time;
Case (2) corresponds to a change in the motivating belief that indicates the risk of heart attack;
Case (3) corresponds to a value belief to be healthy that has changed from a high priority to a lower priority;
Case (4) corresponds to an incompetence belief found at the checking stage;
Case (5) corresponds to an incompatibility belief found at the checking stage;
Case (6) corresponds to a value belief to be healthy changing from a high to a lower priority; and
Case (7) corresponds to a missing means-end belief.

2.3 Deficit Categories

Both the future-directed intentions and the belief-based goal dynamics concepts differentiate between committing to a contract and contract delivery. In this section we identify the deficit types that cause such failures.

Consolidating future-directed intentions and belief-based goal dynamics, the main causes for a breach of contract are:
1. changes to the beliefs on which commitments were made; and

2. changes in agent's expectation/priority from commitment time to execution time.

We summarize such changes into three deficit categories: belief, intention and plan deficits.

2.3.1 Belief deficit

A belief deficit is said to occur if an agent forgets or does not know when to execute an action, to whom to communicate with to execute an action, or when to execute an action that is necessary for meeting a contractual obligation.

The manner in which belief deficits affects agent behaviour and contractual obligations can be described within the frameworks of Bratman and Castelfranchi. The volitional dimension of practical reasoning requires that, if the agent's future-directed intention manages to survive until the time of action and if the agent senses that the time to perform an action has arrived, the agent will execute the intentions as committed. The belief deficit that occurs when the agent forgets or does not know when to execute an action results in the agent not sensing that the time to perform the action has arrived. A different kind of belief deficit can arise when motivating beliefs change. Belief-based goal dynamics views motivating beliefs as essential for the maintenance of intentions, so that any change to such beliefs could, at a point of re-deliberation, result in the dropping of intentions that were otherwise contractually committed. We class such changes as belief deficits when the motivating belief is no longer true of the real world (or the possible world when the contractual commitment was made); otherwise, we consider the change to be an intention deficit.

Example: The patient intends to visit the podiatrist in July. He is unaware that the appointment is at 10.00am on 1st July (he either has forgotten the date or believes it to be on another day). This example can be formulated as follows.

Let I denote an intention and B a belief.
The patient commits to the plan at time *t0*. The patient's intention at time *t0* is denoted by I_{to}:
I_{to} *(if time = timeOfAppointment(podiatrist)) then visit(podiatrist)*
Status of the real world at time *t1* where $t1 > t0$ is
time = 10.00am 01/07/2009
Mental status of the patient at time *t1* is
I_{t1} *(if time = timeOfAppointment(podiatrist)) then visit(podiatrist)*
If the patient has forgotten the appointment:
$\neg B_{t1}$*(timeOfAppointment(podiatrist) = 10.00am 01/07/2009)*
or, if the patient believes the appointment to be on another day:
B_{t1}*(timeOfAppointment(podiatrist) = 10.00am 05/07/2009)*

The patient still intends to visit the podiatrist on 1st July 2009 at 10.00am as committed. That is, the patient has not changed his intention. The fact that he does not know that the visit is on 1st July (or his belief that it is scheduled on another day such as 5th July), however, means that this intention will not trigger the required action. Therefore, while the patient believes he is still consistent with his contractual obligations, the "belief deficit" may result in the patient not realising his obligations in the real world.

We note that there is a difference between forgetting about the appointment and believing it to be on another day. In the former case, the absence of a belief that is a precondition for an existing intention may trigger an action to determine the time of appointment, whereas this would not be so in the latter case. However, at our level of analysis, we class both as belief deficits.

2.3.2 Intention deficit

An intention deficit arises when an agent drops a committed intention or changes the priority of goals, resulting in a modification or a reordering of intentions so that the agent's behaviour no longer conforms with the contractual obligations.

The volitional dimension of practical reasoning requires that, for a committed intention to get executed, nothing should interfere with the intention. In addition, the reasoning centered dimension allows that, given new information or changes in the agent's expectations, the agent may reconsider future-directed intentions. These events can result in the agent dropping a committed intention or changing the priority of a committed intention.

Similarly, the incompatibility and preference beliefs that were active at the deliberation process should remain the same at the checking stage for the deliberated (committed) intentions to get executed. However, changes in these beliefs after the contractual commitment is made can result in the agent performing another reasoning process (deliberation) before acting on the intention. This subsequent deliberation process can reorder the agent's intentions such that it executes another higher priority intention while dropping the originally committed intention. In addition, belief-based goal dynamics provides that the chosen goals are transferred to executive goals based on preconditions. Therefore, if the preconditions are not satisfied, the agent will drop the chosen goal (committed intention).

Example: The patient chooses to "go to football" rather than "visit a provider" on the day he has the appointment with the provider. At commitment time, the patient may not have a belief about a football match being played on the same date as the appointment date. He acquires this new information after the commitment is made but before the appointment is due. Hence, the patient may reconsider his intention and decides to go to football. From belief-based goal dynamics, this example can be viewed as a change of preference belief after the deliberation is made at the commitment time. The preference belief type that is activated in this case can be an urgency belief that provides an expiry context to the goal of going to football; for example, that it is the last day of football season. As a result, the patient may be persuaded to go to football rather than visit the podiatrist.

2.3.3 Plan deficit

A plan deficit arises when the agent does not know the means for carrying out an intention or achieving a goal that is necessary for fulfilling the contractual obligations.

Once the commitment or the future-directed intention is made, the party fails to reason from intended end to intended means. According to belief-based goal dynamics, the chosen goal (commitment) fails at the checking stage due to a lack of means-end beliefs. Hence, the chosen goal will not get selected as an executive goal, resulting in a breach of the contract.

Example: The patient intends to set an appointment with the podiatrist. But he does not know how to proceed with setting an appointment. That is, the patient does not have

Table 3: Mapping among future-directed intentions, belief-based goal dynamics and deficit categories.

Dimension	Sub dimension	Concept	Mapping belief type	Deficit
Volitional dimension		Time has arrived to execute the action	Motivating beliefs	Belief deficits
		Nothing interferes with the intention	Incompatibility + Preference + Precondition	Intention deficit
Reasoning centered dimension	Settled on a course of action	New information	Incompatibility + Preference + Precondition	Intention deficit
		Change in what I want	Incompatibility + Preference + Precondition	Intention deficit
	Further reasoning between now and time to execute intention	Reason from intended end to intended means	Means-end beliefs	Plan deficit

a plan for setting an appointment (e.g., by calling the podiatrist office in a timely way) and thereby cannot determine a means to achieve the desired end.

The mapping between future-directed intentions, belief-based goal dynamics and deficit categories is shown in Table 3.

3. FRAMEWORK FOR ADHERENCE SUPPORT

In this section, we propose an adherence support architecture to reduce contract violations resulting from the three types of failures described in Section 2.3. Our proactive failure prevention strategy consists of three parts:

- the detection of possible deficits;
- the recognition of the possible deficit type; and
- intervention to eliminate or ameliorate the deficits.

3.1 Detection of Possible Deficits

The detection of possible deficits is carried out with the notions of:

The execution of an obligation as outlined in the contract usually depends on the successful execution of a priori event(s) or achievement of certain intermediate contract state(s). The occurrence or non-occurrence (depending on the way they are defined) of these a priori events and states can be taken as an indicator of the likely success of the agent meeting the contractual obligations.

Such a priori events or contract states are called precursors. That is, a precursor is an event, a pattern of events, a state, a pattern of states, or a combination of state and event patterns that has a positive or negative influence over achieving contract obligations as planned. Precursors are either time dependent (e.g., setting an appointment prior to a provider visit) or time independent (the patient's Systolic blood pressure > 180 or diastolic blood pressure > 110)[3]. Depending on the way precursors are defined, an occurrence of a favourable precursor indicates that the contract is currently been executed as planned. An unfavourable precursor indicates a possible impending contract violation which may be associated with any deficits in contract parties' mental states.

Each obligation is associated with one or many domain specific precursors. Let O denote the set of obligations and P the set of precursors. The mapping between an obligation $o \in O$ and a precursor $p \in P$ is usually context dependent and can be represented by a branching tree structure where the context determines the selection of a branch. The context defines the current state of the contract (or the system or the environment). We represent this mapping using the notion of an *obligation-precursor-operator*: $OPO \subseteq O \times BoolExp \rightarrow P$, which associates an obligation $o \in O$ with one or more boolean expression $C \in BoolExp$ and for each of these maps uniquely to a precursor $p = OPO(o, C)$ with $p \in P$. We represent an obligation-precursor-operator instance as a triple $< o, C, p >$.

Once OPO is defined, the (non-)occurrences of precursors are detected at execution-time by applying a context sensitive goal-directed algorithm:

repeat
  envContext := getEnvironmentContext
  precursorsToDetect := selectMatchingPrecursors(envContext)
end repeat

The function *selectMatchingPrecursors* is of the form:

selectMatchingPrecursors(envContext)
  let $A = \{\}$
  for each $a, a \in OPO$
    if $(C = envContext)$
      let $p \in A$
  return A

Once the precursors are detected, the next step in failure prevention is to identify the reason for such (non-)occurrences of these precursors.

3.2 Mental-State Recognition to Identify Possible Deficits

If a favourable precursor $P_f, P_f \in P$ occurs further investigations are not necessary as the P_f indicates that the contract is currently been executed as planned. We define the set of unfavourable precursors as P_{uf} where $P_{uf} \in P$. In the absence of malicious and self-interested contract parties, elements in P_{uf} occur as a result of "deficits" (as we have defined them) in the contract party's mental attitude(s). The corresponding deficits prevent the party from carrying out an action required for the successful fulfilment of the contract.

In general, it is important to be able to recognise the type of deficit (i.e., a belief, intention or plan deficit) so as to be able to intervene effectively. To do this requires that we be able to recognise or otherwise characterise the mental state of the "offending" agent.

In our proposed framework, we assign an adherence agent (AA) to each contract party. The role of the AA is, in part, to recognise the mental states of the contract parties. (This role is similar to that of observer agents in some mental-state recognition research [15]). Each AA has knowledge of the obligations of the corresponding contract party. Each AA uses domain-specific processes to represent the mental state recognition stratetegies for each precursor. The adherence agents can adopt a combination of intrusive [10] and overhearing [11] agent behaviour monitoring approaches to determine the context of the contract (environment/system). At the end of mental-state recognition process, each AA associates a missing precursor, p to a deficit, d, where d can be a belief, intention or a plan deficit. Once the deficit is

identified, a tailored intervention strategy can be applied to try to reverse the deficit.

3.3 Intervention to Reduce Deficits

The aim of an intervention is to compensate for any missing favourable precursors, P_f (the missing event(s), state(s) or the pattern of events and states) or to reverse the effect of unfavourable precursors that have occurred, P_{uf}. Let I denote the set of interventions. The mapping between the obligations and the precursors, the mapping between precursors and obligations depends on the context of the contract at the time of precursor detection and mental-state recognition. As before, we represent this dependence using a branching tree structure defined by a *precursor-intervention-operator*: $PIO \subseteq P \times BoolExp \rightarrow I$, which associates a precursor $p \in P$ with one or more conditions C and for each of these with a unique intervention $i = PIO(p, C)$ where $i \in I$. We represent a precursor-intervention-operator instance as a triple $< p, C, i >$.

Once the PIO is defined, the applicable intervention strategy for a detected precursor is obtained by applying the following context sensitive goal-directed algorithm:

applicableIntervention := selectIntervention(detectedPrecursor, environmentContext)

The function *selectIntervention* is of the form:

```
selectIntervention(detectedPrecursor, environmentContext)
  let A = {}
  for each a, a ∈ PIO
    if (p = detectedPrecursor)
      if (C = environmentContext)
        let i ∈ A
  return A
```

If multiple interventions correspond to a given *detectedPrecursor* and *environmentContext*, *selectIntervention* function returns a set with multiple items. In such situations, a single intervention is identified based on effectiveness and cost.

The process of detecting a possible deficit, recognizing the mental state that caused the deficit, and intervening to change the mental-state and ameliorate the deficit is summarized in the following algorithm.

```
repeat
  envContext := getEnvironmentContext
  precursorsToDetect := selectMatchingPrecursors(envContext)
  if (((occurred(precursorsToDetect)) and
  (isFavourable(precursorsToDetect))) or
  ((not(occurred(precursorsToDetect))) and
  (not(isFavourable(precursorsToDetect)))))
    doNothing
  if (((not(occurred(precursorsToDetect))) and
  (isFavourable(precursorsToDetect)))
  or ((occurred(precursorsToDetect)) and
  (not(isFavourable(precursorsToDetect)))))
    applicableIntervention :=
    selectIntervention(precursorsToDetect, envContext)
    intervene using applicableIntervention
end repeat
```

4. IMPLEMENTATION

A multi-agent implementation was carried out for chronic disease management in health care as a proof-of-concept for a complex customer care management system. The system consists of two types of agents:

- Contract party agent (CPA): Each contract party is assigned a CPA that acts as a personal assistant to the party. Contract parties communicate with the system via the corresponding CPA. Any deficits that occur in service delivery stage is captured through CPAs; and
- Adherence agent for each contract party (AA): Each CPA has a corresponding AA for adherence support. AAs are implemented according to the theory in Section 3.

In chronic disease management, the customers (patients) and the health care providers (service providers) commit to maintain and deliver services (an example is shown in Table 1). We discuss the patient's commitment to visit the care providers within the due time frame in detail and provide screen outputs for managing the repeat prescriptions and maintaining measurements such as blood glucose and blood pressure within the applicable range [3].

4.1 Service Delivery Specification: Domain-Based Agent Specification

Considering the distinct functionalities associated with the patient and the care providers, we consider two specific CPAs, the $CPA_{patient}$ and $CPA_{careprovider}$. Each CPA is assigned a corresponding Adherence Agent (AA). The service delivery support specification defines the data and processes of $CPA_{patient}$, $CPA_{careprovider}$, $AA_{patient}$ and $AA_{careprovider}$. The AA specification defines the obligations in the care plan, the current state of the obligations and the failure prevention definitions. The specification of the CPAs contains contract obligations as data and processes to perform such obligations and to receive intervening messages and reply to them. In the current implementation, the service delivery support specification is specified as part of the agent code. The agents are implemented using `Jason` [4], an interpreter for AgentSpeak(L), based on the BDI architecture [16].

To uniquely identify patients and care providers, patients are assigned an Electronic Health Record (EHR) identifier prefix with "c"(for customer) and the health providers are assigned with a unique provider identification number (PIN) prefix with "p" (for provider). Bob (patient) as CPA_{c1}, Harry (podiatrist) as CPA_{p1}, Mary (optometrist) as CPA_{p2}, and the corresponding adherence agents AA_{c1}, AA_{p1} and AA_{p2}.

The care plan obligations of the patient are represented as beliefs of the $CPA_{patient}$:
1. visit(?PIN, ?Month, ?Year)
2. exercise(?Exercise, ?Frequency)
3. renewPrescription(?Medication, ?Month, ?Year)

Examples: visit(p1,07,2009), visit(p1,10,2009), visit(p1,01,2009), visit(p2,12,2009), exercise(walk1km, daily), renewPrescription(Diamicron, 04, 2009), renewPrescription(Diamicron, 07, 2009), renewPrescription(Diamicron, 10, 2009)

Specifications of care plan obligations of $AA_{patient}$: These specifications include
1. All the belief structures of $CPA_{patient}$;

2. The additional beliefs required for the successful execution of the contractual obligations. For example, to visit a provider, the patient has to set an appointment a certain number of days prior to the planned visit. The number of days may vary with the provider. The belief waitingTime(?PIN, ?Days) defines the number of days in advance an appointment has to be set with the provider, ?PIN, e.g., waitingTime(p1, 28); and
3. The current state of the obligations. For example, appointment(?PIN,?Month, ?Year) states that an appointment has been set with the provider, ?PIN for the month, ?Month and the year, ?Year.

Specifications for failure prevention: These specifications include obligations to precursor operators (OPO) (Section 3.1), precursor detection strategies, mental state recognition processes (Section 3.2), and precursors to intervention operators (PIO) (Section 3.3).

Example: Failure prevention specification for the patient's commitment to planned visits.
An OPO defines that the patient has to set an appointment with the provider within a specified waiting time:

obligation: visit(?PIN, ?Month, ?Year)
context: waitingTime(?PIN, ?NumberofDays) and
(= DifferenceinDays(?Month ?Year) ?NumberofDays)
precursor: appointment(?PIN, ?Month, ?Year)

Note: The DifferenceinDays is a function definition which returns the number of days between the current date and the first day of ?Month and ?Year.

Specification for detecting the above precursor:

goal: CheckForAppointment(?PIN, ?Month, ?Year)
context: waitingTime(?PIN, ?NumberofDays) and
(= DifferenceinDays(?Month ?Year) ?NumberofDays)
body: if (¬ (appointment(?PIN, ?Month, ?Year) then
DetectDeficitOfSetAppointment(?PIN, ?Month, ?Year)

The body contains one or many goals required to be executed to achieve the main goal. The main goal *CheckForAppointment* will be a goal for $AA_{patient}$ to execute each day, but the successful execution depends on the context.

Specification of mental-state recognition processes (MSRP) to detect the type of deficit associated with the patient:
MSRP1
goal: DetectDeficitOfSetAppointment(?PIN, ?Month, ?Year)
context: true
body: RequestWaitingTime(?PIN)

MSRP1 requests the waitingTime for the provider ?PIN from the patient. The patient's response get stored as a belief waitingTimeFromPatient(?PIN, ?Days).

MSRP2
goal: DetectDeficitOfSetAppointment(?PIN, ?Month, ?Year)
context: ?waitingTime(?PIN, ?NumberofDays) and
waitingTimeFromPatient(?PIN, ?Days) and
(= ?NumberofDays ?Days)

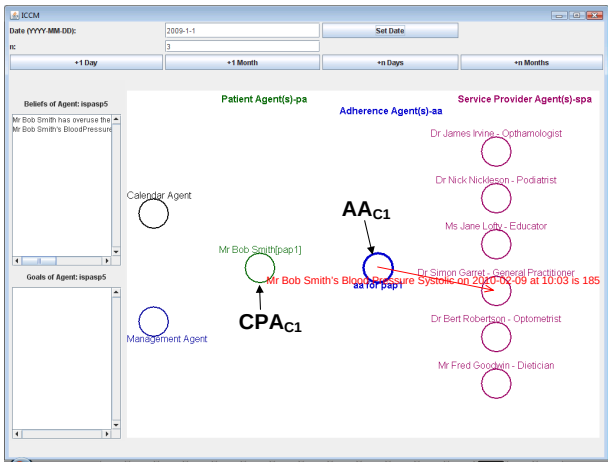


Figure 2: Adherence agent sends a measurement alert to a care provider.

body: GetRiskLevelFromPatient(heartattack)

MSRP2 requests the heart attack risk level from the patient. The patient's response get stored as a belief riskLevelFromPatient(heartattack, ?PatientLevel).

The execution outcome of MSRP1 can be that the patient's belief about the waiting time is incorrect: waitingTimeFromPatient(?PIN, ?Days) and waitingTime(?PIN, ?NumberofDays) and (≠ ?Days ?NumberofDays) → belief deficit.

The execution outcome of MSRP2 can be that the patient's belief on his risk of a heart attack is less than the actual value:
riskLevelFromPatient(heartattack, ?PatientLevel) and
riskLevel(heartattack, ?ActualLevel) and
(≠ ?PatientLevel ?ActualLevel) → intention deficit

Specifications of precursor-intervention-operators (PIO):
PIO1
precursor: appointment(PIN, ?Month, ?Year)
context: waitingTimeFromPatient(?PIN, ?Days) and
waitingTime(?PIN, ?NumberofDays) and
(≠ ?Days ?NumberofDays)
intervention: remindToSetAppointment(?PIN, ?Month, ?Year)

PIO2
precursor: appointment(PIN, ?Month, ?Year)
context: riskLevelFromPatient(heartattack, ?PatientLevel)
and riskLevel(heartattack, ?ActualLevel)
and (≠ ?PatientLevel ?ActualLevel)
intervention: remindToSetAppointment(?PIN, ?Month, ?Year)
informCurrentRisk(heartattack, ?ActualLevel)

Similar domain-specific failure prevention strategies are specified for managing repeat medications and maintaining measurements within predefined limits, For those we present a screen from the implementation:
If at any time the patient's blood glucose (or other measurement) is outside the acceptable range, AA_{c1} alerts the care providers (Figure 2).

5. RELATED RESEARCH

Our aim in this research is to study means for reducing service delivery failures resulting from "deficits" in the mental attitudes of human contract parties. As such, our interest is in predictive monitoring approaches rather than reseach in replanning, plan repair and plan failure recovery, in which remedial actions are considered after a plan failure occurs. Current predictive monitoring frameworks [11] use detection followed by an intervention strategy to avoid contract violations. We argue that when contract parties are autonomous human agents, an intervention does not succeed unless the reason for the possible violation is determined. The reason for the possible violation is encapsulated within the mental attitudes of the agent; that is, the mental state of the contract parties plays a major role here. If the detection and intervention mechanisms ignore the agent's mental state, the intervention may be ineffective. Therefore, for an intervention to be effective, the monitoring framework should capture and interpret the mental attitudes of contract parties. The proposed deficit recognition strategy aims to capture this intermediate layer between detection and recognition.

In addition, the importance of mental-state recognition is already realised in health care. For example, the diabetes management program offered by American Healthways [13] uses the trans-theoretical model of behaviour change [14] to identify patients' readiness for health care interventions. This approach determines how amendable a patient is to making a lifestyle change at a particular point in time, which in turn helps to tailor the selected intervention.

From the perspective of electronic contract formation and management, the CONTRACT project [12] is the most recent and relevant. In this framework, each contract is associated with two types of states: Critical States (CS) and Danger of Violation (DOV) states. CS define the states which are compulsory for the successful execution of contract. That is, at the service delivery stage, if a CS state does not occur, it is identified as a violation of the contract. The DOV states indicate a possible violation of the contract. The DOV states are not explicitly states in the contract. Our definition of precursors (Section 3.1) is similar to the DOV states. The extension in our work is that once a DOV (or absence of a precursor) is identified, we carry out a mental-state recognition process to identify the deficit which prevented the occurrence of the precursor. The mental-state recognition process help to select a tailored and more effective intervention.

6. CONCLUSIONS AND FUTURE WORK

This paper presents the complexities introduced by human contract parties in meeting service delivery objectives in customer life cycle management. It describes the effect of deficits in mental attitudes on the successful delivery of services involving goal-directed agents. It also proposes an adherence support architecture either to reverse such deficits or to identify reasons for revising contractual obligations.

Currently we are in the process of establishing an understanding of the adherence support architecture through theory and simulations. To do so, we associate costs and benefits with contract failures and successes, as well as costs associated with precursor recognition, deficit recognition and intervention. We aim to use the theoretical findings to establish general principles of intervention and the simulation to investigate more complex scenarios and other problem domains.

7. REFERENCES

[1] agpn. Chronic disease management, 2007. Accessed: 12. November 2007.

[2] G. Anderson and K. Wilson. Chronic disease in california: Facts and figures, 2006.

[3] D. Australia and RACGP. Diabetes management in general practice. guidelines for type 2 diabetes. 2009.

[4] R. Bordini, J. Huebner, and M. Wooldridge. *Programming Multi-Agent Systems in AgentSpeak using Jason*. Wiley New York, 2006.

[5] M. E. Bratman. *Intentions, Plans, and Practical Reason*. Harvard University Press, 1987.

[6] D. Bu, E. Pan, D. Johnston, J. Walkler, J. Adler-Milstein, D. Kendrick, J. Hook, C. Cusack, B. D.W., and B. Middleton. The value of information technology-enabled diabetes management. *Healthcare Information and Management System Society (HIMSS)*, 2007.

[7] C. Castelfranchi and F. Paglieri. The role of beliefs in goal dynamics: Prolegomena to a constructive theory of intentions. *Synthese*, 155(2):237–263, 2007.

[8] M. Georgeff. E-health and the Transformation of Healthcare, 2007.

[9] C. Guttmann, I. Thomas, M. Georgeff, K. Wickramasinghe, H. Gharib, S. Thompson, and H. Schmidt. Towards an intelligent agent framework to manage and coordinate collaborative care. In *Proc. First CARE 2009*, LNCS. Springer-Verlag, accepted in 2009, to appear in 2010.

[10] B. Horling, B. Benyo, and V. Lesser. Using self-diagnosis to adapt organizational structures. In *Proceedings of AAMAS*, page 536. ACM, 2001.

[11] G. Kaminka, D. Pynadath, and M. Tambe. Monitoring teams by overhearing: A multi-agent plan-recognition approach. *JAIR*, 17(1):83–135, 2002.

[12] N. Oren, S. Miles, M. Luck, S. Modgil, N. Faci, S. Alvarez, J. Vazquez, and M. Kollingbaum. Contract based electronic business systems theoretical framework. Technical Report D2.2, King's College London, 2008.

[13] J. Pope, L. Hudson, and P. Orr. Case study of American Healthways' diabetes disease management program. *Health care financing review*, 27(1):47, 2005.

[14] J. Prochaska and C. DiClemente. Toward a comprehensive model of change. *Treating addictive behaviors: Processes of change*, pages 3–27, 1986.

[15] A. Rao. Means-end plan recognition-towards a theory of reactive recognition. *Proc PKRR*, pages 497–508, 1994.

[16] A. Rao. Agentspeak(L): BDI agents speak out in a logical computable language. In *Proceedings of MAAMAW 96*, volume LNAI Vol. 1038, pages 42–55, London, 1996. Springer-Verlag.

[17] K. Wickramasinghe, C. Guttmann, M. Georgeff, H. Gharib, I. Thomas, S. Thompson, and H. Schmidt. Agent-based intelligent collaborative care management. In *Proc. AAMAS-Volume 2*, pages 1387–1388. IFAAMS, 2009.

Using A Normative Framework to Explore the Prototyping of Wireless Grids

Tina Balke
Chair of Information Systems Management
University of Bayreuth
tina.balke@uni-bayreuth.de

Marina De Vos, Julian Padget
Dept. of Computer Science
University of Bath
{mdv,jap}@cs.bath.ac.uk

Frank Fitzek
Multimedia Information and Signal Processing
University of Aalborg
ff@es.aau.dk

ABSTRACT

The capacity for normative frameworks to capture the essential features of interactions between components in open architectures suggests they might also be of assistance in an early, rapid prototyping phase of system development, helping to refine concepts, identify actors, explore policies and evaluate feasibility. As an exercise to examine this thesis, we investigate the concept of the wireless grid. Wireless grids have been proposed to address the energy issues arising from a new generation of mobile phones, the idea being that local communication with other mobile phones, being cheaper, can be used in combination with network communication to achieve common goals while at the same time extending the battery duty cycle. This results in a social dilemma, as it is advantageous for rational users to benefit from the energy savings without any contributing to the cooperation, as every commitment has its price. We present a necessarily simplified model, whose purpose is to provide us with the foundation to explore issues in the management of such a framework, policies to encourage collaborative behaviour and the means to evaluate the effects on energy consumption.

Categories and Subject Descriptors

I.2.4 [**Computing Methodologies**]: Artificial Intelligence—*Knowledge Representation Formalisms and Methods, Distributed Artificial Intelligence*

Keywords

Wireless Grids, Answer Set Programming, Norms, Reciprocity

1. INTRODUCTION

This paper reports on a feasibility study into how and whether institutional models can help in evaluating the concept of wireless grids. While that is the specific topic of the paper, the broader contribution is that of asking the question of how such normative model building can be of use in an early design phase, long before hardware or software is available, in order to evaluate both principles and alternative policies — that might have significant consequences subsequently.

In technology neutral terms, the problem we consider is of some digital content to be distributed to a collection of nodes that support an expensive (in terms of power and money) connection via a structured network and a cheaper connection via an ad-hoc network. The task is to minimise the cost of the distribution of this digital content by using a combination of the structured and ad-hoc networks. The model can essentially be parameterised by the cost functions for the (un)structured network technology. The particular case that interests us is the forthcoming 4G mobile phone network where the structured network uses a traditional cellular link and the ad-hoc network uses IEEE 802.11 (wireless LAN) with the ethernet transport protocol. The motivation for the idea of such a "wireless grid" is that local communication over (wireless) ethernet uses significantly less power than communicating with the network basestation and that duration of battery duty cycle is a major usability factor for users.

The deployment of third generation (3G) of mobile network systems is in progress, but a quite different next generation network (called Fourth Generation or 4G) is under development that is intended to cause a paradigm shift in the cooperation architecture of wireless communication [14]. While for 3G the industry focused on technology for enabling voice and basic data communications (technology-centric-view), the emphasis in 4G is more user-centric [24]. Consequently, studies to find possible drivers for consumer demand for mobile devices, such as the one by TNS [21] across 15 countries in mid-2004, have been conducted. This study revealed that it was not high performance that was attractive to consumers, but rather useful, convenient and enjoyable services coupled with ubiquitous infrastructures for constant connection. In addition, "two days of battery life during active use" topped the wish list of key features in 14 of the 15 countries surveyed.

Batteries have fixed capacity that puts limits on the operational time for a device in one charge cycle. The increasing sophistication of mobile phones and their evolution into smart phones offering Internet access, imaging (still and video), audio and access to new services, has had a significant impact on power consumption, leading to shorter stand-by times, as well as the problem of rising battery temperature unless there is active cooling [19].

Fitzek and Katz [9] have proposed a way around some of these issues with the concept of a wireless grid, in which users share resources in a peer-to-peer fashion that uses less power but this requires a difficult to obtain collaboration between the users. The contribution of this paper is to build an institutional model of the interactions between handsets and basestation and between handsets in order to provide a foundational model from which to be able to explore policies, identify suitable sanctions and evaluate potential gains from reduced power consumption.

The remainder of the paper is structured as follows: in the next section (2) we cover three aspects of the background, namely (i) normative frameworks, (ii) a detailed discussion of the wireless grid scenario, and (iii) the energy model: what agent actions cost in terms of power consumption. Then, in section 3 we describe the action model—what the agents may do—before presenting some results from its analysis. We conclude in section 4 with a discussion of the related work, results and future directions.

2. TECHNICAL CONTEXT

The first section here serves to provide a brief description of the event-based normative framework that is used later for the model. The second provides a detailed description of the some technical issues surrounding the wireless grid idea, highlighting in particular, actual energy costs and the risk of free-loading, which latter has some elements that echo issues with common-pool resource problems.

2.1 Normative Frameworks

The concept of the normative framework—sometimes also called an institution, sometimes a virtual organisation—has become firmly embedded in the agent community as a necessary foil to the essential autonomy of agents, in just the same way as societal conventions and legal frameworks have been developed to constrain people. In both the physical and the virtual worlds—and the emerging combination of the two—the arguments in favour centre on the minimization of disruptive behaviour and supporting the achievement of the goals for which the normative framework has been conceived and thus also the motivation for submission to its governance by the participants.

While the concept remains attractive, its realization in a computational setting remains a subject for research, with a wide range of logics [1, 4, 6] and tools [20, 22, 12], to cite but a few. We do not include an extensive and detailed case for the purpose and value of normative frameworks here—this can be found in [23, 5], for example.

2.1.1 Formal Model

To provide context for this paper, we give an outline of a formal event-based model for the specification of normative frameworks that captures all the essential properties, namely empowerment, permission, obligation and violation. Extended presentations can be found in the citations above.

The essential elements of our normative framework are: (i) events ($\mathcal{E}$), that bring about changes in state, and (ii) fluents ($\mathcal{F}$), that characterise the state at a given instant. The function of the framework is to define the interplay between these concepts over time, in order to capture the evolution of a particular framework through the interaction of its participants. We distinguish two kinds of event: normative events ($\mathcal{E}_{inst}$), that are the events defined by the framework and exogenous ($\mathcal{E}_{ex}$), that are outside its scope, but whose occurrence triggers normative events in a direct reflection of the "counts-as" principle [13]. We further partition normative events into normative actions ($\mathcal{E}_{act}$) that denote changes in normative state and violation events ($\mathcal{E}_{viol}$), that signal the occurrence of violations. Violations may arise either from explicit generation, from the occurrence of a non-permitted event, or from the failure to fulfil an obligation. We also distinguish two kinds of fluent: *normative fluents* that denote normative properties of the state such as permissions, powers and obligations, and *domain fluents* that correspond to properties specific to the normative framework itself.

The evolution of the state of the framework is achieved through the definition of two relations: (i) the generation relation: this implements counts-as, in that it specifies how the occurrence of one (exogenous or normative) event generates another (normative) event, subject to the empowerment of the actor. Formally, this can be expressed as $\mathcal{G} : \mathcal{X} \times \mathcal{E} \rightarrow 2^{\mathcal{E}_{inst}}$, where $\mathcal{X}$ denotes a formula over the (normative) state and $\mathcal{E}$ an event, whose confluence results in an institutional event, and (ii) the consequence relation, that specifies the initiation and termination of fluents subject to the performance of some action in a state matching some expression, or formally $\mathcal{C} : \mathcal{X} \times \mathcal{E} \rightarrow 2^{\mathcal{F}} \times 2^{\mathcal{F}}$.

Again, for the sake of context, we summarize the semantics of our framework and cite [7] for an in-depth discussion. The semantics is defined over a sequence, called a trace, of exogenous events. Starting from the initial state, each exogenous event is responsible for a state change, through initiation and termination of fluents, that is achieved by a three-step process: (i) the transitive closure of $\mathcal{G}$ with respect to a given exogenous event determines all the (normative) events that result (ii) to this we add all violations of events not permitted and all obligations not fulfilled, giving the set of all events whose consequences determine the new state, so that (iii) the application of $\mathcal{C}$ to this set of events, identifies all fluents to initiate and terminate with respect to the current state in order to obtain the next state. So for each trace, we can obtain a sequence of states that constitutes the model of the normative framework. As with human regulatory settings, normative frameworks become useful when it is possible to *verify* that particular properties are satisfied for all possible scenarios. In order to do so, we need to incorporate a computational model in our formal representation.

2.1.2 Implementation

This formalisation is realized as a computational model [7] through Answer Set Programming [3, 11] and it is this representation that is the subject of the evaluation process described in Section 3.2. In [7] it was shown that the formal model of an normative framework could be translated to an $AnsProlog$ [7] program, a logic program under the answer set semantics, such that the answer sets of the program correspond to the traces of the framework. A detailed description of the mapping can be found there.

$AnsProlog$ is a knowledge representation language that allows the programmer to describe a problem and the requirements on the solutions in an intuitive way, rather than the algorithm to find the solutions to the problem. Programs consists of rules interpreted under the answer set semantics. Answer set solvers like CLASP [10] or SMODELS [16] process the $AnsProlog$ specification and return the solutions, in this case the traces, as answer sets. Answer set programming, a logic programming paradigm, permits, in contrast to related techniques like the event calculus [15] and C+ [8], the specification of both problem and query as an executable program, thus eliminating the gap between specification and verification language. But perhaps more importantly, both languages are identical, allowing for more straightforward verification and validation.

A level of abstraction can be added using a domain-specific action, InstAL[7], and query language, InstQL[12], which can be both translated in the $AnsProlog$ in order to specify not only the valid traces but those that exhibit features of interest. We use InstALto describe our scenario in Section 3. The action language uses semi-natural language to describe the various components of the normative framework and allows type definitions to avoid grounding problems when translating to $AnsProlog$. For example, events are defined by `typeOfEvent event namOfEvent;` with type being one of `exogenous`, `create`, `inst` or `violation`, while fluents are defined by `fluent nameofFluent(ParameterType, ...);`. Generation of normative events from exogenous events is specified using the `generates` statement, while `initiates` and `terminates` define the two parts of the consequence relation. Conditions on the state are expressed using `if`. The `initially` statement serves to specify the set of fluents that characterise the initial state after the normative framework is created. For our model we are interested in all traces that lead to success, so we did not require the additional facilities of the query language InstQL. Instead we specified the fluents or events we wanted to show or hide directly in $AnsProlog$ using the special syntax `#show` and `#hide`.

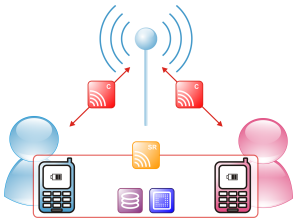


Figure 1: Wireless Grid Communication Architecture

2.2 The Wireless Grid Scenario

2.2.1 *The Wireless Grid Architecture*

As described in the introduction, to overcome the energy problems of 4th generation mobile phones, Fitzek and Katz [9] proposed the establishment of *wireless grids* as shown in Figure 1 [9].

In these wireless grids, ubiquitous mobile devices with potentially different capabilities are expected to create ad-hoc connections and to cooperate and share their limited resources for the benefit of the community. The cooperation between the mobile devices thereby is enabled with the help of a short range communication link, such as WLAN or Bluetooth. Compared to the traditional cellular HSDPA communication with the base-station, the advantage of the short-range communication is much higher bandwidth while using much less power, as it will be shown section 2.2.2. Thus, for example the battery and CPU power needed on the short link is significantly lower than it would be needed on the cellular [19].

Looking at the implementation side of the short range connection several options are available as today's mobile devices are equipped with several local area network technologies. In this paper we will focus on the IEEE802.11 WLAN specification, that allows for mobile devices communicating directly with each other and according to Perrucci et al. [19] has the highest energy saving potential.

For a better understanding of the wireless grid idea we will now briefly present a scenario that we can refer back to later. This scenario is set in a football stadium: while watching one game, the fans are very likely to be interested in games that take place at the same time at another place. As they cannot watch two games at the same time, they might use mobile phones in order to get information about other games. A likely problem for the infrastructure provider is that once a goal has been scored in another game, fans want to watch the other goal on their mobile phones and all try to stream the video file from the base station at the same time, thus overloading it. The bandwidth of the base station connection is divided into several channels that are sent out sequentially within one time frame. Thereby — up to a certain technical maximum — each mobile phone is allocated one slot. As the total bandwidth of a base station is fixed, the more mobile phone users are given a slot, the smaller the bandwidth that can be assigned to each channel gets. As a result the download times increase, leading both to more battery consumption and lower quality in the streaming service.

In contrast to the normal "non-cooperative" scenario in which a single mobile phone user would need to receive all sub-streams over the cellular link resulting in the above mentioned problems, using the cooperation envisioned in the wireless grid scenario, users could share the task by receiving a subset of the multicast channels over the cellular link from the base station and exchanging the missing pieces over the short range link.

2.2.2 *The Energy Advantage in IEEE802.11*

To understand the IEEE802.11 WLAN wireless grid scenario and its energy implications better, this section will focus on the technical aspects of the scenario (especially the WLAN transmission) in more detail. We thereby denote A to the set of agents in the scenario. When thinking about the energy implications of the wireless grid scenario, the following basic definition of energy $[E]$ should be kept in mind, which states that energy consumption in terms of battery depends on two aspects: the power $[P]$ consumed per connection type and the time $[t]$ needed for the actual transmission:

$$Energy = Power * Time\ [Joules] \quad (1)$$

So what is the energy consumption in this scenario? The total energy consumption is the energy consumed over the tradition cellular 3G connection (E_{3G}) plus that over the short link (i.e. WLAN) connection (E_{WLAN}). In case of no cooperation the latter costs 0, i.e. it is assumed that the WLAN connection is turned off and the football fan has to stream the complete video using the 3G connection. In case of wireless grid cooperation it is assumed that both connections (WLAN and 3G) are turned on and the devices help one another in a peer-to-peer-like fashion. Assuming $|A_{Coop}|$ cooperating agents in the scenarios for example, each agent only needs to stream only a part of the total video from the base station (i.e. $\frac{1}{|A_{Coop}|}$ in an ideal scenario) and obtain the missing chunks from the other cooperation partners using the short link connection. Therefore the energy consumption in the cooperation case (E_{Coop}) comprises:

1. The energy consumed for streaming part of the video from the base station using the 3G link ($E_{3G,rx}$) (plus the energy consumed while the 3G connection is idle ($E_{3G,i}$)),
2. The energy consumed for receiving the remaining chunks of the video on the WLAN connection ($E_{WLAN,rx}$),
3. The energy consumed for sending the own chunks to the other participants via the WLAN connection ($E_{WLAN,tx}$), and
4. The energy consumed in idle phases (i.e. when not transmitting or receiving anything but waiting for the next interaction) ($E_{WLAN,i}$).

Taking into account equation 1, by replacing the E with the respective $P * t$-values, one can analyse the power consumption as well as the transmission times of the scenario in the cooperative and non-cooperative case in detail. Representative power and time values for the transmission in the different states using 3G and WLAN connection can be found in [19, p.D10] for example, which are based on measurements from a Nokia N95. These numbers indicate that although the power needed for the WLAN and the 3G state are about the same, the data rate for the 3G link (0.193 Mbit/s for the receiving state) is significantly lower than that of WLAN (5.115 Mbit/s, receiving state, 30m distance) leading to significantly worse transmission times and consequently a much worse energy per bit ratio for the 3G link. The energy consumed in the idle states is not significant and therefore neglected in this paper.

This suggests that the cooperation scenario has a significant advantage in energy consumption, compared to the conventional cellular communication architecture, especially if the number of cooperating mobile phones is high and a large proportion of the data transmission can be done via the short-link connection.

2.2.3 *The Reciprocity Problem in Wireless Grids*

Although the wireless grid scenario shows huge advantages with regard to the battery consumption, it also has the intrinsic weakness of distributed cooperative architectures: it relies on cooperation to succeed. The cooperation idea in the wireless grid, as shown in figure 2(a), is as follows:

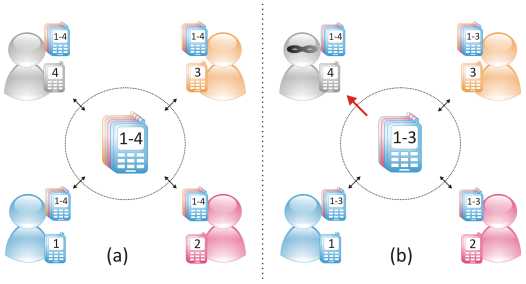


Figure 2: The Reciprocity Problem in Wireless Grids

1. The participants volunteer their resources, forming a common pool which can be used by all of them in order to achieve a common goal, such as file streaming. The utility which users can obtain from the pooled resources is much higher than they can obtain on their own. For example, in the football stadium scenario,both download time and battery consumption are reduced. However, the problem is that commitment comes at a cost, in the form of battery consumption for sending file chunks, i.e. $E_{WLAN,tx}$. As a consequence, (bounded) rational users would prefer to access the resources without any commitment of their own, as shown in figure 2
2. Thus, as shown in (b), the grey agent in the top left corner (with blindfold) can enjoy the full benefits from the common pool without committing anything itself, hence cheating on the three other agents.

However, if a substantial number of users followed this selfish strategy, the network itself would be at stake, depriving all users of the benefits [17]. The reason for this is straightforward: network users can have strategic behaviour and are not necessarily obediently cooperating by making their resources available without the prospect of rewards for their good behaviour. Unreciprocated, there is no inherent value to cooperation for a user. A lone cooperating user draws no benefit from its cooperation, even if the rest of the network does. Guaranteed cost paired with uncertainty or even lack of any resulting benefit does not induce cooperation in a (bounded) rational, utility-maximising user. Without any further incentives, rational users therefore would not cooperate in such an environment and all be worse off than if they cooperated [2].

2.2.4 *The Energy Model*

Utility quantification is being used by the (bounded rational) agents (i.e. agents that only have partial information about their environment, including other agents) to determine the utility of the different possible actions and choose their actions in such as way that maximises their utility. Concerning their knowledge that they can rely on when calculating utilities, we assume the agents to not have knowledge of the whole system, but only a small part of it within their vicinity.

We are going to explain how the agents determine the utility of an action by using the football stadium scenario described earlier. However, to keep the example simple, for the utility considerations we will concentrate on the interaction of two agents only and formulate the considerations in such a way that they can easily be expanded to any number of agents.

The agents we envision, both want to stream the same file G in the stadium. In order to get the complete file, they can cooperate and thereby reduce their energy consumption or stream the file themselves using a cellular link connection. The exchange is done in chunks ($g \in G$).

As described above, the issue in the particular wireless grid scenario that we consider here is that the different agents have different subsets of G (i.e. parts of the file) already and each is trying to obtain the full set by exchanging parts of their subsets of G with one another. Thus, looking at a potential exchange, from the perspective of an agent a_i, for each chunk only two mutually exclusive situations can occur: either the agent does, or does not, have a given chunk. This can be expressed in terms of the set H_{a_i} (the set of chunks agent a_i possesses; $H_{a_i} \subseteq G$) and the corresponding complement set (with regard to G) H'_{a_i} that represents the set of chunks agent a_i wants.

In an exchange, an agent a_1 will try to obtain the set of the missing chunks H'_1 and in turn can potentially provide the set H_1. Let H_2 being the chunks agent a_2 possesses and let agent a_1 and a_2 enter an exchange process ($H_1 \cup H_2 \subseteq G$). In order to reflect the local connectivity properties, we write $\overline{A}_{a_1}$ ($\subseteq A$) to denote those $a_j \in A, j \neq i$ that are within communication range of a_i. The local radius of each agent is determined by the transaction protocol dependent signal radius of its mobile phone.

What is important to the agent now are the utility of the different action alternatives. Thus, an agents needs to consider the utility of using the short-link cooperation (including the costs for searching short-link cooperation partners in the first place) compared to the cellular link as well as the utility of reciprocating in contrast to the one when cheating on the other agents.

The search costs are the costs that accumulate for the agents for searching the missing chunks. The basic idea is that costs of sending out a request message (RM) for cooperation using WLAN transmission are fixed and independent of the number of chunks requested. However, the number of messages an agent has to sent before it finds an agent that is willing to cooperate with him and can sent him at least one his missing chunks depends on the success probability $p = f(\left|\overline{A}\right|, H'); p \in [0, 1]$ for a single message. "Success" thereby means finding a cooperation partner with at least one missing chunk. As stated above, the probability p is a result of the function of the number of agents in the neighbourhood $\left|\overline{A}\right|$ and of the number of chunks missing H'. To this point we have no specific function on how these two components intersect with regard to p, however the general notion of the influence shall be described. Thus, for the missing chunks, we contend, without evidence at this point, that p has a proportional relation with the missing chunks of the form $H'_{a_i} \propto p$. Our rationale starts from the assumption that the chunks are distributed uniformly over all agents. Thus, if missing many chunks an agent is more likely to find another agent that can offer any of the missing chunks, whereas the probability is lower if it is only missing a small number of specific chunks. Besides the number of missing chunks, p is furthermore dependent on the number of agents in the neighbourhood, i.e. the number of other agents $\left|\overline{A}_{a_i}\right|$ an agent a_i can see locally[1]. The probability p is proportional to $\left|\overline{A}\right|$ as well. The idea behind this idea is that the more other agents are in the vicinity of an agent, the higher the probability of finding an agents that responds positive to the request when searching for the chunks.

To give an example for p, in a football-stadium where many people are in one place and want to download the same file (e.g. a replay of a goal), it will approach 1 as there are many people searching for and offering the same chunks, while it tends to approach 0 when there are fewer people searching for and offering the same chunks. Once an agent has found a transaction partner, they can

[1]For reasons of simplicity it is assumed that the number of agents in the neighbourhood has no volatility, but remains the same throughout the process.

exchange chunks. Thus the maximum number of chunks available for exchange is the intersection of the set an agent can offer to the transaction partner (i.e. all the chunks it has) and that the transaction partner needs; and vice versa, i.e. $H_1 \cap H_2'$ & $H_2 \cap H_1'$.

Looking at our example, in the course of the exchange both agents have the option to cooperate (i.e. deliver what they promised) or defect and not send their chunks. As a consequence of this, two different utility situations can occur. Thus, in the cooperation case, based on opportunity cost considerations, the utility is calculated by taking into account what it would have cost for an agent to download the chunks from the base station using the 3G connection ($E_{3G,rx}$) reduced by the costs of receiving the chunks on a short range WLAN link from another agent ($E_{WLAN,rx}$) minus the costs for sending its own chunks ($E_{WLAN,tx}$). The latter cost can be saved by the agent if it defects. However, assuming that the transaction partner stops the transaction if being cheated and no further chunks can be exchange (tit-for-tat), in this case the agent will have search for a new transaction partner for the remaining missing chunks. This results in search costs that could otherwise have been saved. The specific energy cost $E_{a,b}$ where $a \in WLAN, 3G; b \in tx, rx, idle$ have already been determined by Perrucci et al. [18] for single bits. As a first approximation, using a constant *bpc* (i.e. bit per chunk) these could be mapped to the chunks in the model.

Using the *bpc* mapping and the figures by Perrucci et al. and substituting them with the variables of our utility considerations an agent is able to compute an utility for all the actions available to him and decide on the action to take as a consequence.

3. FORMALIZING THE WIRELESS GRID SCENARIO

Now that we have explained the wireless grid scenario in some detail from the technological perspective, we now shift focus to the normative framework.

We observe three perspectives to the wireless grid scenario:

1. The actions that agents may take, as prescribed by the normative framework,
2. The utility functions that quantify battery costs for a given action, and
3. The agents that populate the normative frameworks and choose which action to take, informed by the utility functions.

In this paper, our focus is on the (normative) actions and the utility functions (see section 2.2.4): we will address their integration through the agents that participate in the normative framework in future work.

3.1 The Normative Framework

The model is preliminary in that it focuses on the essential interactions and the communication costs that arise from those interactions. Although a more elaborate model is desirable from a realistic point of view, more details would also distract and complicate while not adding to the presentation.

The features of the the prototypical scenario are:

- $1 \times$ base-station: B
- $m \times$ agents: $A = \{a_1, \ldots, a_m\}$
- $1 \times$ digital good: G divided into
- $n \times$ chunks: $\{g_1, \ldots, g_n\}$

We further assume that $n|m$, which is to say the number of chunks is a multiple of the number of agents.

3.1.1 Negotiation, obtaining and sharing

We identify three phases to the interactions for handset to basestation and handset to handset:

- Negotiation: assign g_i to a_j s.t. $f : G \rightarrow A$ and

$$f^{-1} : A \rightarrow G^{n|m} \text{ s.t. } f^{-1}(a_i) = \{g_j, f(g_j) = a_i\}$$

- Obtaining: agent a_i receives chunks $f^{-1}(a_i)$ from B
- Sharing: agent a_1 sends chunks $f^{-1}(a_i)$ to and recieves chunks $G \setminus f^{-1}(a_i)$ from other agents.

While these three phases are distinguished, they not need to be sequential. Of course, the negotiation phase has to proceed both the obtaining and sharing phases. Sharing is possible as soon as downloading has commenced; thus the two can be interleaved. In the following paragraphs we discuss each phase in more detail and how each is encoded in InstAL.

Each InstALspecification starts with the identification of the normative framework, the different types of variables it will use (their values can be specified in a domain file) and the fluents and events it will recognise. The full definition can be seen in Figure 3. The meaning of the various element will become clear when we discuss the different phases.

Negotiation Phase: We are not particularly concerned with the technicalities of the negotiation phase — any off-the-shelf protocol could be employed — as long as the post-condition is satisfied: that each chunk is assigned to exactly one agent and that each agent is assigned the same number of chunks — these conditions can readily be relaxed at the cost of a lengthier specification. This state of affairs is reflected in the initial state of the model (see Figure 6, lines 104–105) via the `obtainChunk` fluents indicating which agents are tasked with obtaining which blocks from the basestation. Together with their chunk assignment the agents receive the necessary permission to do so (lines 102–103).

Obtaining Phase: This is where each agent downloads its assigned chunks from the basestation. This process should result in each agent holding $n|m$ distinct chunks. Because the basestation uses several different frequencies (frequency division multiplexing), many agents may download chunks simultaneously. We refer to a frequency division in the model as a channel. Of course, there is a physical limit to the number of frequency divisions and hence the number of simultaneous agent connections. The full specification of this phase can be seen in Figure 4. Each agent can only physically obtain one chunk at a time from the base station, while each channel can only be used to obtain one chunk. This is modelled by the fluent `cbusy`. The first InstAL rule (lines 34–36) indicates that a request to obtain a chunk is granted (`intObtain`) whenever there is an available channel and the agent is not busy obtaining another chunk. When a block is obtained the agent and the channel will become busy for a fixed amount of time — 2 time steps in this case (lines 42–43). From the first instant of the agent interacting with the base station, it is deemed to have obtained the block, so parts can be shared (line 41). As soon as a channel and an agent become engaged, the framework takes away the power from the agent and from the channel to engage in any other interactions (lines 53–54), stops the agent from needing the chunk and cancels the permission to obtain the chunk again later on (lines 55 and 56, respectively).

Each exogenous event generates a transition to mark the passing of time (lines 38–39). The `clock` event indicates that no agent was interacting with the normative framework. The `transition` event reduces the duration of the interaction between the channel and agent (line 46). When the the interaction comes to an end, `transition` restores the power for agents to obtain chunks via the channel and for the agent to obtain more chunks (lines 48–51). The event also terminates any busy fluents that are no longer needed (line 58).

Sharing Phase: In this phase each agent shares its chunks with

```
institution grid;

type Agent;
type Chunk;
type Time;
type Channel;
type ConnectionPoint;

exogenous event clock;
exogenous event obtain(Agent,Chunk,Channel);
exogenous event download(Agent,Agent,Chunk);

create event creategrid;

inst event intObtain(Agent,Chunk,Channel);
inst event intShare(Agent);
inst event intDownload(Agent,Chunk);
inst event transition;

violation event misuse(Agent);

fluent obtainChunk(Agent,Chunk);
fluent hasChunk(Agent,Chunk);
fluent abusy(Agent,Time);
fluent cbusy(ConnectionPoint,Time);

fluent previous(Time,Time);
fluent matchA(Agent,ConnectionPoint);
fluent matchC(Channel,ConnectionPoint);
```

Figure 3: Declaration of types and events in the model

```
obtain(A,X,C) generates intObtain(A,X,C)
  if not cbusy(C1,T1), not cbusy(A1,T2),
     matchA(A,A1), matchC(C,C1);

obtain(A,X,C) generates transition;
clock generates transition;

intObtain(A,X,C) initiates hasChunk(A,X);
intObtain(A,X,C) initiates
  cbusy(A1,2), cbusy(C1,2)
  if matchA(A,A1), matchC(C,C1);

transition initiates cbusy(A,T2)
  if cbusy(A,T1), previous(T1,T2);
transition initiates pow(intObtain(A,X,C))
  if cbusy(A1,1), matchA(A,A1);
transition initiates pow(intObtain(A,X,C))
  if cbusy(C1,1), matchC(C,C1);

intObtain(A,X,C) terminates pow(intObtain(A,X1,C1));
intObtain(A,X,C) terminates pow(intObtain(B,X1,C));
intObtain(A,X,C) terminates obtainChunk(A,X);
intObtain(A,X,C) terminates perm(obtain(A,X,C1));

transition terminates cbusy(A,Time);
```

Figure 4: Generation and consequence relations for obtaining

```
download(A,B,X) generates
  intDownload(A,X), intShare(B)
  if hasChunk(B,X), not abusy(A,T1), not abusy(B,T2);

download(A,B,X) generates transition;
clock generates transition;

viol(intDownload(A,X)) generates misuse(A);

intDownload(A,X) initiates hasChunk(A,X);
intShare(B) initiates perm(intDownload(B,X));
intDownload(A,X) initiates abusy(A,3);
intShare(B) initiates abusy(B,3);


transition initiates abusy(A,T2)
  if abusy(A,T1), previous(T1,T2);
transition initiates pow(intDownload(A,X))
  if abusy(A,1);
transition initiates pow(intShare(B))
  if abusy(B,1);

intDownload(A,X) terminates perm(intDownload(A,X));
intDownload(A,X) terminates pow(intDownload(A,X));
intDownload(A,X) terminates pow(intShare(A));
intShare(B) terminates pow(intDownload(B,X));
intShare(B) terminates pow(intShare(B));

misuse(A) terminates pow(intDownload(A,X)),abusy(A,T);
intDownload(A,X) terminates perm(intDownload(A,Y));

transition terminates abusy(A,Time);
```

Figure 5: Generation and consequence relations for sharing

```
initially
  pow(transition), perm(transition),
  perm(clock),
  pow(intObtain(A,B,C)),perm(intObtain(A,B,C)),
  perm(obtain(alice,x1,C)), perm(obtain(alice,x3,C)),
  perm(obtain(bob,x2,C)), perm(obtain(bob,x4,C)),
  obtainChunk(alice,x1), obtainChunk(alice,x3),
  obtainChunk(bob,x2), obtainChunk(bob,x4);

initially
  pow(transition), perm(transition),
  perm(clock),
  pow(intDownload(Agent,Chunk)), pow(intShare(Agent)),
  perm(download(Agent,Agent1,Chunk)),
  perm(intDownload(Agent,Chunk)), perm(intShare(Agent));
```

Figure 6: Initial state of the model, post negotiation

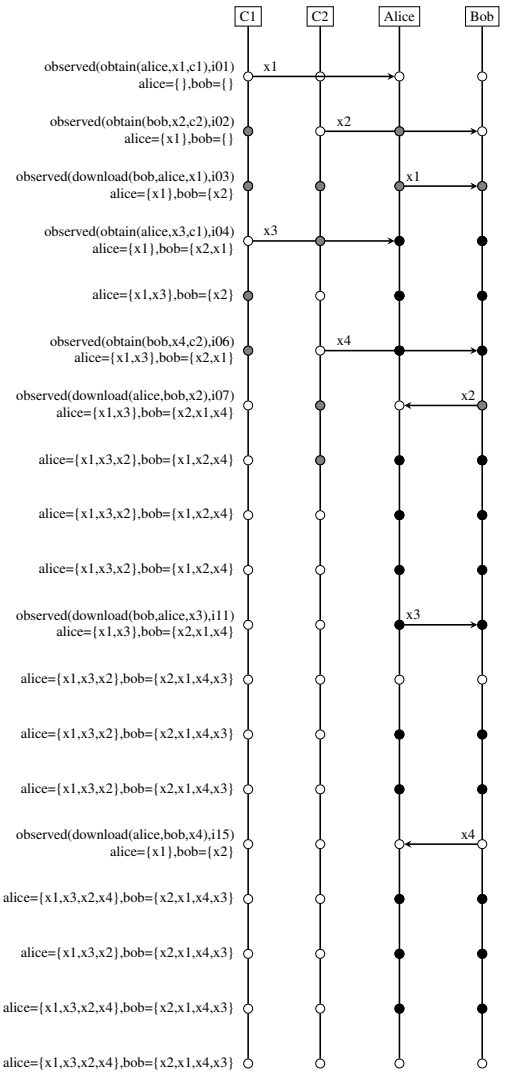


Figure 7: One trace of the interaction between alice, bob and the channels of base-station

another agent, with the goal that at the end of the process, each agent has a complete set of the chunks. The full specification can be found in Figure 5. The idea behind the model is more or less the same as with obtaining blocks, only that we build in a mechanism to encourage agents to share their chunks with others rather than just downloading them. To be able to monitor the different costs of obtaining a chunk from the basestation or from a peer, we introduced the fluent `abusy`. When a chunk is downloaded from a peer, the agent loses permission to download another chunk until it has shared a chunk with another agent (lines 85 and 73 respectively). Continuous downloading without sharing (no permission is granted to download) results in a violation event named `misuse` (line 70). The penalty we chose to implement in our model is that the violation agent loses the power to `intDownload` (Line 91), which means that for all intents and purposes it has been expelled from the peer group. Initially, agents are given the permission and power to download one chunk (Figure 6 lines 112-114).

Figures 3 to 6 give the complete characterisation of our wireless grid scenario. When translated to $AnsProlog$ and combined with the non-framework-dependent program components, we obtain all the possible traces over a specified number of time instances. A successful trace makes sure that at the end all agents have all chunks and are no longer engaged. Figure 7 shows a graphical representation of a successful trace for a scenario with two agents (bob and alice), four chunks (x1, x2, x3 and x4) and a base-station with two channels (c1 and c2). The little circles indicate the time steps. If they are light grey this means the device is `cbusy` while dark grey indicates `abusy`. The arrows indicate which block goes to which agent. The left-hand side labels indicate the exogenous event and the current spread of chunks. The observed event `clock` has not displayed to de-clutter the diagram.

3.1.2 Sanctioning

The model as presented in Figure 5 takes a rather harsh position on sanctioning in that the violating agent is expelled — the power to get chunks from other agents is rescinded. In fact, this is both harsh and counter-productive, because given the initial state shown in Figure 6, the chunk assignment is not 1-resilient — meaning the distribution cannot be achieved in the case of the expeltion of 1 agent, unless in the very special case where the expeltion occurs when the other agent no longer require any chunks for this agent. Full 1-resilient assignment can be achieved with two chunks for each of three agents, in which each chunk is assigned to two agents and of course, n-resilience can be achieved by each agent downloading all the chunks from the base-station. In terms of the effect on the group goal, the ejection scenario is equivalent to one of the agents leaving the ad-hoc network. In either case, for an a-priori solution there is a trade-off to be explored in delivering i-resilience, based on the estimated number agent failures and on the additional cost of replicated basestation downloads. Alternatively, some agents may engage autonomously in additional basestation downloads for the sake of the group goal.

A more practical sanction may be to lock the offending agent out of the sharing process for a number of time steps, but as with the above scenario, this is only effective if it does not impact the group goal.

3.2 Evaluation

Now that we have set out the normative framework and how to quantify communication costs for the particular situation of a 3G structured network and an ethernet ad-hoc network (see section 2.2.4), we can use the model to examine the traces for expected, but also unexpected behaviour and, simply by counting the number of `cbusy` and `abusy` states, get an estimate for battery consumption under different initial conditions.

Each of the models of our framework contains information about the energy consumption of each of agents in the form of the messages they have been passing in the form of the exogenous events `obtain` and `download` and the amount of time they have been spending communicating with the base-station, the number of occurrences of `cbusy`, and communicating with the another agent by means of the number of time `abusy` occurs.

The model is at this stage being used as an off-line tool and generates all possible traces. The likelihood of a high proportion of these trace occurring in practice, depends on the relative intelligence and (bounded) rationality of the agents participating in the normative framework, e.g. continuously trying the download a chunk when you are busy Our model avoids taking the behaviour of agents into account, as we believe this is responsibility of the agent designer. Our traces give all possible ways the system can be used. Traces showing unwanted or irrelevant can easily be filtered out using queries.

By changing the durations for obtaining and sharing chunks and altering the penalties imposed on agents not conforming to the norms, we are able to study a variety of situations and finding the most appropriate enforcement mechanisms.

4. DISCUSSION

In this paper, we have presented a normative framework as a mechanism to help understand and to model the economic challenges that might arise in the context of a wireless grid. We have developed a model for the actions of the agents that participate in such a grid and hence provide us with a basic energy model, that may be used by the agents as part of a utility maximization decision-making process.

This was the first time we had modelled a complete, but simplified, realistic scenario of a normative framework. While InstALis very intuitive and makes the task significantly more approachable it still lacks certain features that would make the modelling process easier. To model that channels and handsets were busy during a given period we had to resort to introducing the fluent `previous`, as InstALdoes not allow arithmetic in its rules (which the underlying $AnsProlog$ does allow). The current version of InstALalso does not allow hierarchies in its type structure or polymorphic propositions. Ideally we would have liked Device to be a superset of both `Channel` and `Agent`, such that we would not have had to resort to the `matchC` and `matchA` fluents, which are a technical artifice to overcome a linguistic weakness. The answer sets representing the traces contain significant numbers of atoms, making debugging difficult. Neither InstALnor InstQLhave built-in mechanisms to filter the output. So for debugging purposes we often referred to the underlying $AnsProlog$ program and its `#show` and `#hide` functionalities, although those are not very flexible. Thus, the exercise has identified a number of practical issues that need to be addressed to make InstALmore usable.

The modelling of the wireless grid scenario gave us also a good insight into our formal model. The model does not allow us to expel an agent completely from a normative framework, as all the observed events are automatically empowered. While this can be partially remedied by removing the empowerment of consequent normative events, as we have done in the sharing phase, it raises interesting issues on how membership of a normative framework should be handled.

The traces of the normative framework give an indication of how much energy each of the handsets will be using when the trace would be executed. It also allows us to test different sanctioning

techniques and compare their efficiency. However, a number of the traces that are produced by our simple model, while valid, stand very little chance of being executed by rational agents. Agents are not going to repeatedly downloading the same block, trying to download/obtain a block when they were busy. While this can be easily added to the model, we believe that this should be encoded by the agents rather the normative framework. From a normative perspective we are only interested in correct, valid traces.

A particularly intriguing line of research, arising from the capacity to compute such traces, is to explore those (economic) mechanisms that might alleviate the effects of free-loading, in a more subtle, and less draconian way, than the simple sanction of expelling, that has been applied here.

Both the wireless grid scenario and the energy model are necessarily simplified and demand expansion. As stated earlier in the paper some functions such as the one defining p, i.e. the probability of finding a cooperation partner that has the right chunks, have to be specified. Further aspects of interest to be included in the model are error rates on the different communication links as well as the aspect that agents are moving within the environment and as a consequence the neighbourhood of an agent is constantly changing.

Furthermore, our current model has very simple penalty mechanisms for violating agents. It does not allow for more elaborate forms of sanctions as was demonstrated in Section 3.1.2 when we blocked agents that obtained unallocated chunks from the base-station. However, with regard to future work we plan to develop several enforcement mechanisms in order to address the reciprocity problem in more detail. The idea thereby is to take the existing model as a reference point and analyse the additional benefits and costs resulting from different normative mechanisms.

Acknowledgements: Tina Balke is partially supported by a grant from the German Academic Exchange Service (DAAD).

5. REFERENCES

[1] A. Artikis, M. Sergot, and J. Pitt. Specifying electronic societies with the Causal Calculator. In F. Giunchiglia, J. Odell, and G. Weiss, editors, *Proceedings of AOSE)*, LNCS 2585. Springer, 2003.

[2] R. Axelrod. The emergence of cooperation among egoists. *The American Political Science Review*, 75(2):306–318, 1981.

[3] C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge Press, 2003.

[4] G. Boella and L. van der Torre. Constitutive Norms in the Design of Normative Multiagent Systems, City College. In *Proceedings of CLIMA-VI)*, 2005.

[5] O. Cliffe. *Specifying and Analysing Institutions in Multi-Agent Systems using Answer Set Programming*. PhD thesis, University of Bath, 2007.

[6] O. Cliffe, M. De Vos, and J. Padget. Answer set programming for representing and reasoning about virtual institutions. In K. Inoue, S. Ken, and F. Toni, editors, *CLIMA VII*, volume 4371 of *LNAI*, pages 60–79, 2006. Springer.

[7] O. Cliffe, M. De Vos, and J. Padget. Specifying and reasoning about multiple institutions. In P. Noriega, J. Vázquez-Salceda, G. Boella, O. Boissier, V. Dignum, N. Fornara, and E. Matson, editors, *Coordination, Organization, Institutions and Norms in Agent Systems II - AAMAS 2006 and ECAI 2006 International Workshops, COIN 2006*, volume 4386 of *LNCS*, pages 67–85. Springer Berlin / Heidelberg, 2007.

[8] Enrico Giunchiglia, Joohyung Lee, Vladimir Lifschitz, Norman McCain, and Hudson Turner. Nonmonotonic causal theories. *Artificial Intelligence, Vol. 153, pp. 49-104*, 2004.

[9] F. H. P. Fitzek and M. D. Katz. Cellular controlled peer to peer communications: Overview and potentials. In F. H. P. Fitzek and M. D. Katz, editors, *Cognitive Wireless Networks*, pages 31–59. Springer, 2007.

[10] M. Gebser, B. Kaufmann, A. Neumann, and T. Schaub. Conflict-Driven Answer Set Solving. In *Proceeding of IJCAI07*, pages 386–392, 2007.

[11] M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3-4):365–386, 1991.

[12] L. Hopton, O. Cliffe, M. De Vos, and J. Padget. Instql: A query language for virtual institutions using answer set programming. In J. Dix, M. Fisher, and P. Novàk, editors, *Proceedings of the 10th International Workshop on Computational Logic in Multi-Agent Systems (ClimaX)*, IfI Technical Report Series, pages 87–104, 2009. Institut für Informatik, Technische Universität Clausthal.

[13] A. J. Jones and M. Sergot. A Formal Characterisation of Institutionalised Power. *ACM Computing Surveys*, 28(4es):121, 1996.

[14] M. D. Katz and F. H. P. Fitzek. Cooperation in 4g networks - cooperation in a heterogenous wireless world. In F. H. P. Fitzek and M. D. Katz, editors, *Cooperation in Wireless Networks: Principles and Applications*, pages 463–496. Springer, 2006.

[15] R. A. Kowalski and F. Sadri. Reconciling the event calculus with the situation calculus. *Journal of Logic Programming*, 31(1–3):39–58, Apr. 1997.

[16] I. Niemelä and P. Simons. Smodels: An implementation of the stable model and well-founded semantics for normal LP. In J. Dix, U. Furbach, and A. Nerode, editors, *Proceedings of LPNMR'97*, volume 1265 of *LNAI*, pages 420–429,1997. Springer.

[17] E. Ostrom. Coping with tragedies of the commons. *Annual Review of Political Science*, 2:493–535, June 1999. Workshop in Political Theory and Policy Analysis; Center for the Study of Institutions, Population, and Environmental Change.

[18] G. P. Perrucci and F. Fitzek. Measurements campaign for energy consumption on mobile phones. Technical report, Aalborg University, 2009.

[19] G. P. Perrucci, F. H. Fitzek, and M. V. Petersen. Energy saving aspects for mobile device exploiting heterogeneous wireless networks. In *Heterogeneous Wireless Access Networks*. Springer US, 2009.

[20] M. Sergot. (C+)++: An Action Language for Representing Norms and Institutions. Technical report, Imperial College, London, 2004.

[21] TNS. Two-day batter life tops wish list for future all-in-one phone device. Technical report, Taylor Nelson Sofres, 004.

[22] J. Vázquez-Salceda, V. Dignum, and F. Dignum. Organizing multiagent systems. *AAMAS*, 11(3):307–360, 2005.

[23] Virginia Dignum. *A Model for Organizational Interaction Based on Agents, Founded in Logic*. PhD thesis, Utrecht University, 2004.

[24] K. Wrona and P. Mähönen. Analytical model of cooperation in ad hoc networks. *Telecommunication Systems*, 27(2–4):347–369, 2004.

A Probabilistic Mechanism for Agent Discovery and Pairing Using Domain-Specific Data

Dimitris Traskas
CACI Ltd.
Andover, UK
dtraskas@caci.co.uk

Julian Padget
Dept. of Computer Science
University of Bath
Claverton Down, Bath, UK
jap@cs.bath.ac.uk

John Tansley
CACI Ltd.
Andover, UK
jtansley@caci.co.uk

ABSTRACT

Agent discovery and pairing is a core process for many multi-agent applications and enables the coordination of agents in order to contribute to the achievement of organisational-level objectives. Previous studies in peer-to-peer and sensor networks have shown the efficiency of probabilistic algorithms in object or resource discovery. In this paper we maintain confidence in such mechanisms and extend the work for the purpose of agent discovery for useful pairs that eventually coordinate to enhance their collective performance. The key difference in our mechanism is the use of domain-specific data that allows the discovery of relevant, useful agents while maintaining reduced communication costs. Agents employ a Bayesian inference model to control an otherwise random search, such that at each step a decision procedure determines whether it is worth searching further. In this way it attempts to capture something akin to the human disposition to give up after trying a certain number of alternatives and take the best offer seen. We benchmark the approach against exhaustive search (to establish an upper bound on costs), random and tabu—all of which it outperforms—and against an independent industrial standard simulator—which it also outperforms. We demonstrate using synthetic data—for the purpose of exploring the resilience of the approaches to extreme workloads—and empirical data, the effectiveness of a system that can identify "good enough" solutions to satisfy holistic organisational service level objectives.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence-Coherence and Coordination, Multiagent Systems; G.3 [**Probability and Statistics**]: Probabilistic Algorithms

General Terms

Algorithms, Performance, Experimentation

Keywords

agent discovery, self-organization, peer-to-peer

1. INTRODUCTION

Agent discovery and pairing in a decentralised multi-agent environment that consists of thousands of agents with different roles and skills and interconnected in various topologies needs to be scalable, efficient, robust and flexible enough to adapt both to changes in requirements and changes in the environment. A number of techniques have been proposed to tackle these challenges; our aim is to demonstrate a simple, effective mechanism that does not require significant computation or centralised control and maintains reduced communication overheads.

Decentralised multi-agent architectures typically consist of self-organising and coordinating agents that do not have any dependencies on a global control system. Information becomes available to agents locally through some form of sensing or messaging and is used to meet the objectives and to satisfy the constraints of the organisation in which the agents participate. Such environments are often highly dynamic and agents can join or leave it rapidly for a variety of reasons such as host failure or migration to a different node in the network. They also need to discover and communicate with other agents in order to exploit services that are being made available. The domain of peer to peer networking has inspired us with useful ideas that have been applied to the development of service discovery algorithms [1] [12]. In general service discovery is achieved using pull or push protocols or a combination of the two over structured or unstructured networks. Push-based protocols require the advertisement of available services thus creating unnecessary overhead when demand is low. On the other hand a pull-based approach has the disadvantage of a search over the network but benefits from the elimination of advertisement messages.

In this study we focus on unstructured networks with a pull-based approach that fully utilises information gathered *during* the search. Observations made by the agent while hopping can be used to update or to infer the probability that a subsquent hop will result in reaching a better-suited agent with which to interact. An essential element of this mechanism is an organisational model that exploits available domain-specific data. We use this data to tag agents in the network during an initial bootstrapping and introduction process and create clusters of agents with similar profiles. The result is a form of overlay network that provides useful information to any agent starting a search operation.

A case study from the business domain, specifically the call centre sector, is used to evaluate the technique and generate quantifiable outcomes. The subject of our case-study is the call allocation process—that is how to select the appropriate handler for a call, being essentially a kind of resource discovery problem. Synthetic and empirical data-sets allow us to make useful comparisons with a commercial simulator and with the metrics generated by a real world call centre, while at the same time providing supporting evidence for general results for low-cost resource discovery.

The remainder of this paper is laid out as follows: Section 2 discusses other work related with agent discovery and pairing and Section 3 provides the motivation for our work. In Section 4, we describe the proposed system architecture in detail and in Section 5, we present the case study that we used to validate our theory. In Section 6, we discuss our findings and finally end this paper with our conclusions and future vision in Section 7.

2. RELATED WORK

Service discovery mechanisms are necessarily closely related to the type of networks used. For example unstructured systems characterised by a loose architecture where agent-peers can join, leave or fail at any point are much more resilient, flexible and scalable but have high communication costs during a random search. Our work focuses on such networks and attempts to improve the random search process by employing a probabilistic model. In contrast structured systems hold information about services fixed on certain nodes and use techniques such as distributed hash tables (DHT) [2] for service discovery but suffer from scalability issues.

Common mechanisms encountered in the service discovery literature are agent matchmaking, gossip-based protocols and probabilistic search. Agent matchmaking [20] can be classified into (i) centralised using market bidding mechanisms [19] [18] [16], (ii) centralised but using a middle agent or broker [10] [5], (iii) and fully decentralised where agents use local information to form clusters and randomly find useful pairs [13]. The main disadvantages of centralised mechanisms when applied to large scale systems are scalability, robustness and flexible dynamic behaviour. A decentralised matchmaker design is typically preferable and can potentially benefit from a probabilistic approach, as we will shortly demonstrate.

Gossip-based mechanisms are based on probabilistic flooding where an agent sends messages to a certain percentage of its neighbours that it believes they are available [14] [4]. Part of the gossiping approach is the random search of agents which we believe can be improved in a specific problem domain by employing a probabilistic model that utilises any data available.

Probabilistic techniques that have been previously investigated within the context of resource discovery in sensor networks [15, 3] or object searching in peer-to-peer networks [17, 6] seem to offer some of the characteristics we seek. The models employed often use network level information and observations from previous searches to inform the discovery process. The key difference with our work is the use of domain-specific data that allows the discovery of relevant, useful agents while maintaining reduced communication costs. Our objective is to allow agents to discover the best pair possible in order to coordinate towards optimal collective performance. Agent discovery in an application context requires specific problem knowledge which is why we have to ground our experiments in a particular domain in order to demonstrate its effect.

3. BACKGROUND AND MOTIVATION

The past few decades have seen rising consumer demands for more competitive products and services and have increased the need for a more flexible and adaptive business operating model. For many businesses, existing software systems fail to adapt and evolve rapidly while the complexity of their management is becoming a limiting factor in further growth. In an effort to solve these problems business managers choose the reductionist approach; essentially reducing the number of products and services offered or even the size of their customer base under the mantra of focusing on "core business". The desired outcome of this approach is to simplify the processes and systems in place—however reduced performance and profits can also result.

One of the aims of this work is to use the multi-agent paradigm to tackle these challenges and develop software solutions that can operate in highly dynamic environments. Our long term goal is the development of business agent societies which can be characterised by the following: (i) open, scalable and heterogeneous, (ii) fully decentralised and autonomous, (iii) distributed across different geographical locations and areas of the business, (iv) constrained by business rules and policies.

A typical model for these systems consists of service providers and service consumers. In an agent society with the aforementioned characteristics the consumer will attempt to locate the provider without having any prior knowledge of location or global system state. Consumers are self-interested and need to discover and pair with a suitable provider so that they can get optimum returns from their specific utility function. The society can be seen as a large search space that contains a great number of provider-consumer pair combinations but with only a few satisfying business constraints and delivering optimum performance. From this perspective the problem is that of optimisation: essentially, how to find the best pair possible with the minimum number of messages. We believe that "best" is not always necessary and for many application domains a "good enough" solution can be adequate as long as the response is timely and overheads are low.

4. SYSTEM ARCHITECTURE

This section describes the system architecture for the model used in this study. In the absence of a central control regime we have an heterogeneous society of agents that are fully connected with each other. Agents can join and leave the society over time without significantly affecting processes in place or other interactions. In general the model consists of service consumers and service providers however this does not necessarily imply that a single agent cannot be both or either at different times. Depending on the problem domain agents could potentially change role in response to a changing environment. This dual nature of agents is not part of our investigation but it is something we intend to explore in future work. In a provider-consumer society, we are always going to have consumers searching the network for providers which will result in a successful pairing that enables the completion of an agent's individual goals. Nevertheless this one-sided view might limit the effectiveness of the discovery algorithm and there is no reason that a provider to consumer search would not also work.

One of the fundamental elements of our design is the use of agent-local memory to store useful information such as member addresses, cluster formations, skill distribution or agent availability. Each agent has two types of memory, long term and short term that are used for different purposes. Long term memory is simply a kind of cache with all the unique identifiers of agents keyed by type or skill and used for searching relevant areas of the network. New or disconnected members are respectively added or removed from this list over time. Short term memory is used during the discovery process to store the recently visited members and cleared when the search terminates. Elements in the short-term memory are ranked by fitness where fitness can be defined as the result of a utility function that relates a given pair of agents. An internal mechanism ensures that the two types of memory are synchronised and updated at the same time when a message arrives with agent membership information.

The model uses a form of a semantic overlay network of skill-based clusters that are updated every time a new agent joins. We use a tagging mechanism like that proposed by Holland [9] where agents are tagged based on their type and grouped in clusters. The clusters allow agents to limit their search and target specific areas of the network without wasting time or messages. There is no limitation to the number of clusters an agent can belong to, since cluster membership depends solely on the number of services that an agent can provide.

There are certain key processes that do not change from one

problem domain to another and which we explain in detail below. The messages used in the model can be seen next to each process. Messages have four fields: the sender, the recipient, the subject, and any useful data.

A. Introduction Process [PING]

A new provider or consumer joining the society follows an introduction procedure. On initialisation it will receive a list of all members of the society and start sending *Ping* messages in random order to all other agents; in essence announcing its existence. The messages sent contain information that describe the sender and is used by the recipients to classify the new agent. Existing members will examine the information received – specific to a problem domain – and decide if the new agent would be useful to store in local memory or not. Information such as the type of an agent or availability are some examples. If the type of a new member indicates a useful future pairing then its unique identifier and all related information get stored in local memory, otherwise it is discarded. For example a service provider who does not have the required skills to provide a service at this current time to a customer agent would not be added in local memory. This process is similar to an agent naming service however we wish to maintain the lack of any dependencies on centralised control systems and structures. It is required to work in a very dynamic environment where agents join and leave all the time. Incorrect information about the clusters does not stop the system operating however it does have an impact on performance because of the limited *visibility* of the agents.

B. Disconnection Process [DISCONNECT]

A consumer agent disconnects from the network when it has accomplished its tasks while a provider disconnects when it is time for a break or the end of the working shift. In the case of host migration the agent will have to inform the other agents with an alternative message and thus separate the two distinct events. The disconnection process is simple and requires a minimal *Disconnect* message that is sent to cluster members only. This message does not obstruct or halt any current activity but only updates local memories. If a host fails then it is obvious that one or more agents will suddenly disconnect and, as a result, the necessary updates will not be performed. This will only result in temporary delays in the discovery process but under any circumstances stop currently active processes.

C. Discovery Process [QUERY / QUERYHIT / QUERYREJECT]

The specific requirements of a particular problem domain will ultimately dictate how agents behave. When an agent – either consumer or provider – decides to search the skill clusters for candidate pairs it first checks long term memory. As previously mentioned, skill clusters are stored in long term memory and provide the addresses of any connected agent. Specific tasks require specific skills and thus a search on a single cluster. From a list of potential addresses in the cluster one is picked randomly and a *Query* message sent to it. The message might contain data and that again depends on the specific problem. For every *Query* message a hop counter is incremented to track the number of messages sent. The hop counter is compared to the number of agents that are of interest for a future pairing and the process ultimately stops once that number is reached. There can only be two outcomes from this message:

- *QueryHit*: The *Query* has been successful and the recipient replies to the sender with (domain-specific) information such as waiting time or level of importance. At that point any information sent along with the agent address are stored in short term memory and supplied to the probabilistic decision engine. The result of the decision engine is the probability that a subsequent *Query* will return a better result. If that probability is very low then the agent will stop the search and select the highest ranking agent from short term memory for an *Offer*. If the probability is high then the search will continue.
- *QueryReject*: If the recipient has already formed a pair with someone else or is ready to disconnect from the network then it will send a *QueryReject* reply. On receipt of the reply the originator of this sequence of events will remove the address from short term memory and check if there is anyone else in the list in order to continue the search.

D. Pairing Process [OFFER / ACCEPT / REJECT]

When the decision engine returns with a small probability that the next hop will produce a better pair the search process is stopped and an *Offer* message is sent to the highest ranking agent. The recipient will either *Accept* that offer or *Reject* it. We have experimented briefly with a model where offers were not immediately accepted but rather the receiving agent waited a bit more to see if something better arrives. In this case overall system performance dropped because waiting time was increased and good offers were used by other agents.

4.1 Probabilistic Discovery and Pairing

In the decision engine described previously, each agent continuously updates its belief about the state of the world around it, using the messages it receives from other agents. This set of beliefs is then used by each agent when searching for a good pair, in order to estimate whether it is worth continuing the search, or whether further searching is unlikely to produce a better match. This model enables the agent to make an informed choice, by estimating whether the next hop is likely to provide a more suitable pair than the best one it has found so far. The updating of local beliefs is based on a Bayesian framework. Each agent starts off with a prior belief about the state of the system around it which is currently the same for all agents of a given type but as all calculations are purely local, these could easily be made to change from agent to agent. This prior belief is then updated by the agent based on the messages it receives, to give a posterior distribution on the state of the relevant part of the system. As the agent receives more and more information, its probabilistic model reflects the world around it more and more accurately. This updating of prior (before the data) to posterior (after the data) beliefs occurs in a very specific way, according to Bayes' rule:

$$P(H \mid D) = \frac{P(D \mid H)P(H)}{P(D)}$$

where D is the observed data and H is the particular hypothesis being tested by the agent. The prior probability P(H) represents the agent's prior belief in the hypothesis before seeing any data. The model evidence P(D |H) reflects how well a particular hypothesis predicts the data, and the evidence P(D) provides normalisation over all hypotheses. We can see that if we have two hypotheses with equal priors, the hypothesis that predicts the data more accurately will end up with the highest posterior probability, as we would expect.

Suppose that an agent society has one provider and a number of consumers. We also assume that the provider is ready to serve a consumer, and wants to serve the most suitable of the available

consumers. Rather than perform an exhaustive search by sending and receiving messages to all consumers, this provider can use its model of the distribution of consumer types to poll a subset of consumers. This provider's initial estimate of the distribution of consumer types is necessarily imprecise, but as more consumers are polled, the accuracy of this estimate improves. This estimate combined with the knowledge of the best consumer seen so far, enables the provider to stop any further hops once it believes, that the probability of the next hop returning a better match than the best one seen so far is sufficiently low. Two main assumptions are required in order to specify this probabilistic inference process fully. The first is the agent's prior belief in the state of the system. This should be as broad a prior as possible if the state of the system can be variable, but may be as informative as desired if the system is more static, and a reasonable guess can be made as to its state without polling many consumers. In this implementation each agent resets its belief each time it goes through a new process of pair discovery. However in a more static environment agents could quite easily retain some information from any previous search and incorporate this into the prior for the next search. The second parameter required is the probability threshold at which each agent stops performing any further hops, and stays with the best pairing found up to that point. This threshold is defined initially at a low level and adjusted accordingly after experimentation with the specific problem. After each hop the agent evaluates the probability that the next hop will return a better solution than the best it has encountered so far. If this probability falls below the threshold searching is terminated and the best solution encountered is selected.

In order to use the probabilistic engine across different domains it is necessary to mine useful information from the data during an initial modelling exercise. The mechanism requires domain knowledge not necessarily expert but sufficient to make useful observations from the data which can be used by the bayesian inference model. For example across many domains one could use the type of service required and estimate a distribution of consumers waiting in a queue. The question that arises is if this modelling process could become part of the intelligence of an agent however for this study we rely on the modeller.

4.2 Random and Tabu Mechanisms

Given the probabilistic orientation of our research, we have also investigated two other approaches that have produced interesting outcomes. We have experimented with a completely random discovery process where the agent randomly hops from one node to another and decides to stop at random also. Every agent has a given probability of stopping the search after each hop—this is a hypergeometric distribution of the number of hops. When the search is complete short term memory is used to rank the visited agents and make an offer in exactly the same way as with the probabilistic approach. Although there is complete lack of control in the system, experiments conducted with our case study demonstrate interesting results that we briefly discuss in a later section. The other obvious technique is tabu pairing where the agent decides to pair with the first potential candidate. A single *Offer* message is sent and the candidate added in short term memory. If a rejection is received the next candidate will be contacted until the list is exhausted. Our aim with these mechanisms is to explore fully the potential of our design and produce comparisons with the Bayesian model that demonstrate its effectiveness.

5. CALL CENTRE CASE STUDY

Call centres are fundamental to the operation of numerous large organisations, emergency and government agencies and all types of customer service providers [7]. Call centre management and critical aspects of it such as the call routing process which we focus on in this paper, are becoming increasingly complicated for the same reasons mentioned in our motivation section. The term 'call-routing' is probably most commonly associated with telecommunications networks, where the task is to avoid hot-spot creation, maximise throughput and minimise latency. However call-routing in call centres is a somewhat different problem that is closer to distributed resource discovery and allocation. Conventional implementations of call-routing in call centres are tightly controlled, centralised systems of asynchronous components, where all the decision-making is embedded in a single element—the call router—that communicates with the call-handlers (typically known as agents in the call-centre literature: here we use the term "handler" to alleviate confusion with the term (software) agent).

It is important to note that our aim with this first study is not necessarily to improve the performance of any of the current call routing algorithms but to demonstrate that a multi-agent system can be adopted effectively in this sector. One of the potential benefits of a decentralised architecture based on agents which can deliver similar performance with current solutions is reduced telephony and maintenance costs. Instead of using a central server that operates in a similar manner with a centralised matchmaker, agents spawned by the customers on mobile or fixed lines would locate the call handlers directly by employing a probabilistic discovery protocol over a peer to peer network. The same base technology could offer improved simulation capabilities and the evaluation of alternative operation models or allow the interconnection of other areas of the business with the call centre.

5.1 A Multi-Agent Approach to Call Routing

During the call routing process a call arrives from the telecommunications network and is ready for allocation by the 'Automatic Call Distributor' (ACD). The ACD processes a set of business rules which determine which call has the highest priority such as the call type or skill, as it is usually referred to in this sector. Another parameter used is handler availability, the resources waiting the longest without work will be at the top of the allocation list excluding those on breaks. A new call is inserted into a queue of calls that are ordered by waiting time and skill priority and when a new resource becomes available the ACD gets notified and begins the allocation process. Customers who wait too long terminate their link with the ACD and their call gets registered as abandoned and removed from the queue. We address the call routing process from a MAS perspective and adopt agents in representing the key elements of it. We have developed a conceptual model of the call centre for our simulations which consists of: (i) the *Call Centre*, (ii) the *Handlers*, (iii) the *Calls* which contain customer information, skill required and time of arrival, (iv) the *Skills* which are the different types of call a handler can process, (v) and finally the *Skill Groups* which are groups of skills call handlers can have; an insurance skill group may consist of motor, home and pet insurance.

For the call allocation process we further developed a multi-agent model based on the concepts described above and where agent *Handlers* use a probabilistic mechanism to discover *Call* agents that have been waiting the longest and have the highest skill priority. A significant difference between our model and the standard industry algorithm, is the allocation of calls to *any* handler irrespective of whether they are the longest waiting. For each customer to be serviced, we spawn a *Call* agent. Any useful information such as time of arrival in the system, type of service required or customer details are contained within the *Call*—this is (some of) the domain-specific data. For simulation purposes we model the aban-

donment behaviour of customers using a patience-time model. We have implemented this model based on the exponential distribution [11], that is recognised as appropriate for this kind of simulation delay. For each call that joins the network we generate randomised abandonment times using the formula:

$$\text{Time in seconds} = (\log(1 - r) - \sigma)$$

where r is a random number between 0 and 1 and σ is the average patience time. The same model is considered suitable to generate the handling times of agent handlers. With *Calls* being defined as agents in the system, queueing is resolved and managed by the *Call* itself. The core processes of the *Handler* and *Call* are explained in Figure 1.

We have implemented an agent-based discrete event simulator to create the agent network required for our call centre simulations. The simulator mimics the characteristics of the Cougaar [8] agent platform in the use of plug-in components—rather than behaviours—to program agent actions and the use of a blackboard-like publish and subscribe mechanism—rather than messaging. The blackboard encourages the use of events for agent interaction and is the primary mechanism for state management and message exchange. In a similar manner our simulator uses events such as the arrival of a new call, or a *QueryHit* to activate agents and a blackboard mechanism to send messages[1]. For simulation purposes, the model required a *Time* update message to wake up every agent and trigger new activity.

The simulation begins by loading a call centre model following that defined earlier and initialising the *Handlers* with shift data. Calls are randomly generated using a predefined call density per interval in a day and added in a list ordered by time of arrival. On every tick a call is removed from the list and a new agent injected into the system with the relevant call information. Time updates are subsequently sent with the time of arrival of the new call and the simulation runs.

5.2 Bayesian Model for Call Discovery

In this implementation all *Handler* agents actively maintain a simple Bayesian model of their immediate environment and use it for optimising the call discovery process. More specifically, this search is halted when the estimated probability that the next hop would return a better call than the best one seen so far falls below a threshold of 30%, a percentage that we consistently used throughout our experiments after initial testing. A *Handler* agent needs to track both the distribution of skills and the distribution of waiting times for each skill to get a reasonable picture of the world around it.

5.2.1 Estimate of Call Distribution By Skill

For *NS* skills, the distribution of waiting calls by skill can be described by a categorical distribution, so that for skills S_1 to S_{NS} the probability of a call in the queue being of a particular call type is P_i, where $\sum_{i=1}^{NS} P_i = 1$. An agent needs to estimate this distribution using the observed number of calls n_i for all skills. Assuming a conjugate Dirichlet distribution on the agent's prior belief and using a non-informative Jeffrey's prior, we can write the mean prior estimate P(S_i) for P_i as:

$$P(S_i) = \frac{1}{NS}$$

And the mean posterior estimate where N is the total number of

Handler Agent

1. *Initialise:* The agent is initialised with shift information and a list of all members of the network.
2. *Ping:* The agent sends *Ping* messages to everyone as part of the introduction process. As service providers and consumers can be effectively both we allow all agents to follow this procedure although the information is not used at this stage.
3. *Hop:* The agent checks current availability using local time and shift information. If there is no break, then it checks the list of available *Calls* in skill clusters that can be serviced. From the list of *Calls* and skill clusters one is randomly selected and a *Query* message sent and the short term memory is initialised.
4. *Query Hit/Query Reject:* If the recipient *Call* is available and has not abandoned or received an offer already it replies with either a *QueryHit* message that contains the waiting time, or a *QueryReject*. On a *QueryHit* reply the *Handler* adds the *Call* in short term memory and uses the Bayesian inference engine to decide if a further hop is required. If the probability is low then the discovery process ends and an *Offer* message sent to the *Call* that has been waiting the longest and has the highest skill priority. Otherwise the search continues until the list of *Calls* is depleted or the probability of a better pair gets lower than the system defined threshold.
5. *On Accept:* If the recipient *Call* is still active and ready for handling it replies with an *Accept* message. The reply triggers the handling process which requires the *Handler* to make a direct connection with the customer and start providing a service.
6. *Call Disconnect:* If a *Call* disconnects a *CallDisconnect* message is received and the agents unique address removed from long term memory.

Call Agent

1. *Initialise:* The agent is initialised with customer details, skill priority and time of arrival.
2. *Ping:* Once again *Ping* messages are sent to every member of the society containing skill(s) (required) and time of arrival.
3. *Query:* When a *Query* is received the local time and offers from other agents are tested and on success a reply with a *QueryHit* posted, otherwise a *QueryReject*.
4. *Offer:* If an *Offer* is made and the agent is still active an *OfferAccept* is sent as a reply otherwise an *OfferReject*.
5. *Abandon:* When customer patience has reached the limit the *Call* abandons and the disconnection process begins by sending *CallDisconnect* messages to the society.
6. *Call Disconnect:* Same process as with the *Handler* above.

Figure 1: Process descriptions for Handler and Call agents

[1]This apparently curious approach demands a brief explanation: we developed our own simulator primarily for speed of prototyping. Earlier simulations had been constructed directly in Cougaar, but turn-around time and some concerns about the long-term viability of Cougaar led us to the temporary pragmatic solution described, where the style of agent programming and communication follows the event-driven model that characterises Cougaar, with the aim of returning to the Cougaar framework in due course.

calls observed as:

$$P(S_i) = \frac{n_i + \frac{1}{2}}{N + \frac{NS}{2}}$$

This flat prior across the categories could be potentially tailored more accurately if the distribution of skills was known up-front to be non-uniformly distributed.

5.2.2 Estimate of Call Distribution By Waiting Time

An estimate for the mean call waiting time is calculated by each *Handler* agent, for each skill of interest. We follow the common assumption that call waiting times are modelled by an exponential distribution which holds for certain theoretical call centre models. To get an estimate of the mean call time we use a quick short-cut rather than the more rigorous full conjugate distribution formulation (usually using an Inverse Gamma conjugate prior).

Our estimate for the mean posterior waiting time is:

$$t_{EST} = \frac{t_0 + \sum_{j=1}^{ND} t_i}{ND + 1}$$

Where ND is the total number of observed data points. This formulation implies a prior waiting time estimate t_0. As we are assuming an exponential distribution of waiting times, the probability that the waiting time t of the next *Call* agent has been waiting longer than the waiting time t_B of the longest *Call* visited so far is given by:

$$\begin{aligned} P(t > t_B) &= \int_{t_B}^{\infty} P(t|t_{EST})\, dt = \int_{t_B}^{\infty} \frac{1}{t_{EST}}\, e^{-\frac{t}{t_{EST}}}\, dt \\ &= e^{-\frac{t_B}{t_{EST}}} \end{aligned}$$

5.2.3 Estimating The Probability of a Better Call

We can now calculate the probability P_{next} that the next *Call* queried will be better than the best *Call* seen so far, by combining the probabilities of both the different skills and waiting times:

$$\begin{aligned} P_{next} &= P(SP_{next} > SP_B) \\ &\quad + P(SP_{next} = SP_B) \\ &\quad * P(t_{next} = t_B^{SP_B}) \end{aligned}$$

This is the probability that either the skill priority SP_{next} of the next call is of a higher priority than the highest seen so far SP_B or that the skill priority will be equal to the best seen so far SP_B and the call waiting time t_{next} will be higher than the best seen so far for that skill type. This estimate can then be directly compared to a threshold value to make a decision about the next action.

5.3 Experiments

For validation purposes we conducted a number of simulation experiments with different scales and using synthetic, and empirical data provided by our sponsor [2]. We compared the performance of the two solutions with that of an industry-standard call centre simulator from our sponsor called Call Centre Workshop (CCW) and actual data provided by one of our clients. CCW is a commercial product used by a significant number of clients from the software, retail, banking, insurance and mobile phone sectors. It is a discrete event simulator which allows users to easily set-up call centre models and alternative routing algorithms. For our experiments we used a set of performance metrics to validate and compare the results that are common in this industry. For space economy reasons we are only presenting Service Level (SL%) which is the percentage of calls answered within a business-specific time frame. This time frame is called Telephone Service Factor (TSF) and is usually in the range of 20-30 seconds. SL% can be used to measure intra-day, daily and weekly performance and is normally defined as:

$$\text{SL\%} = \frac{\text{Calls Answered before TSF}}{\text{Calls Answered} + \text{Calls Abandoned}} \times 100$$

We also track the total number of messages required for agent discovery—that is, a handler discovering a call. The comparison plots presented in a later section use SL% and message count per interval to measure efficiency in terms of business performance and communication costs. We compare the Bayesian inference model against the standard approach with results from CCW and other techniques such as random, tabu and exhaustive. Our aim is to find the upper and lower limits of the call allocation space and understand where the probabilistic mechanism stands. The exhaustive search goes through every *Call* agent available and as expected this requires the most messages. The tabu search should be at the lower limits of the search using only one message per call and our expectation with the random and probabilistic techniques is to be somewhere in the middle.

Table 1: Synthetic Model (5 skills, 50 handlers, 7720 calls)

Skills	TSF (secs)	Priority	AHT (secs)	APT (secs)
LOANS	20	1	240	180
MORTGAGES	20	2	240	180
PET INSURANCE	20	2	180	180
MOTOR INSURANCE	20	3	240	180
CREDIT CARD	20	1	240	180

For the first experiment we created a synthetic dataset with the aim of testing the agent prototype and allowing all different scenarios to be handled by the agents. The attributes of the model used are summarised in table 1 where (AHT) is Average Handle Time and (APT) Average Patience Time. We run that model for all the different discovery mechanisms and for 10 iterations. During the second phase of the experimentation process we used real customer data. The call centre selected for our experiments is part of one of the UK's leading mobile phone retailers. The client provided us with actual data from one of their main call centres which we use to simulate one day with 560 Handlers, 43,365 incoming calls and 13 different skills. Calls received are considered to be within SL% if they have been answered within 20 seconds from the time of arrival. Skill handling times – time it take to handle a call – were varied through the day, while customer patience time is drawn from an exponential distribution with a mean value of 180 seconds. For the purpose of this work we set-up a very simple call routing model with no skill priorities and a direct mapping of skills to skill groups. We then loaded the data mentioned above into the agent model and executed the simulation only for 5 runs due to the time it takes to execute the simulation for each of the discovery mechanisms. We compared our results with the actual SL% values which the client's resource planning team calculated after collecting all the raw data stored for that day in the ACD database.

6. RESULTS AND DISCUSSION

Below we present comparisons with the metrics produced from our experiments. The graphs in figure 2 show changes in SL%

[2]CACI Ltd. `http://www.caci.co.uk`

throughout the day as well as query counts for every *Call* agent discovered. As anticipated the synthetic model is over-stretched and agents can hardly cope with the workload. The overall variance in SL% between CCW and the different discovery mechanisms is between 1%-4%. More specifically the probabilistic approach has an average of 46% throughout the day when CCW has 49.5%, random has 45% and finally tabu and exhaustive searches are almost identical at 49%. The significant difference however is in the number of messages. The probabilistic approach requires an average of 300 queries through the day to deliver 46% when exhaustive is at the upper end with 1800 queries on average, random 640 and tabu 310. We believe these results demonstrate that the probabilistic search is efficient while allowing agents to find suitable pairs and contribute towards good global performance.

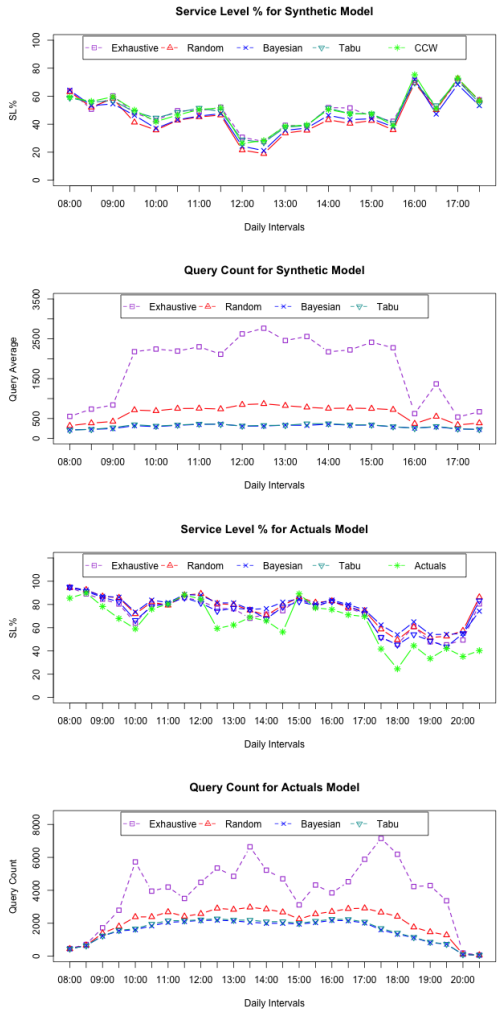


Figure 2: Service Level and Message Count comparisons for synthetic and empirical data experiments.

The real call centre experiments with an actual average SL% through the day of 64% are of greater scale and complexity and show a lot more promise for the probabilistic mechanism. In this instance the agent prototype that uses the probabilistic search performs at 77% compared to 73% tabu, 76% random and 72% exhaustive. For this study these results are far more important because we are using actual figures from a real world call centre rather than artificially devising the skill sets, the handling times or the number of incoming calls as we did in the earlier synthetic test. There is an important variation of 8%-13% here in performance between actuals and the agent models which can be explained by the different algorithm used for the call allocation process however the two systems follow the same trend. In one case the actual system is assigning calls to the longest available handler using a Router and in the other agent *Handlers* make an informed decision as to which *Call* to handle. The total number of query messages required on average per run are: 40,000 for probabilistic, 41,500 for tabu, 54,000 for random and 102,000 for exhaustive.

We notice that the difference in the number of query messages between the bayesian approach and tabu is greater when using the actual data model rather than the earlier synthetic test. Tabu although effective does not scale as well as the probabilistic mechanism because of discovery conflicts. During tabu all *Handlers* attempt to handle the same call by choosing an identical candidate from their list and ignore other potentially good pairs. When one of them succeeds the rest start again but once more target the same *Call* agent. This process repeats until all *Calls* are handled but is wasteful in terms of queries required. The random discovery is not as effective as the Bayesian model either; this result was as we anticipated. The *Handlers* do not use any information gathered while hopping from one *Call* to another and just randomly stop at a specific point in time. Nevertheless the technique performs much better than we thought it would and proves that random policies can be equally effective as a much more controlled design.

The results presented in this section confirm that the multi-agent prototypes developed are effective and inspire us to further experimentation. Below we provide a summary of our findings:

- We have shown that a decentralised multi-agent system that uses a probabilistic discovery and pairing mechanism can be applied effectively and shows comparable performance with standard centralised designs. Sometimes it is possible to outperform exhaustive search by employing a Bayesian inference model that can inform decisions. When compared to other techniques such as an exhaustive search, random walk or tabu allocation it shows similar or better performance but with significantly less communication overheads.

- Our solution is simple in principle and efficiently handles task prioritisation and queueing without the use of an agent-mediator or super nodes. Instead it relies on active Consumer / Work / Task agents that can self-manage. The system can host as many types of Consumers as desired without affecting the basic model and discovery process.

- We have demonstrated that it is not necessary to have an architecture with any centralised control structures in place that will dictate how the system operates. Our model has a degree of randomness which allows it to search more efficiently for good pairs of agents that will deliver reasonable performance and follows much more relaxed design principles.

- Finally we have shown that a multi-agent architecture can be used effectively to manage a business process with a real

world example from the call centre industry. Agent *Handlers* are capable of discovering customer *Calls* that join the network while trying to adhere to business rules governing skill priority and customer waiting time.

7. CONCLUSIONS

We have described a probabilistic discovery and pairing mechanism in which service providers or consumers use available domain-specific data to calculate the probability that the next agent in the search will result in a better match. With lack of global control and by using local information we have shown how agents can inform their decision making by updating a Bayesian inference model. In order to demonstrate the efficiency of the probabilistic mechanism we have applied our ideas to the business domain in the context of call centres. We have developed a decentralised multi-agent prototype to address the call allocation problem and run simulation experiments with synthetic and empirical data. Results show that our approach is at least as effective, when measured by QoS, as current centralised approaches. When compared to multicasting or random walk techniques it demonstrates reduced communication overheads. Agents learn from their observations while hopping and with relatively little messaging manage to discover suitable pairs. Potentially, gossip-based or distributed matchmaking mechanisms that require an initial random search could also benefit from the technique we have outlined.

Future work involves further exploration of the model and its usage in other application domains. We wish to experiment with problems where there is a high number of disconnected and failed agents and measure their impact on the discovery mechanism. Another aspect of this research that interests us is geographical location of agents and how a probabilistic search can use such information to improve pairing. We also plan to improve the Bayesian model using more rigorous conjugate distributions to represent the agent beliefs which would provide a better starting point for the discovery process. Finally we would like to investigate alternative topologies such as scale-free or small-worlds networks and experimentally evaluate designs where *Providers* and *Consumers* both search for each other simultaneously.

8. ACKNOWLEDGEMENTS

We wish to thank CACI Ltd for their support and we would like to note that our findings and conclusions do not necessarily reflect those of the sponsor.

9. REFERENCES

[1] S. Androutsellis-Theotokis and D. Spinellis. A survey of peer-to-peer content distribution technologies. In *ACM Computing Surveys (CSUR)*, volume 36, pages 335–371. ACM Press, 2004. ISSN:0360-0300.

[2] H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Looking up data in p2p systems. *Commun. ACM*, 46(2):43–48, 2003.

[3] R. Biswas, S. Thrun, and L. J. Guibas. A probabilistic approach to inference with limited information in sensor networks. In *IPSN '04: Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 269–276, New York, NY, USA, 2004. ACM.

[4] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *Information Theory, IEEE Transactions on*, 52(6):2508–2530, 2006.

[5] K. Czajkowski, I. T. Foster, N. T. Karonis, C. Kesselman, S. Martin, W. Smith, and S. Tuecke. A resource management architecture for metacomputing systems. In *IPPS/SPDP '98: Proceedings of the Workshop on Job Scheduling Strategies for Parallel Processing*, pages 62–82, London, UK, 1998. Springer-Verlag.

[6] R. A. Ferreira, M. K. Ramanathan, A. Awan, A. Grama, and S. Jagannathan. Search with probabilistic guarantees in unstructured peer-to-peer networks. In *P2P '05: Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing*, pages 165–172, Washington, DC, USA, 2005. IEEE Computer Society.

[7] N. Gans, G. Koole, and A. Mandelbaum. Telephone call centers: a tutorial and literature review. In *MANUFACTURING AND SERVICE OPERATIONS MANAGEMENT*, pages 79–141. MSOM, 2003.

[8] A. Helsinger, M. Thome, and T. Wright. Cougaar: a scalable, distributed multi-agent architecture. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 2, pages 1910–1917. IEEE, 2004. ISSN: 1062-922X. ISBN: 0-7803-8566-7. INSPEC Accession Number: 8393468. Digital Object Identifier: 10.1109/ICSMC.2004.1399959.

[9] J. H. Holland. *Hidden Order: How Adaptation Builds Complexity*. The Perseus Books Group, 1995. ISBN-13: 9780201407938.

[10] D. Kuokka and L. Harada. Matchmaking for information agents. In *IJCAI'95: Proceedings of the 14th international joint conference on Artificial intelligence*, pages 672–678, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[11] D. J. Lilja. *Measuring computer performance: a practitioner's guide*. Cambridge University Press, New York, NY, USA, 2000.

[12] E. Meshkova, J. Riihijärvi, M. Petrova, and P. Mähönen. A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. *Computer Networks*, 52(11):2097–2128, 2008.

[13] E. Ogston and S. Vassiliadis. Local distributed agent matchmaking. In *CooplS '01: Proceedings of the 9th International Conference on Cooperative Information Systems*, pages 67–79, London, UK, 2001. Springer-Verlag.

[14] E. Simonton, B. K. Choi, and S. Seidel. Using gossip for dynamic resource discovery. In *ICPP '06: Proceedings of the 2006 International Conference on Parallel Processing*, pages 319–328, Washington, DC, USA, 2006. IEEE Computer Society.

[15] F. Stann and J. Heidemann. Bard: Bayesian-assisted resource discovery in sensor networks. Technical report, USC/Information Sciences Institute, July 2004.

[16] K. Sycara, J. Lu, M. Klusch, and S. Widoff. Matchmaking among heterogeneous agents on the internet. In *in AAAI Spring Symposium on Intelligent Agents in Cyberspace*, 1999.

[17] D. Tsoumakos and N. Roussopoulos. Adaptive probabilistic search for peer-to-peer networks. In *Peer-to-Peer Computing, 2003. (P2P 2003). Proceedings. Third International Conference on*, pages 102–109, 2003.

[18] D. Veit, C. Weinhardt, and J. P. Müller. Multi-dimensional matchmaking for electronic markets. *Applied Artificial Intelligence*, 16(9-10):853–869, 2002.

[19] N. Vulkan and N. R. Jennings. Efficient mechanisms for the supply of services in multi-agent environments. *Decis. Support Syst.*, 28(1-2):5–19, 2000.

[20] G. Weiss. *Multiagent Systems*. The MIT Press, 1999.

Author Index