Lecture Notes in Artificial Intelligence 5384

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

Subseries of Lecture Notes in Computer Science

Iyad Rahwan Pavlos Moraitis (Eds.)

Argumentation in Multi-Agent Systems

Fifth International Workshop, ArgMAS 2008
Estoril, Portugal, May 12, 2008
Revised Selected and Invited Papers

Series Editors

Randy Goebel, University of Alberta, Edmonton, Canada
Jörg Siekmann, University of Saarland, Saarbrücken, Germany
Wolfgang Wahlster, DFKI and University of Saarland, Saarbrücken, Germany

Volume Editors

Iyad Rahwan
British University in Dubai
Faculty of Informatics
Dubai, UAE
and
University of Edinburgh
School of Informatics
Scotland, UK
E-mail: irahwan@acm.org

Pavlos Moraitis
Paris Descartes University
Department of Mathematics and Computer Science
75270, Paris Cedex 06, France
E-mail: pavlos@mi.parisdescartes.fr

Library of Congress Control Number: 2009920684

CR Subject Classification (1998): I.2.11, C.2.4, H.5.2-3, D.3.2, F.4.1

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-642-00206-4 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-00206-9 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12602292 06/3180 5 4 3 2 1 0

Preface

This volume presents the latest developments in the growing area of research at the interface of argumentation theory and multiagent systems. This area has grown tremendously with many papers appearing in the recent special issue of the *Artificial Intelligence Journal* on "Argumentation" and the special issue of IEEE Intelligent Systems on "Argumentation Technologies."

Over the last few years, argumentation has been gaining increasing importance in multiagent systems, mainly as a vehicle for facilitating rational interaction (i.e., interaction which involves the giving and receiving of reasons). This is because argumentation provides tools for designing, implementing and analyzing sophisticated forms of interaction among rational agents. Argumentation has made solid contributions to the practice of multiagent dialogues. Application domains include: legal disputes, business negotiation, labor disputes, team formation, scientific inquiry, deliberative democracy, ontology reconciliation, risk analysis, scheduling, and logistics. A single agent may also use argumentation techniques to perform its individual reasoning because it needs to make decisions under complex preferences policies, in a highly dynamic environment.

Most papers in this volume appeared in the proceedings of the 5th International Workshop on Argumentation in Multiagent Systems (ArgMAS 2008), which took place in Estoril, Portugal, in conjunction with the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS). This continues the success of the ArgMAS workshop series, which took place in tandem with AAMAS in New York in 2004, Utrecht in 2005, Hakodate in 2006, and Honolulu in 2007.

Often we invite additional papers on the topic of argumentation in multiagent systems from the main AAMAS conference for the given year, in order to bring together the very best of the year's work on argumentation in MAS into a single volume. This time, we invited revised papers on argumentation in MAS from AAMAS 2008. The first invited paper, by Tim Miller and Peter McBurney, describes the specification of protocols as first-class entities for annotation and matching. The second invited paper, by Nils Bulling, Carlos I. Chesñevar and Jürgen Dix, explores the use of argumentation in modeling coalition formation processes. The third invited paper, by Yuqing Tang and Simon Parsons, explores how public and private argumentation policies can be integrated.

In addition, we invited a paper by Angelika Foerst, Achim Rettinger and Matthias Nickles, from the AAMAS International Workshop on Agent-Based Complex Automated Negotiations (ACAN). These additional contributions were selected on the basis of their scientific quality and relevance to the topics emphasized here. Our objective has been to offer a comprehensive and up-to-date overview of this rapidly evolving landscape, as we did in the previous volumes of

this series which were all published by Springer (LNAI 3366, LNAI 4049, LNAI 4766, and LNAI 4946).

We conclude this preface by extending our gratitude to the members of the Steering Committee and members of the Program Committee who together helped make the ArgMAS workshop a success. We also thank the authors for their enthusiasm to submit papers to the workshop, and for revising their papers on time for inclusion in this book.

November 2008

Iyad Rahwan
Pavlos Moraitis

Organization

Program Chairs

Iyad Rahwan	British University in Dubai, UAE (Fellow) University of Edinburgh, UK
Pavlos Moraitis	Paris Descartes University, France

ArgMAS Steering Committee

Antonis Kakas	University of Cyprus, Cyprus
Nicolas Maudet	Université Paris Dauphine, France
Peter McBurney	University of Liverpool, UK
Pavlos Moraitis	Paris Descartes University, France
Simon Parsons	City University of New York, USA
Iyad Rahwan	British University in Dubai, UAE (Fellow) University of Edinburgh, UK
Chris Reed	University of Dundee, UK

Program Committee

Leila Amgoud	IRIT, France
Katie Atkinson	University of Liverpool, UK
Jamal Bentahar	Laval University, Canada
Guido Boella	Università di Torino, Italy
Brahim Chaib-draa	Laval University, Canada
Carlos Chesnevar	Universitat de Lleida, Spain
Frank Dignum	Utrecht University, The Netherlands
Yannis Dimopoulos	University of Cyprus, Cyprus
Sylvie Doutre	IRIT, Toulouse, France
Rogier van Eijk	Utrecht University, The Netherlands
Frank Guerin	University of Aberdeen, UK
Joris Hulstijn	Utrecht University, The Netherlands
Anthony Hunter	University College, London, UK
Nikos Karacapilidis	University of Patras, Greece
Nicolas Maudet	Université Paris Dauphine, France
Peter McBurney	University of Liverpool, UK
Jarred McGinnis	Royal Holloway, University of London, UK
Sanjay Modgil	Cancer Research UK
Pavlos Moraitis	Paris Descartes University, France
Søren Holbech Nielsen	Aalborg University, Denmark

Tim Norman	University of Aberdeen, UK
Nir Oren	University of Aberdeen, UK
Fabio Paglieri	ISTC-CNR, Rome, Italy
Xavier Parent	King's College, UK
Simon Parsons	City University of New York, USA
Philippe Pasquier	University of Melbourne, Australia
Laurent Perrussel	IRIT, Toulouse, France
Enric Plaza	Spanish Scientific Research Council, Spain
Henri Prade	IRIT, Toulouse, France
Henry Prakken	Utrecht University, The Netherlands
Alun Preece	University of Aberdeen, UK
Iyad Rahwan	British University in Dubai, UAE (Fellow) University of Edinburgh, UK
Sarvapali Ramchurn	University of Southampton, UK
Chris Reed	University of Dundee, UK
Michael Rovatsos	University of Edinburgh, UK
Hajime Sawamura	Niigata University, Japan
Carles Sierra	IIIA-CSIC, Spain
Guillermo Simari	Universidad Nacional del Sur, Argentina
Elizabeth Sklar	City University of New York, USA
Francesca Toni	Imperial College, London, UK
Leon van der Torre	University of Luxembourg, Luxembourg
Paolo Torroni	Università di Bologna, Italy
Bart Verheij	Maastricht University, The Netherlands
Gerard Vreeswijk	Utrecht University, The Netherlands
Doug Walton	University of Winnipeg, Canada
Simon Wells	University of Dundee, UK
Mike Wooldridge	University of Liverpool, UK

Table of Contents

Part I: Argument-Based Reasoning

Part II: Argumentation and Dialogue

Part III: Strategic and Pragmatic Issues

Personality-Based Practical Reasoning

Thomas L. van der Weide, Frank Dignum, John-Jules Ch. Meyer,
Henry Prakken, and Gerard A.W. Vreeswijk

University of Utrecht
{tweide,dignum,jj,henry,gv}@cs.uu.nl

Abstract. In virtual training scenarios, agent technology can be used to build a virtual tutor that assists a student during training. In a dialogue using argumentation schemes, the virtual tutor provides reasons to the students to explain why a particular action is the most sensible. The tutor determines the best action using practical reasoning. The justification of this action is selected based on the personality type of the student. This paper studies how agent technology could be used to make a virtual tutor that assists the student during the training. In particular, we study how the tutor can generate persuasive arguments for what the student should do.

1 Introduction

The context of this paper is the training of firemen in virtual scenarios, such as the following:

> In a remote place a truck is involved in an accident, and catches on fire. On the truck there is a sign stating that there is gasoline inside. Gasoline is highly flammable and can cause an explosion when set on fire. Near the truck there are several injured people who are not able to move.

In this scenario there are several decisions that the student needs to make. For example, whether to first evacuate the injured people or first to extinguish the fire in the truck. If the student is training as commander, he might have to persuade his team members to take a particular course of action. In this case he has to learn to use the personality type of his team mates in order to give the right commands. E.g. one type of person might be concentrating on the overall situation and miss the sign on the truck. Another might start the standard procedure of using water to extinguish the fire, only thinking about a possible huge explosion, but ignoring the injured people. So, this is the first point where an argumentation dialogue should be supported.

The second situation in which a personality based dialogue can take place in the training scenario is when the student requires feedback while training in virtual scenarios. For example, when the student makes a mistake, he needs to understand what went wrong and why. A virtual tutor can assist a student in this process.

During training the virtual tutor will stop the simulation and engage in a dialogue with the student about the best action at that particular time. This

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 3–18, 2009.

paper focuses on how the personality type of the student can be used in selecting the best justification for a certain action in the case of such a feedback situation during the training. Personality theory explains how individuals of a certain type conduct their reasoning; from this we can ascertain what information they would be most receptive to initially. Presumptive reasoning using argumentation schemes is used to perform practical reasoning, i.e. to reason about what action is the most sensible thing to do.

This paper is structured as follows. After the background has been sketched in section 2, section 3 describes how practical reasoning is done using argumentation schemes, and describes an algorithm to select the best argument based on the personality type of the student. Section 4 shows how the theory can be applied to the scenario. Finally, section 5 provides some conclusions and directions for future work.

2 Background

2.1 Personality Type Theory

Within personality psychology there are many theories that study personality and individual differences. Type theories classify persons into personality types. A popular type theory, the Myers-Briggs Type Indicator (MBTI), is based on the typological theory of [1]. Although the scientific basis of both MBTI and Jung have been questioned, the theory describes aspects that we recognize in everyday life, and seem interesting for agent technology. Furthermore, [2] describes how to adapt communication to an individual using its personality type.

The Myers-Briggs Type Indicator (MBTI), as introduced in [3], is a personality questionnaire designed to identify certain psychological differences as described in [1] work. These psychological differences include attitude, perception function, the judgment function, and lifestyle. The personality type determines what effective communication is. [2] describes how to use MBTI to communicate effectively. There are two important elements in effective communication: what is communicated, and how it is communicated. This paper focuses on what is communicated. As we focus on the content of the message we will use only the Sensing/Intuition preference, and therefore omit three of the four dimensions, type dynamics, and type development. Our theory thus should be seen more as an example of how personality type should be incorporated in the argumentation framework rather than an all encompassing framework.

People who prefer Sensing first want and give information that is real, concrete, practical, factual, and specific, whereas people who prefer Intuition first want and give information that is insightful, opens possibilities, uses the imagination, presents an overview or synthesis, and shows patterns. Sensing people ask what and how questions; they speak of what is or what has been and give precise factual descriptions. Intuition people ask what if and why questions; they speak of what might be, what the main issue is, and what jumped out using 'sort of' and general impression descriptions. All of us can and do use both Sensing

and Intuition to gather information, but each of us has a natural preference for one over the other.

2.2 Practical Reasoning

Using argumentation schemes to derive conclusions is a form of presumptive reasoning which is used commonly in everyday life [4]. Presumptive reasoning is non-monotonic since it is always subject to revision or correction when new information becomes available. Presumptive reasoning using argumentation schemes is used in [5] to do practical reasoning. Our goal is to make a virtual tutor that explains to a student fireman why a particular action is the best action to take in a certain situation. Using argumentation schemes is an appropriate way to do this since people use them naturally, and it allows the student to ask questions and to attack the conclusions.

In [4] Walton presents the argument from consequences that was already present in [6]:

> If action a is brought about,
> then good (bad) consequences will / might occur.
> Therefore, a should (not) be brought about.

However, in [4] (p. 77) Walton notes that the argument from consequences is highly problematic, in light of its treatments in logic textbooks of that time. In [7] Walton presents the following reasoning scheme for practical reasoning called the *sufficient condition scheme for practical reasoning*:

> G is a goal for a,
> doing A is sufficient for a to carry out G,
> therefore, a ought to do A.

The sufficient condition scheme for practical reasoning was later extended in [5] by separating the notion of a goal into: the state of affairs brought about the action, the goal (the desired features in that state of affairs), and the value (the reason why those features are desired). The extended argumentation scheme is as follows:

> $AS1$:
> In circumstances R,
> one should perform action A,
> to achieve new circumstances S,
> which realise goal G,
> which promotes value V

Preferences based upon individual values emerge through the practical reasoning process. [5] uses the term *values* to denote some actual descriptive social attitude/interest which an agent may or may not wish to uphold or subscribe to and they provide an actual subjective reason for wanting to bring about a particular state.

Disagreement about the conclusion is divided into two categories: those that dispute facts, and those that dispute value preferences. This paper takes $AS1$ as a starting point for its argumentation framework since it has a clear connection to BDI agents, which we plan to use for implementing the virtual tutor. It is clear that it does not contain references to personality type. So, the main goal of the paper is to fit elements of personality types in this scheme and the dialogue rules.

3 Basic Formalism

This section describes how to optimise the justification to perform an action for a specific personality type by anticipating on what justification is preferred by that personality type. The justification to do an action is an argumentation scheme with the conclusion that you should perform a particular action. Next, critical questions are described that test the validity of this argumentation scheme, and we give our interpretation of the natural interest that personality types have for specific critical questions. Finally, we describe what kind of answers personality types prefer to hear. We assume that the personality type of the student is known to the system (very simple, fast tests exist to find the MBTI type of a person).

3.1 Basic Notions

In this subsection some basic notions, which are mostly taken from [5], are described which will be used in later sections. We define separate sets containing the basic elements of the framework. A predicate logic is used in the standard way extended with several relations and one function.

- a finite, non-empty set, $State$, of states
- a finite, non-empty set, $Action$, of actions the student can perform
- a finite, non-empty set, $Prop$, of propositions in the predicate logic
- a finite, non-empty set, $Goal$, of goals where $Goal \subset Prop$
- a finite, non-empty set, $Value$, of values
- a relation $results(a, r, s)$ with $a \in Action$ and $r, s \in State$ to be read as: performing action a in state r results in state s
- a relation $realizes(s, g)$ with $s \in State$ and $g \in Goal$ to be read as: state s realizes goal g
- a relation $precludes(a, b)$ with $a, b \in Action$ to be read as: action a precludes action b
- a predicate $oughtToDo(a)$ with $a \in Action$ to be read as: it is sensible to perform action a in the current state
- a function $effect : Goal \times Value \to \{+, 0, -\}$ to be read as: the effect of the given goal on the given value is $+$ when the value is promoted, 0 when there is no effect on the value, and $-$ when the goal demotes the value
- $p \vdash_{arg} q$ to be read as that q can be derived using argumentation schemes from p

Besides these basic elements, in later sections new relations will be introduced when needed.

3.2 Practical Reasoning Scheme

We use a slight modification of the argumentation scheme for practical reasoning as used in [5]. We have modified it by making the premises and the conclusions explicit.

PR:
premise 1: The current state is r,
premise 2: performing action a in state r results in state s,
premise 3: s realizes goal g,
premise 4: g promotes value v,
conclusion: therefore, you should perform a

The argumentation scheme PR provides a justification for the conclusion that you should perform action a. It states that the current state is r, and performing action a leads to a new state s which realizes your goal g. This goal promotes your value v, and therefore action a is good and you should perform it. It assumes that the student wants to promote value v, and that the student creates goals in order to promote value v. The formal notation of PR is

$$r \wedge results(a, r, s) \wedge realizes(s, g) \wedge (effect(g, v) = +) \Rightarrow oughtToDo(a)$$

For example, in our scenario a practical argument that justifies extinguishing the fire in the truck could be: in the current state where there is a fire in the truck and injured people being near, extinguishing the fire will result in no fire in the truck, which will realize the goal of having no explosion, which promotes the value of saving lives.

The argumentation scheme PR is not fool proof as it is a form of presumptive reasoning, hence a person can disagree with it, or can wonder whether the premises actually hold. To test the justification critical questions can be asked to which the proponent should respond. If the proponent cannot give satisfying answers, the conclusion that you should perform a is weakened.

The critical questions in table 1 are based on [5], and test the premises made in PR by asking whether the stated premises are true. Table 1 also provides the corresponding attacks which the receiver of the argumentation scheme can make when he has additional information.

The first four critical questions, CQ1-4, question the explicit premises in PR. CQ5-8 question implicit premises in PR which may not be obvious at first sight, but are relevant for the conclusion.

For example, attacks of CQ5-8 in our scenario could be an alternative action that results in the state that realizes the goal of no explosion, for example by removing the gasoline from the truck. An alternative goal to promote the value of saving lives would be to evacuate the people near the truck. The action of extinguishing the fire also promotes the value of minimising material damage.

3.3 Answers

To provide answers to the critical questions associated with PR, [5] is extended by using argumentation schemes with as conclusion the answer to the critical

Table 1. Critical questions and possible attacks associated PR

	Critical Question	**Possible Attack**
1.	is it true that the current state is r?	$\neg r$
2.	is it true that performing action a in r results in state s?	$\neg results(a, r, s)$
3.	is it true that state s realize goal g?	$\neg realize(s, g)$
4.	is it true that goal g promotes value v?	$\neg promote(g, v)$
5.	are there alternative actions that result in state that realizes goal g?	$a' \neq a \wedge results(a', r, s') \wedge realize(s', g) \wedge s' \neq s$
6.	are there alternative goals that promote value v?	$(effect(g', v) = +) \wedge g' \neq g$
7.	does a demote or promote other values?	$results(a, r, s) \wedge realize(s, g') \wedge \neg(effect(g', v') = 0) \wedge g \neq g' \wedge v \neq v'$
8.	does a preclude another action which promotes some value?	$a \neq a' \wedge results(a', r, s') \wedge realize(s', g') \wedge (effect(g', v') = +) \wedge precludes(a, a')$

question. Walton describes in [4] 25 argumentation schemes, but here only a few are explained that are necessary for the scenario.

Argument from Expert Opinion. The argument from expert opinion states that when a true expert asserts $p \in Prop$ that is within his expertise, then it is reasonable to take p to be true. To represent this argumentation scheme, the following new notions are added to our predicate logic:

- a finite, non-empty set of experts called E
- a finite, non-empty set of domains of expertise called D
- a relation $expert(e, d)$ with $e \in E$ and $d \in D$ to be read as expert e is an expert in the domain d
- a relation $within(p, d)$ with $p \in Prop$ and $d \in D$ to be read as proposition p is within the domain of expertise d
- a relation $assert(e, p)$ with $e \in E$ and $p \in Prop$ to be read as expert e asserted that p is true

Now we can introduce the argument from expert taken from [4]:

AE:
premise: e is an expert in domain d,
premise: e asserts that p is known to be true,
premise: p is within d,
conclusion: therefore, p may (plausibly) be taken to be true.

formally: $expert(e, d) \wedge assert(e, p) \wedge within(p, d) \Rightarrow p$

Table 2 describes the critical questions and corresponding attacks of AE. The critical questions are taken from [4].

Table 2. Critical questions and corresponding possible attacks associated with AE

	Critical Question	**Possible Attack**
1.	Is e a genuine expert in d?	$\neg expert(e, d)$
2.	Did e really assert p?	$\neg assert(e, p)$
3.	Is p relevant to firefighting?	$\neg within(p, d)$
4.	Is p consistent with what other experts in d say?	$expert(f, d) \wedge f \neq e \wedge assert(f, q) \wedge (p \wedge q \vdash \bot)$
5.	Is p consistent with known evidence in d?	$p \wedge q \wedge within(q, d) \vdash \bot$

Our application deals mostly with a single domain, namely firefighting, but in more complex situations there will be other domains like for example health care, police force, and traffic regulation.

Argument from Observation. The argument from observation is a simplified version of the argument from sign and the argument from evidence to hypothesis, both from [4]. The argument from observation is that because I have observed that p is the case, p is the case. For this a new predicate is introduced which is added to the logic:

- a predicate $observation(p)$ with $p \in Prop$ to be read as I have observed that p is the case

With this predicate, we can now introduce the argument from observation:

AO:
premise: p is observed in this situation,
conclusion: therefore, generally p is true

formally: $observation(p) \Rightarrow p$

The associated critical questions and attacks are described in table 3.

Table 3. Critical questions and possible attacks associated with AO

	Critical Question	**Possible Attack**
1.	How certain was the observation?	$\neg observation(p)$
2.	Is the observation consistent with other observations?	$observation(p) \wedge observation(q) \wedge (p \wedge q \vdash \bot)$

Argument from Cause to Effect. The argument from cause to effect, taken from [4], argues that if an event takes place, then that will cause an effect. For example, fire and gasoline brought together will cause an explosion. For this we need a new relation which is added to the predicate logic:

– a relation $cause(s, t)$ with $s, t \in State$ to be read as state s causes state t to occur

The argument from cause to effect is as follows:

ACE:
premise: generally, if p occurs, then q will (or might) occur,
premise: in this case, p occurs,
conclusion: therefore, in this case, p will (or might) occur.

formally: $cause(p, q) \wedge p \Rightarrow q$

The associated critical questions and attacks are described in table 4.

Table 4. Critical questions and possible attacks associated with ACE

	Critical Question	**Possible Attack**
1.	How strong is the causal generalization (if it is true at all)?	$\neg cause(p, q)$
2.	Is the evidence cited strong enough to warrant the generalization as stated?	$p \wedge cause(p, q) \wedge \neg q$
3.	Are there other factor that interfere with or counteract the production of the effect in this case?	$p_2 \wedge (\neg cause(p \wedge p_2, q) \vee cause(p_2, \neg q))$

The above argument schemes and critical questions do not contain personality elements yet. These will be added in the next subsection.

3.4 Personality Types and Practical Reasoning

Table 5 is taken from [2] and and quotes several characteristics of people who prefer Sensing and people who prefer Intuition to provide some insights to the reader of what the Sensing and Intuition functions are. [2] is not written with argumentation schemes nor software agents in mind, so the techniques described in this paper are our interpretation of [2].

[2] explains that the Sensing function considers facts and focuses on what is: what is the problem, what is the purpose or goal, what is the time frame, what is the status of resources, etc. The Intuition function generates possibilities and focuses on what could be: what are the ideas, what are the possibilities, what is the vision, dreams and the ideals.

Table 6 is taken from [2] and provides suggestions how to gear teaching towards people preferring Sensing or Intuition. This table is relevant for our virtual tutor since our tutor teaches the student what to do in a certain situation.

The Sensing Function. A SENSER trusts his senses, experience, and respects what is proven. A thorough body of evidence is built by first focusing on real,

Table 5. Characteristics of the Sensing and Intuition functions

People who prefer Sensing	**People who prefer Intuition**
trust experience	trust hunches and inspirations
respect what is proven	use imagination to create something
first want and give information that is real, concrete, practical, factual, and specific	first want and give information that is insightful, opens possibilities, uses the imagination, presents an overview or synthesis, and shows patterns
give precise descriptions	use 'sort of' and general impression descriptions
give factual statements	give analogies and metaphors

Table 6. Suggestions for teaching to people preferring Sensing and Intuition

People who prefer Sensing	**People who prefer Intuition**
Present facts and realistic details, paying attention to parts of the whole and steps of the process	Present options and possibilities
Provide thorough, concrete data	Provide analogies, symbols, and theoretical models
Allow the listener to build a body of evidence step-by-step from the details to the theory and to interact with the information about practical, hans-on information	Allow the listener to attach details, facts, and steps onto the conceptual idea and to interact with the information through imagination and insights

concrete, practical, factual, and specific information. For example, observations are trusted since they come from the senses, and are real and concrete. With a thorough body of evidence, a detailed understanding is obtained of the problem, the goal, and the relation to the values. With a proper understanding of the goal, a SENSER reasons about what actions will realize the goal. Since the goal is clear, reasoning about the action can be practical, specific, and detailed. To a INTUIT, a SENSER might appear to be literal, narrow-minded, or stuck in a rut.

The practical reasoning process of a SENSER first focuses on obtaining a solid understanding on what is the case and the problem. When presented the practical reasoning scheme PR, a SENSER naturally starts with obtaining a solid understanding by using CQ1 to ask for justification for what the current state r is. When given an answer with the conclusion that r is the case, a recursive process of asking critical questions is started until either observations or expert knowledge is given, or when his knowledge can sufficiently answer the critical questions. With a proper understanding of r, CQ2 can be answered more concretely since the resulting state s from action a depends on r. With a detailed description of s, CQ3 can be answered by justifying that s realizes goal g. Similarly, CQ4 can be answered by justifying that g promotes value v.

When presented a practical reasoning scheme that provides proper justification to perform action a, a SENSER is pretty satisfied because of his practical nature. However, he might reason further, using CQ5, about whether there are alternative actions that realize goal g that are better than the proposed action a. He might also use CQ6 to reason about realizing other goals that promote v, use CQ7 to reason about the effect a has on other values, and use CQ8 to reason about whether a precludes another action that promotes other values.

The Intuition Function. An INTUIT trusts hunches and inspirations, and looks for immediate and long-range implications by using intuition and imagination. Intuitive reasoning is reasoning without being aware of all conscious reasoning steps. Naturally, an INTUIT first wants information that is insightful, opens possibilities, uses the imagination, presents an overview, and shows patterns. Insightful information gives an overview of how to reach a conclusion, but does not give all reasoning steps that are required to reach that conclusion. Since the understanding is based upon hunches and intuition rather than facts and rules, an INTUIT may be wrong about the conclusions he takes. To a SENSER he might appear ungrounded and impractical.

When presented the reasoning scheme PR, the practical reasoning process of an INTUIT starts with CQ1 to get an overview or big picture of the current state r in which he does not make all reasoning steps that are required to derive r. Using hunches about that, for example, some observations indeed lead to a particular situation, the big picture of r is constructed. After getting an overview of r, an INTUIT uses hunches to get an overview of which state s results from action a, and how s realizes goal g and how g promotes v. Now that a global picture is obtained of r and what and why to do a, an INTUIT naturally starts reasoning about possibilities using critical questions CQ5-8. CQ5 asks for other possible actions that realize g. CQ6 asks for other possible goals that promote values, opening again possibilities. CQ7 asks for the effect of a on other values which for example could be long-term effects. CQ7 thus asks for the implication of the facts. CQ8 asks whether a precludes other actions, thereby asking for the implication of and relation between facts.

3.5 Algorithm

This subsection provides two algorithms that generate justifications why a particular action should be done. One algorithm constructs the justification for people who prefer Sensing, the other constructs the justification for people who prefer Intuition. In this subsection, all references to critical questions refer to critical questions of PR. Critical questions of other argumentation schemes are not anticipated on, but can be handled when the student asks for them.

Inside the algorithms some new notation is used. The function $tell(X)$ should be read as tell the student that X. Furthermore, the relations are used here as sets but should be read as all the instantiations of that relation. For example, the set $cause$ is all $cause(p, q)$ that are true with $p, q \in Prop$.

Input: $r \wedge results(a, r, s) \wedge realizes(s, g) \wedge (effect(g, v) = +) \Rightarrow oughtToDo(a)$
// CQ1
given minimal sets $O \subseteq observation$, $E \subseteq expert \cup within \cup assert$, and $C \subseteq cause$ are minimal sets such that $O, E, C \vdash_{arg} r$;
tell($derivation(\{O, E, C\}, r)$);
// CQ2
given minimal sets $rs \subseteq results$, and $c \subseteq cause$ such that $a, r, rs, c \vdash_{arg} s$;
tell($derivation(a, r, \{rs, c\}, s)$);
tell($realize(s, g)$); // CQ3
tell($effect(g, v) = +$) ; // CQ4
// PR: the student should perform a
tell($r \wedge results(a, r, s) \wedge realizes(s, g) \wedge (effect(g, v) = +) \Rightarrow oughtToDo(a)$)

Algorithm 1. How to give an argument for a person who prefers Sensing

Input: $r \wedge results(a, r, s) \wedge realizes(s, g) \wedge (effect(g, v) = +) \Rightarrow oughtToDo(a)$
// CQ1
given minimal sets $O \subseteq observation$, $E \subseteq expert \cup within \cup assert$, and $C \subseteq cause$ are minimal sets such that $O, E, C \vdash_{arg} r$;
tell($O \vdash_{arg} r$);
// CQ2 and CQ3
given minimal sets $r \subseteq results$, and $c \subseteq cause$ such that $a, r, r, c \vdash_{arg} s$;
tell($a, r \vdash_{arg} s \wedge realize(s, g)$);
// CQ5
foreach $a' \in Action$ *with* $a' \neq a \wedge results(a', s') \wedge realizes(s', g)$ **do**
  tell($results(a', s') \wedge realizes(s', g)$);
end
// CQ6
foreach $g' \in Goal \wedge g' \neq g \wedge (effect(g', v) = +)$ **do**
  tell($effect(g', v) = +$);
end
// CQ7
foreach $g' \in Goal$ *with* $g' \neq g \wedge realizes(s, g') \wedge \neg(effect(g', v) = 0)$ **do**
  tell($realizes(s, g') \wedge effect(g', v)$);
end
// CQ8
foreach $a' \in Action$ *with* $a' \neq a \wedge (precludes(a, a') \vee precludes(a', a))$ **do**
  tell($precludes(a, a')$);
end

Algorithm 2. How to give an argument for a person who prefers Intuition

Algorithm 1 gives the justification that is adapted to a student who prefers Sensing. A person who is sensing is mostly interested in CQ1-4 of PR, therefore the algorithm does not anticipate answering CQ5-8 of PR. The notation used inside the algorithms is not precise enough at this stage, but is aimed to bring across the idea. The function $derivation(P, r)$ with $P \subseteq Prop$ and $r \in Prop$ returns all the steps required to derive r from P. A person who prefers Sensing can use this derivation to completely understand why r is the case. The function $derivation(a, r, P, s)$ with $a \in Action$, $r, s \in State$, and $P \subseteq Prop$ gives all the reasoning steps required to derive that when performing action a in state r results in state s. This derivation is now given loosely, where later is may need some action logic.

Algorithm 2 gives the justification that is adapted to a student that prefers Intuition. A person who prefers Intuition is mostly interested in CQ5-8 of PR, but also needs to know the basic answers to CQ1-4. To answer CQ1, the full derivation of why r is the current state is not given, but only the extract, namely the observations and the conclusion. The student will use hunches and intuition to check whether r is indeed the case. Similarly for CQ2, not the full derivation of why performing action a in state r results in state s, but only an extract. CQ3 and CQ4 will be seen by a person who prefers Intuition, so they are not anticipated on. CQ5-8 are answered completely but again without proper derivations.

4 Application

In our scenario the virtual tutor stops the simulation and gives the student arguments that the student should extinguish the fire. However, the tutor must first reason about what to do. Next, the mental state of the tutor is described using atoms as described in table 7.

Table 7. The meaning of the literals used in our scenario

Atom	**Meaning**
fire	there is a fire in the truck
people	there are injured people near the truck
gasoline	there is gasoline in the truck
explosion	the truck explodes
death	the injured people near the truck will die
saveLives	the value that as many as possible lives should be saved
minDamage	the value that the amount of material damage should minimized
extFire	the action where the fireman extinguishes the fire in the truck
evacuate	the action where the fireman evacuates the injured people near the truck
pump	the action where the fireman pumps the gasoline out of the truck

The sets are instantiated as follows

$$\begin{aligned} Action &= \{extFire, evacuate, pump\} \\ Goal &= \{g_1 \equiv \neg explosion \wedge \neg death, g_2 \equiv \neg people \wedge \neg death\} \\ Value &= \{saveLives, minDamage\} \\ E &= \{e\} \\ D &= \{firefighting\} \end{aligned}$$

This means that the tutor believes that there are three actions that the student can perform: extinguish the fire in the truck, evacuate the injured people, and pump the gasoline out of the truck. Furthermore, the tutor has two goals: g_1 and g_2 where g_1 is the goal of realizing no explosion in the truck and preventing the injured people to die, and g_2 is the goal of realizing that the injured people are not near the truck and do not die. The tutor considers two values: saving lives, and minimizing damage. There is only one expert, namely e, and there is only one domain of expertise, namely firefighting.

The relations and predicates are instantiated as follows. For simplicity, all the statements that can be made are within the firefighting domain of expertise.

$$\begin{aligned} &\{observation(fire), observation(gasoline), observation(people), \\ &assert(e, fire \wedge gasoline \rightarrow explosion), \\ &\forall p \in Prop[within(p, firefighting)], \\ &results(extFire, fire, \neg fire), results(evacuate, people, \neg people), \\ &results(pump, gasoline, \neg gasoline), \\ &precludes(extFire, evacuate), precludes(extFire, pump), \\ &precludes(evacuate, pump), \\ &realizes(\neg explosion, g_1), realizes(\neg gasoline, g_1), realizes(\neg people, g_2), \\ &cause(explosion \wedge people, death)\} \end{aligned}$$

Finally, the effects of the goals on the values are as follows:

$$\begin{aligned} effect(g_1, saveLives) &= + \\ effect(g_1, minDamage) &= + \\ effect(g_2, saveLives) &= + \\ effect(g_2, minDamage) &= - \end{aligned}$$

4.1 Practical Reasoning

First, the argument from observation, AO, is used to evaluate the observations.

$$observation(fire) \Rightarrow fire \quad (1)$$

$$observation(gasoline) \Rightarrow gasoline \quad (2)$$

$$observation(people) \Rightarrow people \quad (3)$$

Next, the argument from expert opinion, AE, is used to conclude that *explosion* will occur. Since we use a simplification, namely every $p \in Prop$ is within the domain of expertise, the within clause is always true and therefore not used.

$$expert(e, firefighting) \wedge assert(e, fire \wedge gasoline \rightarrow explosion) \Rightarrow fire \wedge gasoline \rightarrow explosion \quad (4)$$

The argument from cause to effect, ACE, is used to conclude that *death* will occur:

$$cause(explosion \wedge people, death) \wedge explosion \wedge people \Rightarrow death \quad (5)$$

From $results(extFire, fire, \neg fire)$ can be concluded that $\neg fire$ and therefore $\neg explosion$. Since $realize(\neg explosion, g_1)$ we can conclude that g_1 is realized and because of $effect(g_1, saveLives) = +$ we can construct the practical reasoning scheme PR which is instantiated as follows:

$$\begin{aligned} r &= death, \\ a &= extFire, \\ s &= \neg explosion, \\ g &= g_1 \\ v &= saveLives \end{aligned}$$

4.2 Argument for the Student

In this subsection the algorithms as presented in subsection 3.5 are applied to construct justification for that the student should extinguish the fire.

Students Who Prefer Sensing. In algorithm 1 the tutor starts by giving the derivation of how the current state r can be derived from observations, expert knowledge, and causal knowledge. As such the tutors tells the student the complete derivation of r from the previous subsection.

Next, the tutor tells the student the complete derivation of why performing action a in state r results in state s. This derivation will be something like:

$$results(extFire, fire, \neg fire) \vdash_{arg} \neg fire \vdash_{arg} \neg explosion$$

Next, the tutor tells that $realizes(\neg explosion, g_1)$ where g_1 is $\neg explosion \wedge \neg death$ which means that the state that results from performing $extFire$ realizes goal g_1. Next, the student is told that $effect(g_1, saveLives) = +$ which means that goal g_1 promotes the value to save lives. Finally, the student is told the whole practical reasoning scheme with the conclusion that $extFire$ should be performed: $r \wedge results(a, r, s) \wedge realizes(s, g) \wedge (effect(g, v) = +) \Rightarrow oughtToDo(a)$.

When transformed into text, this argument given by the tutor could look as follows:

> I observe that there is a fire in the truck and the truck contains gasoline. Experts say that gasoline is highly flammable, so when the fire reaches the gasoline, there will be a big explosion. I observe that there are several persons near the truck who are injured and cannot get away. If there is a big explosion in the truck, those people will die. Because of my value to save lives, I have the goal to save those people near the truck. The action 'extinguish the fire' will prevent the fire from reaching the gasoline, which prevents an explosion, and thus saves those people from dying because of the explosion. Therefore, you should extinguish the fire.

Students Who Prefer Intuition. In algorithm 2 the tutor starts by giving an overview of the current state r:

$$observation(fire) \wedge observation(gasoline) \wedge observation(people) \vdash_{arg} r$$

Next, an overview is given of that performing action $extFire$ in r results in state s by telling that:

$$extFire \wedge r \vdash_{arg} s \wedge realize(s, g)$$

Next, CQ5 will be anticipated on by giving the alternative actions that realize goal g. The tutor tells the student the following:

$$results(pump, \neg gasoline) \wedge realizes(\neg gasoline, g_1)$$
$$results(evacuate, \neg people) \wedge realizes(\neg people, g_2)$$

CQ6 asks for all other goals that promote value $saveLives$. In this case, the tutor will tell $effect(g_2, v) = +$. Next, the tutor answers CQ7 by telling whether $extFire$ promote or demotes other values. In this case, the tutor tells the student that $effect(extFire, minDamage) = +$. Next, CQ8 is answered by telling the student that $precludes(extFire, pump)$ and $precludes(extFire, evacuate)$. From this, the student will conclude that $extFire$ should be done.

When transformed into text, this argument given by the tutor could look as follows:

> Because there is a fire in the truck and there is gasoline in the truck, the truck will explode killing the pople that are near. By extinguishing the fire in the truck, the explosion will not occur, and thus the people will survive. Pumping the gasoline out of the truck might also prevent an explosion, and evacuating the injured people may result in that those people are not near the truck when it explodes. Extinguishing the fire minimizes material damage. However, extinguishing the fire precludes pumping away the gasoline and evacuating the people.

5 Conclusions

In this paper we have demonstrated how arguments can be adapted to personality types. We have concentrated on only one aspects of personalities: the sensing/intuition dimension. Because this dimension mainly influences the content

of the arguments it alters the argumentation basically at the level of argument formation. The argument schemes that are used are almost identical to the classical ones used in the literature, but the way critical questions are chosen and answers are given is different for different personality types.

It is obvious that much work remains to be done in order to fully integrate personality types in the argumentation framework. However, we already have shown that it can make a crucial difference in the way argumentation functions. Especially in time critical situations where not all critical questions can be asked at leisure, it is important to start with the ones that most connect to the personality type and give the answers that provide most needed information.

We plan to extend the present work in several ways. First we will make the formalisation more precise such that we can reason more formally about things like "cause", "effect", etc.

We also will incorporate the other personality dimensions in the framework and check the consequences. Finally we aim at performing some real-time experiments in virtual training in order to see the effects of our approach.

References

1. Jung, C.: Psychological Types (1921)
2. Zeisset, C.: The art of dialogue. Center for Applications of Psychological Type, Ince (2006)
3. Myers, I.: The Myers-Briggs Type Indicator. Princeton, NJ (1962)
4. Walton, D.N.: Argumentation Schemes for Presumptive Reasoning. Lawrence Erlbaum Associates, Mahwah (1996)
5. Atkinson, K., Bench-Capon, T., McBurney, P.: Computational representation of practical argument. Synthese 152(2), 157–206 (2006)
6. Windes, R., Hastings, A.: Argumentation & Advocacy. Random House (1965)
7. Walton, D.N.: Practical Reasoning: Goal-Driven, Knowledge-Based, Action-Guiding Argumentation. Rowman & Littlefield (1990)

Argumentation Based Resolution of Conflicts between Desires and Normative Goals

Sanjay Modgil and Michael Luck

Department of Computer Science, Kings College London

Abstract. Norms represent what ought to be done, and their fulfillment can be seen as benefiting the overall system, society or organisation. However, individual agent goals (desire) may conflict with system norms. If a decision to comply with a norm is determined exclusively by an agent or, conversely, if norms are rigidly enforced, then system performance may be degraded, and individual agent goals may be inappropriately obstructed. To prevent such deleterious effects we propose a general framework for argumentation-based resolution of conflicts amongst desires and norms. In this framework, arguments for and against compliance are arguments justifying rewards, respectively punishments, exacted by 'enforcing' agents. The arguments are evaluated in a recent extension to Dung's abstract argumentation framework, in order that the agents can engage in *meta-level* argumentation as to whether the rewards and punishments have the required motivational force. We provide an example instantiation of the framework based on a logic programming formalism.

1 Introduction

Requirements for conflict resolution arise in open multi-agent systems in which goals of individual agents conflict with norms imposed by the system to regulate individual agent behaviour. If the decision to comply with a norm is determined exclusively by an individual, then system performance may be degraded. Hence, institutional or social pressure to comply may be brought about by *system agents* exacting punishments and grants rewards [11,17]. This may be appropriate for closed static systems, but compromises the flexibility of dynamic open systems in which rigid enforcement of norms may lead to both unwarranted obstruction of agent goals and degraded system performance. For example, an agent's goal may be obstructed by enforcing compliance with a norm that is justified by system-held beliefs about the context. However, these beliefs may be erroneous. In addition, it may not always be able to anticipate at design time, contexts in which compliance with norms does or does not coincide with the best interests of the system, and when enforcement mechanisms have insufficient motivational force. In such cases, an agent might appeal to higher level *motivations* [9], arguing that in pursuing its own goal it is indeed acting in the interests of the system as a whole, or that exacted punishments (or rewards) for non-compliance (or compliance) are outweighed by the benefits of pursing its own goal.

In this paper we propose a general argumentation-based framework that evaluates arguments for and against compliance with norms, in order to prevent unwarranted obstruction of individual goals and degraded system performance. As in [6,11], norms

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 19–36, 2009.

are interpreted as *system goals* that individual agents are required to realise, and that may conflict with the *individual goals* or *desires* of an agent. Punishments and rewards are the individual goals of system agents responsible for enforcement. In general, an argument for a goal justifies realisation of that goal based on beliefs that are themselves the outcome of argumentation based reasoning about what is the case. An argument for a system goal may then mutually attack an argument for a conflicting individual goal, and arguments for punishment and reward goals attack the argument for an individual goal. It is the success of these attacks that determines which of the arguments prevail and thus whether or not there is a reasoned case for compliance[1]. In general, an attack succeeds as a defeat if the attacked argument is not stronger than or *preferred* to its attacker [1]. As in [4], preferences may be derived from a relative ordering on the values that the arguments promote. In this paper, preferences among arguments for goals are similarly evaluated. For example, a 'reward argument' will successfully attack (defeat) an argument for an individual goal if an agent is persuaded that the reward is of greater utility to it than the individual goal it is required to abandon in favour of compliance with the system goal. The proposed framework will thus need to account for:

1. **Social mechanisms for enforcing compliance**: An agent Ag's argument for an individual goal g may be attacked by arguments for the (punishment and reward) goals $g', \ldots$ of *other* agents, where the attacks are not based on direct conflicts between g and $g', \ldots$. For example, a reward (punishment) may facilitate (hinder) some other goal that Ag is already committed to realising.
2. **Motivational argumentation**: Flexible and adaptive agents need to engage in motivational argumentation over the respective merits of goals. Hence, argumentation frameworks in which preferences [1] and value orderings [4] on arguments are 'given', and not themselves subject to reasoning, do not suffice. Rather, there is a requirement for argumentation based reasoning over the preferences themselves.

Existing work addresses argumentation-based resolution of conflicts among goals ([2], [10], [16]), and [16] explicitly considers conflicts between individual goals and norms. However, no existing work accounts for social mechanisms, whereby an agent's decision as to which goals to pursue is influenced by other agents' goals. Only [10] accounts for argumentation over preferences, but does so in the object level logic programming language, whereby rules express priorities over other rules. In this paper, we aim at an abstract framework in which preferences are not restricted to rule priorities, but can account for any criteria for valuating argument strength, including those that relate to the argument as a whole (e.g., as in [4]). We therefore make use of a recent extension to Dung's seminal abstract argumentation semantics [8]. In a Dung framework, arguments are related by a binary conflict-based relation, and the winning (justified) arguments under different extensional semantics are evaluated. The underlying logic, and definition of the logic's constructed arguments and conflict relation, is left unspecified, enabling instantiation by various logical formalisms. Dung's semantics thus serves as a general framework capturing various species of non-monotonic reasoning [5], and, more generally for conflict resolution. Hence, approaches to argumentation based agent reasoning

[1] In philosophical parlance we are adopting an *externalist* rather than *internalist* view, where the latter consider norms to be intrinsically motivating.

often conform to these semantics, whereby an agent's inferences (e.g. denoting beliefs or goals) can be defined in terms of the claims of the justified arguments constructed from the underlying theory (an argument essentially being a proof of a candidate inference — the argument's claim — in the underlying logic). In [12,13], Dung's semantics have been extended to accommodate arguments that *express preferences between other arguments*, where no assumption is made as to how these preferences are defined in the instantiating formalism.

In Section 2 we review the extended semantics described in [12,13]. The main contributions of this paper are then described in Sections 3, 4 and 5. In Section 3 we describe a general framework for argumentation based resolution of conflicts between system norms and agent goals. Specifically, we combine the extended argumentation semantics with the normative model of [11] in which compliance with norms is enforced through punishments and rewards modelled as the goals of enforcement agents. The framework thus provides for social mechanisms for enforcing compliance, and motivational argumentation. Section 4 then describes a logic programming instantiation of the general framework. Section 5 illustrates the instantiation with an extended example. Finally, Section 6 concludes with a discussion of related and future work.

2 Extended Argumentation Frameworks for Agent Reasoning

2.1 Dung's Argumentation Framework

A Dung argumentation framework is a tuple $(Args, \mathcal{R})$ where $\mathcal{R} \subseteq (Args \times Args)$ can denote either an 'attack' or 'defeat' relation, and where the latter can be understood as an attack that succeeds given the available preference information. An argument $A \in Args$ is defined as acceptable w.r.t. some $S \subseteq Args$, if for every B such that $(B, A) \in \mathcal{R}$, there exists a $C \in S$ such that $(C, B) \in \mathcal{R}$. Intuitively, C 'reinstates' A. Dung then defines the acceptable extensions of $(Args, \mathcal{R})$ under different extensional semantics. In this paper we focus on the admissible and preferred semantics. Letting $S \subseteq Args$ be conflict free if no two arguments in S are related by $\mathcal{R}$, then:

Definition 1. *Let $S \subseteq Args$ be a conflict free set.*

- *S is admissible iff each argument in S is acceptable w.r.t. S*
- *S is a preferred extension iff S is a set inclusion maximal admissible extension*

An argument is said to be justified if it belongs to all preferred extensions of a framework.

2.2 Motivating Extended Argumentation Frameworks

We now motivate extending Dung's framework with the following example (that will be referred to later in Section 3).

Example 1. Consider two agents $Ag1$ and $Ag2$ exchanging arguments A, $B \ldots$ about the weather forecast for Hawaii.

$Ag2$: "According to the BBC it will be cool in Hawaii" = A

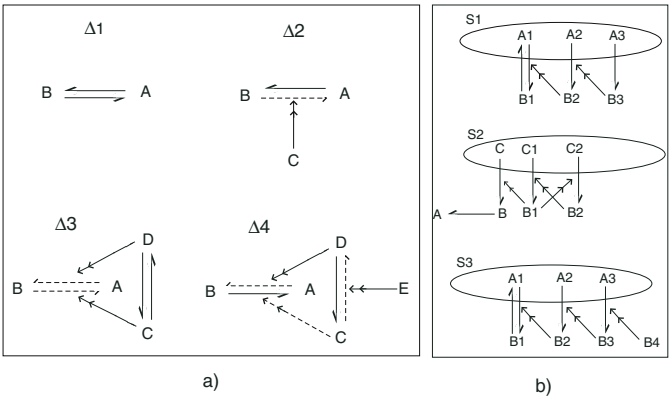


Fig. 1. Motivating *EAF*s

$Ag1$: "According to CNN it will be hot in Hawaii" = B
$Ag2$: "But the BBC are more trustworthy than CNN" = C
$Ag1$: "However, statistics show CNN are more accurate than the BBC" = D
$Ag1$: "And a statistical comparison is more rigorous and rational than basing a comparison on your instincts about their relative trustworthiness" = E

Arguments A and B symmetrically attack, i.e., (A, B),$(B, A) \in \mathcal{R}$. $\{A\}$ and $\{B\}$ are the preferred extensions, and so neither argument is justified. We then have argument C claiming that A is preferred to B. Hence B does not successfully attack (defeat) A, but A does defeat B. Intuitively, C is an argument for A's repulsion of, or defence against, B's attack on A; i.e., C attacks B***'s attack on*** A ($\Delta 2$ in Figure 1a)) so that B's attack on A does not succeed as a defeat. B's attack on A is, as it were, cancelled out, and we are left with A defeating B. Evaluating the preferred extensions on the basis of $\mathcal{R}$ denoting the defeat relation, we now have the single preferred extension containing A. Now, given D claiming a preference for B over A and so attacking A's attack on B, neither defeat the other and so once again we have two preferred extensions. Since C and D claim contradictory preferences they attack each other ($\Delta 3$). These attacks can themselves be subject to attacks in order to determine the defeat relation between C and D and, in so doing A and B. E attacks the attack from C to D ($\Delta 4$), so that D defeats C, B defeats A, and $Ag1$'s argument that it will be hot in Hawaii is now justified.

2.3 Defining Extended Argumentation Frameworks

Example 1 illustrates requirements for arguments attacking attacks. Hence, as in [12,13], an *Extended Argumentation Framework* is defined as follows:

Definition 2. *An* Extended Argumentation Framework *(*EAF*) is a tuple (*$Args$*,* $\mathcal{R}$*,* $\mathcal{D}$*) such that* $Args$ *is a set of arguments, and:*

- $\mathcal{R} \subseteq Args \times Args$
- $\mathcal{D} \subseteq (Args \times \mathcal{R})$
- *If* $(C, (B, A))$*,* $(D, (A, B)) \in \mathcal{D}$ *then* (C, D)*,* $(D, C) \in \mathcal{R}$

Notation 1. We may write $A \rightharpoonup B$ to denote $(A, B) \in \mathcal{R}$. If in addition $(B, A) \in \mathcal{R}$, then $A \rightleftharpoons B$. Also, $D \twoheadrightarrow (A \rightharpoonup B)$ denotes $(D, (A, B)) \in \mathcal{D}$.

The defeat relation is now parameterised w.r.t. some set S of arguments. This accounts for an attack's success as a defeat being relative to preference arguments already accepted in some set S, rather than relative to some externally given preference ordering.

Definition 3. *A defeats$_S$ B, denoted by $A \rightarrow^s B$, iff $(A, B) \in \mathcal{R}$ and $\neg\exists D \in S$ s.t. $(D,(A, B)) \in \mathcal{D}$.*

Referring to Example 1, A defeats$_\emptyset$ B but A does not defeat$_{\{D\}}$ B. The notion of a conflict free set S of arguments is now defined. Notice that it may be that an argument A asymmetrically attacks an argument B, so that given $D \twoheadrightarrow (A \rightharpoonup B)$, neither A nor B defeat$_S$ each other if $D \in S$. This means that both A and B may be accepted together in the same extension (where any extension is required to be conflict free). For example, if B is an argument for an action, and A claims that (for example) the action is too costly, it may be that an agent decides to execute the action while accepting that it is expensive (in value based argumentation [4], D is an argument claiming that the value promoted by B's action is greater than A's value of 'cost'). In the following section we will show that such *preference dependent asymmetric* attacks are also appropriate when resolving conflicts between norms and desires.

Definition 4. *S is conflict free iff $\forall A, B \in S$: if $(A, B) \in \mathcal{R}$ then $(B, A) \notin \mathcal{R}$, and $\exists D \in S$ s.t. $(D,(A, B)) \in \mathcal{D}$.*

The definition of acceptability of an argument A w.r.t. a set S for an *EAF* is motivated in some detail in [12,13]. It references the notion of a *reinstatement set* for a defeat, in order that an intuitive requirement on what it means for an argument to be acceptable w.r.t. an admissible set S of arguments is satisfied: *if A is acceptable with respect to S, then $S \cup \{A\}$ is admissible* (referred to as the fundamental lemma in Dung [8]).

Definition 5. *Let $S \subseteq Args$ in $(Args, \mathcal{R}, \mathcal{D})$. Let $R_S = \{X_1 \rightarrow^S Y_1, \ldots, X_n \rightarrow^S Y_n\}$ where for $i = 1 \ldots n$, $X_i \in S$. Then R_S is a reinstatement set for $C \rightarrow^S B$, iff:*

- *$C \rightarrow^S B \in R_S$, and*
- *$\forall X \rightarrow^S Y \in R_S$, $\forall Y'$ s.t. $(Y',(X, Y)) \in \mathcal{D}$, $\exists X' \rightarrow^S Y' \in R_S$*

Definition 6. *Let $(Args, \mathcal{R}, \mathcal{D})$ be an* EAF. *$A \in Args$ is acceptable w.r.t. $S \subseteq Args$ iff $\forall B \in Args$ s.t. $B \rightarrow^S A$, $\exists C \in S$ s.t. $C \rightarrow^S B$ and there is a* reinstatement set *for $C \rightarrow^S B$.*

In Figure 1b), $A1$ is acceptable w.r.t. $S1$. We have that $B1 \rightarrow^{S1} A1$, and $A1 \rightarrow^{S1} B1$. The latter defeat is itself *challenged* by $B2$. However, $A2 \rightarrow^{S1} B2$, which in turn is challenged by $B3$. But then, $A3 \rightarrow^{S1} B3$. We have the reinstatement set $\{A1 \rightarrow^{S1} B1, A2 \rightarrow^{S1} B2, A3 \rightarrow^{S1} B3\}$ for $A1 \rightarrow^{S1} B1$. Also, A is acceptable w.r.t. $S2$ given the reinstatement set $\{C \rightarrow^{S2} B, C1 \rightarrow^{S2} B1, C2 \rightarrow^{S2} B2\}$ for $C \rightarrow^{S2} B$. Finally $A1$ is not acceptable w.r.t $S3$ since no argument in $S3$ defeats$_{S3}$ $B4$.

Admissible and preferred semantics for *EAF*s are now given by Definition 1, where conflict free is defined as in Definition 4. (In [12,13], the complete, stable and grounded

semantics are similarly defined for *EAF*s, i.e., in the same way as for Dung frameworks). Referring to Example 1, $\{B, D, E\}$ is the single preferred extension. In [12,13] we show that *EAF*s inherit many of the fundamental results holding for extensions of a Dung framework. This suggests that much of the work building on Dung's framework can readily be reformulated for *EAF*s, including work on argument game proof theories and dialogue frameworks. In particular, Dung's fundamental lemma is satisfied, implying that the set of all admissible extensions of an *EAF* form a complete partial order w.r.t. set inclusion, and so for each admissible S there exists a preferred extension S' such that $S \subseteq S'$.

To conclude, the extended semantics accommodates arguments that express preferences between other arguments, while preserving the abstraction of a Dung framework; no commitments are made to how preferences are defined in the instantiating logical formalism. We now make use of the extended semantics in a framework for conflict resolution in normative systems, and show that the ability to engage in argumentation based reasoning *about*, as well as *with*, defeasible and possibly conflicting preference information, provides for agent flexibility and adaptability.

3 A Framework for Conflict Resolution in Normative Systems

This section describes a framework in which agents engage in dialogues to decide which amongst conflicting desire based and normative goals are to be pursued. Agent submit arguments for goals, where these arguments attack each other, and then engage in motivational argumentation over the relative utility of states in which the goals are realised. This equates to arguing over preferences between arguments, and so which attacks succeeds as defeats. The arguments and attacks defined in the course of a dialogue thus instantiate an *EAF*, and the goals to be pursued are those claimed by the justified arguments of the *EAF*. Note that the agents also argue over the beliefs justifying adoption of goals. In this way, the agents are first required to agree that the goal being proposed for adoption is indeed warranted by what is believed to be the case. Section 3.1 first sets out some general assumptions about the kinds of agents modelled by the framework, and the dialogues these agents participate in. Section 3.2 then describes how conflicts between individual agent goals and system norms are resolved through argumentation based dialogues over beliefs, and goals proposed by individual agents and agents acting on behalf of the system.

3.1 Agents and Dialogues

The proposed framework abstracts from the logics for agent reasoning, assuming only *BDI* type agents (e.g. those instantiating the *BOID* architecture [7]) and a declarative interpretation of goals as beliefs holding in some future state. Each agent has a belief base consisting of facts and rules, and a goal base containing rules for deriving goals. From amongst all the goals that are derivable, those that an agent commits to realising are referred to as intentions. An intention persists in an agent's intention base until such a time as it is realised by a plan (the agent's planning component is not modelled here).

As in [11]'s model of normative multi-agent systems, four types of goal are distinguished. Individual agent goals, which we refer to here as *desires*, may conflict with *normative goals*. For example, an agent $Ag1$'s desire to stay on Waikiki beach in Hawaii, may conflict with the normative goal of staying in a cheap hotel. $Ag1$ may decide to comply or not comply with the norm, based on rewards and punishments exacted by system agents (specifically *enforcement* agents). Rewards and punishments are also individual goals of enforcement agents, but are punishment, respectively reward goals, from the perspective of the agent being punished, respectively rewarded. *Punishment goals* hinder the punished agent's intentions if that agent decides not to comply with the norm. For example, a punishment may be to deny the funding that $Ag1$ needs to fulfill its intention to visit Leipzig for a meeting. *Reward goals* benefit the achievement of the rewarded agent's intentions if it decides to comply. For example, a reward for an agent who intends to have a laptop, may be to provide the agent with a laptop.

In general, goals are derived by rules whose antecedents refer to what the agent believes and its current intentions. Extending the scenario described in Example 1, suppose agent $Ag1$ believes it will be hot in Hawaii, and it intends to attend a conference in Hawaii. Then it derives the desire to stay on Waikiki beach. The goals of system agents are derived in the same way, and may additionally refer to the intentions of other agents. For example, if $Ag1$ intends to attend a conference, then the normative goal of staying in a cheap hotel is derived (in either $Ag1$'s goal base or the goal base of a system agent responsible for informing other agents of their obligations). An enforcement agent Ag_P may derive the punishment goal of denying $Ag1$ the funding for a meeting, given $Ag1$'s intention to attend the meeting, and Ag_P's belief that the meeting is not related to an EU project. Rules in the goal base can also capture the sub-goal relationship. For example, if $Ag1$ intends to visit Leipzig for a meeting, then it derives the sub-goal goal of having funding for the visit. Finally, we assume argument construction from agents' bases is defined in some underlying logic.

Definition 7. *Let $\{Ag_1, \ldots, Ag_n\}$ be a set of agents, where for $i = 1 \ldots n$, Ag_i is equipped with a belief base $\mathcal{B}_i$, an intention base $\mathcal{I}_i$, and a goal base $\mathcal{G}_i$. For $i = 1$, let argument A be constructed from $\mathcal{B}i \cup \mathcal{G}i \cup \bigcup_{i=1}^{n} \mathcal{I}i$.*

If A is constructed only from $\mathcal{B}i$, then A is a belief argument *of Ag_i, otherwise A is a* goal argument *of Ag_i.*

In general, we write $bel(A)$ to denote the beliefs in A. We also write claim*(A) to identify an argument A's claim.*

The basic idea is that individual and system agents engage in argumentation-based conflict resolution (*persuasion*) dialogues to determine which amongst the arguments for beliefs and goals are justified in the *EAF*s of Section 2. The goals that are the claims of justified arguments are then adopted as intentions. In persuasion dialogues (reviewed in [3]) a proponent makes a claim — the *topic* of the dialogue — and (one or more) opponents attempt to persuade the proponent that the claim does not hold. In general such a dialogue d is a sequence of moves $m_1, \ldots, m_i, \ldots$, where the first move m_1 is a locution introducing the topic as an assertion or claim of an argument. Here, we simply assume that the topic of d can be referred to as *topic*(d). Dialogue *protocols* vary from model to model, and specify the legal moves at each stage of the dialogue, where a move can be an assertion of a proposition or an argument, a challenge to a premise in

an argument, a concession of a proposition or argument, and so on. Models also vary on the rules for termination of a dialogue. However, in general, the arguments submitted and constructed (from the propositions asserted) during the course of a dialogue can be organised into an argumentation framework [15]. If an argument for the topic is justified, then the proponent wins the dialogue. Formalising dialogue models is to be addressed in future work. Here, we refer only to an *EAF* constructed on the basis of a dialogue.

Definition 8. *Let* $d = m_1, \ldots, m_n$ *be a terminated dialogue where* topic$(d) = \alpha$*, and* $AG = \{Ag_1, \ldots, Ag_m\}$ *the participants in* d*. We say that the* EAF $\Delta = (Args, \mathcal{R}, \mathcal{D})$ *constructed on the basis of* d*, is:*

- *a belief* EAF *iff every argument in* $Args$ *is a belief argument of some* $Ag \in AG$
- *a goal* EAF *iff every argument in* $Args$ *is either a belief or goal argument of some* $Ag \in AG$[2].

3.2 Arguing about Beliefs and Goals

An agent's argument A for a desire may conflict with (and so mutually attack) an argument B for a normative goal. Arguments for punishment and reward goals may in turn attack A and so reinstate the argument B for the normative goal. The success of these attacks as defeats depends on argumentation over preferences between the arguments (corresponding to meta-level motivation-based argumentation over the relative utility of states in which the goals are realised).

Prior to agents submitting goal arguments in a dialogue, the beliefs in the argument justifying the goal may themselves by subject to debate[3]. In our running example, $Ag1$'s desire to stay on Waikiki beach is contingent on its belief that it will be hot in Hawaii. A system agent may successfully persuade $Ag1$ that it will be cool in Hawaii. Hence $Ag1$ will not submit the argument for its desire, precluding the possibility of norm violation (in Example 1 the outcome is in favour of $Ag1$'s argument that it *will* be hot). Furthermore, the beliefs in arguments for system goals may be challenged. Thus, an agent may successfully argue that the beliefs justifying a normative goal may be erroneous; hence the normative goal does not have to be adopted and unwarranted obstruction of the conflicting desire is prevented. Suppose arguments A and B for the conflicting goals of staying on Waikiki and staying at a cheap hotel have been submitted. Ag_P will not submit an argument C for the punishment goal of denying $Ag1$ funding for the Leipzig meeting, if $Ag1$ successfully persuades Ag_P that the meeting *is* related to an EU project. Again, this prevents unwarranted obstruction of $Ag1$'s intention to attend the meeting. Of course, Ag_P may then be motivated to submit an argument for an alternative punishment goal to enforce compliance.

[2] Of course, in the limiting case where only arguments can be submitted as locutions, then each m_i in d corresponds to an argument in $Args$, and a protocol for d would require that m_i attack some m_j, $j < i$, or some attack between m_j and m_k, $j < i, k < i$.

[3] Arguing over beliefs justifying a goal *prior* to arguing over the relative merits of goals precludes 'wishful thinking'; i.e., one wouldn't want that argumentation over which goals to adopt (which future state to realise) influences what is believed about the current state of the world.

Definition 9. *Let $AG = \{Ag_1, \ldots, Ag_n\}$. Then A is an* **agreed** *goal argument of $Ag \in AG$ if for every $\alpha \in bel(A)$:*

if there is a terminated dialogue d with topic α, participants $AG \subseteq \{Ag_1, \ldots, Ag_n\}$, and Δ is a belief EAF *constructed on the basis of d, then α is the conclusion of a justified argument of Δ.*

We now describe how argumentation over goals proceeds. Consider the case where a normative goal g' conflicts with a desire g (in the simplest case $g' \equiv \neg g$ in the underlying logic). In general, we say that the goal argument A' for g' *conflicts* with the goal argument A for g. In a goal *EAF*, A and A' attack each other since an agent can either adopt g and not g', or g' and not g.

Suppose such an EAF, where $Ag1$ submits A claiming 'stay on Waikiki beach', and A' claiming 'stay in cheap hotel', mutually attacks A. An enforcement agent can then submit an argument P for a punishment goal p, that, in the terminology of [11], *hinders* some intention of $Ag1$. In our running example, p = 'deny funding for meeting'. Now P does not directly attack on A's goal; it does so in the sense that if the attack succeeds, then $Ag1$ will not pursue its desire, and will comply with the norm. Note also, that the attack is a preference dependent asymmetric attack. $Ag1$ might argue (B) that it is of more value to him to stay on Waikiki beach then attend the meeting. That is, $B \twoheadrightarrow (P \rightharpoonup A)$, and it may now be that A *and* P are justified; $Ag1$ adopts its desire, *and* accepts the punishment. An alternative punishment may then need to be submitted to see if it has the required enforcing effect. Finally, an enforcement agent can submit an argument R for a reward goal r, that, in the terminology of [11], *benefits* some intention of Ag. For example r = 'provide the agent with a laptop', benefiting $Ag1$'s intention to have a laptop. R symmetrically attacks A. Either $Ag1$ accepts the reward and drops the desire, or vice versa.

Definition 10. *Let AG=$\{Ag_1, \ldots, Ag_n\}$ be a set of agents. Let $Args_G$=$\bigcup_{i=1\ldots n}\{A|A$ is a goal argument for $Ag_i\}$. Let $\mathcal{I}_{AG} = \bigcup_{i=1\ldots n}\mathcal{I}i$. Then:*

- conflicts $\subseteq Args_G \times Args_G$
- hinders $\subseteq Args_G \times \mathcal{I}_{AG}$
- benefits $\subseteq Args_G \times \mathcal{I}_{AG}$[4]

Definition 11. *Let $AG = \{Ag_1, \ldots, Ag_n\}$, and for some $Ag \in AG$, let A be a goal argument of Ag, $\mathcal{I}$ the intention base of Ag.*

Let A' be the goal argument of some $Ag' \in AG$, $Ag' \neq Ag$. Then:

- A' goal attacks A *and* A goal attacks A' *if* conflicts*(A', A) or* benefits*(A', ι) for some $\iota \in \mathcal{I}$*
- A' goal attacks A *if* hinders*(A', ι) for some $\iota \in \mathcal{I}$*

We now specify some constraints on a dialogue that begins with a topic that is a goal proposed for adoption as an intention. We do so by expressing constraints on the goal *EAF* constructed on the basis of the dialogue. These are that the goal arguments are agreed, and can only be attacked by goal arguments as defined above, and only belief arguments are used in arguing over the relative merits of the goals.

[4] Note that an agent's desires may 'internally' conflict. We so not here directly address conflict resolution in such cases. Note also that an agent's goals may benefit/hinder its own intentions.

Definition 12. *Let $AG = \{Ag_1, \ldots, Ag_n\}$ be a set of agents. Let d be a terminated dialogue with topic α, and participants $AG' \subseteq AG$, where:*

- *α is the conclusion of an agreed goal argument A of some agent $Ag \in AG'$.*
- *$\Delta = (Args, \mathcal{R}, \mathcal{D})$ is the goal* EAF *constructed on the basis of d, where:*
  i) for any goal arguments $B, A \in Args$, $(B, A) \in \mathcal{R}$ iff A and B are agreed goal arguments, and B goal attacks A.
  ii) If $(C,(B, A)) \in \mathcal{D}$ then C is a belief argument for some agent in AG'

If the topic α of the dialogue is an agent's desire, and α is the claim of a justified argument in the dialogue's goal *EAF*, then α is updated to the agent's intention base, and any punishment goal that is the claim of a justified argument is updated to the corresponding enforcement agent's intention base. If α is not the claim of a justified argument, and there is a justified argument for a normative goal β, then β is updated to the agent's intention base, and any reward goal that is the claim of a justified argument is updated to the corresponding enforcement agent's intention base.

4 Instantiating the Framework

In this section we describe an example instantiation of the framework. Agent goals, beliefs and intentions are represented in [14]'s *argument based logic programming with defeasible priorities* (ALP-DP). An ALP-DP theory's arguments are defined as sequences of chained rules. Some rules can express priorities on other rules, so that one can construct *priority arguments* whose claims determine preferences between other mutually attacking arguments. Preferences between priority arguments can also be established on the basis of other priority arguments. [14] then defines the justified arguments of a theory under Dung's grounded semantics. In [12,13] the arguments and attacks defined by an ALP-DP theory instantiate an *EAF*, and an equivalence result with the *EAF*'s justified arguments (under the grounded semantics) is shown. By giving an *EAF* semantics for ALP-DP one can, unlike [14], also:

1. characterise the justified arguments of an ALP-DP theory under the preferred semantics; and
2. model preference dependent asymmetric attacks.

Both these features are employed when instantiating an *EAF*. Note that ALP-DP models both negation as failure and strict negation. To simplify the presentation, we describe a restricted version of ALP-DP — ALP-DP* — which does not include negation as failure.

Definition 13. *Let (S, D) be a ALP-DP* theory where S is a set of strict rules of the form $s : L_0 \wedge \ldots \wedge L_m \rightarrow L_n$, D a set of defeasible rules $r : L_0 \wedge \ldots \wedge L_j \Rightarrow L_n$, and:*

- *Each rule name r (s) is a first order term. Henceforth,* head*(r) denotes the consequent L_n of the rule named r.*
- *Each L_i is an atomic first order formula, or such a formula preceded by strong negation $\neg$.*

Strict rules represent information that is beyond debate (note that neither $\rightarrow$ nor $\Rightarrow$ admit contraposition). We also assume that the language contains a two-place predicate symbol $\prec$ for expressing priorities on rule names, and that any S includes the following strict rules expressing the properties of a strict partial order on $\prec$:

- $o1 : (x \prec y) \wedge (y \prec z) \rightarrow (x \prec z)$
- $o2 : (x \prec y) \wedge \neg(x \prec z) \rightarrow \neg(y \prec z)$
- $o3 : (y \prec z) \wedge \neg(x \prec z) \rightarrow \neg(x \prec y)$
- $o4 : (x \prec y) \rightarrow \neg(y \prec x)$

Definition 14. *An argument A based on the theory (S, D) is:*

1. *a finite sequence $[r_0, \ldots, r_n]$ of ground instances of rules such that*
   - *for every i ($0 \leq i \leq n$), for every literal L_j in the antecedent of r_i there is a $k < i$ such that* head*(r_k) = L_j.*
     We say that claim*(A) =* head*(r_n), and if* head*(r_n) = $x \prec y$ then A is called a 'singleton priority argument'.*
   - *no distinct rules in the sequence have the same head*

   or
2. *a finite sequence $[r_{0_1}, \ldots r_{n_1}, \ldots, r_{0_m}, \ldots r_{n_m}]$, such that for i=1 ... m, $[r_{0_i}, \ldots r_{n_i}]$ is a singleton priority argument. We say that A is a 'composite priority argument' and* claim*(A) =* head*(r_{n_1})...*head*(r_{n_m}) is the ordering claimed by A*

In [14], arguments are exclusively defined by item 1. We additionally define composite priority arguments so that an ordering, and hence a preference, can be claimed by a single argument rather than a set of arguments (as in [14]).

Definition 15. *For any arguments A, A' and literal L:*

- *A is strict iff it does not contain any defeasible rule; it is defeasible otherwise.*
- *L is a conclusion of A iff L is the head of some rule in A*
- *If T is a sequence of rules, then $A + T$ is the concatenation of A and T*

Note that an argument has only one claim, but may have many conclusions corresponding to the heads of the contained rules. We now instantiate the abstract definition 7 of an agent and its constructed arguments. Note that intentions are represented by the goal arguments that have previously been used to justify their adoption. Hence, an agent's goal arguments will be constructed from its belief and goal base, and the claims (named by the name of the rule whose head is the claim) of intention arguments in all agents' intention bases.

Definition 16. *Let $\{Ag_1, \ldots, Ag_n\}$ be a set of agents, where for $i = 1 \ldots n$:*
- $\mathcal{B}_i$ and $\mathcal{G}_i$ are ALP-DP theories, and $\mathcal{I}_i$ is a set of arguments.*
- A is a belief argument of Ag_i iff it is based on $\mathcal{B}_i$
- A is a goal argument of Ag_i iff it is based on $\mathcal{B}_i \cup \mathcal{G}_i \cup \bigcup_{i=1\ldots n}\{r : claim(B) | B \in \mathcal{I}_i, head(r) = claim(B)\}$

[14] motivates definition of attacks between arguments that account for the ways in which arguments can be extended with strict rules:

Definition 17. *$A1$ attacks $A2$ on the pair $(L, \neg L)$ if there are sequences S1 and S2 of strict rules such that $A1 + S1$ is an argument with conclusion L and $A2 + S2$ is an argument with a conclusion $\neg L$.*

In the following example illustrating attacks between belief arguments, we will without loss of generality simply assume that all beliefs are contained in a single theory. Only in the example at the end of this section, in which we illustrate argumentation over goals, will we identify the individual agents involved. Following [14], every rule with terms $t_1, \ldots, t_n$ is named with a function expression $r(t_1, \ldots, t_n)$ where r is the rules's informal name. For example, $r(p(X,Y), q(X,Y))$ names the rule $p(X,Y) \Rightarrow q(X,Y)$. To maintain readability we will only write the function-symbol part of the rule name, and as an abuse of notation, arguments will be represented as sequences of rule names rather than the rules these names identify.

Example 2. Let $tr(X,Y)$, $st(X,Y)$ and $ra(X,Y)$ respectively denote that X is more trustworthy, statistically accurate, and rational than Y.

Let $S = \{o1 \ldots o4\} \cup \{s1 : temp(X, cool) \rightarrow \neg temp(X, hot),$
$s2 : temp(X, hot) \rightarrow \neg temp(X, cool)\}$.

Let $D = \{bbc :\Rightarrow temp(hawaii, cool),$
$cnn :\Rightarrow temp(hawaii, hot),$
$c1 :\Rightarrow tr(bbc, cnn),$
$d1 :\Rightarrow st(cnn, bbc),$
$c2 : tr(X,Y) \Rightarrow Y \prec X,$
$d2 : st(X,Y) \Rightarrow Y \prec X,$
$e1 :\Rightarrow ra(d2, c2),$
$e2 : ra(X,Y) \Rightarrow Y \prec X\}$

$A = [bbc]$, $B = [cnn]$, $C = [c1, c2]$, $D = [d1, d2]$.
$E = [e1, e2]$ with conclusions $ra(d2, c2)$ and $c2 \prec d2$, and claim $c2 \prec d2$.
A and B attack each other since $A + s1$ has conclusion $\neg temp(hawaii, hot)$ and B has conclusion $temp(hawaii, hot)$. C and D attack each other since C has conclusion $cnn \prec bbc$ and $D + o4$ has conclusion $\neg(cnn \prec bbc)$

We now define the relations *conflicts*, *hinders* and *benefits*, and goal attacks for ALP-DP* goal arguments. Note that the notion of *benefits* requires that a goal argument of a rewarding agent be extended (as in the definition of attack) with strict rules that link the reward goal to the intention that it benefits (this will be illustrated in the example concluding this section).

Definition 18. *Let A be a goal argument of an agent Ag and $\mathcal{I}$ the intention base of Ag.*

Let B be any goal argument of an agent Ag', where $\mathcal{B}' = (S', D')$ is the belief base of Ag'. We say that:

• conflicts*(B, A) if B attacks A as in definition 17.*

For any $I \in \mathcal{I}$:

• hinders*(B, I) if B attacks I as in definition 17*

• benefits(B, I) *if* claim$(B + S1)$ = claim(I) *for some possibly empty sequence of strict rules* $S1$ *in* S'

Then:

- B *and* A *goal attack each other on the pair* $(claim(B), claim(A))$ *if* conflicts(B, A)
- B *and* A *goal attack each other on the pair* $(claim(B), claim(A))$ *if* benefits(B, I) *for some* $I \in \mathcal{I}$
- B *goal attacks* A *on the pair* $(claim(B), claim(A))$ *if* hinders(B, I) *for some* $I \in \mathcal{I}$

To determine a preference amongst attacking arguments, [14] defines the sets of relevant *defeasible* rules to be compared, and an ordering on these sets. Here, the ordering on such sets is based on the ordering claimed by a given priority argument.

Definition 19. *If* $A+S$ *is an argument with conclusion* L*, the defeasible rules* $R_L(A+S)$ ***relevant*** *to* L *are:*

1. $\{r_d\}$ *iff* A *includes defeasible rule* r_d *with head* L
2. $R_{L_1}(A + S) \cup \ldots \cup R_{L_n}(A + S)$ *iff* A *is defeasible and* S *includes a strict rule* $s : L_1 \wedge \ldots \wedge L_n \rightarrow L$

Definition 20. *Let* C *be a priority argument claiming the ordering* $\prec$*. Let* R *and* R' *be sets of defeasible rules. Then* $R' > R$ *iff* $\forall r' \in R', \exists r \in R$ *such that* $r \prec r'$*.*

Intuitively, R can be made better by replacing some rule in R with any rule in R', while the reverse is impossible. Now, given two arguments A and B, it may be that for belief arguments they attack on more than one conclusion. For goal arguments they goal attack on a single pair of conclusions (the goals claimed by the arguments). Given a priority ordering $\prec$ claimed by argument C, we say that A is preferred$_\prec$ to B if for every pair (L, L') of conclusions on which they attack, the set of A's defeasible rules relevant to L is stronger $(>)$ than the set of B's defeasible rules relevant to L'.

Definition 21. *Let* C *be a priority argument claiming* $\prec$*. Let* $(L_1, L'_1), \ldots, (L_n, L'_n)$ *be the pairs on which* A *attacks, or goal attacks* B*, where for* $i = 1 \ldots n$*,* L_i *and* L'_i *are conclusions in* A *and* B *respectively. Then* A *is preferred*$_\prec$ *to* B *if for* $i = 1 \ldots n$*,* $R_{L_i}(A + S_i) > R_{L'_i}(B + S'_i)$

In example 2, C and D attack each other on the pair ($cnn \prec bbc$, $\neg(cnn \prec bbc)$), and $R_{cnn \prec bbc}(C)$ = $\{c2\}$, $R_{\neg(cnn \prec bbc)}(D)$ = $\{d2\}$. E claims $c2 \prec d2$, and so D is preferred$_{c2 \prec d2}$ to C. Note also, that given C, A is preferred$_{cnn \prec bbc}$ to B, and given D, B is preferred$_{bbc \prec cnn}$ to A. We can now instantiate an *EAF* with the arguments, their attacks, and priority arguments claiming preferences and so attacking attacks:

Definition 22. *The* EAF *(*$Args$*,* $\mathcal{R}$*,* $\mathcal{D}$*) for a theory* (S, D) *is defined as follows.* $Args$ *is the set of arguments given by definition 14, and* $\forall A, B, C \in Args$*:*

1. *(C,*$(B, A)) \in \mathcal{D}$ *iff* C *claims* $\prec$ *and* A *is preferred*$_\prec$ *to* B
2. (A, B)*,*$(B, A) \in \mathcal{R}$ *if* A *and* B *attack as in definition 17, or* A *and* B *goal attack as in definition 18*

The belief *EAF* obtained by the arguments and attacks for our running example is shown in figure 1a). $\{E, D, B\}$ is the single preferred extension of the *EAF*. We can now constrain a goal EAF constructed on the basis of a dialogue between agents, as defined in definition 12.

5 An Extended Example

We now illustrate the previous section's formalism with an extended version of our Hawaiian example, in which we assume that every goal argument is agreed.
In what follows we use the following shorthand:

ha = 'Hawaii', wa = 'Waikiki beach', le = 'Leipzig', att, = 'attend', $conf$ = 'conference', $meet$, = 'meeting', $cheap$ = 'cheap hotel', lap = 'laptop', $fund$ = 'have funding', and $deny\_f$ = 'deny funding'.

Also, predicates may refer to the agents themselves. For example, $att(ag, conf, ha)$ denotes the goal of ag to attend a conference in Hawaii. Also, variables will begin with uppercase letters and constants with lowercase letters. For example, $deny\_f(ag_P, AgX, meet, L)$ denotes the goal of agent ag_P to deny funding for any agent AgX to attend a meeting in some location L.

Let $\{ag_a, ag_N, ag_P, ag_R\}$ be a set of agents. We describe each agent's knowledge bases. Note that we may not show all the goal rules used to construct arguments in the intention base of each agent. Also, as before, we may simply write the rule name rather than the rule the name identifies.

ag_a:
$\mathcal{I}$ =
$\{$ $[ia1 :\Rightarrow att(ag_a, conf, ha)]$, $[ia2 :\Rightarrow att(ag_a, meet, le)]$,
$[ia2 :\Rightarrow att(ag_a, meet, le), ia3 : att(ag_a, meet, le) \Rightarrow funds(ag_a, meet, le)]$,
$[ia4 :\Rightarrow have(ag_a, lap)]$ $\}$
$\mathcal{G}$ =
$\{$ $ga1 : temp(ha, hot) \wedge att(ag_a, conf, ha) \Rightarrow stay(ag_a, wa)\}$
$\mathcal{B}$ =
$\{$ $ba0 :\Rightarrow temp(ha, hot)$,
$ba1 :\rightarrow norm\_des(gn1, ga1)$,
$ba\_self : norm\_des(X, Y) \Rightarrow X \prec Y$,
$ba2 :\Rightarrow project\_funds(high)$,
$ba3 : project\_funds(high) \Rightarrow except(ba\_self, bn\_social)$,
$ba\_excep : except(X, Y) \Rightarrow Y \prec X$,
$ba4 :\Rightarrow gp1 \prec ga1\}$

ag_N:
$\mathcal{I} = \emptyset$
$\mathcal{G} = \{$ $gn1 : att(AgX, conf, L) \Rightarrow stay(AgX, cheap, L)\}$
$\mathcal{B} = \{$ $bn1 : stay(AgX, cheap, ha) \rightarrow \neg stay(AgX, wa)$,
$bn2 :\rightarrow norm\_des(gn1, ga1)$,
$bn\_social : norm\_des(X, Y) \Rightarrow Y \prec X$ $\}$

ag_P:
$\mathcal{I} = \emptyset$
$\mathcal{G}$ = { $gp1 : att(AgX, meet, L) \wedge \neg type(meet, eu, L)$
$\Rightarrow deny\_f(ag_P, AgX, meet, L)$}
$\mathcal{B}$ = { $bp1 :\Rightarrow \neg type(meet, eu, le)$,
$bp2 : deny\_f(ag_P, AgX, meet, L) \rightarrow \neg funds(AgX, meet, L)$}

ag_R:
$\mathcal{I} = \emptyset$
$\mathcal{G}$ = { $gr1 : have(AgX, lap) \Rightarrow provide(ag_R, AgX, lap)$}
$\mathcal{B}$ = { $br1 : provide(ag_R, AgX, lap) \rightarrow have(AgX, lap)$,
$br2 :\rightarrow rew\_des(gr1, ga1)$,
$br\_rew\_suffice : rew\_des(X, Y) \Rightarrow Y \prec X$ }
1) ag_a initiates a dialogue with goal argument $A1$ = $[ba0, ia1, ga1]$ claiming the goal $stay(ag_a, wa)$, having already persuaded a system agent that it will indeed be hot in Hawaii.

2) ag_N submits $A2$ = [$ia1$,$gn1$] ($AgX = ag_a$, $L = ha$), where $A2$ and $A1$ goal attack each other (see figure 2) on the pair
$(stay(ag_a, cheap, ha), stay(ag_a, wa))$.

This symmetric goal attack is based on *conflicts* $(A2, A1)$ which obtains because $A2 + [bn1]$ and $A1$ attack (as in def.17) on the conclusion pair $(\neg stay(ag_a, wa), stay(ag_a, wa))ag_N$ also submits the social ordering argument $B1$ = $[bn2, bn\_social]$ claiming $ga1 \prec gn1$, and so $B1 \twoheadrightarrow (A1 \rightharpoonup A2)$.

3) ag_a submits:
- the selfish ordering argument $B2$ = $[ba1, ba\_self]$ claiming $gn1 \prec ga1$, and so $B2 \twoheadrightarrow (A2 \rightharpoonup A1)$

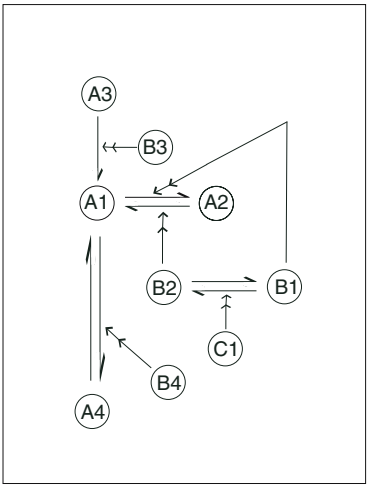


Fig. 2. *EAF* based on argumentation based dialogue over goals

- an argument claiming that the selfish behaviour type is preferred to the social behaviour type given the exceptional circumstances in which the remaining project budget is high:
$C1$ = [$ba2, ba3, ba\_excep$,] claiming $bn\_social \prec ba\_self$, and so $C1 \twoheadrightarrow (B1 \rightharpoonup B2)$.

The single preferred extension contains $A1$

4) ag_P attempts to enforce compliance by submitting $A3$ = [$ia2, bp1, gp1$] given that it is agreed that the meeting is not an Eu project meeting.
$A3 + [bp2]$ attacks (as in def.17), and so hinders, ag_a's intention [$ia2, ia3$]. Hence, $A3$ goal attacks $A1$ on the pair $(deny\_f(ag_P, ag_a, meet, le), stay(ag_a, wa))$.

5) However, ag_a prefers to stay on the beach and be denied funding by ag_P for the leipzig meeting. It may be that ag_a has another source of funding in mind. We do not encode the rationale for the preference, but simply assume the priority argument $B3$ = [$ba4$] claiming $gp1 \prec ga1$. Hence $B3 \twoheadrightarrow (A3 \rightharpoonup A1)$. Since $A3$'s attack on $A1$ is asymmetric:

The single preferred extension contains $A1$ **and** $A3$

6) ag_R attempts to enforce compliance with $A4$ = [$ia4, gr1$] offering to provide ag_a with a laptop. This benefits ag_a's intention to have a laptop since *claim*([$ia4, gr1$] + [$br1$]) = *claim*[$ia4$]. Hence, $A4$ and $A1$ goal attack each other ($A4 \rightleftharpoons A1$) on the pair $(provide(ag_R, ag_a, lap), stay(ag_a, wa))$.

ag_R believes the reward is of sufficient strength that ag_a will prefer the reward to staying on Waikiki beach. ag_R submits $B4$ = [$br2, br\_rew\_suffice$] claiming $ga1 \prec gr1$. Hence, $B4 \twoheadrightarrow (A1 \rightharpoonup A4)$. This is accepted by ag_a and the dialogue terminates.

The single preferred extension contains $A2$ **and** $A4$

ag_a's intention set can then be updated with $A2$. ag_R's intention set can then be updated with $A4$. ag_a intends now to book a cheap room in Hawaii, and ag_R intends to provide ag_a with a laptop.

6 Conclusions

In this paper we have proposed a framework for argumentation-based resolution of conflicts in normative multi-agent systems, and have illustrated instantiation of the framework with a logic programming formalism. The framework provides for agents to argue over the beliefs justifying goals, conflicting preferences brought to bear in argumentation over beliefs, and metalevel motivational argumentation over the states represented by desire based goals, and normative, punishment and reward goals argued for by other agents. In this way, unwarranted obstruction of individual agents' desires is precluded, and enforcement of compliance can appropriately account for the motivations of the agents and erroneously held beliefs about the contexts in which the agents find themselves.

As mentioned in Section 1, existing approaches to argumentation-based resolution of conflicts amongst goals ([2],[10],[16]) do not model social mechanisms deployed to enforce compliance with norms. In [16], norms are represented as bridge rules that

describe the relationships between mental attitudes. Argumentation based resolution of conflicts amongst goals derived using these rules exploits a preference relation on these rules. In [2], only conflicts amongst desire based goals are addressed. Argumentation over the beliefs that justify desires conforms to the Dung semantics. However selection of desires does not account for their relative importance and does not conform to the Dung semantics. Rather, the maximal (under set inclusion) sets of desires that can be consistently realised are chosen. However, goal selection *does* account for the feasibility of plans for realising goals, and this is a factor that our work needs to account for in future work.

Future work will also investigate instantiation of the framework by formalisms with explicit BDI type modalities. Further work is also required before evaluation of the framework based on prototypical implementations can proceed. In particular, we intend development of argument game proof theories, algorithms and dialogue protocols for *EAF*s. Since *EAF*s inherit the fundamental results shown for Dung frameworks, our approach will adopt the methodologies deployed in specification of game based algorithms [18] and protocols [15] based on the Dung semantics. We also believe that our approach is applicable to resolution of conflicts arising between an individual agent's conflicting desires, and between conflicting norms. Both cases often require reasoning about abstract values and motivations. Furthermore, conflict resolution may lead to refinement and evolution of a system's norms. Finally, one of the key novel features of our framework is that an agent's decision as to which goals to pursue is influenced by other agents' goals. We believe that we can abstract from the normative application of the framework to consider other contexts in which the impact of other agents' goals can be modelled through argumentation based mechanisms.

Acknowledgements. The research described in this paper is partly supported by the European Commission Framework 6 funded project CONTRACT (INFSO-IST-034418). The opinions expressed herein are those of the named authors only and should not be taken as necessarily representative of the opinion of the European Commission or CONTRACT project partners.

References

1. Amgoud, L.: Using Preferences to Select Acceptable Arguments. In: Proc. 13th European Conference on Artificial Intelligence, pp. 43–44 (1998)
2. Amgoud, L., Kaci, S.: On the generation of bipolar goals in argumentation-based negotiation. In: Rahwan, I., Moraïtis, P., Reed, C. (eds.) ArgMAS 2004. LNCS, vol. 3366, pp. 192–207. Springer, Heidelberg (2005)
3. ASPIC Deliverable D2.1: Theoretical frameworks for argumentation. (June 2004), `http://www.argumentation.org/PublicDeliverables.htm`
4. Bench-Capon, T.J.M.: Persuasion in Practical Argument Using Value-based Argumentation Frameworks. Journal of Logic and Computation 13(3), 429–448 (2003)
5. Bondarenko, A., Dung, P.M., Kowalski, R.A., Toni, F.: An abstract, argumentation-theoretic approach to default reasoning. Artificial Intelligence 93, 63–101 (1997)
6. Broersen, J., Dastani, M., Hulstijn, J., van der Torre, L.W.N.: Goal Generation in the BOID Architecture. Cognitive Science Quarterly Journal 2(3-4), 428–447 (2002)

7. Dastani, M., van der Torre, L.: Programming BOID-Plan Agents: Deliberating about Conflicts among Defeasible Mental Attitudes and Plans. In: Proc 3rd Int. Joint Conference on Autonomous Agents and Multiagent Systems, pp. 706–713 (2004)
8. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77, 321–357 (1995)
9. d'Inverno, M., Luck, M.: Understanding agent systems, 2nd edn. Springer, Heidelberg
10. Kakas, A., Moraitis, P.: Argumentation based decision making for autonomous agents. In: Proc. Second international joint conference on autonomous agents and multiagent systems, pp. 883–890 (2003)
11. Lopez, F., Lopez, Y., Luck, M., D'Inverno, M.: A normative framework for agent-based systems. J. Computational and Mathematical Organization Theory 12(2-3), 227–250 (2006)
12. Modgil, S.: An Abstract Theory of Argumentation That Accommodates Defeasible Reasoning About Preferences. In: Mellouli, K. (ed.) ECSQARU 2007. LNCS, vol. 4724, pp. 648–659. Springer, Heidelberg (2007)
13. Modgil, S.: Reasoning About Preferences in Argumentation Frameworks. Technical Report, `http://www.dcs.kcl.ac.uk/staff/modgilsa/ArguingAboutPreferences.pdf`
14. Prakken, H., Sartor, G.: Argument-based extended logic programming with defeasible priorities. Journal of Applied Non-Classical Logics 7, 25–75 (1997)
15. Prakken, H.: Coherence and flexibility in dialogue games for argumentation. Journal of logic and computation 15(6), 1009–1040 (2005)
16. Gaertner, D., Toni, F.: Conflict-free normative agents using assumption-basedargumentation. In: Rahwan, I., Parsons, S., Reed, C. (eds.) Argumentation in Multi-Agent Systems. LNCS, vol. 4946. Springer, Heidelberg (2008)
17. Moses, Y., Tennenholtz, M.: Artificial Social Systems. Computers and AI 14(6), 533–562 (1995)
18. Vreeswijk, G.: An algorithm to compute minimally grounded and admissible defence sets in argument systems. In: Proc. 1st International Conference on Computational Models of Argument, pp. 109–120 (2006)

A Constrained Argumentation System for Practical Reasoning

Leila Amgoud[1], Caroline Devred[2], and Marie-Christine Lagasquie-Schiex[1]

[1] IRIT–UPS, Toulouse - France
{amgoud,lagasq}@irit.fr
[2] LERIA, Angers - France
devred@info.univ-angers.fr

Abstract. *Practical reasoning* (PR), which is concerned with the generic question of what to do, is generally seen as a two steps process: (1) *deliberation*, in which an agent decides what state of affairs it wants to reach –that is, its *desires*; and (2) *means-ends reasoning*, in which the agent looks for plans for achieving these desires. A desire is *justified* if it holds in the current state of the world, and *feasible* if there is a plan for achieving it. The agent's *intentions* are thus a consistent subset of desires that are both justified and feasible. This paper proposes the first argumentation system for PR that computes in one step the intentions of an agent, allowing thus to avoid the drawbacks of the existing systems. The proposed system is grounded on a recent work on constrained argumentation systems, and satisfies the rationality postulates identified in argumentation literature, namely the *consistency* and the *completeness* of the results.

Keywords: Argumentation, Practical Reasoning.

1 Introduction

Practical reasoning (PR) [16], is concerned with the generic question "what is the right thing to do for an agent in a given situation". In [22], it has been argued that PR is a two steps process. The first step, often called *deliberation*, consists of identifying the desires of an agent. In the second step, called *means-end reasoning*, one looks for ways for achieving those desires, *i.e.* for actions or plans. A desire is *justified* if it holds in the current state of the world, and is *feasible* if it has a plan for achieving it. The agent's intentions, what the agent decides to do, is a consistent subset of desires that are both justified and feasible.

What is worth noticing in most works on practical reasoning is the use of arguments for providing reasons for choosing or discarding a desire as an intention. Indeed, several argumentation-based systems for PR have been proposed in the literature [3,13,15]. However, in most of these works, the problem of PR is modeled in terms of at least two separate systems, each of them capturing a given step of the process. Such an approach may suffer from a serious drawback. In fact, some desires that are not feasible may be accepted at the deliberation step to the detriment of other justified and feasible desires. Moreover, the properties of those systems are not investigated.

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 37–56, 2009.

This paper proposes the first argumentation system that computes the intentions of an agent in one step. The system is grounded on a recent work on *constrained* argumentation systems [9]. These last extend the well-known general system of Dung [10] by adding constraints on arguments that need to be satisfied by the extensions returned by the system. Our system takes then as input i) three categories of arguments: *epistemic* arguments that support beliefs, *explanatory* arguments that show that a desire holds in the current state of the world, and *instrumental* arguments that show that a desire is feasible, ii) different conflicts among those arguments, and iii) a particular constraint on arguments that captures the idea that for a desire to be pursued it should be both feasible and justified. This is translated by the fact that in a given extension each instrumental argument for a desire should be accompanied by at least an explanatory argument in favor of that desire. The output of our system is different sets of arguments as well as different sets of intentions. The use of a constrained system makes it possible to compute directly the intentions from the extensions. The properties of this system are deeply investigated. In particular, we show that its results are safe, and satisfy the rationality postulates identified in [5], namely consistency and completeness.

The paper is organized as follows: Section 2 recalls the basics of a constrained argumentation system. Section 3 presents the logical language. Section 4 studies the different types of arguments involved in a practical reasoning problem, and Section 5 investigates the conflicts that may exist between them. Section 6 presents the constrained argumentation system for PR, and its properties are given in Section 7. The system is then illustrated in Section 8.

2 Basics of Constrained Argumentation

Argumentation is an established approach for reasoning with inconsistent knowledge, based on the construction and the comparison of arguments. Many argumentation formalisms are built around an underlying logical language and an associated notion of logical consequence, defining the notion of argument. The argument construction is a monotonic process: new knowledge cannot rule out an argument but gives rise to new arguments which may interact with the first argument. Since knowledge bases may give rise to inconsistent conclusions, the arguments may be conflicting too. Consequently, it is important to determine among all the available arguments, the ones that are ultimately "acceptable". In [10], an abstract argumentation system has been proposed, and different acceptability semantics have been defined.

Definition 1. ([10] – Basic argumentation system) *An* argumentation system *is a pair* $\mathsf{AF} = \langle \mathcal{A}, \mathcal{R} \rangle$ *with* $\mathcal{A}$ *is a set of arguments, and* $\mathcal{R}$ *is an attack relation (*$\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$*).*

Before recalling the acceptability semantics of Dung [10], let us first introduce some useful concepts.

Definition 2. ([10] – Conflict-free, Defence) *Let* $\mathcal{E} \subseteq \mathcal{A}$.

- $\mathcal{E}$ *is* conflict-free *iff* $\nexists\ \alpha, \beta \in \mathcal{E}$ *such that* $\alpha\ \mathcal{R}\ \beta$.
- $\mathcal{E}$ defends *an argument* α *iff* $\forall\ \beta \in \mathcal{A}$, *if* $\beta\ \mathcal{R}\ \alpha$, *then* $\exists\ \delta \in \mathcal{E}$ *such that* $\delta\ \mathcal{R}\ \beta$.

Dung's semantics are all based on a notion of admissibility.

Definition 3. ([10] – Acceptability semantics) *Let $\mathcal{E}$ be a set of arguments.*

- *$\mathcal{E}$ is an* admissible *set iff it is conflict-free and defends every element in $\mathcal{E}$.*
- *$\mathcal{E}$ is a* preferred extension *iff it is a maximal (w.r.t. set-inclusion) admissible set.*
- *$\mathcal{E}$ is a* stable extension *iff it is a preferred extension that attacks all arguments in $\mathcal{A}\backslash\mathcal{E}$.*

Note that every stable extension is also a preferred one, but the converse is not always true.

The above argumentation system has been generalized in [9]. The basic idea is to explicit *constraints* on arguments that should be satisfied by the above Dung's extensions. For instance, one may want that the two arguments α and β belong to the same stable extension. These constraints are generally expressed in terms of a propositional formula built from a language using $\mathcal{A}$ as an alphabet.

Definition 4. ([9] – Constraints on arguments, Completion of a set of arguments) *Let $\mathcal{A}$ be a set of arguments and $\mathcal{L}_{\mathcal{A}}$ be a propositional language defined using $\mathcal{A}$ as the set of propositional variables.*

- *C is a* constraint *on $\mathcal{A}$ iff C is a formula of $\mathcal{L}_{\mathcal{A}}$.*
- *The* completion *of a set $\mathcal{E} \subseteq \mathcal{A}$ is: $\widehat{\mathcal{E}} = \{\alpha \mid \alpha \in \mathcal{E}\} \cup \{\neg\alpha \mid \alpha \in \mathcal{A} \setminus \mathcal{E}\}$. A set $\mathcal{E} \subseteq \mathcal{A}$* satisfies *$C$ iff $\widehat{\mathcal{E}}$ is a model of C ($\widehat{\mathcal{E}} \vdash C$).*

A constrained system is defined as follows:

Definition 5. ([9] – Constrained argumentation system) *A* constrained argumentation system *is a triple $\mathsf{CAF} = \langle \mathcal{A}, \mathcal{R}, C\rangle$ with C is a constraint on arguments of $\mathcal{A}$.*

Let us recall how Dung's extensions are extended in constrained systems. As said before, the basic idea is to compute Dung's extensions, and then to keep among those extensions the ones that satisfy the constraint C.

Definition 6. ([9] – C-admissible set) *Let $\mathcal{E} \subseteq \mathcal{A}$. $\mathcal{E}$ is* C-admissible *iff*

1. *$\mathcal{E}$ is admissible,*
2. *$\mathcal{E}$ satisfies the constraint C.*

Note that the empty set is admissible, however, it is not always C-admissible since $\widehat{\varnothing}$ does not always imply C.

Definition 7. ([9] – C-extensions) *Let $\mathcal{E} \subseteq \mathcal{A}$.*

- *$\mathcal{E}$ is a* C-preferred extension *iff $\mathcal{E}$ is maximal for set-inclusion among the C-admissible sets.*
- *$\mathcal{E}$ is a* C-stable extension *iff $\mathcal{E}$ is a C-preferred extension that attacks all arguments in $\mathcal{A}\backslash\mathcal{E}$.*

Now that the acceptability semantics are defined, we are ready to define the status of any argument.

Definition 8. (Argument status) *Let* CAF *be a constrained argumentation system, and $\mathcal{E}_1, \ldots, \mathcal{E}_x$ its extensions under a given semantics. Let $\alpha \in \mathcal{A}$.*

- α *is* accepted *iff* $\alpha \in \mathcal{E}_i$, $\forall \mathcal{E}_i$ *with* $i = 1, \ldots, x$.
- α *is* rejected *iff* $\nexists \mathcal{E}_i$ *such that* $\alpha \in \mathcal{E}_i$.
- α *is* undecided *iff* α *is neither accepted nor rejected.*

One can easily check that if an argument is rejected in the basic system AF, then it will also be rejected in CAF.

Property 1. Let $\alpha \in \mathcal{A}$. Under the stable or preferred semantics, if α is rejected in AF, then α is also rejected in CAF.

Proof. Let $\alpha \in \mathcal{A}$. Assume that α is rejected in AF, and that α is not rejected in CAF.

Case of Stable Semantics: Since α is not rejected in CAF, then there exists $\mathcal{E}_i$ that is a C-stable extension of CAF, and $\alpha \in \mathcal{E}_i$. In [9], it has been shown (Prop. 6) that every C-stable extension is also a stable extension. Consequently, $\mathcal{E}_i$ is also a stable extension. Since α is rejected in AF, then $\alpha \notin \mathcal{E}_i$, contradiction.

Case of Preferred Semantics: Since α is not rejected in CAF, then there exists $\mathcal{E}_i$ that is a C-preferred extension of CAF, and $\alpha \in \mathcal{E}_i$. In [9], it has been shown (Prop. 4) that each C-preferred extension is a subset of a preferred extension. This means that $\exists \mathcal{E}$ such $\mathcal{E}$ is a preferred extension of AF and $\mathcal{E}_i \subseteq \mathcal{E}$. However, since α is rejected in AF, then $\alpha \notin \mathcal{E}$, contradiction with the fact that $\alpha \in \mathcal{E}_i$.

3 Logical Language

This section presents the logical language that will be used throughout the paper. Let $\mathcal{L}$ be a *propositional language*, and $\equiv$ be the classical equivalence relation. From $\mathcal{L}$, a subset $\mathcal{D}$ is distinguished and is used for encoding *desires*. By desire we mean a state of affairs that an agent wants to reach. Elements of $\mathcal{D}$ are *literals*. We will write $d_1, \ldots, d_n$ to denote desires and the lowercase letters will denote formulas of $\mathcal{L}$.

From the above sets, *desire-generation* rules can be defined. A desire-generation rule expresses under which conditions an agent may adopt a given desire. A desire may come from beliefs. For instance, "if the weather is sunny, then I desire to go to the park". In this case, the desire of going to the park depends on my belief about the weather. A desire may also come from other desires. For example, if there is a conference in India, *and* I have the desire to attend, then I desire also to attend the tutorials. Finally, a desire may be unconditional, this means that it depends on neither beliefs nor desires. These three sources of desires are captured by the following desire-generation rules.

Definition 9. (Desire-Generation Rules) *A* desire-generation rule *(or a* desire rule*) is an expression of the form*

$$b \wedge d_1 \wedge \ldots \wedge d_{m-1} \hookrightarrow d_m, \text{ where}$$

b *is a propositional formula of* $\mathcal{L}$ *and* $\forall d_i$, $d_i \in \mathcal{D}$. *Moreover,* $\nexists d_i, d_j$ *with* $i, j \leq m$ *such that* $d_i \equiv d_j$. $b \wedge d_1 \wedge \ldots \wedge d_{m-1}$ *is called the* body *of the rule (this body may be empty; this is the case of an unconditional desire), and* d_m *is its* consequent.

The meaning of the rule is "if the agent *believes* b and *desires* $d_1, \ldots, d_{m-1}$, then the agent will *desire* d_m as well". Note that the same desire d_i may appear in the consequent of several rules. This means that the same desire may depend on different beliefs or desires. In what follows, a desire rule is consistent if it depends on consistent beliefs and on non contradictory desires.

Definition 10. (Consistent Desire Rule) *A desire rule* $b \wedge d_1 \wedge \ldots \wedge d_{m-1} \hookrightarrow d_m$ *is* consistent *iff* $b \nvdash \bot$, $\forall i = 1 \ldots m$, $b \nvdash \neg d_i$ *and* $\nexists d_i, d_j$ *with* $i, j \leq m$ *such that* $d_i \equiv \neg d_j$. *Otherwise, the rule is said* inconsistent.

An agent is assumed to be equipped with *plans* provided by a given planning system. The generation of such plans is beyond the scope of this paper. A plan is a way of achieving a desire. It is defined as a triple:

- a set of pre-conditions that should be satisfied before executing the plan,
- a set of post-conditions that hold after executing the plan, and
- the desire that is reached by the plan.

Definition 11. (Plan) *A* plan *is a triple* $\langle S, T, x \rangle$ *such that*

- S *and* T *are* consistent *sets of formulas of* $\mathcal{L}$,
- $x \in \mathcal{D}$,
- $T \vdash x$ *and* $S \nvdash x$.

Of course, there exists a link between S and T. But this link is not explicitly defined here because we are not interested by this aspect of the process. We just consider that the plan is given by a correct and sound planning system (for instance [11,17]).

In the remaining of the paper, we suppose that an agent is equipped with three *finite bases*:

- a base $\mathcal{K} \neq \varnothing$ and $\mathcal{K} \neq \{\bot\}$ containing its *basic beliefs* about the environment (elements of $\mathcal{K}$ are propositional formulas of the language $\mathcal{L}$),
- a base $\mathcal{B}_d$ containing its "consistent" desire rules,
- a base $\mathcal{P}$ containing its plans.

Using $\mathcal{B}_d$, we can characterize the *potential desires* of an agent as follows:

Definition 12. (Potential Desires) *The set of* potential desires *of an agent is* $\mathcal{PD} = \{d_m | \exists b \wedge d_1 \wedge \ldots \wedge d_{m-1} \hookrightarrow d_m \in \mathcal{B}_d\}$.

These are "potential" desires because it is not yet clear whether these desires are justified and feasible or not.

4 Typology of Arguments

The aim of this section is to present the different kinds of arguments involved in practical reasoning. There are mainly three categories of arguments: one category for supporting/attacking beliefs, and two categories for justifying the adoption of desires. Note that the arguments will be denoted with lowercase greek letters.

4.1 Justifying Beliefs

The first category of arguments is that studied in argumentation literature, especially for handling inconsistency in knowledge bases. Indeed, arguments are built from a knowledge base in order to support or to attack potential conclusions or inferences. These arguments are called *epistemic* in [12]. In our application, such arguments are built from the base $\mathcal{K}$. In what follows, we will use the definition proposed in [18].

Definition 13. (Epistemic Argument) *Let $\mathcal{K}$ be a knowledge base. An* epistemic argument *α is a pair $\alpha = \langle H, h\rangle$ s.t:*

1. *$H \subseteq \mathcal{K}$,*
2. *H is consistent,*
3. *$H \vdash h$ and*
4. *H is minimal (for set $\subseteq$) among the sets satisfying conditions 1, 2, 3.*

The support *of the argument is given by the function* $\texttt{SUPP}(\alpha) = H$*, whereas its* conclusion *is returned by* $\texttt{CONC}(\alpha) = h$*. $\mathcal{A}_b$ stands for the set of all epistemic arguments that can be built from the base $\mathcal{K}$.*

4.2 Justifying Desires

A desire may be pursued by an agent only if it is *justified* and *feasible*. Thus, there are two kinds of reasons for adopting a desire: i) the conditions underlying the desire hold in the current state of world; ii) there is a plan for reaching the desire. The definition of the first kind of arguments involves two bases: the belief base $\mathcal{K}$ and the base of desire rules $\mathcal{B}_d$. In what follows, we will use a tree-style definition of arguments [20]. Before presenting that definition, let us first introduce the functions $\texttt{BELIEFS}(\delta)$, $\texttt{DESIRES}(\delta)$, $\texttt{CONC}(\delta)$ and $\texttt{SUB}(\delta)$ that return respectively, for a given argument δ, the beliefs used in δ, the desires supported by δ, the conclusion and the set of sub-arguments of the argument δ.

Definition 14. (Explanatory Argument) *Let $\langle \mathcal{K}, \mathcal{B}_d\rangle$ be two bases.*

- *If $\exists \hookrightarrow d \in \mathcal{B}_d$ then $\longrightarrow d$ is an* explanatory argument *(δ) with* $\texttt{BELIEFS}(\delta) = \varnothing$, $\texttt{DESIRES}(\delta) = \{d\}$, $\texttt{CONC}(\delta) = d$, $\texttt{SUB}(\delta) = \{\delta\}$.
- *If α is an epistemic argument, and $\delta_1, \ldots, \delta_m$ are explanatory arguments, and $\exists\ \texttt{CONC}(\alpha) \wedge \texttt{CONC}(\delta_1) \wedge \ldots \wedge \texttt{CONC}(\delta_m) \hookrightarrow d \in \mathcal{B}_d$ then $\alpha, \delta_1, \ldots, \delta_m \longrightarrow d$ is an* explanatory argument *(δ) with* $\texttt{BELIEFS}(\delta) = \texttt{SUPP}(\alpha) \cup \texttt{BELIEFS}(\delta_1) \cup \ldots \cup \texttt{BELIEFS}(\delta_m)$, $\texttt{DESIRES}(\delta) = \texttt{DESIRES}(\delta_1) \cup \ldots \cup \texttt{DESIRES}(\delta_m) \cup \{d\}$, $\texttt{CONC}(\delta) = d$, $\texttt{SUB}(\delta) = \{\alpha\} \cup \texttt{SUB}(\delta_1) \cup \ldots \cup \texttt{SUB}(\delta_m) \cup \{\delta\}$.

$\mathcal{A}_d$ stands for the set of all explanatory arguments that can be built from $\langle \mathcal{K}, \mathcal{B}_d\rangle$ with a consistent DESIRES *set.*

One can easily show that the set BELIEFS of an explanatory argument is a subset of the knowledge base $\mathcal{K}$, and that the set DESIRES is a subset of $\mathcal{PD}$.

Property 2. Let $\delta \in \mathcal{A}_d$. The inclusions $\texttt{BELIEFS}(\delta) \subseteq \mathcal{K}$ and $\texttt{DESIRES}(\delta) \subseteq \mathcal{PD}$ hold.

Proof. Let $\delta \in \mathcal{A}_d$.
Let us show that $\texttt{BELIEFS}(\delta) \subseteq \mathcal{K}$. $\texttt{BELIEFS}(\delta) = \bigcup \texttt{SUPP}(\alpha_i)$ with $\alpha_i \in \mathcal{A}_b \cap \texttt{SUB}(\delta)$. According to the definition of an epistemic argument α_i, $\texttt{SUPP}(\alpha_i) \subseteq \mathcal{K}$, thus $\texttt{BELIEFS}(\delta) \subseteq \mathcal{K}$.

Let us show that $\texttt{DESIRES}(\delta) \subseteq \mathcal{PD}$. This is a direct consequence from the definition of an explanatory argument and the definition of the set $\mathcal{PD}$.

Note that a desire may be supported by several explanatory arguments since it may be the consequent of different desire rules.

The last category of arguments claims that "a desire may be pursued since it has a plan for achieving it". The definition of this kind of arguments involves the belief base $\mathcal{K}$ and the base of plans $\mathcal{P}$.

Definition 15. (Instrumental Argument) *Let $\langle \mathcal{K}, \mathcal{P} \rangle$ be two bases, and $d \in \mathcal{PD}$. An* instrumental *argument is a pair $\pi = \langle \langle S, T, x \rangle, d \rangle$ where*

1. $\langle S, T, x \rangle \in \mathcal{P}$,
2. $S \subseteq \mathcal{K}$,
3. $x \equiv d$.

$\mathcal{A}_p$ stands for the set of all instrumental arguments that can be built from $\langle \mathcal{K}, \mathcal{P}, \mathcal{PD} \rangle$. The function CONC *returns for an argument π the desire d. The function* Prec *returns the pre-conditions S of the plan, whereas* Postc *returns its post-conditions T.*

The second condition of the above definition says that the pre-conditions of the plan hold in the current state of the world. In other words, the plan can be executed. Indeed, the base $\mathcal{P}$ may contain plans whose pre-conditions are not true. Such plans cannot be executed and their corresponding instrumental arguments do not exist.

In what follows, $\mathcal{A} = \mathcal{A}_b \cup \mathcal{A}_d \cup \mathcal{A}_p$. Note that $\mathcal{A}$ is *finite* since the three initial bases ($\mathcal{K}$, $\mathcal{B}_d$ and $\mathcal{P}$) are finite.

5 Interactions among Arguments

An argument constitutes a reason for believing, or adopting a desire. However, it is not a proof that the belief is true, or that the desire can be adopted. The reason is that an argument can be attacked by other arguments. In [7,14], it has been shown that an argument may also be supported by other arguments. However, taking into account the supports between arguments in a practical reasoning context will be the subject of future work. In this section, we will investigate the different kinds of conflicts among the arguments identified in the previous section.

5.1 Conflicts among Epistemic Arguments

An argument can be attacked by another argument for three main reasons: i) they have contradictory conclusions (this is known as *rebuttal*), ii) the conclusion of an argument contradicts a premise of another argument (*assumption attack*), iii) the conclusion of an argument contradicts an inference rule used in order to build the other argument (*undercutting*). Since the base $\mathcal{K}$ is built around a propositional language, it has been shown in [2] that the notion of assumption attack is sufficient to capture conflicts between epistemic arguments.

Definition 16. *Let* $\alpha_1, \alpha_2 \in \mathcal{A}_b$. $\alpha_1 \mathcal{R}_b \alpha_2$ *iff* $\exists h' \in \texttt{SUPP}(\alpha_2)$ *s.t.* $\texttt{CONC}(\alpha_1) \equiv \neg h'$.

Note that the relation $\mathcal{R}_b$ *is not symmetric*. Moreover, one can show that there are no self-defeating arguments.

In [6], the argumentation system $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$ has been applied for handling inconsistency in a knowledge base, say $\mathcal{K}$. In this particular case, a full correspondence has been established between the stable extensions of the system and the maximal consistent subsets of the base $\mathcal{K}$. Before presenting formally the result, let us introduce some useful notations:

- Let $\mathcal{E} \subseteq \mathcal{A}_b$, $\texttt{Base}(\mathcal{E}) = \bigcup H_i$ such that $\langle H_i, h_i \rangle \in \mathcal{E}$.
- Let $T \subseteq \mathcal{K}$, $\texttt{Arg}(T) = \{\langle H_i, h_i \rangle | H_i \subseteq T\}$.

Property 3 ([6]). Let $\mathcal{E}$ be a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$.
$\texttt{Base}(\mathcal{E})$ is a maximal (for set $\subseteq$) consistent subset of $\mathcal{K}$ and $\texttt{Arg}(\texttt{Base}(\mathcal{E})) = \mathcal{E}$.

Property 4 ([6]). Let T be a maximal (for set $\subseteq$) consistent subset of $\mathcal{K}$.
$\texttt{Arg}(T)$ is a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$ and $\texttt{Base}(\texttt{Arg}(T)) = T$.

A direct consequence of the above result is that if the base $\mathcal{K}$ is not reduced to $\bot$, then the system $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$ has at least one non-empty stable extension.

Property 5. The argumentation system $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$ has non-empty stable extensions.

Proof. Since $\mathcal{K} \neq \{\bot\}$ and $\mathcal{K} \neq \varnothing$ then the base $\mathcal{K}$ has at least one maximal (for set inclusion) consistent subset, say T. According to Property 4, $\texttt{Arg}(T)$ is a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$.

5.2 Conflicts among Explanatory Arguments

Explanatory arguments may also be conflicting. Indeed, two explanatory arguments may be based on two contradictory desires.

Definition 17. *Let* $\delta_1, \delta_2 \in \mathcal{A}_d$. $\delta_1 \mathcal{R}_d \delta_2$ *iff* $\exists d_1 \in \texttt{DESIRES}(\delta_1)$, $d_2 \in \texttt{DESIRES}(\delta_2)$ *such that* $d_1 \equiv \neg d_2$.

Property 6. The relation $\mathcal{R}_d$ is symmetric and irreflexive.

Proof. The proof follows directly from the definition of the attack relation $\mathcal{R}_d$.

Note that from the definition of an explanatory argument, its set DESIRES cannot be inconsistent. However, its set BELIEFS may be inconsistent. The union of the beliefs sets of two explanatory arguments may also be inconsistent. Later in the paper, we will show that it is useless to explicit these kinds of conflicts, since they are captured by conflicts between explanatory arguments and epistemic ones (see Property 9 and Property 10).

5.3 Conflicts among Instrumental Arguments

Two plans may be conflicting for four main reasons:

- their pre-conditions are incompatible (*i.e.* the two plans cannot be executed at the same time),

- their post-conditions are incompatible (the execution of the two plans will lead to contradictory states of the world),
- the post-conditions of a plan and the preconditions of the other are incompatible (*i.e.* the execution of a plan will prevent the execution of the second plan in the future),
- their supporting desires are incompatible (indeed, plans for achieving contradictory desires are conflicting; their execution will in fact lead to a contradictory state of the world).

The above reasons are captured in the following definition of attack among instrumental arguments. Note that a plan cannot be incompatible with itself.

Definition 18. *Let* $\pi_1, \pi_2 \in \mathcal{A}_p$ *with* $\pi_1 \neq \pi_2$. $\pi_1 \mathcal{R}_p \pi_2$ *iff:*

- $\texttt{Prec}(\pi_1) \wedge \texttt{Prec}(\pi_2) \models \bot$, *or*
- $\texttt{Postc}(\pi_1) \wedge \texttt{Postc}(\pi_2) \models \bot$, *or*
- $\texttt{Postc}(\pi_1) \wedge \texttt{Prec}(\pi_2) \models \bot$ *or* $\texttt{Prec}(\pi_1) \wedge \texttt{Postc}(\pi_2) \models \bot$.

Property 7. The relation $\mathcal{R}_p$ is symmetric and irreflexive.

Proof. The proof follows directly from the definition of the attack relation $\mathcal{R}_p$.

One can show that if two plans realize conflicting desires, then their corresponding instrumental arguments are conflicting too.

Property 8. Let $d_1, d_2 \in \mathcal{PD}$. If $d_1 \equiv \neg d_2$, then $\forall \pi_1, \pi_2 \in \mathcal{A}_p$ s.t. $\texttt{CONC}(\pi_1) = d_1$ and $\texttt{CONC}(\pi_2) = d_2$, then $\pi_1 \mathcal{R}_p \pi_2$.

Proof. Let $d_1, d_2 \in \mathcal{PD}$. Suppose that $d_1 \equiv \neg d_2$. Let us also suppose that $\exists\, \pi_1, \pi_2 \in \mathcal{A}_p$ with $\texttt{CONC}(\pi_1) = d_1$, and $\texttt{CONC}(\pi_2) = d_2$. According to Definition 15, it holds that $\texttt{Postc}(\pi_1) \vdash d_1$ and $\texttt{Postc}(\pi_2) \vdash d_2$. Since $d_1 \equiv \neg d_2$, then $\texttt{Postc}(\pi_2) \vdash \neg d_1$. However, the two sets $\texttt{Postc}(\pi_1)$ and $\texttt{Postc}(\pi_2)$ are both consistent (according to Definition 11), thus $\texttt{Postc}(\pi_1) \cup \texttt{Postc}(\pi_2) \vdash \bot$. Thus, $\pi_1 \mathcal{R}_p \pi_2$.

In this section, we have considered *only binary conflicts* between plans, and consequently between their corresponding instrumental arguments. However, in every-day life, one may have for instance three plans such that any pair of them is not conflicting, but the three together are incompatible. For simplicity reasons, in this paper we suppose that we do not have such conflicts.

5.4 Conflicts among Mixed Arguments

In the previous sections we have shown how arguments of the same category can interact with each other. In this section, we will show that arguments of different categories can also interact. Indeed, epistemic arguments play a key role in ensuring the acceptability of explanatory or instrumental arguments. Namely, an epistemic argument can attack both types of arguments. The idea is to invalidate any belief used in an explanatory or instrumental argument. An explanatory argument may also conflict with an instrumental argument when this last achieves a desire whose negation is among the desires of the explanatory argument.

Definition 19. *Let* $\alpha \in \mathcal{A}_b$, $\delta \in \mathcal{A}_d$, $\pi \in \mathcal{A}_p$.

- $\alpha \mathcal{R}_{bd} \delta$ *iff* $\exists h \in \texttt{BELIEFS}(\delta)$ *s.t.* $h \equiv \neg\texttt{CONC}(\alpha)$.
- $\alpha \mathcal{R}_{bp} \pi$ *iff* $\exists h \in \texttt{Prec}(\pi)$, *s.t.* $h \equiv \neg\texttt{CONC}(\alpha)$.
- $\delta \mathcal{R}_{pdp} \pi$ *and* $\pi \mathcal{R}_{pdp} \delta$ *iff* $\texttt{CONC}(\pi) \equiv \neg d$ *and* $d \in \texttt{DESIRES}(\delta)$[1].

As already said, the set of beliefs of an explanatory argument may be inconsistent. In such a case, the explanatory argument is attacked (in the sense of $\mathcal{R}_{bd}$) for sure by an epistemic argument.

Property 9. Let $\delta \in \mathcal{A}_d$. If $\texttt{BELIEFS}(\delta) \vdash \bot$, then $\exists \alpha \in \mathcal{A}_b$ such that $\alpha \mathcal{R}_{bd} \delta$.

Proof. Let $\delta \in \mathcal{A}_d$. Suppose that $\texttt{BELIEFS}(\delta) \vdash \bot$. This means that $\exists T$ that is minimal for set inclusion among subsets of $\texttt{BELIEFS}(\delta)$ with $T \vdash \bot$. Thus[2], $\exists h \in T$ such that $T \backslash \{h\} \vdash \neg h$ with $T \backslash \{h\}$ is consistent. Since $\texttt{BELIEFS}(\delta) \subseteq \mathcal{K}$ (according to Property 2), then $T \backslash \{h\} \subseteq \mathcal{K}$. Consequently, $\exists \langle T \backslash \{h\}, \neg h \rangle \in \mathcal{A}_b$ with $h \in \texttt{BELIEFS}(\delta)$. Thus, $\langle T \backslash \{h\}, \neg h \rangle \mathcal{R}_{bd} \delta$.

Similarly, when the beliefs of two explanatory arguments are inconsistent, it can be checked that there exists an epistemic argument that attacks at least one of the two explanatory arguments.

Property 10. Let $\delta_1, \delta_2 \in \mathcal{A}_d$ respecting $\texttt{BELIEFS}(\delta_1) \nvdash \bot$ and $\texttt{BELIEFS}(\delta_2) \nvdash \bot$. If $\texttt{BELIEFS}(\delta_1) \cup \texttt{BELIEFS}(\delta_2) \vdash \bot$, then $\exists \alpha \in \mathcal{A}_b$ such that $\alpha \mathcal{R}_{bd} \delta_1$, or $\alpha \mathcal{R}_{bd} \delta_2$.

Proof. Let $\delta_1, \delta_2 \in \mathcal{A}_d$ with $\texttt{BELIEFS}(\delta_1) \nvdash \bot$ and $\texttt{BELIEFS}(\delta_2) \nvdash \bot$.
Suppose that $\texttt{BELIEFS}(\delta_1) \cup \texttt{BELIEFS}(\delta_2) \vdash \bot$. So, $\exists T_1 \subseteq \texttt{BELIEFS}(\delta_1)$ and $\exists T_2 \subseteq \texttt{BELIEFS}(\delta_2)$ with $T_1 \cup T_2 \vdash \bot$ and $T_1 \cup T_2$ is minimal for set inclusion, *i.e.* $T_1 \cup T_2$ is a minimal conflict. Since $\texttt{BELIEFS}(\delta_1) \nvdash \bot$ and $\texttt{BELIEFS}(\delta_2) \nvdash \bot$, then $T_1 \neq \varnothing$ and $T_2 \neq \varnothing$. Thus, $\exists h \in T_1 \cup T_2$ such that $(T_1 \cup T_2) \setminus \{h\} \vdash \neg h$. Since $T_1 \cup T_2$ is a minimal conflict, then each subset of $T_1 \cup T_2$ is consistent, thus the set $(T_1 \cup T_2) \setminus \{h\}$ is consistent. Moreover, according to Property 2, $\texttt{BELIEFS}(\delta_1) \subseteq \mathcal{K}$ and $\texttt{BELIEFS}(\delta_2) \subseteq \mathcal{K}$. Thus, $T_1 \subseteq \mathcal{K}$ and $T_2 \subseteq \mathcal{K}$. It is then clear that $(T_1 \cup T_2) \setminus \{h\} \subseteq \mathcal{K}$. Consequently $\langle (T_1 \cup T_2) \setminus \{h\}, \neg h \rangle$ is an argument of $\mathcal{A}_b$.

If $h \in T_1$ then $\langle (T_1 \cup T_2) \setminus \{h\}, \neg h \rangle \mathcal{R}_{bd} \delta_1$, and if $h \in T_2$ then $\langle (T_1 \cup T_2) \setminus \{h\}, \neg h \rangle \mathcal{R}_{bd} \delta_2$.

Conflicts may also exist between an instrumental argument and an explanatory one since the beliefs of the explanatory argument may be conflicting with the preconditions of the instrumental one. Here again, we'll show that there exists an epistemic argument that attacks at least one of the two arguments.

Property 11. Let $\pi \in \mathcal{A}_p$ and $\delta \in \mathcal{A}_d$ with $\texttt{BELIEFS}(\delta) \nvdash \bot$. If $\texttt{BELIEFS}(\delta) \cup \texttt{Prec}(\pi) \vdash \bot$ then $\exists \alpha \in \mathcal{A}_b$ such that $\alpha \mathcal{R}_{bd} \delta$, or $\alpha \mathcal{R}_{bp} \pi$.

[1] Note that if $\delta_1 \mathcal{R}_{pdp} \pi_2$ and there exists δ_2 such that $\texttt{CONC}(\delta_2) = \texttt{CONC}(\pi_2)$ then $\delta_1 \mathcal{R}_d \delta_2$.

[2] Since T is $\subseteq$-minimal among inconsistent subsets of $\texttt{BELIEFS}(\delta)$, then each subset of T is consistent.

Proof. Let $\delta \in \mathcal{A}_d$ and $\pi \in \mathcal{A}_p$. Suppose that $\texttt{BELIEFS}(\delta) \nvdash \bot$. Since $\texttt{BELIEFS}(\delta) \nvdash \bot$ and $\texttt{Prec}(\pi) \nvdash \bot$, then $\exists T \subseteq \texttt{BELIEFS}(\delta) \cup \texttt{Prec}(\pi)$ with $\texttt{BELIEFS}(\delta) \cap T \neq \varnothing$, $\texttt{Prec}(\pi) \cap T \neq \varnothing$ and T is the smallest inconsistent subset of $\texttt{BELIEFS}(\delta) \cup \texttt{Prec}(\pi)$.

Since $T \vdash \bot$, then $\exists h \in T$ such that $T\backslash\{h\} \vdash \neg h$ with $T\backslash\{h\}$ is consistent. Since $\texttt{BELIEFS}(\delta) \subseteq \mathcal{K}$ and since $\texttt{Prec}(\pi) \subseteq \mathcal{K}$, then $T \subseteq \mathcal{K}$. Consequently, $T\backslash\{h\} \subseteq \mathcal{K}$. Thus, $\langle T\backslash\{h\}, \neg h\rangle \in \mathcal{A}_b$.

If $h \in \texttt{BELIEFS}(\delta)$, then $\langle T\backslash\{h\}, \neg h\rangle \ \mathcal{R}_{bd}\ \delta$. If $h \in \texttt{Prec}(\pi)$, then $\langle T\backslash\{h\}, \neg h\rangle \ \mathcal{R}_{bp}\ \pi$.

Later in the paper, it will be shown that the three above propositions are sufficient for ignoring these conflicts (between two explanatory arguments, and between an explanatory argument and an instrumental one). Note also that explanatory arguments and instrumental arguments are not allowed to attack epistemic arguments. In fact, a desire cannot invalidate a belief. Let us illustrate this issue by an example borrowed from [19]. An agent thinks that it will be raining, and that when it is raining, she gets wet. It is clear that this agent does not desire to be wet when it is raining. Intuitively, we should get one extension $\{rain, wet\}$. The idea is that if the agent believes that it is raining, and she will get wet if it rains, then she should believe that she will get wet, regardless of her likings. To do otherwise would be to indulge in *wishful thinking*.

6 Argumentation System for PR

The notion of constraint which forms the backbone of constrained argumentation systems allows, in the context of PR, the representation of the link between the justification of a desire and the plan for achieving it (so between the explanatory argument in favor of a given desire and the instrumental arguments in favor of that desire). A constrained argumentation system for PR is defined as follows:

Definition 20. (Constrained argumentation system for PR) *The constrained argumentation system for practical reasoning is the triple* $\mathsf{CAF}_{\mathsf{PR}} = \langle \mathcal{A}, \mathcal{R}, C\rangle$ *with:*

- $\mathcal{A} = \mathcal{A}_b \cup \mathcal{A}_d \cup \mathcal{A}_p$,
- $\mathcal{R} = \mathcal{R}_b \cup \mathcal{R}_d \cup \mathcal{R}_p \cup \mathcal{R}_{bd} \cup \mathcal{R}_{bp} \cup \mathcal{R}_{pdp}$
- *and* C *a constraint on arguments defined on* $\mathcal{A}$ *respecting* $C = \wedge_i(\pi_i \Rightarrow (\vee_j \delta_j))$ *for each* $\pi_i \in \mathcal{A}_p$ *and* $\delta_j \in \mathcal{A}_d$ *such that* $\texttt{CONC}(\pi_i) \equiv \texttt{CONC}(\delta_j)$.

Note that the satisfaction of the constraint C implies that each plan of a desire is taken into account only if this desire is justified. Note also that we consider that there may be several plans for one desire but only one desire for each plan. Nevertheless, for each desire there may exist several explanatory arguments.

An important remark concerns the notion of defence. This notion has two different semantics in a PR context. When we consider only epistemic or explanatory arguments, the defence corresponds exactly to the notion defined in Dung's argumentation systems and in its constrained extension: an argument α attacks the attacker of another argument β; so α "reinstates" β; without the defence, β cannot be kept in an admissible set. Things are different with instrumental arguments: when an instrumental argument

attacks another argument, this attack is always symmetric (so, each argument defends itself against an instrumental argument). In this case, it would be sufficient to take into account the notion of conflict-free in order to identify the plans which belong to an admissible set. However, in order to keep an homogeneous definition of admissibility, the notion of defence is also used for instrumental arguments knowing that it is without impact when conflicts from an instrumental argument are concerned.

Note that $\varnothing$ is always a C-admissible set of $\mathsf{CAF}_{\mathsf{PR}}$. The reason is that $\varnothing$ is admissible (as shown by Dung in [10]) and that all π_i variables are false in $\widehat{\varnothing}$, so $\widehat{\varnothing} \vdash C$[3]. Thus, $\mathsf{CAF}_{\mathsf{PR}}$ has at least one C-preferred extension. Moreover, the extensions do not contain the "good" plans of non-justified desires. Thus, the use of the constraint makes it possible to filter the content of the extensions and to keep only useful information.

At some places of the paper, we will refer by $\mathsf{AF}_{\mathsf{PR}} = \langle \mathcal{A}, \mathcal{R} \rangle$ to a basic argumentation system for PR, *i.e.* an argumentation system without the constraint, and $\mathcal{A}$ and $\mathcal{R}$ are defined as in Definition 20.

Remember that the purpose of a practical reasoning problem is to compute the intentions to be pursued by an agent, *i.e.* the desires that are both justified and feasible.

Definition 21 (Set of intentions). *Let $\mathcal{I} \subseteq \mathcal{PD}$. $\mathcal{I}$ is a* set of intentions *iff there exists a C-extension $\mathcal{E}$ (under a given semantics) of $\mathsf{CAF}_{\mathsf{PR}}$ such that for each $d \in \mathcal{I}$, there exists $\pi \in \mathcal{A}_p \cap \mathcal{E}$ such that $d = \mathtt{CONC}(\pi)$.*

Our system provides an interesting solution to the PR problem. It computes directly sets of intentions, and identifies the state of the world as well as the plans necessary for achieving these intentions.

7 Properties of the System

The aim of this section is to study the properties of the proposed argumentation system for PR. Since the proposed $\mathsf{CAF}_{\mathsf{PR}}$ is grounded on the abstract constrained argumentation system defined in [9], it is natural that it inherits the results got in [9]. However, the following result, whose proof is obvious, holds in the context of PR but not in the general case.

Property 12. Let $\mathsf{CAF}_{\mathsf{PR}} = \langle \mathcal{A}, \mathcal{R}, C \rangle$. The set Ω of C-admissible sets defines a complete partial order for $\subseteq$.

An important property shows that the set of epistemic arguments in a given stable extension of $\mathsf{AF}_{\mathsf{PR}}$ is itself a stable extension of the system $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$. This shows clearly that stable extensions are "complete" w.r.t. epistemic arguments.

Property 13. If $\mathcal{E}$ is a stable extension of $\mathsf{AF}_{\mathsf{PR}}$, then the set $\mathcal{E} \cap \mathcal{A}_b$ is a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$.

Proof. Let $\mathcal{E}$ be a stable extension of $\mathsf{AF}_{\mathsf{PR}}$. Let us suppose that $\mathcal{E}' = \mathcal{E} \cap \mathcal{A}_b$ is not a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$. Two cases exist:

[3] This is due to the particular form of the constraint for PR. This is not true for any constraints (see Section 2 and [9]).

Case 1: $\mathcal{E}'$ is not conflict-free. This means that there exist $\alpha, \alpha' \in \mathcal{E}'$ such that $\alpha \mathcal{R}_b \alpha'$. Since $\mathcal{E}' = \mathcal{E} \cap \mathcal{A}_b$, then $\alpha, \alpha' \in \mathcal{E}$. This means that $\mathcal{E}$ is not conflict-free. This contradicts the fact that $\mathcal{E}$ is a stable extension.

Case 2: $\mathcal{E}'$ does not attack every argument that is not in $\mathcal{E}'$. This means that $\exists \alpha \in \mathcal{A}_b$ and $\notin \mathcal{E}'$ and $\mathcal{E}'$ does not attack (w.r.t. $\mathcal{R}_b$) α. This means that $\mathcal{E}' \cup \{\alpha\}$ is conflict-free, thus $\mathcal{E} \cup \{\alpha\}$ is also conflict-free, and does not attack an argument that is not in it (because only an epistemic argument can attack another epistemic argument and all epistemic arguments of $\mathcal{E}$ belong to $\mathcal{E}'$). This contradicts the fact that $\mathcal{E}$ is a stable extension.

Another important property of $\mathsf{AF}_{\mathsf{PR}}$ is that it has stable extensions.

Property 14. The system $\mathsf{AF}_{\mathsf{PR}}$ has at least one non-empty stable extension.

Proof. (Sketch) $\mathsf{AF}_{\mathsf{PR}}$ can be viewed as the union of 2 argumentation systems: $\mathsf{AF}_b = \langle \mathcal{A}_b, \mathcal{R}_b \rangle$ and $\mathsf{AF}_{dp} = \langle \mathcal{A}_d \cup \mathcal{A}_p, \mathcal{R}_d \cup \mathcal{R}_p \cup \mathcal{R}_{pdp} \rangle$ plus the $\mathcal{R}_{bd} \cup \mathcal{R}_{bp}$ relation. The system AF_b has stable extensions (according to Prop. 5). Let $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be those extensions. The system AF_{dp} is symmetric in the sense of [8] since the relation $\mathcal{R}_d \cup \mathcal{R}_p \cup \mathcal{R}_{pdp}$ is symmetric. In [8], it has been shown that such a system has stable extensions which correspond to maximal (for $\subseteq$) sets of arguments that are conflict-free. Let $\mathcal{E}'_1, \ldots, \mathcal{E}'_m$ be those extensions.

These two systems are linked with the $\mathcal{R}_{bd} \cup \mathcal{R}_{bp}$ relation. Two cases can be distinguished:

- **Case1:** $\mathcal{R}_{bd} \cup \mathcal{R}_{bp} = \varnothing$. $\forall \mathcal{E}_i, \mathcal{E}'_j$, the set $\mathcal{E}_i \cup \mathcal{E}'_j$ is a stable extension of $\mathsf{AF}_{\mathsf{PR}}$. Indeed, $\mathcal{E}_i \cup \mathcal{E}'_j$ is conflict-free since $\mathcal{E}_i$, $\mathcal{E}'_j$ are both conflict-free, and the relation $\mathcal{R}_{bd} \cup \mathcal{R}_{bp} = \varnothing$. Moreover, $\mathcal{E}_i \cup \mathcal{E}'_j$ defeats every argument that is not in $\mathcal{E}_i \cup \mathcal{E}'_j$, since if $\alpha \notin \mathcal{E}_i \cup \mathcal{E}'_j$, then: i) if $\alpha \in \mathcal{A}_b$, then $\mathcal{E}_i$ defeats w.r.t. $\mathcal{R}_b$ α since $\mathcal{E}_i$ is a stable extension. Now, assume that $\alpha \in \mathcal{A}_d \cup \mathcal{A}_p$. Then, $\mathcal{E}'_j \cup \{\alpha\}$ is conflicting since $\mathcal{E}'_j$ is a maximal (for $\subseteq$) set that is conflict-free. Thus, $\mathcal{E}'_j$ defeats α.
- **Case2:** $\mathcal{R}_{bd} \cup \mathcal{R}_{bp} \neq \varnothing$. Let $\mathcal{E}$ be a maximal (for set inclusion) set of arguments that is built with the following algorithm:
  1. $\mathcal{E} = \mathcal{E}_i$
  2. while ($\exists \beta \in \mathcal{A}_p \cup \mathcal{A}_d$ such that $\mathcal{E} \cup \{\beta\}$ is conflict-free) do $\mathcal{E} = \mathcal{E} \cup \{\beta\}$

  This algorithm stops after a finite number of steps (because $\mathcal{A}_p \cup \mathcal{A}_d$ is a finite set) and gives a set of arguments which is $\subseteq$-maximal among the conflict-free sets which include $\mathcal{E}_i$. It is easy to see that $\mathcal{E}$ is stable because, by construction, $\forall \gamma \in (\mathcal{A}_p \cup \mathcal{A}_d) \setminus \mathcal{E}$, $\exists \gamma' \in \mathcal{E}$ such that $\gamma' \mathcal{R} \gamma$, and, because $\mathcal{E}_i \subseteq \mathcal{E}$, we also have $\forall \alpha \in \mathcal{A}_b \setminus \mathcal{E}$, $\exists \alpha' \in \mathcal{E}$ such that $\alpha' \mathcal{R} \alpha$.

So, the system $\mathsf{AF}_{\mathsf{PR}}$ has a stable extension.

Let us now come back to the three critical cases of conflicts among arguments that are not explicitly captured by the six attack relations defined in Section 5. The first case concerns explanatory arguments whose sets of beliefs are inconsistent. It can be checked that such arguments are rejected in the system $\mathsf{CAF}_{\mathsf{PR}}$.

Property 15. Let $\delta \in \mathcal{A}_d$ with $\texttt{BELIEFS}(\delta) \vdash \bot$. Under the stable semantics, the argument δ is rejected in $\mathsf{CAF}_{\mathsf{PR}}$.

Proof. (Sketch) Let $\delta \in \mathcal{A}_d$ with $\texttt{BELIEFS}(\delta) \vdash \bot$. According to Property 14, the system $\mathsf{AF}_{\mathsf{PR}}$ has at least one stable extension. Let $\mathcal{E}$ be one of these stable extensions. Suppose that $\delta \in \mathcal{E}$. According to Property 13, the set $\mathcal{E} \cap \mathcal{A}_b$ is a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$. Moreover, we can show that $\exists \alpha \in \mathcal{E} \cap \mathcal{A}_b$ such that $\alpha \mathcal{R}_{bd} \delta$. This contradicts the fact that a stable extension is conflict-free. Thus, δ is rejected in $\mathsf{AF}_{\mathsf{PR}}$. According to Prop. 1, δ is also rejected in $\mathsf{CAF}_{\mathsf{PR}}$.

Similarly, it can be checked that if two explanatory arguments have conflicting beliefs, then they will never belong to the same stable extension at the same time.

Property 16. Let $\delta_1, \delta_2 \in \mathcal{A}_d$ respecting $\texttt{BELIEFS}(\delta_1) \nvdash \bot$ and $\texttt{BELIEFS}(\delta_2) \nvdash \bot$. If $\texttt{BELIEFS}(\delta_1) \cup \texttt{BELIEFS}(\delta_2) \vdash \bot$, then $\nexists \mathcal{E}$ C-stable extension of $\mathsf{CAF}_{\mathsf{PR}}$ such that $\delta_1 \in \mathcal{E}$ and $\delta_2 \in \mathcal{E}$.

Proof. (Sketch) Let $\delta_1, \delta_2 \in \mathcal{A}_d$ s.t. $BELIEFS(\delta_1) \nvdash \bot$, $\texttt{BELIEFS}(\delta_2) \nvdash \bot$, and $\texttt{BELIEFS}(\delta_1) \cup \texttt{BELIEFS}(\delta_2) \vdash \bot$. Let $\mathcal{E}$ be a C-stable extension of $\mathsf{CAF}_{\mathsf{PR}}$. Thus, $\mathcal{E}$ is also a stable extension of $\mathsf{AF}_{\mathsf{PR}}$. Suppose that $\delta_1 \in \mathcal{E}$ and $\delta_2 \in \mathcal{E}$. According to Property 13, the set $\mathcal{E} \cap \mathcal{A}_b$ is a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$. Moreover, we can easily show that $\exists \alpha \in \mathcal{E} \cap \mathcal{A}_b$ such that $\alpha \mathcal{R}_{bd} \delta_1$, or $\alpha \mathcal{R}_{bd} \delta_2$. This contradicts the fact that a stable extension is conflict-free.

Similarly, if the beliefs of an explanatory argument and an instrumental one are conflicting, the two arguments will not appear in the same stable extension.

Property 17. Let $\delta \in \mathcal{A}_d$ and $\pi \in \mathcal{A}_p$ with $\texttt{BELIEFS}(\delta) \nvdash \bot$. If $\texttt{BELIEFS}(\delta) \cup \texttt{Prec}(\pi) \vdash \bot$ then $\nexists \mathcal{E}$ with $\mathcal{E}$ is a C-stable extension of $\mathsf{CAF}_{\mathsf{PR}}$ such that $\delta \in \mathcal{E}$ and $\pi \in \mathcal{E}$.

Proof. (Sketch) Let $\delta \in \mathcal{A}_d$ and $\pi \in \mathcal{A}_p$ with $\texttt{BELIEFS}(\delta) \nvdash \bot$ and $\texttt{BELIEFS}(\delta) \cup \texttt{Prec}(\pi) \vdash \bot$. Let $\mathcal{E}$ be a C-stable extension of $\mathsf{CAF}_{\mathsf{PR}}$. Thus, $\mathcal{E}$ is also a stable extension of $\mathsf{AF}_{\mathsf{PR}}$. Let us assume that $\delta \in \mathcal{E}$ and $\pi \in \mathcal{E}$. Since $\mathcal{E}$ is a stable extension of $\mathsf{AF}_{\mathsf{PR}}$, then $\mathcal{E}' = \mathcal{E} \cap \mathcal{A}_b$ is a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$ (according to Property 13). Moreover, it can be checked that when $\texttt{BELIEFS}(\delta) \cup \texttt{Prec}(\pi) \vdash \bot$ then $\exists \alpha \in \mathcal{E}'$ s.t. $\alpha \mathcal{R}_{bd} \delta$ or $\alpha \mathcal{R}_{bp} \pi$. This means that $\mathcal{E}$ attacks δ or $\mathcal{E}$ attacks π. However, $\delta \in \mathcal{E}$ and $\pi \in \mathcal{E}$. This contradicts the fact that $\mathcal{E}$ is conflict free.

The next results are of great importance. They show that the proposed argumentation system for PR satisfies the "consistency" rationality postulate identified in [5]. Indeed, we show that each stable extension of our system supports a consistent set of desires and a consistent set of beliefs. Let $\mathcal{E} \subseteq \mathcal{A}$, the following notations are defined: $\texttt{Bel}(\mathcal{E}) = (\bigcup_{\alpha_i \in \mathcal{E} \cap \mathcal{A}_b} \texttt{SUPP}(\alpha_i)) \cup (\bigcup_{\delta_j \in \mathcal{E} \cap \mathcal{A}_d} \texttt{BELIEFS}(\delta_j)) \cup (\bigcup_{\pi_k \in \mathcal{E} \cap \mathcal{A}_p} \texttt{Prec}(\pi_k))$ and $\texttt{Des}(\mathcal{E}) = (\bigcup_{\delta_j \in \mathcal{E} \cap \mathcal{A}_d} \texttt{DESIRES}(\delta_j)) \cup (\bigcup_{\pi_k \in \mathcal{E} \cap \mathcal{A}_p} \texttt{CONC}(\pi_k))$.

Theorem 1. *(Consistency) Let $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be the C-stable extensions of $\mathsf{CAF}_{\mathsf{PR}}$. $\forall \mathcal{E}_i$, $i = 1, \ldots, n$, it holds that:*

- $\texttt{Bel}(\mathcal{E}_i) = \texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$,

- $\texttt{Bel}(\mathcal{E}_i)$ *is a* $\subseteq$*-maximal consistent subset of* $\mathcal{K}$ *and*
- $\texttt{Des}(\mathcal{E}_i)$ *is consistent.*

Proof. Let $\mathcal{E}$ be a C-stable extension of $\mathsf{CAF}_{\mathsf{PR}}$. Thus, $\mathcal{E}$ is also a stable extension of $\mathsf{AF}_{\mathsf{PR}}$.

1. Let us show that the set $\texttt{Bel}(\mathcal{E}_i) = \texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$. In order to prove this, one should handle two cases:

1.1. $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b) \subseteq \texttt{Bel}(\mathcal{E}_i)$. This is implied by $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b) = \bigcup \texttt{SUPP}(\alpha_i)$ with $\alpha_i \in \mathcal{E}_i \cap \mathcal{A}_b$ (cf. definition of $\texttt{Bel}(\mathcal{E})$).

1.2. $\texttt{Bel}(\mathcal{E}_i) \subseteq \texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$. Let us suppose that $\exists h \in \texttt{Bel}(\mathcal{E}_i)$ and $h \notin \texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$. According to Property 13, $\mathcal{E}_i \cap \mathcal{A}_b$ is a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$. Moreover, according to [6], $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$ is a maximal (for set-$\subseteq$) consistent subset of $\mathcal{K}$[4]. However, $\texttt{Bel}(\mathcal{E}_i) \subseteq \mathcal{K}$, then $h \in \mathcal{K}$. Since $h \notin \texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$, then $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b) \cup \{h\} \vdash \bot$ (this is due to the fact that $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$ is a maximal (for set-$\subseteq$) consistent subset of $\mathcal{K}$). Thus, $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b) \vdash \neg h$. This means that $\exists H \subseteq \texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$ such that H is the minimal consistent subset of $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$, thus $H \vdash \neg h$. Since $H \subseteq \mathcal{K}$ (since $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b) \subseteq \mathcal{K}$), then $\langle H, \neg h \rangle \in \mathcal{A}_b$. However, according to [6], $\texttt{Arg}(\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)) = \mathcal{E}_i \cap \mathcal{A}_b$. Besides, $h \in \texttt{Bel}(\mathcal{E}_i)$, there are three possibilities:

- $h \in \texttt{BELIEFS}(\delta)$ with $\delta \in \mathcal{E}_i$. In this case, $\langle H, \neg h \rangle \ \mathcal{R}_{bd}\ \delta$. This contradicts the fact that $\mathcal{E}_i$ is a stable extension that is conflict-free.
- $h \in \texttt{Prec}(\pi)$ with $\pi \in \mathcal{E}_i$. In this case, $\langle H, \neg h \rangle \ \mathcal{R}_{bp}\ \pi$. This contradicts the fact that $\mathcal{E}_i$ is a stable extension that is conflict-free.
- $h \in \texttt{SUPP}(\alpha)$ with $\alpha \in \mathcal{E}_i$. This is impossible since the set $\mathcal{E}_i \cap \mathcal{A}_b$ is a stable extension, thus it is conflict free.

2. Let us show that the set $\texttt{Bel}(\mathcal{E}_i)$ is a maximal (for set inclusion) consistent subset of $\mathcal{K}$. According to the first item of Theorem 1, $\texttt{Bel}(\mathcal{E}_i) = \texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$. However, according to Property 13, $\mathcal{E}_i \cap \mathcal{A}_b$ is a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$, and according to [6], $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$ is a maximal (for set-$\subseteq$) consistent subset of $\mathcal{K}$. Thus, $\texttt{Bel}(\mathcal{E}_i)$ is a maximal (for set inclusion) consistent subset of $\mathcal{K}$.

3. Let us show that the set $\texttt{Des}(\mathcal{E}_i)$ is consistent. Let us suppose that $\texttt{Des}(\mathcal{E}_i)$ is inconsistent, this means that $\bigcup \texttt{DESIRES}(\delta_k) \cup \bigcup \texttt{CONC}(\pi_j) \vdash \bot$ with $\delta_k \in \mathcal{E}_i$ and $\pi_j \in \mathcal{E}_i$. Since $\texttt{Des}(\mathcal{E}_i) \subseteq \mathcal{PD}$ (according to Property 2), then $\exists d_1, d_2 \in \texttt{Des}(\mathcal{E}_i)$ such that $d_1 \equiv \neg d_2$. Three possible situations may occur:

a. $\exists \pi_1, \pi_2 \in \mathcal{E}_i \cap \mathcal{A}_p$ such that $\texttt{CONC}(\pi_1) = d_1$, and $\texttt{CONC}(\pi_2) = d_2$. This means that $\pi_1 \mathcal{R}_p \pi_2$, thus $\pi_1 \mathcal{R} \pi_2$. This is impossible since $\mathcal{E}_i$ is a stable extension, thus it is supposed to be conflict-free.

b. $\exists \delta_1, \delta_2 \in \mathcal{E}_i \cap \mathcal{A}_d$ such that $d_1 \in \texttt{DESIRES}(\delta_1)$ and $d_2 \in \texttt{DESIRES}(\delta_2)$. This means that $\delta_1 \mathcal{R}_d \delta_2$, thus $\delta_1 \mathcal{R} \delta_2$. This is impossible since $\mathcal{E}_i$ is a stable extension, thus it is supposed to be conflict-free.

c. $\exists \delta \in \mathcal{E}_i \cap \mathcal{A}_d, \exists \pi \in \mathcal{E}_i \cap \mathcal{A}_p$ such that $d_1 \in \texttt{DESIRES}(\delta)$ and $d_2 = \texttt{CONC}(\pi)$. Since $d_1 \in \texttt{DESIRES}(\delta)$, thus $\exists \delta' \in \texttt{SUB}(\delta)$ such that $\texttt{CONC}(\delta') = d_1$. This means that $\delta' \mathcal{R}_{pdp} \pi$, thus $\delta' \mathcal{R} \pi$. However, since $\delta \in \mathcal{E}_i$, thus $\delta' \in \mathcal{E}_i$. This is impossible since $\mathcal{E}_i$ is a stable extension, thus it is supposed to be conflict-free.

[4] Because $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b) = \bigcup \texttt{SUPP}(\alpha_i)$ with $\alpha_i \in \mathcal{E}_i \cap \mathcal{A}_b$; so, $\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b) = \texttt{Base}(\mathcal{E}_i \cap \mathcal{A}_b)$.

As direct consequence of the above result, an intention set is consistent. Formally:

Theorem 2. *Under the stable semantics, each set of intentions of* $\mathsf{CAF}_{\mathsf{PR}}$ *is consistent.*

Proof. Let $\mathcal{I}$ be a set of intentions of $\mathsf{CAF}_{\mathsf{PR}}$. Let us suppose that $\mathcal{I}$ is inconsistent. From the definition of an intention set, it is clear that $\mathcal{I} \subseteq \texttt{Des}(\mathcal{E}_i)$ with $\mathcal{E}_i$ is a C-stable extension of $\mathsf{CAF}_{\mathsf{PR}}$. However, according to Theorem 1 the set $\texttt{Des}(\mathcal{E}_i)$ is consistent.

Our system satisfies also the rationality postulate concerning the closedness of the extensions [5]. Namely, the set of arguments that can be built from the beliefs, desires, and plans involved in a given stable extension, is that extension itself. Let $\mathcal{E}_i$ be a C-stable extension. $\mathcal{A}$s is the set of arguments built from $\texttt{Bel}(\mathcal{E}_i)$, $\texttt{Des}(\mathcal{E}_i)$, the plans involved in building arguments of $\mathcal{E}_i$, and the base $\mathcal{B}_d$.

Theorem 3. *(Closedness) Let* $\mathcal{E}_1, \ldots, \mathcal{E}_n$ *be the* C*-stable extensions of* $\mathsf{CAF}_{\mathsf{PR}}$. $\forall \mathcal{E}_i$, $i = 1, \ldots, n$, *it holds that:* $\texttt{Arg}(\texttt{Bel}(\mathcal{E}_i)) = \mathcal{E}_i \cap \mathcal{A}_b$ *and* $\mathcal{A}$s $= \mathcal{E}_i$.

Proof. Let $\mathcal{E}_i$ be a C-stable extension of the system $\mathsf{CAF}_{\mathsf{PR}}$. $\mathcal{E}_i$ is also a stable extension of $\mathsf{AF}_{\mathsf{PR}}$ (according to [9]).
1. Let us show that $\texttt{Arg}(\texttt{Bel}(\mathcal{E}_i)) = \mathcal{E}_i \cap \mathcal{A}_b$. According to Theorem 1, it is clear that $\texttt{Bel}(\mathcal{E}_i) = \texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)$. Moreover, according to Property 13, $\mathcal{E}_i \cap \mathcal{A}_b$ is a stable extension of $\langle \mathcal{A}_b, \mathcal{R}_b \rangle$. Besides, according to [6] $\texttt{Arg}(\texttt{Bel}(\mathcal{E}_i \cap \mathcal{A}_b)) = \mathcal{E}_i \cap \mathcal{A}_b$, thus $\texttt{Arg}(\texttt{Bel}(\mathcal{E}_i)) = \mathcal{E}_i \cap \mathcal{A}_b$.
2. Let us show that $\mathcal{A}$s $= \mathcal{E}_i$. The case $\mathcal{E}_i \subseteq \mathcal{A}$s is trivial. Let us show that $\mathcal{A}$s $\subseteq \mathcal{E}_i$. Let us suppose that $\exists y \in \mathcal{A}$s and $y \notin \mathcal{E}_i$. There are three possible situations:

2.1. $y \in \mathcal{A}\text{s} \cap \mathcal{A}_b$: Since $y \notin \mathcal{E}_i$, this means that $\exists \alpha \in \mathcal{E}_i \cap \mathcal{A}_b$ such that $\alpha \mathcal{R}_b y$. Thus, $\texttt{SUPP}(\alpha) \cup \texttt{SUPP}(y) \vdash \bot$. However, $\texttt{SUPP}(\alpha) \subseteq \texttt{Bel}(\mathcal{E}_i)$ and $\texttt{SUPP}(y) \subseteq \texttt{Bel}(\mathcal{E}_i)$, thus $\texttt{SUPP}(\alpha) \cup \texttt{SUPP}(y) \subseteq \texttt{Bel}(\mathcal{E}_i)$. This means that $\texttt{Bel}(\mathcal{E}_i)$ is inconsistent. According to Theorem 1 this is impossible.

2.2. $y \in \mathcal{A}\text{s} \cap \mathcal{A}_d$: Since $y \notin \mathcal{E}_i$, this means that $\exists x \in \mathcal{E}_i$ such that $x \mathcal{R} y$. There are three situations:

2.2.1. $x \in \mathcal{A}_b$ This means that $\texttt{BELIEFS}(y) \cup \texttt{SUPP}(x) \vdash \bot$. However, $\texttt{BELIEFS}(y) \cup \texttt{SUPP}(x) \subseteq \texttt{Bel}(\mathcal{E}_i)$. Thus, $\texttt{Bel}(\mathcal{E}_i)$ is inconsistent. This contradicts Theorem 1.

2.2.2 $x \in \mathcal{A}_d$This means that $\texttt{DESIRES}(y) \cup \texttt{DESIRES}(x) \vdash \bot$. However, $\texttt{DESIRES}(y) \cup \texttt{DESIRES}(x) \subseteq \texttt{Des}(\mathcal{E}_i)$. Thus, $\texttt{Des}(\mathcal{E}_i)$ is inconsistent. This contradicts Theorem 1.

2.2.3. $x \in \mathcal{A}_p$ This means that $\texttt{DESIRES}(y) \cup \texttt{CONC}(x) \vdash \bot$. However, $\texttt{DESIRES}(y) \cup \texttt{CONC}(x) \subseteq \texttt{Des}(\mathcal{E}_i)$. Thus, $\texttt{Des}(\mathcal{E}_i)$ is inconsistent. This contradicts Theorem 1.

2.3. $y \in \mathcal{A}\text{s} \cap \mathcal{A}_p$: Since $y \notin \mathcal{E}_i$, this means that $\exists x \in \mathcal{E}_i$ such that $x \mathcal{R} y$. There are three situations:

2.3.1. $x \in \mathcal{A}_b$ This means that $x \mathcal{R}_{bp} y$, thus $\texttt{SUPP}(x) \cup \texttt{Prec}(y) \vdash \bot$. However, $\texttt{SUPP}(x) \cup \texttt{Prec}(y) \subseteq \texttt{Bel}(\mathcal{E}_i)$. Thus, $\texttt{Bel}(\mathcal{E}_i)$ is inconsistent. This contradicts Theorem 1.

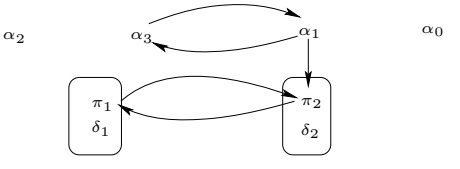
2.3.2. $x \in \mathcal{A}_d$ This means that $x\mathcal{R}_{pdp}y$, thus $\texttt{DESIRES}(x) \cup \texttt{CONC}(y) \vdash \bot$. However, $\texttt{DESIRES}(x) \cup \texttt{CONC}(y) \subseteq \texttt{Des}(\mathcal{E}_i)$. Thus, $\texttt{Des}(\mathcal{E}_i)$ is inconsistent. This contradicts Theorem 1.

2.3.3. $x \in \mathcal{A}_p$ This means that $x\mathcal{R}_p y$. There are three different cases:

- $\texttt{Prec}(x) \cup \texttt{Prec}(y) \vdash \bot$. However, $\texttt{Prec}(x) \cup \texttt{Prec}(y) \subseteq \texttt{Bel}(\mathcal{E}_i)$. Thus, $\texttt{Bel}(\mathcal{E}_i)$ is inconsistent. This contradicts Theorem 1.
- $\texttt{Postc}(x) \cup \texttt{Prec}(y) \vdash \bot$. We know that y is built using one of the plans of $\mathcal{E}_i$, say $p = \langle S, T, d\rangle$. Thus, $\exists \pi \in \mathcal{E}_i$ such that $\pi = \langle p, d'\rangle$. Thus, $\texttt{Postc}(x) \cup \texttt{Prec}(\pi) \vdash \bot$, consequently, $x\mathcal{R}\pi$. This is impossible since $\mathcal{E}_i$ is a stable extension, thus it is supposed to be conflict-free.
- $\texttt{Postc}(x) \cup \texttt{Postc}(y) \vdash \bot$. Since $y \in \mathcal{A}$s, thus y is built using one of the plans of $\mathcal{E}_i$, say $p = \langle S, T, d\rangle$. Thus, $\exists \pi \in \mathcal{E}_i$ such that $\pi = \langle p, d'\rangle$. Thus, $\texttt{Postc}(x) \cup \texttt{Postc}(\pi) \vdash \bot$, consequently, $x\mathcal{R}\pi$. This is impossible since $\mathcal{E}_i$ is a stable extension, thus it is supposed to be conflict-free.

8 Illustrative Example

In this section, we illustrate the above system on a simple example.



The meaning of these arguments is the following:

- α_0: My AAMAS paper is accepted and AAMAS conference is in Portugal so I go to AAMAS in Portugal
- α_1: My AAMAS paper is accepted and it is scheduled Day D so I am not available Day D
- α_2: My sister's wedding is scheduled Day D
- α_3: My sister's wedding is scheduled Day D so I must be available Day D
- δ_1: I go to AAMAS in Portugal so I desire to visit Portugal
- δ_2: My sister's wedding is scheduled Day D so I desire to go to my sister's wedding Day D
- π_1: My AAMAS paper is accepted, my institute pays my AAMAS mission, AAMAS is in Portugal so I can realize my desire to visit Portugal
- π_2: I am available Day D, my sister's wedding is scheduled Day D, I know where and how to go to my sister's wedding Day D so I can realize my desire to go to my sister's wedding Day D

So, we have:

- the constraint: $C = (\pi_1 \Rightarrow \delta_1) \wedge (\pi_2 \Rightarrow \delta_2)$;

- the C-preferred and C-stable extensions are $\mathcal{E}_1 = \{\alpha_2, \alpha_0, \alpha_3, \pi_2, \delta_2, \delta_1\}$, $\mathcal{E}_2 = \{\alpha_2, \alpha_0, \alpha_3, \pi_1, \delta_1, \delta_2\}$, $\mathcal{E}_3 = \{\alpha_2, \alpha_0, \alpha_1, \pi_1, \delta_1, \delta_2\}$,
- the sets of intentions are { visit Portugal }, { go to my sister's wedding }.

9 Related Works

A number of attempts have been made to use formal models of argumentation as a basis for PR. In fact the use of arguments for justifying an action has already been advocated by philosophers like Walton [21] who proposed the famous *practical syllogism*:

- G is a goal for agent X
- Doing action A is sufficient for agent X to carry out G
- Then, agent X ought to do action A

The above syllogism, which would apply to the means-end reasoning step, is in essence already an argument in favor of doing action A. However, this does not mean that the action is warranted, since other arguments (called counter-arguments) may be built or provided against the action.

In [1], an argumentation system is presented for generating consistent plans from a given set of desires and planning rules. This was later extended with argumentation systems that generate the desires themselves [3]. This system suffers from three main drawbacks: i) exhibiting a form of wishful thinking, ii) desires may depend only on beliefs, and iii) some undesirable results may be returned due to the separation of the two steps of PR. Due to lack of space, we will unfortunately not give an example where anomalies occur using that approach. In [15], the problem of wishful thinking has been solved. However, the separation of the two steps was kept. Other researchers in AI like Atkinson and Bench Capon [4] are more interested in studying the different argument schemes that one may encounter in practical reasoning. Their starting point was the above practical syllogism of Walton. The authors have defined different variants of this syllogism as well as different ways of attacking it. However, it is not clear how all these arguments can be put together in order to answer the critical question of PR "what is the right thing to do in a given situation?". Our work can be viewed as a way for putting those arguments all together.

10 Conclusion

The paper has tackled the problem of practical reasoning, which is concerned with the question "what is the best thing to do at a given situation?" The approach followed here for answering this question is based on argumentation theory, in which choices are explained and justified by arguments. The contribution of this paper is two-fold. To the best of our knowledge, this paper proposes the first argumentation system that computes the intentions in one step, *i.e.* by combining desire generation and planning. This avoids undesirable results encountered by previous proposals in the literature. This has been possible due to the use of constrained argumentation systems developed in [9]. The second contribution of the paper consists of studying deeply the properties of argumentation-based practical reasoning.

This work can be extended in different ways. First, we are currently working on relaxing the assumption that the attack relation among instrumental arguments is binary. Indeed, it may be the case that more than two plans may be conflicting while each pair of them is compatible. Another important extension would be to introduce preferences to the system. The idea is that beliefs may be pervaded with uncertainty, desires may not have equal priorities, and plans may have different costs. Thus, taking into account these preferences will help to reduce the intention sets into more relevant ones.

In [7], it has been shown that an argument may not only be attacked by other arguments, but may also be supported by arguments. It would be interesting to study the impact of such a relation between arguments in the context of PR. Another area of future work is investigating the proof theories of this system. The idea is to answer the question "is a given potential desire a possible intention of the agent ?" without computing the whole preferred extensions. Finally, an interesting area of future work is investigating the relationship between our framework and axiomatic approaches to BDI agents.

References

1. Amgoud, L.: A formal framework for handling conflicting desires. In: Nielsen, T.D., Zhang, N.L. (eds.) ECSQARU 2003. LNCS, vol. 2711, pp. 552–563. Springer, Heidelberg (2003)
2. Amgoud, L., Cayrol, C.: Inferring from inconsistency in preference-based argumentation frameworks. International Journal of Automated Reasoning 29(2), 125–169 (2002)
3. Amgoud, L., Kaci, S.: On the generation of bipolar goals in argumentation-based negotiation. In: Rahwan, I., Moraïtis, P., Reed, C. (eds.) ArgMAS 2004. LNCS, vol. 3366, Springer, Heidelberg (2005)
4. Atkinson, K., Bench-Capon, T., McBurney, P.: Justifying practical reasoning. In: Reed, C., Grasso, F., Carenini, G. (eds.) Proceedings of the Fourth Workshop on Computational Models of Natural Argument (CMNA 2004), pp. 87–90 (2004)
5. Caminada, M., Amgoud, L.: An axiomatic account of formal argumentation. In: Proceedings of the 20th National Conference on Artificial Intelligence (AAAI 2005), pp. 608–613. AAAI Press, Menlo Park (2005)
6. Cayrol, C.: On the relation between argumentation and non-monotonic coherence-based entailment. In: Proceedings of the International Joint Conference on Artifitial Intelligence (IJCAI 1995), pp. 1443–1448 (1995)
7. Cayrol, C., Lagasquie-Schiex, M.-C.: On the acceptability of arguments in bipolar argumentation frameworks. In: Godo, L. (ed.) ECSQARU 2005. LNCS, vol. 3571, pp. 378–389. Springer, Heidelberg (2005)
8. Coste-Marquis, S., Devred, C., Marquis, P.: Symmetric argumentation frameworks. In: Godo, L. (ed.) ECSQARU 2005. LNCS, vol. 3571, pp. 317–328. Springer, Heidelberg (2005)
9. Coste-Marquis, S., Devred, C., Marquis, P.: Constrained argumentation frameworks. In: Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006), pp. 112–122 (2006)
10. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77(2), 321–358 (1995)
11. Ghallab, M., Nau, D., Traverso, P.: Automated planning, theory and practice. Elsevier, Morgan Kaufmann (2004)

12. Harman, G.: Practical aspects of theoretical reasoning. The Oxford Handbook of Rationality, 45–56 (2004)
13. Hulstijn, J., van der Torre, L.: Combining goal generation and planning in an argumentation framework. In: Heskes, T., Lucas, P., Vuurpijl, L., Wiegerinck, W. (eds.) Proceedings of the 15th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC 2003), Katholieke Universiteit Nijmegen, October 2003, pp. 155–162 (2003)
14. Karacapilidis, N., Papadias, D.: Computer supported argumentation and collaborative decision making: the HERMES system. Information systems 26(4), 259–277 (2001)
15. Rahwan, I., Amgoud, L.: An Argumentation-based Approach for Practical Reasoning . In: Weiss, G., Stone, P. (eds.) 5th International Joint Conference on Autonomous Agents & Multi Agent Systems, AAMAS 2006, Hakodate, Japan, pp. 347–354. ACM Press, New York (2006)
16. Raz, J.: Practical reasoning. Oxford University Press, Oxford (1978)
17. Russel, S., Norvig, P.: Artificial Intelligence. A modern approach. Prentice-Hall, Englewood Cliffs (1995)
18. Simari, G.R., Loui, R.P.: A mathematical treatment of defeasible reasoning and its implementation. Artificial Intelligence 53, 125–157 (1992)
19. Thomason, R.H.: Desires and defaults: A framework for planning with inferred goals. In: Cohn, A.G., Giunchiglia, F., Selman, B. (eds.) KR 2000: Principles of Knowledge Representation and Reasoning, Proceedings of the Seventh International Conference, Breckenridge, Colorado, USA, pp. 702–713. Morgan Kaufmann, San Francisco (2000)
20. Vreeswijk, G.: Abstract argumentation systems. Artificial Intelligence 90(1–2), 225–279 (1997)
21. Walton, D.: Argument schemes for presumptive reasoning, vol. 29. Lawrence Erlbaum Associates, Mahwah (1996)
22. Wooldridge, M.J.: Reasoning about Rational Agents. MIT Press, Cambridge (2000)

An Argumentation Framework Based on *Strength* for Ontology Mapping

Cássia Trojahn[1], Paulo Quaresma[1], and Renata Vieira[2]

[1] Departamento de Informática, Universidade de Évora, Portugal
[2] Faculdade de Informática, Pontifícia Universidade Católica do Rio Grande do Sul, Brazil
cassia@di.uevora.pt, pq@di.uevora.pt, renata.vieira@pucrs.br

Abstract. In the field of ontology mapping, using argumentation to combine different mapping approaches is an innovative research area. We had extended the Value-based Argumentation Framework (VAF) in order to represent arguments with *confidence degrees*, according to the similarity degree between the terms being mapped. The mappings are computed by agents using different mapping approaches. Based on their preferences and confidences, the agents compute their preferred mapping sets. The arguments in such preferred sets are viewed as the set of globally acceptable arguments. In previous work we had used discrete classes to represent the *confidence degrees* (certainty and uncertainty). In this paper, we propose to use continuous values from the interval [0,1]. Here, *confidence* is treated as *strength*. Using a threshold for the *strength* we can reduce the set of mappings and adjust the values of precision. We evaluate the use of *strength* against the previous confidence as discrete classes. The results are promising, especially what concerns precision.

1 Introduction

Ontology mapping is the process of linking corresponding terms from different ontologies. The mapping result can be used for ontology merging, agent communication, query answering, or for navigation on the Semantic Web. [19], [20], and [7] present a broad overview of the various approaches on automated ontology matching. Basically, the ontology mapping problem involves to combine different approaches. Using argumentation to solve this problem is an innovative research.

We had extended an Argumentation Framework, namely Value-based Argumentation Framework (VAF)[3], in order to represent arguments with *confidence degrees*. The VAF allows to determine which arguments are acceptable, with respect to different *audiences* represented by different agents. We then associate to each argument a *confidence degree*, representing how confident an agent is in the similarity of two ontology terms.

Our agents apply different mapping approaches and cooperate in order to exchange their local results (arguments). Next, based on their preferences and confidence of the arguments, the agents compute their preferred mapping sets. The arguments in such preferred sets are viewed as the set of globally acceptable

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 57–71, 2009.

arguments. Our approach is able to give a formal motivation for the composite mapping approaches.

In previous work [24][25] we had used discrete classes to represent the *confidence degrees* (certainty and uncertainty). In this paper, we propose to use continuous values from the interval [0,1]. Here, *confidence* is treated as *strength*. Using a threshold for the *strength* we can reduce the set of mappings and adjust the values of precision. In a scenario where the mappings must be defined on the fly (i.e., web systems involving agent communication), precision is preferred than recall. On the other side, when the mapping system is used to help users in the mapping process, it is interesting to reduce the set of mappings. We evaluate the use of *strength* against the previous discrete classes. The results are promising, specially what concerns precision.

The paper is structured as follows. Firstly, in Section 2, we comment on ontologies and approaches for ontology mapping. Section 3 presents the Argumentation Framework, upon which our model rely. Section 4 presents our Strength based Argumentation Framework (S-VAF). Section 5 presents the evaluation. Section 6 comments on related work. Finally, section 7 presents the final remarks and future work.

2 Ontologies and Ontology Mapping Approaches

The standard definition of ontology is from [10]: "an explicit specification of the conceptualization of the domain". From this definition [8] point out that: (a) the ontology makes things explicit – without an ontology many design assumptions may be implicit in the executable representation; (b) the ontology is supposed to be formal: the notions it captures are thus precise and unambiguous; (c) the ontology concerns some specific domain; (d) the ontology represents a conceptualization – different people will conceptualize a domain differently according to experience, and their tasks in the domain – and there is no a single ontology applicable to a domain. Specifically, ontologies contain the types of *objects* in the domain; the *attributes* which these objects may have; the *relationships* which these objects may enter into; and the *values* that the attributes may have for particular types.

Ontology mapping is the process of finding correspondences between two ontologies, using as input their types of objects (classes), attributes, relationships or value of attributes. For instance, if two objects correspond, they mean the same thing, or closely related things. [19], [20], and [7] present a broad overview of the various approaches on automated ontology matching. In this paper, we focus in how to combine mapping approaches using argumentation. Three specific kinds of mapping approaches are considered: lexical ([22][18]), semantic and structural (see [11]). Lexical approaches apply metrics to compare string similarity. One well-known measure is the edit distance [14], which is given by the minimum number of operations (insertion, deletion, or substitution of a single character) needed to transform one string into another.

Semantic approaches consider the semantic relations between concepts to measure the similarity between them, usually on the basis of semantic oriented linguistic resources. The well-known WordNet[1] database, a large repository of English semantically related items, has been used to provide these relations. This kind of mapping is complementary to the pure string similarity metrics. It is not uncommon the cases where string metrics fail to identify high similarity between strings that represent completely different concepts (i.e, the words "score" and "store"). Semantic-structural approaches have been explored [11]. In this case, the positions of the terms in the ontology hierarchy are considered, i.e, terms more generals and terms more specifics are also considered as input to the mapping process.

3 Argumentation Framework

Our argumentation framework for ontology mapping is based on the Value-based Argumentation Frameworks (VAF)[3], a development of the classical argument system of Dung [6]. First, we present the Dung's framework, upon which the VAF rely. Next, we present the VAF and our extended framework.

3.1 Classical Argumentation Framework

Dung [6] defines an argumentation framework as follows.

Definition 3.1.1. An Argumentation Framework is a pair $AF = (AR, attacks)$, where AR is a set of arguments and *attacks* is a binary relation on AR, i.e., $attacks \subseteq \text{AR} \times \text{AR}$. An *attack*(A,B) means that the argument A attacks the argument B. A set of arguments S attacks an argument B if B is attacked by an argument in S.

The key question about the framework is whether a given argument A, $A \in AR$, should be accepted. One reasonable view is that an argument should be accepted only if every attack on it is rebutted by an accepted argument [6]. This notion produces the following definitions:

Definition 3.1.2. An argument $A \in AR$ is *acceptable* with respect to set arguments *S(acceptable(A,S))*, if $(\forall\ x)(x \in AR) \wedge (attacks(x,A)) \longrightarrow (\exists\ y)(y \in S) \wedge attacks(y,x)$

Definition 3.1.3. A set S of arguments is *conflict-free* if $\neg(\exists\ x)(\exists\ y)((x \in S) \wedge (y \in S) \wedge attacks(x,y))$

Definition 3.1.4. A conflict-free set of arguments S is *admissible* if $(\forall x)(x \in S) \longrightarrow acceptable(x,S)$

Definition 3.1.5. A set of arguments S is a *preferred extension* if it is a maximal (with respect to inclusion set) admissible set of AR.

[1] http://www.wordnet.princeton.edu

A *preferred extension* represent a consistent position within AF, which can defend itself against all attacks and which cannot be further extended without introducing a conflict. The purpose of [3] in extending the AF is to allow associate arguments with the social values they advance. Then, the attack of one argument on another is evaluated to say whether or not it succeeds by comparing the preferences of the values advanced by the arguments concerned.

3.2 Value-Based Argumentation Framework

In Dung's frameworks, attacks always succeed. However, in many domains, including the one under consideration, arguments lack this coercive force: they provide reasons which may be more or less persuasive [13]. Moreover, their persuasiveness may vary according to their audience. The VAF is able to distinguish attacks from successful attacks, those which defeat the attacked argument, with respect to an ordering on the preferences that are associated with the arguments. It allows accommodate different audiences with different interests and preferences.

Definition 3.2.1. A Value-based Argumentation Framework (VAF) is a 5-tuple $VAF = (AR, attacks, V, val, P)$ where $(AR, attacks)$ is an argumentation framework, V is a nonempty set of values, val is a function which maps from elements of AR to elements of V and P is a set of possible audiences. For each $A \in AR$, $val(A) \in V$.

Definition 3.2.2. An Audience-specific Value Based Argumentation Framework (AVAF) is a 5-tuple $VAF_a = (AR, attacks, V, val, valpref_a)$ where AR, emphattacks, V and val are as for a VAF, a is an audience and $valpref_a$ is a preference relation (transitive, irreflexive and asymmetric) $valpref_a \subseteq V \times V$, reflecting the value preferences of audience a. $valpref(v_1, v_2)$ means v_1 is preferred to v_2.

If V contains a single value, or no preference between the values has been defined, the AVAF becomes a standard AF. If each argument can map to a different value, a Preference Based Argumentation Framework is obtained [1].

Definition 3.2.3. An argument $A \in AR$ defeats$_a$ (or *successfully attacks*) an argument $B \in AR$ for audience a if and only if both $attacks(A,B)$ and not $valpref(val(B), val(A))$.

Definition 3.2.4. An argument $A \in AR$ is *acceptable* to audience a ($acceptable_a$) with respect to set of arguments S, $acceptable_a(A,S))$ if $(\forall\ x)\ ((x \in AR \wedge defeats_a\ (x,A)) \longrightarrow (\exists y)((y \in S) \wedge defeats_a(y,x)))$.

Definition 3.2.5. A set S of arguments is *conflict-free* for audience a if $(\forall\ x)(\forall\ y)((x \in S \wedge y \in S) \longrightarrow (\neg attacks(x,y) \vee valpref(val(y),val(x)) \in valpref_a))$.

Definition 3.2.6. A *conflict-free* set of argument S for audience a is *admissible* for an audience a if $(\forall x)(x \in S \longrightarrow acceptable_a(x,S))$.

Definition 3.2.7. A set of argument S in the VAF is a *preferred extension* for audience *a (preferred$_a$)* if it is a maximal (with respect to set inclusion) *admissible* for audience *a* of AR.

In order to determine the preferred extension with respect to a value ordering promoted by distinct audiences, [3] introduces the notion of *objective* and *subjective* acceptance.

Definition 3.2.8. An argument $x \in AR$ is *subjectively* acceptable if and only if x appears in the preferred extension for some specific audiences but not all. An argument $x \in AR$ is *objectively* acceptable if and only if, x appears in the preferred extension for every specific audience. An argument which is neither objectively nor subjectively acceptable is said to be *indefensible*.

3.3 *Strength* Based Argumentation Framework (S-VAF)

We extend the VAF in order to represent arguments with *strength*, which represents the confidence that an agent has in some argument. One element has been added to VAF: a function which maps from arguments to real values from the interval [0,1]. We assumed that the *strength* is a relevant criterion in the ontology mapping domain, representing the confidence measure by using the mapping approach.

Definition 3.3.1. A Strength based Argumentation Framework (S-VAF) is a 6-tuple *(AR, attacks,V,val,P,valS)* where *(AR,attacks,V,val, P)* is a value-based argumentation framework, and *valS* is a function which maps from elements of AR to real values from the interval [0,1] representing the *strength* of the argument.

Definition 3.3.2. In the S-VAF, an argument $x \in AR$ defeats$_a$ an argument $y \in AR$ for audience *a* if and only if *attacks(x,y)* $\wedge$ *((valS(x) > valS(y))* $\vee$ *(¬ valpref(val(y),val(x))* $\wedge$ *(¬ (valS(y) > valS(x)))).*

An attack succeeds if (a) the *strength* of the attacking argument is greater than the *strength* of the argument being attacked; or if (b) the argument being attacked does not have greater preference value than attacking argument (or if both arguments relate to the same preference values) and the *strength* of the argument being attacked is not greater than the attacking argument.

Definition 3.3.3. In the S-VAF, an argument $A \in AR$ is *acceptable* to audience *a* (*acceptable$_a$*) with respect to set of arguments *S, acceptable$_a$(A,S))* if *(∀ x) ((x ∈ AR* $\wedge$ *defeats$_a$ (x,A))* $\longrightarrow$ *(∃y)((y ∈ S)* $\wedge$ *defeats$_a$(y,x))).*

Definition 3.3.4. In the S-VAF, a set S of arguments is *conflict-free* for audience *a* if *(∀x)(∀y) ((x ∈ S* $\wedge$ *y ∈ S)* $\longrightarrow$ *(¬attacks(x, y)* $\vee$ *(¬(valS(x) > valS(y))* $\wedge$ *(valpref(val(y), val(x))* $\vee$ *(valS(y) > valS(x)))))).*

Definition 3.3.5. A set of argument S in the S-VAF is a *preferred extension* for audience *a (preferred$_a$)* if it is a maximal (with respect to set inclusion) *admissible* for audience *a* of AR.

It is important to distinguish the difference between *values* and *strengths*. There are different types of agents representing different mapping approaches. Each approach represent a *value* and each agent represents an audience, with preferences between the *values*. The *values* are used to determine the preference between the different agents. Moreover, each agent generates arguments with a *strength*, based on the confidence returned by the mapping technique. So, we extended the VAF in order to define a new notion of argument acceptability which combines *values* (related with the agent's preference) and *strength* (confidence degree of an argument). If our criterion was based only on the *strength* of the arguments, a Preference Based Argumentation Framework could be used [1].

4 S-VAF for Ontology Mapping

In this paper we consider three *values*: lexical (L), semantic (S), and structural (E) (i.e. $V = \{L, S, E\}$, where $V \in$ S-VAF). These values represent the mapping approach used by the agent and are also used to represent the audiences. Each audience has an ordering preference between the *values*. For instance, the lexical agent represents an audience where the value L is preferred to the values S and E. Our idea is not to have an individual audience with preference between the agents (i.e., semantic agent is preferred to the other agents), but it is to try accommodate different audiences (agents) and their preferences.

4.1 Argumentation Generation

First, the agents work in an independent manner, applying the mapping approaches and generating mapping sets. The mapping result will consist of a set of all possible correspondences between terms (type of objects) of two ontologies. A mapping m can be described as a 3-tuple $m = (t_1,t_2,h)$, where t_1 corresponds to a term in the ontology 1, t_2 corresponds to a term in the ontology 2, and h is one of {+,-} depending on whether the argument is that m does or does not hold. Now, we can define arguments as follows:

Definition 4.1. An *argument* $\in AR$ is a 3-tuple $x = (m,a,s)$, where m is a mapping; $a \in V$ is the value of the argument (lexical, semantic or structural); s is the *strength* of the argument.

Lexical Agent. This agent adopts the *lexical similarity* proposed by [18]. This metric is based on the Levenshtein distance [15] and considers the length of the compared terms to compute the final *lexical similarity*. A value from the interval [0,1] is returned, where 1 indicates high similarity between two terms.

Differently from the previous work [24][25], the agents are able to deal with compound terms. The first step in this process is the tokenization, where the terms are parsed into tokens by a tokenizer. The *strength* of an argument is computed according to the *lexical similarity* between each token of the two compared terms. Table 1 shows the possible values to s and h, where t_{Sn} correspond

Table 1. h and s to lexical audience

s	+ (h)
1	t_{S1} lexically similar to t_{T1}
calc-s	t_{S1} lexically similar to some t_{T1}, ..., t_{Tn} t_{S1}, ..., t_{Sn} some lexically similar to t_T t_{S1}, ..., t_{Sn} some lexically similar to some t_{T1}, ..., t_{Tn}
s	- (h)
0	otherwise

Table 2. h and s to semantic audience

s	+ (h)
1	t_{S1} semantic relation with t_{T1}
calc-s	t_{S1} some semantic relation with some t_{T1}, ..., t_{Tn} t_{S1}, ..., t_{Sn} some semantic relation with t_T t_{S1}, ..., t_{Sn} some semantic relation with some t_{T1}, ..., t_{Tn}
s	- (h)
0	otherwise

to some token of the source term (source ontology), and t_{Tn} correspond to some token of the target term (target ontology). Two tokens are *lexically similar* if the *lexical similarity* is greater than a *threshold* r.

When all tokens are *lexically similar* with each other, the terms match and the *strength* of the argument is 1. If *some* tokens of the terms are *lexically similar*, the *strength* is computed according to the number of tokens that matches, according to the *calc-s* formula, where T_S is the term from the source ontology, T_T is the term from the target ontology, and nM is the number of tokens that match between T_S and T_T:

$$\textit{calc-s} = max\left(0, \frac{max(\mid T_S \mid, \mid T_T \mid) \; - \; nM)}{max(\mid T_S \mid, \mid T_S \mid)}\right)$$

If there are no lexically similar tokens between the terms, the agent is not sure that the terms map (i.e., *strength* equals to 0), because this agent knows that other agent can resolve this mapping. In the specific case, if there is no lexical similarity between the terms, the semantic agent can resolve that mapping.

Semantic Agent. This agent considers semantic relations (i.e., synonym, hyponym, and hypernym) between terms to measure the similarity between them, on the basis of WordNet[2] database. Table 2 shows the possible values to s and h according to the semantic similarity.

When all tokens have semantic relation with each other, the *strength* of the argument is 1. If *some* tokens have semantic relation, the *strength* is computed according to the number of semantically related tokens (formula presented above).

[2] http://www.wordnet.princeton.edu

Otherwise, if there are no semantic relation between the tokens, the agent is not sure that the terms map (i.e., *strength* equals to 0), because this agent knows that other agent can resolve the mapping. In the specific case, when the searched terms are not available in WordNet, the lexical agent can decide the mapping. It is common because there is no complete lexical database for every domain (i.e., WordNet is incomplete for some domains).

Structural Agent. The structural agent considers the positions of the terms in the ontology hierarchy to verify if the terms can be mapped. First, it is verified if the super-classes of the compared terms are lexically similar. If not, the semantic similarity is used. For instance, if the super-classes of the terms are not lexically similar, but they are synonymous, an argument $x = (\mathrm{m,E},s)$, where $m = (\mathrm{t_1,t_2},+)$, is generated, where s varies according to the rules from Tables 1 or 2.

However, there are two main differences among the *strengths* returned by the lexical, semantic, and structural agents. As Table 1 and Table 2, when the agents can not resolve the mapping, the *strength* of the corresponding argument is 0. However, if the structural agent does not find similarity (lexical or semantic) between the super-classes of the compared terms, it is because the terms can not be mapped (i.e., the terms occurs in different contexts). Then, the *strength* for no mapping is 1. Otherwise, if the structural agent finds similarity between the super-classes of the compared terms, it is because they can be mapped, but it does not mean that the terms have lexical or semantic similarity, then the *strength* for the mapping is 0. For instance, for the terms "Publication/Topic" and "Publication/Proceedings", the structural agent indicates that the terms can be mapped because they have the same super-class, but not with *strength* 1 because it is not able to indicate that the terms are similar. Otherwise, for the terms "Digital-Camera/Accessories" and "Computer/ Accessories", the agent can indicate that the terms can not be mapped because they occur in different contexts (*no-mapping* with *strength* equal to 1).

4.2 Preferred Extension Generation

After generating their set of arguments, the agents exchange with each other their arguments and generate their *attacks* set. An *attack* (or counter-argument) will arise when we have arguments for the mapping between the same terms, but with conflicting values of h. For instance, an argument $x = (\mathrm{m_1,L},+)$ have as an *attack* an argument $y = (\mathrm{m_2,E},-)$, where $\mathrm{m_1}$ and $\mathrm{m_2}$ refer to the same terms in the ontologies. The argument y also represents an *attack* to the argument x.

As an example, consider the mapping between the terms "Subject" and "Topic" and the lexical and semantic agents. The lexical agent generates an argument $x = (\mathrm{m,L},0)$, where $m = (\mathrm{subject}_S,\mathrm{topic}_T,-)$; and the semantic agent generates an argument $y = (\mathrm{m,S},1)$, where $m = (\mathrm{subject}_S,\mathrm{topic}_T,+)$. For both lexical and semantic audiences, the set of arguments is AR= {x,y} and the *attacks* = {(x,y),(y,x)}.

When the set of arguments and attacks have been produced, the agents need to define which of them must be accepted. To do this, the agents compute their

preferred extension, according to the agent's preferences and *strengths* of the arguments. A set of arguments is *globally subjectively acceptable* if each element appears in the preferred extension for some agent. A set of arguments is *globally objectively acceptable* if each element appears in the preferred extension for every agent. The arguments which are neither objectively nor subjectively acceptable are considered *indefensible*.

In the example above, considering the lexical(L) and semantic(S) audiences, where $L \succ S$ and $S \succ L$, respectively, for the lexical audience, the argument y successfully attacks the argument x, while the argument x does not successfully attack the argument y for the semantic audience. Then, the preferred extension of both lexical and semantic agents is composed by the argument y.

5 Argumentation Model Evaluation

Let us consider that three agents need to obtain a consensus about mappings that link corresponding class names in two different ontologies. We have used three groups of ontologies: parts of Google and Yahoo web directories[3](Test 3), product schemas[4](Test 4), and company profiles[5](Test 8). In *Test 3*, the source ontology has 9 terms and the target ontology has 6 terms, resulting 54 possible mappings (comparisons term by term). The terms are formed from 1 to 2 tokens (for instance, "Art-History"). In *Test 4*, the source ontology has 5 terms and the target ontology has 6 terms, resulting 30 possible mappings. The terms are formed from 1 to 3 tokens. Finally, the source and target ontologies in *Test 8* have 10 and 16 classes, respectively, resulting 160 possible mappings. The terms are composed from 1 to 5 tokens (for instance "Oil-and-Gas-Exploration-and-Production" or "Petroleum-Product-Distribution").

As a mapping quality evaluation, the measures of precision, recall and f–measure were used. Precision is defined by the number of correct automated mappings divided by the number of mappings that the system returned. It measures the system's correctness or accuracy. Recall indicates the number of correct mappings returned by the system divided by the number of manual mappings. It measures how complete or comprehensive the system is in its extraction of relevant mappings. F–measure is a weighted harmonic mean of precision and recall.

First, we compared the results from using *confidence* as discrete classes (*certainty* and uncertainty), based on E-VAF, as proposed in [24][25], against the results from using *strength* as continuous values. When considering only the mappings (h equals $+$) with *certainty* (Figure 1 (a)) and the mappings with *strength* equals to 1 (Figure 2 (a)), the values for f-measure (and corresponding precision and recall) were the same, for the three tests. However, when considering both mappings with *certainty* and *uncertainty* (Figure 1 (b)) against the use of a *threshold* (0.70) (Figure 2 (b)), better values of precision were obtained using *strength*.

[3] http://dit.unitn.it/~accord/Experimentaldesign.html (Test 3)
[4] http://dit.unitn.it/~accord/Experimentaldesign.html (Test 4)
[5] http://dit.unitn.it/~accord/Experimentaldesign.html (Test 8)

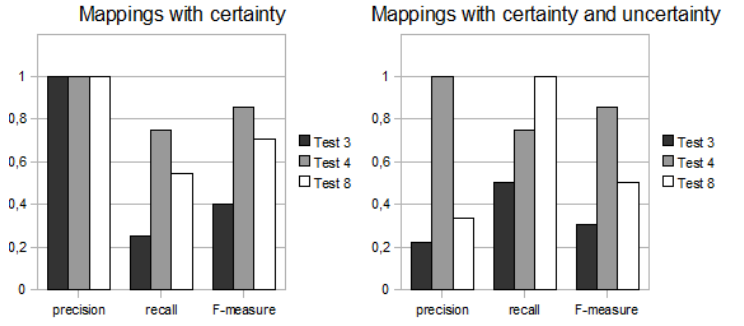


Fig. 1. Mappings with *confidence*: (a) *certainty*; (b) *certainty* + *uncertainty*

Next, we analyzed more specifically the use of different values of *threshold* (Figure 3). When using a low *threshold*, the recall is 1 and the precision is lower. When using a high *threshold* (0.70), the precision is 1 and the recall is lower. In a scenario where the mappings must be defined on the fly, the precision of the mappings is more valuable than the recall (i.e., web systems involving agent communication).

On the other side, when the mapping system is used to help users in the mapping process, it is interesting reduce the set of mappings. When using the *confidence uncertainty* it is not possible. However, we can do that using *thresholds* for *strength*. Specifically for *Test 8* (larger ontology), Figure 4 shows the number of mappings using different values for *threshold* (40 mappings returned when considering the mapping with *certainty* and *uncertainty*). In the scenario under consideration, there are two advantages to use *strength* and *thresholds*. First, the user can adjust the *threshold*. Second, when reducing the set of mapping, it is easier for the user to analyze the resulting mappings. As shown in Figure 4, using the *threshold* the set can be reduced to 26, 12 and 6 mappings. In this sense, our system can help the users to reduce the set of possible mappings, using different *thresholds* for *strength*.

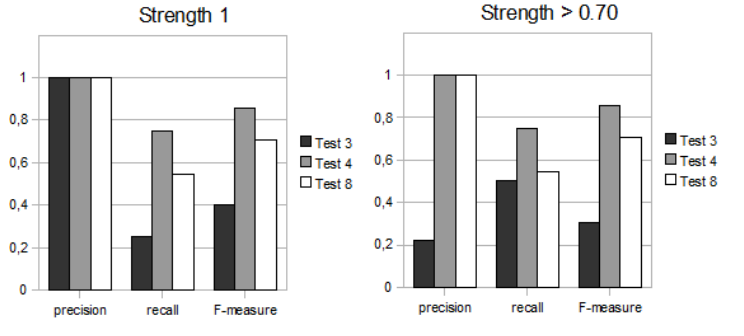


Fig. 2. Mappings with *strength*

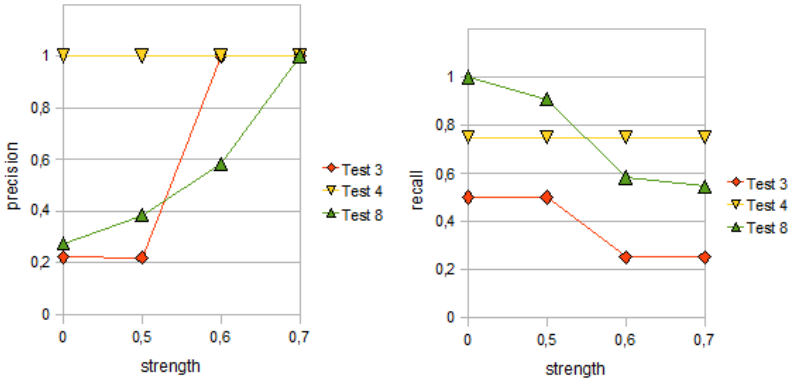


Fig. 3. Precision and recall for the three tests using different thresholds

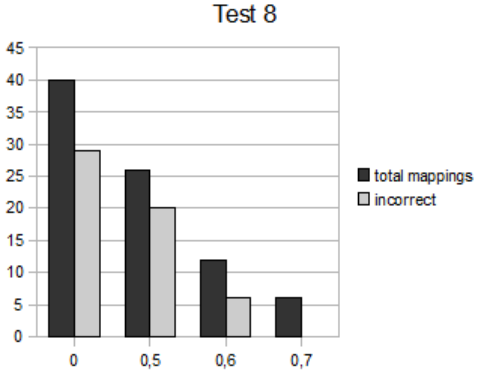


Fig. 4. Comparative results

Second, we compared our proposal with three mapping systems: Cupid[16], COMA[5], and S-Match[9]. The comparative results among these three systems are available in [9]. We utilized these results as criteria to evaluate our argumentation model, but the details of these tests (implementations, time of run, processor, etc) are not available. The evaluation of ontology mapping systems still lacks well established benchmarks, therefore our choices on evaluation were based on the availability of reported results of previous systems. Figure 5 shows the comparative results. We used a threshold r equals to 0.8 for the lexical agent classifies the mappings (terms with lexical similarity greater than 0.8 are considered similar) and a threshold to eliminate the mappings that have *strength* below 0.75. Our model returned better precision than Cupid and COMA, and equal precision when compared to S-Match (precision equal to 1). When comparing the f–measure values, our model had better result than Cupid.

Differently from these works, our model uses argumentation to combine mapping approaches. Cupid uses a weighted similarity which is a mean of linguistic and structural similarities. COMA represents a generic system to combine

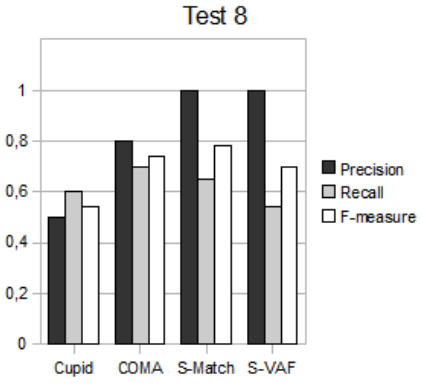


Fig. 5. Comparative results

matching results, which is a set of mapping elements specifying the matching schema elements together with a similarity 2 [0,1] indicating the plausibility of their correspondence. S-Match algorithm is based on the semantic and structural similarities, where the semantic matcher provides the input to the structural matcher.

Although our implementation does not provide the best solution for the ontology mapping problem for these experimental tests as yet, we claim that our main contribution is to propose a model that can be used to combine different approaches.

Using argumentation has the following advantages: the agents are independent to each other; many other agents can be easily added to our model, without having to modify the implementation; there are several techniques for ontology mappings, which can be adapted according to domain, kind of ontologies, and available resources (for instance, in the context of some languages, there is no lexical databases such WordNet).

6 Related Work

In the field of ontology argumentation few approaches are being proposed. Basically, the closer proposal is from [13][12], where an argument framework is used to deal with arguments that support or oppose candidate correspondences between ontologies. The candidate mappings are obtained from an Ontology Mapping Repository (OMR) – the focus is not how the mappings are computed – and argumentation is used to accommodate different agent's preferences. In our approach mappings are computed by the specialized agents described in this paper, and argumentation is used to solve conflicts between the individual results.

We find similar proposals in the field of ontology negotiation. [23] presents an ontology to serve as the basis for agent negotiation, the ontology itself is not the object being negotiated. A similar approach is proposed by [4], where agents agree on a common ontology in a decentralized way. Rather than being the goal of each

agent, the ontology mapping is a common goal for every agent in the system. [2] presents an ontology negotiation model which aims to arrive at a common ontology which the agents can use in their particular interaction. We, on the other hand, are concerned with delivering mapping pairs found by a group of agents using argumentation. [21] describes an approach for ontology mapping negotiation, where the mapping is composed by a set of semantic bridges and their inter-relations, as proposed in [17]. The agents are able to achieve a consensus about the mapping through the evaluation of a confidence value that is obtained by utility functions. According to the confidence value the mapping rule is accepted, rejected or negotiated. Differently from [21], we do not use utility functions. Our model is based on cooperation and argumentation, where the agents change their arguments and by argumentation they select the preferred mapping.

7 Final Remarks and Future Work

In this paper we proposed to use continuous values to represent the *strength* of arguments, which represents the *confidence degree* that an agent has in the mapping, according to the similarity degree between the ontology terms. We had previously extended an Argumentation Framework, namely Value-based Argumentation Framework (VAF)[3], in order to represent arguments with *confidence* with discrete values.

Using a *threshold* for the *strength* we can reduce the set of mappings and adjust the values of *precision*. In a scenario where the mappings must be defined on the fly (i.e., web systems involving agent communication), the precision is preferred than the recall. On the other, when the mapping system is used to help users in the mapping process, it is interesting reduce the set of mappings, what cannot be done when using the discrete classes. Moreover, the *strength* as a continuous value is more expressive than the discrete classes, especially when dealing with compound terms.

We evaluated the use of *strength* against the previous discrete classes using three groups of ontologies. The results are promising, especially what concerns *precision*.

In the future, we intend to develop further tests considering a benchmark of ontologies[6]; verify the impact of using only *strengths* in our model; and use the mapping as input to an ontology merge process in the question answering domain.

References

1. Amgoud, L., Cayrol, C.: On the acceptability of arguments in preference-based argumentation. In: 14th Conference on Uncertainty in Artificial Intelligence (UAI 1998), San Francisco, California, juillet 1998, pp. 1–7. Morgan Kaufmann, San Francisco (1998)
2. Bailin, S., Truszkowski, W.: Ontology negotiation between intelligent information agents. The Knowledge Engineering Review 17(1), 7–19 (2002)

[6] http://oaei.ontologymatching.org/

3. Bench-Capon, T.: Persuasion in practical argument using value-based argumentation frameworks. Journal of Logic and Computation 13, 429–448 (2003)
4. van Diggelen, J., Beun, R., Dignum, F., van Eijk, R., Meyer, J.C.: Anemone: An effective minimal ontology negotiation environment. In: Proceedings of the Fiftheen International Conference on Autonomous Agents and Multi-Agent Systems, pp. 899–906 (2006)
5. Do, H.H., Rahm, E.: Coma - a system for flexible combination of schema matching approaches. In: Proceedings of the 28th Conference on Very Large Databases, pp. 610–621 (2002)
6. Dung, P.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n–person games. Artificial Intelligence 77, 321–358 (1995)
7. Euzenat, J., Shvaiko, P.: Ontology matching. Springer, Heidelberg (2007)
8. Gangemi, A., Pisanelli, D.M., Steve, G.: A formal ontology framework to represent norm dynamics. In: Congreso Internacional de Culturas y Sistemas Jurídicos Comparados (2005)
9. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: S-match: an algorithm and an implementation of semantic matching. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 61–75. Springer, Heidelberg (2004)
10. Gruber, T.R.: Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In: Guarino, N., Poli, R. (eds.) Formal Ontology in Conceptual Analysis and Knowledge Representation, Deventer, The Netherlands. Kluwer Academic Publishers, Dordrecht (1993)
11. Hakimpour, F., Geppert, A.: Resolving semantic heterogeneity in schema integration: an ontology approach. In: Proceedings of the International Conference on Formal Ontology in Informational Systems, pp. 297–308 (2001)
12. Laera, L., Blacoe, I., Tamma, V., Payne, T., Euzenat, J., Bench-Capon, T.: Argumentation over ontology correspondences in mas. In: Durfee, M., Yokoo, E.H. (eds.) Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (2007)
13. Laera, L., Tamma, V., Euzenat, J., Bench-Capon, T., Payne, T.R.: Reaching agreement over ontology alignments. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 371–384. Springer, Heidelberg (2006)
14. Levenshtein, I.: Binary codes capable of correcting deletions, insertions an reversals. In: Cybernetics and Control Theory (1966)
15. Levenshtein, V.: Binary Codes Capable of Correcting Deletions and Insertions and Reversals. Soviet Physics Doklady 10(8), 707–710 (1966)
16. Madhavan, J., Bernstein, P., Rahm, E.: Generic schema matching with cupid. In: Proceedings of the Very Large Data Bases Conference, pp. 49–58 (2001)
17. Maedche, A., Motik, B., Silva, N., Volz, R.: Mafra - a mapping framework for distributed ontologies. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) EKAW 2002. LNCS, vol. 2473, pp. 235–250. Springer, Heidelberg (2002)
18. Maedche, A., Staab, S.: Measuring similarity between ontologies. In: Proceedings of the European Conference on Knowledge Acquisition and Management, pp. 251–263 (2002)
19. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB 10, 334–350 (2001)
20. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. Technical report, Informatica e Telecomunicazioni, University of Trento (2004)

21. Silva, N., Maio, P., Rocha, J.: An approach to ontology mapping negotiation. In: Proceedings of the K-CAP Workshop on Integrating Ontologies (2005)
22. Stoilos, G., Stamou, G., Kollias, S.: A string metric for ontology alignment. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 624–637. Springer, Heidelberg (2005)
23. Tamma, V., Wooldridge, M., Blacoe, I., Dickinson, I.: An ontology based approach to automated negotiation. In: Proceedings of the IV Workshop on Agent Mediated Electronic Commerce, pp. 219–237 (2002)
24. Trojahn, C., Quaresma, P., Vieira, R.: A cooperative approach for composite ontology mapping. LNCS Journal of Data Semantic (to appear, 2007)
25. Trojahn, C., Quaresma, P., Vieira, R.: An extended value-based argumentation framework for ontology mapping with confidence degrees. In: Fourth International Workshop on Argumentation in Multi-Agent Systems (ArgMAS 2007). Workshop at International Conference on Autonomous Agents and Multiagent Systems (AAMAS) (2007)

Contextual Extension with Concept Maps in the Argument Interchange Format

Ioan Alfred Letia and Adrian Groza

Technical University of Cluj-Napoca
Department of Computer Science
Baritiu 28, RO-400391 Cluj-Napoca, Romania
{letia,adrian}@cs-gw.utcluj.ro

Abstract. In our approach of argumentation we focus on formalizing the context of arguments and its propagation within the argumentation chain, aiming to facilitate the re-usability of arguments in the World Wide Argument Web. The contextual extension is based on intensional operators used to update the context for different arguments. We extend the ontology of the Argument Interchange Format with context nodes and visualize the arguments as concept maps.

1 Introduction

We are in the age when we can imagine a World Wide Argument Web (WWAW) infrastructure, native to the Internet, enhancing software agents with the ability to debate, rise argumentation, or analyze ideas, in order to provide a more effective dissemination of the information to the more and more knowledge driven, but lost, human agents.

The WWAW [1] is a large scale network of inter-connected arguments created by human agents in a structured manner. From the idea of integrating structured argumentation within the WWW [2], the current vision aims to create an infrastructure for mass-collaborative editing of structured arguments in the style of the Semantic Wikipedia. One desiderata of the WWAW is to employ a unified argumentation ontology that can be extended [1].

The current trend consists in developing hybrid approaches that combine the advantages of formal (logic-based) and informal (argumentation schemes-based, diagramming reasoning) ideas [3]. Among the variety of prototype systems that support argumentation: Rationale [4], Araucaria [5], Carneadas [6], Reasonable, Magtalo[1], Aver[2], Compendium[3], none seem to overcome a minimum number of users. The above approaches aim to simplify the argumentation process by providing graphical representations and by hiding irrelevant or improbable information. During the inference process, this information is left aside and is no longer accessible in a later stage.

[1] MultiAgent Argumentation, Logic and Opinion at www.arg.computing.dundee.ac.uk
[2] Argument visualization for evidential reasoning.
[3] http://compendium.open.ac.uk

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 72–89, 2009.

Consequently, this research focuses on formalizing the context of arguments and its propagation within the argumentation chain, aiming to facilitate i) the re-usability of arguments in WWAW and ii) identifying inconsistencies between consequents and the actual contexts or world state. The vision is that the resulting context-aware argument networks will make use of the Argument Interchange Format (AIF) ontology for developing large scale argumentation networks.

2 Aspects of Arguments

The acceptance of an argument is a combination of intrinsic and extrinsic factors. A successful argument in a context might have no relevance in another one. Consider a chess player who browses opening collections, trying to figure out the best move for a particular board position. The collection provides a trace of moves (the structure or *form* of the argumentation chain), and a value estimating the current position[4] (the *content* of the argument). Happily, during one tournament, he meets a known position and he recalls the best recommended move. Facing this situation, a wise player takes into consideration three contextual factors (*context* of the argument):

1. *Social context*: the re-usability of the move depends on the knowledge about the current opponent: "If I make that move, I will enter in an "open position", and my opponent loves such positions."
2. *Intentional context*: the re-usability of the move is influenced by the current goal: "If I re-use the move, I will get some advantage, but the position becomes unstable. My goal in this game is to obtain a tie, therefore I should better consider other options."
3. *Dialectical context*: the re-usability depends on the time available: "After this move the position will become very complex. My remaining time is less than that of my opponent and I cannot afford it in this situation."

The multifaceted argument is modeled by three vectors: form, content and context.

Form. The form reveals the structure of the argument: the layout and the link between reasons and conclusion [7]. Analyzing the form shows if the premises are capable of supporting a justified conclusion, on the assumption that the antecedents are true. If the structure of the argument is weak the argument will be weak too. If there is a high reliance on the form of the argument, the argument is called strict: whenever the premises are true, so is the conclusion. Strict arguments are associated with the analysis of concepts or epistemic knowledge. If this is not the case, we have defeasible arguments: the link between premises and conclusion is weak; therefore it can be attacked with undercutting defeaters [8].

The newly proposed AIF ontology [9] focuses on the representation of the *argument form*. Patterns of arguments come in different shapes and we find two approaches: logic based (modus ponens, defeasible modus ponens, modus tollens,

[4] A qualitative one, as "white ahead" in chess algebraic notation, or a quantitative one, a subunit number computed by chess programs.

abductive arguments, inductive arguments), and a more informal one, given by argumentation schemes (presumptive, inductive, or defeasible argumentation schemes [10]).

Content. The analysis of the content of an argument deals with two issues: i) it reveals if the premises are actually true, and ii) it assures that the set of antecedents are semantically coherent [7]. If the form of the argument encapsulates common patterns of human reasoning, the content of the argument is domain dependent: to establish the degree of truth of the premises requires knowledge of the domain[5].

Regarding the first issue, the degree of support (*dos*) assigned to an argument can be expressed either qualitatively or quantitatively. To evaluate the degree of belief in an argument a flattening function is necessary to aggregate different representations of the reliance upon subarguments [2]. Some inference engines for computing the acceptability of arguments have been developed in the ASPIC project[6]. To prove the AIF concepts in this prototype, each node has a degree of support ($dos \in [0, 1]$) attribute. Also, the computation of the *dos* of a conclusion based on the *dos* of its premises is based on the weakest link principle[7]. In the large-scale, open context of WWAW, these attributes might not suffice due to: i) standards of evaluating arguments are domain dependent; ii) the applicable principle of inference for computing the reliance on an argument may change during the course of argumentation. iii) the applicable principle of inference depends on the current context; iv) different principles require different attributes attached to the premises (instead of the degree of support) such as fuzzy numbers or rough intervals.

Regarding the second issue, a standard of thematic coherence must be defined in order to validate the content of an argument. Arguments usually contain questionable premises: "is the probability of a premise so high?", "is the source that posted the argument reliable?". Critical questions on the argumentation scheme model deal with the content analysis by questioning the truth or the semantic coherence of the premises.

Context. Arguments are conveyed for a particular purpose in the context of an action. In order to effectively support a consequent, flexible control must be exercised over the extrinsic factors by providing a context [7]. The context helps agents to discover the available means of persuasion for the current debate. The success of an argument-based agent in WWAW regards its ability to re-use arguments by changing their context[8]. The following contextual dimensions can be formalized for a general argument.

[5] With the exception of *tautologies*, where the truth depends only on the form, regardless of the content. On the contrary, some fallacious reasoning, or arguments with bad form, can be perfectly acceptable in specific contexts.

[6] http://www.argumentation.org

[7] The *dos* of the consequent is the minimum degree of support of its antecedents.

[8] To re-use arguments is certainly an easier task for a software agent than to create them from scratch. The re-use of arguments would be equivalent to re-creating them in a different context.

Dialectical Context. It refers to the discourse or the debate protocol in which the arguments have been conveyed. The communication context formalizes the participants (IDs, roles such as pro, con, persuader, buyer, seller), the topic of the dialog (useful when searching arguments in WWAW), or the type of dialog (persuasion, negotiation, dispute resolution, interview[9]). The last issue opens the perspective of developing protocol-based reasoning agents in WWAW.

Intentional Context. Usually, the utterance of an argument serves in achieving a goal during the debate, negotiation, or persuasion protocol in which the argumentation takes place. The intentional context models the relationship between the specific arguments and the plans of the arguers [7]. Thus, a good argument is one which fits the current goal of the arguer. Providing an intentional context representation helps to mediate a debate by accepting only the relevant arguments.

Social Context. It encapsulates the human factors related to the context, or agent attitudes and strategies in the case of interacting software entities. The human factors might refer to: information on the user (knowledge of habits, emotional state), social environment (co-location of friends, social interaction), cultural issues (e.g. acquisition of context), relationship between the specific arguments and the plans of the arguers. The context of an argument can be seen as representing subjective perspectives on the argument.

3 Extending the Argument Interchange Format

Two extensions of the AIF ontology are: Argument Schemes [1], and Protocol Interaction Application Nodes [11]. The first one enhances agents with both reasoning capabilities: logic based argumentation and scheme based argumentation, and it also focuses on representing the *form of an argument*. The second one allows agents to represent the dialectical part of arguments.

We introduce a new node type, *context node* ($CO-node$), arguably needed since context exists independently of any object. One context may be used to evaluate different arguments, while the same argument can be evaluated in different contexts. The separation of the argument structure, modeled with *I-nodes* and *Scheme-nodes*, from contexts, provides more power to the re-use of arguments, and flexibility in the representation and acceptance [12].

Definition 1. *The extended-AIF ontology has five disjoints sets of nodes:* N_I, N_S, N_{PIA}, N_F, *and* N_{CO}.

- *An information node* $I-node \in N_I$ *represents passive information of an argument such as: claim, premise, data, locution, etc.*
- *A scheme node* $S-node \in N_S$ *captures active information or domain-independent patterns of reasoning. The schemes are split in three disjoint sets, whose elements are: rule of inference schemes* ($RA-node$), *conflict application node* ($CA-node$), *preference application node* ($PA-node$).

[9] It can point to a more elaborate dialog topology.

- *Forms of arguments $f \in N_F$ model argumentation schemes, by defining the premise descriptor, the conclusion descriptor, presumptions and exceptions.*
- *Protocol interaction nodes ($PIA-node$) are used to constrain the dialog moves within an argumentation process.*
- *Context application nodes ($CO-nodes \in N_{CO}$) are used to capture the context of the above node types in order to increase the re-usability of arguments in WWAW.*

$RA-nodes$ are used to represent logical rules of inference such as modus ponens, defeasible modus ponens, modus tollens. $CA-nodes$ represent declarative specifications of possible conflicts. $PA-nodes$ allow to declaratively specify preferences among evaluated nodes. $F-nodes$ focus on the form aspect of arguments by allowing the introduction of argumentation schemes in the AIF ontology. A $PIA-node$ encodes the range of possible speech acts as reply to an *I-node* of type locution, and their preconditions and effects [11]. In WWAW, a mediator agent deploys $PIA-nodes$ for dialog representation accessible to the participating agents. When dealing with such a node, one can either i) use this node, by providing $I-nodes$ encapsulating the speech acts specified into the $PIA-node$, or ii) attack the node by instantiating a scheme node having the $PIA-node$ as conclusion.

Definition 2. *An argument map Θ in AIF is a directed graph consisting of a set N of nodes and a binary relation $\overset{edge}{\longrightarrow}$: $N \times N$ representing edges, where $\nexists(i,j) \in \overset{edge}{\longrightarrow}$, where both $i, j \in N_i$.*

The informal semantics of the edges from a CO-node to the existing nodes of the AIF-core ontology is:

- to an *I-node*: apply a context to the data in the I-node;
- to an *RA-node*: apply a context to the inference application in the RA-node;
- to a *CA-node*: apply a context to the conflict application in the CA-node;
- to a *PA-node*: apply a context to the preference application in the RA-node;
- to a *PIA-node*: apply a context to a move in the dialog in the PIA-node;
- to an *F-node*: apply a context in relation to the presumptions in the F-node.

The inverse relation, from the nodes of the AIF ontology to a CO-node, is:

- from an *I-node*: I-node data is used to apply a context;
- from an *RA-node*: infer a conclusion in the form of a context application;
- from a *CA-node*: apply a conflict definition to the context application in the CO-node
- from a *PA-node*: apply a preference on a context application;
- from a *PIA-node*: apply a dialog move on the context application in the CO-node.
- from an *F-node*: apply an argumentation scheme on a context application in the CO-node.

4 Context Calculus in the Extended AIF

4.1 Context Representation

We approach the context issue by using the intensional programming paradigm, which has its foundations in intensional logic. Intensional logic adds *dimensions* to logical expressions and *intensional operators* are used to navigate in the context space. Consider the following *I-node* of type claim:

$$I - node_1 : \text{"}This\ year\ the\ acceptance\ rate\ of\ this\ conference\ is\ 25\%.\text{"}$$

The claim is intensional because its truth value (or content) depends on the context in which it is evaluated. Two intensional operators in the $I - node_1$ are "this year" and "this conference", which refer to two contextual dimensions: *time* and *conference*. One extension $I' - node_1$ is illustrated in table 1 where the content of the claim depends on the year and conference name, and it is represented as boolean values.

Table 1. Extension of the claim $I - node_1$

	AAMAS	*IAT*	*ECAI*
2008	True	True	True
2007	True	False	False
2006	False	False	True

The context is defined as a subset of finite union of relations [12], where *DIM* represents dimension names and the function f_{dimtag} associates a tag X_i with each $D_i \in DIM$.

Definition 3. *A context C, given DIM and* $f_{dimtotag}$*, is a finite subset of* $\bigcup_{i=1}^{n} P_i$*, where* $P_i = d_i \times f_{dimtotag}(d_i), 1 \leq i \leq n$*. The degree of context C is* $|\Delta|$*, where* $\Delta \subset DIM$ *represents the dimensions that appear in C. A context C is simple if* $(d_i, x_i), (d_j, x_j) \in c \Rightarrow d_i \neq d_j$*. A simple context of degree 1 is called a micro context.*

4.2 Context Application Schemes

We formalize context operators as rule application schemes in AIF, with the following operators [12]:

- Set schemes: *difference* $\ominus$, *conjunction* $\sqcap$, disjunction $\sqcup$;
- Selectors: *projection* $\downarrow$, *hiding* $\uparrow$;
- Constructors: *cons* $[\_:\_]$, used to create a micro context, *enrich* $\otimes$, reduce $\div$;
- Predicates: *comparison* $=$;
- Change schemes: *override* $\oplus$, *substitute* $/$, *choice* $|$ (accepts a finite number of contexts and nondeterministically returns one of them).

Protocol application nodes can be used to define precedence rules for all these operators. In the case of the override operator (figure 1), if c_1 is a context with

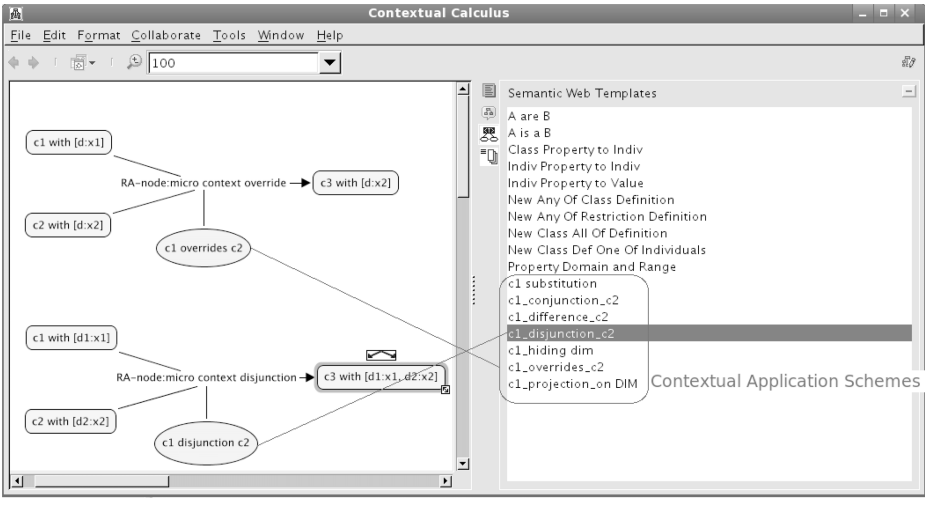


Fig. 1. Override and disjunction operators as RA-nodes in a micro context

dimension d and tag x_1, c_2 is a context with the same dimension d and tag x_2, the resulting context c_3 will have the tag x_2 associated with the dimension d. Accepting the conclusion of an $F - node$, the context of that conclusion is enriched with the presumptions assumed during the inference process. In the case of the disjunction operator, the resulting context c_3 will incorporate both dimensions and the corresponding tags ([d1:x1, d2:x2]) of the input contexts.

Example: Consider two contexts:

$$c_1 = [year : 2008, year : 2007, conf : AAMAS]$$

$$c_2 = [year : 2008, conf : AAMAS, location : Estoril]$$

and the dimension set DIM={conf, location}. Applying the contextual operators we obtain:

- c_2 overrides c_1: $c_1 \oplus c_2$=[year:2008, conf:$AAMAS$, location:Estoril];
- c_1 difference c_2: $c_1 \ominus c_2$=[year:2007];
- c_1 conjunction c_2: $c_1 \sqcap c_2$=[year:2008, conf:$AAMAS$];
- c_1 disjunction c_2: $c_1 \sqcup c_2$=[year:2008, year:2007, conf:$AAMAS$ location: Estoril];
- c_1 projection on DIM: $c_1 \downarrow D$=[conf:$AAMAS$];
- c_1 hiding DIM: $c_1 \downarrow D$=[conf:$AAMAS$];
- c_2 substitution [conf:IAT, location:Sydney]: $c_1/$ [conf:IAT, location:Sydney]: [year:2008, conf:$AAMAS$, location:Sydney].

5 Visualizing AIF Arguments in Concept Maps

As concept maps provide intuitive visualizations of the argument networks, we have chosen CMap servers[10] to provide robust displays of the arguments in the WWAW. When an argument map is saved on a CMap server, a web page version is also stored. A WWW browser is therefore sufficient to browse the argumentation chains.

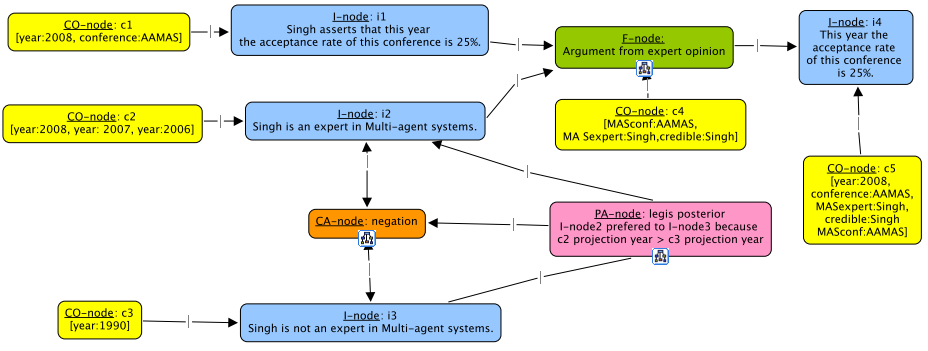


Fig. 2. Contextual Argument in CMaps

5.1 Example of Contextual Form Node

Two premises are defined, in the argument map illustrated in the figure 2, by the $I - node_1$ and $I - node_2$ concepts. $I - node_1$ has two explicit intensional operators, captured by the context node c_1.

$$c_1 = [year : 2008, conference : AAMAS]$$

$I - node_2$ has a hidden context time, captured by the context node:

$$c_2 = [year : 2008, year : 2007, year : 2006]$$

stating that the claim in $I - node_2$ is known to be true in the years 2008, 2007, and 2006. Based on the presumptive $F - node_{ArgumentFromExpertOpinion}$, the claim may be plausibly taken to be true. The $F - node_{ArgumentFromExpertOpinion}$ points to its structural representation as conceptual map (figure 3). Anyone who wants to inspect or attack it can browse its presumptions or expected exceptions. Every $F - node$ has its own formula to propagate the context to the conclusions. In the case of an *Argument from Expert Opinion*, the context of the major premise is enriched with the assumed presumptions, using the relation *enriches context* (figure 3).

$$c_4 = c_1 \sqcup [MASexpert : Singh, credible : Singh]$$

[10] http://cmap.ihmc.us/

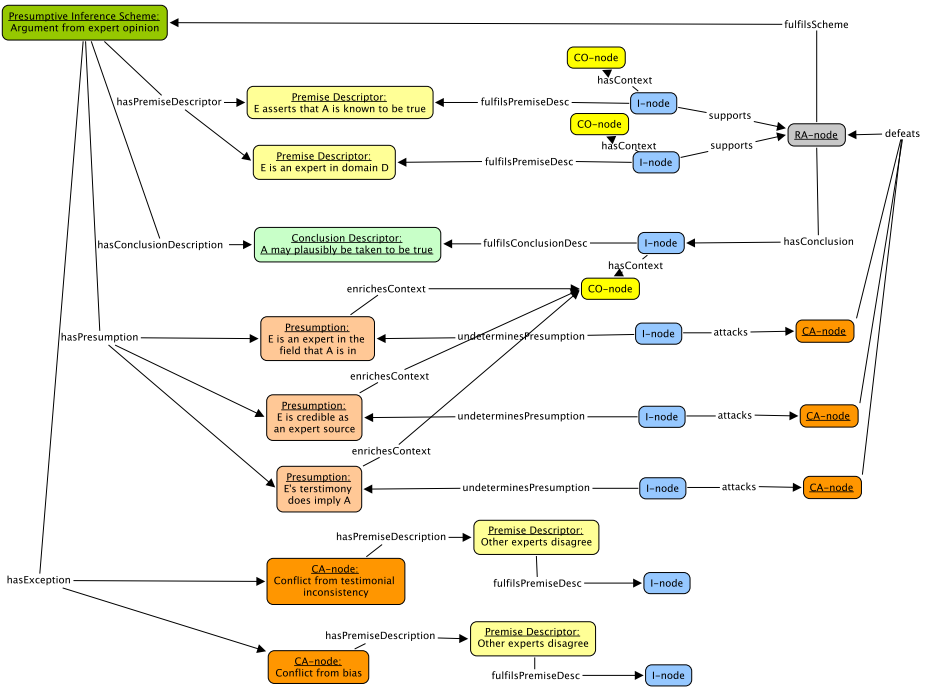


Fig. 3. F-node: *Argument from expert opinion* enriched with context, in CMaps

By applying the $CO - node_4$ in this scheme some presumptions are stated explicitly ("Singh is an expert in MAS" and "Singh is credible"), but presumptions that have not been initially encapsulated in the F-node, as "AAMAS and IAT are MAS conferences", may also be added.

$$c_4 = c_4 \sqcup [MASconf : AAMAS, MASconf : IAT]$$

As these presumptions are usually domain dependent, the context-node is used to represent them. Suppose that the claim in i_3 is identified on the Internet. Therefore, the conflict application node *negation* may be applied to represent the rebuttal attacking relation between nodes i_2 and i_3 (figure 2).

One question regards how contextual information may impact argument reusability. Depending on the context dimensions of the data in $I - node_3$ that the arguer can obtain, the specific preference application criteria ($PA - node$) can be used to resolve the conflict. The difference between the contexts of conflicting nodes is useful when searching for proper preference criteria.

$$c_3 \ominus c_2 = [year : 1990]$$

In this case, the difference refers to the time dimension, and is also relevant to the content of the argumentation chain. Therefore, the preference application node

legisposterior can be used (figure 2). The "Legis Posterior" principle stipulates that the last known data or norm dominates. By comparing the time dimension of both conflicting nodes, using formulas of the context calculus:

$$c_1 \downarrow year > c_3 \downarrow year$$

the i_2 is considered to have its content true. By clicking on the $PA - node_{LegisPosterior}$, a new concept map will be opened, revealing its structure.

When we want to re-use the above argument for another multi-agent system conference, we enrich the $CO - node_1$ context:

$$c_1 \sqcup [conf : IAT] = [year : 2008, conf : AAMAS, conf : IAT]$$

Given that a context c_5=[conf:IAT] has been defined, the override operation:

$$c_1 \oplus c_5 = [year : 2008, conf : IAT]$$

might also be used.

5.2 Example of Contextual Protocol

In the chess example considered in section 2 (figure 4), the first move $e4$ encapsulated in the data node $I - node_1$ has the contextual information:

$$c_1 = [player : DeepBlue, goal : 1/2 - 1/2, time : 2.00h]$$

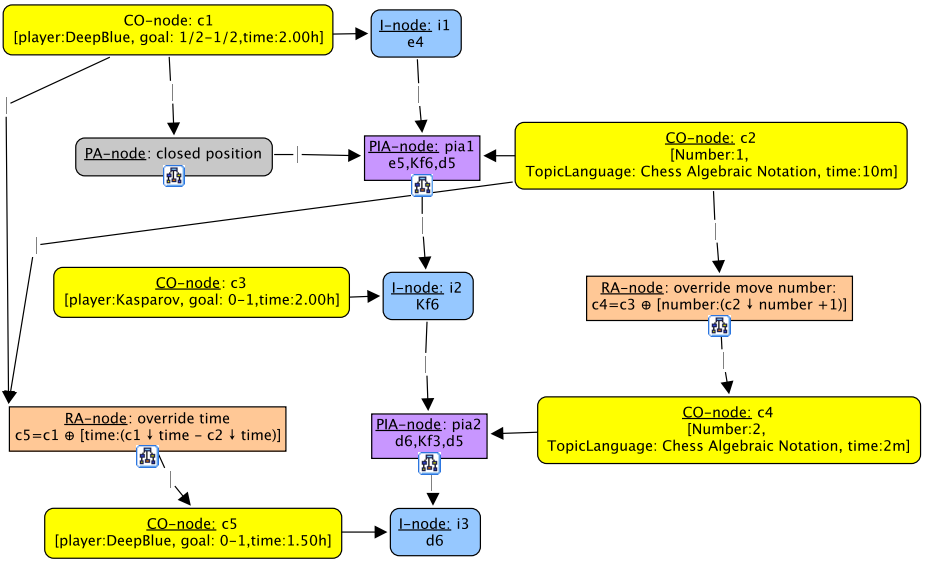


Fig. 4. Contextual protocol

which encapsulates the social context of the game (*player* dimension), the intentional context (the *goal* is to obtain a tie), and the dialectical context (available *time* to end the protocol). It points to the protocol application node pia_1 recommending the best (according to a preference criterion $PA - node_1$) possible moves in the current state (figure 4). The preference criterion *closed positions* depends on the current context. The PIA_1 node has a context denoting the topic language and the elapsed time.

$$c_2 = [TopicLanguage : ChessAlgebraicNotation, time : 10m]$$

From the available locutions, the opponent chooses *Kf6* in the node i_2. This passive information has a different context attached:

$$c_3 = [player : Kasparov, goal : 0 - 1, time : 2.00h]$$

where both the social aspect and the intentional context have been changed. Observe that the context attached to a PIA-node denotes objective information, while the context attached to the locutions in I-nodes has a subjective perspective on the game. Using the available contextual information, the context can be updated by instantiating the rule application node *override time*. Thus, the *time* dimension of the context attached to the next move will be overridden by the remaining time, the difference between the available time (in c_1) and the elapsed time (in c_2).

$$c_5 = c_1 \oplus [time : (c_1 \downarrow time - c_2 \downarrow time)]$$

The context c_4 of the protocol node PIA_2 is calculated similarly, based on the $RA - node_2$, which overrides the number of the move.

$$c_4 = c3 \oplus [number : (c_2 \downarrow number + 1)]$$

5.3 CMap Functionalities for WWAW

The following functionalities from the CMap tool can be used to visualize the WWAW architecture:

- *Deploying arguments in WWAW*: The system allows users to save their arguments on the available public servers, if the proper user name and password are provided.
- *Searching Arguments*: The CMap tool provides searching capabilities for identifying arguments within both public argument maps and the WWW.
- *Validating and fixing links*. Due to the dynamics of WWW resources, web pages having the role of supporting arguments might no longer be available. The tool can check if a chain of an argument is available at a certain time.
- *Allowing modification of argument maps*: If the proper user name and password are provided the user can modify publicly deployed argument maps, in the style of Wikipedia.
- *Public character of the arguments*: Some debates, such as Online Dispute Resolution, need to maintain some arguments as private. Even if they are posted on the WWW, only the arbitrator might have the right to read them.

– *Providing evidence:* The piece of evidence is often relevant in the course of argumentation. An argument is stronger if some evidence is provided for its premises. The system enhances parties with the ability to point towards relevant evidence in a different number of formats: video, html pages.

The resulting conceptual maps are saved in the XML format, which allows the integration of software agents within the WWAW.

5.4 Crisis Mediation

This section illustrates how a real life scenario is modeled using our approach. Engineering the conceptual argumentation maps is based on four templates (right part of figure 5): i) the basic AIF nodes (i, ra, ca, pia, pa, f, co); ii) the contextual operators nodes ($disjunction$, $conjunction$, $substitution$, $difference$, $hiding$, $overrides$, $projection$); iii) the existing argumentation schemes modeled as f-nodes (*Argument from expert opinion*, *Argument from position to know*, *Argument from reputation*, *Argument from legal rule*, etc.); and semantic web templates that facilitate interaction with software agents (*A are B*, *A is B*, *Property domain and range*, *Class property to Individual*, etc.). Using the above (extensible) templates, the process of modeling the argumentation chain is simplified for the human agent. At the same time, the IHMC_COE[11] tool that we use is able to export the argument map in the OWL format. Therefore a software agent knowing the AIF ontology will be able to reason on the existing argumentation chains.

We consider the crisis scenario that took place in 1995 between Canada and Spain. Canada, acting unilaterally to protect depleted fishing stocks, has seized a Spanish trawler just outside Canadian territorial waters. The crisis started when a Canadian fishing patrol cut the net of a Spanish trawler caught overfishing in international waters. Canada confiscated illegal nets whose mesh size were to small, so that turbot too young to spawn would be caught. Spain claimed that the seizure of the ship was a breach of international law; therefore the evidence discovered could not be used against them. The crisis has several contextual dimensions, such as legal, economical, political, environmental. The legal context is exemplified in the following paragraphs.

Assume that the starting legal context is given by the *NAFO* (Northwest Atlantic Fisheries Organization) illustrated by CO-node c_1 in figure 5. Based on the Argument from expert opinion f_1, the nets used by the Spanish trawler don't let the turbot fish to spawn (i_2). On the one hand, in the context $c_1 = [law : NAFO]$, which stipulates that the fish cannot be caught before reaching a certain age (i_3), based on the f_3 argument from legal rule, one can infer that Spain has breached the regulations (i_6). If we also consider the issue in the context of year 1995 (c_1), by applying the ra_0 contextual operator, we validate the fact that the fishing quota for EU boats is set to 34000 tones (i_4). In the context c_2, Canada claims that, according to its observations (f_2), the fishing quota has already reached 70.000 tones (i_5). This discrepancy, modeled by the

[11] http://coe.ihmc.us

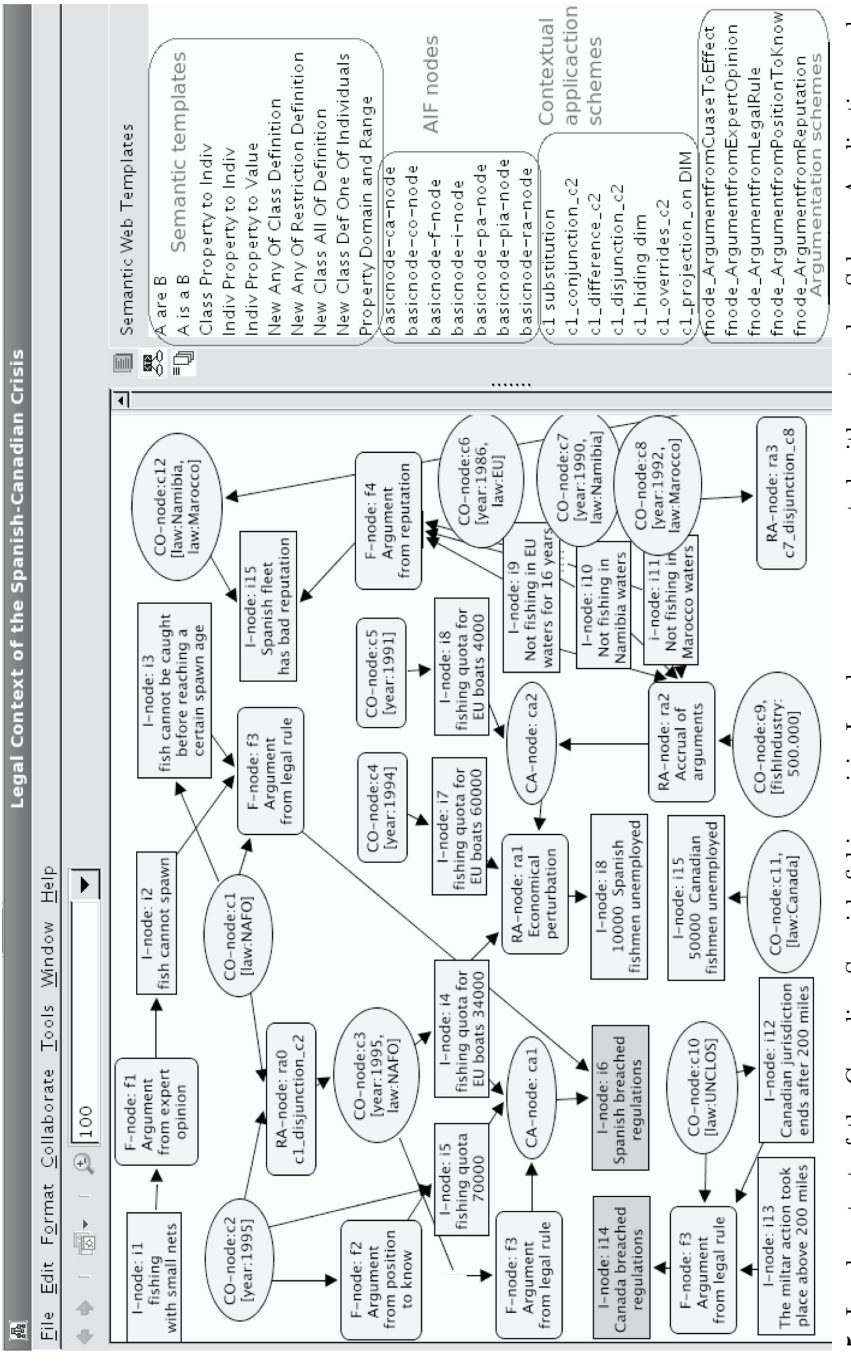


Fig. 5. Legal context of the Canadian-Spanish fishing crisis. I-nodes are represented with rectangles, Scheme Application nodes with rounded rectangles, and CO-nodes with ovals.

conflict application node ca_1 and based on the same f_3 pattern of argumentation also supports the consequent i_6. Notice that the argument from legal rule f_3 was applied in a different context c_3, opposite to its first application in the c_1 context, which exemplifies the re-use of arguments in different contexts.

The Spanish government claims that the recently (context node c_4) decreasing of fishing quota from 60000 (i_7) to 34000 tones has produced economical perturbations (ra_1), which might lead to serious labor problems in the Spanish fishing industry (i_8). If one looks at the issue from the context of 1991 (c_5), when the fishing quota was only about 4000 tones (i_8), one can find a conflict (ca_2) between Spanish claims, which defeats the application of the ra_1 scheme[12].

The Spanish advocate can enrich the legal context by the EU legislation (c_6). When Spain entered the European Union in 1986, it was not allowed to fish in European waters for 16 years (i_9). In the same line, under the Namibia law (c_7), one can find that in 1990 the Spanish boats have been kicked out from its territorial waters (i_{10}). Similar regulations (i_{11}) have been put into force by Morocco in 1992 (c_8). These correspond to the period when Spanish fleet started to over-fish in North Atlantic. In the context of 500000 employers in the Spanish fishing industry (c_9), by accrual of these perturbations (ra_2), the social argument of the Spanish government has stronger support, but it also contributes to the bad reputation of the Spanish fleet (i_{15}). Note that the Spanish fleet bad reputation is inferred only in the context c_{12} of Namibia and Morocco legislation, obtained from:

$$c_{12} = (c_7 \downarrow law) \sqcup (c_8 \downarrow law)$$

From the viewpoint of the other side, Canada has stricter regulations than *UNCLOS* (The United Nation Convention Law of the Sea), but they are applied only to its own citizens. As a consequence of these norms (c_{11}), 50000 fishermen became unemployed (i_{15}) and the Canadian government spent 3 billion Canadian dollars to assist them. Thus, both Canadian and Spanish governments face similar social problems.

On the other hand, *UNCLOS* stipulates that only 200 miles are under Canadian jurisdiction. Consequently, in this legal context c_{10}, considering that Canadian action took place after this limit (i_{13}), the Canada is the one that has breached the law. Note again, that the same argument from legal rule was reused in a different context.

One conclusion is that context is very relevant to understand and to identify the causes and solutions of such a crisis. By exporting the above structure in OWL, the software agents can analyze and contribute to the argument map.

6 Related Work

A fundamental difference between human and agent societies is that humans demonstrate some heterogeneity in their interpretation of what an argument

[12] By attacking the link between the premises i_4, i_7 and the conclusion i_8, it represents an undercutting defeater.

represents. The AIF ontology basically tries to structure argumentation without affecting this flexibility. Compared to existing work, our approach refines the flexibility provided by the AIF ontology, adding the context explicitly.

Rationale [4] is an instance of an emerging category of argumentation tools, aiming to improve argumentation abilities of human agents, based on the semi-formal concept of diagramming reasoning. The advocated advantages of the tool consists in its usability and semi-formality. Quite the contrary, this research focuses on re-usability, flexibility, and the open world assumption needed in large environments such as WWAW. In the WWAW arguments are no longer ordered sequentially or chronologically as in the discussion threads, but rather according to their functional role. The context of an argument is introduced to facilitate the composition of argument along several contextual dimensions.

The Logical Argument Mapping [13] (LAM) provides for structuring arguments a seven step methodology. The ontology of LAM maps distinguishes statements and relations. Statements have a graphical representation according to their importance for cognitive change. Contrary to the AIF ontology, relations in LAM have a fixed set of labels: therefore, opposes, refutes, rejects, questions, supports, etc. An *AU:* tag is used to identify the author of an argument. Instead, we use PIA-nodes and the dialectical context of the argument to represent the dialogical aspect of an argument.

Different types of premises are used in the Carneades Argumentation framework [6]: ordinary, presumptions, and exceptions. The context of an argument depends on the status of the claims (accepted, rejected), proof standard (preponderance of evidence, beyond reasonable doubt) and weights attached to claims. In our approach, these notions regard the content of the argument, while the context is closely related to the presumptions in Carneades.

7 Discussion

Argumentation schemes in F-node are fixed structures of inference [1] reflecting common patterns of human reasoning. In our view, they should sometimes be slightly changed in order to fit a particular case. Contextual nodes allow to extend the presumptions in a particular argumentation scheme. Therefore, unexpected rebuttal facts which attack the newly introduced assumptions can be accepted to defeat the conclusion of the scheme.

The relevant question is what makes an argument successful. The strength of its form, the truth of its content, its application in the adequate context, a combination of these aspects? How will successful arguments replicate within WWAW, as *memes*[13] for instance, is the subject of new fields of exploration. A situation in which some arguments will be preferred by humans and others by software agents is not very hard to imagine.

In order to deploy agent-based applications on the WWAW, ideas from the REST architecture style of web services [14] can be applied, by considering each

[13] Term coined by Richard Dawkins on the analogy of *gene*, to define the cultural copying unit.

argument as a web resource. The agent progresses through an argument chain by selecting links, in this case state transition, resulting in a new page transferred to the user, according to the requested form, content, or context. In this approach, the re-usability of the arguments is increased due to the *loose coupling* property of the argument networks. Also, this design reflects the fact that, in a debate, the information is revealed gradually. Depending on the form, content, or context provided at each stage of the argumentation, the new state will be computed accordingly.

According to the premises of the game theoretic approach, a rational agent cannot be persuaded. It will always choose the best action independently of persuasion attempts. In the utility-based negotiation model [15] agents do not attempt to persuade each other or to explain why the proposal should be accepted, which are not necessary in domains with complete information. The fish dispute between Canada and Spain is seen as such a domain. In our view, each crisis is characterized by gradually revealed information. Of course, we assume that a crisis is an unanticipated event, not designed by some political circles. Consequently, at the beginning of the crisis, each party takes some actions based only on partial or distorted pieces of information. The need of negotiation itself is questioned when all the information is available: a decision system which compute the optimal outcome will suffice.

Quite the opposite to repetitive disputes, such as simple e-commerce contract breaches, international disputes are characterized by high dependence on context. They depend on the political context, social context, economical context, and a very complex and multi-jurisdictional legal context. For instance, in [15] the name of the states have been hidden during experiments. Consequently, the context was deeply altered, which leads to a severe limitation of the means of negotiation and the ability of negotiators to identify new solutions during the mediation. The experiments also deviate from the real life in the sense that only one negotiator was used. During an international crisis an entire team of experts is empowered by the government to handle the issue. We argue that the argumentation approach based on collaborative editing of conceptual maps is better suited to such scenarios.

8 Conclusions

Our approach aims to refine the argumentation process, by providing a context to each claim or scheme application. The contributions of this paper are: i) extending AIF ontology with context nodes, and ii) enacting AIF ontology as concept maps.

Future work regards the representation of data in I-node within AIF. In order to be effectively used by the software agents, the facts should be available as pieces of evidence that agents can refer to. The Common Knowledge Library can be particularly useful to represent the evidence in a structural form[14]. Although Compendium/ClaiMaker [16,17] has some similarities with our work, we have

[14] http://piex.publ.kth.se/ckl/index.html

preferred to use CMap as it was easier to connect to our system, but further consideration will be paid to that research in the near future.

In the game theoretic approach each player is selfish. It aims to maximize its expected utility and it does not take into consideration equity and social welfare. Quite the opposite, the argumentation approach aims to maximize the global welfare. As in real life, it is characterized by cooperation too, and not just competition between agents.

This paper focused on simplifying the argumentation process for the human agent, but keeps enough formality to allow interaction with software agents. The future work deals with the use of the exported argument map in OWL by the AIF-based software agents.

Acknowledgments

We are grateful to the anonymous reviewers for the very useful comments. Part of this work was supported by the grant 27702-990 from the National Research Council of the Romanian Ministry for Education and Research.

References

1. Rahwan, I., Zablith, F., Reed, C.: Laying the foundations for a world wide argument web. Artificial Intelligence 171, 897–921 (2007)
2. Reed, C.: Representing and applying knowledge for argumentation in a social context. AI and Society 11, 138–154 (1997)
3. Hunter, A.: Real arguments are approximate arguments. In: 22nd AAAI Conference on Artificial Intelligence, pp. 66–71 (2007)
4. van Gelder, T.: Rationale: Making people smarter through argument mapping. Law, Probability and Risk (submitted, 2007)
5. Reed, C., Rowe, G.: Araucaria: Software for argument analysis, diagramming and representation. International Journal on Artificial Intelligence Tools 13, 961–979 (2004)
6. Gordon, T.F., Prakken, H., Walton, D.: The Carneades model of argument and burden of proof. Artificial Intelligence 171, 875–896 (2007)
7. O'Rourke, M.: Critical Thinking Handbook. University of Idaho (2005)
8. Pollock, J.L.: Defeasible reasoning with variable degrees of justification. Artificial Intelligence 133, 233–282 (2001)
9. Chesnevar, C., McGinnis, J., Modgil, S., Rahwan, I., Reed, C., Simari, G., South, M., Vreeswijk, G., Willmott, S.: Towards an argument interchange format. The Knowledge Engineering Review 21, 293–316 (2006)
10. Reed, C., Walton, D.: Towards a formal and implemented model of argumentation schemes in agent communication. Autonomous Agents and Multi-Agent Systems 11, 173–188 (2005)
11. Modgil, S., McGinnis, J.: Towards characterising argumentation based dialogue in the argument interchange format. In: Argumentation in Multi-Agent Systems, May 2007, Hawai US (2007)
12. Alagar, V.S., Paquet, J., Wan, K.: Intensional programming for agent communication. In: Leite, J., Omicini, A., Torroni, P., Yolum, p. (eds.) DALT 2004. LNCS, vol. 3476, pp. 48–56. Springer, Heidelberg (2005)

13. Hoffmann, M.: Logical argument mapping: a cognitive-change-based method for building common ground. In: 2nd International Conference on the Pragmatic Web, Tilburg, The Netherlands, pp. 41–47 (2007)
14. Fielding, R.T., Taylor, R.N.: Principled design of the modern web architecture. In: 22nd International Conference on Software Engineering, pp. 407–416. ACM, New York (2000)
15. Kraus, S., Hoz-Weiss, P., Wilkenfeld, J., Andersen, D.R., Pate, A.: Resolving crises through automated bilateral negotiations. Artificial Intelligence 172, 1–18 (2008)
16. Buckingham Shum, S.: Hypermedia discourse: Contesting networks of ideas and arguments. In: Priss, U., Polovina, S., Hill, R. (eds.) ICCS 2007. LNCS, vol. 4604, pp. 29–44. Springer, Heidelberg (2007)
17. Uren, V., Buckingham Shum, S., Bachler, M., Li, G.: Sensemaking tools for understanding research literatures: Design, implementation and user evaluation. International Journal of Human-Computer Studies 64, 420–445 (2006)

Command Dialogues

Katie Atkinson[1], Rod Girle[2], Peter McBurney[1], and Simon Parsons[3]

[1] Department of Computer Science, University of Liverpool, UK
{K.M.Atkinson,mcburney}@liverpool.ac.uk
[2] Department of Philosophy, University of Auckland, New Zealand
r.girle@auckland.ac.nz
[3] Department of Computer and Information Science, Brooklyn College, New York, USA
parsons@sci.brooklyn.cuny.edu

Abstract. We propose a representation of imperatives in computational systems, and a multi-agent dialogue protocol to argue over these. Our representation treats a command as a presumptive argument for an action to be executed by a designated agent, together with a set of associated critical questions whose answers may defeat the presumption. The critical questions enable the identification of attacks on the uttered command, and so can be used to specify a dialogue game protocol for participants to argue over the command. We present a formal syntax for part of the protocol, called **CDP**, and outline denotational semantics for both commands and for the protocol.

Keywords: agent communications, argument schemes, commands, dialogue games, imperatives, interaction protocols, semantics.

1 Introduction

Computational processes may malfunction or they may interfere with the successful operation of other processes, whether intentionally or not. If a process in a single, centralized computer system malfunctions, the thread that is in overall control of the system can de-activate or delete the malfunctioning process. In a computer system with multiple threads of control, such as a distributed e-commerce system, there may be no central thread with the power to de-activate or delete malfunctioning processes on remote machines; at best, a co-ordinating thread may instruct the relevant remote thread controlling a malfunctioning process to de-activate or delete it. If the remote entities in the distributed system are autonomous, such an instruction may or may not be obeyed. If these remote entities are intelligent, they may also question or seek justification for an instruction prior to deciding whether or not to obey it. Humans do this constantly, even in organizations with strong hierarchical command structures, such as the military.

How should such instructions be formally represented and issued? And once issued, how should they be questioned or challenged, and how justified? Girle [10] presented the syntax of a dialogue protocol, **IDL3**, which enables arguments over commands. The protocol was based on a Hamblin-style dialogue game, **DL3** (also defined in [10]), with additional locutions for issuing and responding to instructions. If challenged, an instruction may be justified by reference to some proposition (indicating some fact

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 93–106, 2009.

about the world), and/or by reference to some further action which the commanded action is intended to enable. For many applications, such as the computer-aided instruction systems to which dialogue games were first applied [4,19], this level of granularity of justification may be sufficient, especially if participants share over-arching goals. However, multi-agent computational applications typically involve participants with possibly-conflicting beliefs, goals, and even values.[1] We therefore believe a more finely-grained approach is required, in order that a dialogue over commands is better able to elucidate the differences between participating agents. Moreover, [10] gives no semantics for **IDL3**, although an axiomatic semantics defined in terms of the pre- and post-conditions of dialogue utterances would not be difficult to define.

What makes command utterances different to other locutions regarding actions, such as proposals, promises, requests or entreaties? Firstly are the pre-conditions of the utterance: the valid utterance of a command pre-supposes the existence of a regulatory environment or a social context in which one party (the speaker) has some right to require another party (the hearer) to perform some action at the behest of the first party, something Habermas has called the *normative rightness* of a directive [11]. Such an environment or context, if it exists, creates a *prima facie* obligation on the hearer to obey any legally-issued commands by the speaker. Of course, whether such a context actually pertains between the parties may be a matter of disagreement between them, and thus itself open to question and challenge. A protocol for multi-party dialogues over commands should be sufficiently expressive to permit such disputation. Secondly are the conditions of revocation of the utterance: normally only the issuer of a command utterance has the contextual power to retract the utterance and thus to revoke the requirement that the action stated be performed by the recipient of the command. This dialogic aspect of the semantics of a command — that revocation privileges are limited only to the initial speaker, even after a command may be accepted by a recipient — distinguishes commands from many other types of action locutions, such as promises, proposals and entreaties, as two of us explain in greater detail in [17].

Thirdly, are the possible forms of challenge to a command utterance: questioning or challenging a command may involve disagreement not only with the appropriateness, suitability or feasibility of a proposed action, as in the protocol for arguments over action of [2], but also disagreement with the normative rightness of the command utterance. As Habermas argues, *"refusing imperatives normally means rejecting a claim to power"* [11, p. 325]. Thus dialogue over commands may involve discussions over the right of the agent issuing a command to do so, or to do so at the particular time of the utterance, or to do so to the intended recipient. Any protocol for command dialogs needs, therefore, to enable arguments over such issues.

The approach adopted in this paper begins by presenting a novel formalism for representing commands, similar to the computational representation of proposals for actions given in [2]. That representation construed an action proposal as an argument scheme for practical reasoning, i.e., as a presumption for action along with a collection of relevant critical questions whose answers may defeat the presumption, following

[1] Values, as used in this sense, represent the social interests promoted through achieving the goal. Thus they are qualitative measures of the desirability or non-desirability of achieving a goal.

the account of Walton [25]. The current paper extends that representation to deal with command utterances, and identifies the specific critical questions appropriate for response to such commands. Section 2 presents our argument schemes for commands, and the associated critical questions are articulated in Section 3. These argument schemes enable the specification of a dialogue protocol, called **CDP**, in which commands can be issued, rationally questioned, challenged and justified, and accepted or rejected. The syntax for **CDP** is presented in Section 4, and we outline denotational semantics for commands and for the protocol in Section 5. Our semantic framework draws on recent work by Reed and Norman [24] formalizing Hamblin's Action-State Semantics for imperatives [12] and by two of us in the semantics of action dialogues [16,17]. The paper concludes with a brief summary of our contribution and a discussion of future work.

2 Representing Commands

We first propose a representation of commands. We assume two dialogue participants, called *Commander* and *Receiver*, respectively. After opening, a dialogue between the two begins with Commander issuing an instruction or command to Receiver to execute some action. In the spirit of Hamblin [12], we allow this instruction to either specify the action to be executed or to specify a (partial or complete) world state to be achieved as a result of successful execution of an action, or both. For simplicity, we assume that actions either successfully achieve their intended goals, or they do not, and, apart from this binary outcome, we do not represent any degree of uncertainty regarding the execution or the effects of actions. We assume that both Commander and Receiver model time as discrete, and represented by the positive integers, and that they share a common clock (and thus always agree on the current time).

Our approach treats commands as presumptive arguments for action by Receiver, represented as argument schemes with an associated list of critical questions, as [2] does for action proposals. Accordingly, we have two basic representations for commands, either as an instruction to Receiver to perform at a certain time a certain action (*Command Argument Scheme for Actions, no. 1*, or *CAS-A1*) or as an instruction to Receiver to make true at a certain time a certain set of propositions describing a world-state (*Command Argument Scheme for States, no. 1*, or *CAS-S1*):

CAS-A1: Commander instructs Receiver to perform, at time t, action α.
CAS-S1: Commander instructs Receiver to bring about, at time t, state S.

An instruction issued according to CAS-S1 allows Receiver the freedom to choose whatever action Receiver believes can best achieve state S at time t. We number these schemes because each has a variant, in which Commander provides a justification for the instruction. Building on the account of [2], a justification for an instruction could include statements regarding: the current situation in which the instructed action is to be performed (represented by the symbol R in our notation below); an indication of the social context (X) enabling Commander to issue instructions to Receiver; the state of affairs (S) expected to be achieved by the performance of the instructed action (α); the goal (G) of the action, meaning the features of the state of affairs desired by Commander; the value (v) to be realized or enhanced by the achievement of the goal,

which provides a reason why those features are desirable. Thus, goals are (partial or complete) states of the world, represented by conjunctions of propositions, while values are functions over these states. As such, goals have truth values, and these values may (at least, in principle) be verified objectively, independently of the particular agents engaged in the dialogue. In contrast, values, not being propositions, do not have truth values, and only have value with reference to a particular agent or group of agents; they may be evaluated differently by different agents, or differently by any one agent at different times. The division of the consequences of an action into a set of implied logical statements (the goal G) and the impact of realization of these statements on some value v means that dialogue participants can potentially separate objective from subjective assessments of the new circumstances arising from successful performance of the action.

With such a structure, each argument scheme above has a variant (numbered 2) which provides a justification for the instruction, as follows:

CAS-A2: Given the social context X,
In the current circumstances R,
Commander instructs Receiver to perform action α at time t,
Which will result in new circumstances S,
Which will realize goal G,
Which will in turn promote value v.

CAS-S2: Given the social context X,
In the current circumstances R,
Commander instructs Receiver to bring about state S at time t,
Which will realize goal G,
Which will in turn promote value v.

Denoting Commander by Com and Receiver by Rec, we can represent these as:

CAS-A2: $[Com:\ Rec]:\ X, R \overset{\alpha,t}{\rightarrow} S \models G \uparrow v$

CAS-S2: $[Com:\ Rec]:\ X, R \rightarrow (S,t) \models G \uparrow v$

Our notation also allows for commands to be justified on the basis of some value v being *demoted*, denoted $\downarrow v$. If we allow the action notation α to include representation of negative actions (i.e., not doing something), then demotion of values also permits representation of commands in which Receiver is instructed not to do some action because doing so may be deleterious for some value or may actively promote some value. For simplicity, we ignore issues regarding relationships between the timing and durations of actions on the one hand, and the timing of goal- and state-realizations, on the other.

3 Critical Questions

The justifications for instructions given by CAS-A2 and CAS-S2 provide a basis for elaborating a set of associated critical questions. In this paper, we present only the questions for instructions under Scheme CAS-A2, and these questions fall naturally into the following categories:

- Questions regarding the selection of the action
- Questions regarding the selection of Receiver to perform the action
- Questions regarding the authority of Commander to issue the instruction to Receiver
- Questions regarding performance of the action, including questions regarding its timing.

3.1 Questioning the Choice of Action

Most of the critical questions related to the selection of the action α stated in instruction CAS-A1 or CAS-A2 are the same as the questions already articulated in [3]. We list these again here (numbered *CQA1*, ..., *CQA18*, to indicate that these are Critical Questions regarding the Action). For simplicity of presentation, we delete references to the time t specified in the command.

CQA1: Are the believed current circumstances true?
CQA2: Assuming this, will the action bring about the stated new circumstances?
CQA3: Assuming all of these, will the action bring about the desired goal?
CQA4: Does the goal promote the value intended?
CQA5: Are there alternative ways of realizing the same new circumstances?
CQA6: Are there alternative ways of realizing the same goal?
CQA7: Are there alternative ways of promoting the same value?
CQA8: Does doing the action have a side effect which demotes the value?
CQA9: Does doing the action have a side effect which demotes some other value?
CQA10: Does doing the action promote some other value?
CQA11: Does doing the action preclude doing some other action which would promote some other value?
CQA12: Is the action possible?
CQA13: Are the current circumstances as described possible?
CQA14: Are the new circumstances as described possible?
CQA15: Can the desired goal be realized?
CQA16: Is the value indeed a legitimate value?
CQA17: Do the new circumstances already pertain?
CQA18: Has the action already been performed?

The last two questions are additional to those in [3]. A reason not to obey a command is that the action being commanded has already been performed at an earlier time and the resulting world-state already brought into being. This may be a fact known to Receiver but not yet known to Commander, especially if they are situated in a distributed computational environment.

Commander may respond to any of these questions with further elaboration and justification of the instruction, or with its withdrawal. The type of justification dialogue between Commander and Receiver will differ, according to the particular critical question posed by Receiver; these different dialog types are listed in [3].

3.2 Questioning the Choice of Receiver to Perform the Action

This second category deals with questions regarding why Receiver, *in particular*, was selected by Commander to perform the action stated in the instruction. Here, we are

dealing with questions regarding availability and capability of agents other than Receiver to execute the instruction, and the criteria used by Commander in selecting Receiver. The critical questions are:

CQR1: Is there another agent available to perform the action at the stated time?
CQR2: Is there another agent with sufficient knowledge to perform the action at the stated time?
CQR3: Is there another agent with access to the resources required to perform the action at the stated time?
CQR4: Is there another agent with sufficient experience to perform the action at the stated time?
CQR5: Is there another agent with the appropriate skillset able to perform the action at the stated time?
CQR6: Is there another agent who can see to it that the action is performed at the stated time?
CQR7: Is there another agent more suitable than Receiver to perform the action at the stated time?

In responding to such questions, Commander may provide information on the expected consequences of Receiver performing the action and/or Receiver not performing the action. These consequences could be for Receiver, for Commander, for other agents in the system as individuals, or for the system as a whole. Statements by Commander of such consequences can be understood as elaborations of the Consequential State (S), the Goal (G), and the Value (v) components of CAS-A2. If the answer to any of the questions CQR1 – CQR7 is YES, then the action to be performed may be re-assigned to another agent, not Receiver. In that case, Commander will need to withdraw the instruction issued to Receiver, and issue a new instruction to the other agent. Note that answers to questions CQR1 – CQR6 may all be YES, and yet the answer to question CQR7 may be NO: Receiver may be assessed by Commander as the best, or most suitable, by some criteria, of all those agents able, willing and available to do the task.

3.3 Questioning Commander's Authority to Command Receiver

The third category of critical questions are those regarding the authority of Commander to issue an instruction to Receiver within the social context stated in CAS-A2. They deal with issues such as: the social status of Commander; the social roles of the two agents; and the social or individual consequences of obeying or disobeying orders. The critical questions that arise here are:

CQX1: Under what authority does Commander have the power to issue such an instruction to any agent?
CQX2: Under what authority does Commander issue the instruction to Receiver?
CQX3: Under what authority does Commander issue the instruction to Receiver to perform the stated action?
CQX4: Under what authority does Commander issue the instruction to Receiver to perform the stated action at the stated time?
CQX5: What are the consequences of compliance with the instruction?
CQX6: What are the consequences of non-compliance with the instruction?

Responses to these questions will depend upon the nature of the social context (denoted by X in CAS-A2), which links Commander and Receiver. As with the previous category of questions, the consequences of compliance or non-compliance with the instruction may impact upon the Receiver, the Commander, other agents in the system, and/or the system as a whole; similarly, statements by Commander of such consequences in response to Questions CQX5 or CQX6 can be understood as elaborations of the Consequential State (S), the Goal (G), and the Value (v) components of CAS-A2. Implementing computational entities able to entertain and respond to questions such as these would, of course, require a formal representation of the social context connecting the agents, as for example in the work of Karunatillake [14], or in multi-agent systems with explicitly-defined roles, rights and responsibilities, such as those developed using software engineering methodologies such as *Gaia* [27].

3.4 Questions Regarding the Performance of the Action

The final category of questions relate to the performance and timing of the action. As with the questions above, the answers to these questions may inhibit an agent's ability or make it impossible to execute an instructed action, and so may serve to defeat the command. However, these questions may also be asked by a Receiver willing to obey the command, in order to clarify the instruction. In this category are the following questions:

CQP1: By what time should Receiver complete the action?
CQP2: How long should the action take to complete?
CQP3: At what place should Receiver perform the action?
CQP4: How (by what methods) should Receiver perform the action?
CQP5: May Receiver delegate the performance of the action?
CQP6: May Receiver perform the action in concert with other agents?
CQP7: For joint actions, with whom should Receiver perform the action?

Joint actions and delegated actions lead to consideration of issues such as: the division of actions into constituent tasks; co-ordination and timing of these tasks; the willingness and availability of other agents to execute their assigned tasks; etc. For reasons of simplicity, we ignore these issues in this initial account.[2]

4 Protocol Syntax

We now present the syntax of a multi-agent dialogue protocol to enable agents to issue, accept, reject, question, challenge, justify and retract commands. We call the protocol the *Command Dialogue Protocol* (**CDP**). The syntax is based to some extent on that of the **PARMA Protocol** for arguments over proposals for action [2], extended to deal with commands and associated questions and challenges. As is standard in the agent communications literature, we represent agent utterances by a two-layer syntax: an inner, or content, layer and an outer, or wrapper, layer. The outer layer comprises locutions which express the illocutionary force of the inner content. The inner layer includes

[2] Note that the argument scheme for action proposals of [2] has recently been extended in [1] to deal with joint actions.

the following elements (expressed in suitable formal representations): a time-stamp; an identifer for the speaker of the utterance; an identifier for the intended recipient of the locution; and the conversational content of the utterance.[3]

Generic (uninstantiated) locutions are denoted with just the wrapper as, for example, in WITHDRAW(.), while instantiated locutions are denoted with both wrapper and contents shown, as in WITHDRAW($t, Ag1, Ag2$), where t is a time-stamp, and $Ag1$ is an agent identifier indicating the speaker, and $Ag2$ the intended recipient of the utterance. For simplicity, we assume there are just two participating agents, Commander and Receiver, denoted Com and Rec as before. We also assume, as in [17], that CDP contains standard control locutions for participants to initiate, enter into and withdraw from dialogues using the protocol.

After the opening, a command dialogue using CDP begins with Commander issuing a command to Receiver to perform a specified action at a specified time; Table 1, discussed below, presents the locutions available to Commander for this utterance. Following this utterance, CDP then permits Receiver to respond to the command, using the locutions in Table 2, discussed below. If Receiver issues a question or challenge to Commander in response to the command, Commander may respond in turn, using the locutions in Table 1 to answer a question, or to state or justify an aspect of the command, or to retract some aspect of the command. The next four sub-sections discuss the utterances, responses and counter-responses in more detail.

4.1 Issuing or Retracting a Command

As in Section 3, we assume argumentation scheme CAS-A2 is used to issue instructions. Table 1 shows the locutions available to Commander for stating the instruction and for stating the elements of its justification, and for retracting any of these statements after their utterance. We suppress the agent identifiers and time-stamp from these expressions, except in the case of the utterances indicating or cancelling the action to be undertaken. We also assume, for simplicity, that the value v is one promoted by the action. The time variable t specified in the following locutions refers to the start time of execution of the action, α. Note that, in this protocol, only the agent issuing an instruction has the dialectical power to retract it and thus to revoke the command it embodies.

Table 1. Locutions to issue or revoke a command, or elements of its justification

Command & Justification Locutions	Retraction Locutions
State_context(X)	Deny_context(X)
State_circumstances(R)	Deny_circumstances(R)
State_action_command(Com, Rec, α, t)	Retract_action_command(Com, Rec, α, t)
State_consequences(α, t, R, S)	Deny_consequences(α, t, R, S)
State_logical_ consequences(S, G)	Deny_logical_ consequences(S, G)
State_purpose(v)	Deny_purpose(v)

[3] We assume faultless transmission, so that the intended recipient is also always the hearer of the utterance.

Table 2. High-level locutions in response to a command

Accept_action_command(Com, Rec, α, t)
Refuse_action_command(Com, Rec, α, t)
Question_action_command(Com, Rec, α, t)
Challenge_action_command(Com, Rec, α, t)
Done_action_command(Com, Rec, α, t)
Action_done(β, s)

4.2 Responding to a Command

We now now list the high-level locutions available to Receiver to respond to an instruction from Commander. These are shown in Table 2. Receiver may accept or refuse the instruction issued by Commander, with the first or the second locutions shown there. The third locution, *Question_action_command(.)*, allows Receiver to indicate a desire to seek clarification or further information from Commander. As stated earlier, such clarification may be sought whether Receiver intends to obey or not to obey the instruction. The fourth locution, *Challenge_action_command(.)*, indicates that Receiver wishes to challenge the instruction or some aspect of its justification. These two locutions will then lead to subsequent utterances by Receiver with specific questions or challenges, as discussed in Section 4.3 below.

The locution *Done_action_command(*Com, Rec, α, t*)*, uttered by Receiver, indicates that Receiver has executed at time t the action α which was assigned by Commander in the command utterance. It may be that Receiver has only partially executed action α, or has executed it at some other time, or has executed some other action whose performance obviates the need to execute action α, or whose performance precludes the execution of α. In such cases, Receiver may respond with an appropriate instantiation of the final locution in Table 2, namely *Action_done(*β, s*)*, where β denotes some action, and s a time-point. This locution can also be used if action α or some other action β has been executed by another agent, not Receiver, and such execution obviates or precludes now the need to execute α at time t. Thus, this final locution in Table 2 permits Receiver to make an initial response to an instruction in the case where the commanded action does not need to be undertaken.

4.3 Questioning or Challenging a Command

The content of the various questions and challenges available to Receiver in response to an instruction issued by Commander are those listed in Sections 3.1 to 3.4. For each of the questions listed there, we may specify two associated dialogue locutions, in a similar manner to that adopted for the specification of the **PARMA Protocol** for the first 16 critical questions of Section 3.1 [2]. One of these two dialogue locutions is intended to seek from Commander further information or a justification of the relevant issue. The other locution is intended to deny a justification provided by Commander of the relevant issue. Consider, for example, critical question CQR1 in Section 3.2, which asks if there is another agent available to perform the action stated in the command at the time specified. Receiver may respond with a question to Commander, *asking* if there

Table 3. Locutions to question or attack the choice of agent to perform the action

CQ	Locutions to question the choice of agent	Locutions to challenge the choice of agent
CQR1	Ask_if_other_agents_available(.)	Deny_no_other_agents_available(.)
CQR2	Ask_if_other_agents_with_knowledge(.)	Deny_no_other_agents_with_knowledge(.)
CQR3	Ask_if_other_agents_with_resources(.)	Deny_no_other_agents_with_resources(.)
CQR4	Ask_if_other_agents_with_experience(.)	Deny_no_other_agents_with_experience(.)
CQR5	Ask_if_other_agents_with_skillset(.)	Deny_no_other_agents_with_skillset(.)
CQR6	Ask_if_other_agents_can_see_to_it(.)	Deny_no_other_agents_can_see_to_it(.)
CQR7	Ask_if_other_agents_more_suitable(.)	Deny_no_other_agents_more_suitable(.)

is another agent available to perform the stated action at the stated time. Or, Receiver may respond with a challenge to Commander, *denying* the assertion that no other such agent exists. The dialectical force of these two utterances is, of course, very different: a challenge, unlike a question, is an attack by Receiver on the dialectical position of Commander.

Accordingly, corresponding to the 38 critical questions presented in Sections 3.1–3.4, **CDP** has 76 locutions available to Receiver for questioning or challenging the command and its justification issued by Commander.[4] For reasons of space we do not list all of these here. Instead, Table 3 provides an example of these locutions, giving the 14 locutions corresponding to the seven critical questions CQR1–CQR7 of Section 3.2; these seven critical questions relate to the choice of agent selected by Commander to perform the action specified in the instruction.

4.4 Responding to a Question or Challenge to a Command

If Receiver utters a question or challenge to a command locution uttered by Commander, Commander has several alternatives in responding to it. Commander may say nothing at all or may withdraw from the interaction, as may happen at any time in any dialogue between autonomous agents. Commander may provide answers to questions or provide justifications to challenges from Receiver by stating (or re-stating) some aspect of the justification for the command, using the locutions in the left-hand column of Table 1. In the case where these justifications have already been uttered in the dialogue, and Receiver questions or challenges them, Commander may utter supporting evidence or arguments, again using the locutions in the left-hand column of Table 1, but instantiated with new content. In response to a question or challenge, Commander may also retract the initial command, using the *Retract_action_command(.)* locution of Table 1. Commander may also simply re-state the command.

The Command Dialogue Protocol has no guarantee of eventual termination or even of loop-freeness (i.e., freeness from circularity), since Commander may simply repeatedly restate a command which Receiver repeatedly rejects, or which Receiver

[4] In case this number of locutions is thought prolix, recall that **CDP** is intended for machine-to-machine communications; for comparison, the machine interaction protocol, Hypertext Transfer Protocol (HTTP), defines 41 standard status-code responses to a GET command, and allows for several hundred additional non-standard codes [20].

repeatedly accepts but never executes. Whether such features are considered desirable or not will depend upon the goals and values of the dialogue participants. For automated applications, it may be useful to explore modifications to **CDP** to avoid or ameliorate such protocol features, perhaps using the *instruction completion blocks* of **IDL3** [10, p. 259], and/or limits on the number of repetitions of the same utterance by an agent [15].

5 Outline of Semantics

We now present a semantic framework for **CDP**, which we merely sketch, for space reasons. We require both a semantics for command statements, and for the CD Protocol overall. For commands, we can utilize the double-possible-worlds semantics recently proposed by Reed and Norman [24] to formalize Hamblin's Action-State Semantics for imperatives [12]. This formal semantics includes representation both for world-states and for events (including actions); both are needed to accommodate actions which do not achieve their intended outcome-state, and states arising not through any deliberate action. Reed and Norman propose a ternary relationship across states and events, indicating that a given world-state is accessible from another by means of a particular event. Additional accessibility relationships encode time relationships between world-states, events and one another, in such a way that alternative models of time are possible. We have designed our representation for commands in order to map neatly into this semantic framework.

For the semantics of the protocol **CDP**, it would be straightforward to define an axiomatic semantics of pre- and post-conditions for each legal utterance in terms of their effects on the state of the dialogue. Given a suitable mental model of the participating agents, for example, in terms of the Beliefs, Desires and Intentions of each agent, it would be possible to extend these pre- and post-conditions to include the mental states of the participants, as does the Semantic Language *SL* of the FIPA Agent Communications Language *ACL* [7].[5] Instead of this, we adopt the denotational trace semantics approach presented for dialogues over action in [16], as refined in [17]. In this approach, participants in a multi-agent dialogue are viewed as jointly creating and manipulating objects in a shared conceptual space of action-intentionality tokens, analogously to the joint creation of natural language semantics in Discourse Representation Theory in linguistics [13]. Each token in the shared space represents a possible action, tagged with meta-information providing information about the preferences and intentions of agents regarding that action. Examples of such meta-information includes: the identity of the dialogue participant who first proposed the action; the identity(ies) of the intended executor(s) of the action; the identities of dialogue participants, if any, who have currently endorsed the proposal; and the identities of participants with the power to revoke the proposal. The tags on tokens enable the shared token space to be partitioned into sub-spaces, with different read-, write-, and delete-permissions applying in different sub-spaces. These sub-spaces may also be viewed as generalizations of the participants' Commitment Stores. Preferences between different actions expressed in the dialogue by participants are represented by labeled arrows between tokens.

[5] However, such an extension to include mental states would not, in general, be verifiable, since a sufficiently-clever agent could always simulate any required mental states [26].

In the case of commands, only the proposer of an instruction, Commander, has the power to revoke it, which simplifies the tags (and hence the sub-space partition structure). Moreover, **CDP** does not (currently) permit expression of preferences, so there are no arrows created between distinct tokens. Each command utterance in a dialogue using **CDP** causes the creation of an action-intentionality token in an appropriate subspace of the shared space. Utterances to indicate acceptance or rejection of a command likewise lead to tagging of the associated token, or equivalently, the creation of tokens in other sub-spaces of the overall space. A complete articulation of the mapping between **CDP** utterances and objects in the token-space semantics will be presented in later work.

6 Conclusion

We have a proposed a novel representation for commands in computational systems, with each command represented as a presumptive argument scheme for action by a designated agent along with a list of critical questions, in the manner of [2]. We have then used this representation to specify a multi-agent dialogue protocol for command dialogues, **CDP**. A denotational semantic framework for this protocol has also been outlined, with commands interpreted via the Action-State double-possible-worlds semantics of [24], and dialogue utterances understood as intended to manipulate action-intentionality tokens in a shared conceptual space, following [17]. Such a semantic framework may be viewed as a tuple space co-ordination model [9], with tokens (i.e., tuples) manipulated by agents using law-governed Linda [18]. In other work [5,6], one of has shown how such a semantic framework can enable the straightforward implementation of agent interaction protocols, using the TuCSoN tuple centre platform [22].

Our work in this paper lies in the field of software engineering of multi-agent systems using argumentation: we have presented a principled framework for the design of a machine-to-machine dialogue protocol which would enable commands to be issued, questioned, challenged and justified between autonomous software agents. To the best of our knowledge, **CDP** is the first computational framework to enable multi-agent command dialogues.

Of course, much work remains to be done before this framework would be ready for production deployment. In addition to a full articulation of the semantics, and the development of a prototype implementation, several other aspects require further research. Firstly, a computational representation of the social context between the participants, as in [14], may be useful to provide a stronger semantic underpinning of those critical questions relating to the authority of the Commander (CQX1–CQX6). In some social contexts, of course, agents issuing commands may have access to arsenals of rewards or threats to ensure compliance; although a different topic to that of command dialogs, work in that area may be relevant, e.g., [23]. Secondly, we plan to refine our model of time, so as to enable more sophisticated treatment of the relationships between the timing and duration of actions and the timing of achievement of world-states and goals. We also plan to consider how to incorporate uncertainty regarding the success or failure of actions, and discussion of the relative costs and benefits of alternative actions, possibly drawing on the qualitative decision theory of [8]. Such work would support the development of appropriate critical questions for argument scheme CAS-S2, which we

did not consider in this paper. Finally, recent work in multi-agent systems has looked at delegation and responsibility, e.g., [21]; it may be valuable to explore the relationships between that work and multi-agent dialogs over commands requiring joint action for their fulfilment.

Acknowledgments. We thank the anonymous referees and the audience at ArgMAS 2008 for their comments. Two authors (PM and SP) are grateful for partial financial support received from the EC *Information Society Technologies* (IST) programme, through project ASPIC (IST-FP6-002307). This work was also partially supported by the US Army Research Laboratory and the UK Ministry of Defence under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

1. Atkinson, K., Bench-Capon, T.: Action-based alternating transition systems for arguments about action. In: Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI 2007), pp. 24–29. AAAI Press, Menlo Park (2007)
2. Atkinson, K., Bench-Capon, T., McBurney, P.: A dialogue game protocol for multi-agent argument for proposals over action. Autonomous Agents and Multi-Agent Systems 11(2), 153–171 (2005)
3. Atkinson, K., Bench-Capon, T., McBurney, P.: Computational representation of practical argument. Synthese 152(2), 157–206 (2006)
4. Bench-Capon, T.J.M., Dunne, P.E., Leng, P.H.: Interacting with knowledge-based systems through dialogue games. In: Proceedings of the Eleventh International Conference on Expert Systems and Applications, Avignon, pp. 123–140 (1991)
5. Doutre, S., McBurney, P., Wooldridge, M.: Law-governed linda as a semantics for agent interaction protocols. In: Dignum, F., Dignum, V., Koenig, S., Kraus, S., Singh, M.P., Wooldridge, M. (eds.) Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2005), Utrecht, The Netherlands, pp. 1257–1258. ACM Press, New York (2005)
6. Doutre, S., McBurney, P., Wooldridge, M., Barden, W.: Information-seeking agent dialogs with permissions and arguments. Technical Report ULCS-05-010, Department of Computer Science, University of Liverpool, Liverpool, UK (2005), `www.csc.liv.ac.uk/research/techreports/tr2005/tr05010abs.html`
7. FIPA. Communicative Act Library Specification. Standard SC00037J, Foundation for Intelligent Physical Agents, December 3 (2002)
8. Fox, J., Parsons, S.: Arguing about beliefs and actions. In: Hunter, A., Parsons, S. (eds.) Applications of Uncertainty Formalisms. LNCS (LNAI), vol. 1455, pp. 266–302. Springer, Heidelberg (1998)
9. Gelernter, D.: Generative communication in Linda. ACM Transactions on Programming Languages and Systems 7(1), 80–112 (1985)
10. Girle, R.: Commands in Dialogue Logic. In: Gabbay, D.M., Ohlbach, H.J. (eds.) FAPR 1996. LNCS (LNAI), vol. 1085, pp. 246–260. Springer, Heidelberg (1996)

11. Habermas, J.: The Theory of Communicative Action, Reason and the Rationalization of Society, Heinemann, London, vol. 1 (1984); Translation by T. McCarthy
12. Hamblin, C.L.: Imperatives. Basil Blackwell, Oxford (1987)
13. Kamp, H., Reyle, U.: From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory. Kluwer, Dordrecht (1993)
14. Karunatillake, N.C.: Argumentation-Based Negotiation in a Social Context. Phd, School of Electronics and Computer Science, University of Southampton, Southampton, UK (2006)
15. Krabbe, E.C.W.: The problem of retraction in critical discussion. Synthese 127(1-2), 141–159 (2001)
16. McBurney, P., Parsons, S.: A denotational semantics for deliberation dialogues. In: Rahwan, I., Moraïtis, P., Reed, C. (eds.) ArgMAS 2004. LNCS, vol. 3366, pp. 162–175. Springer, Heidelberg (2005)
17. McBurney, P., Parsons, S.: Retraction and revocation in agent deliberation dialogs. Argumentation 21(3), 269–289 (2007)
18. Minsky, N.H., Leichter, J.: Law-governed Linda as a coordination model. In: Ciancarini, P., Nierstrasz, O., Yonezawa, A. (eds.) ECOOP-WS 1994. LNCS, vol. 924, pp. 125–146. Springer, Heidelberg (1995)
19. Moore, D.J.: Dialogue Game Theory for Intelligent Tutoring Systems. PhD thesis, Leeds Metropolitan University, Leeds (1993)
20. Network Working Group. Hypertext Transfer Protocol — HTTP/1.1. Technical Report RFC 2616, Internet Engineering Task Force (June 1999)
21. Norman, T.J., Reed, C.: Delegation and responsibility. In: Castelfranchi, C., Lespérance, Y. (eds.) ATAL 2000. LNCS, vol. 1986, p. 136. Springer, Heidelberg (2001)
22. Omicini, A., Denti, E.: From tuple spaces to tuple centres. Science of Computer Programming 41(3), 277–294 (2001)
23. Ramchurn, S.D., Sierra, C., Godo, L., Jennings, N.R.: Negotiating using rewards. In: Nakashima, H., Wellman, M.P., Weiss, G., Stone, P. (eds.) Proceedings of the Fifth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2006), Hakodate, Japan, pp. 400–407. ACM Press, New York (2006)
24. Reed, C., Norman, T.: A formal characterisation of Hamblin's action-state semantics. Journal of Philosophical Logic 36, 415–448 (2007)
25. Walton, D.N.: Argumentation Schemes for Presumptive Reasoning. Lawrence Erlbaum Associates, Mahwah (1996)
26. Wooldridge, M.J.: Semantic issues in the verification of agent communication languages. Journal of Autonomous Agents and Multi-Agent Systems 3(1), 9–31 (2000)
27. Zambonelli, F., Jennings, N.R., Wooldridge, M.: Developing multiagent systems: the Gaia methodology. ACM Transactions on Software Engineering Methodology 21(3), 317–370 (2003)

Argumentation and Artifact for Dialogue Support

Enrico Oliva[1], Mirko Viroli[1], Andrea Omicini[1], and Peter McBurney[2]

[1] Alma Mater Studiorum–Università di Bologna, Cesena, Italy
[2] University of Liverpool, Liverpool L69 3BX UK

Abstract. Intelligent and autonomous software agents may engage in dialogue and argument with one another, and much recent research has considered protocols, architectures and frameworks for this. Just as with human dialogues, such agent dialogues may be facilitated by the presence of a mediator, able to summarise different positions, identify common assumptions and inconsistencies, and make appropriate interventions in the dialogue. Drawing on the theory of co-ordination artifacts in multi-agent systems, we propose a formal framework to explicitly represent the functions of a mediator artifact. We then describe an implementation of this framework using the TuCSoN coordination infrastructure for MAS, where the mediator artifact is realised by a tuple centre—a programmable tuple space.

1 Introduction

Proponents of public policy conversations and decision-making processes usually emphasise the need for a human moderator or mediator to be involved in the interaction, e.g., Forester [3]. The mediator may act to ensure fairness and equality of access by all participants, may assist participants to clarify their positions and to argue more effectively, and may even seek to reconcile opposing views. Similarly, the designers of computer-aided argumentation systems have also provided support for human mediators; for example, the developers of *Zeno* define their system as "a mediation system" [5, p. 10]:

> "a kind of computer-based discussion forum with particular support for argumentation. In addition to the generic functions for viewing, browsing and responding to messages, a mediation system uses a formal model of argumentation to facilitate retrieval, to show and manage dependencies between arguments, to provide heuristic information focusing the discussion on solutions which appear most promising, and to assist human mediators in providing advice about the rights and obligations of the participants in formally regulated decision making procedures."

Just as with human interactions, and for the same reasons, many of the functions provided by mediators could be useful when software agents engage in argumentation with one another. Most of these mediator functions are better supported through some algorithmic procedure, rather than by some articulated process of rational deliberation. In earlier work [8], we presented a conceptual framework for a central co-ordinating

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 107–121, 2009.

entity in an argumentation dialogue, called a *Co-Argumentation Artifact (CAA)*, to provide co-ordination services to the participating agents, allowing them to share, store and exchange arguments with one another. A CAA is an artifact, a computational entity used by the agents, specialised in argumentation reasoning. Vesting the CAA with its own argumentation capabilities meant that this entity, like the participants, could elaborate over the arguments stored. For example, the CAA could embody algorithms determining whether a particular argument is acceptable (under a specified semantics of argumentation) with respect to the global knowledge of all the participants. We reprise the CAA framework in Section 3.1.

It is easy to imagine that the CAA could undertake more sophisticated interventions in the dialogue, resembling complex, automated tasks of a human mediator. To this end, in this paper we extend our earlier concept of a central co-ordinating artifact to be a dialogue artifact (DA), acting as a mediator between the participating agents, enacting the agents participation in the dialogue. We do this, first, by articulating, in Section 2, the possible functions of the mediator artifact; for reasons of space, we do not consider all such functions here. We then present a formalisation of some of such mediator artifact functions in Section 4, drawing on recent work in the theory of communications artifacts in multi-agent systems [11,18]. We follow this with a description of a prototype implementation we have undertaken in the TuCSoN coordination infrastructure, in Section 5. Finally, Section 6 concludes the paper.

2 Functionalities of Multi-agent Argumentation Support

In an agent and human society, argumentative reasoning system and dialogue system have a central role enabling exchange of knowledge, common sense reasoning, dispute resolution and argumentative communication between agents.

Our model of a multi-agent argumentation system follows the A&A meta-model [12,17], which defines MAS as composed of two kinds of entities: agents, in charge of autonomous and proactive behaviour, and *artifacts*, providing function-oriented services to agents in a passive way. An agent is an autonomus computational entity situated in an enviroment, which is composed by artifacts; an artifact is hence a social construct shared by the agents that enables, constraints or mediates their activities.

In this work we want to propose a model that merges concepts from argumentation and artifact theories, and offers a coherent framework to cater for these reasoning and coordination abilities that agents need to exhibit. More precisely, we brings the theory of argumentation and of dialectical agents interaction to a level of operationalisation.

Dialogue participants, of course, need to be able to generate, evaluate, contest and defend arguments as they interact with one another through the dialogue. But the mediator artifact also needs this argumentation functionality if it is to find common ground between different participants, or to clarify their differences. For example, if the mediator is to convince two participants that their opposed positions in fact share common assumptions or that one position implies the other, then the mediator artifact may need – in an automated way – to create, present and defend a case to the participants.

Consequently, we define two types of artifacts: Dialogue Artifacts (DA), to support dialectical agent interaction, and Co-Argumentation Artifacts (CAA), to support agent

argumentative reasoning. The DA realises the mediator artifact enabling argumentative communication among a multiplicity of entities. DA exploits directly CAA functionalities in order to drive the dialectical interaction between agents through argumentative reasoning over the argument set stored inside the centralised CAA.

The CAA realises coordination services based on argumentative reasoning between multi entities. For example, the CAA could determine whether a particular argument is acceptable (under a specified semantics of argumentation) with respect to the global knowledge of all the participants. Briefly the CAA is composed of arguments, represented in a suitable computational form, and a collection of algorithms deployed over this argument set. The CAA is well introduced in a recent paper by Oliva et al [8].

The combination of both artifacts DA and CAA provides the support of basic and advanced functionality for automatic mediation services in a MAS based on argumentation.

2.1 Dialogue Artifact

The dialogue artifact requires some basic functionality to support the exchange of arguments in a dialogue between the participants. This basic functionality includes:

1. Storage of the dialogue protocol (e.g. in a library of such protocols)
2. Storage of the specifications of the dialogue protocol
3. Storage of the complete history of a dialogue as it proceeds
4. The ability to refuse to allow agent utterances which do not conform to the current protocol in use
5. The ability to suggest next moves which are legal according to the current protocol in use in a dialogue
6. The ability to receive and store confidential information from the participating agents, such as their preferences in a negotiation. The mediator could then aggregate such information (across multiple agents), and/or seek to identify and reconcile differences.

Also, the dialogue artifact could act as sophisticated mediator of the discussion, by providing in an automated way the following services:

- Seeking to resolve any disputes over the rules of the protocol
- Providing rewards or penalties to agents for breaking the protocol rules
- Having the power to admit or to expel agents to/from the dialogue
- Suggesting a new protocol, when needed.
- Supporting multiple simultaneous bilateral interactions.
- Assigning roles, rights and responsibilities to agents at run-time, as, for example, in an action protocol, assigning the role of winner to a particular agent near the end of the interaction.
- Identifying conflicts and inconsistencies between commitments made by agents in a dialogue, for example, if an agent commits to sell a car it is also trying to purchase.
- Identifying agent utterances which are not relevant to the current state of the dialogue, and refusing to permit these to be made.
- Providing automated alerts to inform agents that dialogues on particular topics are about to start, or to end, or that particular commitments have just been made.
- Combining different dialogues on the same topic.

More advanced functions could also include:

- Annotation of protocols with their properties, for protocols stored in the protocol library, for instance, the possible outcomes of a protocol, its computational complexity, and so on.
- Storing the outcomes of past dialogues, for example, the commitments remaining at the end of the dialogue.
- Tracking agent commitments across multiple dialogues.
- Using previous dialogues to create an independent assessment of the reputation of participating agents.
- Storage of the entire history of past dialogues. These may be required for regulatory or legal reasons, e.g., in stock market transactions.

In this paper, we present a Dialogue Artifact supporting dialectical argumentation in a MAS scenario, which provides some of the basic functionalities listed above. The DA is based over two formal systems: an argumentation system and a dialogue system, explained in the next sections.

3 Argumentation and Dialogue: Formal Definitions

In our earlier work [8], we introduced an argumentation system and an artifact abstraction to support the co-ordination functions necessary to support agent argumentation. We now extend that framework to handle argumentation dialogues, drawing on the theory of organisations and roles in multi-agent systems of Omicini *et al.* [11] and the formal language used to define an agent interaction protocol, in the work of Viroli *et al.* [18]. In our approach we describe a dialogue in terms of a labelled process algebra, where labels denote roles, as in [11], and the process algebra specifies the interaction protocol, as in [18].

We assume that the interaction is between a finite number N of intelligent software agents, and that each agent has a range of possible utterances (or actions) at each step in the dialogue (i.e., this is a multi-move protocol). Formally, a multi-agent dialogue system for argumentation is composed of two parts: an argumentation system, and a dialogue system. The definition of the argumentation system is discussed based on the work in [8], and builds on various earlier argumentation frameworks.

3.1 Argumentation System

Prakken and Vreeswijk [16] observe that an argumentation system is generally composed of five elements (although sometimes implicitly): (1) a logical language; (2) an argument definition; (3) a concept of conflict among arguments; (4) a concept of defeated argument; (5) a concept of argument acceptability. In this section we define an argumentation system as a reference point for our work. We take inspiration from Dung's framework [2], and we also define the structure inside the arguments.

The object language of our system is a first-order language, where Σ contains all well-formed formulae. The symbol $\vdash$ denotes classical inference (different styles will be used like deduction, induction and abduction) $\equiv$ denotes logical equivalence, and $\neg$ or *non* is used for logical negation.

Table 1. Deductive Inference: (MP) Modus Ponens, (MMP) Multi-Modus Ponens and (MT) Modus Tollens; Inductive and Abductive Inference: (θ-su) θ-subsumption, (Ab) Abductive

Deductive Inference		Inductive Inference	
MP	$\frac{A \quad A \to B}{B}$	θ-su	$\frac{B}{R}$ *where* $R\theta \subseteq B$
MT	$\frac{\neg A \quad B \to A}{\neg B}$	Abductive Inference	
MMP	$\frac{B_1,\ldots B_n \quad (B_1,\ldots B_n) \to C}{C}$	Ab	$\frac{B \quad A \to B}{A}$

Definition 1 (argument). *An* argument *is a triple* $A = \langle B, I, C \rangle$ *where* $B = \{p_1, \ldots, {}_n\} \subseteq \Sigma$ *is a set of* beliefs, $\vdash_I \in \{\vdash_d, \vdash_i, \vdash_a\}$ *is the inference style (respectively,* deduction, induction, *or* abduction*), and* $C = \{c_1, \ldots, c_n\} \subseteq \Sigma$ *is a set of* conclusions*, such that:*

1. *B is consistent*
2. $B \vdash_I C$
3. *B is minimal, so no subset of B satisfying both 1 and 2 exists*

The types of inference we consider for deduction, induction and abduction are shown in Table 1. Modus Ponens (MP) is a particular case of Multi-Modus Ponens (MMP) with only one premise. The inference process θ-subsumption derives a general rule R from specific beliefs B, but is not a legal inference in a strict sense.

For defeat of arguments, the definition is not straightforward because there are different type of attack well defined in [16]. Following those definitions, two possible types of attack are 'conclusions against conclusions' – called *rebuttals* – and 'conclusions against beliefs'—called *undercuts*.

Definition 2 (contrary). *The contrary (or attack) relation R is a binary relation over* Σ *that* $\forall p_1, p_2 \in \Sigma$, $p_1 R p_2$ *iff* $p_1 \equiv \neg p_2$

Definition 3 (undercut). *Let* $A_1 = \langle B_1, I_1, C_1 \rangle$ *and* $A_2 = \langle B_2, I_2, C_2 \rangle$ *be two distinct arguments,* A_1 *is an* undercut *for* A_2 *iff* $\exists h \in C_1$ *such that* hRb_i *where* $b_i \in B_2$

Definition 4 (rebuttal). *Let* $A_1 = \langle B_1, I_1, C_1 \rangle$ *and* $A_2 = \langle B_2, I_2, C_2 \rangle$ *be two distinct arguments,* A_1 *is a* rebuttal *for* A_2 *iff* $\exists h \in C_1$ *such that* hRc_i *where* $c_i \in C_2$

The definitions of acceptability and admissibility used in our framework are those of Dung in [2]. The following definitions are the basic ones in our argumentation system and follow from Dung's framework.

Definition 5 (conflict-free set). *An argument set S is a* conflict free *set iff there exist no* $A_i, A_j \in S$ *such that* A_i *attacks* A_j.

Definition 6 (collective defense). *An argument set S* defends collectively *all its elements if* $\forall$ *argument* $B \notin S$ *where B attacks* $A \in S \quad \exists \; C \in S : C$ *attacks B.*

Definition 7 (admissible set). *An argument set S is a* admissible *set iff S is conflict free and S defends collectively all its elements.*

Definition 8 (preferred extension). *An argument set S is a* preferred extension *iff S is a maximal set among the admissible set of A.*

An argument is acceptable in the context of preferred semantics if an argument is in some/all preferred extensions (credulous/sceptical acceptance).

Definition 9 (credulous acceptability). *An argument A is* credulous acceptable *if A ∈ at least one preferred extension.*

Definition 10 (sceptical acceptability). *An argument A is* sceptical acceptable *if A ∈ all preferred extensions.*

For further details, including an implementation and examples of this argumentation framework, we refer the interested reader to our earlier paper [8].

3.2 Dialogue System

In this section we present a novel formalisation of a multi-agent dialogue system. Our intention is to capture the rules that govern legal utterances, as well as the effects of utterances on the commitment stores of the dialogue. We use a process algebra approach in the style of [18] to represent the possible paths that a dialogue may take, and to represent explicitly the operations to and from the commitment store. We proceed by considering each element of a dialogue system in turn: (1) the communication language; (2) the interaction protocol; and (3) the protocol semantics.

Because a dialogue is a dialectical exchange of arguments, we assume that arguments and counter-arguments are represented and expressed in the formal language defined above in Section 3.1. Agents may exchange arguments, along with facts, with one another in the form of instantiated parameters in their utterances.

Communication Language. The agents need to share a same communication language *CL* in order to exchange information. The role of *CL* as a language used for internal knowledge representation and reasoning is explained in [14]. We let F denote a set of terms representing *facts*, and $\mathscr{A}$ the set terms representing all *arguments* able to be represented in Σ following the definition of an argument given in Definition 1. Our *CL* is defined in order to support all six primary dialogue types as identified by [19]: persuasion, inquiry, negotiation, information seeking, deliberation and eristic.

Definition 11 (communication language). *Our* communication language *is a set of locutions L_c. A locution $l \in L_c$ is a expression of the form $perf_{name}(Arg_1,\ldots,Arg_n)$ where $perf_{name}$ is a element of the set P of performatives and Arg_x is either a fact or an argument.*

An agent performing a dialogue using the communication language can utter a locution composed of facts and arguments. A fact is represented by syntax `fact(Terms)` and an argument with `argument(B,I,C)`. The definitions to manage attacking and undercutting arguments are provided by the underlying argumentation system given in Definition 1. In Example 1 an agent wants to communicate the classical example of argument like *All men are mortal, Socrates is a man, Socrates is mortal*, so it uses a `Argue` locution with an `argument` parameter.

Example 1. `Argue(argument(name,beliefs([human(Socrates)], [clause(mortal(X), [human(X)])]),infer(MP),conclusions ([mortal(Socrates)])))`.

Examples of performatives to support an instance of an *Information Seeking Dialogue* could be: `OpenDialogue`, `Ask`, `Tell`, `DontTell`, `Provide`, `Argue`, and so on. Further details about this form of dialogue and its complete locutions are presented in [1] (see also Example 2).

Dialogue Protocol. In our framework the dialogue protocol is a complete description of all possible dialogue paths, from the perspective of an external entity observing the dialogue between the agents. The protocol indicates the possible paths of a dialogue, specifies the source and target of each message, and shows the relationship between utterances and the content of commitment stores. Our approach basically describes the step-by-step behaviour of an external entity acting as a mediator, hence enabling the allowed interactions. Hence, technically, we find it useful to model a dialogue in terms of a process algebra with standard composition operators (sequence, parallel, iteration), and whose atomic actions represent either agent utterances, or interactions with the commitment store (writing, reading, or removing a commitment).

On the one hand, Prakken [15] proposes a general definition of locution where a move m is denoted by four elements: (1) identifier, (2) speaker (or source), (3) speech act, and (4) intended recipient (or target). Following this model, we provides a definition of a speech act, as follows:

Definition 12 (action). *An* action *A is defined by the syntax $A ::= s : L_c | s[t_1,\ldots,t_n] : L_c$ where s indicates the source, and $[t1,\ldots,t_n]$ indicates the (optional) targets of the message.*

On the other, beyond this, we include additional atomic operations K over commitment stores—many of them can actually occur into one argumentation artifact. To this end, the commitment store is viewed as a set of tuples as in [7]: such tuples are manipulated by the commands of the Linda language [4]—`in`, `rd` and `out`.

Definition 13 (term action). *A* term action *K has the syntax $K ::= in(C,X)|out(C,X)|rd(C,X)$, where C is a term representing the commitment store identifier, and X is a term representing the commitment.*

Specifically, the commands `in(C,t)`, `rd(C,t)` and `out(C,t)` respectively consumes, reads and puts a tuple `t` in the commitment store `C`. These actions are useful to manage the private or public commitment store in relation to the dialogue execution. In particular, they can operate, for example, as action-preconditions in order to restrict or constrain the next action choice, and thus enable only certain future dialogue paths. For instance, if at a given time a sub-dialogue is guarded by operation `rd(c,commit(a))`, then it is allowed to proceed only if `commit(a)` occurs in the commitment store.

Definition 14 (protocol). *A* protocol *P is a composition of action from sets A and K, defined by syntax $P ::= 0 | A.P | K.P | P + P | (P \parallel P) | !P$ where the symbols $.,+,\|,!$ denote respectively sequence (action prefix), choice, parallel composition, and infinite replication operators, and the symbol 0 denotes the empty protocol.*

For example, an abstract dialogue protocol definition is given by $D := (s : a_1 + s : a_2).(s : a_3 + s : a_4).s : a_5$ where agent s is only allowed to execute a sequence of three actions: the sequence composed of a first action consisting of either action a_1 or action a_2, then a second action consisting of either a_3 or a_4, and then a third action comprising a_5. A protocol specifies a set of actions histories that the agents might execute. As another example of a protocol definition, consider $D := s : a_1 \parallel s : a_1 \parallel s : a_1 \parallel t : a_2 \parallel t : a_3$ where agent s invokes a_1 three times, agent t can invoke a_2 and a_3 only once, but in whichever order.

To illustrate this framework, we present a specification for an Information-Seeking Dialogue (f is seen as a variable over the content of communication):

Example 2 (Information Seeking Dialogue). This protocol involves two agents: an agent `s` *controlling information, and an agent* `c` *trying to persuade* `s` *to give him the permission to access. Operation* `rd(permission(c,f))` *is the instruction by which the protol instance checks whether* `c` *can be given the permission: if this is the case, permission is provided and the dialogue ends; otherwise,* `c` *should try to persuade* `s` *by arguing an* `ADD`*, which can then by either accepted or refused by* `s`*.*

```
c:Opendialogue.
s:Opendialogue.
c:Ask(f).(
     rd(permission(c,f)).
     s:Tell(f).
     s:Provide(f).
     s:Argue(permission(c,f),YES,A).
     s:Enddialog.
     c:Enddialog
    +
     rd(not(permission(c,f))).
     s:DontTell(f).
     s:Argue(permission(c,f),NO,B).
     c:Argue(permission(c,f),ADD,A).(
        s:Argue(permission(c,f),NO,A).
        s:Enddialog.
        c:Enddialog
       +
        s:Accept(A,permission(c,f)).
        out(accept(permission(c,f))).
        s:Provide(f).
        s:Enddialog.
        c:Enddialog
)   )
```

3.3 Operational Semantics

Following Hamblin [6], we assume that each agent is associated to a knowledge base, accessible to all agents, containing its commitments made in the course of the dialogue.

Commitments are understood as statements which the associated agent must support, while they remain in the commitment store, if these statements are either questioned or attacked by other agents. We can now use the notion of commitment store and the transition system given in Definition 16 to define an operational semantics for the dialogue system. This semantics describes the evolution over time of the dialogue state and the states of commitment store (seen as composition of all commitment stores). In essence, the commitment store is the knowledge repository of the dialogue as a whole, and it is expressed in our framework as a multiset of terms.

Definition 15 (commitment store). *A* commitment store *C is a multiset of terms and it is defined by the syntax* $C ::= 0 | (C|C) | X$ *where X is a term, and* 0 *is the empty set.*

Definition 16 (operational semantics). *The* operational semantics *of our dialogue system is described by a labelled transition system* $\langle S, \rightarrow, I \rangle$*, where* $S ::= (C)P$ *represents the state of dialogue system (protocol P running with commitment store C), I is the set of interactions (labels) composed of* $i ::= \tau | a$*, and* $\rightarrow$ *is a transition relation of the kind* $\rightarrow \subseteq S \times I \times S$.

As usual, we write $s \xrightarrow{i} s'$ in place of $\langle s, i, s' \rangle \in \rightarrow$, meaning the dialogue system moves from state s to s' due to interaction i—either an action a, or an internal step τ (an operation over the commitment store). We introduce a congruence relation $\equiv$, which syntactically equates similar states:

$$0 + P \equiv P \qquad P + Q \equiv Q + P \qquad (P + Q) + R \equiv P + (Q + R) \qquad !P \equiv P | !P$$
$$0 \parallel P \equiv P \qquad P \parallel Q \equiv Q \parallel P \qquad (P \parallel Q) \parallel R \equiv P \parallel (Q \parallel R)$$

We use also notation $t\{x/y\}$, to mean term t after applying the most general substitution between terms x and y—x should be an instance of y, otherwise the substitution notation would not make sense. Finally, we define operational rules that describe the behavior of the dialogue system as follows:

$$(C)out(x).P \xrightarrow{\tau} (C|x)P \qquad (K-OUT)$$
$$(C|x)rd(y).P \xrightarrow{\tau} (C|x)P\{x/y\} \qquad (K-RD)$$
$$(C|x)in(y).P \xrightarrow{\tau} (C)P\{x/y\} \qquad (K-IN)$$
$$(C)(P+Q) \xrightarrow{i} (C')P' \quad \text{if} \quad (C)P \xrightarrow{i} (C')P' \qquad (OP-SUM)$$
$$(C)(P|Q) \xrightarrow{i} (C')(P'|Q) \quad \text{if} \quad (C)P \xrightarrow{i} (C')P' \qquad (OP-PAR)$$
$$(C)a.P \xrightarrow{a'} (C)P\{a'/a\} \qquad (ACT)$$

Rule (K-OUT) provides the semantic of `out` operation, expressing that x term is added to the commitment store C, and process continuation can carry on. Rules (K-RD) and (K-IN) similarly handle operation `rd` and `in`: the use of substitution operator guarantees that the term x in the commitment store is an instance of the term x to be retrieved. Rules (OP-SUM) and (OP-PAR) provide the semantics for choice and parallel operators in the standard way. Finally rule (ACT) expresses that locution a' is executed that is an instance of the allowed one a, and accordingly process continuation P can carry on.

4 The Dialogue Artifact

As mentioned above, the A&A meta-model for MAS as discussed in [8] views agents engaged in argumentative communication as making use of an abstraction, called a Co-Argumentation Artifact, to communicate, to exchange information, data and arguments, and to record their public commitments. The current work extends this abstraction by formally defining a Dialogue Artifact (DA), able to support and mediate the communication between agents engaged in a dialogue under the system defined in Section 3 above.

Definition 17 (Dialogue Artifact). *A* Dialogue Artifact *is a triple* $DA = \langle DP, CS, IC \rangle$*, where*

- *DP is a collection of specifications of dialogue protocols*
- *CS is a collection of commitment stores*
- *IC is a collection of specifications of interaction control (IC)*

The *DP*, *CS* and *IC* components are in turn defined in the following subsections.

Dialogue Protocols. The class *DP* is a collection of formal specifications of dialogue protocols, with each protocol specified using a labelled process algebra, as in Definition 14. Protocols in *DP* may also be annotated with identifiers and with their properties, such as their termination complexity. When agents engage in dialogue using a protocol in the collection *DP*, they make utterances according to the permitted sequences defined by the protocol specification. Accordingly, the Dialogue Artifact is able to verify that utterances proposed by agents in a dialogue are valid under the protocol; the DA is also able to use the specification to suggest potential legal utterances to participating agents at each point in the dialogue.

Commitment Stores. For any particular collection of agents and any particular dialogue they undertake, the collection *CS* specifies a set of stores representing the private and public Commitment Stores of each participant, together with a central Commitment Store for the dialogue as a whole. The Dialogue Artifact can support the dialogue by holding these stores. The private Commitment Stores are also held by the DA to record confidential information entrusted to it by the participants, such as their private valuations of some scarce resource (in the case of Negotiation dialogues) or arguments based on privileged information (in the case of dialogues over beliefs). Sharing such information with the DA may allow the DA to elaborate over such stores while not revealing private information of individual agents.

We can classify the various types of stores according to the access permissions (write, read, and delete permissions) holding on each store, as shown in Table 2. The cells of the table indicate the access permissions pertaining to different types of Commitment Stores (the rows of the table), depending on the agent seeking access (the columns of the table). The Dialogue Artifact may also store other relevant information, such as the sequence of locutions exchanged in the current dialogue, which would be stored in the Central Commitment Store. These stores do not have an algebraic structure, but rather a declarative representation of the contents with a proper classification.

Table 2. Commitment Stores - Read (R), write (W) and delete (D) Permissions

Type	Agent A	All Agents	Mediator Artifact
Private Commitment Store of Agent A	R/W/D	-	R
Public Commitment Store of Agent A	R/W/D	R	R
Central Commitment Store	R	R	R/W/D

Interaction Control. The third component of the Dialogue Artifact, denoted as *IC*, is a collection of specifications for interaction control. *IC* roughly follows the MVC (Model View Control) pattern, where the model is the dialogue specification in *DP*, the view is the *CS* component with dialogue trace, and the control is represented by the *IC* specification. The control rule of the dialogue is represented by the labelled transition system introduced in previous sections, modelling the evolution over time of the agent interaction protocol. Three operators can be used to control the dialogue:

$$next^I(s) = \left\{ i : s \xrightarrow{i} s' \right\} \; next^S(s) = \left\{ s' : \exists i, s \xrightarrow{i} s' \right\} \; next^{IS} = \left\{ (i, s') : s \xrightarrow{i} s' \right\}$$

Operator $next^I(s)$ yields the next admissible interactions i from state s. Operator $next^S(s)$ yields the states reachable from s in one step. Operator $next^{IS}$ yields couples (i,s) instead.

The *IC* component realises the above three operators in order to identify which potential utterances are legal for any agent at any point in the dialogue. The basic primitives `in,rd,out` to manage arguments and facts in commitment stores allow the *IC* to identify which constraints on the future course of dialogues are created by the existing commitments. For instance, the *IC* could permit only one utterance in a choice point basing the decision on state of commitment store. Also, it could work with an argument set over some advanced structures such as conflict free sets and preferred extensions presented in Section 3.1 to determine for instance the acceptability of an argument.

DA Functionalities. It is straightforward to see that all six basic functionalities of the central Dialogue Artifact listed in Section 2.1 can be performed by a Dialogue Artifact defined as a triple $DA = \langle DP, CS, LI \rangle$ as above. The collection *DP* provides the functionalities of items 1 and 2, the storage of protocols and their formal specifications; the Central Commitment Store of the collection *CS* provides storage for the history of a dialogue (item 3); similarly, the private Commitment Store components of the collection *CS* provide storage for confidential information communicated from agents to the DA (item 6); the formal specification of a protocol in *DP* (as given by the process algebra formalism we have used above) permits the DA to identify potential utterances which do not conform to the protocol (item 4); and, both the formal protocol specifications in the collection *DP* and the logics of interaction in *IC* permit the DA to suggest possible legal next moves (item 5).

5 TuCSoN Implementation

The technological support to build a *DA* is provided by the TuCSoN coordination infrastructure for MAS introduced in [13]. TuCSoN provides MAS with coordination

```
dialoguesession(infoseek,close)
participant(infoseek,2)
dialogue(infoseek,[act(C,openDialogue(C,T)),
       act(T,openDialogue(C,T)),act(C,ask(Arg)+
       (act(T,tell(arg1),cs(T,out(commit(arg1)))))])
currentpar(infoseek,0)
```

Fig. 1. Example of Dialogue State (*DP* component)

```
%reacts from agent next moves request
reaction(rd(nextmoves(Dialogue,S)),(
  rd_r(dialoguestate(Dialogue,S)),
  out_r(findall(S,Dialogue))
)).
reaction(out_r(findall(S,Dialogue)),(
  in_r(findall(S,Dialogue)),
  findall(A,transition(S,A,Q),L),%collect all next legal moves
  out_r(nextmoves(Dialogue,L))
)).
```

Fig. 2. Implementation of $next^I$ operator in ReSpecT

abstractions called *tuple centres* where agents write, read and consume logic tuples via simple communication operations (`out`, `rd`, `in`, `inp`, `rdp`). In particular, `inp`, `rdp` respectively consume and read matching tuples in the same way `in`, `rd`; unlike `in`, `rd` they fail if the tuple is not present when the request is served. As programmable tuple spaces [10], tuple centers can play the role of agent mediator, where coordination rules are expressed in terms of logic specification tuplesof the ReSpecT language—an event driven language over the multi-set of tuples [9]. Since tuple centre can be used as a general-purpose support for MAS artifacts, we exploited TuCSoN logic tuple centres in order to implement *DA*.

In this framework, agents utter a locution by means of an `out(move(Dialogue, AgentID,Locution))` in the tuple centre. The automatic actions executed over the commitment store are represented by the term `cs(ID,out(commit(...)))`—where `out` could be replaced by `in` or `rd` operations. The *CS* class is composed of `commit` tuples that are put in the tuple space as facts and arguments expressed in logic tuple notation.

The dialogue is written in terms of tuples `dialogue(name,AList)` where `AList` is the list of actions reifying in tuple form the operators choice `act(A1)+(act(A2))`, parallel `par(A1,A2)` and sequence `A1,A2`. Figure 1 shows a dialogue protocol composed by some basic information on dialogue state and few steps of the *Information Seeking Dialogue* protocol. The tuples that form the *DP* component are: `participant` (potential number of participants), `dialogue` (dialogue protocol), `dialoguestate` (actual protocol dialogue state), and `currentpar` (actual number of participants). In addition, an open dialogue session also uses tuple `session(AgentID,infoseek, open)` for each dialogue participant.

The key idea of the *IC* implementation is shown in figure 3, where the reactions implementing the control of dialogue interaction are presented. In particular, the code implements the dialogue state transition after an agent action, the search of next admissible move after an agent request, and also makes it possible the automatic interaction with the commitment store executing `cs` actions. Such mechanisms make it possible

```
transition(cs(Id,A),cs(Id,A),zero).
transition(act(Id,A),act(Id,A),zero).
transition([Act],A,zero):-!,transition(Act,A,zero).
transition([Act,Act2],A,Act2):-!,transition(Act,A,zero).
transition([Act|S],A,S):-transition(Act,A,zero).
transition(S1+S2,A,R1):-transition(S1,A,R1).
transition(S1+S2,A,R2):-transition(S2,A,R2).
%Start reaction
reaction(out(move(Dialogue,Id,Act)),(
   in_r(dialoguestate(Dialogue,S)),
   out_r(transition(S,act(Id,Act),C,Dialogue))
)).
reaction(out_r(transition(S,A,S1,Dialogue)),(
   transition(S,A,S2), %make the state transition
   in_r(transition(S,A,S1,Dialogue)),
   out_r(dialoguestate(Dialogue,S2)),
   out_r(findall(S2,Dialogue))
)).
reaction(out_r(findall(S,Dialogue)),(
   in_r(findall(S,Dialogue)),
   findall(cs(Id,Commit),transition(S,cs(Id,Commit),Q),L), %collect all next commits
   out_r(nextcsmoves(Dialogue,L))
)).
reaction(out_r(nextcsmoves(D,[H|T])),(
   in_r(nextcsmoves(D,[H|T])),
   out_r(excommit(H)), %call execution commit
   out_r(looknext(D,T))
)).
reaction(out_r(looknext(D,[E])),(
   in_r(looknext(D,T)),
   out_r(nextcsmoves(D,T))
)).
reaction(out_r(looknext(D,T)),(
   T==[], in_r(looknext(D,[])),
   in_r(nextcsmoves(D,[]))
)).
%Implementation of K-OUT, K-IN and K-RD
reaction(out_r(excommit(cs(Id,out(A)))),(
   out_r(A), in_r(excommit(cs(Id,out(A)))),
   in_r(dialoguestate(Dialogue,S)),
   out_r(transition(S,cs(Id,Act),C,Dialogue))
)).
reaction(out_r(excommit(cs(Id,in(A)))),(
   in_r(A),  out_r(excommit(cs(Id,in(A)))),
   in_r(dialoguestate(Dialogue,S)),
   out_r(transition(S,cs(Id,Act),C,Dialogue))
)).
reaction(out_r(excommit(cs(Id,rd(A)))),(
   rd_r(A),  in_r(excommit(cs(Id,rd(A)))),
   in_r(dialoguestate(Dialogue,S)),
   out_r(transition(S,cs(Id,Act),C,Dialogue))
)).
```

Fig. 3. Control of Interaction: Checking agent legal locution, Making dialogue protocol transition and executing automatically *cs* actions are the basic function here implemented in ReSpecT

for a dialogue to be driven automatically by the state of the commitment store. FAs an example, Figure 2 shows the ReSpecT implementation of the $next^I$ operator.

6 Conclusions

In this paper we propose a conceptual architecture for a multi-agent dialogue system in which participants are assisted by a mediator, called a *Dialogue Artifact*. To the best of

our knowledge, there is no other research which combines at an operative level dialogue and argumentation reasoning through the use of a mediation artifact. The functions of such a mediator are the basic functionalities we have identified as part of a longer list of potential mediation or moderation functions in agent argumentation dialogues. Our Dialogue Artifact is an extension of our previous concept of a Co-Argumentation Artifact (CAA), and builds on that earlier work. We also draw on the recent theory of communication artifacts in MAS to formalise the properties of the Dialogue Artifact. Our paper also reported on a prototype implementation of these ideas we have undertaken in the TuCSoN coordination framework. In future work, we plan to formalise more of the potential mediator functions as listed in Section 2. While some of such functions will be straighforward to formalise – e.g., identifying conflicts between commitments, providing automated alerts to agents concerning upcoming dialogues – others, such as run-time assignment of rights and responsibilities to dialogue participants, are likely to result more challenging.

Also we aim at extending the underlying argumentation system by introducing *labels*. In fact, labelled arguments should make it possible to capture different sorts of certainty resulting from the different types of inference applied. Moreover, we plan to exploit labels to fix preferred ordering in an arguments set and to define stricter attack relation.

Acknowledgments

We are grateful for partial financial support from the EC's *Information Society Technologies* programme through project ASPIC (IST-FP6-002307).

References

1. Doutre, S., McBurney, P., Wooldridge, M.: Law-governed Linda as a semantics for agent dialogue protocols. In: Dignum, F., Dignum, V., Koenig, S., Kraus, S., Singh, M.P., Wooldridge, M. (eds.) 4rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), Utrecht, The Netherlands, July 25-29, 2005, pp. 1257–1258. ACM Press, New York (2005)
2. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77(2), 321–358 (1995)
3. Forester, J.: The Deliberative Practitioner: Encouraging Participatory Planning Processes. MIT Press, Cambridge (1999)
4. Gelernter, D.: Generative communication in Linda. ACM Transactions on Programming Languages and Systems 7(1), 80–112 (1985)
5. Gordon, T.F., Karacapilidis, N.: The Zeno argumentation framework. In: Proceedings of the Sixth International Conference on AI and Law, pp. 10–18. ACM Press, New York (1997)
6. Hamblin, C.L.: Fallacies. Methuen, London, UK (1970)
7. McBurney, P., Parsons, S.: Posit spaces: a performative theory of e-commerce. In: Wooldridge, M., Rosenschein, J.S., Sandholm, T., Yokoo, M. (eds.) Proceedings of AAMAS 2003, pp. 624–631. ACM Press, New York (2003)

8. Oliva, E., McBurney, P., Omicini, A.: Co-argumentation artifact for agent societies. In: Rahwan, I., Parsons, S., Reed, C. (eds.) Argumentation in Multi-Agent Systems. LNCS, vol. 4946, pp. 31–46. Springer, Heidelberg (2008)
9. Omicini, A.: Formal ReSpecT in the A&A perspective. In: Canal, C., Viroli, M. (eds.) 5th International Workshop on Foundations of Coordination Languages and Software Architectures (FOCLASA 2006), CONCUR 2006, Bonn, Germany, University of Málaga, Spain, August 31, 2006, pp. 93–115 (2006)
10. Omicini, A., Denti, E.: From tuple spaces to tuple centres. Science of Computer Programming 41(3), 277–294 (2001)
11. Omicini, A., Ricci, A., Viroli, M.: An algebraic approach for modelling organisation, roles and contexts in MAS. Applicable Algebra in Engineering, Communication and Computing 16(2-3), 151–178 (2005); Special Issue: Process Algebras and Multi-Agent Systems
12. Omicini, A., Ricci, A., Viroli, M.: Agens Faber: Toward a theory of artefacts for MAS. Electronic Notes in Theoretical Computer Sciences 150(3), 21–36 (2006); 1st International Workshop "Coordination and Organization" (CoOrg 2005), COORDINATION 2005, Namur, Belgium, . Proceedings, April 22 (2005)
13. Omicini, A., Zambonelli, F.: Coordination for Internet application development. Autonomous Agents and Multi-Agent Systems 2(3), 251–269 (1999)
14. Parsons, S., McBurney, P.: Argumentation-based communication between agents. In: Huget, M.-P. (ed.) Communication in Multiagent Systems. LNCS, vol. 2650, pp. 164–178. Springer, Heidelberg (2003)
15. Prakken, H.: Coherence and flexibility in dialogue games for argumentation. Journal of Logic and Computation 15(6), 1009–1040 (2005)
16. Prakken, H., Vreeswijk, G.: Logical systems for defeasible argumentation. In: Gabbay, D.M., Guenther, F. (eds.) Handbook of Philosophical Logic, vol. 4, pp. 219–318. Kluwer, Dordrecht (2002)
17. Ricci, A., Viroli, M., Omicini, A.: Programming MAS with artifacts. In: Bordini, R.H., Dastani, M., Dix, J., El Fallah Seghrouchni, A. (eds.) PROMAS 2005. LNCS (LNAI), vol. 3862, pp. 206–221. Springer, Heidelberg (2006)
18. Viroli, M., Ricci, A., Omicini, A.: Operating instructions for intelligent agent coordination. Knowledge Engineering Review 21(1), 49–69 (2006)
19. Walton, D.N., Krabbe, E.C.W.: Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning. SUNY Press (1996)

Co-ordination and Co-operation in Agent Systems: Social Laws and Argumentation

Katie Atkinson and Trevor Bench-Capon

Department of Computer Science
University of Liverpool
Liverpool L69 3BX, UK
{katie,tbc}@liverpool.ac.uk

Abstract. The social laws paradigm represents an important approach to the co-ordination of behaviour in multi-agent systems. In this paper we examine the relationship between social laws and rational behaviour, by which we mean behaviour that can be justified by a defensible argument. We describe how social laws have previously been defined and used within the context of Action-Based Alternating Transition Systems (AATSs). We then show how an account of argumentation for practical reasoning in agent systems, also based on AATSs, can be used to determine what is rational for the agents to do in the absence and presence of such laws. The reasoning involved is both of a practical and epistemic nature: agents need to make decisions about what to do based upon the assumptions that they make about the states they find themselves in, and crucially, they also need to reason about what the other agents in the scenario will do. What is rational for the agents to do has implications for the need for social laws, the ways in which social laws can help the situation, the form the social laws should take, and the likelihood of compliance with the social laws. This paper demonstrates how we can think about social laws and rational behaviour in a single framework, so as to identify these implications in particular scenarios, and so frame social laws accordingly.

1 Introduction

Co-ordination within multi agent systems can be addressed through numerous different approaches. One important approach is through the use of social laws (e.g. [10][9]) that constrain the behaviour of agents within a scenario so that compliance with the law ensures that either some particular undesirable state is avoided or that some desirable state is eventually reached. In practice the realisation of social laws takes a variety of forms, ranging from mere conventions of etiquette, through moral conventions, to laws which have legislative force. In [12] it has been shown that such laws can be effectively expressed and understood using Action-based Alternating Transition Systems (AATSs) and Alternating-time Temporal Logic (ATL). In the absence of such laws, however, agents will not behave arbitrarily. In some cases, it may be enough for agents to behave rationally to guarantee the desired outcomes. In others the laws may be essential to guide

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 122–140, 2009.

the behaviour of the agents. In still others it may be rational for one or more agents to violate the laws, potentially rendering them ineffective. Thus the need for, the benefits of, the form of, and the effectiveness of, social laws all require some consideration of what is rational for agents to do in the various situations, both where there are social laws and in the "state of nature" without them. In this paper we demonstrate how an argumentation based approach to practical reasoning, also based on AATSs, can be used to determine what is rational for the agents to do in the absence of hard constraints enforcing such social laws. In doing so we consider a particular example, taken from [12], concerning the co-ordination of the movement of two trains. Using this example we consider how the reasoning differs depending upon the view that other agents take of their counterparts within the scenario, both what these other agents are likely to do, and the degree to which the interests of the other agents are respected. Our approach differs from that of Castelfranchi [7], in which social action is considered in terms of agents adopting the goals of others. On our view agents do not adopt goals of other agents, although their actions may further these goals, but rather choose to constrain their actions so as to enhance, or at least not threaten, the interests of other agents. Another characterisation of selfish and social agents is given in the context of the BOID architecture [6]. There agents are considered to have obligations as well as the standard beliefs, desires and intentions, and a selfish agent is one which prefers its desires to its obligations, and a social agent one which prefers its obligations to its desires. In our approach these conflicts are not resolved by a policy of this sort, but by considering the effect on the interests of the agents concerned. For us, actions are justified through the promotion of some social value, and these values may be promoted in respect of the agent itself or some other agent. For example, the promotion of liberty is always through an increase in the liberty of some particular agent. Selfish agents are, on our account, those who place most weight on values promoted in respect of themselves: they will prefer to promote a relatively unimportant value enhanced in respect of themselves over even important values promoted in respect of others: for example they may be willing to seek their own happiness even when it may jeopardise the safety of other agents.

The rest of the paper is structured as follows. In Section 2 we provide the details of the example scenario that we will use, which is taken from [12]. In Section 3 we briefly describe the background theory of practical reasoning that we use to enable agents to formulate and critique arguments about what to do, and so provide justifications of their actions. In Section 4 we show how this theory can be used to drive the reasoning in the scenario. In Section 5 we provide a general discussion of social laws that draws on our examples and covers consideration of when they are required, how they operate and what form they should take. Section 6 finishes the paper with some concluding remarks.

2 Social Laws

In [12] van der Hoek et al. make use of Action-Based Alternating Transition Systems (AATSs) to explore social laws as a means of co-ordinating multi-agent

systems, and we will use their notation. In an AATS transitions between states are governed by joint actions which are composed from the individual actions of the agents involved. A formal definition of an AATS is given by van der Hoek et al. in [12]. It is their example that we will make use of here to extend the exploration to encompass consideration of what is rational for the agents to do in various scenarios.

The example has two trains, one running eastwards and one running westwards. For most of the circuit each train has its own track, but this narrows to a single track shared by the trains where the track enters a narrow tunnel. If both trains enter the tunnel together, therefore, they will crash. The trains may be in one of three states, *away* from the tunnel, *waiting* to enter the tunnel, or *in* the tunnel. At each point they may move (away to waiting to in to away) or stay still. Two particular aspects of the scenario are relevant: *safety*, i.e. ensuring that there is no crash, and *progress*, i.e. ensuring that the trains keep moving as much as possible whilst avoiding a crash. Initially they are both away from the tunnel. The transitions of the AATS for the scenario are shown in Table 1. Here the nine states in the scenario are labelled q0–q8. In each state each agent may choose one of two actions, move or do nothing. When the actions of the two agents are combined, this results in four joint actions: j0, where both trains do nothing; j1, where the eastbound train does nothing but the westbound train moves; j2, where the eastbound train moves but the westbound train does nothing; and j3, where both trains move. The final column of the table gives the interpretation function that shows which propositions are true in each state (away, waiting or in) subscripted for each train.

The transitions can be shown diagramatically, as in Figure 1 below. In the diagram each of the states is labelled with its number (one of q0–q8) in the bottom right hand corner. Each state also contains two propositions to determine the status of each train: the top proposition represents the eastbound train's status, the bottom proposition the westbound's. Each proposition can be set to either 0, when the train is away, 1, when the train is waiting to enter the tunnel, or 2, when the train is in the tunnel. The arcs are labelled with the joint actions, as described in Table 1.

Table 1. Transitions/Pre-conditions/Interpretation

q/j	j_0	j_1	j_2	j_3	π(q)
q_0	q_0	q_1	q_3	q_5	{away$_E$, away$_W$}
q_1	q_1	q_2	q_5	q_6	{away$_E$, waiting$_W$}
q_2	q_2	q_0	q_6	q_3	{away$_E$, in$_W$}
q_3	q_3	q_5	q_4	q_7	{waiting$_E$, away$_W$}
q_4	q_4	q_7	q_0	q_1	{in$_E$, away$_W$}
q_5	q_5	q_6	q_7	q_8	{waiting$_E$, waiting$_W$}
q_6	q_6	q_3	q_8	q_4	{waiting$_E$, in$_W$}
q_7	q_7	q_8	q_1	q_2	{in$_E$, waiting$_W$}
q_8	q_8	–	–	--	{in$_E$, in$_W$}

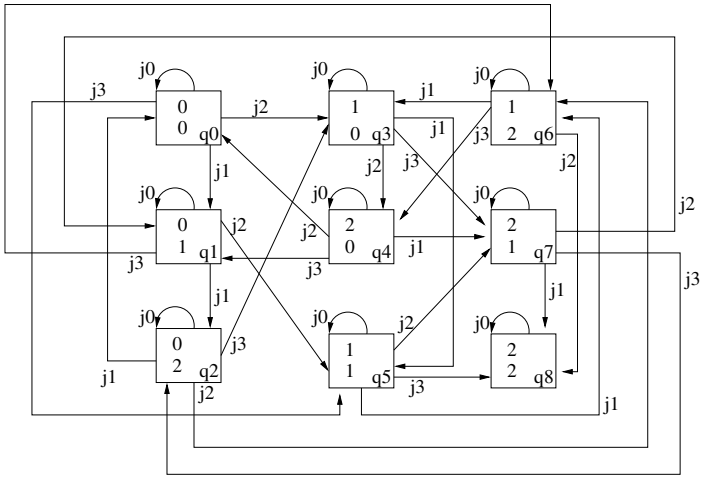


Fig. 1. State transition diagram for scenario

In general, the social laws approach is intended to constrain the behaviour of agents in particular states, so as to achieve certain objectives, typically that some state is avoided, or that some state is eventually reached. A social law is said to be *effective* if compliance with the law ensures that the objectives are achieved. The main objective of the above example is to ensure that there is no collision, that is, that state q8 is never reached, with a secondary objective that the trains are able to make progress and so able to reach any of the states away, in and waiting. The undesirable state q8 can only be reached from one of the three states q5, q6 and q7. So, to ensure that q8 is not reached, a social law is needed to restrict the behaviour of the agents in these three cases. One such law proposed and formalised in [12], which we will call SL1, is as follows:

1. when both trains are waiting (q5) the eastbound train should not move;
2. when the westbound train is in the tunnel and the eastbound is waiting (q6) the eastbound train should not move;
3. when the eastbound train is in the tunnel and the westbound is waiting (q7), then the westbound train should not move.

As is shown in [12] this social law is effective, in the sense that if it is obeyed, it will ensure that the trains do not collide. As noted there, however, this law is asymmetric, in that it favours the westbound train over the eastbound train (although a very similar social law could be made which favoured the eastbound train, by modifying the first condition). Note that SL1 does not guarantee that any progress will be made since a train could remain in the tunnel indefinitely without violating SL1. SL1 thus seems to assume that agents will choose to move from the tunnel at the first opportunity, and so no social law is required to ensure that they do so. We will also make this assumption and focus our subsequent discussion on state q5, as representative of a state in which there is a danger of

collision. Of course, we could make a social law to ensure that this assumption is satisfied, consistent with the conditions of SL1, by additionally insisting that the trains should move whenever they are in the tunnel.

The argumentation based model of practical reasoning that we use in this paper enables us to evaluate the law in terms of what the agents would choose to do in the absence of any social law. So, we view the situation as a practical reasoning problem. In our account there is no need for the agents to reason about each others' beliefs since the relevant situation is fully described in the publicly available structure of the AATS. Moreover, in our example, agents have perfect information as to the situation and so disagreements as to how it is represented in the AATS do not arise. For this reason we have no need to use epistemic logic.

In the next section we provide an overview of the argumentation based approach that we make use of to model the problem.

3 Background Theory of Practical Reasoning

In [2] an argument scheme and associated critical questions are presented to enable agents to propose, attack and defend justifications for action. Such an argument scheme follows Walton [13] in viewing reasoning about action (practical reasoning) as presumptive justification - *prima facie* justifications of actions can be presented as instantiations of an appropriate argument scheme, and then critical questions characteristic of the scheme used can be posed to challenge these justifications, which can be used to overturn the presumption. The argument scheme AS1 developed by Atkinson [2] is an extension of Walton's *sufficient condition scheme for practical reasoning* [13]. AS1 is stated as follows:

AS1 In the current circumstances R,
We should perform action A,
Which will result in new circumstances S,
Which will realise goal G,
Which will promote some value V.

In this scheme Walton's notion of a goal has been made more precise by distinguishing three elements it encompasses: the state of affairs brought about by the action; the goal proper (the desired features in that state of affairs); and the value (the reason why those features are desirable)[1].

The underlying idea in making this distinction is that the agent performs an action to move from one state of affairs to another. The new state of affairs may have many differences from the current state of affairs, and it may be that only some of them are actively desired by the agent. The significance of these differences is that they make the new state of affairs better with respect to some good valued by the agent and typically the new state of affairs will be better through improving the lot of some *particular* agent.

[1] In this sense values represent the social interests promoted through achieving the goal. Thus they are qualitative, as opposed to quantitative, measures of the desirability of a goal.

Agents act so as to bring about states of affairs that promote the particular values that are of concern to the individual agents. Thus, each agent has a preference ordering on the values it considers relevant in the particular scenario. We can therefore characterise agents' behaviour with respect to the ordering that they place on values. As mentioned previously, the two values of concern in the train scenario are 'safety' and 'progress'. *Prudent* agents will place a higher value on safety, but *reckless* agents will value progress more highly. In the examples that we discuss in Section 4 we use the terms *selfish* and *moral* to describe the behaviour of agents, following [3]. Note that when a value is promoted it is done so in virtue of one agent or both agents. Thus progress can be promoted in virtue of the eastbound train moving, the westbound train moving, or both moving. Similarly safety is demoted when the eastbound train crashes, the westbound train crashes, or both crash. We therefore subscript values to show in virtue of which agent progress and safety are affected. Selfish agents prefer their own interests to those of others and thus they will rank promotion of values in respect of themselves more highly than any values promoted in respect of others. For example, considering the value ordering from the perspective of a selfish prudent eastbound agent will give us the following: $\text{safety}_E > \text{progress}_E > \text{safety}_W > \text{progress}_W$. Moral agents, on the other hand, will take the other agents' interests into account, and so a prudent moral agent will believe that safety promoted in respect of others is more important than its own progress. A truly moral agent will give the value equal rank, whichever agent it is promoted in respect of, but, as discussed in [3], it remains morally acceptable to prefer promotion of a value in respect of oneself to promotion of that value in respect of others, provided the ordering of values is consistent. Note that moral agents need not be *sacrificial* (i.e. they do not favour promoting a value in respect of the other agent over promoting it in respect of themselves). Note also that the values are ordered according to the preference of the *agent itself*: a prudent agent is not required to rank the progress of a reckless agent more highly than that agent's safety, even though that is the preference of the other agent. Nor is a moral agent required to adopt values of the other agent if it does not recognise their worth: if the westbound train had a value "excitement" promoted by near collisions, the eastbound train would be under no moral compulsion to consider that to be a value. Thus, considering the value ordering from the perspective of a moral prudent eastbound agent will give us the following: $(\text{safety}_E = \text{safety}_W) > (\text{progress}_E = \text{progress}_W)$.

Associated with AS1 are seventeen different critical questions [4] that challenge the presumptions in instantiations of AS1. Each critical question can be seen as an attack on the argument it is posed against and examples of such critical questions are: "Are the circumstances as described?", "Does the goal promote the value?", "Are there alternative actions that need to be considered?". The full list of critical questions, and their interpretation in terms of AATS, can be found in [4]. In the next section we make use of a selection of these critical questions and upon doing so we will make clear the attack that the critical question is asserting.

Using argument scheme AS1 and its associated critical questions to produce arguments for reasoning about matters of practical action, we should expect to see one or more *prima facie* justifications advanced stating, explicitly or implicitly, the current situation, an action, the situation envisaged to result from the action, the features of that situation for which the action was performed and the value promoted by the action, together with negative answers to critical questions directed at those claims. In the example that we provide in the next section the argumentation is expressed in natural language terms for ease of understanding, though we note that the machinery to express these arguments formally for use with an AATS is given in [4] and it would be a simple task to express them in this notation.

Finally, we note that the argument scheme AS1 can also be used in a negative form (AS2):

AS2 In a particular set of circumstances R,
We should not perform action A,
As it would lead to a particular state of affairs S
Which entails some 'goal' G,
Which demotes some value V.

This negative version of AS1 can thus be used in scenarios where the onus is on avoiding some undesirable outcome rather than achieving some positive outcome, as is the case in the examples we present in the next section.

4 Example

Using the example scenario described in Section 2, we now show how the two agents in this scenario, the eastbound train and the westbound train, will each reason about what to do, in the absence of constraints, by instantiating the argument schemes AS1 and AS2, and posing the appropriate critical questions. We do so in respect of a number of different scenarios, based on the different possible value orderings that determine the agents' behaviour. We make the assumption in each case in this scenario that perfect information is available to both trains so that there is no epistemic uncertainty as to the current state. Now, in order to decide what to do, each agent will need to consider how it can promote its values, taking into consideration what the other agent will do. Of course, not all agents will act in the same way in a given scenario, and this will be reflected in the weight given to the arguments. We consider a number of different scenarios in turn, though the objective in every case is to avoid collision, i.e. avoid state q8.

4.1 Scenario 1: Reasoning in the Absence of Social Laws

We begin by considering how agents in the scenario will act in the absence of any social law. The reasoning starts in the initial state q0 where both trains are away. In q0 there are no controversial decisions to be made, each train may move

knowing that the other is away. Each agent can thus instantiate an argument for moving, and can assume that the other agent will reason in the same way. However, since it is not possible to reach the state in which collision occurs from q0, the action of the other agent will not make an important difference and so need not be of real concern. Thus, the eastbound train will instantiate an argument as follows:

Arg1: In state q0, I should perform j2, to achieve waiting status in q3, promoting the value progress.

No attack can be successfully used against this argument since even if the assumption that the other agent will also choose to move is proved incorrect, the resulting state is equally good as far the eastbound train is concerned. We can see that Arg1 will similarly hold for states q1–q4 since none of the actions that can be performed in these states lead to the undesirable q8, regardless of how the other agent acts. Thus in each of these cases progress can be pursued without risk. The remaining states, q5–q7, can however, lead to q8. This is shown in Figure 2 below, which gives the subset of the scenario containing the states from which a collision can occur.

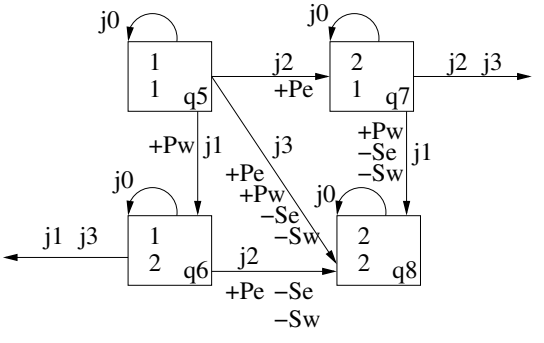


Fig. 2. State transition diagram for states q5–q8

The arrows from states q6 and q7 that do not lead to any states signify that if these actions are performed, they will lead back to safe states (which are not of concern for this part of the scenario). Additionally, as in [3], in this diagram the transitions are labelled with the relevant values that are promoted or demoted by the transitions, with respect to each agent. Recall, the two values of concern are progress (P) and safety (S). Thus, for example, where a transition is labelled by +Pe, this indicates that progress is promoted for the eastbound train. For reasons of space we omit value labels from transitions that are neutral with respect to value promotion (i.e. where the transition neither promotes nor demotes safety or progress for either agent).

Selfish Agents. Let us consider first the state q5 from the perspective of the eastbound train using the assumption that both agents are selfish. Since selfishness means that the agent will want to better its own interests, it will instantiate

an argument to move in order to promote progress. When there is more than one action that will promote progress, the agent will choose the one (if any) that does not demote some other value. So, considering the eastbound train in q5, the following argument will be put forward:

Arg2: In state q5, I should perform j2, to reach q7 and enter tunnel, promoting the value progress.

We can immediately see that this argument poses a problem if the other agent, reasoning in the same way, also decides to move, since this will result in j3, rather than j2 being performed, which leads to the collision in q8. We can thus pose a critical question against this instantiation. The particular critical questions that is applicable here is CQ17: *can the other agent be guaranteed to perform its part of the joint action?*[2]

Obj1: Agent W cannot be guaranteed to act so as to execute j2.

Remembering that each agent in this scenario expects the other to act in a selfish way and thus move, we can see that the objection raised in this critical question is upheld; the agent can be expected to act in this way since, being selfish, it will want to promote its progress by moving, so j3 will be executed. Thus, Arg2 is defeated by the attack of Obj1 and will be abandoned. Now the eastbound agent must consider whether there is any argument for executing j3:

Arg3: In state q5, I should perform j3, to reach q8 and enter tunnel, promoting the value progress.

Whilst we can see that executing j3 will indeed promote progress, it will, however, lead to the state in which collision occurs. Thus we can critically question Arg3, since the action has a side effect that demotes another value. CQ9 raises such an objection:

Obj2: Action j3 has a side effect that demotes the value safety.

Again, we can see that this objection is upheld and, since normally safety is more important than progress, Arg3 will be abandoned. This leaves only one choice for the eastbound train: to stay still. The agent will be indifferent as to whether j0 or j1 is executed since both have the same effect with respect to value promotion, each being neutral with respect to both its values. Whilst j1 does in fact promote progress for the westbound train, this has no influence on the eastbound train since it reasons in a selfish manner. If we consider the possibility that the eastbound train would choose to execute j0, this argument would again be subject to questioning through CQ17. In response to this we can say that the westbound train, being selfish, will actually prefer to move and so state q6 will be reached with no detriment. This would bring the reasoning round full circle since the westbound train would then need to consider whether the other agent would act so as to guarantee that j1 would be executed. We can thus see that

[2] Arguments that instantiate a critical question are labelled with 'Obj', to distinguish them from those arguments that instantiate an argument scheme ('Args').

that a symmetric set of arguments to those given above would be generated by the westbound train. The overall result of the reasoning would mean that each train would remain still, as shown through the following argument:

Arg4: In state q5, I should perform j0, which would avoid collision, so as not to demote the value safety.

No critical question can be successfully posed against this argument. However, again, although the collision is avoided for prudent agents, the reasoning results in an undesirable situation since deadlock is created as neither train has an argument to move. It is from the need to avoid this effect that the requirement for a social law arises. In the absence of such a law the deadlock is broken only when one of the agents becomes sufficiently reckless to prefer progress to safety. Should both agents do so at the same time, a collision will occur.

So far we have assumed that the agents in this scenario are all acting selfishly. We should therefore consider if the outcome of the reasoning would be any different if the agents are not in fact selfish, but instead 'moral' agents. That is, they are not selfish, but neither are they sacrificial in that they do not favour the other agent's interests over their own.

Moral Agents. Starting over where the initial state is q0, there are again no potentially dangerous actions so each agent will choose to move. The problem states remain q5–q7. Again in q5, each agent will have an argument to move, as in the previous scenario, and Arg2 will be put forward. Likewise, CQ17 can again be posed to state that the other agent cannot be guaranteed to act so as to execute j2. However, this time when the eastbound train considers that this non-compliance with j2 will lead to j3 being executed and subsequently q8 will be reached, it will, unlike in the previous scenario, consider the values of the other agent. So, whilst the same line of reasoning will still apply, the eastbound train will now have an *additional* attack, through the use of CQ9, that can be posed against Arg3:

Obj3: Action j3 has a side effect that demotes the value safety of the westbound train.

In order to see how this extra argument affects the evaluation of the set of arguments, we can organise them into a value-based argumentation framework (VAF) [5], which is an extension to Dung's abstract Argumentation Frameworks (AFs) [8]. AFs provide a means of evaluating the acceptability of a set of arguments in terms of the attack relations between them. VAFs extend Dung's AFs to accommodate different *audiences* with different values and interests. Within a VAF, which arguments are accepted depends on the ranking that the audience (characterised by a particular preference ordering on the values) to which they are addressed gives to the values motivating the argument. In essence, attacks are removed if the value of the attacking argument is ranked below the value of the attacked argument. The VAF for the arguments relevant to this scenario, from the viewpoint of the eastbound train, is given in Figure 3.

To evaluate the status of the arguments we need to consider the preference ordering on values to resolve the conflicts between the arguments. As stated

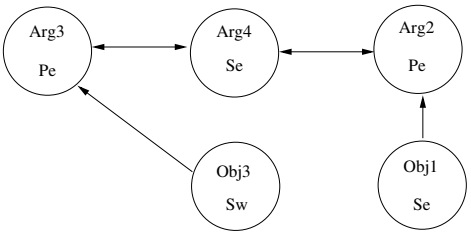


Fig. 3. VAF for moral agents

previously, safety is preferred to progress in all cases, thus Arg3 will always be defeated. The point to note however, is that the attack of Obj3 on Arg3 succeeds in this scenario since moral agents rank values without considering which agent they are promoted in respect of. But, this particular attack would not succeed for the previous case where the agents are all selfish, even though Arg3 would still be defeated by Arg4, assuming the agents are prudent. The result of the reasoning here is that both trains will again remain still, but this time there is an additional reason, which will prevent even a reckless agent from moving, provided it is moral enough to consider the other agent's safety, even if it does not care about its own. Thus, although the reasoning on the parts of both agents will avoid the undesirable q8, here a social law is essential, not to avoid collision, but in order to break the deadlock. In this case, where the agents are considerate of each other in this way, the need for a social law is greater, since even recklessness will not resolve the impasse.

The scenarios discussed so far consider a situation which was symmetrical with respect to the interests of the agents. Now suppose we alter the scenario so that one of the agents, say the westbound, is instead a pedestrian and both agents are again assumed to act selfishly. In this case, the state to avoid remains q8, but the reasons for avoiding it have now altered since a collision would only impact negatively upon the pedestrian (where 'safety' and 'progress' remain the only two values of concern): the train would be unharmed by the collision.

Considering again the problematic q5, the reasoning of the eastbound agent will again begin with the proposal of Arg2, against which CQ17 can be posed. In this case the train's safety is not now compromised in q8: the transition between q5 and q8 will be labelled only with the value 'safety' demoted in respect of the pedestrian. Thus, the eastbound train is aware that the pedestrian, not willing to compromise his safety, will not move, leaving the train free to enter the tunnel. But, now the critical question CQ9 does not apply, since even if the pedestrian ignores the risk, the resulting situation will not demote the safety of the train. In this case the rationality of the agents should be enough to ensure that q8 is avoided and the need for a social law to enforce such behaviour is dispelled, since the agents will behave in compliance with SL1 anyway. However, we may wish to actually enforce such behaviour through the issue of a social law since consideration must be given to the situation where an agent may be reckless and a collision ensues. The law now, however, is for the pedestrian's own good,

rather than to provide the co-ordination required in the case of two trains. In such a case we may wish to implement punishments or sanctions against such behaviour, as we discuss later in Scenario 3.

The previous example considers the reasoning of the agents where one is a pedestrian, the other a train and both are selfish. Does the outcome change if the agents are moral? Again, considering the problematic state q5, we can see that the reasoning of the train will begin by it proposing Arg2, against which CQ17 can again be posed. Once more, the train would expect the pedestrian to comply with not entering the tunnel, yet in this case the train will act in consideration of the pedestrian's values in addition to its own. Thus, the VAF for the situation will be updated so that the argument based on the eastbound train's safety no longer appears in it. Here the train will not move since the argument for moving is defeated by the argument demoting the pedestrian's safety. However, the pedestrian, also reasoning that the danger only concerns himself, will not move either, leading to a deadlock situation. In this case a social law is clearly needed to avoid deadlock, even if only the train is moral.

4.2 Scenario 2: Reasoning in the Presence of Social Laws

We now consider how the reasoning will change when a social law is present. As demonstrated above, reasoning in the scenario in the absence of a social law will indeed avoid the undesirable state, but a deadlock situation arises. This is true for both the case in which the agents are selfish, and that in which they act in accordance with 'morality'. Now let us consider the effect of introducing the social law SL1, as stated in Section 2. When such a social law is in place, the agents in the scenario have a change in information about the actions of each other. Thus all agents may still act selfishly, i.e. in accordance with their own interests, but there is now an assumption that the other agents will all obey the law. This effectively excludes certain joint actions (those containing the prohibited action) from the AATS. This makes j3 unavailable in q5 (SL1.1), j2 unavailable in q6 (SL1.2) and j1 unavailable in q7 (SL1.3). So, for our problematic situation, state q5, the social law ensures that the agents will not act so as to end up in q8, and the deadlock is broken through the law specifying which agent should move into the tunnel first. This means that the westbound train will generate an argument for moving:

Arg5: In state q5, I should perform j1, to reach q6 and enter tunnel, promoting the value progress.

CQ17 can still be posed against this argument to test the presumption that the eastbound train can be guaranteed to act so as to execute j1. However, the response to this argument is now that the eastbound train will act so as to execute j1, because it will obey the social law and so cannot bring about the performance of j3. But, as noted previously, SL1 is asymmetric in that it favours the westbound train over the eastbound train. Nonetheless, even though the agents are not treated equally by the law, it does in fact provide more benefit to them both than the case where there is no law. Since the choices that are forced

through the law are rational in any case, it follows that both trains will move sooner, and without the need to degenerate into recklessness, than they would if the social law was not in place. In this way, adherence to the law is reinforced through rationality. The same outcome is also true for the situation in which the agents are 'moral' as opposed to selfish.

If we now return to the example where one agent is a pedestrian and the other a train, we noted previously that a social law is required where the agents are acting morally. Again, we consider the application of SL1 in this situation. As before, the law works so as to remove the deadlock, but since a choice must be made as to which party will get to move first, the law will again favour one of the parties. However, in order to reinforce the behaviour that rationality suggests, the choice of who goes first in this case should not be an arbitrary one. Here, the social law should be defined so as to allow the train to move first, since it is the party against which no danger is posed. Now the moral train can enter the tunnel assured that the pedestrian will wait, and any threat to the pedestrian's safety comes from the pedestrian's own disobedience. If, on the other hand, the law tried to make the train wait, the train would have no reason other than conforming to the law to wait, and so there would be temptation to violate the law. Moreover, the pedestrian might be reluctant to jeopardise his safety by trusting that the train would comply. Such a law might therefore lead to collisions when the train disregarded the law, and to deadlocks when the pedestrian did not trust the train to comply. By reinforcing rather than conflicting with the rational choice, the law is more likely to be followed since it does not penalise the party who has a rational justification for non-compliance. This suggests that in general when framing the law we should consider which parties benefit the most when compared with the situations without the law, if the laws are to be as effective as possible.

4.3 Scenario 3: Social Laws with Sanctions

As noted above, the presence of social laws should prescribe[3] the behaviour of the agents. However, since the agents are autonomous, they cannot always be guaranteed to adhere to the social laws in place. Thus, we consider how obedience to the law can be achieved through the use of sanctions. Sanctions can take two forms; they may operate in relation to a value representing the stigma associated with violating the law, or they may operate through undesirable consequences relating to the state reached when the law is violated i.e. the agent is in some way punished for violating the law. In the presence of these new elements we now consider how the reasoning in the scenario will differ in state q5.

We begin with the case of the selfish agents. In q5 SL1 states that the east-bound train should remain stationary. However, this agent, as in the previous scenarios, will have an argument for moving into the tunnel based on pursuit of

[3] Although we make use of deontic notions such as obligations and their violations, we do not give any precise characterisation of them here. The relationship between deontic logic and ATL is the topic of [14] which introduces Normative ATL and provides definitions of obligation and permission in terms of an AATS.

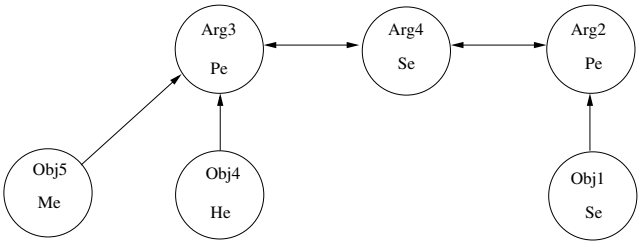


Fig. 4. VAF for scenario with sanctions

progress. There remains an argument against moving, but a reckless agent may be tempted to ignore this and move in an attempt to get into the tunnel first, thus violating the law. However, we can now introduce the third value of 'honour' into the scenario, whereby any transition that ignores the law demotes this value. All actions that are executed in adherence with the law will promote the value. So, where there may be temptation to break the law, e.g. the eastbound train does not want to wait in q5, there is now another critical question, based upon the demotion of another value, that can be posed against the argument to move:

Obj4: Action j2 has a side effect that demotes the value honour.

In the simple scenario considered here it may be that honour does not feature highly in a value ordering and thus Arg3 will resist the attack of Obj4. However, in a more complex scenario where honour plays a role in future interactions, this may be enough of a sanction to deter violation. As an additional, or alternative, to this form of sanction we may also take a quantitative measure into account. We can thus add a proposition to each of the states to represent some kind of monetary possession. Here, where an agent violates the social law the state reached will actually be different to that intended through the application of the sanction, as expressed through critical question CQ2:

Obj5: Action j2, does not lead to q7.

Of course, with the addition of this proposition the state transition diagram will now need to be altered to show in which states money is decreased for each agent. So, the losses made in such situations should be enough to deter the agent from violating the law here, if money is ranked higher than progress. We can see that this is the case by considering the VAF for the arguments shown in Figure 4, where the values 'honour' (H) and 'money' (M) are introduced to ground the appropriate arguments.

The above view is for that of selfish agents, so we again consider how the reasoning changes in the case of moral agents, for the scenario where there is a social law with sanctions. Here, unlike the case of selfish agents, the temptation to violate the law will be removed since the agents take each others' interests into account; an objection based on CQ9 can again be posed against Arg3 to

state demotion of the other agent's safety, which will be enough to stop the violation.

However, we note that the above holds for this particular scenario because there are no conflicts *within* a value, i.e. in our scenario safety always trumps progress and there is no situation here where an agent is forced to choose between its own safety and the other agent's, nor its own progress and the other agent's. There are, however, other example scenarios where such a choice is required, and these in turn could lead to the temptation to violate a sanction. In such cases we need need to make a distinction between different levels of morality, in order to resolve the conflict. One such account of these different levels has been given in [3]. There, a distinction is made between 'moral' agents, as we have used in our example where values are ordered but within each value agents are treated equally, and 'noble' agents, where values are ordered in a moral sense, but within a value an agent prefers the other's interests. For example, a noble eastbound agent would order values as follows: $\text{safety}_W > \text{safety}_E > \text{progress}_W > \text{progress}_E$. There are numerous everyday examples that can be cited in which such a distinction is required. Consider the scenario of being sat on a train in which all the seats are occupied and a pregnant woman boards the train. In such a case, the norms of society are such that it is expected that a person with no impediment would give up their seat for the woman. Whilst a moral agent would value his own comfort equally to other peoples', he would be required to be noble, i.e. value the comfort of a pregnant woman over his own, in order to be forced to act and give up his seat. Whilst temptation to violate such a norm would not exist in the case of a noble agent, for the moral agent, whose value ordering conforms to a lesser standard of morality, the temptation would arise. Here, a sanction based upon demotion of honour could again be employed in order to force compliance.

Finally, concerning sanctions, there is one further problem to be considered before too much reliance is placed on them: sanctions require that the transgression be detected and the punishment enforced. But this will not always be the case. Sanctions therefore require an additional agent, the agent responsible for enforcing the social laws, to be modelled in the system, and additional joint actions since the sanction may or may not be enforced. In many cases, this reintroduces uncertainty into the situation, since the agent cannot know that the transgression will be followed by the sanction. So an agent reasoning in the presence of sanctions will still have an argument for the transgressive act, albeit one subject to CQ17, since the sanction may be enforced and the expected state not reached. The agent's decision will then need to balance the risk of detection against the gains resulting from transgression. In such cases other agents similarly will not be as sure of their assumption that the law will be complied with, since conformity now requires a degree of judgment on the part of the other agent. In many cases therefore, where detection is not assured, the social laws fail to provide the essential increase in certainty as to how others will behave.

5 Discussion

From the above considerations, we can attempt to draw some generalisations, about when social laws are required, how they operate and the form they should take. We will draw upon the above discussions, and additionally add some illustrations from road traffic practice.

In some cases no social law is really needed, as illustrated above where the social law requires the pedestrian to give way to the train, since the agents will avoid the collision, provided they act in their own interest and expect the other agent to do so. This is the situation with regard to pedestrians crossing the road: since prudence should lead them to avoid crossing in front of cars, no law is necessary, although parents do try to teach their children to value safety over progress.

In other cases, represented in the example by the scenario with two trains, a social law is needed, since agents will be prevented from acting freely in accordance with their preferences because of uncertainty about what the other agent will do. What the social law does is to remove this uncertainty: since the social law requires the eastbound train to give way, the westbound train can confidently enter the tunnel. In road traffic scenarios, this gives rise to conventions as to which side of the road should be driven on: it is crucial that the agents know what the others will do if collisions are to be avoided. Of course, such a convention does constrain behaviour but agents will willingly accept the constraint since everyone gains: the preference is to drive on the same side as others, not on a particular side. While in this case the effect of the law is the same for both agents, in the train case the westbound train gains more, since it is effectively enabled to act in accordance with its best interests without fear that the eastbound train will act so as to endanger it. Nonetheless, the eastbound train also gains, since the social law assures it of eventual progress without danger, whereas, without the social law, progress cannot be made without an agent becoming sufficiently reckless to desire progress even at the risk of its safety. The role of the social law here is purely one of *co-ordination*, which is enabled by increasing confidence in what the other agent will do.

In other situations agents are constrained with no advantage to themselves. The example scenario is where the train is required to give way to the pedestrian. Since the train is in no danger from the collision, the arguments to act in conformity with the law relate to benefits to the pedestrian, not the agent which is constrained. On roads this is the situation with pedestrian crossings: it is concern for the safety of the pedestrian that should induce the car to comply with the convention that the pedestrian has right of way in such cases. The social law is socially justified in that it gives a substantial benefit to one agent at a small cost to the other, but compliance does rely on a degree of moral sense on the part of the car drivers. In consequence pedestrians often hesitate at crossings, particularly if a car is approaching at speed, since they do not trust the driver to comply with the law.

Of course, pedestrian crossings are backed with the force of law. Nonetheless, observance of pedestrians rights here is largely self regulating: prosecutions are unlikely unless an accident actually occurs, and pedestrians are typically

sufficiently wary when using them that this is a rare occurrence. Compliance is, however, reinforced by the wish to avoid the stigma of being thought inconsiderate, and the possibility of legal sanction. In other cases, such as for example, road tax, sanctions take on more importance: this law is frequently flouted, and would be flouted even more but for likelihood of prosecution. Self regulation is possibly sufficient when the loss is small and the benefits to others are obvious, but as the constraints become more burdensome and the benefits less immediately visible the need for sanctions increases.

Social laws essentially work by reducing uncertainty as to what others will do in a given situation, thus allowing the consequences of one's own actions to be more predictable. Sometimes there will be gains for everyone, so that selfish agents will, given the assurance about how others will act, rationally conform. In other cases it may be necessary for the agents to consider values promoted in respect of others to motivate conformity. This is illustrated by advertisements encouraging conformity to speed limits which emphasise the danger to oneself in the case of motorways, and the danger to others in the case of low speed limits in residential areas. The former can appeal to self interest, but the latter requires a sensitivity to the danger to others. In some extreme cases – military draft in wartime may be an example -- conformity requires the agent to put the interests of others before of its own most valued interests, and here sanctions will normally be essential to ensure conformity.

In her work on emergence of norms, Ullmann-Margalit [11] distinguishes norms of co-ordination, where both parties benefit from compliance, from norms of co-operation where an increase in the common good comes at the price of a decrease of individual goods. This is the situation represented by the classic game theory scenario known as the Prisoner's Dilemma, and she refers to such norms as PD-norms.

Co-ordination norms are unproblematic, since given the resolution of uncertainties offered by the social law, rational agents will freely choose to comply with them. For PD-norms, however, the rational situation is defection, not compliance, as is well established in game theory. Ullmann-Margalit suggests three ways of inducing compliance: making defection impossible; making defection unattractive through sanctions; and, what she calls "honour", which involves a sufficiently strong sense of identity with the other agent to mean that the interests of the other are given sufficient weight to induce co-operation.

The first method, making violation impossible, is the approach typically taken in Electronic Institutions e.g. [1]. There, for example, if a participant in an auction is not permitted to bid according to the norms of the institution, this action is simply unavailable. While this is possible in a structured situation such as is provided by an electronic institution, there are several problems with this as a general solution. First it violates the autonomy of the agents: they are forced to obey the norm, and so their freedom is constrained. Secondly agent interaction in open systems is desirable in less structured contexts, when it is impossible to impose these constraints. But most importantly, it is part of the nature of social laws that there are occasions when it is desirable that they are violated.

Occasionally it is necessary to drive on the wrong side of the road to avoid an accident: we would not want this to be impossible. In a medical emergency we may not only allow, but desire, speed limits to be exceeded. In complex environments norms can conflict, and we would wish our agents to solve this conflict rationally with regard to the particular situation, rather than blindly following the norms.

Sanctions, as mentioned above, can be effective, given a regime in which detection is sufficiently certain and the punishments sufficiently great. This requirement, however, may be very difficult to achieve in a loosely structured environment. In an agent society, however, there are further problems: what sanctions are appropriate to agents, and how can they be applied? Possibly the best that can be done is through concern for reputation (compare the feedback mechanism used by auction website *ebay*), but the opportunities to cloak identities in cyberspace make this at best a flimsy defence.

For the third strategy to be possible we need to have agents that have the ability to reason about what to do in a particular way, so that they consider the general interest as well as their own. In so far as PD-norms are desirable in situations where neither making violation impossible nor enforcing sanctions is practicable, this seems to be the only solution. Respect for social laws and consideration for others are necessary parts of the functioning of human society: without a certain degree of compliance with social laws through simple consideration of others, life would be intolerable. It might be thought that consideration for others should be a desirable feature of agent societies also: we would not employ a person we believed to be amoral or dishonest, so why should we be prepared to unleash agents with no sense of moral duty on an unsuspecting world? We bring up our children to respect the interests of others, so should we not implement our agents in the same way?

6 Concluding Remarks

In this paper we have considered social laws in the context of rational decision making. We have seen that at one extreme some social laws simply provide the necessary degree of certainty about how others will behave to enable good results to come from rational, self-interested action. At the other extreme, other social laws will require the backing of certain and heavy sanctions to make compliance rational. In between there are situations where rationality leads to compliance if the welfare of the other agents involved in the situation is taken into account. When framing social laws we need to consider whether they will be adhered to: doubt as to the compliance of other agents will restore the uncertainty the social law was designed to resolve, and the social law will be rendered ineffective. When framing social laws these factors need to be considered: sometimes that will lead us to prefer one formulation over another, particularly if the social law favours one agent over another. There are also implications for designing reasoning agents: social laws will often depend on some sense of social obligation for their effectiveness, and so agents need to be designed to be capable of reasoning so as to consider the interests of others.

References

1. Arcos, J.L., Esteva, M., Noriega, P., Rodríguez, J.A., Sierra, C.: Engineering open environments with electronic institutions. Journal on Engineering Applications of Artificial Intelligence 18(2), 191–204 (2005)
2. Atkinson, K.: What Should We Do?: Computational Representation of Persuasive Argument in Practical Reasoning. PhD thesis, Department of Computer Science, University of Liverpool, Liverpool, UK (2005)
3. Atkinson, K., Bench-Capon, T.: Addressing moral problems through practical reasoning. In: Goble, L., Meyer, J.-J.C. (eds.) DEON 2006. LNCS, vol. 4048, pp. 8–23. Springer, Heidelberg (2006)
4. Atkinson, K., Bench-Capon, T.: Action-based alternating transitions systems for arguments about action. In: Proceedings of the Twenty Second Conference on Articicial Intelligence (AAAI 2007), pp. 24–29 (2007)
5. Bench-Capon, T.: Persuasion in practical argument using value based argumentation frameworks. Journal of Logic and Computation 13(3), 429–448 (2003)
6. Broersen, J., Dastani, M., Hulstijn, J., Huang, Z., van der Torre, L.: The BOID architecture: conflicts between beliefs, obligations, intentions and desires. In: Proceedings of Autonomous Agents 2001, pp. 9–16. ACM Press, New York (2001)
7. Castelfranchi, C.: Modelling social action for AI agents. Artificial Intelligence 103(1-2), 157–182 (1998)
8. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77, 321–357 (1995)
9. Moses, Y., Tennenholtz, M.: Artificial social systems. Computers and Artificial Intelligence 14(6), 533–562 (1995)
10. Shoham, Y., Tennenholtz, M.: On the synthesis of useful social laws for artificial agent societies. In: Proceedings of the Tenth Conference on Artificial Intelligence (AAAI 1992), pp. 276–281 (1992)
11. Ullmann-Margalit, E.: The Emergence of Norms. Clarendon Press, Oxford (1977)
12. van der Hoek, W., Roberts, M., Wooldridge, M.: Social laws in alternating time: effectiveness, feasibility and synthesis. Synthese 156(1), 1–19 (2007)
13. Walton, D.N.: Argumentation Schemes for Presumptive Reasoning. Lawrence Erlbaum Associates, Mahwah (1996)
14. Wooldridge, M., van der Hoek, W.: On obligations and normative ability: Towards a logical analysis of the social contract. Journal of Applied Logic 3, 396–420 (2005)

Annotation and Matching of First-Class Agent Interaction Protocols

Tim Miller[1] and Peter McBurney[2]

[1] Department of Computer Science and Software Engineering
University of Melbourne
Victoria, Australia, 3010
tmill@csse.unimelb.edu.au
[2] Department of Computer Science,
University of Liverpool
Liverpool, L69 7ZF, UK
mcburney@liverpool.ac.uk

Abstract. Many practitioners view agent interaction protocols as rigid specifications that are defined *a priori*, and hard-code their agents with a set of protocols known at design time — an unnecessary restriction for intelligent and adaptive agents. To achieve the full potential of multi-agent systems, we believe that it is important that multi-agent interaction protocols are treated as *first-class* computational entities in systems. That is, they exist at runtime in systems as entities that can be referenced, inspected, composed, invoked and shared, rather than as abstractions that emerge from the behaviour of the participants. Using first-class protocols, a goal-directed agent can assess a library of protocols at runtime to determine which protocols best achieve a particular goal. In this paper, we presented three methods that enable agents to determine if a protocol achieves a specified goal. The two most promising approaches allow an agent to *summarise* a protocol that it has learned by calculating the outcomes achieved by the protocol, and *annotate* the protocol with these summaries. The agent can *match*, via annotations, which protocols in a library achieve a given goal.

Keywords: multi-agent systems, agent interaction, first-class protocols, annotation, matching.

1 Introduction

In the distributed environments of multi-agent systems, interaction protocols are seen as a promising approach to coordination in multi-agent systems. However, rather than viewing interaction protocols as rigid specifications that are defined *a priori*, with agents being hard-coded to follow the protocol rules — a restriction that is out of place with the vision of agents being intelligent and adaptive — we believe it is important that agent interaction protocols are first-class computational entities, allowing agents to select, reference, share, compose, invoke and inspect protocols at runtime. Such an approach would allow agents

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 141–158, 2009.

to assess which protocols achieve their goals, and to learn the rules and effect of new protocols at runtime.

In previous work, we proposed the $\mathcal{RASA}$ framework [9,10], which regards protocols as first-class entities. These first-class protocols are documents that exist within a multi-agent system, in contrast to hard-coded protocols, which exist merely as abstractions that emerge from the messages sent by the participants. To promote decoupling of agents from the protocols they use, $\mathcal{RASA}$ contains a formal, executable language for protocol specification, about which agents can reason to determine the rules and outcomes of protocols.

A major goal of research into first-class protocols is for agents to maintain a library of interaction protocols, and to be able to select the protocol that best suits the goals that it wants to achieve at a given time and in a given environment. For this, agents must be able to quickly and correctly determine the outcomes that can result for an interaction protocol, and compare protocols in their library. In this paper, we present methods for *annotating* a protocol with its possible outcomes, so that it does not have to determine the outcomes each time it is trying to find a suitable protocol, and for *matching* a protocol that achieves a given goal, using the annotations. Emphasis is placed on protocols specified in the $\mathcal{RASA}$ protocol language, but such ideas would be applicable to other protocol languages with operators similar to $\mathcal{RASA}$'s.

2 The $\mathcal{RASA}$ Framework

The $\mathcal{RASA}$ framework was designed to allow us to represent and reason about first-class protocols, and investigate the types of statements we can make about them. The $\mathcal{RASA}$ specification language was designed as an example of the minimal requirements for a successful first-class protocol specification language. First presented in [10], along with its operational semantics, the language uses constraint languages and process algebra to specify interaction protocols. In this section, we briefly present this language, and a logic for reasoning about the outcomes of protocols specified in this language.

2.1 Modelling Information

Rather than devise a new language for expressing information, or using an existing language, we take the approach that any constraint language can be used to model the universe of discourse, provided that it has a few basic constants, operators and properties. This allows us to express and study a wider variety of protocols, such as those that use description logic, constraint programming languages, or even predicate and modal logic's. It also permits us to use different mechanisms for defining protocol meaning, such as norms and commitments.

Definition 1. *Cylindric constraint system.* We assume that the underlying communication language fits the definition of a *cylindric constraint system* proposed by De Boer et al. [3]. They define a cylindric constraint system as a complete algebraic lattice, $\langle C, \sqsupseteq, \sqcup, \text{true}, \text{false}, Var, \exists \rangle$. In this structure, C is the set of

atomic propositions in the language, for example $X = 1$, $\sqsupseteq$ is an entailment operator, true and false are the least and greatest elements of C respectively, $\sqcup$ is the least upper bound operator, Var is a countable set of variables, and $\exists$ is an operator for hiding variables. The entailment operator defines a partial order over the elements in the lattice, such that $c \sqsupseteq d$ means that the information in d can be derived from c.

A constraint is one of the following: an atomic proposition, c, for example, $X = 1$, where X is a variable; a conjunction, $\phi \sqcup \psi$, where ϕ and ψ are constraints; or $\exists_x \phi$, where ϕ is a constraint and $x \in Var$. We extend this notation by allowing negation on the right of an entailment operator, for example, $\phi \sqsupseteq \neg\psi$ is true if and only if $\phi \sqsupseteq \psi$ is not. Other propositional operators are then defined from these, for example, $\phi \vee \psi \mathrel{\widehat{=}} \neg(\neg\phi \sqcup \neg\psi)$ and $\phi \rightarrow \psi \mathrel{\widehat{=}} \neg\phi \vee \psi$. We use $vars(\phi)$ to refer to the free variables that occur in ϕ; that is, the variables referenced in ϕ that are not hidden using $\exists$.

2.2 Modelling Protocols

The $\mathcal{RASA}$ protocol specification language is an action language. The actions – messages sent across channels – manipulate a *shared social state*, for example, a set of commitments between the participants. An agent can send a message across a channel, and the definition of that message informs other participants to update their copy of the state. Messages are specified using *atomic protocols* of the form

$$\psi \xrightarrow{c(i,j).\phi} \psi',$$

in which ψ represents the precondition that must hold in the current social state for the message to be sent, $c(i, j)$ represents the channel from participant i to participant j, ϕ is the message template, and ψ' is the postcondition, which specifies the effect this message has on the state. We omit (i, j) when we do not care who the sender and receiver of the message is.

A special type of atomic protocol is the *empty protocol*: $\psi \rightarrow \epsilon$, which specifies that no message is required to be sent if ψ holds in the current social state.

Compound protocols can be built up from atomic protocols and empty protocols using operators. If π_1 and π_2 are protocols, then $\pi_1; \pi_2$ is their sequential composition, $\pi_1 \cup \pi_2$ is a choice between them. The protocol $\mathbf{var}_x^{\psi} \cdot \pi_1$ represents a protocol the same as π_1, except that a local variable x is available over the scope of π_1, but with the constraints ψ on x remaining unchanged throughout that scope. Any variable x already in the state is out of scope until π_1 finishes executing.

A *protocol specification* is a collection of protocol definitions of the format $N(x, \ldots, y) \mathrel{\widehat{=}} \pi$, in which N is a name, $x, \ldots, y \in Var$, and π represents a protocol. Protocols can be referenced from other protocols via their name.

An iteration operator, π^*, is derivable from these operators by declaring a recursive protocol using names. π^* is defined as the name N, in which

$$N \mathrel{\widehat{=}} \epsilon \cup \pi; N.$$

A positive iteration operator, π^+, representing one or more iterations of π, is defined as $\pi; \pi^*$.

2.3 Reasoning about Protocol Outcomes

$\mathcal{RASA}$ defines a logic for reasoning about protocols. By logic, we mean a syntax, semantics, and proof system. The logic is concerned with protocol outcomes; that is, the state of the protocol after it is executed. For this reason, we use a restricted version of propositional dynamic logic [7], tailored to the $\mathcal{RASA}$ specification language, and derived a proof system that corresponds to the system for dynamic logic.

The syntax for a proposition in this logic is defined using the following grammar, assuming that ϕ_0 is a constraint in the underlying constraint language:

$$\phi \ ::= \ \phi_0 \quad | \quad \phi \wedge \phi \quad | \quad \neg\phi \quad | \quad [\pi]\phi$$

A formal semantics and proof system for this logic has been defined in [9]. ϕ_0 is true if it holds in the underlying constraint system. $\phi \wedge \psi$ and $\neg\phi$ are defined as conjunction and negation respectively. The interesting operator, $[\pi]\phi$, which is found in propositional dynamic logic, has the meaning that ϕ holds for every possible outcome of the protocol π. That is, no matter which possible dialogue is executed in the protocol π, the proposition ϕ will hold after the protocol has executed.

As usual in dynamic logic, we define the dual operator $\langle\pi\rangle\phi$, which means that ϕ holds in at least one possible outcome of the protocol π, and is shorthand for $\neg[\pi]\neg\phi$. That is, ϕ holds in at least one end state of π if and only if $\neg\phi$ does not hold in all of them. We use subscripts on the Greek letters ϕ and ψ to indicate something that is strictly a constraint; that is, ϕ_0 is a constraint, while ϕ can be a dynamic logic proposition (including a constraint).

We use this logic to represent protocol outcomes, match protocols, as well as to prove the correctness of our method for annotation and matching.

3 Definitions

Before we continue with our presentation, we first define some terms.

For the rest of this paper, the term *protocol* will refer to a $\mathcal{RASA}$ definition of a protocol that does not have the *stuckness* property, as defined in earlier work [11]. A protocol has the stuckness property if and only if, at any point during the execution of the protocol, the protocol has not terminated, and the rules of the protocol prevent any participant from making an utterance. Therefore, in this work, we are assuming that the protocol has been proved to be *stuckness-free*, using a proof method such as that presented in [11].

The *weakest precondition* of a protocol is the weakest (or most general) constraint from which a protocol cannot become stuck. The *maximal postcondition* is the strongest constraint that results from a protocol being executed under its weakest precondition.

An *annotation* of a protocol is some information that is attached to a protocol, but is not part of the protocol's definition. An annotation could contain information such as where the protocol originated, security properties, or efficiency properties. In this paper, we are concerned solely with *outcome annotations*: annotations that document the outcomes that may result if a protocol is executed.

A *goal* is a state of the world that an agent would like to bring about, or maintain. In this paper, we assume that a goal is represented as a constraint in the underlying constraint language.

Given a goal, ϕ_G, an initial state, ψ_I (the state of the world from which an agent wants to achieve the goal – generally the current state), a *weak matching* protocol is a protocol, π, in an agent's protocol library, such that

$$\psi_I \rightarrow \langle\pi\rangle\phi_G.$$

That is, from the initial state, at least one outcome of the protocol entails the goal.

A *strong matching* protocol is a protocol that achieves a goal for all outcomes, assuming that there exists at least one outcome[1]. Formally:

$$\psi_I \rightarrow [\pi]\phi_G.$$

All strong matching protocols are also weak matching protocols.

To find all matches for a goal ϕ_G from the state ψ_I, the agent could simply use the proof system discussed in Section 2.3. That is, for every protocol π, if the proof $\psi_I \rightarrow \langle\pi\rangle\phi_G$ is successful, then π is a weak match.

However, for a large protocol library, this is an expensive operation to perform each time an agent wants to find a protocol that achieves a certain goal. In Section 4.3, we present a matching method that uses outcome annotations to prove the above, but is more efficient once the outcomes annotations have been derived.

4 Matching Protocols via Proof

Outcome annotations for a protocol are derivable directly from the protocol definition itself. In this section, we discuss an algorithm for deriving annotations of protocols, including iterative protocols, and terminating recursive protocols; that is, recursively defined protocols that always terminate. We prove that this method is sound and complete.

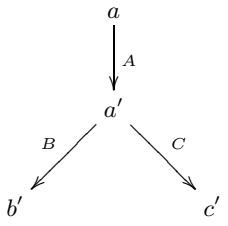
4.1 Representing Outcome Annotations

Outcome annotations are represented as theorems in the logic presented in Section 2.3. For example, the annotation

$$\psi_0 \rightarrow [\pi]\phi_0$$

specifies that, if executed from any state that satisfies the weakest precondition ψ_0, the protocol π is guaranteed to achieve the outcome ϕ_0.

[1] This assumption is subsumed by our assumption that a protocol is free from stuckness.



$a \rightarrow [A]a'$
$a \rightarrow [A;B](a' \sqcup b')$
$a \rightarrow [A;C](a' \sqcup c')$

$a' \rightarrow [B](a' \sqcup b')$

$a' \rightarrow [C](a' \sqcup c')$

Fig. 1. An abstract syntax tree for a protocol, and its annotations

We enforce on strict condition: outcomes must be specified using the underlying constraint language, not the dynamic logic. That is, for an annotation $[\pi]\phi_0$, ϕ_0 must not contain any expressions of the form [ ] or $\langle\rangle$. We assume that the goals of agents are specified using (or at least translated to) the constraint language.

Our goal is to annotate, for each protocol in a protocol library, not the outcomes that it achieves, but the outcomes of the *paths* that make up this protocol.

As an example, consider the protocol $A;(B \cup C)$, in which $A \mathrel{\widehat{=}} a \xrightarrow{c.a_m} a'$, $B \mathrel{\widehat{=}} \text{true} \xrightarrow{c.b_m} b'$, and $C \mathrel{\widehat{=}} \text{true} \xrightarrow{c.c_m} c'$. Figure 1 shows the abstract syntax tree of the protocol, and the annotations that we want to be able to derive. This protocol contains two paths: (1) A followed by B; and (2) A followed by C.

The first annotation would be annotated to protocol A, the second and third to the entire protocol, and the final two to B and C respectively. In this paper, we are concerned solely with how these outcome annotations are derived, and do not discuss the attachment of annotations to protocols.

4.2 Deriving Outcome Annotations

Instead of proving annotations for the specific goals of an agent, it may be more efficient if the agent derives annotations directly from the protocol definition. It would not be possible to annotate the protocol with every constraint that could hold in an end state, so instead, we document the outcomes as *maximal postconditions*. By maximal, we mean a constraint such that every end state satisfies that constraint, and that it is satisfied only by those end states. Therefore, every end state would entail the maximal postcondition. We also document the maximal postconditions of the sub-protocols that make up a compound protocol, to help agents choose between multiple protocols that each achieve a goal, and to help them reason about which interactions best achieve their goals.

The advantage of this approach is that each annotation needs to be calculated only once, and it remains valid for the lifetime of the system/protocol. In fact, annotations can be added to protocols and shared between agents, therefore not requiring each agent to derive them.

To derive annotations, we specify a set of *annotation rules*. When receiving a new protocol, an agent applies these rules to the protocol, and its sub-protocols, adding the annotations. Annotation rules are specified as theorems in the $\mathcal{RASA}$

logic. Each rule is of the form $\phi \wedge \phi' \wedge \ldots \rightarrow \psi$, resembling a Horn clause, and should be read: *if ϕ,ϕ',... can be derived as maximal postcondition annotations, then ψ is a maximal postcondition annotation.* Each of the annotation rules in this section has been proved using the axiom system defined in [9].

Global Annotations. The most straightforward of the rules is that if a protocol π achieves ϕ_0 for all outcomes, and has at least one outcome, then it must achieve ϕ_0 for at least one outcome.

$$[\pi]\phi_0 \wedge \langle\pi\rangle\text{true} \;\rightarrow\; \langle\pi\rangle\phi_0$$

This is similar to the D axiom found in many modal logics. We define a protocol to be *executable* when it contains at least one interaction that can occur fully, while following the rules of the protocol. For a protocol, π, that is not executable, (for example, a protocol whose precondition is false), the proposition $[\pi]\phi$ will hold for any ϕ; that is, for all end states, of which there are none, ϕ holds. Therefore, if false is provable at all end states, either there are no end states, or all end states are equivalent to false, which is represented as $\neg[\pi]\neg$false. This is equivalent to $\langle\pi\rangle$true from the definition of $\langle\ \rangle$.

The above rule is unnecessary for deriving annotations, because when search for a weak match, an agent would simply need to check if $[\pi]\phi_0$ was an annotation. However, it is necessary for this specification because weak matches are often premises of other annotation rules.

Annotating $\psi_0 \rightarrow \epsilon$. An empty protocol receives only one annotation:

$$[\psi_0 \rightarrow \epsilon]\psi_0$$

This rule contains no premise. An empty protocol does not change the state, therefore, the postcondition is anything that is true before the execution of the protocol. In the case of the maximal postcondition, the only information that we can derive is that the precondition must hold for the protocol to execute, therefore, the maximal postcondition is the precondition ψ_0.

Annotating $\psi_0 \xrightarrow{c.\phi_m} \psi'_0$. An atomic protocol receives only one annotation:

$$\langle\psi_0 \xrightarrow{c.\phi_m} \psi'_0\rangle\text{true} \;\rightarrow\; [\psi_0 \xrightarrow{c.\phi_m} \psi'_0](\psi'_0 \sqcup \phi_m \sqcup \exists_{vars(\psi'_0 \sqcup \phi_m)}\psi_0)$$

The premise of this rule insists that the protocol is executable. The maximal postcondition of an atomic protocol $\psi_0 \xrightarrow{c.\phi_m} \psi'_0$ is the constraint that corresponds to the $\psi'_0 \sqcup \phi_m$ (the constraint that specifies the postcondition), conjoined with the constraints on variables from the precondition ψ_0 that have not been changed by the protocol; that is, those that are not free in ψ'_0 or ϕ_m.

Annotating $\pi_1; \pi_2$. Two annotation rules are associated with sequentially composed protocols:

$$[\pi_1][\pi_2]\phi_0 \;\rightarrow\; [\pi_1;\pi_2]\phi_0$$

$$\langle\pi_1\rangle\langle\pi_2\rangle\phi_0 \;\rightarrow\; \langle\pi_1;\pi_2\rangle\phi_0$$

The first says that if, for every end state of π_1, the maximal postcondition of π_2 is ϕ_0, then the maximal postcondition of $\pi_1;\pi_2$ is also ϕ_0. The second is similar, but for the $\langle\rangle$ operator. Note that, as a result of the global annotation rule specified in Section 4.2, an agent will also derive the annotation $\langle\pi_1;\pi_2\rangle\phi_0$ from $[\pi_1]\langle\pi_2\rangle\phi_0$ or $\langle\pi_1\rangle[\pi_2]\phi_0$.

Recall from introduction to Section 4, goals must be constraints. Therefore, annotations of the form $[\pi_1][\pi_2]\phi_0$ cannot occur. To derive the equivalent for $[\pi_1][\pi_2]\phi_0$ (and similarly for $\langle\rangle$), one must derive the maximal postcondition of π_2 under the initial state that is the maximal postcondition of π_1. That is, for the annotation $[\pi_1]\psi_0$, derive the maximal postcondition for $\psi_0 \rightarrow [\pi_2]\phi_0$. This can be expressed as the following rule.

$$[\pi_1]\psi_0 \wedge (\psi_0 \rightarrow [\pi_2]\phi_0) \quad \rightarrow \quad [\pi_1;\pi_2]\phi_0$$

$$\langle\pi_1\rangle\psi_0 \wedge (\psi_0 \rightarrow \langle\pi_2\rangle\phi_0) \quad \rightarrow \quad \langle\pi_1;\pi_2\rangle\phi_0$$

These rules are read differently to others, because it is unlikely that there will be annotations $\psi_0 \rightarrow [\pi_2]\phi_0$ or $\psi_0 \rightarrow \langle\pi_2\rangle\phi_0$. In these cases, the rules are read that if $[\pi_1]\psi_0$ (respectively $\langle\pi_1\rangle\psi_0$) is an annotation, and then calculate the maximal postcondition, ϕ_0, of π_2 under the state ψ_0. ϕ_0 is then the maximal postcondition of $\pi_1;\pi_2$.

Annotating $\pi_1 \cup \pi_2$. Choice protocols are the most difficult to annotate because they offer more than one alternative, each with a maximal postcondition, and our method must derive information that covers each of these. We propose the following three rules, each which is straightforward to prove.

$$\langle\pi_1\rangle\phi_0 \quad \rightarrow \quad \langle\pi_1 \cup \pi_2\rangle\phi_0$$

$$\langle\pi_2\rangle\phi_0 \quad \rightarrow \quad \langle\pi_1 \cup \pi_2\rangle\phi_0$$

$$[\pi_1]\phi_0 \wedge [\pi_2]\psi_0 \quad \rightarrow \quad [\pi_1 \cup \pi_2](\phi_0 \vee \psi_0)$$

However, the final rule above is not adequate as an annotation. Recall from the start of this section, that we restrict the annotations to propositions of the form $[\pi]\phi_0$, in which ϕ_0 is a constraint. Constraint stores cannot hold negations or disjunctions, and as a result, the application of this rule is non-trivial, and in many cases, deriving the maximal postcondition is not possible.

Despite this, we can still derive some information that is useful. For example, take the following choice protocol definition:

$$N \mathrel{\widehat{=}} x = 1 \xrightarrow{c.a(x)} x = y \sqcup x \in [0..7] \quad \cup$$
$$x = 1 \xrightarrow{c.b(x)} x \neq y \sqcup x \in [3..10]$$

Despite the fact that these two postconditions are inconsistent with each other (because one contains $x = y$ and the other $x \neq y$), we can still derive common information. The maximal postcondition is $x \in [3..7]$, so we could derive the annotation $[N](x \in [3..7])$ — that is, we know that x will be in the range $[3..7]$ whichever interaction is executed. Such an annotation could prove useful for an agent.

However, except for the most trivial cases (e.g. where $\phi_0 \sqsupseteq \psi_0$ or $\psi_0 \sqsupseteq \phi_0$), calculating this is beyond the means of any constraint solver known to the

authors, because constraint solvers are not designed to find the most general constraint store that is consistent with two unrelated, and possible inconsistent, constraints.

To find a solution such as the one above, we propose an approach in which an agent analyses different parts of the constraints. This does not necessarily find the best solution, but it can derive a constraint that satisfies parts of both ϕ_0 and ψ_0.

To do this, we consider the variables in the two constraints. Clearly, any maximal postcondition of a choice must only reference variables that are in both ϕ_0 and ψ_0 — any variables in only one will not be constrained in the maximal postcondition of the choice protocol. Therefore, what we aim to do is derive a set of constraints, each of which is relevant to only a subset of the variables. For example, if we take the two postconditions from above and hide y from both, then the constraint $\exists_y(x = y \sqcup x \in [0..7]) \sqcup \exists_y(x \neq y \sqcup x \in [3..10])$ is reduced to the constraint $x \in [3..7]$, which is the maximal postcondition relative to x, so we can use this as an annotation. If we hide x instead, then the resulting constraint is unsatisfiable, so this is not considered as an annotation.

In this example, we examine the variables in ϕ_0 and ψ_0, and look at constraints that result from hiding some of these variables. It is not always useful to hide only one variable, otherwise we lose information about the constraints between variables. Instead, we hide different combinations of variables. To obtain the relationships between all variables, one can take an approach that, for every set of variables $Z \subset vars(\phi_0) \cap vars(\psi_0)$, check if the constraint $\exists_Z \phi_0 \sqcup \exists_Z \psi_0$ is satisfiable — something which we hope is straightforward for any constraint solver to check. If this is satisfiable, then we add the annotation $[\pi_1 \cup \pi_2](\exists_Z \phi_0 \sqcup \exists_Z \psi_0)$.

We note that this approach is sound but not complete. That is, such an approach will always produce annotations that are correct, but they are not guaranteed to be annotations containing the maximal postcondition. For example, take the following definition:

$$P \mathrel{\widehat{=}} x = 1 \xrightarrow{c.a(x)} x = 1 \quad \cup \quad x = 1 \xrightarrow{c.b(x)} x = 2$$

Here, either $x = 1$ or $x = 2$ will hold in the outcome. Therefore, the proposition $[P]x \in [1..2]$ is valid, but our annotation rules fail to derive this, because $x = 1 \sqcup x = 2$ is not satisfiable, and as a result, we derive only the annotations $\langle P \rangle x = 1$ and $\langle P \rangle x = 2$ (using the first two rules), even though the annotation $[P]x \in [1..2]$ is the annotation containing the maximal postcondition.

While this approach is sound and may prove useful, the approach for calculating it is somewhat undesirable, because for n variables, we have $2^n - 1$ different combinations to check, and a worst case of $2^n - 1$ annotations. For a large n, deriving annotations is costly, as is searching through annotations to match protocols. For large n, agents could selectively choose to calculate annotations based on the variables in their current goals. That is, if they generally have goals that related to certain variables in the system, then calculate only the annotations for those variables.

However, all is not lost for outcome derivations. Theorem 1 (Section 4.3) shows that the final rule does not need to be applied to find a suitable protocol, provided that the first two rules for choice protocols are applied. While not necessary, applying the above rule may reduce the runtime complexity of finding a suitable protocol. This is discussed further in Section 4.3.

Annotating $\mathbf{var}_{x}^{\psi_0} \cdot \pi$. Annotation of variable declarations is straightforward. If the maximal postcondition of the sub-protocol π is ϕ_0, then the maximal postcondition of $\mathbf{var}_{x}^{\psi_0} \cdot \pi$ is ϕ_0 with the value of x constrained to the same value as it is before execution. For this we introduce a new variable x_0 and constrain it to be equal to x. In the postcondition, we then constrain that x must be equal to x_0. We know that the constraints on x_0 have not been changed by π because it is a fresh variable and therefore not referenced in in π. Finally, the variable x_0 is hidden using the $\exists$ operator, because x_0 is not part of the maximal postcondition.

$$\psi_0 \rightarrow [\pi]\phi_0 \quad \rightarrow \quad x = x_0 \rightarrow [\mathbf{var}_{x}^{\psi_0} \cdot \pi]\exists_{x_0}(x = x_0 \sqcup \phi_0)$$

In which x_0 is fresh

Annotating π^*. To annotate an iterative protocol, we derive information from the protocol that is iterated over. The first rules specifies that if a protocol π achieves ϕ_0 for all outcomes, then iterating it one or more times achieves ϕ_0 as well. The second rules specifies that if a protocol π achieves ϕ_0 for at least one outcome, then iterating zero or more times must also achieve ϕ_0.

$$[\pi]\phi_0 \rightarrow [\pi;\pi^*]\phi_0$$

$$\langle\pi\rangle\phi_0 \rightarrow \langle\pi^*\rangle\phi_0$$

In addition to the above, one can also annotate the protocol to specify the case of zero iterations of the protocol. In such a case, the state of the protocol remains unchanged.

The first rule above is perhaps perhaps the only annotation rule that does not follow from its definition. However, the soundness of this rule is straightforward to prove.

From the premise of the rule, we know that one iteration of π will result in ϕ_0. After that iteration, one of two properties hold: either ϕ_0 entails the precondition of π, or is does not. If the former, then π cannot iterate again, because the precondition is not satisfied. Therefore, the postcondition must be ϕ_0. If the latter, then π can either terminate, leaving the postcondition as ϕ_0, or it can iterate again. If it iterates again, we know that the result will be ϕ_0, because ϕ_0 holds for every outcome of π. Therefore, the resulting postcondition is ϕ_0. Applying this argument inductively, we see that the strongest postcondition of $\pi;\pi^*$ is ϕ_0.

Annotating $N(x)$. To derive outcome annotations for name references, an agent can simply derive the annotations for the protocol that the name references. The downside to this is that for a recursively-defined protocol, an infinite number of unfoldings will result. In other work, we are looking at ways to annotate recursively-defined protocols.

Annotating $\pi_1; (\pi_2 \cup \pi_3)$. Protocols of the form $\pi_1; (\pi_2 \cup \pi_3)$ (and $(\pi_1 \cup \pi_2); \pi_3$) are treated as special cases because they represent the merging or splitting of interactions. Theories of business process modelling often refer to these as *or-splits* and *or-joins* respectively, because they represent the splitting and joining of single traces with multiple traces respectively.

We can derive important annotations from protocols of this format, mainly those that annotate the outcomes achieved by the different interactions. This is important because it gives agents additional information for choosing protocols that achieve their goals, as well as choosing which interactions best achieve their goal. The rules specified so far in this section fail to take into account these special cases. We specify two annotation rules for protocols of this form.

$$\langle \pi_1; \pi_2 \rangle \phi_0 \;\rightarrow\; \langle \pi_1; (\pi_2 \cup \pi_3) \rangle \phi_0$$

$$\langle \pi_1; \pi_3 \rangle \phi_0 \;\rightarrow\; \langle \pi_1; (\pi_2 \cup \pi_3) \rangle \phi_0$$

These say that if the protocols $\pi_1; \pi_2$ or $\pi_1; \pi_3$ have at least one end state in which ϕ_0 is the maximal postcondition, then the composite protocol $\pi_1; (\pi_2 \cup \pi_3)$ also has at least one end state such that ϕ_0 holds. This is clear from the semantics of $\langle\rangle$, and is provable by showing that $\pi_1; (\pi_2 \cup \pi_3)$ is equivalent to $\pi_1; \pi_2 \cup \pi_1; \pi_3$, and using the annotation rules from Section 4.2.

Note that the annotations produced on the right hand side of the rules will be derived from the rules for choice and sequential composition, but the annotations in the premise will not. Therefore, these two rules exist solely to document that one must annotate $\pi_1; \pi_2$ and $\pi_1; \pi_3$.

We also note that protocols of the form $(\pi_1 \cup \pi_2); \pi_3$ have similar annotation rules, however, these should be clear to the reader, so they are omitted.

4.3 Using Derived Annotations

Agents use annotations to search for protocols that achieve their goals, and to guide them through the execution of a protocol. In this section, we present a method for determining whether an annotated protocol achieves a given goal. The process of selecting a protocol, should there be more than one such match, and the process of reasoning about interaction are more likely to be varied between different agent implementations, so they are not discussed here. However, the process of matching protocols is more straightforward and likely to follow a similar pattern between implementations.

If an agent has a goal, ϕ_G, then it must find a protocol that achieves ϕ_G. To do this, it could either search through annotations of protocols until it finds a protocol that satisfies its needs, or search through all protocols and then make a choice if multiple protocols achieve its needs. In this section, we focus only on the process of assessing whether a protocol achieves the goal — that is, matching protocols — assuming that the annotations have been derived using the rules from Section 4.2. This method is guaranteed to find a protocol, if one exists.

A First Attempt. For a goal ϕ_G and protocol π, take the annotations of π and then perform the following:

1. For every annotation of the format $[\pi]\phi_A$, test $\phi_A \sqsupseteq \phi_G$; that is, test whether the maximal postcondition satisfies the goal. If this entailment is successful, and the initial state under which the protocol will be executed satisfies its precondition, add π to the list of strong matching protocols.
2. If step 1 fails, for every annotation of the format $\langle\pi\rangle\phi_A$, test $\phi_A \sqsupseteq \phi_G$. If this entailment is successful, and the initial state under which the protocol will be executed satisfies its precondition, add π to the list of weak matching protocols.

This is a reasonable way to match protocols, however, it does not guarantee that an agent will find a protocol that satisfies its goal, even if one exists. For example, take a situation in which an agents goal is $x = 1$. We have an annotation $[\pi](x \in [1..5] \sqcup x = y)$, but the above process fails to match this because $x \in [1..5] \sqcup x = y$ does not entail $x = 1$. However, it may be the case that one end state satisfies $x = 1$, but because the annotations document only *maximal* postconditions, there is no annotation such that $x = 1$.

Clearly, adding annotations for every possible postcondition is at best, expensive, and at worst, impossible. Instead, we add an extra step to the process which is not expensive, and which guarantees that we find a matching protocol.

A Complete Approach. A complete approach requires us to assess the maximal postconditions in more detail. To do this, we perform an additional test for every annotation, while still performing the naive approach. For a goal ϕ_G and protocol π, take the annotations of $[\pi]\phi_A$ and $\langle\pi\rangle\phi_A$ and then perform the following:

1. Test whether ϕ_G and ϕ_A are consistent with each other; that is, $\phi_G \sqcup \phi_A \not\sqsupseteq$ false. If the goal and postcondition are consistent, then it may be that there is an outcome stronger than the maximal postcondition that satisfies our goal, as in the example above. If not, then π can never achieve the goal ϕ_G, so do not continue.
2. If step 1 succeeds, attempt to prove $\psi_I \rightarrow [\pi]\phi_G$, in which ψ_I is the initial state in which the protocol will be executed. If this is provable, add π to the list of strong matching protocols.
3. If step 1 succeeds and step 2 fails, attempt to prove $\psi_I \rightarrow \langle\pi\rangle\phi_G$, in which ψ_I is the initial state in which the protocol will be executed. If this is provable, add π to the list of weak matching protocols.

Using our example above, if an agent has a protocol with the maximal postcondition $x \in [1..5]$, and a goal $x = 1$, then it can calculate in a straightforward manner that $x = 1$ is consistent with $x \in [1..5]$. From here, it tries to prove if it is possible that $x = 1$ in all outcomes of the protocol: $[\pi]x = 1$. If this holds, then its possible for the agent to achieve its goals with this protocol. If not, the it tries to prove $\langle\pi\rangle x = 1$. The steps of proving $[\pi]x = 1$ (and $\langle\pi\rangle x = 1$) are in fact necessary, because it may be possible that the goal and maximal postconditions are consistent, but that the goal is not achieved by the protocol. For example, consider a case in which $x \in [1..5]$ is also the only constraint that holds for x in all outcomes of a protocol:

$$N \mathrel{\widehat{=}} x = 0 \xrightarrow{c.a(y)} y = 1 \sqcup x \in [1..5]$$

Here, because the sender cannot constrain the value of x (it not being part of the message), the only postcondition is $y = 1 \sqcup x \in [1..5]$, so $x = 1$ is not achieved. However, consider the following alternate protocol, in which $y = 1$ is replaced with $y = x$ in the postcondition:

$$N \mathrel{\widehat{=}} x = 0 \xrightarrow{c.a(y)} y = x \sqcup x \in [1..5]$$

An agent can constrain the value of x , and therefore $x = 1$ is a possible end state, and $\langle N \rangle x = 1$ could be proved.

We also note that attempting the proof $[\pi]\phi_G$ is beneficial, rather than only proving $\langle \pi \rangle \phi_G$. That is, it is possible that $\phi_A \not\sqsupseteq \phi_G$, but that all outcomes hold for ϕ_G, which may initially seem counter-intuitive. This is best demonstrated with the following example:

$$P \mathrel{\widehat{=}} x = 1 \xrightarrow{c.a(x)} x = 1 \quad \cup \quad x = 1 \xrightarrow{c.b(x)} x = 2$$

Here, either $x = 1$ or $x = 2$ will hold in the outcome. Therefore, the proposition $[P]x \in [1..2]$ is valid, but, as discussed in Section 4.2, our annotation rules fail to derive this. As a result, the goal $x \in [1..2]$ would not be matched, but the proof $[P]x \in [1..2]$ would succeed.

Although the approach outlined in this section requires a proof to be performed at runtime, similar to the matching-via-proof method described in Section 4, checking whether or not the goal is consistent with the maximal postcondition before attempting the proof would eliminate the need to do these proofs for many protocols. This is advantageous because proofs in the dynamic logic are more computationally expensive than the matching rules, especially for large protocols.

Theorem 1. *This annotation and matching method is sound and complete.*

Proof. To prove this theorem, we need to demonstrate that any matched protocol achieves the goal, and that only protocols that achieve the goal are matched.

Soundness is straightforward to prove. For a goal ϕ_G, and annotations $[\pi]\phi_A$ and $\langle \pi \rangle \phi_A$, a match is noted when at least one of the following two conditions hold:

1. $\phi_A \sqsupseteq \phi_G$; or
2. $\phi_G \sqcup \phi_A \not\sqsupseteq$ false and $[\pi]\phi_G$ (respectively $\langle \pi \rangle \phi_G$).

Part (1) is trivially sound from modus ponens and the modal logic inference rule of necessitation; Part (2) is trivially sound from $[\pi]\phi_G$ (or $\langle \pi \rangle \phi_G$). Additionally, annotations are added using the rules from Section 4.2, which are valid theorems of our logic. Therefore, the method is sound.

Completeness is less straightforward to prove. For completeness, we must prove that, for any true formula $[\pi]\phi_G$ (or $\langle \pi \rangle \phi_G$), the rules defined in Section 4.2 will produce an annotation $[\pi]\phi_A$ (respectively $\langle \pi \rangle \phi_A$), such that either:

1. $\phi_A \sqsupseteq \phi_G$; or
2. $\phi_G \sqcup \phi_A \not\sqsupseteq$ false and $[\pi]\phi_G$ (respectively $\langle \pi \rangle \phi_G$).

This is proved via *reductio ad absurdum*. It is trivially valid that every valid protocol receives an annotation, and from the soundness proof, we know that each annotation is correct, therefore, we know there is a correct annotation $\langle\pi\rangle\phi_A$. We prove this only for the case of $\langle\pi\rangle\phi_A$, because from the global annotation rule, this will also be valid for $[\pi]\phi_A$, for executable π.

If our method is incomplete, then there exists ϕ_G such that $\langle\pi\rangle\phi_G$ holds, but that:

1. $\phi_A \not\sqsupseteq \phi_G$; and
2. $\phi_G \sqcup \phi_A \sqsupseteq$ false or $\neg\langle\pi\rangle\phi_G$.

That is, π is not matched as achieving ϕ_G, despite the fact that it is a valid postcondition. It suffices to prove that item 2 above is a contradiction. Firstly, $\neg\langle\pi\rangle\phi_G$ contradicts the assumption that ϕ_G is a valid postcondition.

Secondly, if ϕ_A is the maximal postcondition of at least one interaction in π, then it must be that any postcondition of that interaction is consistent with ϕ_A. It holds trivially from the annotation rules that every interaction in a protocol receives an annotation, therefore, it cannot be that each annotation on π is inconsistent with ϕ_G if ϕ_G is a valid postcondition.

If ϕ_A is not the maximal postcondition (recall from Section 4.2 that the final annotation rule for choice protocols does not necessarily derive the maximal postcondition), then it may be that the case that an annotation on ϕ_A is inconsistent with the goal ϕ_G. However, the third rule from Section 4.2 is not necessary for completeness, because the first two rules cover such a case: if $\langle\pi_1 \cup \pi_2\rangle\phi_G$ is true, then is must be that $\langle\pi_1\rangle\phi_G$ or $\langle\pi_2\rangle\phi_G$, one of which will be derived from the first two rules. Therefore, the other annotations rules defined in Section 4.2 will have annotated $\pi_1 \cup \pi_2$ with their maximal postconditions, one of which must be consistent with ϕ_G, and therefore the protocol will be matched.

□

From this, we see that the final annotation rule defined in Section 4.2 is redundant. However, it may be useful because in some cases, it can prevent an agent from having to discharge proofs for propositions $[\pi]\phi_G$ and $\langle\pi\rangle\phi_G$. Whether the rule is used would clearly be a policy of individual agents. Such a decision is specific to the strategy of the agents, not the protocol itself, and therefore it is out of scope of this paper.

5 Annotating and Matching Pre/Postcondition Models

When using our framework, we often specify protocols as pre/postcondition models; that is, models that specify a relationship between pre-states and post-states of protocols. The semantics of atomic protocols (and therefore compound protocols) does not support specifying postconditions as a relation with the pre-state. For example, consider a case in which we want to increment the integer value of a variable, x. The only way to specify this is to specify that the value of x is 1 greater than before the message was sent, which is not possible using an atomic protocol; the postcondition merely represents a constraint between state

variables,with no way of referencing pre-state values. However, we use the variable declaration operator to simulate this behaviour:

$$N \mathrel{\widehat{=}} \mathbf{var}_{x_0}^{x_0=x} \cdot x < 10 \xrightarrow{c.a(x)} x = x_0 + 1$$

Recall that the constraints placed on a locally declared variable are maintained throughout its entire scope. Therefore, the constraints on x_0 in the postcondition are that it equals the constraints on x in the pre-state. If we were to execute this message sending in the state $x = 1$, then the postcondition would resolve to the following: $\exists_x(x = 1 \sqcup x_0 = x) \sqcup x = x_0 + 1$. The only solution for this is $x_0 = 1 \sqcup x = 2$. The scope of the variable x_0 would end, and the post-state would be $x = 2$.

Annotating protocols using the rules in Section 4.2 would annotate this correctly, however, we would lose all information about the relationship between the pre-state and post-state. That is, we would have an annotation $[\pi]x < 11$, which contains no information about the relationship between the pre-state and $x < 11$. To preserve this relationship, we propose annotating such variable declarations in a different manner.

This new approach to annotation requires us to explicitly consider the pre-state in the annotation rules. If we label our pre-state as ψ_0, then rule is as follows:

$$x_0 = x \wedge \psi_0 \rightarrow [\pi]\phi_A \quad \rightarrow \quad [\mathbf{var}_{x_0}^{x_0=x} \cdot \pi]\exists_{x_0}(\exists_x(\psi_0 \sqcup x_0 = x) \sqcup \phi_A)$$

in which $vars(\phi_A) = x \cup x_0$. Therefore, this rule says that if ϕ_A is the maximal postcondition of π, then the maximal postcondition of the variable declaration protocol, under the initial state ψ_0, is calculated by assigning the pre-state occurrences of each variable x to a local variable x_0, and hiding this pre-state occurrence. Conjoin this with the postcondition, which specifies the relationship between each x and its pre-state counterpart, x_0. Finally, the local variables are hidden, because they are out of scope. When the agent reads this annotation, it substitutes in the initial state for ψ_0, giving it the postcondition.

As an example, take the protocol from above that increments a variable x. The annotation would be the following:

$$[\mathbf{var}_{x_0}^{x_0=x} \cdot \pi]\exists_{x_0}(\exists_x(\psi_0 \sqcup x_0 = x) \sqcup x = x_0 + 1).$$

Substituting in the current state, for example, $x = 1$, will result in the constraint

$$\exists_{x_0}(\exists_x(x = 1 \sqcup x_0 = x) \sqcup x = x_0 + 1)$$

which simplifies to $\exists_{x_0}(x_0 = 1 \sqcup x = x_0 + 1)$, which in turn, simplifies to $x = 2$.

We note that the above annotation is in fact meta-level, because ψ_0 in this case is a variable in the annotation, rather than just a meta-variable used to represent a constraint. Substituting in the current state for ψ_0 will give us the precondition/postcondition annotation. Unless the constraint solver supports constraints as variables, this substitution must be done before asking the constraint solver to provide a solution.

This approach overcomes many weaknesses of the previous approaches: it does not require the agent to perform entire proofs, therefore reducing the overhead of

the matching-via-proof method from Section 4; and it is not a general annotation, overcoming the problem of having to perform dynamic logic proofs if the goal entails the end state of the annotation, as described in Section 4.2. The obvious disadvantage to using this approach is that it is restricted only to protocols that are modelled using the precondition/postcondition approach.

6 Related Work

There are various aspects of related literature for this work. What we do not consider closely related are approaches using modal and dynamic logic for protocol specification, such as the work performed by Giordano et al. [5] and Brak et al. [1]. Their aim is to specific protocols using dynamic and modal logics, whereas our protocols are specified using the $\mathcal{RASA}$ language, and PDL is used to reason about the protocols.

Several authors have investigated the idea of executable protocol specifications. McGinnis and Miller [8] summarise and compare the work in this field.

As far as the authors are aware, there has been no investigation into the annotation or matching of protocols to date. Temporal projection, the process of calculating the outcome of given plan (or, more generally, the outcome of a sequence of actions), carries many of the same goals as our work. Much work has been done on temporal project using modal logic, such as early work by Hanks and McDermott [6], to more recent work, such as that by Predinger and Schurz [12]. Existing work on temporal projection is not complete for our purposes, because $\mathcal{RASA}$ protocols contain branching; plans are generally assumed to be single sequences of actions. Also, our work differs because we aim to annotate a program with its maximal postcondition, which will hold when executed from any state, whereas planners take into consideration only the current state of their system.

Our concept of protocol libraries is similar to plan libraries, such as those found in many BDI and goal-oriented agent framework, such the Procedural Reasoning System [4]. However, plan postconditions are typically specified manually, rather than calculated, and matching is performed by unifying the postcondition with the goal, which is different to our approach of matching, due to us using constraint languages rather than logical languages, and using maximal postconditions.

Clement and Durfee [2] present a method for summarising precondition and postconditions of hierarchical task network (HTN) plans. However, they consider only plans with branching, interleaving, and sequencing, and do not discuss summarising iterative or recursive plans.

Specification matching for component-based software engineering has been explored in the past, such as in [13]. Motivation for specification matching is similar to our motivation for protocol matching: to find a component that satisfies a specification. These approaches are considerably different to ours. They consider only pre- and postcondition models, and as a result, they do not consider cases in which the goal specification (the equivalent of an agent's goal) is stronger than the postcondition. This is not necessary in pre- and postcondition models.

7 Discussion and Other Work

In this paper, we have presented three methods for annotating and matching first-class protocols specified in $\mathcal{RASA}$. Each of these methods has advantages and disadvantages. The method that we believe is the most useful involves annotating a protocol with its maximal postcondition, and then using these annotations to match the protocols that an agent will test at runtime to find a protocol that achieves a given goal.

In related work, we are investigating how protocol libraries can be stored and searched for efficient protocol matching, as well as how to annotate recursively-defined protocols. An implementation of the annotation rules in this paper is also under development. We are also investigating several other interesting aspects of first-class protocols, such as runtime composition of protocols, which would permit agents to compose new protocols if no protocol can be found that achieves a certain goal. In future work, we plan to assess other techniques for protocol annotation and matching, such as analysing properties other than outcomes, for example, the number of participants.

Acknowledgements

The authors are grateful for financial support from the EC-funded PIPS project (EC-FP6-IST-507019), the EC-funded ASPIC project (IST-FPC-002307), and the EPSRC Market-Based Control project (GR/T10657/01).

References

1. Brak, R.L., Fleuriot, J.D., McGinnis, J.: Theorem proving for protocol languages. In: Proceedings of the European Union Multiagent Systems Workshop (2004)
2. Clement, B., Durfee, E.: Theory for coordinating concurrent hierarchical planning agents using summary information. In: Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference, pp. 495–502. AAAI, Menlo Park (1999)
3. De Boer, F.S., Gabbrielli, M., Marchiori, E., Palamidessi, C.: Proving concurrent constraint programs correct. ACM Transactions on Programming Languages and Systems 19(5), 685–725 (1997)
4. Georgeff, M.P., Lansky, A.L.: Reactive reasoning and planning. In: Proc. of the 6th National Conference on Artificial Intelligence, pp. 677–682 (1987)
5. Giordano, L., Martelli, A., Schwind, C.: Specialization of interaction protocols in a temporal action logic. Electronic Notes on Theoetical Computer Science 157(4), 3–22 (2006)
6. Hanks, S., McDermott, D.: Nonmonotonic logic and temporal projection. Artificial Intelligence 33(3), 379–412 (1987)
7. Harel, D., Kozen, D., Tiuryn, J.: Dynamic Logic. MIT Press, Cambridge (2000)
8. McGinnis, J., Miller, T.: Amongst first-class protocols. In: Artikis, A., O'Hare, G.M.P., Stathis, K., Vouros, G. (eds.) ESAW 2007. LNCS, vol. 4995, Springer, Heidelberg (2008)

9. Miller, T., McBurney, P.: Executable logic for reasoning and annotation of first-class agent interaction protocols. TR ULCS-07-015, University of Liverpool, Dept of Computer Science (2007)
10. Miller, T., McBurney, P.: Using constraints and process algebra for specification of first-class agent interaction protocols. In: O'Hare, G.M.P., Ricci, A., O'Grady, M.J., Dikenelli, O. (eds.) ESAW 2006. LNCS (LNAI), vol. 4457, pp. 245–264. Springer, Heidelberg (2007)
11. Miller, T., McBurney, P.: On illegal composition of first-class agent interaction protocols. In: Dobbie, G., Mans, B. (eds.) Thirty-First Australasian Computer Science Conference. CRPIT, vol. 74, pp. 127–136. Australian Computer Society (2008)
12. Prendinger, H., Schurz, G.: Reasoning about action and change: A dynamic logic approach. Journal of Logic, Language, and Information 5(2), 209–245 (1996)
13. Zaremski, A., Wing, J.: Specification matching of software components. ACM Transactions on Software Engineering and Methodology 6(4), 333–369 (1997)

Argumentation- vs. Proposal-Based Negotiation: An Empirical Case Study on the Basis of Game-Theoretic Solution Concepts

Angelika Först[1], Achim Rettinger[1], and Matthias Nickles[2]

[1] Department of Informatics
Technische Universität München
85748 Garching, Germany
angelika.foerst@gmail.com,
rettinger@cs.tum.edu
[2] Department of Computer Science
University of Bath
Bath BA2 7AY, UK
m.l.nickles@bath.ac.uk

Abstract. Recently, argumentation-based negotiation has been proposed as an alternative to classical mechanism design. The main advantage of argumentation-based negotiation is that it allows agents to exchange complex justification positions rather than just simple proposals. Its proponents maintain that this property of argumentation protocols can lead to faster and beneficial agreements when used for complex multiagent negotiation. In this paper, we present an empirical comparison of argumentation-based negotiation to proposal-based negotiation in a strategic two-player scenario. We apply a game-theoretic solution as a benchmark, which requires full knowledge of the stage games. Our experiments show that in fact the argumentation-based approach outperforms the proposal-based approach with respect to the quality of the agreements found and the overall time to agreement.

1 Introduction

Integration of individual entities into complex, open, and heterogeneous systems like the internet and peer-to-peer networks is ubiquitous. The potential of these systems is grounded in the interaction between their parts. Since they are often heterogeneous, interacting autonomous and intelligent agents [13] tend to have conflicting interests, but often they still can profit from coordinating their actions with other agents or even cooperating with each other. Hence, coordination techniques and mechanisms rapidly gain importance in the field of distributed artificial intelligence. Central to the concept of intelligent agents is their capability to reason about themselves and their environment. This aspect is usually not exploited by game-theoretic approaches [11] to automated negotiation and thus these approaches often lack flexibility. In recent years, argumentation-based negotiation [2] has been suggested as an approach to negotiation. It takes advantage of the abilities of intelligent agents to reason about rich interaction scenarios where complex justification positions (and not just simple proposals) can be

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 161–180, 2009.

exchanged [8,9]. Therefore this approach is currently enjoying increasing popularity in the field of negotiation research. However, until today, only very few approaches exist in which the performance of argumentation-based negotiating agents, bargaining agents and game-theoretic solution concepts can actually be compared in a specific scenario.

The problem definition of this paper is driven mainly by two aspects: Firstly, many negotiation settings are well-researched and have been analysed using game-theoretic techniques. The merits are that optimal negotiation mechanisms and strategies can be provided for a broad range of problems, which are also used in real-world scenarios, e.g. auctions. But then, the applicability of such solutions is often restricted to specific situations. Secondly, the emerging field of argumentation-based negotiation endeavours to overcome some of the fundamental limitations of the game-theoretic approach, notably partial knowledge, inconsistent beliefs and bounded rationality. Substantial work has been done in this field, and a number of implementations have been realised (see [5] for recent theoretical and software approaches to argumentation-based negotiation). However, until today, very little work exists in which different approaches are implemented and the performance of argumentation-based negotiating agents, bargaining agents and game-theoretic solution concepts can actually be compared in a specific scenario. The objective of this paper is to examine the benefits of different types of negotiation in a complex and stochastic environment in which agents only dispose of partial, incomplete knowledge. For this purpose, a negotiation framework is implemented, together with negotiating agents using different negotiation mechanisms. The performance of our solution concepts is evaluated empirically by benchmarking their performance against a provably optimal solution borrowed from game theory that requires complete and fully observable information.

Our evaluation shows that the different negotiation mechanisms that were tests can be clearly ranked with respect to their performance. The upper benchmark is set by the employment of a game theoretic mediator with complete knowledge who discharges the agents from negotiation by computing the optimal outcome for them. If agents are bound to negotiate under incomplete knowledge, the argumentation-based approach is clearly favourable to bargaining with respect to a number of evaluation criteria.

This paper is structured as follows: In the next section we introduce the environment within which the negotiating agents are situated. In Section 3 we present our solution concepts in an abstract form. The verification of our working hypothesis was conducted through extensive empirical evaluation - Section 4 is dedicated to the presentation of the experimental setup, the main experimental results, and an interpretation of our findings. Section 5 concludes with a summary and suggestions for future work on the topic.

2 The Testbed

In the following, we describe the testbed used for the subsequent evaluation and comparison task. Our testbed is designed in a way that makes the negotiation scenario complex enough to draw meaningful conclusions while keeping the negotiation processes comprehensible and analyzable. In game theoretic terms our scenario is based on the most general framework of games, namely general sum stochastic games [6,12]. In our case players additionally have to deal with incomplete and partially observable

information - as possessions of other players are not public - making it difficult to apply game theoretic solutions.

The scenario the agents are situated in is a production game. All players receive different kinds of resources. Each player tries to collect a certain number of resources of one type at a time to assemble products. By selling their products agents earn game points. The functionality of a player's resource store is equivalent to that of a FIFO (First In, First Out) queue. Hence, elements are added to one end of the queue (the *tail*), and taken off from the other (the *head*). The production unit however resembles a *stack* based on the LIFO (Last In, First Out) principle. Elements are added and removed only on one end. Thus, the game is called *Queue-Stack-Game*. One additional behaviour applies to the production units of this game. They can hold only one type of resources at a time and loose their previous content if new elements of a non-matching resource type are added.

Each round, every player is assigned a sequence of new resources, which are uniformly drawn from the available resource types. These elements are added in sequence to the tail of the queue. Next, a number of resources is taken off the head of the queue and added to the stack. As a consequence, the previous content of the stack might be lost if any of the new resources is of a non-matching type. To avoid this waste, players can negotiate with their peers and offer to give away resources from their queues. In doing so, they might be able to create sequences of identically typed resources of a certain length and thereby succeed in the game.

The following section describes the rules and phases of the Queue-Stack-Game in detail.

2.1 Production

There are a number of game parameters and restrictions that apply to the production process of the Queue-Stack-Game, which are listed here:

- Each agent can produce only one product at a time
- A product consists of a number of identically typed resources, this number being a game parameter, namely *stackCapacity*
- The types of resources and the order in which they are allocated to the producers are random. The number of resources each player receives per round is fixed though and is a parameter of the game, namely *getPerRound*
- The incoming sequence of resources cannot be altered by the agent before being added to the queue
- Each player is forced to input *pushPerRound* resources from the head of his queue into the production unit in each round
- If the type of any newly input resource does not match the type of the product being currently assembled, this product is spoiled and thrown away
- The players are admitted to remove elements of any types from their queue in order to give them to one of their fellow players
- If a player receives resources, he is allowed to arrange them in the desired order before they are immediately fed into the production unit

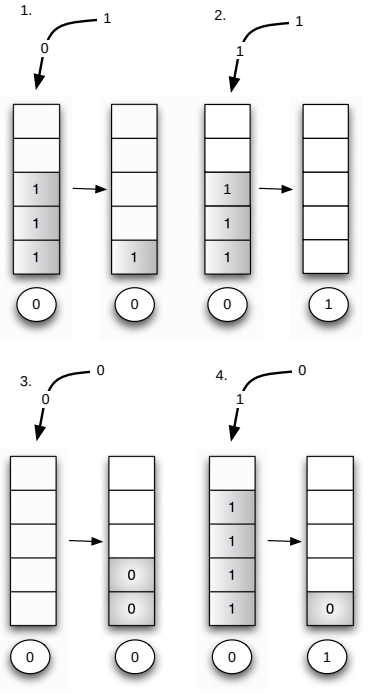


Fig. 1. Examples illustrating the behaviour of a player's stack when additional resources are pushed

2.2 Allocation

Each round is divided into two phases, namely *allocation* and *negotiation*. In the *allocation* phase, *getPerRound* new random resources are enqueued in all players' resource stores. The resources allocated to the different players are independently generated. Subsequently, each agent is forced to remove the *pushPerRound*-first elements from the head of his queue and to push them onto the stack, maintaining their ordering. If the production unit already contains some elements and their type does not match the newly pushed resources, the old contents of the stack are wasted. Figure 1 illustrates four examples of feeding resources into the stack.

The examples show the state of the stack before and after new resources have been pushed. We assume two different types of resources, **0** and **1**. The number of game points owned in the current situation is shown underneath each stack. In situation (1) all elements of the stack are discarded when the **0** token is pushed, as the types do not match. The **0** token itself is also thrown away, when the next resource, a **1** token is pushed. In situation (2), the player has more luck. The two resources pushed complete the product, which the player can sell and thus is rewarded. The production unit is empty now, ready to accept new resources of any type. Situation (3) shows how resources are added to an empty stack. In example (4) the first of the pushed resources completes the stack, the player sells the completed product, earns a reward and the stack is emptied before the next resource is pushed.

2.3 Generating Possible Worlds

We will now formalise the notion of a state in the Queue-Stack-Game and outline the process of generating a set of possible worlds with respect to a particular state. A state s_c contains the following elements:

- The condition of the queue after resources have been removed, referred to as $queue(s_c)$
- the condition of the stack after transfer received from another player has been pushed, referred to as $stack(s_c)$,
- the number of rewards, $rewards(s_c)$,
- the set of resources received from another player, $get(s_c)$,
- the set of resources removed from the queue in order to be transferred to the other player, $give(s_c)$,
- $noWaste(s_c)$, a flag indicating whether elements of the stack were wasted when $get(s_c)$ was pushed,
- $earnedReward(s_c)$ a flag being set to 1 if a reward was earned when pushing $get(s_c)$ or 0 otherwise.

The queue of a state s_c can be generated by removing each possible subset of resources from the previous queue $queue(s_{c-1})$. The removed resources are $give(s)$. Which resources can be received from other players is not known to the agent, as he has no insight into his opponents' resource situation. So all possible combinations of resource types up to an arbitrary total amount are considered. As the resources can be pushed in any order, $get(s_c)$ is generated for each permutation of the received transfer. $stack(s_c)$ is the resulting stack, after $get(s_c)$ has been pushed. $rewards(s_c)$ is the number of rewards the agent possesses afterwards. $noWaste(s_c)$ and $earnedReward(s_c)$ are needed when calculating the utility for a state.

The deal that produced a state is implicit to the state. When we speak of the utility of a deal, we mean the utility of the state which results from execution of the deal.

2.4 Evaluating Possible Worlds – the Utility Function

We now need a numerical *utility function* which measures the quality of a state. A utility function u maps a state or a sequence of states to a real number [10]. The following criteria could be used to describe a "good" queue.

1. The more resources the agent possesses, the better.
2. Blocks of identically typed resources contained in the queue should be of maximum length; ideally, the length is a multiple of the number of resources needed to earn a reward.
3. Preferably, no elements of the stack should be wasted when resources are pushed.
4. Resources at the head of the queue which are to be pushed in the next round should carry more weight than resources at the back end of the queue.
5. As few resources as possible should be given to other players.
6. As many resources as possible should be received.

Equation 1 captures criteria 1 and 2. It computes the base utility for a state s. In any sequence of resources, each element is either of type **0** or **1**, or white and black, respectively. Single elements of identical type are indistinguishable. Resource sequences can thus be represented as a sequence of blocks containing identically typed resources. $b_i(r)$ is taken to denote the ith block of a sequence r. $amount(b_i(r))$ is the number of resources $b_i(r)$ contains and $type(b_i(r))$ denotes the type of resources in block $b_i(r)$. k denotes the number of blocks in r. $stackCapacity$ is the capacity of the stack, in other words, the number of resources required to obtain one unit of reward. The sequence of all resources that a player possesses is the concatenation of his stack and queue. Concatenation is represented by the "|" operator.

$$baseUtility(s) = \frac{1}{k} \sum_{i=0}^{k} \frac{amount(b_k(stack(s)|queue(s)))}{stackCapacity}$$

Each block is considered as a fraction of a complete stack. *stackCapacity* resources in a row are equivalent to one unit of reward. The equation computes the average reward that can be achieved.

Next, we will describe the course of one round of the Queue-Stack-Game. Each round is divided into two phases, namely *allocation* and *negotiation*. In the *allocation* phase, *getPerRound* new random resources are enqueued in all players' resource stores. The resources allocated to the different players are independently generated. Subsequently, each agent is forced to remove the *pushPerRound*-first elements from the head of his queue and to push them onto the stack, maintaining their ordering. For details on the allocation phase please see Appendix 2.2.

Having completed the *allocation* phase, the players enter the *negotiation* phase. The outcome of a successful negotiation is a *deal*, describing which sets of resources are to be exchanged between players. Hence, the agents engage in *practical reasoning*. The exchange of resources is the only means for agents to take action during the game. If a player chooses not to negotiate or not to agree to any deal proposed to him, his succeeding in the game entirely depends on the random resource sequence he is allocated. If players cannot find an agreement, the *default deal* is forced. The default deal entails no actions of the players, thus the resource situation of all players remains unchanged. The available locutions are *propose*, *reject*, *accept* and *inform*. The *negotiation protocol*, i.e. the communication rules are defined as follows:

1. The negotiation terminates immediately after an acceptance message is uttered by one of the participants.
2. The negotiation terminates with the default deal if a player quits the negotiation.
3. The players take turns in proposing deals. If a player cannot propose a new deal, he is forced either to accept a previously offered deal or to quit the negotiation.
4. All deals offered during the negotiation can be accepted at any point in time later on as long as they have not been rejected.
5. A counterproposal can be preceded by a critique and a rejection.

This protocol entails that agents have to receive up to three messages (*inform*, *reject*, *propose*) until they are allowed to respond.

After the outcome of the negotiation is set, the deal is executed. The resources each player receives from fellow players are pushed onto the stack, whereby the player himself can dictate the order in which they are to be pushed. Eventually, the players are rewarded if they were able to complete their stack and thus sold a product.

3 Three Approaches to the Game

3.1 Employing a Mediator

Our first approach to designing successful players involves the consultation of a trusted mediator. We assume the mediator does not take part in the game and is unbiased towards any of the players. The players truthfully reveal their resource situation and their utility function to the mediator. The mediator has thus perfect information of the players' private states. Using this knowledge, all possible *offers* per player can be computed. Here, offer refers to a subset of the queue which the owner offers to give to an fellow player. The space of all possible deals is thus the Cartesian product of each player's offer vector. Through the utility function each player assigns a utility value to each possible deal. By knowing the utility functions, the mediator can compute these values per deal and player.

The next task is to determine the *optimal* deal for both agents. We adapt the axioms of the Nash Bargaining Solution [7] to define optimality: *Pareto efficiency* (there is no other deal which improves the payoff of at least one agent without another agent being worse off), *Invariance* (utility functions only represent preferences over outcomes, the actual cardinalities of the utilities do not matter), *Independence of irrelevant alternatives* (if outcome o is the solution and other outcomes $o' \neq o$ are removed from the set of all possible outcomes, o still remains the solution) and *Symmetry* (the optimal solution remains the same as long as the set of utility functions is the same. Which player has which utility function does not influence the outcome.) According to the *Nash Bargaining Solution*, the optimal deal o^* is the deal that maximises the product of the players' utilities. Formally:

$$o^* = \arg\max_o [(u_1(o) - u_1(o_{default})) \times (u_2(o) - u_2(o_{default}))]$$

where n is the number of players and $u_i(o')$ is the utility which player i assigns to deal o'. The mediator chooses the deal from the set of all generated deals that satisfies this equation. He then proposes this deal to the players, whom we assume to accept.

The advantage of the mediator approach is obvious. The outcome is guaranteed to be Pareto efficient. Hence, it is impossible to find a deal where both players are better off. The Nash Bargaining Solution respects each player's interests as far as possible without being biased towards any particular player, and promotes fairness. This approach has some shortcomings though, which limit its practicability. First of all, it requires the existence of a mediator whom the players trust, so they will reveal their utility functions and their resource situation. If the players are concerned with privacy issues in general and if they do not trust the meditator they might not agree to collaborate with that

mediator. Furthermore, they might not be content with the solution found, because there are deals with which the individual would be better off. The individual players are not necessarily interested in maximising the social welfare [4] but are only concerned with maximising their own profit. Additionally, the realisation of a mediator can be very complex and inefficient in real world scenarios.

The mediator will serve as a benchmark to which we compare the negotiation outcomes achieved by the argumentation-based agent described in the following sections.

3.2 Proposal- and Argumentation-Based Negotiating Agents

In this section we describe two designs of agents, both capable to negotiate by exchanging proposals with negotiation partners. While the proposal-based negotiating agent's abilities are restricted to the exchange of proposals, the argumentation-based negotiating (ABN) agent can use arguments to *justify* his negotiation stance and to *critique* proposals he has received from fellow players. Arguments can be arbitrary logical formulae with literals taken from a given vocabulary.

Algorithm 1. Negotiation strategy

```
receive δ_{j,r}
δ_{i,r+1} ← bestDealPossible()
δ_{j,best} ← bestDealReceived()
if δ_{i,r+1} = ⊥ then
    if utility(δ_{i,default}) ≥ utility(δ_{j,best}) then
        ACCEPT δ_{i,default}
    else
        ACCEPT δ_{j,best}
    end if
else
    if utility(δ_{i,r+1}) < utility(δ_{j,best}) then
        ACCEPT δ_{j,best}
    else
        allArguments[] ← generateArguments(δ_{j,r})
        if allArguments[] ≠⊥ then
            argument ← selectBestArgument(allArguments[])
            INFORM argument
        end if
        if utility(δ_{i,r}) < utility(δ_{j,best}) then
            REJECT δ_{j,r}
        end if
        PROPOSE δ_{i,r+1}
    end if
end if
```

Agent Architecture Overview. The architecture of our argumentation-based agent follows the abstract architecture described in [8]. All incoming proposals are stored in a Proposal Database. If arguments are employed, these are stored as well. As a model of his environment, the agent maintains a set of possible worlds to which he will possibly

agree. This set is continuously adapted during negotiation, i.e. possible worlds are removed after arguments or rejections of his proposals have been received and evaluated. According to his negotiation strategy, the agent then decides whether to accept or reject the last proposal. The ABN agent can then generate different types of arguments [3], in our case either a critique or a justification to inform his opponent why he is not inclined to accept the proposal. Of all generated arguments one is selected which will be uttered as a response. According to the adapted negotiation protocol (see Section 2.4), the agent cannot reply to every message received, but is bound to wait until he receives a proposal. So, not every incoming locution triggers an outgoing locution. The next sections describe the main components of the agent architecture in detail.

The Negotiation Strategy. In this section, we describe the negotiation strategy both our agents pursue. Each agent generates a set of possible deals which he will propose to his opponent one after another, starting with the deal with highest utility, followed by deals with descending utility. This ensures that the opponent knows all deals which would yield higher utility for the proposing agent, before being given the chance to accept a new deal. On the other hand, an agent waits until he is not able to make a proposal with higher utility himself before he accepts a deal. Thus, the use of this strategy aims to maximise the utility of the outcome for both players. Algorithm 1 shows the strategy of an agent using arguments in pseudo-code notation. Removing lines 14 to 18 yields the strategy of our proposal-based agent, who simply accepts or rejects deals without criticising them. Algorithm 1 is executed by the agents in every round of the game to determine the next locutions in the negotiation.

Deals received from the negotiation partner carry a j subscript, own proposals an i subscript. After agent j has offered deal $\delta_{j,r}$ to agent i in round r, agent i computes the best deal he is able to propose $\delta_{i,r+1}$ as a counterproposal (line 2). This is the deal with the highest utility based on the current resource situation, which has not been offered yet. Additionally, the deal with highest utility of all deals received from agent j in earlier rounds ($\delta_{j,best}$) is determined (line 3). If no proposable deal could be found agent i terminates negotiation. Either by accepting $\delta_{j,best}$ if executing this deal improves the utility compared to the current situation in round r or by accepting the default deal and thus leaving the resource situation unchanged. If the best proposable deal $\delta_{i,r+1}$ has lower utility than the best deal received already $\delta_{j,best}$, agent i accepts this $\delta_{j,best}$. Otherwise, the agent has incentive to pursue negotiation and proposes the best deal possible $\delta_{i,r+1}$ (line 22). Furthermore the agent will reject the latest offer if it is not the deal with highest utility of all offers received so far (lines 19 to 20). Lines 14 to 17 show the generation of all possible arguments concerning the latest offer and selection of the best argument which is then uttered before making the counterproposal $\delta_{i,r+1}$.

In summary, it can be stated that the agent will accept the deal with the highest utility of all deals he was offered (*bestDealReceived*) when all deals left to propose have lower utility. He will withdraw from the negotiation and thus accept the default deal if he cannot make any more proposals, but has not received any offer whose utility exceeds that of the current situation. An explicit reject is stated with respect to the current offer if there is already another offer with higher utility.

Generating and Selecting Arguments. Next, we explain how argument generation (line 14, "generateArguments") and argument selection (line 16, "selectBestArgument") is managed.

The negotiation language we designed contains just two basic elements. On the one hand, the statement $quit\_negotiation(agent)$, which an agent utters if he stops negotiating. On the other hand, $give(a, b, r, t)$ where a and b are agents, r is an amount of resources and t denotes a round of the game. The semantics of this statement is that agent a gives the resources r to agent b in round t. By combining statements using logical connectives, it is possible to create complex expressions with varying meaning. A deal, as it describes the exchange of resources between two players, consists of the conjunction of two statements:

$$give(a, b, r1, t) \ \wedge \ give(b, a, r2, t)$$

Arguments can serve two purposes in our approach: justification ("I cannot provide you with six white resources in the current round 13, because I only have four") or critique ("I reject your offer to give me four whites in exchange for three blacks in the current round 7, because I do not want to get four whites at all"). Each proposal is hence examined as to whether it contains one or more actions which either cannot be performed or are not desirable. An action is deemed not desirable if it is not contained in any deal considered in the agent's store of possible worlds. The argument generated then consists of the conjunction of the negated actions.

Here are two arguments which agent a sends to agent b. The following example corresponds to the above justification:

$$\neg give(a, b, fourWhites, 13) \wedge \neg give(a, b, fiveWhites, 13) \wedge \neg give(a, b, sixWhites, 13)$$

It states that the agent cannot give four, five or six white resources. A possible critique could be $\neg give(b, a, fourWhites, 7)$. Agent a does not want to receive four white resources under any circumstances.

The question of which of all arguments generated is to be uttered is answered by one simple rule: If a justification was generated and it has not yet been uttered, it will be selected. Otherwise, the critique is selected.

Interpreting Arguments. Now we will address the issue of how to evaluate incoming arguments. Arguments are statements about the opponent's mental attitude, i.e. his beliefs about possible worlds. Consisting of formulas on propositions of the form "*give(a, b, r, t)*", they describe the set of deals he might be willing to accept at all. Hence they are used to refine the set of possible offers. We assume our agents to be honest, so arguments are believed to be true. If an argument is received, its interpretation with respect to the current set of possible worlds is determined. The interpretation is a subset of the universe (the current set of possible worlds). This subset is then regarded as the new set of possible worlds. A set of possible worlds can be regarded as a logical formula of the form

$$\underbrace{give(a,b,r_1,t) \wedge give(b,a,r_2,t)}_{deal_1} \vee \ldots$$

$$\ldots \vee \underbrace{give(a,b,r_n,t) \wedge give(b,a,r_m,t)}_{deal_k}$$

The r_i stand for arbitrary sets of resources. A r_i can appear in several deals.

Incoming arguments are transformed into a normal form, so that negations are pushed inward and all operators but $\vee$ and $\wedge$ are resolved. Then the subset of possible deals, which is denoted by the argument, is determined using the following inductive definitions in Table 1 where ϕ and ψ are any arguments.

Table 1. Interpretation of arguments as subsets of possible worlds

Argument	Interpretation
$give(a,b,r,t)$	set of all deals which include $give(a,b,r,t)$
$\neg give(a,b,r,t)$	set of all deals which do not include $give(a,b,r,t)$
$\phi \wedge \psi$	all deals which are elements of the intersection of the sets which are the interpretation of ϕ and ψ
$\phi \vee \psi$	all deals which are elements of the union of the sets which are the interpretation of ϕ and ψ

4 Evaluation

This section describes the empirical evaluation conducted to answer our central research questions: Do agents who use arguments in the negotiation perform better in our complex trading scenario than agents who are confined to exchanging proposals? Are agents using argumentation-based negotiation capable of reaching optimal deals? In the first section, we introduce the evaluation criteria for which data was gathered during the test runs. Section 4.2 describes the experimental setup. Finally, in Section 4.3, we evaluate our experimental findings with respect to our key research questions.

4.1 Evaluation Criteria

In our experimental evaluation, we consider a number of evaluation criteria which allow for measuring certain aspects of agent performance. The following subsections introduce these criteria one by one. Moreover, we outline why and to what extent we consider the criteria to be suitable metrics for evaluation.

– Rewards: Rewards earned over time are the most natural criterion for the Queue-Stack-Game, as this measure is used to determine the winner after one or more rounds, and hence it also reflects the game-playing ability of any given agent strategy. The rewards a player has earned are influenced mainly by two factors. On the one hand, they depend on how lucky the player has been in terms of the resources that were randomly allocated to him. On the other hand, their number increases with the quality of the game strategy adopted, and in particular also with the quality of the chosen negotiation strategy. Hence, it is important to have several rounds in one game, so that the distribution of resources becomes fair.

- Social Welfare: Social welfare is a means of assessing the well-being of a society of agents as a whole, i.e. taking into account the well-being of all individual agents [1]. In literature, there are different measures for social welfare, e.g. *Egalitarian* or *Utilitarian* social welfare. We employed the Nash Bargaining Solution which is proven to promote pareto-optimal deals, which means that no other deal is preferred by every other agent. The mediator computes the optimal deal according to this measure. The deals achieved by each of the negotiation methods can be compared to this optimal deal and thus a "degree of optimality" can be established for each method. Furthermore, the deals achieved by ABN and by bargaining can be compared to each other with respect to the optimal deal. Social welfare is also an adequate indicator if all agents get better deals by arguing, or if, e.g., an increase of utility of Player1 can only be realised at Player2's expense, thus promoting unjust deals.
- Number of Communication Units: As a measure for the amount of communication, we define a communication unit for our negotiation language. Deals and arguments are generated by combining different statements of the form $give(a, b, r, t)$. We define this as one communication unit. Operators are not accounted for by this measure, i.e. a conjunctive expression has the same "value" as two atomic expressions. Similar to the other measures discussed above, the absolute number of communication units is meaningless in itself. However, counting the communication units allows for a comparison of the amount of communication across the different negotiation mechanisms. In the ABN approach, we distinguish communication units that were sent as part of proposals and messages that carried arguments.
- Number of Negotiation Steps: The number of steps the agents negotiate for per round is an apt measure to determine the speed of a negotiation style. Reaching an agreement in fewer steps is considered better, if the agreement is as good as the agreement that could have been reached within an unbounded number of negotiation rounds. Even if no agreement is reached, the less time and effort is invested to find out that no co-operation is possible or desired, the better.
- Number of Possible Worlds: Our agents maintain a changing number of possible worlds during negotiation. The absolute number of possible worlds is not interesting *per se*. Their number is highly dependent on the current resource situation of the agent, and can initially be very large. What we really want to show is that our agents are capable of reducing the number of possible worlds while negotiating. Examining the decrease in possible worlds over negotiation steps offers valuable clues on how the negotiation outcomes are achieved. This is because the number and the utility of the remaining possible worlds directly influence an agent's decision to accept or reject an offered deal.

4.2 Experimental Setup

In the following, we describe the tests that were run to generate the test data. The Queue-Stack-Game was played in three different settings. Scenario one comprises a meditator as described in Section 3.1 in addition to two players. In the second scenario, players can exchange deals but not arguments. Finally, in the third scenario, players are capable of exchanging arguments in addition to proposals. Ten consecutive games consisting

Table 2. Frequency of data collection

Test Criterion	Mediator	Player
Nash Bargaining Solution	end of round	n/a
Utilitarian Social Welfare	end of round	n/a
Egalitarian Social Welfare	end of round	n/a
Rewards	n/a	start and end of round
Utility	n/a	start and end of round
Communication Units (Proposals)	n/a	end of negotiation
Communication Units (Arguments)	n/a	end of negotiation
Negotiation Steps	n/a	end of negotiation
Possible Worlds	n/a	each negotiation-step

of ten rounds per game are played in all three scenarios. The sequences of resources which are allocated to each player from the deck in each round are identical in all three settings. What is left to chance is the random time the agents wait before uttering their initial proposal as soon as their *NegotiationBehaviour* is started. In summary, the course of the game in our three settings is solely dependent on the negotiation and its outcome. Hence, we can draw conclusions from the players' success in the game to their ability to negotiate.

The data for the empirical evaluation of the scenarios is logged by the players and the mediator, if existent, during the test runs. The criteria are logged with different frequency and in different phases of the game. Table 2 provides an overview of all logged test variables.

4.3 Evaluation of Experimental Results

In this section we evaluate and interpret our experiments with respect to the above mentioned test criteria.

Rewards. Earned rewards measure an agent's success in the Queue-Stack-Game. The graphs in Figure 2 show the number of rewards earned by players Player1 (top) and Player2 (bottom) in every round of the game, respectively. The recurring decline of rewards is due to the start of a new game every ten rounds. That is, the agents are restarted with zero rewards in every eleventh round. Figure 3 shows the average reward per round earned by both players in different test scenarios. Both agents perform best when guided by a mediator, which matches our initial expectations. Comparing the scenarios where the agents actually negotiate with each other shows that both agents achieve better results when using arguments in addition to the proposal exchange.

Social Welfare. Figure 4 depicts the average social welfare for each experimental setting, namely "Argumentation", "Proposal exchange" and "Mediator use". Different measures for social welfare were used, the most common being Utilitarian social welfare (sum of each player's payoff) and the Nash product (product of each player's payoff). We computed social welfare on the basis of actual rewards, not based on the utility of the negotiation outcome for reasons described above. As a matter of design, the mediator maximised the Nash product of the players' utilities, which are a heuristic for the

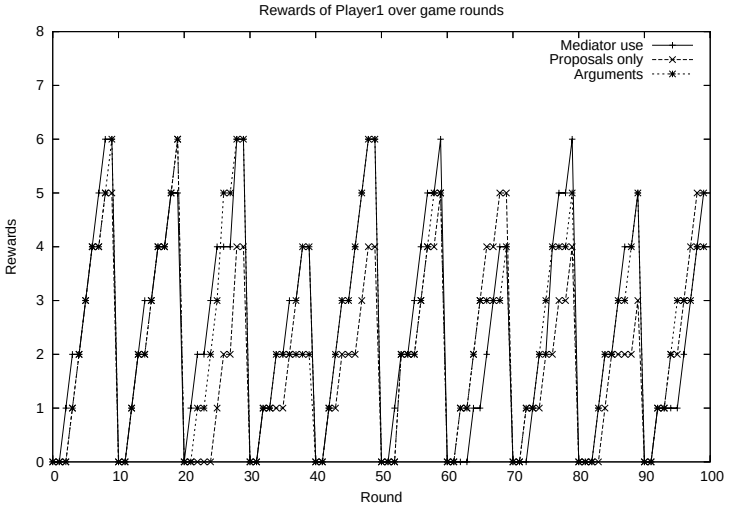


Fig. 2. Earned rewards over game rounds of Player1 for different test scenarios

expected reward. Quite naturally, this lead to the best average game results compared to the other mechanisms and considering any of our suggested measures. Likewise, it becomes apparent that the agents of the "Argumentation" scenario achieved the second best results and thus performed better than negotiating agents who were restricted to proposal exchange.

Table 3 summarises the percentage of games won by each player. The number of successful negotiations which ended with an agreement can be increased by 19,7 % from 66 to 79 of 100 by the use of arguments. Whereas in scenario 2 both agents accepted equally often, Player1 ended 9 more negotiations with an acceptance in scenario 3 whereas Player2 accepted in just two more negotiations.

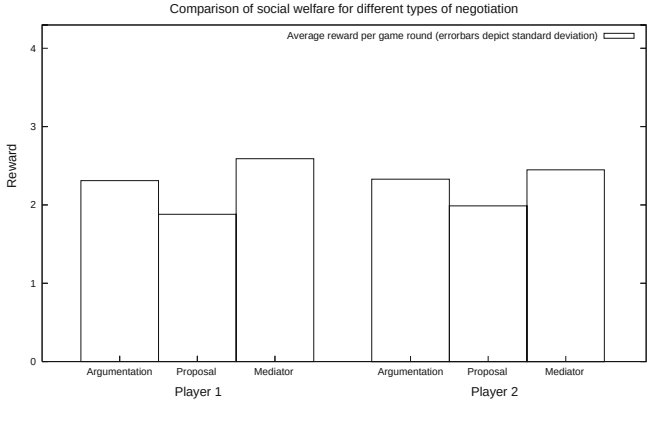


Fig. 3. Average rewards earned per game in different test scenarios. Left: Player1, Right: Player2.

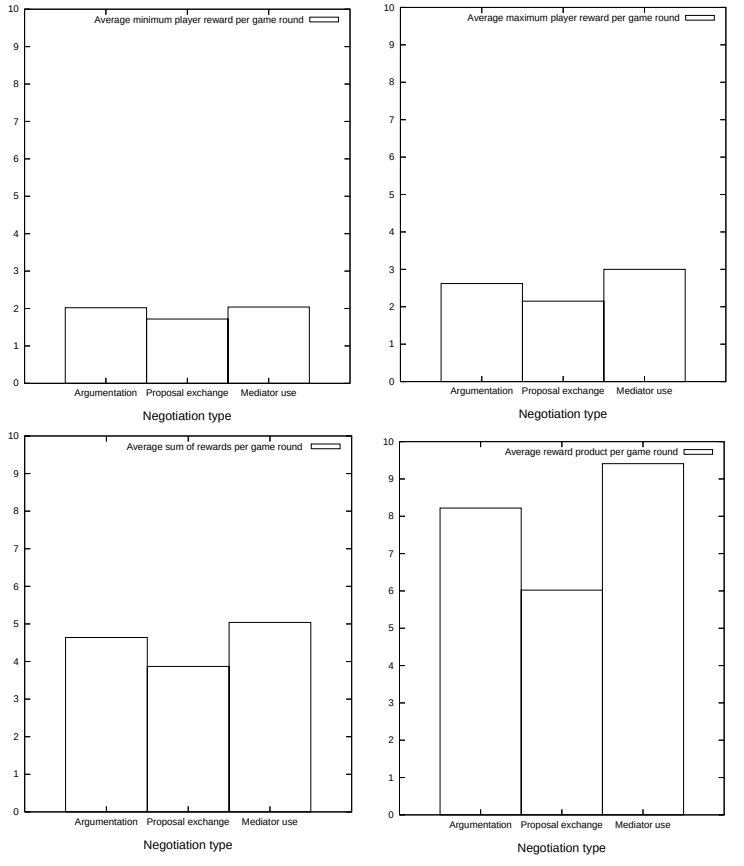


Fig. 4. Comparison of average social welfare based on earned rewards for "argumentation" (left), "proposal exchange" (middle), "mediator use" (right). Measures for social welfare (from top to bottom, left to right): minimum (egalitarian social welfare), maximum (elitist), sum (utilitarian social welfare), product (Nash product).

Possible Worlds. The agents in scenario 2 and 3 mainly differ in the way an agent's set of possible worlds is maintained. The exchange of arguments aims at the refinement of the set of possible worlds, and thus the removal of worlds that are not acceptable to any of the agents. Hence, we look at how the number of possible worlds changes over negotiation rounds.

In Figure 5 the average decrease of possible worlds over rounds is plotted for Player2 for the two negotiation scenarios. The curve for agent Player1 is almost identical, we therefore omit it.

Comparing the plots of the two different test scenarios, one observation is obvious: The decrease of possible worlds proceeds much faster when arguments are used. After ten negotiation steps, agents in scenario 2 still maintain more than 80% of the worlds they initially considered possible on the average. By that time, agents in scenario 3 have removed over 80% of their initial worlds and maintain only less than 20% after ten steps.

Table 3. Acceptance rates of negotiation scenarios

Scenario	Player1	Player2	No agreement
2: Proposals only	32%	34%	33%
3: With arguments	43%	36%	21%

After termination of the negotiation process, ABN agents have eliminated about 65% of their initially possible worlds on the average, their counterparts in scenario 2 have been able to remove a mere 42%.

Negotiation Steps. Considering the average number of negotiation steps, agents of the different scenarios needed to come to an agreement, the average of 39 steps of the scenario with argumentation lies clearly under the average of 76 of the scenario where only proposals are exchanged, see Figure 6. This means that when using arguments the negotiation terminates after little more than half of the steps required using pure proposal exchange in the average. This is due to the faster decrease of possible worlds in scenario 3, and it is also due to the fact that once the negotiation has started no arguments can be produced which entail an increase of possible worlds. In the few cases where the ABN agents need more steps to come to an agreement, this can be still be justified by the better negotiation outcome these agents achieve compared to their proposal-exchanging counterparts.

Communication Units. The total number of communication units (in the sense defined above) averages 113.12 in scenario 2 and 44.01 in scenario 3. Latter number is composed of 12.87 units used on arguments and 31.14 units describing outcomes. Even the sum of proposals and arguments in scenario 3 does not get close to the average of

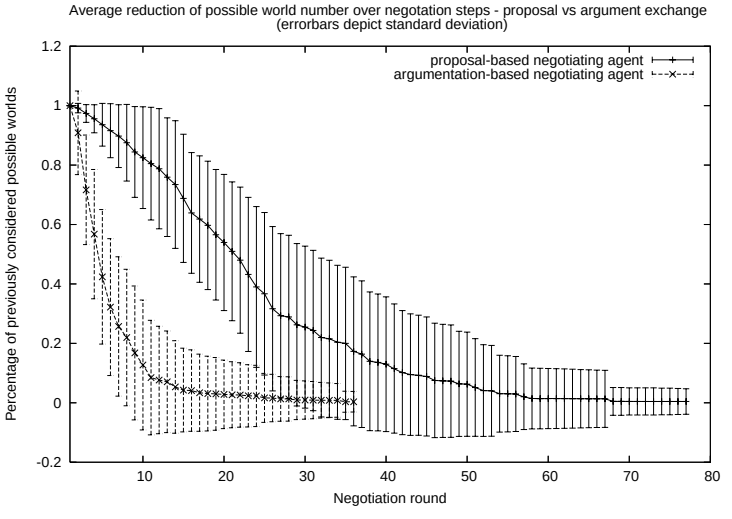


Fig. 5. Player2: Average reduction of possible world number over negotation steps in scenario "proposal exchange" (continous errorbars) and "argument exchange" (dashed errorbars)

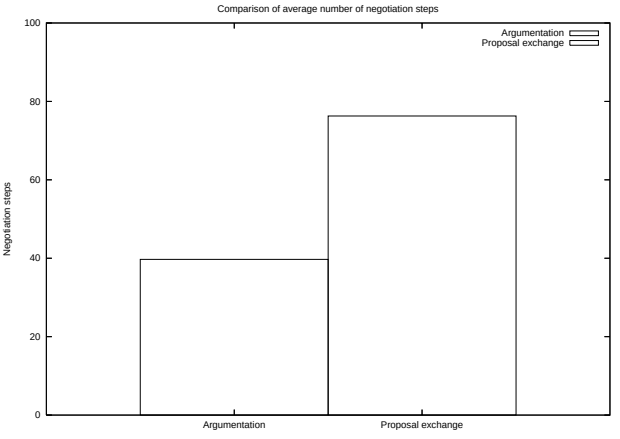


Fig. 6. Average number of negotiation steps: "argumentation" (left), "proposal exchange" (right)

units exchanged for proposals in scenario 2. This result is obtained by the richer semantics of the language which is used for argument exchange. The use of logic allows for using concise descriptions of subsets of possible worlds. If the negotiation language is restricted to deals and a set of possible deals is to be encoded, there is no alternative to enumerating the elements of this set. As the elements are deals and each deal equals two communication units, the number of units needed to encode a set is twice the cardinality of the set. Using logic this number still constitutes the worst case, but due to dependencies between different deals which contain identical actions, a subset of possible worlds can usually be encoded with fewer communication units. Hence, the use of logic as negotiation language allows for the reduction of communication overhead.

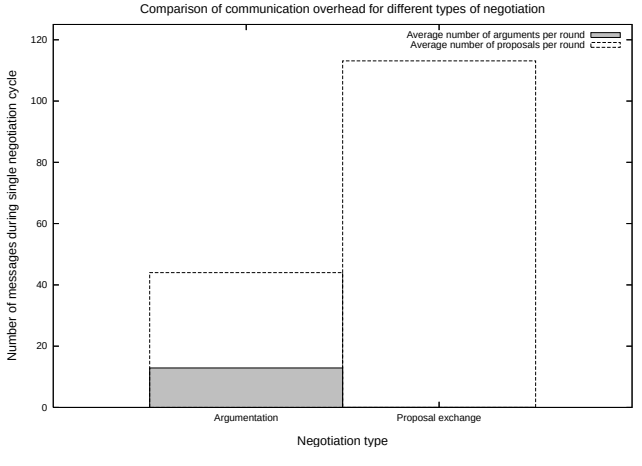


Fig. 7. Average number of communication units sent during negotiation

4.4 Summary

In this section we introduced the evaluation criteria that were considered in the conducted experiments. We then described the experimental setup for the data generation. Furthermore, we were able to show through a thorough analysis of experimental results that additional use of arguments during negotiation in the Queue-Stack-Game not only drastically reduces the duration and communication overhead of negotiation, but also that the quality of the achieved agreements is higher (in the sense that the resulting deals cause a higher increase in agents' payoffs and thus they perform better in the game). This is due to the refinement of the set of possible worlds by exchanging arguments which accompany the rejection of deals. Not only is the actually rejected deal eliminated from the opponent's set of possible worlds, but so is also every deal that shares the undesired aspects that caused the rejection of the explicitly proposed deal.

5 Conclusion and Future Work

In this work we presented the implementation of three different negotiation mechanisms in an environment which is only partially observable, i.e. the state and the preferences of an agent's peer are not known to him, and which incorporates stochastic elements, i.e. subsequent states are not solely dependent on the actions which are carried out by agents. Two agents are randomly assigned resources which they can use in a specific manner to earn rewards. In most cases agents need to exchange resources with their opponent to be successful. Hence, the agents need to come to an agreement about which type of resources and how many of them they want to exchange.

We approached this problem from three different angles. Our first solution was the employment of a trustworthy mediator, toward whom the agents disclose their preferences and resource situation. Using this complete information, the mediator can compute the optimal solution and dictate the outcome of the negotiation. Our second solution comprised agents who engaged in bargaining. They were restricted to a simple exchange of proposals to come to an agreement. Then, in our third scenario we provided the negotiating agents with the additional capability to accompany proposals or rejection of proposals with arguments. These arguments can either be a detailed critique of an previously received proposal, telling the opponent exactly which aspects of the proposal are undesirable. Or, an argument can carry information about the sender's negotiation stance and thus explain why a proposed deal cannot be fulfilled by the sender.

We extensively tested these solution concepts in identical experimental settings, allowing for a detailed comparison of the performance of the three negotiation mechanisms. As expected, agents perform best when consulting a mediator. When actually engaging in negotiation with each other, the use of arguments proves beneficial in various ways. Not only decrease communication overhead and duration of negotiation significantly, but agents simultaneously reach better agreements. Hence we were able to verify our working hypothesis, that the use of arguments enables better deals in an generic example scenario with partial, incomplete knowledge compared to negotiation that is purely based on proposal exchange.

A number of aspects could not be addressed and were beyond the scope of this paper. The following is a list of issues that could be the basis for future research:

- Although agents receive information about the internal state of their opponent, they do not actually try to create a model of their opponent, which they could use over several rounds. This aspect gains importance if agents are allowed to cheat. From the offers the opponent has made his resource situation could at least be inferred partially or inconsistencies in his offers and arguments could be detected.
- In the light of potentially agents that are not trustworthy it is necessary to reassess the process of argument evaluation. If the truthfulness of the arguments cannot be taken for granted, it is not advisable to accept all implications of the arguments without examining whether one believes the argument or not.
- Commitment to future actions is not considered as potential part of an agreement, even though the design of the negotiation language would allow it.
- Our agents have not been equipped with the ability to learn or plan, two essential aspects of intelligent agents.
- Also the use of the mediator could be reassessed. Players might not be required to execute the deal they have been advised to perform. They could bear that deal in mind and engage in negotiation nonetheless, leaving open which agreement they will pursue. Again, this problem is within the scope of research on computational trust.

By our strong efforts to keep our implementation generic in the choice of the tools and design we hope to contribute to the further investigation of these important issues.

References

1. Eatwell, J., Milgate, M., Newman, P. (eds.): The New Palgrave: A Dictionary of Economics, vol. 2, pp. 460–482. Macmillan, London (1987)
2. Jennings, N.R., Parsons, S., Noriega, P., Sierra, C.: On argumentation-based negotiation. In: Proceedings of the International Workshop on Multi-Agent Systems, Boston, USA (1998)
3. Kraus, S.: Automated negotiation and decision making in multiagent environments. In: Luck, M., Mařík, V., Štěpánková, O., Trappl, R. (eds.) ACAI 2001 and EASSS 2001. LNCS, vol. 2086, pp. 150–172. Springer, Heidelberg (2001)
4. Lomuscio, A., Wooldridge, M.J., Jennings, N.R.: A classification scheme for negotiation in electronic commerce. In: Sierra, C., Dignum, F.P.M. (eds.) AgentLink 2000. LNCS, vol. 1991, p. 19. Springer, Heidelberg (2001)
5. Maudet, N., Parsons, S., Rahwan, I.: Argumentation in multi-agent systems: Context and recent developments. In: Maudet, N., Parsons, S., Rahwan, I. (eds.) ArgMAS 2006. LNCS, vol. 4766, pp. 1–16. Springer, Heidelberg (2007)
6. Murray, C., Gordon, G.: Multi-robot negotiation: Approximating the set of subgame perfect equilibria in general sum stochastic games. In: Schölkopf, B., Platt, J., Hoffman, T. (eds.) Advances in Neural Information Processing Systems, vol. 19. MIT Press, Cambridge (2007)
7. Nash, J.F.: The bargaining problem. Econometrica 18(2), 155–162 (1950)
8. Rahwan, I., Ramchurn, S., Jennings, N., McBurney, P., Parsons, S., Sonenberg, L.: Argumentation-based negotiation (2004)
9. Rahwan, I., Sonenberg, L., McBurney, P.: Bargaining and argument-based negotiation: Some preliminary comparisons. In: Proceedings of the AAMAS Workshop on Argumentation in Multi-Agent Systems, New York (2004)

10. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach. Pearson Education, London (2003)
11. Sandholm, T.W.: Distributed rational decision making. In: Weiß, G. (ed.) Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, pp. 201–258. MIT Press, Cambridge (1999)
12. Wang, X., Sandholm, T.: Reinforcement learning to play an optimal nash equilibrium in team markov games. In: Becker, S.T.S., Obermayer, K. (eds.) Advances in Neural Information Processing Systems, vol. 15, pp. 1571–1578. MIT Press, Cambridge (2003)
13. Weiß, G. (ed.): Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. MIT Press, Cambridge (1999)

Argumentation-Based Information Exchange in Prediction Markets

Santi Ontañón[1] and Enric Plaza[2]

[1] CCL, Cognitive Computing Lab Georgia Institute of Technology,
Atlanta, GA 303322/0280
santi@cc.gatech.edu
[2] IIIA, Artificial Intelligence Research Institute - CSIC, Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Catalonia, Spain
enric@iiia.csic.es

Abstract. The purpose of this paper is to investigate how argumentation processes among a group of agents may affect the outcome of group judgments. In particular we will focus on prediction markets (also called information markets) and we will investigate how the existence of social networks (that allow agents to argue with one another to improve their individual predictions) effect on group judgments. Social networks allow agents to exchange information about the group judgment by arguing about the most likely choice based on their individual experience. We develop an argumentation-based deliberation process by which the agents acquire new and relevant information. Finally, we experimentally assess how different social network connectivity and different data distribution affect group judgment.

1 Introduction

The purpose of this paper is to investigate how argumentation processes among a group of agents may affect the outcome of group judgments. In particular we will focus on prediction markets (also called information markets) and we will investigate how the existence of social networks (that allow agents to argue with one another to improve their individual predictions) effect on group judgments using prediction markets.

There are different ways to aggregate the information held by a group of agents. According to C. R. Sunstein [17] there are three main paradigms to achieve *group judgments*, that is to say a joint decision or prediction based on aggregating the information or preferences of a group of agents (Sunstein deals with human agents, while we will focus only on artificial software agents). One paradigm is using statistical means to aggregate the group information: techniques like plurality voting, Condorcet voting or weighted voting define aggregation functions based on statistical means (i.e. on diminishing the joint error). Human committees, panels and juries use these techniques — and groups of agents also, see for example [11] where learning agents' joint predictions are compared when using plurality voting vs. weighted voting.

A second paradigm is that of deliberation, where arguments in favor or against a joint judgment are exchanged by the member agents of a group. Human public and private institutions traditionally favor deliberative ways of taking decisions, and certain

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 181–196, 2009.

accounts of democracy are based on the deliberation process. The main feature here is that rough preferences are not considered sufficient to justify a joint judgment, and deliberation provides reasons by an exchange of *arguments* by individuals with different information and diverse perspectives. Agents can also use argumentation to deliberate on joint judgments, as for example in the work reported in [13].

The third paradigm is the one this paper focuses on: *prediction markets*, also known as *information markets*. Prediction markets' goal is to aggregate information based on a *price signal* emitted by the members of a group. The advantage of the price signal is that it encapsulates both the information and the preferences of a number of individuals. In this approach, the task of aggregating information is achieved by *creating a market*, and that market should offer the right *incentives* for the participating people or agents to disclose the information they possess.

The purpose of this paper is to analyze the effect of social network relationships in group judgment —specifically in prediction markets. These social networks allow agents to exchange information about the prediction task domain. We model this information exchange as an argumentation process, where an agent A tells an agent A' its prediction S together with an argument α intended to justify why this prediction is correct. Agent A can agree or disagree with S, and in the case of disagreement A' communicates to A a counterargument or a counterexample that contradicts α. Agent A may keep its original prediction S or change it to some new prediction S' due to the counterarguments and counterexamples A has exchanged with one or more other agents. Social networks establish the different possible graphs of trusted acquaintances with which an agent can soundly exchange information; several simple social networks are tested in order to analyze the impact of information exchange.

The structure of the paper is as follows: the next section describes the Multiagent Prediction Market (MPM) and discusses the assumptions to use such mechanism for group judgment; section 3 describes the argumentation processes among agents that models the information exchange among agents; then section 4 presents an empirical evaluation of MPM in a prediction domain and we assess (1) the effect of using a prediction market instead of a voting scheme, and (2) the effect upon prediction markets of information exchange. Finally, section 5 presents related work and section 6 discusses the contributions of the paper and the foreseeable future work.

2 Multiagent Prediction Market

Essentially, a Multiagent Prediction Market (MPM) is composed of (a) a *prediction task domain*, (b) a market broker agent A_D, (c) a collection of participating agents $\mathcal{A}$, and two parameters: M (maximum bet) and X (a percentage bonus).

In this paper we will address only single-issue predictions and we will assume that the prediction task domain is characterized by an enumerated collection of *alternatives* or *solutions* $\mathcal{S} = \{S_1, ..., S_K\}$ and the prediction task is to select the correct one for the current *situation* or *problem* P. The participating agents is a multiagent system composed of n agents $\mathcal{A} = \{A_1, ..., A_n\}$. For a specific market, given a *problem* P every agent receives P, generates its individual prediction, and then it can bet up to a quantity M_P on one single alternative.

Let $B_{A_i} = \langle S, b \rangle$ be the bet made by a particular agent A_i, where S is the predicted solution, and b is the amount bet. Let $\mathcal{B}_\mathcal{P} = \{B_{A_i}, ..., B_{A_n}\}$ be the set of all bets made by all the agents in the market MPM_P. We will use the dot notation to refer to elements inside a tuple, e.g. we will write $B.b$ to refer to the amount bet in B. We define $B_P = \sum_{B \in \mathcal{B}} B.b$ as the total amount of money bet by all the agents, and $B_{S_k} = \sum_{B \in \mathcal{B} | B.S = S_k} B.b$ the total amount of money bet for a particular solution S_k.

The broker agent A_D receives those bets (amounting to a total quantity B_P) and determines the joint prediction as the alternative (say S_r) invested with the highest accumulated bet, as follows: $S_r = \arg\max_{S_k \in \mathcal{S}} B_{S_k}$. When the correct solution S_c of P becomes known, the broker agent A_D checks whether the joint prediction was accurate ($S_r = S_c$). If it was, then those agents that bet for S_c receive a reward. Specifically, an agent A_i who bet for the correct solution receive the reward $r_{A_i} = \frac{1}{B_S}(B_P \times B_{A_i}.b \times c)$, where $c = \frac{100+X}{100}$ is a factor that ensures that the agents receive more money than they bet if they win. Intuitively, the winner agents receive all the money bet by all the agents (i.e. B_P), but multiplied by the factor c, to provide an incentive. In our experiments we have set the percentage bonus $X = 10\%$, thus, $c = 1.1$.

The rationale of this design is to provide a twofold incentive: a) for the agents to reveal their true prediction, and b) also to benefit from the the joint accuracy.

Concerning the participating agents, we make the assumptions that (1) the individual agents possess a way to determine the confidence in an individual prediction and (2) the agents possess an argumentative capability that supports the information exchange with other agents regarding the prediction task domain. The first assumption requires that the agent is not only capable of making a prediction, but also establishing the likelihood of that specific prediction to be correct, i.e. a degree of confidence for each specific prediction. Rationality dictates that the more confident an agent with respect to a prediction, the higher the quantity to bet on that prediction. The second assumption allows the agents to perform an information exchange phase (that we model as an argumentation process), and thus generate more informed predictions.

3 Information Exchange in Social Networks

Social networks views social structures as composed of nodes and links, where nodes are individuals or organizations and links are their relationships. For the purpose of this paper, we will focus on individual agents as nodes and acquaintances as their links.

In our framework, a social network is a collection of *acquaintance* directional relations $N = \{(A_{i_1}, A_{j_1}), ..., (A_{i_m}, A_{j_m})\}$, where an agent A_i has another agent A_j as an acquaintance only if $(A_i, A_j) \in N$. Figure 1 shows three examples of social networks: In the leftmost one, each agent has one acquaintance, in the middle one, each agent has two acquaintances, and in the rightmost one each agent has three acquaintances.

Before declaring a prediction on the market, an agent A_i will first try to exchange information with its acquaintances. Thus, A_i will engage in argumentation processes about the correct solution of the problem at hand with each of its acquaintances before making a prediction — following the argumentation formalism we introduced in [13].

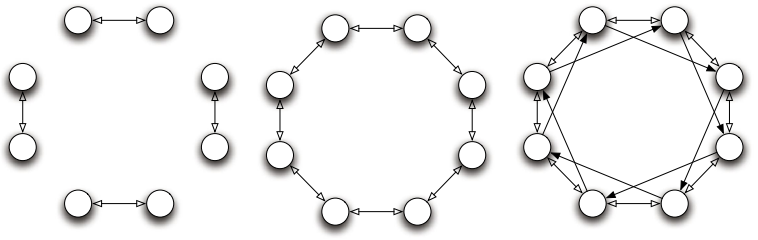


Fig. 1. Three of social networks among 8 agents where each agent has 1, 2 or 3 acquaintances

3.1 Problem-Centered Information Exchange as Argumentation

An agent A_i can obtain new information concerning the solution of a problem P by engaging in an argumentation process with another agent A_j, that might have information unknown to A_i. During an argumentation process, two agents exchange information concerning the solution of a specific problem P. Specifically, an agent may generate an argument in favor of a particular solution and send it to the other agent. Agents can also analyze a received argument, and agree or disagree with it. When an agent disagrees with an argument, it might generate a counterargument or a counterexample. By exchanging arguments and counterarguments two agents may reach a consensus about which is the most plausible solution for a given problem taking into account the information that both of them have. Therefore, the individual solution reached after an argumentation process is in principle more informed, and thus more likely to be correct.

3.2 MPM with CBR Agents

In our framework, each agent uses Case-Based Reasoning (CBR) [1] in order to generate predictions. Thus, each agent A_i owns a case base C_i, composed of a collection of cases, $C_i = \{c_1, ..., c_m\}$. A case is a tuple $c = \langle P, S \rangle$ containing a case description P and a solution $S \in \mathcal{S}$. We will use the terms *problem* and *case description* indistinctly.

CBR agents can solve problems by themselves, using CBR problem solving methods. Moreover, agents can also try to obtain information from other agents in order to increase their prediction accuracy. In a prediction market, given that each individual agent is interested in maximizing its prediction accuracy (in order to obtain a higher reward), it is rational for an agent to try to obtain the maximum information possible from other agents before making its prediction.

Argumentation provides a formal and well founded way to problem-centered information exchange. We will next summarize the case-based approach to multiagent argumentation introduced in [13]: the kind of arguments and counterarguments supported, how CBR agents generate arguments, and how agents compare arguments. Finally, we will present a specific argumentation protocol for information exchange in prediction markets, that agents can use to increase the accuracy of their predictions.

3.3 Arguments and Counterarguments

For our purposes an *argument* α generated by an agent A is composed of a statement S and some information D endorsing the fact that S is correct. In the context of CBR

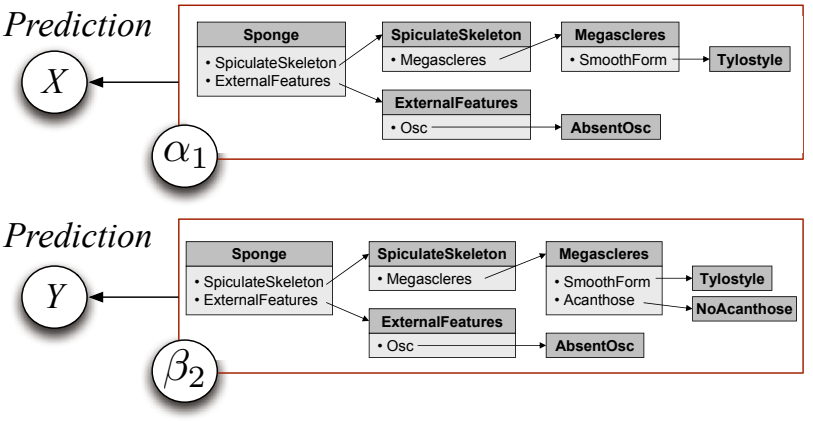


Fig. 2. Relationship between two arguments:β_2 is a counterargument of α_1 because β_2 is a refinement of α_1 and predicts Y that is different from α_1's prediction X

agents, agents argue about predictions for new problems and can provide two kinds of information: a) specific cases $\langle P, S\rangle$, and b) justified predictions: $\langle A, P, S, D\rangle$. Using this information, we can define three types of arguments: justified predictions, counterarguments, and counterexamples.

A *justified prediction* α is generated by an agent A_i to argue that A_i believes that $\alpha.S$ is the correct solution for problem P because of justification $\alpha.D$.

A *counterargument* β is an argument offered in opposition to another argument α. In our framework, a counterargument consists of a justified prediction $\langle A_j, P, S', D'\rangle$ generated by an agent A_j with the intention to rebut an argument α generated by another agent A_i, that endorses a different solution S' with a justification D'.

Figure 2 shows two arguments from our experimental setting in section 6. First notice that each argument is predicting a different solution: α_1 predicts X while β_2 predicts Y. Moreover, α_1 subsumes β_2 (in other words, β_2 is a specialization of α_1), meaning that all problems that satisfy β_2 also satisfy α_1. If the predictions are contradictory ($X \neq Y$) then β_2 is a counterargument of α_1.

A *counterexample* c is a case that contradicts an argument α. Thus, a counterexample is also a counterargument, stating that an argument α is not always true, and the evidence provided is the case c. Specifically, a case c is a counterexample of an argument α if the following conditions hold: $\alpha.D \sqsubseteq c$ and $\alpha.S \neq c.S$, i.e. the case satisfies the justification $\alpha.D$ while determining a solution different to than the predicted by α.

3.4 Argument Generation

In our framework, arguments are generated by the agents from cases, using learning methods. Any learning method able to provide a justified prediction can be used to generate arguments. For instance, decision trees and LID [4] are suitable learning methods. Specifically, in the experiments reported in this paper agents use LID. Thus, when an agent wants to generate an argument endorsing that a specific solution is the correct solution for a problem P, it generates a justified prediction using LID.

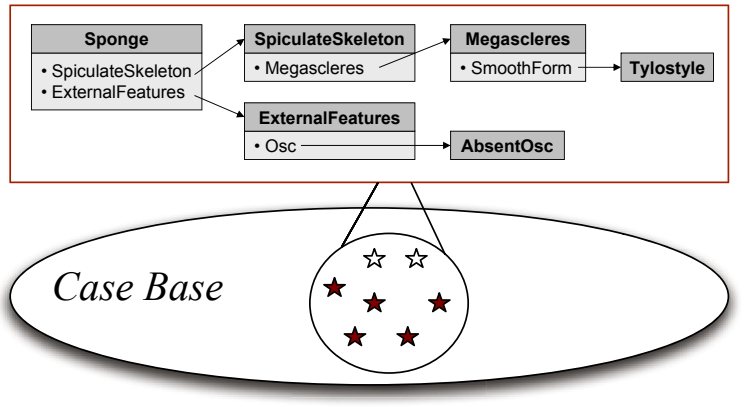


Fig. 3. Relationship between an argument and a case base. Dark stars are cases endorsing the argument while white stars are cases contradicting it.

Agents may try to rebut arguments by generating a counterargument or by finding counterexamples. An agent A_i wants to generate a counterargument β to rebut an argument α when α is in contradiction with the local case base of A_i. Moreover, while generating such a counterargument β, A_i expects that β is preferred over α. For that purpose, agents use a specific policy to generate counterarguments based on the *specificity* criterion [14]. The generation of counterarguments using the specificity criterion puts some requirements on the learning method but techniques LID or ID3 can be easily adapted for this task (as shown in [13]).

For instance, in Figure 2, given an argument α_1 that predicts X asserted by agent A_1 generating a counterargument means that agent A_2 finds a description β_2 such that it is subsumed by α_1 but (according to A_2's experience) predicts a solution $Y \neq X$.

Specifically, in our experiments, when an agent A_i wants to rebut an argument α, uses the following policy: (1) Agent A_i tries to generate a counterargument β more specific than α; if found, β is sent to the other agent as a counterargument of α. If not found, then (2) A_i searches for a counterexample $c \in C_i$ of α. If a case c is found, then c is sent to the other agent as a counterexample of α. If an agent A_i is unable to generate a counterargument or find a counterexample then A_i has no grounds to disagree with argument α and can not rebut that argument.

3.5 Prediction Confidence

We will use a *case-based confidence* measure [13] to determine the degree of confidence of an individual agent in its own argument (justified prediction) and also on the counterarguments received from other agents. The confidence is assessed by the agents via an process of *examination of arguments*. During this examination, an agent will count how many of the cases in its individual case base *endorse* an argument α, and how many cases are counterexamples of α. The more endorsing cases, the higher the confidence; and the more the counterexamples, the lower the confidence.

While examining an argument α, an agent determines the set of cases in its individual case base that are subsumed by $\alpha.D$ (the cases shown as stars in the circle of

Figure 3): the more of these cases that have $\alpha.S$ as solution, the higher the confidence. After examining an argument α, an agent A_i obtains the *aye* and *nay* values: The *aye* value $Y_\alpha^{A_i} = |\{c \in C_i |\ \alpha.D \sqsubseteq c.P \wedge \alpha.S = c.S\}|$ is the number of cases in the agent's case base *subsumed* by the description $\alpha.D$ that has solution $\alpha.S$ proposed by α, while the *nay* value $N_\alpha^{A_i} = |\{c \in C_i |\ \alpha.D \sqsubseteq c.P \wedge \alpha.S \neq c.S\}|$ is the number of cases in the agent's case base *subsumed* by description $\alpha.D$ that *do not* have that solution.

Figure 3 shows an the examination process where, given an argument α, an agent first retrieves all the cases that are subsumed by $\alpha.D$ from the case base, and then counts how many are counterexamples (white stars) or endorsing cases (black stars).

The confidence on an argument α is assessed by an agent A_i as follows:

$$\mathsf{C}_{A_i}(\alpha) = \frac{Y_\alpha^{A_i} + 1}{Y_\alpha^{A_i} + N_\alpha^{A_i} + 2}$$

where the reason for adding 1 to the numerator and 2 to the denominator is akin to the Laplace correction to estimate probabilities.

3.6 Information Exchange Protocol

In this section we will define an information exchange protocol that allows agents in an information market to exchange information with its acquaintances in the social network. Intuitively, an agent will engage into one-to-one argumentation processes with each one of his acquaintances sequentially, trying to improve its prediction at each step. The intuition is that after each discussion, the solution is more likely to be the correct one, since more information has been taken into account to come up with it.

Let us assume that a particular agent A_i wants to generate a prediction for a problem P. Let $F \subseteq \mathcal{A}$ be the set of m acquaintances of A_i. The information exchange protocol initiates a series of argumentation processes between A_i and each of the agents in F in a series of rounds. In the first round $r = 0$, A_i simply generates its individual prediction in the form of an argument α^0. Then, in the next round $r = 1$, A_i will argue with the first agent $A_j \in F$ and refine its prediction into a better one α^1. At the end of round $r = m$, A_i will have a prediction α^m that will be the final one made for the market.

Each one of these argumentation processes in itself consists of a series of cycles. In the initial cycle, each agent states which is its individual prediction for P. Then, at each cycle an agent can try to rebut the prediction made by the other agent. The agents alternate turns in the protocol, and an agent is allowed to send one counterargument or counterexample at its turn. When an agent receives a counterargument or counterexample, it informs the other agent if it accepts the counterargument (and changes its prediction) or not. Moreover, agents have also the opportunity to answer to counterarguments in their turn, by trying to generate a counterargument to the counterargument. At any time the protocol terminates when all the agents agree or when no agent has generated any counterargument during the last two cycles.

During the argumentation protocol, agents can use the following performatives:

- $assert(\alpha)$: the justified prediction held during the next cycle will be α. If multiple asserts are send, only the last one is considered as the currently held prediction.
- $rebut(\beta, \alpha)$: the agent has found a counterargument β to the prediction α.

We will define α_i^t as the prediction that an agent A_i is holding at iteration t of the argumentation protocol, and H_t as the set containing the predictions that each of the two agents hold at a cycle t. The argumentation protocol between an agent A_i, that is currently holding a prediction α_r at a round r of the information exchange protocol, and an acquaintance A_j works as follows:

1. At cycle $t = 0$, the initial argument of A_i will be the one coming from the previous round α_r, thus $\alpha_i^0 = \alpha_r$. The initial argument of A_j will be the result of trying to solve P individually, building a justified prediction using its own CBR method. Then, each agent sends the performatives $assert(\alpha_i^0)$ and $assert(\alpha_j^0)$ respectively to the other agent. Thus, the agents know $H_0 = \langle \alpha_i^0, \alpha_j^0 \rangle$. The turn is given to the first agent A_i.
2. At each cycle t (other than 0), the agents check whether their arguments in H_t agree. If they do, the protocol moves to step 5. If during the last 2 cycles no agent has sent any counterexample or counterargument, the protocol also moves to step 5. Otherwise, the agent A_i who has the turn tries to generate β_i^t (a counterargument or a counterexample) against the argument of the other agent:
   - If β_i^t is a counterargument, then, A_i locally compares α_i^t with β_i^t by assessing their confidence against its individual case base C_i (notice that A_i is comparing its previous argument with the counterargument that A_i itself has just generated and that is about to send to A_j). If $C_{A_i}(\beta_i^t) > C_{A_i}(\alpha_i^t)$, then A_i considers that β_i^t is stronger than its previous argument, changes its argument to β_i^t by sending $assert(\beta_i^t)$ to the rest of the agents (i.e. A_i checks if the new counterargument is a better argument than the one it was previously holding) and $rebut(\beta_i^t, \alpha_j^t)$ to A_j. Otherwise (i.e. $C_{A_i}(\beta_i^t) \leq C_{A_i}(\alpha_i^t)$), A_i will send only $rebut(\beta_i^t, \alpha_j^t)$ to A_j. In any of the two situations the protocol moves to step 3.
   - If β_i^t is a counterexample c, then A_i sends $rebut(c, \alpha_j^t)$ to A_j. The protocol moves to step 4.
   - If A_i cannot generate any counterargument or counterexample, the turn is given to the next agent, a new cycle $t + 1$ starts, and the protocol moves to state 2.
3. The agent A_j that has received the counterargument β_i^t, locally compares it against its own argument, α_j^t, by locally assessing their confidence. If $C_{A_j}(\beta_i^t) > C_{A_j}(\alpha_j^t)$, then A_j will accept the counterargument as stronger than its own argument, and it will send $assert(\beta_i^t)$ to the other agent. Otherwise (i.e. $C_{A_j}(\beta_i^t) \leq C_{A_j}(\alpha_j^t)$), A_j will not accept the counterargument, and will inform the other agent accordingly. Any of the two situations start a new cycle $t+1$, A_i gives the turn to the next agent, and the protocol moves to state 2.
4. The agent A_j that has received the counterexample c retains it into its case base and generates a new argument α_j^{t+1} that takes into account c, and informs the rest of the agents by sending $assert(\alpha_j^{t+1})$ to all of them. Then, A_i gives the turn to the other agent, a new cycle $t + 1$ starts, and the protocol moves to step 2.
5. The argument that A_i is holding is the one that will be carried on to the next round of the information exchange protocol, i.e. when A_i engages in an argumentation with the next agent out of his acquaintances.

Moreover, in order to avoid infinite iterations, if an agent sends twice the same argument or counterargument to the same agent, the message is not considered.

3.7 Bet Generation

At the end of the information exchange protocol, an agent A_i will have a prediction α for a particular solution class. Moreover, in order to participate in a prediction market, the agent has to bet a particular amount of money on its prediction. The more money the agent bets, the bigger the potential reward is, but the bigger the risk. Thus, it is natural for an agent to bet more money when it is more confident that its prediction is correct. For that reason, in our framework, agents bet money proportionally to the confidence (computed as explained in Section 3.5) on their predictions. Since an MPM defines a maximum amount of money M that each agent can bet, each agent will bet $M \times C(\alpha)$, i.e. a proportional amount to its individual confidence. Thus, the bet made by an agent A_i that has a prediction α after the information exchange process will be:

$$B_{A_i} = \langle \alpha.S, M \times C(\alpha) \rangle$$

4 Experimental Evaluation

In this section we will empirically evaluate the performance of prediction markets, comparing it to the performance of normal voting. Moreover we will also study the effect of having different social networks among the agents in the market and how much the quality of data affects the market.

We have made experiments in the sponge data set, a marine sponge identification tasks that contains 280 marine sponges represented in a relational way and pertaining to three different orders of the Demospongiae class. In an experimental run, training cases are distributed among the agents. In the testing stage problems arrive to the market, and each agent will place a bet for the solution they predict is the correct one.

We have performed three sets of experiments. In the first set, we are interested in comparing prediction markets with majority voting, in the second one we want to explore the effect of argumentative information exchange in prediction markets, and finally, the third one explores the effect of varying the quality of the data sample that each agent owns. Each experiment consists of 5 runs of a 5-fold cross validation test. Notice that in step 4 of the argumentation protocol in section 3.6, agents learn from counterexamples coming from other agents. In the experiments we performed, each problem in the test set has to be independent from one another, in order to compute the averages for cross validation. Thus, the learning performed during argumentation is not carried up to the next problem in the test set. We have researched the issue of *learning from communication* in other multiagent scenarios in [12].

4.1 Prediction Markets Versus Majority Voting

For these experiments we evaluated the prediction accuracy of a committee using majority voting consisting of 8 agents with a prediction market consisting of the same 8 agents. The training set is split into 8 parts and each part is sent to an agent. Thus, each agent has an initial case base of about 28 cases.

Agents solving problems using a prediction market didn't do any information exchange for this experiment. The maximum bet was set to $M = 100$, and the incentive

Table 1. Prediction markets accuracy with information exchange along several social networks and with different biases in the individual case bases

social network	*market accuracy*	*individual accuracy*	*average reward*	*majority voting*
0 acquaintances	*89.71%*	*74.21%*	*10.35*	
1 acquaintances	*90.57%*	*83.99%*	*11.42*	
2 acquaintances	*91.29%*	*86.63%*	*12.14*	89.71
3 acquaintances	*91.14%*	*87.64%*	*11.94*	0.20
4 acquaintances	*91.07%*	*88.16%*	*11.85*	0.21

factor was set to $X = 10\%$, thus $c = 1.1$. The results showed that the majority voting achieved a prediction accuracy of 88.93%, while the prediction market achieved an accuracy of 89.71%, a significant improvement. Moreover, agents won an average of 10.35 monetary units per problem solved. In a voting committee, agents are only asked to reveal part of its individual information, namely the preferred alternative for which an individual casts a vote. In a prediction market, however, the amount bet by an individual acts as a "signal" indicating the degree of individual confidence in predicting the preferred alternative as being the correct one. Since the reward is proportional to the bet amount, the agents have an incentive to disclose this additional information.

Since the reward is proportional to the individual prediction confidence, the agents have an incentive to try to improve their individual prediction accuracy and confidence.

4.2 The Effect of Information Exchange

We performed several experiments with different social networks in a prediction market composed of 8 agents. Figure 1 shows some social networks where each agent has 0, 1, 2 or 3 acquaintances; we have performed experiments with 0 to 4 acquaintances and logged the prediction accuracy of the market, the prediction accuracy of each individual agent, and also the average money reward received by each agent per problem.

Table 1 shows that information exchange is positive both for the individual agents and for the market as a whole. We can see that the more acquaintances an agent has, the higher its individual prediction. For instance, agents with 0 acquaintances have an accuracy of 74.21% while agents with 1 acquaintance have an accuracy of 83.99%, and when they have 4 acquaintances, their accuracy is increased to 88.16%. Moreover, the predictive accuracy of the market increases from 89.71% when agents do not perform information exchange, to above 91% when agents have more than 1 acquaintance.

These results also show that the argumentation process of section 3.6 is successful in in acquiring individually valuable information. The increase in individual accuracy and confidence in prediction can only be explained by agents changing their original prediction and confidence value after arguing with other agents.

Another effect we can observe is that the reward that the agents obtain increases when they perform information exchange, starting in 10.35 monetary units per problem when they do not perform information exchange, and going up to close to 12 when agents have 2 acquaintances ore more. It is interesting to notice that the performance of the prediction market doesn't increase linearly with the performance of the individual agents. In fact, the more accurate the individual agents get, the more correlated their

individual predictions are, and thus there is less difference between their individual predictions and the prediction of the market as a whole. This is a well known effect in machine learning (known as the *ensemble effect* [9]), or in economics (related to the Condorcet Jury Theorem). Therefore, if the reward signal that the agents get was only related to its individual accuracy, agents might be interested in their classification accuracy to a point were the correlation is too high, and then the market would not achieve it's optimal accuracy. The reward signal presented in Section 2 takes this into account, and rewards the agents when the market as a whole has high accuracy.

Moreover, Table 1 shows that the reward signal is higher when the market accuracy is higher (in our experiments, when agents have 2 acquaintances), instead of when their individual accuracy is higher. Therefore, the agents have an incentive to be highly accurate, but up to a limit, so that the market as a whole has a high accuracy. In our experiments, the agents receive maximum reward when they collaborate with two acquaintances, and thus it is rational for the agents to do so. As a side effect, the accuracy of the market as a whole is also maximum under those conditions, thus the agents have an incentive to do what is better for the market.

Summarizing, the experiments show that prediction markets can provide incentives for agents to disclose more information, and that information improves the accuracy of joint predictions or group judgments. The MPM is based on disclosing further information interpreted as a bet amount that represents the individual confidence on a prediction. The results also show that the case-based confidence function defined in Section 3.5 provides a good estimation, since the prediction market improves the accuracy.

Concerning information exchange, the experiments show that individual and market accuracy improve. This means that the agents make a more informed prediction, and thus that the argumentation protocol of Section 3.6 is effective in providing agents with enough information to correct previously inaccurate predictions.

4.3 Quality of the Data Sample

The results in the previous section assume that each agent have a good sample of data, i.e. that each agent is competent. We performed a set of experiments where we changed the quality of the data sample that each agent has and evaluated how this affects the performance of the market, as well as the individual agents.

Specifically, we performed experiments where agent have *biased* case bases. A biased case base is one that is not a good sample of the complete data set. The bias of a case base C_i with respect to a data set C is defined by:

$$\mathbb{B}(C_i) = \sqrt{\sum_{k=1\ldots K} \left(\frac{\#(\{c \in C_i | c.S = S_k\})}{\#(C_i)} - \frac{\#(\{c \in C | c.S = S_k\})}{\#(C)} \right)^2}$$

Notice that Case Base Bias is zero when the ratio of cases for each solution class is the same in the case base C_i than in the data set C. The higher the bias, the worst the sample.

Table 2. Prediction markets accuracy with information exchange along several social networks and with different biases in the individual case bases

bias	*social network*	*market accuracy*	*individual accuracy*	*average reward*	*majority voting*
0.2	*0 acquaintances*	*90.14%*	*73.01%*	*10.44*	89.00
	1 acquaintances	*89.86%*	*82.79%*	*10.29*	
	2 acquaintances	*90.07%*	*85.80%*	*10.53*	
	3 acquaintances	*91.21%*	*87.16%*	*11.54*	
	4 acquaintances	*91.36%*	*87.49%*	*11.79*	
0.4	*0 acquaintances*	*88.71%*	*66.43%*	*8.86*	84.86
	1 acquaintances	*87.79%*	*75.95%*	*7.43*	
	2 acquaintances	*89.43%*	*79.56%*	*8.63*	
	3 acquaintances	*90.57%*	*81.7%*	*9.59*	
	4 acquaintances	*90.79%*	*82.16%*	*9.84*	
0.6	*0 acquaintances*	*86.29%*	*58.05%*	*6.7*	83.29
	1 acquaintances	*88.00%*	*70.00%*	*6.75*	
	2 acquaintances	*90.14%*	*74.94%*	*8.05*	
	3 acquaintances	*89.36%*	*74.48%*	*7.33*	
	4 acquaintances	*89.21%*	*75.51%*	*7.21*	
0.6	*0 acquaintances*	*49.71%*	*38.63%*	*-32.36*	55.57
	1 acquaintances	*48.00%*	*41.26%*	*-23.64*	
	2 acquaintances	*61.43%*	*47.52%*	*-16.55*	
	3 acquaintances	*67.43%*	*55.67%*	*-0.02*	
	4 acquaintances	*66.93%*	*55.44%*	*-0.53*	

We performed experiments giving agents case bases with bias 0.2, 0.4, 0.6, and 0.8 (results in Table 1 correspond to bias equal to 0.0). Table 2 shows the accuracy of the market, of the individual agents, and also of majority voting. If we look at the accuracy of majority voting, we can see that it degrades when the bias increases, since the individual agents' predictions also degrades. For instance, the classification accuracy of majority voting degrades from 88.93% when the bias is 0.0 to 83.29% when the bias is 0.6 or to 55.57% when the bias is 0.8 (0.8 is a really large bias in our sponge data set, and each agent almost only knows cases of a single class).

Concerning market's accuracy, increasing the bias also diminishes its accuracy, but remarkably much less than for majority voting; in fact market's accuracy is strongly affected only when the bias is very high. For instance, a prediction market where agents have 4 acquaintances with bias 0.6 has still an accuracy of 89.21% (compared to 83.29% of majority voting). The accuracy of individual agents is much more affected by bias, being reduced from 74.21% with no bias to 58.05% with bias 0.6 when they have no acquaintances. When the bias is even larger (0.8), their accuracy goes down drastically to 38.63%. As the number of acquaintances increases, the individual accuracy largely increases, showing that the argumentation framework allows agents to efficiently exchange information and benefit from information in the case bases of acquaintances. For instance, the individual accuracy with bias 0.6 recovers from 58.05% with no acquaintances to 75.51% with 4 acquaintances. This effect of argumentation is thus responsible for the increase of the market's accuracy with bias 0.6 recovers from 86.29% with no

acquaintances to 89.21% with 4 acquaintances. Even in the extreme case with 0.8 bias, argumentation is able to increase the accuracy of individual agents. Therefore, we can conclude that an argumentation-based process of information exchange is very useful in conditions where individuals do not have a perfect (or very good) sample of data in which to base its decisions. Argumentation allows each individual agent to contrast its empirically-based judgements with those of its peers and acquire new information that, albeit partially, help it recover a better sample of data.

An interesting effect is that when the bias is extreme, the confidence the agents computes is not accurate: this is the only scenario where the market has lower accuracy than simple majority voting and where the agents obtain a negative reward. However, by means of information exchange (having acquaintances) even in this extreme scenario the quality of the individual solutions increases, and thus the market accuracy also increases, from 48.0% accuracy with no information exchange to 67.43% and 66.93% when agents have 3 and 4 acquaintances respectively.

Thus, in summary, we see that by means of information exchange using an argumentation process, agents become much more robust to biased data than standard voting mechanisms, by leveraging available information in the acquaintances' case bases. Agents with biased case bases benefit from exchanging information (by means of argumentation processes) with other agents, and the result of argumentation among agents with different biases is a less biased prediction that can largely overcome the effect of bias (except in the extreme case of 0.8 bias, where we see an improvement, but not up to the levels achieved when there is no bias). However, notice that in our experiments, the cases learnt from other agents during argumentation are not stored (for experimentation purposes), it is part of our future work to explore how retention of cases can help agents to further overcome the effect of bias (continuing the work started in previous work [12]).

5 Related Work

Research on *prediction markets* has been focused on exploiting human knowledge [17], and to our knowledge they have not been used in multiagent systems. Research in MAS is generally focused on negotiation processes and much less on social choice, in the sense of modeling and implementing processes where a group of agents achieve a joint judgment. As argued in [8], computational approaches to social choice can benefit both social choice studies and AI. Impossibility theorems proved in theoretical approaches to social choice do not prevent the design of reasonably fair and robust mechanisms [3].

Other approaches in social choice (different from prediction markets) have been applied to MAS. What we have been calling statistical means approaches (that includes voting) have been applied to MAS, from simple voting to complex schemes such as voting for combinatorial domains [10]. Deliberative approaches to group judgment have also been studied, for instance in [13] a committee of agents argue the pros and cons of a group judgment. Market mechanisms have been applied to resource allocation [15] or other types of market goods. Our focus here is rather different: developing an agent-based information or prediction market for group judgment.

Concerning on argumentation in MAS, previous work focuses on several issues like a) logics, protocols and languages that support argumentation, b) argument selection

and c) argument interpretation. Approaches for logic and languages that support argumentation include defeasible logic [7] and BDI models [16]. An overview of logical models of reasoning can be found at [6]. Moreover, the most related area of research is case-based argumentation. Combining cases and generalizations for argumentation has been already used in the HYPO system [5], where an argument can contain both specific cases or generalizations. Moreover, generalization in HYPO was limited to selecting a set of predefined dimensions in the system while our framework presents a more flexible way of providing generalizations. Furthermore, HYPO was designed to provide arguments to human users, while we focus on agent to agent argumentation. Case-based argumentation has also been implemented in the CATO system[2], that models ways in which experts compare and contrast cases to generate multi-case arguments to be presented to law students.

6 Conclusions

Mechanisms for group judgment (voting, deliberation, etc) are ubiquitous in human societies. However, in addition to the formal structure of the group judgment mechanism, the informal structure play an important role [17]. We have considered here the effect of an informal structure (social networks used to exchange information mediated by argumentation) in a formal group judgment mechanism (MPM). We have shown that these social networks maybe individually useful for artificial agents, since agents may use argumentation to improve their information about the world. Therefore, artificial multiagent systems will also have to deal with the interplay of informal structures together with formal group judgment mechanism.

We have taken a typical task of prediction form a Machine Learning data set and we had goal of developing a simple market called MPM. The basic idea of MPM is that learning agents can use data concerning a *prediction task domain* to predict new unknown problems and, moreover, use the learnt data to implement a confidence estimate of their own predictions. Then, the prediction market design has to be set up to encourage the expression of the agents confidence as a "price signal". Clearly, this is a quite general approach, and different variations can be explored in future work: improving the confidence estimation functions, modifying the market reward scheme or using other machine learning techniques.

We also introduced a process of deliberation based on an argumentation protocol inside the framework of prediction markets. The reason is twofold: first, we wanted to model the idea that people often consult trusted people before making a decision (i.e. they not only learn from experience, but also from communication). Second, current state of the art in multiagent learning suggests that the individual accuracy and confidence increases after a deliberative process [13]. The experiments shown that this is the case: information exchange supported by an argumentation process increases individual accuracy and confidence. As expected, the information exchange also increases the error correlation among agents [12], decreasing the so-called "ensemble effect" that increases joint accuracy over the individual accuracy. The conclusion thus is that information exchange is beneficial to a certain extent, i.e. among a small number of individuals compared to the total number of participating individuals, in such a way that

individual performance is rather increased but error correlation is not much increased. We have also shown that information exchange helps agents with biased views of the problem to overcome their bias and produce more accurate predictions.

Although we presented the results for one data set, any other classification machine learning data set could be used. Current state of the art in multiagent learning suggests that the only difference would be on the degree in which the prediction market surpasses voting [11,13].

As part of our future work, we plan to explore how our techniques will extend to fully open multi-agent systems, where there are several different problems that agents must solve, and not all agents ar competent in all of them, agents use heterogeneous learning mechanisms, and not all agents are trustable. So, agents will have to learn which agents are trustable and which ones are not, and the argumentation process has to be generalized to support heterogeneous learning methods. Our final goal is to define a framework for learning agents with problem solving, learning, collaboration and argumentation capabilities ready to be deployed and be autonomous in an open multi-agent system for real-life application.

Acknowledgements. Support for this work came from projects MID-CBR TIN2006-15140-C03-01, and Agreement Technologies CSD2007-0022.

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. Artificial Intelligence Communications 7(1), 39–59 (1994)
2. Aleven, V.: Teaching Case-Based Argumentation Through a Model and Examples. PhD thesis, University of Pittsburgh (1997)
3. List, C., Pettit, P.: Aggregating sets of judgments: Two impossibility results compared. Synthese 140, 207–235 (2004)
4. Armengol, E., Plaza, E.: Lazy induction of descriptions for relational case-based learning. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS, vol. 2167, pp. 13–24. Springer, Heidelberg (2001)
5. Ashley, K.: Reasoning with cases and hypotheticals in hypo. International Journal of Man-Machine Studies 34, 753–796 (1991)
6. Chesñevar, C.I., Mguitman, A., Loui, R.: Logical models or argument. Computing Surveys 32(4), 336–383 (2000)
7. Chesñevar, C.I., Simari, G.R.: Formalizing Defeasible Argumentation using Labelled Deductive Systems. Journal of Computer Science & Technology 1(4), 18–33 (2000)
8. Chevaleyre, Y., Endriss, U., Lang, J., Maudet, N.: A short introduction to computational social choice. In: van Leeuwen, J., Italiano, G.F., van der Hoek, W., Meinel, C., Sack, H., Plášil, F. (eds.) SOFSEM 2007. LNCS, vol. 4362, pp. 51–69. Springer, Heidelberg (2007)
9. Dietterich, T.G.: Ensemble methods in machine learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000)
10. Lang, J.: Logical preference representation and combinatorial vote. Annals of Mathematics and Artificial Intelligence 42, 37–71 (2004)
11. Ontañón, S., Plaza, E.: Justification-based multiagent learning. In: ICML 2003, pp. 576–583. Morgan Kaufmann, San Francisco (2003)
12. Ontañón, S., Plaza, E.: Case-based learning from proactive communication. In: Proc. 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), pp. 999–1004. IJCAI Press (2007)

13. Ontañón, S., Plaza, E.: Learning and joint deliberation through argumentation in multi-agent systems. In: Proc. AAMAS 2007, pp. 971–978. ACM, New York (2007)
14. Poole, D.: On the comparison of theories: Preferring the most specific explanation. In: IJCAI 1985, pp. 144–147 (1985)
15. Rodriguez-Aguilar, J.A., Sousa, P.: Issues in multiagent resource allocation. Informatica 30, 3–31 (2006)
16. Jennings, N.R., Parsons, S., Sierra, C.: Agents that reason and negotiate by arguing. Journal of Logic and Computation 8, 261–292 (1998)
17. Sunstein, C.R.: Group judgments: Deliberation, statistical means, and information markets. New York University Law Review 80, 962–1049 (2005)

An Argumentative Approach for Modelling Coalitions Using ATL*

Nils Bulling[1], Carlos I. Chesñevar[2,3], and Jürgen Dix[1]

[1] Department of Informatics, Clausthal University of Technology, Germany
{bulling,dix}@in.tu-clausthal.de
[2] CONICET (National Council of Scientific and Technical Research), Argentina
[3] Department of Computer Science and Engineering, Universidad Nacional del Sur, Bahía Blanca, Argentina
cic@cs.uns.edu.ar

Abstract. During the last decade argumentation has evolved as a successful approach to formalize commonsense reasoning and decision making in multiagent systems. In particular, recent research has shown that argumentation can be used to provide a framework for reasoning about *coalition formation*, formalizing the adoption of coalitions by the agents in association with different argumentation semantics. At the same time *Alternating-time Temporal Logic* (ATL for short) has been successfully used to reason about the behavior and abilities of coalitions of agents. However, an important limitation of ATL operators is that they account only for the *existence of successful strategies* of coalitions, not considering whether coalitions can be actually formed.

This paper is an attempt to combine both frameworks in order to develop a logical system through which we can reason at the same time (1) about abilities of coalitions of agents and (2) about the formation of coalitions. In order to achieve this, we provide a formal extension of ATL, called *Coalitional* ATL (CoalATL for short), in which the actual computation of the coalition is modelled in terms of argumentation semantics. Moreover, we integrate goals as agents' incentive to join coalitions. We show that CoalATL's proof theory can be understood as a natural extension of the model checking procedure used in ATL.

1 Introduction and Motivations

During the last decade, argumentation frameworks [23,11] have evolved as a successful approach to formalize commonsense reasoning and decision making in multiagent systems (MAS). Application areas include issues such as joint deliberation, persuasion, negotiation, knowledge distribution and conflict resolution (e.g. [27,24,25,6,21]), among many others. In particular, recent research by Leila

* We thank the editors for the invitation to publish this paper in the ArgMas proceedings. This paper is mostly based on the paper *"Modelling coalitions: ATL + Argumentation"* [7], including also some results presented in *"A Finer Grained Modelling of Rational Coalitions Using Goals"* [8].

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 197–216, 2009.

Amgoud [3,4] has shown that argumentation provides a sound setting to model *reasoning about coalition formation* in MAS. The approach is based on using conflict and preference relationships among coalitions to determine which coalitions should be adopted by the agents. This is done according to a particular argumentation semantics, which can then be computed using a suitable proof theory.

Alternating-time Temporal Logic (ATL) [2] is a temporal logic which can be used for reasoning about the behavior and abilities of agents under various rationality assumptions [17,18,9]. In ATL the key construct has the form $\langle\langle A\rangle\rangle\phi$, which expresses that a coalition A of agents can *enforce* the formula ϕ. Under a model theoretic viewpoint, $\langle\langle A\rangle\rangle\phi$ holds whenever the agents in A have a winning strategy for ensuring the satisfiability of ϕ (independently of the behavior of A's opponents). However, this operator accounts only for the *theoretical existence* of such a strategy, not taking into account whether the coalition A can be actually formed. Indeed, in order to join a coalition, agents usually require some kind of *incentive* (e.g. sharing common goals, getting rewards, etc.), since usually forming a coalition does not come for free (fees have to be paid, communication costs may occur, etc.). Consequently, several possible coalition structures among agents may arise, from which the best ones should be adopted according to some rationally justifiable procedure.

In this paper we present an argumentative approach to extend ATL for modelling coalitions. We provide a formal extension of ATL, CoalATL, by including a new construct $\langle\!|A|\!\rangle\phi$ which denotes that *the group A of agents is able to build a coalition B, $A \cap B \neq \emptyset$, such that B can enforce ϕ*. That is, it is assumed that agents in A work together and try to form a coalition B. The actual computation of the coalition is modelled in terms of a given argumentation semantics [13] in the context of coalition formation [3]. In a second step, we enrich CoalATL with goals. We address the question *why* agents should cooperate. Goals refer to agents' subjective incentive to join coalitions. We show that the proof theory for modelling coalitions in our framework can be embedded as a natural extension of the model checking procedure used in ATL.

The rest of the paper is structured as follows. Section 2 summarizes the main concepts of alternating-time temporal logic (ATL). In Section 3 we introduce the notion of *coalitional framework* [3] as well as some fundamental concepts from argumentation theory. Section 4 provides an argumentation-based view of coalition formation by merging ATL and the coalitional framework introduced in Sections 2 and 3. In Section 5 we incorporate goals to CoalATL, turning to model checking in Section 6. Finally, Sections 7 and 8 discuss related and future work and present the main conclusions obtained.

About this Paper. As already mentioned, this invited paper is based on [7] and [8]. Sections 2-4 and 6-8 are slight variations of [7]. However, some minor conceptual issues have been modified (in accordance with [8]) or added; discussions on future and related work has also been updated. Section 5 presents the work given in [8].

2 ATL

Alternating-time Temporal Logic (ATL) [2] enables reasoning about temporal properties and strategic abilities of agents. The language of ATL is defined as follows.

Definition 1 ($\mathcal{L}_{ATL}$ [2]). *Let $\mathbb{A}\text{gt} = \{a_1, \ldots, a_k\}$ be a nonempty finite set of all agents, and Π be a set of propositions (with typical element p). We denote by "a" a typical agent, and by "A" a typical group of agents from $\mathbb{A}\text{gt}$. $\mathcal{L}_{ATL}(\mathbb{A}\text{gt}, \Pi)$ is defined by the following grammar: $\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A\rangle\rangle \bigcirc\varphi \mid \langle\langle A\rangle\rangle \Box\, \varphi \mid \langle\langle A\rangle\rangle \varphi\, \mathcal{U}\, \varphi$.*

Informally, $\langle\langle A\rangle\rangle \varphi$ expresses that agents A have a *collective strategy to enforce* φ. ATL formulae include the usual temporal operators: $\bigcirc$ ("in the next state"), $\Box$ ("always from now on") and $\mathcal{U}$ (strict "until"). Additionally, $\Diamond$ ("now or sometime in the future") can be defined as $\Diamond\, \varphi \equiv \top\, \mathcal{U}\, \varphi$.

The semantics of ATL is defined by *concurrent game structures.*

Definition 2 (CGS [2]). *A* concurrent game structure (CGS) *is a tuple $\mathcal{M} = \langle \mathbb{A}\text{gt}, Q, \Pi, \pi, Act, d, o\rangle$, consisting of: a set $\mathbb{A}\text{gt} = \{a_1, \ldots, a_k\}$ of* agents*; set Q of* states*; set Π of* atomic propositions*;* valuation of propositions $\pi : Q \to \mathcal{P}(\Pi)$*; set Act of* actions*. Function $d : \mathbb{A}\text{gt} \times Q \to \mathcal{P}(Act)$ indicates the actions available to agent $a \in \mathbb{A}\text{gt}$ in state $q \in Q$. We often write $d_a(q)$ instead of $d(a, q)$, and use $d(q)$ to denote the set $d_{a_1}(q) \times \cdots \times d_{a_k}(q)$ of* action profiles *in state q. Finally, o is a* transition function *which maps each state $q \in Q$ and action profile $\boldsymbol{\alpha} = \langle \alpha_1, \ldots, \alpha_k\rangle \in d(q)$ to another state $q' = o(q, \boldsymbol{\alpha})$.*

A *path* $\lambda = q_0 q_1 \cdots \in Q^\omega$ is an infinite sequence of states such that there is a transition between each q_i, q_{i+1}.We define $\lambda[i] = q_i$ to denote the i-th state of λ. The set of all paths starting in q is defined by $\Lambda_{\mathcal{M}}(q)$.

A (memoryless) *strategy* of agent a is a function $s_a : Q \to Act$ such that $s_a(q) \in d_a(q)$ We denote the set of such functions by Σ_a. A *collective strategy* s_A for team $A \subseteq \mathbb{A}\text{gt}$ specifies an individual strategy for each agent $a \in A$; the set of A's collective strategies is given by $\Sigma_A = \prod_{a \in A} \Sigma_a$ and $\Sigma := \Sigma_{\mathbb{A}\text{gt}}$.

The *outcome* of strategy s_A in state q is defined as the set of all paths that may result from executing s_A: $out(q, s_A) = \{\lambda \in \Lambda_{\mathcal{M}}(q) \mid \forall i \in \mathbb{N}_0\ \exists \boldsymbol{\alpha} = \langle \alpha_1, \ldots, \alpha_k\rangle \in d(\lambda[i])\ \forall a \in A\ (\alpha_a = s_A^a(\lambda[i]) \wedge o(\lambda[i], \boldsymbol{\alpha}) = \lambda[i+1])\}$, where s_A^a denotes agent a's part of the collective strategy s_A.

The semantics of ATL is as follows.

Definition 3 (ATL Semantics). *Let a* CGS *$\mathcal{M} = \langle \mathbb{A}\text{gt}, Q, \Pi, \pi, Act, d, o\rangle$ and $q \in Q$ be given. The semantics of state formulae is given by a satisfaction relation $\models$ as follows:*

$\mathcal{M}, q \models p$ *iff* $p \in \pi(q)$
$\mathcal{M}, q \models \neg\varphi$ *iff* $\mathcal{M}, q \not\models \varphi$
$\mathcal{M}, q \models \varphi \wedge \psi$ *iff* $\mathcal{M}, q \models \varphi$ *and* $\mathcal{M}, q \models \psi$
$\mathcal{M}, \lambda \models \varphi$ *iff* $\mathcal{M}, \lambda[0] \models \varphi$*;*

and for path formulae by

$\mathcal{M}, \lambda \models \bigcirc \varphi$ *iff* $\mathcal{M}, \lambda[1] \models \varphi$
$\mathcal{M}, \lambda \models \Box\, \varphi$ *iff* $\mathcal{M}, \lambda[i] \models \varphi$ *for all* $i \in \mathbb{N}_0$
$\mathcal{M}, \lambda \models \varphi \mathcal{U}\, \psi$ *iff there is an* $i \in \mathbb{N}_0$ *with* $\mathcal{M}, \lambda[i] \models \psi$, *and* $\mathcal{M}, \lambda[j] \models \varphi$ *for all* $0 \leq j < i$.

3 Coalitions and Argumentation

In this section we provide an argument-based characterization of coalition formation that will be used later to extend ATL. We follow an approach similar to [3], where an argumentation framework for generating coalition structures is defined. Our approach is a generalization of the framework of Dung for argumentation [13], extended with a *preference relation*. The basic notion is that of a *coalitional framework*, which contains a set of elements $\mathfrak{C}$ (usually seen as agents or coalitions), an attack relation (for modelling conflicts among elements of $\mathfrak{C}$), and a preference relation between elements of $\mathfrak{C}$ (to describe favorite agents/coalitions).

Definition 4 (Coalitional framework [3]). *A* coalitional framework *is a triple* $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ *where* $\mathfrak{C}$ *is a non-empty set of elements,* $\mathcal{A} \subseteq \mathfrak{C} \times \mathfrak{C}$ *is an attack relation, and* $\prec$ *is a preorder on* $\mathfrak{C}$ *representing preferences on elements in* $\mathfrak{C}$.

Let S *be a non-empty set of elements.* $\mathbb{CF}(S)$ *denotes the set of all coalitional frameworks where elements are taken from the set* S, *i.e. for each* $(\mathfrak{C}, \mathcal{A}, \prec) \in \mathbb{CF}(S)$ *we have that* $\mathfrak{C} \subseteq S$.

The set $\mathfrak{C}$ in Definition 4 is intentionally generic, accounting for various possible alternatives. One alternative is to consider $\mathfrak{C}$ as a set of agents $\mathbb{A}\mathrm{gt} = \{a_1, \ldots, a_k\}$: $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec) \in \mathbb{CF}(\mathbb{A}\mathrm{gt})$. Then, a *coalition* is given by $C = \{a_{i_1}, \ldots, a_{i_l}\} \subseteq \mathfrak{C}$ and "agent" can be used as an intuitive reference to elements of $\mathfrak{C}$. Another alternative is to use a coalitional framework $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ based on $\mathbb{CF}(\mathcal{P}(\mathbb{A}\mathrm{gt}))$. Now elements of $\mathfrak{C} \subseteq \mathcal{P}(\mathbb{A}\mathrm{gt})$ are *groups* or *coalitions* (where we consider singletons as groups too) of agents. Under this interpretation a coalition $C \subseteq \mathfrak{C}$ is a *set of sets* of agents. Although "coalition" is already used for $C \subseteq \mathfrak{C}$, we also use the intuitive reading "coalition" or "group" to address elements in $\mathfrak{C}$.[1] Yet another way is not to use the specific structure for elements in $\mathfrak{C}$, assuming it just consists of abstract elements, e.g. c_1, c_2, etc. One may think of these elements as individual agents or coalitions. This approach is followed in [3].

In the rest of this paper we mainly follow the first alternative when informally speaking about coalitional frameworks, i.e. we consider $\mathfrak{C}$ as a set of agents.

Example 1. Consider the following two coalitional frameworks: (i) $\mathcal{CF}_1 = (\mathfrak{C}, \mathcal{A}, \prec)$ where $\mathfrak{C} = \{a_1, a_2, a_3\}$, $\mathcal{A} = \{(a_3, a_2), (a_2, a_1), (a_1, a_3)\}$ and agent a_3

[1] The first interpretation is a special case of the second (coalitional frameworks are members $\mathbb{CF}(\mathcal{P}(\mathbb{A}\mathrm{gt}))$).

Fig. 1. Figure (a) (resp. (b)) corresponds to the coalitional frameworks defined in Example 1 (resp. 3 (b)). Nodes represent agents and arrows between nodes stand for the attack relation.

is preferred over a_1, i.e. $a_1 \prec a_3$; and (ii) $\mathcal{CF}_2 = (\mathfrak{C}', \mathcal{A}', \prec')$ where $\mathfrak{C}' = \{\{a_1\}, \{a_2\}, \{a_3\}\}, \mathcal{A}' = \{(\{a_3\}, \{a_2\}), (\{a_2\}, \{a_1\}), (\{a_1\}, \{a_3\})\}$ and group $\{a_3\}$ is preferred over $\{a_1\}$, i.e. $\{a_1\} \prec' \{a_3\}$. They capture the same scenario and are isomorphic but $\mathcal{CF}_1 \in \mathbb{CF}(\{a_1, a_2, a_3\})$ and $\mathcal{CF}_2 \in \mathbb{CF}(\mathcal{P}(\{a_1, a_2, a_3\}))$; that is, the first framework is defined regarding single agents and the latter over (trivial) coalitions. Figure 1 (a) shows a graphical representation of the first coalitional framework.

Let $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ be a coalitional framework. For $C, C' \in \mathfrak{C}$, we say that C *attacks* C' iff $C\mathcal{A}C'$. The attack relation represents conflicts between elements of $\mathfrak{C}$; for instance, two agents may rely on the same (unique) resource or they may have disagreeing goals, which prevent them from cooperation. However, the notion of attack may not be sufficient for modelling conflicts, as some elements (resp. coalitions) in $\mathfrak{C}$ may be preferred over others. This leads to the notion of *defeater* which combines the notions of attack and preference.

Definition 5 (Defeater). *Let $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ be a coalitional framework and let $C, C' \in \mathfrak{C}$. We say that C* defeats *C' if, and only if, C attacks C' and C' is not preferred over C (i.e., not $C \prec C'$). We also say that C is a* defeater *for C'.*

Attacks and defeats are defined between *single* elements of $\mathfrak{C}$. As we are interested in the formation of coalitions it is reasonable to consider conflicts between coalitions. Members in a coalition may prevent attacks to members in the same coalition; they protect each other. The concept of defence, introduced next, captures this idea of mutual protection.

Definition 6 (Defence). *Let $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ be a coalitional framework and $C, C' \in \mathfrak{C}$. We say that C'* defends itself against *C if, and only if, C' is preferred over C, i.e., $C \prec C'$, and C'* defends itself *if it defends itself against any of its attackers. Furthermore, C is* defended by *a set $\mathfrak{S} \subseteq \mathfrak{C}$ of elements of $\mathfrak{C}$ if, and only if, for all C' defeating C there is a coalition $C'' \in \mathfrak{S}$ defeating C'.*

In other words, if an element C' defends itself against C then C may attack C' but C is not allowed to defeat C'.

A minimal requirement one should impose on a coalition is that its members do not defeat each other; otherwise, the coalition may be unstable and break up sooner or later because of conflicts among its members. This is formalized in the next definition.

Definition 7 (Conflict-free). *Let $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ be a coalitional framework and $\mathfrak{S} \subseteq \mathfrak{C}$ a set of elements in $\mathfrak{C}$. Then, $\mathfrak{S}$ is called* conflict-free *if, and only if, there is* no *$C \in \mathfrak{S}$ defeating some member of $\mathfrak{S}$.*

It must be remarked that our notions of "defence" and "conflict-free" are defined in terms of "defeat" rather than "attack".[2] Given a coalitional framework $\mathcal{CF}$ we will use argumentation to compute coalitions with desirable properties. In argumentation theory many different semantics have been proposed to define ultimately accepted arguments [13,10]. We apply this rich framework to provide different ways to coalition formation. A semantics can be defined as follows.

Definition 8 (Coalitional framework semantics). *A* semantics *for a coalitional framework $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ is a (isomorphism invariant) mapping $\mathfrak{sem}$ which assigns to a given coalitional framework $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ a set of subsets of $\mathfrak{C}$, i.e., $\mathfrak{sem}(\mathcal{CF}) \subseteq \mathcal{P}(\mathfrak{C})$.*

Let $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ be a coalitional framework. To formally characterize different semantics we will define a function $\mathcal{F}_{\mathcal{CF}} : \mathcal{P}(\mathfrak{C}) \to \mathcal{P}(\mathfrak{C})$ which assigns to a set of coalitions $\mathfrak{S} \in \mathcal{P}(\mathfrak{C})$ the coalitions defended by $\mathfrak{S}$.

Definition 9 (Characteristic function $\mathcal{F}$). *Let $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ be a coalitional framework and $\mathfrak{S} \subseteq \mathfrak{C}$. The function $\mathcal{F}$ defined by*

$$\mathcal{F}_{\mathcal{CF}} : \mathcal{P}(\mathfrak{C}) \to \mathcal{P}(\mathfrak{C})$$
$$\mathcal{F}_{\mathcal{CF}}(\mathfrak{S}) = \{C \in \mathfrak{C} \mid C \text{ is defended by } \mathfrak{S}\}$$

is called characteristic function.[3]

$\mathcal{F}$ can be applied recursively to coalitions resulting in new coalitions. For example, $\mathcal{F}(\emptyset)$ provides all undefeated coalitions and $\mathcal{F}^2(\emptyset)$ constitutes the set of all elements of $\mathfrak{C}$ which members are undefeated *or* are defended by undefeated coalitions.

Example 2. Consider again the coalitional framework $\mathcal{CF}_1$ given in Example 1. The characteristic function applied on the empty set results in $\{a_3\}$ since the agent is undefeated, $\mathcal{F}(\emptyset) = \{a_3\}$. Applying $\mathcal{F}$ on $\mathcal{F}(\emptyset)$ determines the set $\{a_1, a_3\}$ because a_1 is defended by a_3. It is easy to see that $\{a_1, a_3\}$ is a fixed point of $\mathcal{F}$.

We now introduce the first concrete semantics called coalition structure semantics, which was originally defined in [3].

Definition 10 (Coalition structure $\mathfrak{sem}_{\mathbf{cs}}$ [3]). *Let $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ be a coalitional framework. Then*

$$\mathfrak{sem}_{cs}(\mathcal{CF}) := \left\{ \bigcup_{i=1}^{\infty} \mathcal{F}^i_{\mathcal{CF}}(\emptyset) \right\}$$

is called coalition structure semantics *or just* coalition structure *for $\mathcal{CF}$.*

[2] In [3,4] these notions are defined the other way around, resulting in a different characterization of stable semantics.

[3] We omit the subscript $\mathcal{CF}$ if it is clear from context.

For a coalitional framework $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ with a finite set $\mathfrak{C}$[4] the characteristic function $\mathcal{F}$ is continuous [13, Lemma 28]. Since $\mathcal{F}$ is also monotonic it has a least fixed point given by $\mathcal{F}(\emptyset)\uparrow^{\omega}$ (according to Knaster-Tarski). We have the following straightforward properties of coalition structure semantics.

Proposition 1 (Coalition structure). *Let $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ be a coalitional framework with a finite set $\mathfrak{C}$. There is always a unique coalition structure for $\mathcal{CF}$. Furthermore, if no element of $C \in \mathfrak{C}$ defends itself then the coalitional structure is empty, i.e. $\mathfrak{sem}_{cs}(\mathcal{CF}) = \{\emptyset\}$.*

Example 3. The following situations illustrate the notion of coalitional structure:

(a) Consider Example 2. Since $\{a_1, a_3\}$ is a fixed point of $\mathcal{F}_{\mathcal{CF}_1}$ the coalitional framework $\mathcal{CF}_1$ has $\{a_1, a_3\}$ as coalitional structure.
(b) $\mathcal{CF}_3 := (\mathfrak{C}, \mathcal{A}, \prec) \in \mathbb{CF}(\{a_1, a_2, a_3\})$ (shown in Figure 1(b)), is a coalitional framework with $\mathfrak{C} = \{a_1, a_2, a_3\}$, $\mathcal{A} = \{\ (a_1, a_2),\ (a_1, a_3)\ ,\ (a_2, a_1),\ (a_2, a_3),\ (a_3, a_1)\}$ and a_3 is preferred over a_2, $a_2 \prec a_3$, has the empty coalition as associated coalition str., i.e. $\mathfrak{sem}_{\mathrm{cs}}(\mathcal{CF}) = \{\emptyset\}$.

Since the coalition structure is often very restrictive, it seems reasonable to introduce other less restrictive semantics. Each of the following semantics are well-known in argumentation theory [13] and can be used as a criterion for coalition formation (cf. [3]).

Definition 11 (Argumentation Semantics). *Let $(\mathfrak{C}, \mathcal{A}, \prec)$ be a coalitional framework, $\mathfrak{S} \subseteq \mathfrak{C}$ a set of elements of $\mathfrak{C}$. $\mathfrak{S}$ is called*

(a) admissible extension *iff $\mathfrak{S}$ is conflict-free and $\mathfrak{S}$ defends all its elements, i.e. $\mathfrak{S} \subseteq \mathcal{F}(\mathfrak{S})$.*
(b) complete extension *iff $\mathfrak{S}$ is conflict-free and $\mathfrak{S} = \mathcal{F}(\mathfrak{S})$.*
(c) grounded extension *iff $\mathfrak{S}$ is the smallest (wrt. to set inclusion) complete extension.*
(d) preferred extension *iff $\mathfrak{S}$ is a maximal (wrt. to set inclusion) admissible extension.*
(e) stable extension *iff $\mathfrak{S}$ is conflict-free and it defeats all arguments not in $\mathfrak{S}$.*

Let $\mathfrak{sem}_{cs}$ (resp. $\mathfrak{sem}_{complete}$, $\mathfrak{sem}_{grounded}$, $\mathfrak{sem}_{preferred}$ and $\mathfrak{sem}_{stable}$) denote the semantics which assigns to a coalitional structure $\mathcal{CF}$ all its admissible (resp. complete, grounded, preferred, and stable) extensions.

There is only one unique coalition structure (possibly the empty one) for a given coalitional framework, but there can be several stable and preferred extensions. The existence of at least one preferred extension is assured which is not the case for the stable semantics. Thus, the possible coalitions very much depend on the used semantics.

[4] Actually, it is enough to assume that $\mathcal{CF}$ is finitary (cf. [13, Def. 27]).

Example 4. For $\mathcal{CF}_3$ from Example 3 the following holds:

$$\mathfrak{sem}_{\text{cs}}(\mathcal{CF}) = \{\emptyset\}$$
$$\mathfrak{sem}_{\text{admissible}}(\mathcal{CF}) = \{\{a_1\}, \{a_2\}, \{a_3\}, \{a_2, a_3\}\}$$
$$\mathfrak{sem}_{\text{complete}}(\mathcal{CF}) = \mathfrak{sem}_{\text{grounded}}(\mathcal{CF}) = \{\{a_1\}, \{a_2, a_3\}\} =$$
$$\mathfrak{sem}_{\text{preferred}}(\mathcal{CF}) = \mathfrak{sem}_{\text{stable}}(\mathcal{CF}) = \{\{a_1\}, \{a_2, a_3\}\}$$

Analogously, for the coalitional framework $\mathcal{CF}_1$ from Example 1 there exists one complete extension $\{a_1, a_3\}$ which is also a grounded, preferred, and stable extension.

4 Coalitional ATL

In this section we combine *argumentation for coalition formation* and ATL and introduce *Coalitional* ATL (CoalATL). This logic extends ATL by new operators $\langle\!|A|\!\rangle$ for each subset $A \subseteq \mathbb{A}\text{gt}$ of agents. These new modalities combine, or rather integrate, coalition formation into the original ATL cooperation modalities $\langle\langle A\rangle\rangle$. The intended reading of $\langle\!|A|\!\rangle\varphi$ is that the group A of agents *is able to form a coalition* $B \subseteq \mathbb{A}\text{gt}$ *such that some agents of* A *are also members of* B, $A \cap B \neq \emptyset$, and B *can enforce* φ. Coalition formation is modelled by the formal argumentative approach in the context of coalition formation, as described in Section 3, based on the framework developed in [3].

Our main motivation for this logic is to make it possible to reason about the ability of building coalition structures, and not only about an *a priori* specified group of agents (as it is the case for $\langle\langle A\rangle\rangle\varphi$). The new modality $\langle\!|A|\!\rangle$ provides a rather subjective view of the agents in A and their power to create a group B, $A \cap B \neq \emptyset$, which in turn is used to reason about the ability to enforce a given property.

The language of CoalATL is as follows.

Definition 12 ($\mathcal{L}_{ATL^c}$). *Let* $\mathbb{A}\text{gt} = \{a_1, \ldots, a_k\}$ *be a finite, nonempty set of agents, and* Π *be a set of propositions (with typical element* p*). We use the symbol "a" to denote a typical agent, and "A" to denote a typical group of agents from* $\mathbb{A}\text{gt}$*. The logic* $\mathcal{L}_{ATL^c}(\mathbb{A}\text{gt}, \Pi)$ *is defined by the following grammar:*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A\rangle\rangle \bigcirc\varphi \mid \langle\langle A\rangle\rangle \Box\, \varphi \mid \langle\langle A\rangle\rangle \varphi\, \mathcal{U}\, \varphi \mid$$
$$\langle\!|A|\!\rangle \bigcirc\varphi \mid \langle\!|A|\!\rangle \Box\, \varphi \mid \langle\!|A|\!\rangle \varphi\, \mathcal{U}\, \varphi$$

We extend concurrent game structures by *coalitional frameworks* and an *argumentative semantics*. A coalitional framework is assigned to each state of the model capturing the current "conflicts" among agents. In doing so, we allow that conflicts change over time, being thus *state dependent*. Moreover, we assume that coalitional frameworks are agents' dependend. Two intial groups of agents may have different skills to form coalitions. Consider for instance the following example.

Example 5. Imagine the two agents a_1 and a_2 are not able (because they do not have the money) to convince a_3 and a_4 to join. But a_1, a_2 and a_3 together have the money and all four can enforce a property φ. So $\{a_1, a_2\}$ are not able to build a greater coalition to enforce φ; but $\{a_1, a_2, a_3\}$ are. So we are not looking at coalitions per se, but how they evolve from others.

We assume that the argumentative semantics is the same for all states.

Definition 13 (CGM). *A* coalitional game model *(CGM) is given by a tuple*

$$\mathcal{M} = \langle \mathbb{A}\text{gt}, Q, \Pi, \pi, Act, d, o, \zeta, \mathfrak{sem} \rangle$$

where $\langle \mathbb{A}\text{gt}, Q, \Pi, \pi, Act, d, o \rangle$ *is a* CGS, $\zeta : \mathcal{P}(\mathbb{A}\text{gt}) \to (Q \to \mathbb{CF}(\mathbb{A}\text{gt}))$ *is a function which assigns a coalitional framework over* $\mathbb{A}$gt *to each state of the model subjective to a given group of agents, and* $\mathfrak{sem}$ *is an (argumentative) semantics as defined in Definition 8. The set of all such models is given by* $\mathbb{M}(Q, \mathbb{A}\text{gt}, \Pi, \mathfrak{sem}, \zeta)$.

A model provides an argumentation semantics $\mathfrak{sem}$ which assigns all formable coalitions to a given coalitional framework. As argued before we require from a valid coalition that it is not only justified by the argumentation semantics but that it is also not disjunct with the predetermined starting coalition. This leads to the notion *valid coalition.*

Definition 14 (Valid coalition). *Let* A, $B \subseteq \mathbb{A}\text{gt}$ *be groups of agents,* $\mathcal{M} = \langle \mathbb{A}\text{gt}, Q, \Pi, \pi, Act, d, o, \zeta, \mathfrak{sem} \rangle$ *be a* CGM *and* $q \in Q$.

We say that B *is a* valid coalition *with respect to* A, q, *and* $\mathcal{M}$ *whenever* $B \in \mathfrak{sem}(\zeta(A)(q)))$ *and* $A \cap B \neq \emptyset$. *Furthermore, we use* $\text{VC}_{\mathcal{M}}(A, q)$ *to denote the set of all valid coalitions regarding* A, q, *and* $\mathcal{M}$. *The subscript* $\mathcal{M}$ *is omitted if clear from the context.*

Remark 1. Note, that in [7] we assume that the members of the initial group A work together, whatever the reasons might be. So group A was added to the semantics. This ensured that agents in A can enforce ψ on their own, if they are able to do so. Even if A is *not* accepted originally by the argumentation semantics, i.e. $A \notin \mathfrak{sem}(\zeta(A)(q))$. Here, we drop this requirement. As pointed out in [8] the "old" semantics is just a special case of this new one: The operator from [7] can be defined as $\langle\!|A|\!\rangle\gamma \vee \langle\!\langle A \rangle\!\rangle\gamma$.

Moreover, we changed the condition that the predefined group given in the coalitional operator must be a subset of the formed coalition, $A \subseteq B$, to the requirement that some member of the inital coalition should be in the new one, $A \cap B \neq \emptyset$. Both definitions make sense in different scenarios; however, the new one seems to be more generic.

The semantics of the new modality is given by

Definition 15 (CoalATL Semantics). *Let a* CGM $\mathcal{M} = \langle \mathbb{A}\text{gt}, Q, \Pi, \pi, Act, d, o, \zeta, \mathfrak{sem} \rangle$ *a group of agents* $A \subseteq \mathbb{A}\text{gt}$, *and* $q \in Q$ *be given. The semantics of Coalitional* ATL *extends that of* ATL, *given in Definition 3, by the following rule* $(\langle\!|A|\!\rangle\psi \in \mathcal{L}_{ATL^c}(\mathbb{A}\text{gt}, \Pi))$*:*

$\mathcal{M}, q \models \langle\!|A|\!\rangle\psi$ *iff there is a coalition* $B \in \text{VC}(A, q)$ *such that* $\mathcal{M}, q \models \langle\!\langle B \rangle\!\rangle\psi$.

Remark 2 (Different Semantics, $\models_{\mathfrak{sem}}$*).* We have just defined a whole class of semantic rules for modality $\langle\!|\cdot|\!\rangle$. The actual instantiation of the semantics $\mathfrak{sem}$, for example $\mathfrak{sem}_{\text{stable}}$, $\mathfrak{sem}_{\text{pref}}$, and $\mathfrak{sem}_{\text{cs}}$ defined in Section 3, affects the semantics of the cooperation modality.

For the sake of readability, we sometimes annotate the satisfaction relation $\models$ with the presently used argumentation semantics. That is, given a CGM $\mathcal{M}$ with an argumentation semantics $\mathfrak{sem}$ we write $\models_{\mathfrak{sem}}$ instead of $\models$.

The underlying idea of the semantic definition of $\langle\!|A|\!\rangle\psi$ is as follows. A given (initial) group of agents $A \subseteq \mathbb{A}\text{gt}$ is able to form a *valid coalition* B (where A and B must not be disjoint), with respect to a given coalitional framework $\mathcal{CF}$ and a particular semantics $\mathfrak{sem}$, such that B can enforce ψ.

Similarly to the alternatives to our definition of valid coalitions there are other sensible semantics for CoalATL. The semantics we presented here is not particularly dependent on time; i.e., except from the selection of a valid coalition B at the initial state there is no further interaction between time and coalition formation. We have chosen this simplistic definition to present our main idea – the connection of ATL and coalition formation by means of argumentation – as clear as possible.

Proposition 2 ([7]). *Let* $A \subseteq \mathbb{A}\text{gt}$ *and* $\langle\!|A|\!\rangle\psi \in \mathcal{L}_{ATL^c}(\mathbb{A}\text{gt}, \Pi)$. *Then it holds that* $\langle\!|A|\!\rangle\psi \rightarrow \bigvee_{B \in \mathcal{P}(\mathbb{A}\text{gt}), A \subseteq B} \langle\!\langle B \rangle\!\rangle\psi$ *is a validity with respect to* CGM*'s.*

Compared to ATL, a formula like $\langle\!|A|\!\rangle\varphi$ does *not* refer to the ability of A to enforce φ, but rather to the ability of A to *constitute* a coalition B, such that $A \cap B \neq \emptyset$, and then, in a second step, to the ability of B to enforce φ. Thus, two different notions of ability are captured in these new modalities. For instance, $\langle\!\langle A \rangle\!\rangle\psi \wedge \neg\langle\!|\emptyset|\!\rangle\psi$ expresses that group A of agents can enforce ψ, but there is no *reasonable* coalition which can enforce ψ (particularly not A, although they possess the theoretical power to do so).

Example 6. There are three agents a_1, a_2, and a_3 which prefer different outcomes. Agent a_1 (resp. a_2, a_3) desires to get outcome r (resp. s, t). One may

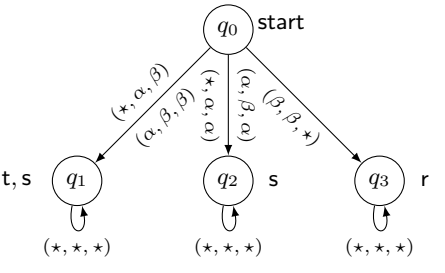


Fig. 2. A simple CGS defined in Example 6

assume that all outcomes are distinct; for instance, a_1 is not satisfied with an outcome x whenever $x \neq \mathsf{r}$. Each agent can choose to perform action α or β. Action profiles and their outcomes are shown in Figure 2. The $\star$ is used as a placeholder for any of the two actions, i.e. $\star \in \{\alpha, \beta\}$. For instance, the profile $(\beta, \beta, \star)$ leads to state q_3 whenever agent a_1 and a_2 perform action β and a_3 either does α or β.

According to the scenario depicted in the figure, a_1 and a_2 cannot commonly achieve their goals. The same holds for a_1 and a_3. On the other hand, there exists a situation, q_1, in which both agents a_2 and a_3 are satisfied. One can formalize the situation as the coalitional game $\mathcal{CF} = (\mathfrak{C}, \mathcal{A}, \prec)$ given in Example 3(b), that is, $\mathfrak{C} = \mathbb{A}\mathrm{gt}$, $\mathcal{A} = \{(a_1, a_2), (a_1, a_3), (a_2, a_1), (a_2, a_3), (a_3, a_1)\}$ and $a_2 \prec a_3$.

We formalize the example as the CGM $\mathcal{M} = \langle \mathbb{A}\mathrm{gt}, Q, \Pi, \pi, Act, d, o, \zeta, \mathfrak{sem} \rangle$ where $\mathbb{A}\mathrm{gt} = \{a_1, a_2, a_3\}$,$Q = \{q_0, q_1, q_2, q_3\}$, $\Pi = \{\mathsf{r}, \mathsf{s}, \mathsf{t}\}$, and $\zeta(A)(q) = \mathcal{CF}$ for all states $q \in Q$ and groups $A \subseteq \mathbb{A}\mathrm{gt}$. Transitions and the state labeling can be seen in Figure 2. Furthermore, we do not specify a concrete semantics $\mathfrak{sem}$ yet, and rather adjust it in the remainder of the example.

We can use pure ATL formulas, i.e. formulas not containing the new modalities $\langle\!| \cdot |\!\rangle$, to express what groups of agents can achieve. We have, for instance, that agents a_1 and a_2 can enforce a situation which is undesirable for a_3: $\mathcal{M}, q_0 \models \langle\!\langle a_1, a_2 \rangle\!\rangle \bigcirc \mathsf{r}$. Indeed, $\{a_1, a_2\}$ and the grand coalition $\mathbb{A}\mathrm{gt}$ (since it contains $\{a_1, a_2\}$) are the only coalitions which are able to enforce $\bigcirc \mathsf{r}$; we have

$$\mathcal{M}, q_0 \models \neg \langle\!\langle X \rangle\!\rangle \bigcirc \mathsf{r} \tag{1}$$

for all $X \subset \mathbb{A}\mathrm{gt}$ and $X \neq \{a_1, a_2\}$. Outcomes s or t can be enforced by a_2: $\mathcal{M}, q_0 \models \langle\!\langle a_2 \rangle\!\rangle \bigcirc (\mathsf{s} \vee \mathsf{t})$. Agents a_2 and a_3 also have the ability to enforce a situation which agrees with both of their desired outcomes: $\mathcal{M}, q_0 \models \langle\!\langle a_2, a_3 \rangle\!\rangle \bigcirc (\mathsf{s} \wedge \mathsf{t})$.

These properties do not take into account the coalitional framework, that is, whether specific coalitions can be formed or not. By using the coalitional framework, we get

$$\mathcal{M}, q_0 \models_{\mathfrak{sem}} \langle\!\langle a_1, a_2 \rangle\!\rangle \bigcirc \mathsf{r} \wedge \neg \langle\!| a_1 |\!\rangle \bigcirc \mathsf{r} \wedge \neg \langle\!| a_2 |\!\rangle \bigcirc \mathsf{r}$$

for any semantics $\mathfrak{sem}$ introduced in Definition 8 and calculated in Example 4. The possible coalition (resp. coalitions) containing a_1 (resp. a_2) is $\{a_1\}$ (resp. are $\{a_2\}$ and $\{a_2, a_3\}$). But neither of these can enforce $\bigcirc \mathsf{r}$ (in q_0) because of (1). Thus, although it is the case that the coalition $\{a_1, a_2\}$ has the theoretical ability to enforce r in the next moment (which is a "losing" situation for a_3), a_3 should not consider it as sensible since agents a_1 and a_2 would not agree to constitute a coalition (according to the coalitional framework $\mathcal{CF}$).

The decision for a specific semantics is a crucial point and depends on the actual application. The next example shows that with respect to a particular argumentation semantics, agents are able to form a coalition which can successfully achieve a given property, whereas another argumentative semantics does not allow that.

Example 7. CoalATL can be used to determine whether a coalition for enforcing a specific property exists. Assume that $\mathfrak{sem}$ represents the grounded semantics. For instance, the statement

$$\mathcal{M}, q_0 \models_{\mathfrak{sem}_{\text{grounded}}} \langle\!\langle \emptyset \rangle\!\rangle \bigcirc \mathtt{t}$$

expresses that there is a grounded coalition (i.e. a coalition wrt to the grounded semantics) which can enforce $\bigcirc \mathtt{t}$, namely the coalition $\{a_2, a_3\}$. This result does not hold for all semantics; for instance, we have

$$\mathcal{M}, q_0 \not\models_{\mathfrak{sem}_{\text{cs}}} \langle\!\langle \emptyset \rangle\!\rangle \bigcirc \mathtt{t}$$

with respect to the coalition structure semantics, since the coalition structure is the empty coalition and $\mathcal{M}, q_0 \not\models \langle\!\langle \emptyset \rangle\!\rangle \bigcirc \mathtt{t}$.

Note that it is easily possible to extend the language by an *update mechanism*, in order to compare different argumentative semantics using formulae inside the object language.

5 Cooperation and Goals

Why should agents join coalitions? Up to now we did not address this question and focussed on *why not* to cooperate. Often cooperation does not come for free and it requires some kind of incentive (i.e. sharing common goals or getting rewards) to offer one's ability in order to support other agents. Coalitional frameworks, however, were mainly used to model conflicts between agents, and therewith, avoid cooperation. In [8] the authors propose *goals* as agents' incentives to join coalitions; the following is based on that work.

5.1 The Framework

In this section a *goal framework* is incorporated into CoalATL models. First of all, each agent is equipped with a *set of goals* $\mathcal{G}_a$ where $a \in \mathbb{A}\text{gt}$ and $\mathcal{G} := \bigcup_{a \in \mathbb{A}\text{gt}} \mathcal{G}_a$. Goals are formulated as ATL-path formulae or conjunctions of them. An agent, say Bill, might have the goal – or rather a dream – that he will sometimes be able to buy a new car *without* asking other people (e.g. his wife Ann). Such a goal can be formulated as $\Diamond \langle\!\langle \text{Bill} \rangle\!\rangle \bigcirc \mathsf{buyNewCar}$. Sometimes Bill would like to *enforce* to buy a new car in the next moment. To assign goals to agents a CGM is extended by a *goal mapping.*

Definition 16 (Goal mapping $\mathfrak{g}$). *A* goal mapping *is a function* $\mathfrak{g} : \mathbb{A}\text{gt} \to (Q^+ \to \mathcal{P}(\mathcal{G}))$ *assigning a set of goals to a given sequence of states and an agent.*

So, a goal mapping assigns a set of goals to a *history.* This is needed to introduce goals into CGM's. The history dependency can be used, for instance, to model when a goal should be removed from the list: An agent having a goal $\Diamond\, \mathtt{s}$ may drop it after reaching a state in which $\mathtt{s}$ holds. Alternatively, a model update

mechanism can be used to achieve the same regarding state-based goal mappings; however, in our opinion the former is more elegant.

So far, we did not say how goals can be actually used to form coalitions. We assume, given some task, that agents having goals satisfied or partly satisfied by the outcome of the task are willing to cooperate to bring about the task. Consider, for instance, the ATL formula $\langle\langle A\rangle\rangle\gamma$. It says that A can enforce γ – the *objective*. In the context of Coalitional ATL it is even more intuitive: $\langle\!|A|\!\rangle\gamma$ means that A is able to from a coalition B which can enforce the objective γ. Of course, rational agents should have reasons to bring about γ in order to work towards γ. In the following we will use the notion *objective* (or objective formula) to refer to both the task itself and the outcome of it. A typical objective is written as o. Agents which have goals fulfilled or at least partly supported by objective o are possible candidates to participate in a coalition aiming at o. We consider CoalATL *objectives* which are CoalATL path formulae.

We say that an objective o *satisfies* goal g, $o \hookrightarrow g$, if the goal g is fulfilled after o has been accomplished. Intuitively, an objective $\Box\,\mathsf{t}$ satisfies goal $\Box\,(\mathsf{t} \vee \mathsf{s})$ an supports goal $\Diamond\,\mathsf{t}$.

5.2 Coalitional ATL with Goals

In this section we link together Coalitional ATL and the goal framework described above. The syntax of the logic is given as in Definition 12. The necessary change takes place in the semantics. We redefine what it means for a coalition to be *valid*.

Up to now valid coalitions were solely determined by coalitional frameworks. Conflicts represented by such frameworks are a coarse, but necessary, criterion for a successful coalition formation process. However, nothing is said about incentives *to join* coalitions, only why coalitions should *not* be joined.

Goals allow to capture the first issue. For a given objective formula o and a finite sequence of states, called *history*, we do only consider agents which have some goal supported by the current objective. CGM*'s with goals* are given as a straightforward extension of CGM's (cf. Definition 13).

Definition 17 (CGM with goals). *A* CGM with goals *(*CGMG*)* $\mathcal{M}$ *is given by a model of* $\mathbb{M}(Q, \mathbb{A}\mathrm{gt}, \Pi, \mathfrak{sem}, \zeta)$ *extended by a set of goals* $\mathcal{G}$ *and a goal mapping* $\mathfrak{g}$ *over* $\mathcal{G}$*. The set of all such models is denoted by* $\mathbb{M}^g(Q, \mathbb{A}\mathrm{gt}, \Pi, \mathfrak{sem}, \zeta, \mathcal{G}, \mathfrak{g})$ *or just* $\mathbb{M}^g$ *if we assume standard naming.*

To define the semantics we need some additional notation. Given a path $\lambda \in Q^\omega$ we use $\lambda[i, j]$ to denote the sequence $\lambda[i]\lambda[i+1]\ldots\lambda[j]$ for $i, j \in \mathbb{N}_0 \cup \{\infty\}$ and $i \leq j$. A *history* is a finite sequence $h = q_1 \ldots q_n \in Q^+$, $h[i]$ denotes state q_i if $n \geq i$, q_n for $i \geq n$, and ε for $i < 0$ where $i \in \mathbb{Z} \cup \{\infty\}$. Furthermore, given a history h and a path or history λ the combined path/history starting with h extended by λ is denoted by $h \circ \lambda$.

Finally, we present the semantics of CoalATL *with* goals. It is similar to Definition 15. Here, however, it is necessary to keep track of the steps (visited states) made to determine the goals of the agents.

Definition 18 (Goal-based semantics of $\mathcal{L}_{ATL^c}$). *Let $\mathcal{M}$ be a* CGMG*, q a state, φ, ψ state-, γ a path formula, and $i, j \in \mathbb{N}_0$. The goal-based semantics of $\mathcal{L}_{ATL^c}$ formulae is given as follows:*

$\mathcal{M}, q, \tau \models \mathsf{p}$ *iff* $\mathsf{p} \in \pi(q)$
$\mathcal{M}, q, \tau \models \varphi \wedge \psi$ *iff* $\mathcal{M}, q, \tau \models \varphi$ *and* $\mathcal{M}, q, \tau \models \psi$
$\mathcal{M}, q, \tau \models \neg\varphi$ *iff not* $\mathcal{M}, q, \tau \models \varphi$
$\mathcal{M}, q, \tau \models \langle\langle A \rangle\rangle \gamma$ *iff there is a strategy* $s_A \in \Sigma_A$ *such that for all* $\lambda \in out(q, s_A)$ *it holds that* $\mathcal{M}, \lambda, \tau \models \varphi$
$\mathcal{M}, q, \tau \models [\![A]\!] \gamma$ *iff there is* $A' \in \text{VC}^g(q, A, \gamma, \tau)$ *such that* $\mathcal{M}, q, \tau \models \langle\langle A \rangle\rangle \gamma$
$\mathcal{M}, \lambda, \tau \models \varphi$ *iff* $\mathcal{M}, \lambda[0], \tau \models \varphi$
$\mathcal{M}, \lambda, \tau \models \Box\, \varphi$ *iff for all* i *it holds that* $\mathcal{M}, \lambda[i], \tau \circ \lambda[1, i] \models \varphi$
$\mathcal{M}, \lambda, \tau \models \bigcirc \varphi$ *iff it holds that* $\mathcal{M}, \lambda[1], \tau \circ \lambda[1] \models \varphi$
$\mathcal{M}, \lambda, \tau \models \varphi \mathcal{U} \psi$ *iff there is a* j *such that* $\mathcal{M}, \lambda[j], \tau \circ \lambda[1, j] \models \psi$ *and for all* $0 \leq i < j$ *it holds that* $\mathcal{M}, \lambda[i], \tau \circ \lambda[1, i] \models \varphi$.

Ultimately, we are interested in $\mathcal{M}, q \models \varphi$ defined as $\mathcal{M}, q, q \models \varphi$.

All the new functionality provided by goals is captured by the new valid coalition function VC^g

Definition 19 (Valid coalitions, $\text{VC}^g(q, A, o, \tau)$). *Let $\mathcal{M} \in \mathbb{M}^g$, $\tau \in Q^+$, $A, B \subseteq \mathbb{A}\text{gt}$, o an* CoalATL *objective.*

We say that B is a valid coalition *after τ with respect to A, o, and $\mathcal{M}$ if, and only if,*

1. *$B \in \mathfrak{sem}(\zeta(\tau[\infty])(A))$, $A \cap B \neq \emptyset$, and*
2. *there are goals $g_{b_i} \in \mathfrak{g}_{b_i}(\tau)$, one per agent $b_i \in B$, such that $o \hookrightarrow_{\mathcal{M},\tau,B} g_{b_1} \wedge \cdots \wedge g_{b_{|B|}}$*

The set $\text{VC}^g(q, A, o, \tau)$ consists of all such valid coalitions wrt to $\mathcal{M}$.

Thus, for the definition of valid coalitions among other things, a goal mapping, a function ζ and a sequence of states τ are required. The intuition of τ is that it represents the history (the sequence of states visited so far including the current state). So, τ is used to determine which goals of the agents are still active.

Finally, we have to define when a goal is satisfied.

Definition 20 (Satisfaction of goals). *Let g be an* ATL*-goal, o an $\mathcal{L}_{ATL^c}$-objective, and $\tau \in Q^+$. We say that objective o* satisfies *g, for short $o \hookrightarrow_{\mathcal{M},\tau,B} g$, with respect to $\mathcal{M}, \tau$, and B if, and only if, there is a strategy $s_B \in \Sigma_B$ such that*

1. *for all $\lambda \in out(\tau[\infty], s_B)$ it holds that $\mathcal{M}, \lambda, \tau \models o$ implies $\mathcal{M}, \lambda \models g$, and*
2. *that there is some path $\lambda \in out(\tau[\infty], s_B)$ with $\mathcal{M}, \lambda, \tau \models o$.*

A goal is satisfied by an objective if each path (enforceable by B) that satisfies the objective does also satisfy the goal. That is, satisfaction of the objective will guarantee that the goal becomes true. The second condition ensures that the

coalition actually has a way to bring about the goal. However, in [8] it is shown that the second condition is superfluous.

It remains to define the semantics for combined (by conjunction) ATL path formulae. Therefore, we extend the ordinary semantics (given in Definition 3) by the following semantic rule: $\mathcal{M}, \lambda \models \gamma_1 \wedge \cdots \wedge \gamma_n$ iff $\mathcal{M}, \lambda \models \gamma_i$ for $i = 1, \ldots, n$.

6 Model Checking ATL^c

In this section we present an algorithm for model checking CoalATL formulae. The model checking problem is given by the question whether a given CoalATL formula follows from a given CGM $\mathcal{M}$ and state q, i.e. whether $\mathcal{M}, q \models \varphi$ [12]. In [2] it is shown that model checking ATL is **P**-complete, with respect to the number of transitions of $\mathcal{M}$, m, and the length of the formula, l, and can be done in time $\mathcal{O}(m \cdot l)$.

For CoalATL we also have to treat the new coalitional modalities in addition to the normal ATL constructs. Let us consider the formula $\langle\!|A|\!\rangle\psi$. According to the semantics of $\langle\!|A|\!\rangle$, given in Definition 15, we must check whether there is a coalition B such that (i) $A \cap B \neq \emptyset$, (ii) B is acceptable by the argumentation semantics, and (iii) $\langle\!\langle B \rangle\!\rangle\psi$. The number of possible candidate coalitions B which satisfy (i) and (ii) is bounded by $|\mathcal{P}(\mathbb{A}\mathrm{gt})|$. Thus, in the worst case there might be *exponentially* many calls to a procedure checking whether $\langle\!\langle B \rangle\!\rangle\psi$. Another source of complexity is the time needed to compute the argumentation semantics. In [14], for instance, it is stated that credulous acceptance[5] using preferred semantics is **NP**-complete.

Both considerations together suggest that the model checking complexity has two computationally hard parts: exponentially many calls to $\langle\!\langle A \rangle\!\rangle\psi$ and the computation of the argumentation semantics. Indeed, Theorem 1 will support this intuition. However, we show that it is possible to "combine" both computationally hard parts to obtain an algorithm which is in $\mathbf{\Delta_2^P} = \mathbf{P^{NP}}$, if the computational complexity to determine whether a given coalition is acceptable are not harder than **NP**.

For the rest of this section, we will denote by $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}}$ the set of all coalitions A such that A is acceptable according to the coalitional framework $\mathcal{CF}$ and the argumentation semantics $\mathfrak{sem}$, i.e. $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}} := \{A \mid A \in \mathfrak{sem}(\mathcal{CF})\}$.

Given some complexity class $\mathcal{C}$, we use the notation $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}} \in \mathcal{C}$ to state that the word problem of $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}}$, i.e., whether A is a member of $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}}$, is in $\mathcal{C}$. Actually in [7] it was statet that $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}} \in \mathbf{P}$ for all semantics $\mathfrak{sem}$ defined in Definition 11. In Figure 3 we propose a model checking algorithm for CoalATL. The complexity result given in the next theorem is modulo the complexity needed to compute membership in $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}}$.

Theorem 1 (Model checking CoalATL [7]). *Let a* CGM $\mathcal{M} = \langle \mathbb{A}\mathrm{gt}, Q, \Pi, \pi, Act, d, o, \zeta, \mathfrak{sem} \rangle$ *be given,* $q \in Q$, $\varphi \in \mathcal{L}_{ATL^c}(\mathbb{A}\mathrm{gt}, \Pi)$, *and* $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}} \in \mathcal{C}$. *Model checking* CoalATL *with respect to the argumentation semantics* $\mathfrak{sem}$[6] *is in* $\mathbf{P^{NP^C}}$.

[5] That is, whether an argument is in *some* preferred extension.

[6] That is, whether $\mathcal{M}, q \models_{\mathfrak{sem}} \varphi$.

function $mcheck(\mathcal{M}, q, \varphi)$**;**
Given a CGM $\mathcal{M} = \langle \mathbb{A}\mathrm{gt}, Q, \Pi, \pi, Act, d, o, \zeta, \mathfrak{sem} \rangle$, a state $q \in Q$, and $\varphi \in \mathcal{L}_{ATL^c}(\mathbb{A}\mathrm{gt}, \Pi)$ the algorithm returns $\top$ if, and only if, $\mathcal{M}, q \models_{\mathfrak{sem}} \varphi$.

case φ **contains no** $\langle\!|B|\!\rangle$**: if** $q \in mcheck_{\mathrm{ATL}}(\mathcal{M}, \varphi)$ **return** $\top$ **else** $\bot$
case φ **contains some** $\langle\!|B|\!\rangle$**:**
- **case** $\varphi \equiv \neg\psi$**: return** $\neg(\mathcal{M}, q, \psi)$
- **case** $\varphi \equiv \psi \vee \psi'$**: return** $mcheck(\mathcal{M}, q, \psi) \vee mcheck(\mathcal{M}, q, \psi')$
- **case** $\varphi \equiv \langle\langle A \rangle\rangle T\psi$**:** Label all states q' where $mcheck(\mathcal{M}, q', \psi) == \top$ with a new proposition yes and return $mcheck(\mathcal{M}, q, \langle\langle A \rangle\rangle T\mathsf{yes})$; T stands for $\Box$ or $\bigcirc$.
- **case** $\varphi \equiv \langle\langle A \rangle\rangle \psi \mathcal{U} \psi'$**:** Label all states q' where $mcheck(\mathcal{M}, q', \psi) == \top$ with a new proposition yes_1, all states q' where $mcheck(\mathcal{M}, q', \psi') == \top$ with a new proposition yes_2 and return $mcheck(\mathcal{M}, q, \langle\!|A|\!\rangle \mathsf{yes}_1 \mathcal{U}\, \mathsf{yes}_2)$
- **case** $\varphi \equiv \langle\!|A|\!\rangle T\psi$**,** ψ **contains some** $\langle\!|C|\!\rangle$**:** Label all states q' where $mcheck(\mathcal{M}, q', \psi) == \top$ with a new proposition yes and return $mcheck(\mathcal{M}, q, \langle\!|A|\!\rangle T\mathsf{yes})$; T stands for $\Box$ or $\bigcirc$.
- **case** $\varphi \equiv \langle\!|A|\!\rangle \psi \mathcal{U} \psi'$**,** ψ **or** ψ' **contain some** $\langle\!|C|\!\rangle$**:** Label all states q' where $mcheck(\mathcal{M}, q', \psi) == \top$ with a new proposition yes_1, all states q' where $mcheck(\mathcal{M}, q', \psi') == \top$ with a new proposition yes_2 and return $mcheck(\mathcal{M}, q, \langle\!|A|\!\rangle \mathsf{yes}_1 \mathcal{U}\, \mathsf{yes}_2)$
- **case** $\varphi \equiv \langle\!|A|\!\rangle \psi$ **and** ψ **contains no** $\langle\!|C|\!\rangle$**:** Non-deterministically choose $B \in \mathcal{P}(\mathbb{A}\mathrm{gt})$
  if
  (1) $B \in (\mathfrak{sem}(\zeta(A)(q)))$,
  (2) $A \cap B \neq \emptyset$, and (⋆)
  (3) $q \in mcheck_{\mathrm{ATL}}(\mathcal{M}, \langle\langle B \rangle\rangle \psi)$
  then return $\top$ **else** $\bot$

function $mcheck_{\mathbf{ATL}}(\mathcal{M}, \varphi)$**;**
Given a CGS $\mathcal{M} = \langle \mathbb{A}\mathrm{gt}, Q, \Pi, \pi, Act, d, o \rangle$ and $\varphi \in \mathcal{L}_{ATL}(\mathbb{A}\mathrm{gt}, \Pi)$, the standard ATL model checking algorithm (cf. [2]) returns all states q with $\mathcal{M}, q \models_{\mathrm{ATL}} \varphi$.

- return $\{q \in Q \mid \mathcal{M}, q \models_{\mathrm{ATL}} \varphi\}$

Fig. 3. A model checking algorithm for CoalATL

The last theorem gives an upper bound for model checking CoalATL with respect to an arbitrary but fixed semantics $\mathfrak{sem}$. A finer grained classification of the computational complexity of $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}}$ allows to improve the upper bound given in Theorem 1. Assume that $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}} \in \mathbf{P}$ and consider the last case of function *mcheck* in Figure 3 labelled by (⋆), $\varphi \equiv \langle\!|A|\!\rangle \psi$. First, a coalition $B \in \mathcal{P}(\mathbb{A}\mathrm{gt})$ is non-deterministically chosen and then, it is checked whether B satisfies the three conditions (1-3) in (⋆). Each of the three tests can be done in deterministic polynomial time. Hence, the verification of $\mathcal{M}, q \models \langle\!|A|\!\rangle \psi$, in the last case, meets the "guess and verify" principle which is characteristic for problems in **NP**. This

brings the overall complexity of the algorithm to $\mathbf{\Delta_2^P}$. More surprisingly, the same result holds even for the case where $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}} \in \mathbf{NP}$.

Corollary 1 ([7]). *If $\mathcal{L}_{\mathfrak{sem},\mathcal{CF}} \in \mathbf{NP}$ (resp. $\mathbf{NP}$-complete) then model checking* CoalATL *is in $\mathbf{\Delta_2^P}$ (resp. $\mathbf{\Delta_2^P}$-complete) with respect to $\mathfrak{sem}$.*

In [14] the complexity of credulous reasoning with respect to the preferred and stable extensions is analyzed and determined to be **NP**-complete. This is in the line with our result: there can be a polynomial number of calls to $mcheck(\mathcal{M}, q, \langle\!|A|\!\rangle\psi)$ (where ψ does not contain any cooperation modality $\langle\!| \cdot |\!\rangle$). Now, the problem of checking whether $mcheck(\mathcal{M}, q, \langle\!|A|\!\rangle\psi)$ holds is very similar to checking whether some argument is credulously accepted. In both cases we have to ask for the existence of a set X with specific properties (in our framework we refer to X as a coalition and in [14] as an argument) which can be validated in polynomial deterministic time.

Corollary 2 ([7]). *Model checking* CoalATL *is in $\mathbf{\Delta_2^P}$ for all semantics defined in Definition 11.*

7 Related and Future Work

Related Work. The main inspiration for our work was the powerful argument-based model for reasoning about coalition structures proposed by Amgoud [3]. Indeed, our notion of coalitional framework (Def. 4) is based on the notion of framework for generating coalition structures (FCS) presented in Amgoud's paper. However, in contrast with Amgoud's proposal, our work is concerned with extending ATL by argumentation in order to model coalition formation.

Previous research by Hattori *et al.* [16] has also addressed the problem of argument-based coalition formation, but from a different perspective than ours. In [16] the authors propose an argumentation-based negotiation method for coalition formation which combines a logical framework and an argument evaluation mechanism. The resulting system involves several user agents and a mediator agent. During the negotiation, the mediator agent encourages appropriate user agents to join in a coalition in order to facilitate reaching an agreement. User agents advance proposals using a part of the user's valuations in order to reflect the user's preferences in an agreement. This approach differs greatly from our proposal, as we are not concerned with the negotiation process among agents, and our focus is on modelling coalitions within an extension of a highly expressive temporal logic, where coalition formation is part of the logical language.

Modelling argument-based reasoning with bounded rationality has also been the focus of previous research. In [26] the authors propose the use of a framework for argument-based negotiation, which allows for a strategic and adaptive communication to achieve private goals within the limits of bounded rationality in open argumentation communities. In contrast with our approach, the focus here is not on extending a particular logic for reasoning about coalitions, as in our case. Recent research in formalizing coalition formation has been oriented

towards adding more expressivity to Pauly's coalition logic [22]. E.g. in [1], the authors define *Quantified Coalition Logic*, extending coalition logic with a limited but useful form of quantification to express properties such as *"there exists a coalition C satisfying property P such that C can achieve ϕ"*. In [5], a semantic translation from coalition logic to a fragment of an action logic is defined, connecting the notions of coalition power and the actions of the agents. However, in none of these cases argumentation is used to model the notion of coalition formation as done in this paper.

It must be noted that the adequate formalization of preferences has deserved considerable attention within the argumentation community. In preference-based argumentation theory, an argument may be preferred to another one when, for example, it is more specific, its beliefs have a higher probability or certainty, or it promotes a higher value. Recent work by Kaci *et al.* [19,20] has provided interesting contributions in this direction, including default reasoning abilities about the preferences over the arguments, as well as an algorithm to derive the most likely preference order.

Future Work. Indeed, in the line with the previous paragraph, one of our future research lines is to extend our current formalization of CoalATL to capture more complex issues in preference handling and to consider more sophisticated semantics.

In the semantics presented in Definition 15 a valid coalition is initially formed and kept until the property is fulfilled. For instance, consider formula $\langle\!\langle A \rangle\!\rangle \Box\, \varphi$. The formula is true in q if a valid coalition B *in* q can be formed such that it can ensure $\Box\, \varphi$. On might strengthen the scenario and require that B must be valid in *each* state on the path λ satisfying φ. Formally, the semantics could be given as follows: $q \models \langle\!\langle A \rangle\!\rangle \Box\, \varphi$ if, and only if, $q \models \varphi$ and there is a coalition $B \in \textsc{vc}(q, A)$ and a common strategy $s_B \in \Sigma_B$ such that for all paths $\lambda \in out(q, s_B)$ and for all $i \in \mathbb{N}$ it holds that $\lambda[i] \models \varphi$ *and* $B \in \textsc{vc}(\lambda[i], A)$. The last part specifies that B must be a valid coalition in each state $q_i = \lambda[i]$ of λ.

In the semantics just presented the formed coalition B must persist over time until φ is enforced. One can go one step further. Instead of keeping the same coalition B it can also be sensible to consider "new" valid coalitions in each time step (wrt. A), possibly distinct from B. This leads to some kind of fixed-point definition. At first, B must be a valid coalition in state q leading to a state in which φ is fulfilled and in which another valid coalition (wrt. A and the new state) exists which in turn can ensure to enter a state in which, again, there is another valid coalition and so on.

Finally, part of our future work also involves the actual implementation of a subset of CoalATL, restricted to some particular argumentation semantics for which proof procedures can be deployed, such as assumption-based argumentation [15] in order to perform experiments to assess our proposal when modelling complex problems. Research in this direction is currently being pursued. Also the model checking complexity of the goal-based semantics is left for future research.

8 Conclusions

In this paper we have presented CoalATL, an extension of ATL which is able to model coalition formation through argumentation. Our formalism includes two different modalities, $\langle\langle A \rangle\rangle$ and $\langle\!| A |\!\rangle$, which refer to different kinds of abilities agents may have. Note that the original operator $\langle\langle A \rangle\rangle$ is used to reason about the pure ability of the very group A. However, the question whether it is reasonable to assume that the members of A collaborate is not taken into account in ATL. With the new operator $\langle\!| A |\!\rangle$ we try to close this gap, providing also a way to focus on *sensible coalition structures*. In this context, "sensible" refers to *acceptable coalitions with respect to some argumentative semantics* (as characterized in Def. 8).

Furthermore, we have defined the formal machinery required for characterizing argument-based coalition formation in terms of the proposed operator $\langle\!| A |\!\rangle$. Coalitions can be actually computed in terms of a given argumentation semantics, which can be given as a parameter within our model, thus providing a modular way of analyzing the results associated with different alternative semantics. This allows us to compare the ability of agents to form particular coalitions and study emerging properties regarding different semantics. Additionally, as outlined in Section 6, the model checking algorithm used in ATL can be extended to CoalATL by integrating suitable proof procedures for argumentation semantics.

Acknowledgements. This research was partially funded by the Projects DAAD-SeCyT (DA0609) and PGI-UNS (24/ZN10). The authors would also like to thank the anonymous reviewers for their useful comments.

References

1. Ågotnes, T., van der Hoek, W., Wooldridge, M.: Quantified coalition logic. In: IJCAI, pp. 1181–1186 (2007)
2. Alur, R., Henzinger, T.A., Kupferman, O.: Alternating-time Temporal Logic. Journal of the ACM 49, 672–713 (2002)
3. Amgoud, L.: An argumentation-based model for reasoning about coalition structures. In: Parsons, S., Maudet, N., Moraitis, P., Rahwan, I. (eds.) ArgMAS 2005. LNCS, vol. 4049, pp. 217–228. Springer, Heidelberg (2006)
4. Amgoud, L.: Towards a formal model for task allocation via coalition formation. In: AAMAS, pp. 1185–1186 (2005)
5. Borgo, S.: Coalitions in action logic. In: IJCAI, pp. 1822–1827 (2007)
6. Brena, R.F., Aguirre, J.-L., Chesñevar, C.I., Ramírez, E.H., Garrido, L.: Knowledge and information distribution leveraged by intelligent agents. Knowl. Inf. Syst. 12(2), 203–227 (2007)
7. Bulling, N., Chesñevar, C., Dix, J.: Modelling coalitions: ATL+argumentation. In: Proceedings of AAMAS 2008, Estoril, Portugal, May 2008. ACM Press, New York (2008)
8. Bulling, N., Dix, J.: A finer grained modeling of rational coalitions using goals. In: Proceedings of the 14th Argentinean Conference on Computer Science CACIC 2008 (to appear, 2008)

9. Bulling, N., Jamroga, W., Dix, J.: Reasoning about temporal properties of rational play. Annals of Mathematics and Artificial Intelligence (to appear)
10. Caminada, M.: Semi-stable semantics. In: Intl. Conference on Computational Models of Argument (COMMA), pp. 121–130 (2006)
11. Chesñevar, C.I., Maguitman, A.G., Loui, R.P.: Logical models of argument. ACM Comput. Surv. 32(4), 337–383 (2000)
12. Clarke, E., Grumberg, O., Peled, D.: Model Checking. MIT Press, Cambridge (1999)
13. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell. 77(2), 321–358 (1995)
14. Dunne, P.E., Bench-Capon, T.J.M.: Two party immediate response disputes: Properties and efficiency. Artif. Intell. 149(2), 221–250 (2003)
15. Gaertner, D., Toni, F.: Computing arguments and attacks in assumption-based argumentation. IEEE Intelligent Systems 22(6), 24–33 (2007)
16. Hattori, H., Ito, T., Ozono, T., Shintani, T.: An approach to coalition formation using argumentation-based negotiation in multi-agent systems. In: Monostori, L., Váncza, J., Ali, M. (eds.) IEA/AIE 2001. LNCS, vol. 2070, pp. 687–696. Springer, Heidelberg (2001)
17. Jamroga, W., Bulling, N.: A general framework for reasoning about rational agents. In: Proceedings of AAMAS 2007, Honolulu, Hawaii, USA, pp. 592–594. ACM Press, New York (2007)
18. Jamroga, W., Bulling, N.: A logic for reasoning about rational agents. In: Sadri, F., Satoh, K. (eds.) Proceedings of CLIMA 2007, Porto, Portugal, Univesidade Do Porto, pp. 54–69 (2007)
19. Kaci, S., van der Torre, L.: Preference-based argumentation: Arguments supporting multiple values. Int. J. Approx. Reasoning 48(3), 730–751 (2008)
20. Kaci, S., van der Torre, L.W.N., Weydert, E.: On the acceptability of incompatible arguments. In: Mellouli, K. (ed.) ECSQARU 2007. LNCS, vol. 4724, pp. 247–258. Springer, Heidelberg (2007)
21. Karunatillake, N.C., Jennings, N.R., Rahwan, I., Ramchurn, S.D.: Managing social influences through argumentation-based negotiation. In: Proc. of AAMAS, pp. 426–428 (2006)
22. Pauly, M.: A modal logic for coalitional power in games. J. Log. Comput. 12(1), 149–166 (2002)
23. Prakken, H., Vreeswijk, G.: Logical Systems for Defeasible Argumentation. In: Gabbay, D., Guenther, F. (eds.) Handbook of Phil. Logic, pp. 219–318. Kluwer, Dordrecht (2002)
24. Rahwan, I., Amgoud, L.: An argumentation based approach for practical reasoning. In: Proc. of AAMAS Conf., pp. 347–354 (2006)
25. Rahwan, I., Pasquier, P., Sonenberg, L., Dignum, F.: On the benefits of exploiting underlying goals in argument-based negotiation. In: Proc. of AAAI Conf., pp. 116–121 (2007)
26. Rovatsos, M., Rahwan, I., Fischer, F.A., Weiß, G.: Practical strategic reasoning and adaptation in rational argument-based negotiation. In: Parsons, S., Maudet, N., Moraitis, P., Rahwan, I. (eds.) ArgMAS 2005. LNCS, vol. 4049, pp. 122–137. Springer, Heidelberg (2006)
27. Tang, Y., Parsons, S.: Argumentation-based dialogues for deliberation. In: Proc. of AAMAS, pp. 552–559 (2005)

A Dialogue Mechanism for Public Argumentation Using Conversation Policies

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

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

Abstract. In this paper, we propose a flexible dialogue mechanism through which a set of agents can establish a coherent set of public beliefs. Flexibility and coherence are achieved by decomposing the dialogue mechanism into two parts, a backbone protocol and a set of conversation policies. The backbone protocol maintains the set of arguments put forward by the agents, and each agent uses a pre-agreed argumentation theory to extract a set of public beliefs from this set of arguments. The flexibility is achieved by distributing the other functions of the dialogue mechanism among a set of conversation policies, some of which are public and some of which are private to each agent.

1 Introduction

Multiagent systems need a mechanism by which they can communicate in order to coordinate their efforts to achieve tasks that are assigned to the system [32]. Furthermore, this mechanism should be flexible enough to enable a human designer to incrementally add more and more building blocks to the mechanism as understanding of the tasks evolve and the tasks themselves change. Many approaches have been proposed for the communication mechanism — see [17,26] for surveys. Argumentation based dialogues [4,24,26] have proved to be a general approach to agent communication in which the agents exchange not only statements of what they believe and what they want but also the reasons why. In this approach, a *protocol* or a *conversation policy* is used to govern the valid sequences of dialogue moves and then argumentation-based reasoning is used by individual agents to resolve the conflicts arising from the information that they hold privately and the messages that they receive.

The set of beliefs held by the agent society as a whole can be implicitly deduced from the common beliefs that all the agents have obtained by their own private argumentation. However, this causes two interrelated problems. Firstly, the way that protocols are specified cannot be separated from the specification of the agents that use the protocol. Indeed, such protocols are typically specified in a way that explicitly depends on the internal mechanisms that the agent uses. Secondly, it is hard to see the impact of the compositions of dialogue protocols on each individual agent's beliefs as well as on the

I. Rahwan and P. Moraitis (Eds.): ArgMas 2008, LNAI 5384, pp. 217–235, 2009.

implicit public belief set [17]. These problems will prevent the dialogues from achieving the desiderata and criteria of a good dialogue as suggested in [17] and [20], such as flexibility and verifiability.

In response to these problems, we propose a flexible dialogue mechanism through which the agents can establish a coherent public belief set. Flexibility and public coherence is achieved together by decomposing the mechanism into two parts — a backbone protocol and a set of conversation policies. The backbone protocol maintains a shared context of messages that have been exchanged between agents in the form of arguments and defeats, and each agent uses a pre-agreed argumentation theory to extract the public belief set from this set of messages. Flexibility is achieved by distributing the remaining functions of the dialogue mechanism among a set of conversation policies. Some of these policies will need to be obeyed publicly to regulate the kinds of arguments that can be asserted into the dialogue context, and other policies will be private to each individual agent and be used to decide which arguments and defeats should be generated out of the agent's own private information base. The public aspect of conversation policies is to have the agents cooperate together to create the set of public beliefs. The private aspect of conversation policies is to offer freedom and flexibility for individual agents to solve the problems from different perspectives.

2 Related Work

Argumentation theory, which is used to establish the public set of beliefs in our mechanism, has been used in a range of ways in artificial intelligence in general [9,10] and multiagent communications in particular [4,24,26]. In the general field of artificial intelligence, argumentation-based reasoning has mainly been used to unify a number of approaches to nonmonotonic reasoning [10], to reason about uncertainty [16], to reason coherently from inconsistent information [2], to perform decision making and practical reasoning [7], and to handle conflicting desires [1].

These applications of argumentation based reasoning suggest that the approach is a solution for resolving conflicts arising from agent communication about issues involving uncertainty, inconsistency, decision making and practical reasoning — all the things that researchers have shown that can be handled using argumentation. However, the most interesting property of argumentation based reasoning to us is the concept of "external stability" [10] through which a set of coherent beliefs is characterized by the relations between the arguments "internally" supporting the beliefs and the arguments "externally" supporting the contradictory beliefs. In an agent society, because of the diversity and the dynamics of the situated environment, messages from different agents and messages from the same agent at different times often convey conflicting information. In many cases, these conflicts can not be resolved by just looking at consecutive messages in a dialogue. Therefore most dialogue protocols — for example [24] — include complex sets of rules about asserting statements into and retracting the statements from the set of public beliefs with the aim of keeping those beliefs coherent. In contrast, in our approach, argumentation, with the "external stability" property, ensures the coherence of the public belief set almost for free by just choosing different semantics and different computation methods for different applications.

Furthermore, there are many recent advances in argumentation based reasoning such as the work of Cayrol and Lagasquie-Schiex [8], Jakobovits and Vermeir [14], Besnard and Hunter [6], and Pollock [23], that have expanded our understanding of what argumentation can be used for, and have created a bridge to possibility theory and plausibility theory in the field of reasoning about uncertainty [12]. The new systems that have emerged from this research are not usable in existing argumentation-based dialogue systems because of the way that the latter tightly couple the dialogue protocol with the underlying argumentation theory. The dialogue mechanism proposed in this paper paves the way to use these new argumentation reasoning theories freely in dialogues by decoupling dialogue from any specific model of argumentation.

3 An Argumentation Framework

An *argumentation framework* is a pair $\langle \mathcal{A}, \mathcal{R} \rangle$ where $\mathcal{A}$ is a set of arguments, and $\mathcal{R}$ is a binary defeat relation over the set of arguments. The set of arguments are induced from an information base, denoted by Σ. Σ is represented in a logical language $\mathcal{L}$ with the standard connectives $\wedge, \vee, \neg, \equiv$. An entailment relationship $\vdash$ is required to be defined on $\mathcal{L}$. Inconsistent information is allowed in Σ to accommodate conflicting information in the information base. The defeat relation $\mathcal{R}$ will be induced from Σ to recapture the inconsistency of the information base at the level of arguments. Once we have the set of arguments and the set of defeats, we adopt a set of principles, principles drawn from the philosophical and linguistic study of human argumentation and fallacious reasoning [31], that we can use to analyze the outcome of the argument set and the defeat set.

The rest of this section will be devoted to describe one such framework and its components formally. The framework is mostly drawn from the work of Amgoud and her colleagues [2,3] with some slight modifications.

Definition 1. *An argument based on Σ is pair (H, h) where $H \subseteq \Sigma$ such that*

1. *H is consistent with respect to $\mathcal{L}$,*
2. *$H \vdash h$,*
3. *H is minimal (for set inclusion).*

H is called the support and h is called the conclusion of the argument. $\mathcal{A}(\Sigma)$ denotes the set of all arguments which can be constructed from Σ.

This definition of argument can be understood as constraints on how pieces of coherence information can be clustered as arguments. Condition (1) is to ensure that an argument is a coherent. The coherence of an agent's information is defined in terms of the consistency of the language $\mathcal{L}$ in which the information is written. Condition (2) can be understood as insisting that the conclusion of an argument should be supported by a set of information in the sense of inference in the language $\mathcal{L}$. Condition (3) can be understood as saying that no redundant information should appear in an argument. This definition of argument is chosen from Amgoud's work because its form is simple. Our proposed dialogue mechanism in Section 4 doesn't prevent the application from choosing another form of argument as long as there is a process to generate the arguments and check their validity.

Definition 2. *(H', h') is a subargument of the argument (H, h) iff $H' \subseteq H$.*

Definition 3. *Let (H_1, h_1), (H_2, h_2) be two arguments of $\mathcal{A}(\Sigma)$.*

1. *(H_1, h_1) rebuts (H_2, h_2) iff $h_1 \equiv \neg h_2$.*
2. *(H_1, h_1) undercuts (H_2, h_2) iff $\exists h \in H_2$ such that $h \equiv \neg h_2$.*
3. *(H_1, h_1) contradicts (H_2, h_2) iff (H_1, h_1) rebuts a subargument of (H_2, h_2).*

The binary relations $rebut$, $undercut$, and $contradict$ gather all pairs of arguments satisfying conditions (1), (2) and (3) respectively.

The relations $rebut$, $undercut$, and $contradict$ will be collectively referred to as $defeat$ if no distinction is necessary, and we will write $defeat((H_1, h_1), (H_2, h_2))$ if (H_1, h_1) rebuts, undercuts or contradicts (H_2, h_2).

The definition of these forms of defeat can be viewed as recapturing the inconsistency of the original information into a conflict relation among the arguments in terms of the fallacious reasoning recorded in the arguments. $rebut$ means that the two arguments leads to conflicting conclusions in the sense of $\mathcal{L}$. $undercut$ means that the one argument's conclusion conflicts with another argument's premise. $contradict$ means that one argument's conclusion conflicts with a conclusion which can be extended, using the inferences in $\mathcal{L}$, from one or many segments of another argument's support. In contrast to $rebut$ and $undercut$, $contradict$ penetrates into arguments and explores various parts of the arguments to detect conflicting points with respect to $\mathcal{L}$.

These notions of defeat are close, but none is equivalent to, or subsumes, the other in general. If we define arguments of the form $(\{a\}, a)$, where the conclusion is also the support, to be *degenerate*, then we can easily show that:

Proposition 1. *Let (H_1, h_1) and (H_2, h_2) be two arguments.*

1. *If (H_1, h_1) rebuts (H_2, h_2) then it also undercuts (H_2, h_2) iff (H_2, h_2) is degenerate.*
2. *If (H_1, h_1) undercuts (H_2, h_2) then it contradicts (H_2, h_2) iff (H_2, h_2) is degenerate,*

Proof. We can easily see the equivalence of rebut, contradict and undercut on a degenerate argument from an example. $(H_1, \neg a)$, rebuts, undercuts and contradicts $(\{a\}, a)$. In general, for rebuttal to entail undercut, the conclusion has to be in the support, and by the minimality condition on arguments, the undercut/rebutted argument must therefore be degenerate. Similarly, for undercut to entail contradiction, the element of the support that is attacked by the undercutter must also be the conclusion of the undercut argument. Hence it must be degenerate.

[2] gives a detailed discussion on how these definitions of defeat will affect the behaviors of an argumentation framework, while [29,30] provide a more detailed discussion on the concepts and forms of defeat. In later sections we will only use $undercut$.

Following Dung's work [10], we have the following component definitions of the theory.

Definition 4. *An* argumentation framework *is a pair, $Args = \langle \mathcal{A}, \mathcal{R} \rangle$, where $\mathcal{A}$ is a set of arguments, and $\mathcal{R}$ is a binary defeat relation over the arguments.*

Definition 5. *Let $\langle \mathcal{A}, \mathcal{R} \rangle$ be an argumentation framework, and $S \subseteq \mathcal{A}$. An argument A is defended by S iff $\forall B \in \mathcal{A}$ if $(B, A) \in \mathcal{R}$ then $\exists C \in S$ such that $(C, B) \in \mathcal{R}$.*

Definition 6. *$S \subseteq \mathcal{A}$. $\mathcal{F}_{\mathcal{R}}(S) = \{A \in \mathcal{A} | A$ is defended by S with respect to $\mathcal{R}\}$.*

Now, for a function $F : D \rightarrow D$ where D is the domain and the range of the function, a fixed point of F is an $x \in D$ such that $x = F(x)$. When the D is associated with an ordering P — for example, P can be set inclusion over the power set D of arguments — x is a *least fixpoint* of F if x is a least element of D with respect to P and x is a fixed point.

Definition 7. *Let $\langle \mathcal{A}, \mathcal{R} \rangle$ be an argumentation framework. The set of acceptable arguments, denoted by $Acc^F_{\mathcal{R}}$, is the least fixpoint of the function $\mathcal{F}_{\mathcal{R}}$ with respect to set inclusion.*

The least fixpoint semantics can be viewed as a mathematical translation of the principle that an argument survives if it can defend itself or be defended by a set of arguments which can also survive all the attacks made upon them.

It is possible to provide alternative semantics for argumentation systems. For example we have the numerical characterization in [6], the string (or tree) characterization in [8], a characterization based on Dempster-Shafer theory [15], and the algebra based characterization [23]. Others are surveyed in [8].

In terms of engineering the reasoning system, given the language $\mathcal{L}$ it should be sufficient to describe the application domain. The concept of argument and defeat that are selected should be such that the logical property of arguments and the defeat defined on them should be strong enough to capture sufficient conflicting patterns of information in the application at the level of arguments. In addition, the argumentation semantics that are selected should be able to produce a set of acceptable arguments that corresponds to the set of correct answers in the relevant application domain. In the following sections, we propose a mechanism to lay out the backbone of a shared argumentation reasoning system and build different conversation policies on top of it.

4 A Dialogue Mechanism

4.1 The Backbone Protocol

In this section, we define a flexible dialogue mechanism that decomposes a dialogue into a backbone protocol and a set of conversation policies. This dialogue mechanism serves a set of agents $\mathcal{T} = \{T_1, ..., T_n\}$ where each agent T_i is equipped with an information base $\Sigma(T_i)$. The set of conversation policies is a set of facilities, some of which are interrelated, to produce arguments and defeats out of the information base and feed them into the backbone protocol. The job of the backbone protocol is then to maintain a unified dialogue context of arguments and defeats between all the agents, and to provide an interface for the agents to query the public beliefs. In contrast to existing protocols, which typically enforce all the requirements on the structure of a conversation, the backbone protocol only assures the integrity and validity of arguments and defeats that have been exchanged, and leaves the other requirements of the dialogue to

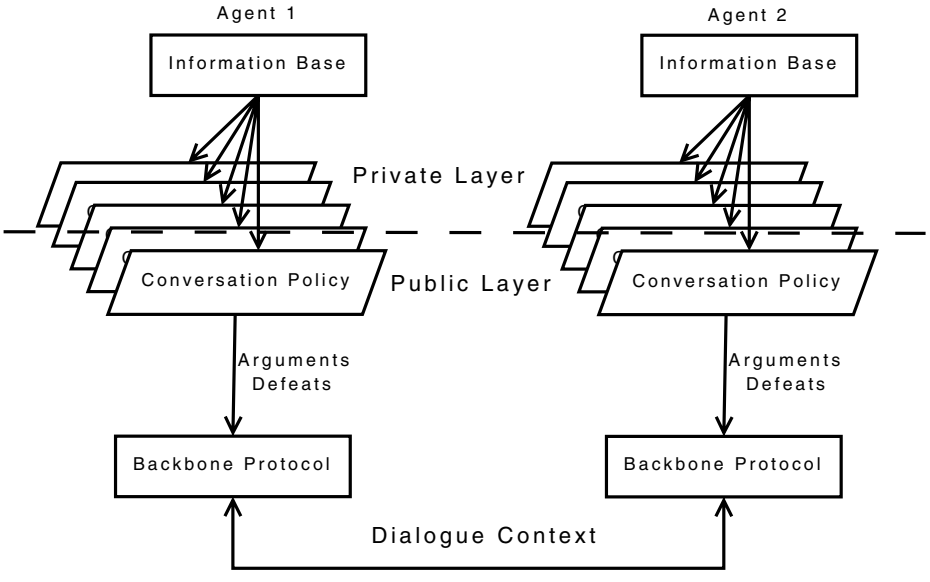


Fig. 1. A dialogue between two agents

conversation policies. As shown in Figure 1, the components of the mechanism can be divided into two layers — the public layer and the private layer — from the view of whether the components can be accessed and verified publicly by the the agents, and whether they require public cooperation among the agents to maintain their functions. The public layer is composed of a backbone protocol and a set of public conversation policies[1]; the private layer is composed of the agents' information bases and a set of private conversation policies. We will discuss conversation policies in section 4.2.

The backbone protocol organizes the messages exchanged by agents as a shared set of valid arguments and defeats, and then uses some agreed argumentation semantics to extract public beliefs from the messages. The prerequisites for using the protocol are that all the agents share the same language $\mathcal{L}$, share the same definition of arguments and defeats, and share the same argumentation semantics. The central notion in the backbone protocol is the *dialogue context*, denoted by C, the shared set of arguments and defeats as well as their supports. C is triple

$$\langle C_\Sigma, C_\mathcal{A}, C_\mathcal{R} \rangle$$

where C_Σ is the set of formulae that have been exchanged; $C_\mathcal{A}$is the set of arguments that have been identified; and $C_\mathcal{R}$ is the set of defeat relations that have been identified. For convenience, we also write $C = C_\Sigma \cup C_\mathcal{A} \cup C_\mathcal{R}$.

[1] We use the term "protocol" for the set of rules that governs the overall structure of a dialogue instance, and the term "conversation policy" for the set of rules that governs, possibly partially, the local structure a segment of a dialogue instance following [19]. However, in the literature, it is common to use the two terms interchangeably. See [17,19] for more discussion.

The implementation of the dialogue context will depend heavily on how conversations between the agents are organized. Here we assume that the configuration only allows pair-wise agent communication. Under this assumption, one of many ways to implement the dialogue context C distributively is to have each agent T_i maintain a copy $C^i = C^i_\Sigma \cup C^i_\mathcal{A} \cup C^i_\mathcal{R}$ of the context C, and regulate the agents to access and modify the context only through a set of pairwise communication locutions:

Definition 8. *Basic pairwise communication locutions:*

$send(T_i, T_j, \varphi)$
- *Precondition: none.*
- T_i *updates* $C^i_\Sigma = C^i_\Sigma \cup \{\varphi\}$.
- T_j *updates* $C^j_\Sigma = C^j_\Sigma \cup \{\varphi\}$.

$send(T_i, T_j, (H, h))$
- *Precondition:* $H \subseteq C^i_\Sigma$, $h \in C^i_\Sigma$, *and* (H, h) *is an argument according to Definition 1.*
- T_i *updates* $C^i_\mathcal{A} = C^i_\mathcal{A} \cup \{(H, h)\}$.
- T_j *updates* $C^j_\mathcal{A} = C^j_\mathcal{A} \cup \{(H, h)\}$.

$send(T_i, T_j, defeat((H, h), (H', h')))$
- *Precondition:* $(H, h) \in C^i_\mathcal{A}$ *and* $(H', h') \in C^i_\mathcal{A}$, *namely the arguments should already exist in the communication context, and* $defeat((H, h), (H', h'))$ *is a defeat according to Definition 3.*
- T_i *records updates* $C^i_\mathcal{R} = C^i_\mathcal{R} \cup \{(defeat((H, h), (H', h'))\}$.
- T_j *updates* $C^j_\mathcal{R} = C^j_\mathcal{R} \cup \{defeat((H, h), (H', h'))\}$.

$query(T_i, T_j, h)$
- *Precondition: None*
- T_i *asks himself and* T_j *to stop sending formulae, arguments, and defeats into the context.*
- *The agents compute simultaneously whether there is an argument* (H, h) *in the set of acceptable arguments* $Acc_{C^i_\mathcal{A}, C^i_\mathcal{R}}$ *according to the argumentation framework* $\langle C^i_\mathcal{A}, C^i_\mathcal{R} \rangle$.

Proposition 2. *The contents of* C^i *for* $i \in \{1, \ldots, n\}$ *are identical if the context is only manipulated by the locutions defined in Definition 8 .*

Proof. Immediate by construction.

The following are a set of macro locutions constructed from the pairwise locutions in Definition 8 and invoked as primitives (i.e. no other pairwise locutions can be invoked by any agent during each macro):

$send(T_i, \varphi)$: Invoke $Send(T_i, T_j, \varphi)$ for every agent T_j, $j \neq i$.
$send(T_i, (H, h))$: Invoke $Send(T_i, T_j, (H, h))$ for every agent T_j, $j \neq i$.
$send(T_i, defeat((H, h), (H', h')))$: Invoke $Send(T_i, T_j, defeat((H, h), (H', h')))$ for every agent T_j, $j \neq i$.
$query(T_i, h)$: Invoke $Query(T_i, T_j, h)$ for every agent T_j, $j \neq i$.

Proposition 3. *For each query* $query(T_i, h)$*, all the participating agents will obtain the same status for* h*.*

Protocol 4.1. A Backbone Protocol

Require: (1) each agent T_i is equipped with a dialogue context C^i, (2) each agent T_i is equipped with a conversation policy CP, (3) all the agents pre-agree on a language $\mathcal{L}$, an argumentation reasoning system AS and its semantics,

1: **repeat**

2: Query CP for an argument (H, h)

- Check whether (H, h) is a valid argument according to AS, if not go to next step
- Check whether $(H, h) \in C^i_{\mathcal{A}}$, if yes go to next step
- Invoke $Send(T_i, p)$ for each $p \in H \cup \{h\}$
- Invoke $Send(T_i, (H, h))$

3: Query CP for a $defeat(A_j, A_k)$

- Check whether A_j and A_k are valid arguments and $A_j, A_k \in C^i$, if not continue the loop
- Check whether $defeat(A_j, A_k)$ is a valid defeat, if not continue the loop
- Invoke $send(T_i, defeat(A_j, A_k))$

4: **until** $query(T_k, h)$ is posted to the dialogue by some agent T_k; after every agent gets the answer, resume the loop

Proof. The status for h evaluated by T_i solely depends on C^i. For all $i \in \{1, \ldots, n\}$, C^i is the same according to Proposition 2, and all T_is share the same semantics of argumentation, therefore all the agents will obtain the same status for h.

With these locutions, we get the backbone Protocol 4.1. This defines a set of prerequisites that all participating agents must satisfy before the execution of the protocol, and a loop of two execution steps: one to handle arguments and another one to handle defeats. The set of prerequisites is that all the agents must maintain a copy of the context, have a conversation policy CP (which will be defined below in section 4.2), and pre-agree on a language $\mathcal{L}$, an argumentation system AS and its semantics. In the body, the protocol loops through two steps: 1) query the conversation policy for an argument, check its validity, make sure it is new to the context, and then add it into the context by sending its component formulae and the (argument) structure explicitly over these formulae using locutions $send(T_i, \varphi)$ and $send(T_i, (H, h))$ respectively; 2) query the conversation policy for a defeat, check its validity, make sure that it is new to the context, and then add it into the context by sending its explicit structure using the locution $(T_i, A_j\ defeats\ A_k)$. The loop can be stopped at any time by any agent that needs the argument status of a belief represented by a formula h, then every agent will compute this status based on its own copy of the dialogue context. In this way, the protocol can guarantee that every agent will have the same answer for h.

4.2 Conversation Policies

In [11], the authors defined the concept of a conversation policy as a declarative specification that governs communication between software agents using an agent communication language. We agree with this notion of conversation policy in general, but as we

use the conversation policies on top of the backbone protocol defined in the previous section (which is actually also a conversation policy in this general definition), we will define our conversation policies as mechanisms that govern the production of arguments and defeats that are then fed into the backbone protocol. Different conversation policies capture different aspects of the communication between agents, and can be tailored to the needs of specific applications.

There are several aspects of a conversation policy that need to be dealt with in addition to the backbone protocol, including:

1. the source and mechanism from which the arguments and defeats are generated;
2. whether it is a private policy which only requires an individual effort or whether it is a public policy which requires cooperation among agents;
3. whether it is verifiable; and
4. whether it is concerned with general public argumentation, or with application-specific problem solving.

In this paper, we only have room to deal with some of these dimensions.

Given two policies CP_1 and CP_2 we can combine them in the following ways, reminiscent of those suggested for dialogue game protocols in [17,19].

- sequential: return the arguments (respectively, defeats) produced by CP_1 first; when no more arguments (defeats) can be produced by CP_1, then return those of CP_2
- alternate: one argument or defeat from CP_1, then one from CP_2, continuing to alternate until no further arguments or defeats can be produced.
- filtering: filter the arguments and defeats from CP_1 and CP_2 with respect to some criteria
- preference selection: compare two arguments or defeats obtained from CP_1 and CP_2 respectively, then select one of them according to some preference.

5 Example Policies

5.1 A Basic Policy

Policy 5.1 is a basic policy, concerned with the general process of public argumentation. It generates arguments and defeats using the reasoning mechanism of $\mathcal{L}$, and it requires no cooperation among agents. It is not possible to verify whether an agent conforms with this policy by analyzing what the agent puts into the dialogue context. Since the criteria to select a formula h from I is unspecified, there is no guarantee that the arguments and defeats put into the dialogue are complete enough to generate a stable argument overall for the topic of interest in terms of the selected argumentation semantics and certainly no guarantee that this will occur within a given amount of time. We can, however, improve on the basic policy.

5.2 Iterative Deepening Dialogue

Policy 5.2 is an improvement on the basic policy. It will still generate arguments and defeats based on the information in agents' information bases and using the reasoning

Policy 5.1. A basic conversation policy

Require: (1) a set of topics of interest $\{h_i\}$ shared by all the agents or held by individual agent, (2) each agent T_i is equipped with a dialogue context C^i and an information base $\Sigma(T_i)$, (3) all the agents pre-agree on a common language $\mathcal{L}$ and an argumentation system AS and its semantics,

1: Initialize $I = \{h_i\}$ and maintain a memory of I during the dialogue
2: On request for an argument,

- if I is empty, return nil
- select a formula h from I,
- construct an argument $A = (H, h) \notin C^i_{\mathcal{A}}$ based on $\Sigma(T_i) \cup C^i_{\Sigma}$ according to the proof theory of $\mathcal{L}$. If such an argument exists, then return it; otherwise return nil
- if all possible arguments for h have been exhausted, let $I = I - \{h\}$

3: Internally decide the defeating points: Select an argument $A = (H, h) \in \Sigma_{\mathcal{A}}$, select a formula $p \in H \cup \{h\}$, let $I = I \cup \{\neg p\}$ if $\neg p \notin I$.
4: On request for a defeat, look for two arguments $A_1, A_2 \in C^i_A$ such that $A_1\ defeats\ A_2 \notin C^i_{\mathcal{A}}$, if such a defeat exists then return it; otherwise return nil

mechanism of the language $\mathcal{L}$, but it does this by generating arguments and defeats in a specific order in the spirit of iterative deepening search. The policy uses three search parameters to limit the resources that the agents can use to generate arguments and defeats: (1) reasoning depth R_D, which controls the maximum number of inference rules that can be used in building arguments, (2) defeat depth D_d, which controls the maximum number of defeats that may be chained together[2], (3) reasoning breadth R_B, which controls the maximum number of arguments an agent can provide for a conclusion. To apply the policy, we will need the agents to synchronize these parameters cooperatively (if not, this will become a specific case of Policy 5.1). This means that this conversation policy is a public one.

The advantage of this policy is that, since it is an exhaustive search through the arguments and defeats that the set of agents can generate, then if there is a set of acceptable arguments that can be distilled from the set of all arguments that each agent can construct on its own (namely $\cup_i \mathcal{A}(\Sigma(T_i))$, the iterative deepening search policy will reach this set after a finite number of iterations.

Using a similar scheme of maintaining some shared parameters, a more efficient policy than Policy 5.2 can be created based on AND-OR tree evaluation to decide whether an argument is acceptable. If Dung's grounded semantics is used, and the argumentation system is finitary — for every argument there are only finite number of defeat arguments — then we have the same policy as used in [3]). This policy is of polynomial complexity in terms of the number of arguments[3]. Alternatively, if we employ the argument schemes and defeat schemes approach of [27], we may be able to tailor the

[2] A defeat chain takes the form of argument A_1 defeats A_2 which defeats A_3.

[3] This does not imply that the whole argumentation process is polynomial in the size of the information base, Σ. The overall time complexity is dependent on checking the validity of an argument which, unless the reasoning mechanism is restricted in some way, will not in general be polynomial.

Policy 5.2. An iterative deepening dialogue policy

Require: (1) A set of topics: $S = \{h_j\}$, (2) a set of agents T_i each with dialogue context C^i, (3) a pre-agreed reasoning depth increment Δ_R, (4) a pre-agreed defeat depth increment Δ_D, (4) a pre-agreed reasoning breadth increment Δ_B.

1: Initially set reasoning depth $R_D = 1$, reasoning breadth R_B = 1, defeat depth $D_d = 1$,
2: Initially set interest points $I = \{h_i\}$
3: Initially set defeat points $D = \emptyset$
4: On request for an argument

- if I is empty, return nil
- select a formula h from I,
- construct an argument $A = (H, h)$ based on $\Sigma(T_i) \cup C^i_\Sigma$ such that
  - $A \notin C^i_\mathcal{A}$
  - A uses at most R_D of the inference rules of $\mathcal{L}$
  - Other than A, there are less than R_B arguments supporting h
- if all possible arguments for h have been exhausted, let $I = I - \{h\}$ if such an argument exists, then return it; otherwise return nil

5: On request for a defeat, look for two arguments $A_1, A_2 \in C^i_A$ such that

- $A_1\ defeats\ A_2 \notin C^i_\mathcal{A}$, and
- the length of any defeat path ended with A_1 is less than D_d,

if such a defeat exists then return it; otherwise return nil

6: Internally decide the defeating points: Select an argument $A = (H, h) \in \Sigma_\mathcal{A}$ such that the length of any defeat path ending with A is less than D_d, select a formula $p \in H \cup \{h\}$, let $I = I \cup \{\neg p\}$ if $\neg p \notin I$.
7: If all the agents can not produce more arguments and defeats, they cooperatively increase the parameters: (1) reasoning depth $R_d \leftarrow R_d + \Delta_R$, (2) reasoning breadth $R_B \leftarrow R_B + \Delta_B$, (3) defeat depth $D_d \leftarrow D_d + \Delta_D$, (4) set interest points back to $I = \{h_i\}$

language to generate a polynomial number of arguments for a specific domain, and then in total we will have a polynomial policy in terms of the number argument schemes and defeat schemes.

5.3 Constructing Arguments Cooperatively

If we use Policies 5.1 and 5.2 then there will be some arguments and defeats that can not be constructed. These are arguments that are constructed using information that is held by different agents, and so is not all available to any single agent. Some of these arguments *might* be constructed by Policies 5.1 and 5.2 — the necessary information being revealed by other arguments that the agents put forward — but there is no guarantee that this will be the case. In general we will need some mechanism to help the agents construct arguments cooperatively, especially in information seeking, inquiry and deliberation dialogues [31]. The following are the basic constructs for this purpose. To better organize the policy, we decompose some primitive functions of the policy as those in the backbone Protocol 4.1. We need the agents to cooperatively maintain a set

Policy 5.3. A policy to construct arguments cooperatively

Require: Requirements are those in Policy 5.1 or those in the Policy 5.2, and in addition that all the agents cooperatively maintain C_G

1: Function as Policy 5.1 or Policy 5.2, with the additional two steps:
2: For a formula $h \in I$, for which the agent cannot construct an argument, use backward chaining to obtain a proof for h, then select an open formula φ (i.e. not in $C_\Sigma \cup \Sigma(T_i)$) in the proof, and invoke $ask\_help(T_i, \varphi)$
3: Select a $\varphi \in C_G$ such that $\varphi \in \Sigma(T_i)$ but $\varphi \notin C_\Sigma$, invoke $offer\_help(T_i, \varphi)$

of goals C_G, the set of goals waiting for additional information (we can think of this as part of the dialogue context). The content of C_G is different from C_Σ in the sense that it is not the information held by any participating agents, but rather a set of symbols indicating the intention of asking agents to provide information. C_G is maintained by the following locutions:

- $ask\_help(T, \varphi)$
  - precondition: $\varphi \notin \Sigma(T)$ and $\varphi \notin C_\Sigma$,
  - T updates $C_G = C_G \cup \{\varphi\}$.
- $offer\_help(T, \varphi)$
  - precondition: $\varphi \in C_G$, and $\varphi \in \Sigma(T)$
  - T updates $C_\Sigma = C_\Sigma \cup \{\varphi\}$
  - T updates $C_G = C_G - \{\varphi\}$

With this set of locutions we can define Policy 5.3. This policy can be used to gain and provide help in constructing arguments, and works by delegating all the other functions to policies like those we discussed above. In Policy 5.3, we do not specify the conditions under which the agents can ask for help and should offer help. Such conditions will be application specific, and will result in specializations of Policy 5.3 that are used in specific situations.

5.4 A Policy for Multiagent Planning

Our final example, Policy 5.4, is a conversation policy that is application specific, and deals with multiagent planning. The policy handles part of the generation of the arguments and defeats using its knowledge about specific problem — formalized in terms of state transitions, plans, resource conflicts and resource reconfiguration — and delegates the other functions to Policy 5.1 or 5.2.

To demonstrate the policy, we consider a simple multiagent planning problem, concerning two agents T_1 and T_2. Making common assumptions from the AI planning literature [21], both of them characterize the world as a set of precisely observable states S; T_1 and T_2 are capable of performing two sets of actions, A_1 and A_2 respectively. The evolution of the world is modeled as three mutually exclusive state transition functions

$$\gamma_1 : S \times A_1^* \rightarrow S$$
$$\gamma_2 : S \times A_2^* \rightarrow S$$
$$\gamma_3 : S \times (A_1 - A_1^*) \times (A_2 - A_2^*) \rightarrow Sn$$

where $A_1^* \subseteq A_1$ and $A_2^* \subseteq A_2$, γ_1 models the state transitions which can be totally controlled by T_1, γ_2 models the state transitions which can be totally controlled by T_2, and γ_3 models the state transitions which can only be controlled cooperatively by the two agents. Two sets of states $G_1 \subseteq S$ and $G_2 \subseteq S$ express the goals of T_1 and T_2 respectively. We denote the set of all possible state transitions as

$$\begin{aligned}\Gamma = &\{(s, a, s') | \gamma_i(s, a) = s' \text{ with } i = 1, 2\} \\ &\cup \{(s, a_1, s_2, s') | \gamma_3(s, a_1, a_2) = s'\}\end{aligned}$$

A plan p is a pair $\langle \Gamma_p, \pi_p \rangle$ where $\Gamma_p \subseteq \Gamma$ is a set of interesting state transitions and $\pi_p \subseteq S \times A$ is a set of state-actions pairs. π_p is a policy, in the planning sense, prescribing what action to take in each state encountered. We assume that there is a carefully designed language $\mathcal{L}$ such that one argument type is a pair $\langle (\Gamma_p, \pi_p), G \rangle$ where Γ_p and π_p together characterize a plan p, and G characterizes the set of states that can be experienced by the policy π_p. Another argument type is pair of the form $\langle H, \neg(s, a) \rangle$ or $\langle H, \neg(s, a_1, a_2) \rangle$ which means that H is a set of formulae in the language that characterize the resource conflicts which prevent the action a or cooperative action pair (a_1, a_2) from being performed in the state s, and one more argument type is a pair of the form $\langle H, (s, a) \rangle$ or $\langle H, (s, a_1, a_2) \rangle$ which means that H characterizes a configuration of resources which enables the action a or (a_1, a_2) in the state s. Therefore we have two types of defeat: (1) an argument $\langle H, \neg(s, a) \rangle$ defeats another argument $\langle (\Gamma_p, \pi_p), G \rangle$ when $(s, a) \in \pi_p$, and (2) an argument $\langle H, (s, a) \rangle$ or $\langle H, (s, a_1, a_2) \rangle$ defeats another argument $\langle H, \neg(s, a) \rangle$. For simplicity, we assume that nothing can defeat the argument of the form $\langle H, (s, a) \rangle$ or $\langle H, (s, a_1, a_2) \rangle$. Namely, we are assuming that the information about the state transitions that is taken as input to the dialogue system is consistent with respect to the language $\mathcal{L}$ being used.

We further assume that the set of all possible arguments is $\mathcal{A}$. There is a set of arguments $A_G \subseteq \mathcal{A}$ such that all the arguments in A_G have the same conclusions $G_1 \cup G_2$. For every argument $a \in A_G$, there is an argument in $b \in A_C$ with conclusions saying that some action in a cannot be performed because of resource conflict; there is subset $A'_C \subset A_C$ such that for every argument $a \in A'_C$ there is an argument $b \in A_R$ saying there is a resource configuration which will make sure that the action in a's conclusion can be performed. Therefore the set of acceptable arguments A_{acc} is the set of all $a \in A_G$ such that if there is some $b \in A_C$ that defeats a then another $c \in A_R$ defeats b. In this setting, A_{acc} corresponds to the set of plans that can achieve the two agents' goals G_1 and G_2 simultaneously. Assume that T_i $(i = 1, 2)$ can construct four nonempty sets of arguments $A_{G,i}, A_{C,i}, A'_{C,i}, A_{R,i}$ (it is possible that two sets of the same argument type from different agents intersect) where:

$$\begin{aligned}A_G &= A_{G,1} \cup A_{G,2} \\ A_C &= A_{C,1} \cup A_{C,2} \\ A_C &= A'_{C,1} \cup A'_{C,2} \\ A_R &= A_{R,1} \cup A_{R,2}\end{aligned}$$

If all the arguments only use a finite number of inferences in the language $\mathcal{L}$, then the backbone protocol 4.1 and the deepening search conversation policy 5.2 can collect the set of A_{acc} in the dialogue at some point. After that point, the two agents

Policy 5.4. A planning policy

Require: Requirements are those of Policy 5.1 or 5.2

1: Function as Policy 5.1 or 5.2, with the following modified request handler:
2: On request for an argument

- if there is an argument a from $A_{R,i}$ (assume there is an additional mechanism to detect this with errors), return a
- if there is an argument b from $A_{G,i}$ (assume there is an additional mechanism to detect this with errors), return b
- otherwise function as Policy 5.1 or 5.2.

enter the stable state in which no matter what the two agents say the set of acceptable arguments, and in turn the acceptable plans, will not change. With this kind of domain knowledge, the agent can employ Policy 5.4. In the policy, two additional mechanisms are used to generate the arguments from $A_{R,i}$ and $A_{G,i}$ as early as possible so that the system can reach the stable state as soon as possible. As we can see in the policy, we allow errors in the policy for generating these arguments, but it won't greatly affect the outcome of the dialogue for two reasons: 1) the argumentation semantics will still characterize the major part of the acceptable arguments as acceptable on the fly if the errors are restricted to a small range, and 2) the iterative deepening conversation policy will eventually generate these arguments using the inference power of $\mathcal{L}$ although it may take a long time for the dialogue to reach the stable stage.

As we can see, the major effort in coming up with a dialogue for multiagent planning involves:

1. representing the problem domain in the language $\mathcal{L}$ and the problem solving schemes in terms of the chosen argumentation system; and
2. identifying shortcuts to generate the most important arguments so as to have the dialogue reach the stable state as soon as possible.

In this way, the dialogue becomes much easier to engineer than the other dialogue approaches for similar problems [5,13,28] in which the problem domain, the problem solving schemes, the underlying logic, the dialogue moves, the dialogue protocols, and the dialogue conversation policies need to be considered all together.

6 Responses to Design Desiderata

There are two well known, overlapping, sets of dialogue design desiderata in the literature, those proposed by Maudet and Chaib-Draa [17] and those proposed by McBurney *et al.* [20]. This section briefly compares our framework against these. Each numbered passage describes how our framework compares with the identically numbered desideratum in the relevant proposal.

6.1 Response to McBurney *et al.*

1) In response to stated dialogue purpose, our mechanism does not impose any restriction on the purpose of the dialogue, different applications can choose their purposes freely by agreeing on a set of interest points represented in the language. 2) In response to the need for diversity of individual purposes, different agents can have different points of interest and these will be subjected to public argumentation to resolve conflicts. 3) In response to the need for inclusiveness, our mechanism allows any agent to participate into the dialogue, and how new agents contribute to the dialogue will depend on the quality of the arguments they can make with respect to the public argumentation semantics. 4) In response to transparency, our mechanism decomposes the functions so that as many components as possible are public. An agent's commitments to the external world should be subject to public argumentation. 5) In response to fairness, our mechanism advocates fairness in terms of the public argumentation semantics: every agent can influence the outcome if it can provide good arguments. 6) In response to the clarity of argumentation theory, most of the argumentation theory is captured explicitly by the shared public argumentation semantics. 7) In response to the separation of syntax and semantics, the semantics of the dialogue is mainly defined by the public argumentation semantics. The conversation policies and the backbone protocols are independent of the semantics. 8) In response to rule-consistency, our mechanism in practice allows any conversation policy, but the backbone protocol will rule out invalid arguments and defeats, and the argumentation semantics will further rule out ultimately defeated arguments to maintain the consistency of the public belief set. 9) In response to encouragement of resolution, our mechanism can output results at any time. Whether the status of the public belief benefits a given agent at that time will depend on the quality of the arguments it has put into the dialogue context. This can be viewed as an incentive to encourage agents to provide the best arguments to resolve conflicts. 10) In response to discouragement of disruption, the argumentation semantics prevent behaviors such as repeatedly uttering the same argument from having an effect on public beliefs. 11) In response to the enablement of self-transformation, there are two aspects. From the viewpoint of an agent influencing public beliefs, all agents are allowed to change their opinions or preference using different arguments — whether these changes will be sanctioned by the set of public beliefs will depend on how good an argument the agent can make for its most recent interest. We leave work on the question of how an agent changes its views to fit in with the set of public beliefs for future work (but see [22]). 12) In response to system simplicity, the decomposition of the dialogue mechanism helps to modularize and thus simplify the major components. 13) In response to computational simplicity, our mechanism allows the private and public conversation polices to be made efficient in order to have public argumentation reach the stable state as early as possible.

6.2 Response to Maudet and Chaib-Draa

In [17], the authors consider their desiderata under the headings of flexibility and specification.

First we consider the desiderata relating to flexibility. 1) In response to the call for a formalism which allows more flexibility than a finite state machine (which is the

mechanism used to specify a number of dialogues), we don't restrict the description form of conversation policies: they can be procedural, declarative, and even object oriented programming style, while the only strict restriction is the form of arguments and defeats in the backbone protocol, then the argumentation semantics actually replace the role of some complex specification constructs in the traditional dialogue protocol, such as backtracking. 2) In response to permitting unexpected messages within conversation policies, we employ a different philosophy: no strict regulation on what to say, but if the agent says something which is defeatable, the argumentation semantics will rule those points out, so in practice, there are no unexpected messages but only effective or ineffective messages. 3) In response to the need to be able to compose conversation policies, we allow conversation policies to be composed in much the same way that the control structures of procedural programming languages allow the commands of such languages to be composed. The effect of these compositions is defined by the argumentation semantics on the dialogue context. 4) In response to the need for a means to reach an agreement on a conversation policy, it is possible in our mechanism to have the agents represent their concerns about the conversation policies in a similar way to that introduced in Section 5.4, and then have them reach an agreement on a conversation policy (a plan) using the conversation policy for multiagent planning defined in Section 5.4.

Next we turn to the desiderata relating to specification. 1) In response to the need for public specifications, we decompose the functions of the mechanism to make as many components public as possible — the public conversation policies, the backbone protocol, and the public argumentation semantics are publicly known, the private conversation policies are the only private parts. 2) In response to adopting a declarative approach, our mechanism does not prevent the adoption of conversation policies that are defined declaratively, but we also think that a clear functional decomposition of the dialogue is as good a way to make the mechanism clear as adopting a declarative approach. 3) In response to exhibiting properties of conversation policies, we believe that there are good ways to do this even though the policies themselves are defined here in a procedural way. As with program verification, we can create a formal description of our procedural policies and verify their properties using model checking. 4) In response to optimizing conversation policies, there is nothing in our approach that rules this out. Individual agents can employ their own private policies to produce important arguments as early as possible based on their knowledge of specific problems. A similar approach can be used for public conversation policies, but the agents need to jointly use their shared knowledge of specific problems to generate the important arguments.

7 Conclusions

In this paper, we propose a flexible dialogue mechanism built on top of a public argumentation system. In this mechanism, we decompose the functions of dialogue into two parts — a backbone protocol which maintains the dialogue context regarding the set of public beliefs, and a set of conversation policies which handle the other aspects of the dialogue regarding the application and further regulation of the public argumentation. In this way, we are free to choose different argumentation theories to maintain the set of

public beliefs, we can incrementally construct and combine conversation policies for the computation of argumentation semantics as well as making effective arguments for the specific applications without concerning the other parts of the dialogue. The publicly accessible part of the dialogue, that is the backbone protocol and the public conversation policies, are in general verifiable because they are using only publicly available information. The private part of the dialogue is open for an individual agent to choose with respect to their individual needs. In this way, we balance between the need for public specification and verification and the need for flexibility.

In our mechanism, the idea of dialogue context shares some similarity with the common game board of Maudet [18], but instead of recording the dialogue moves, we record the content of the moves, the formulae, arguments and defeats, and apply a pre-agreed argumentation semantics on them to compute the status of arguments. In this way, we obtain a compact description of the interaction between the agents and a clear argumentation-theoretic interpretation of the dialogue outcomes. The reply structure proposed in Prakken [24] is close to our usage of argumentation in the dialogue context, but again Prakken applied argumentation to dialogue moves while we use the argumentation directly on the arguments and defeats that make up these moves.

There are a number of ways to extend this work. One future direction is to complete the mechanism with another set of private conversation polices which revise the agent's individual information base in the light of the set of public beliefs. The model used in [4] is a candidate for this set of revision conversation policies, a model that decides how to revise by looking at the status of the arguments in an argumentation system that uses a combination of public and private information. As the maintenance of the public dialogue context is costly, another future direction is to devise distributed algorithms and data structures to efficiently maintain public beliefs, especially in a cooperative society of agents, so that we can extend the dialogue mechanism to be a general multiagent coordination mechanism. A third direction that we want to pursue is to formally verify the properties of the backbone protocol and the conversation policies. In addition, more formal treatments of how to combine conversation policies similar to the approach of dialogue game protocols — such as those mentioned in [19] — will be needed, especially to formalize the way in which agents can reach an agreement on the public conversation policy that they will adopt. One candidate, already hinted at above, is to represent conversation policies as multiagent plans (plans in which the only actions are utterances) and then use the dialogue mechanism defined in Section 5.4 to create a meta-dialogue in which the agents come to agreement on the conversation polices they will adopt.

Acknowledgments

This work was partially supported by the US Army Research Laboratory and the UK Ministry of Defence under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

1. Amgoud, L.: A formal framework for handling conflicting desires. In: Nielsen, T.D., Zhang, N.L. (eds.) ECSQARU 2003. LNCS, vol. 2711, pp. 552–563. Springer, Heidelberg (2003)
2. Amgoud, L., Cayrol, C.: Inferring from inconsistency in preference-based argumentation frameworks. Journal of Automated Reasoning 29(2), 125–169 (2002)
3. Amgoud, L., Cayrol, C.: A reasoning model based on the production of acceptable arguments. Annals of Mathematics and Artificial Intelligence 34(1-3), 197–215 (2002)
4. Amgoud, L., Maudet, N., Parsons, S.: Modeling dialogues using argumentation. In: Proceedings of the Fourth International Conference on Multi-Agent Systems (2000)
5. Atkinson, K., Bench-Capon, T.J.M., McBurney, P.: A dialogue game protocol for multi-agent argument over proposals for action. In: Rahwan, et al. (eds.) [25], pp. 149–161.
6. Besnard, P., Hunter, A.: A logic-based theory of deductive arguments. Artifical Intelligence 128(1-2), 203–235 (2001)
7. Bonet, B., Geffner, H.: Arguing for decisions: A qualitative model of decision making. In: Proceedings of the 12th Annual Conference on Uncertainty in Artificial Intelligence (1996)
8. Cayrol, C., Lagasquie-Schiex, M.-C.: Graduality in argumentation. Journal of Artificial Intelligence Research 2005, 245–297 (2005)
9. Chesñevar, C., Maguitman, A., Loui, R.: Logical models of argument. ACM Computing Surveys 32(4), 337–383 (2000)
10. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77(2), 321–358 (1995)
11. Greaves, M., Holmback, H., Bradshaw, J.: What is a conversation policy? In: Issues in Agent Communication, London, UK, pp. 118–131. Springer, Heidelberg (2000)
12. Halpern, J.Y.: Reasoning about Uncertainty. MIT Press, Cambridge (2003)
13. Hitchcock, D., McBurney, P., Parsons, S.: The eightfold way of deliberation dialogues. International Journal of Intelligent Systems (2004)
14. Jakobovits, H., Vermeir, D.: Robust semantics for argumentation frameworks. Journal of Logic and Computation 9(2), 215–261 (1999)
15. Kohlas, J.: Probabilistic argumentation systems: A new way to combine logic with probability. Journal of Applied Logic 1(3-4), 225–253 (2003)
16. Krause, P., Ambler, S., Elvang-Gøransson, M., Fox, J.: A logic of argumentation for reasoning under uncertainty. Computational Intelligence 11, 113–131 (1995)
17. Maudet, N., Chaib-Draa, B.: Commitment-based and dialogue-game-based protocols: new trends in agent communication languages. Knowledge Engineering Review 17(2), 157–179 (2002)
18. Maudet, N., Evrard, F.: A generic framework for dialogue game implementation. In: Hulstijn, J. (ed.) Proceedings of the 2nd Workshop on Formal Semantics and Pragmatics of Dialogue, University of Twente, The Netherlands, pp. 185–198 (1998)
19. McBurney, P., Parsons, S.: Dialogue game protocols. In: Huget, M.-P. (ed.) Communication in Multiagent Systems. LNCS, vol. 2650, pp. 269–283. Springer, Heidelberg (2003)
20. McBurney, P., Parsons, S., Wooldridge, M.: Desiderata for agent argumentation protocols. In: Proceedings of the 1st International Conference on Autonomous Agents and Multiagent Systems (2002)
21. Nau, D., Ghallab, M., Traverso, P.: Automated Planning: Theory & Practice. Morgan Kaufmann Publishers Inc., San Francisco (2004)
22. Parsons, S., Sklar, E.: How agents revise their beliefs after an argumentation-based dialogue. In: Parsons, S., Maudet, N., Moraitis, P., Rahwan, I. (eds.) ArgMAS 2005. LNCS, vol. 4049, pp. 297–312. Springer, Heidelberg (2006)

23. Pollock, J.L.: Defeasible reasoning with variable degrees of justification. Artificial Intelligence 133(1-2), 233–282 (2001)
24. Prakken, H.: Coherence and flexibility in dialogue games for argumentation. Journal of Logic and Computation 15(6), 1009–1040 (2005)
25. Rahwan, I., Moraïtis, P., Reed, C. (eds.): ArgMAS 2004. LNCS, vol. 3366. Springer, Heidelberg (2005)
26. Rahwan, I., Ramchurn, S.D., Jennings, N.R., Mcburney, P., Parsons, S., Sonenberg, L.: Argumentation-based negotiation. The Knowledge Engineering Review 18(4), 343–375 (2003)
27. Reed, C., Walton, D.: Towards a formal and implemented model of argumentation schemes in agent communication. In: Rahwan, et al. (eds.) [25], pp. 19–30.
28. Tang, Y., Parsons, S.: Argumentation-based dialogues for deliberation. In: Proceedings of the 4th International Conference on Autonomous Agents and Multiagent Systems (2005)
29. Verheij, B.: Rules, Reasons, Arguments. Formal studies of argumentation and defeat. PhD thesis, University of Maastricht (1996)
30. Vreeswijk, G.: The feasibility of defeat in defeasible reasoning. In: Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning (1991)
31. Walton, D.N., Krabbe, E.C.W.: Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning. State University of New York Press, Albany (1995)
32. Weiss, G. (ed.): Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. MIT Press, Cambridge (1999)

Author Index