First International Workshop on the Theory and Applications of Formal Argumentation

(TAFA-11)

http://www.csd.abdn.ac.uk/ niroren/TAFA-11/Welcome.html

16-17 July 2011

Editors/Organisers:[1]

Sanjay Modgil, King's College London

Nir Oren, University of Aberdeen

Francesca Toni, Imperial College London

[1]TAFA-11 is endorsed by the Agreement Technologies COST action (http://www.agreement-technologies.eu/)

Accepted Papers

Full Presentations

- Pietro Baroni, Federico Cerutti, Paul Dunne and Massimiliano Giacomin. Computing with Infinite Argumentation Frameworks: the Case of AFRAs
- Ringo Baumann, Gerhard Brewka and Renata Wong. Splitting Argumentation Frameworks: An Empirical Evaluation
- Elizabeth Black and Katie Bentley. An empirical study of a deliberation dialogue system
- Richard Booth, Martin Caminada, Mikolaj Podlaszewski and Iyad Rahwan. Quantifying Disagreement in Argument-based Reasoning
- Xiuyi Fan and Francesca Toni. A First Step towards Argumentation Dialogues for Discovery
- Maria Laura Cobo, Diego Martinez and Guillermo Simari. Stable extensions in timed argumentation frameworks
- Alan Perotti, Guido Boella, Dov Gabbay, Leendert Van Der Torre and Serena Villata. Conditional labelling for abstract argumentation
- Nicolás Rotstein, Nir Oren and Timothy Norman. Resource Boundedness and Argumentation
- Francesca Toni and Paolo Torroni. Bottom-up argumentation

Short Presentations

- Stefano Bistarelli and Francesco Santini. A Constraint-based Computational Framework for Argumentation Systems
- Wolfgang Dvorak. On the Complexity of Computing the Justification Status of an Argument
- Jenny Eriksson Lundström. A Goal-Oriented Dynamical Computation of Preference of Strategy
- Sebastian Gottifredi, Alejandro Garcia and Guillermo Simari. Argument Types and Typed Argumentation Frameworks
- Patrick Krümpelmann, Matthias Thimm, Marcelo A. Falappa, Alejandro J. Garcia, Gabriele Kern-Isberner and Guillermo R. Simari. Selective Revision by Deductive Argumentation
- Hengfei Li, Nir Oren and Timothy Norman. Probabilistic Argumentation Frameworks
- Paulo Maio and Nuno Silva. A Three-Layer Argumentation Framework
- Rolando Medellin-Gasque, Katie Atkinson, Peter Mcburney and Trevor Bench-Capon. Arguments over Co-operative Plans

- Tjitze Rienstra, Alan Perotti, Serena Villata, Dov Gabbay, Leendert Van Der Torre and Guido Boella. Multi-sorted Argumentation
- Luke Riley, Katie Atkinson, Terry Payne and Elizabeth Black. An Implemented Dialogue System for Inquiry and Persuasion
- Alice Toniolo, Timothy J. Norman and Katia Sycara. Argumentation Schemes for Policy-Driven Planning
- Wietske Visser, Koen Hindriks and Catholijn Jonker. An Argumentation Framework for Qualitative Multi-Criteria Preferences

Panel Discussion

TAFA-11's panel session includes three senior researchers in the area of argumentation:

- Martin Caminada (Université du Luxembourg)
- Stefan Woltran (Vienna University of Technology)
- Carlos Chesnevar (Universidad Nacional del Sur)

The panellists will address and debate (with one another and the workshop participants) some of the questions listed below:

Q1 Which main challenges do we need to face for argumentation theory to have a real impact on applications?

Q2 Are any of the argumentation systems currently available ready for deployment?

Q3 Have we identified suitable "killer" applications already? If not, which direction should we look at for a "killer" application?

Q4 Do we need any further theoretical developments to pave the way towards applications and if so in which direction?

Q5 Which "industry" is most likely to be receptive to our methodologies/techniques?

Q6 Would it be useful to "team up" with any other field (in AI, or Computing, or elsewhere) in order to have a higher impact/more powerful

techniques?

Programme Committee

- Leila Amgoud, IRIT, Toulouse, France
- Katie Atkinson, University of Liverpool, UK
- Pietro Baroni, University of Brescia, Italy
- Floris Bex, University of Dundee, UK
- Elizabeth Black, Universiy of Utrecht, Netherlands
- Guido Boella, Universita di Torino, Italy
- Ivan Bratko, University of Ljubljana, Slovenia
- Gerhard Brewka, University of Leipzig, Germany
- Martin Caminada, University of Luxembourg, Luxembourg
- Carlos Chesnevar, Universidad Nacional del Sur, Argentina
- Sylvie Doutre, University of Toulouse 1, France
- Phan Minh Dung, Asian Institute of Technology, Thailand
- Paul Dunne, University of Liverpool, UK
- Dov Gabbay, KingŠs College London, UK
- Massimilliano Giacomin, University of Brescia, Italy
- Tom Gordon, Fraunhofer FOKUS, Germany
- Anthony Hunter, Univeristy College London, UK
- Antonis Kakas, University of Cyprus, Cyprus
- Nicholas Maudet, Universite Paris Dauphine, France
- Peter McBurney, University of Liverpool, UK
- Sanjay Modgil, King's College London, UK
- Pavlos Moraitis, Paris Descartes University, France
- Timothy J. Norman, University of Aberdeen, UK
- Nir Oren, University of Aberdeen, UK
- Simon Parsons, City University of New York, USA
- Henry Prakken, Utrecht University, and University of Groningen, The Netherlands
- Iyad Rahwan, Masdar Institute, UAE, and Massachusetts Institute of Technology, USA
- Chris Reed, University of Dundee, UK

- Nicolas Rotstein, University of Aberdeen, UK
- Guillermo Simari, Universidad Nacional del Sur, Argentina
- Francesca Toni, Imperial College, London, UK
- Serena Villata, University of Turin, Italy
- Simon Wells, University of Dundee, UK
- Stefan Woltran, TU Vienna, Institute of Information Systems, Austria
- Leon van der Torre, University of Luxembourg, Luxembourg

1 Preface

Recent years have witnessed a rapid growth of interest in formal models of argumentation and their application in diverse sub-fields and domains of application of AI, including reasoning in the presence of inconsistency, non-monotonic reasoning, decision making, inter-agent communication, the semantic web, grid applications, ontologies, recommender systems, machine learning, neural networks, trust computing, normative systems, social choice theory, judgement aggregation and game theory, and law and medicine. Argumentation thus shows great promise as a theoretically-grounded tool for a wide range of applications.

This workshop aims at contributing to the realisation of this promise, by promoting and fostering uptake of argumentation as a viable AI paradigm with wide ranging application, and providing a forum for further development of ideas and the initiation of new and innovative collaborations. The workshop has therefore invited submissions of papers[1] on: formal theoretical models of argumentation and application of such models in (sub-fields of) AI; evaluation of models, both theoretical (in terms of formal properties of existing or new formal models) and practical (in concretely developed applications); theories and applications developed through inter-disciplinary collaborations.

With the above aims and intended impact in mind, the workshop includes an extended panel session in which leading researchers in argumentation address the topic: 'The future of argumentation: what is its added value and how we communicate this to researchers in the Artificial Intelligence community and beyond.'

[1] Post-proceedings will be published in Springer, LNAI.

Computing with Infinite Argumentation Frameworks: the Case of AFRAs

Pietro Baroni[1], Federico Cerutti[1], Paul E. Dunne[2], Massimiliano Giacomin[1]

[1] Dipartimento di Ingegneria dell'Informazione, University of Brescia, via Branze, 38, 25123, Brescia, Italy
{pietro.baroni,federico.cerutti,massimiliano.giacomin}@ing.unibs.it
[2] Department of Computer Science, Ashton Building, University of Liverpool, Liverpool, L69 7ZF, United Kingdom
P.E.Dunne@liverpool.ac.uk

Abstract. In recent years a large corpus of studies has arisen from Dung's seminal abstract model of argumentation, including several extensions aimed at increasing its expressiveness. Most of these works focus on the case of finite argumentation frameworks, leaving the potential practical applications of infinite frameworks largely unexplored. In the context of a recently proposed extension of Dung's framework called AFRA (Argumentation Framework with Recursive Attacks), this paper makes a first step to fill this gap. It is shown that, under some reasonable restrictions, infinite frameworks admit a compact finite specification and that, on this basis, computational problems which are tractable for finite frameworks may preserve the same property in the infinite case. In particular we provide a polynomial-time algorithm to compute the finite representation of the (possibly infinite) grounded extension of an AFRA with infinite attacks. An example concerning the representation of a moral dilemma is introduced to illustrate and instantiate the proposal and gives a preliminary idea of its potential applicability.

1 Introduction

Infinite argumentation frameworks, though encompassed by Dung's theory of abstract argumentation [5], have received relatively limited attention in the literature so that their use as a modelling tool and the relevant computational issues are largely unexplored.

This paper provides a first step towards filling this gap, by considering the case of existence of infinite attacks in a recently proposed extension of Dung's framework called AFRA (*Argumentation Framework with Recursive Attacks*) [2] where "attacks" may themselves be attacked by arguments. The idea of encompassing attacks to attacks in abstract argumentation framework has been first considered in [3], and subsequently investigated and developed, for instance, in [2, 8, 10]. Computational issues in this kind of extended frameworks have been first addressed in [7] for the finite case of EAF [10]. In this paper, we show that, under some mild restrictions, an AFRA with infinite attacks can be represented

through a deterministic finite automaton (DFA), which provides the basis for the efficient solution of semantics-related computational problems. To demonstrate this, we show in particular that a DFA representing the (possibly infinite) grounded extension of an AFRA with infinite attacks can be derived in polynomial time from the DFA representing the AFRA itself.

From a general perspective, the ultimate aim of this paper is to provide an enabling technique for practical applications of infinite argumentation frameworks. While this is a largely open issue, we illustrate the theoretical concepts developed throughout the paper using a preliminary example concerning moral dilemma representation. Of course, the value of the methodology goes beyond both the simple example at hand and the use of the AFRA framework. Indeed the main contribution of this paper is twofold: on one hand, we address the topic of representing an Argumentation Framework through a formal language; and, secondly, we show that this kind of representation can be useful to compute semantics extensions also in the case of infinite Argumentation Frameworks.

The paper is organized as follows. After recalling the preliminary background concepts in Sect. 2, we provide an example encompassing infinite attacks in Sect. 3 and discuss specification mechanisms for AFRAs with infinite attacks in Sect. 4. Section 5 describes the actual specification mechanism adopted in the paper, called DFA$^+$, and Sect. 6 provides a polynomial time algorithm to compute the (representation of) the (possibly infinite) grounded extension of an AFRA starting from its DFA$^+$ specification. Finally Sect. 7 concludes the paper. Some proofs are omitted due to space limitation.

2 Preliminary Background

In this section we define the abstract argumentation models which are the core focus of this article: the AF model [5] with a finite set of arguments and the AFRA model [2].

Definition 1 *An* argumentation framework *(AF) is a pair* $\langle \mathcal{X}, \mathcal{A} \rangle$*, in which* $\mathcal{X}$ *is a finite set of* arguments *and* $\mathcal{A} \subseteq \mathcal{X} \times \mathcal{X}$ *is the* attack relationship*. A pair* $\langle x, y \rangle \in \mathcal{A}$ *is referred to as 'y* is attacked by *x' or 'x* attacks *y'; x* $\in \mathcal{X}$ *is* acceptable with respect to $S \subseteq \mathcal{X}$ *if for every* $y \in \mathcal{X}$ *that attacks x there is some* $z \in S$ *that attacks y. The* characteristic function*,* $\mathcal{F} : 2^{\mathcal{X}} \rightarrow 2^{\mathcal{X}}$ *is the mapping which, given* $S \subseteq \mathcal{X}$*, returns the set of* $y \in \mathcal{X}$ *for which y is acceptable to S. For any set S we define* $\mathcal{F}^0(S) = \emptyset$ *and for* $k \geq 1$ $\mathcal{F}^k(S) = \mathcal{F}(\mathcal{F}^{k-1}(S))$*. The* grounded *extension is the (unique) least fixed point of* $\mathcal{F}$*. We denote by* $GE(\langle \mathcal{X}, \mathcal{A} \rangle) \subseteq \mathcal{X}$ *the grounded extension of* $\langle \mathcal{X}, \mathcal{A} \rangle$*.*

Definition 2 *An* Argumentation Framework with Recursive Attacks *(AFRA) is described by a pair* $\langle \mathcal{X}, \mathcal{R} \rangle$ *where* $\mathcal{X}$ *is a (finite) set of arguments and* $\mathcal{R}$ *consists of pairs of the form* $\langle x, \alpha \rangle$ *where* $x \in \mathcal{X}$ *and* $\alpha \in \mathcal{X} \cup \mathcal{R}$*. For* $\alpha = \langle x, \beta \rangle \in \mathcal{R}$*, the* source *(src) and* target *(trg) of* α *are defined by* $src(\alpha) = x$ *and* $trg(\alpha) = \beta$*. In order to avoid a surfeit of brackets, we describe elements of* $\mathcal{R}$ *as finite length* sequences *of arguments, so that* $x_k\ x_{k-1}\ x_{k-2}\ \cdots\ x_2\ x_1 \in \mathcal{R}$

if $\{x_1, \ldots, x_k\} \subseteq \mathcal{X}$ *(note that an argument may occur* more than once *in this sequence),* $\langle x_2, x_1\rangle \in \mathcal{R}$ *(i.e.* $x_2x_1 \in \mathcal{R}$*) and* $\langle x_j\langle x_{j-1}\ \langle\cdots\ x_1\rangle\rangle\rangle \in \mathcal{R}$*, with* $2 < j \leq k$*. Letting* $\mathcal{C} = \mathcal{R} \cup \mathcal{X}$*, for* $\alpha \in \mathcal{R}$ *and* $\beta \in \mathcal{C}$*,* α *is said to* defeat β *(*$\alpha \rightarrow \beta$*) whenever any of the following hold:*

1. $trg(\alpha) = \beta$
2. $trg(\alpha) = src(\beta)$ *with* $\beta \in \mathcal{R}$*,* $\alpha = xy$ *and* $\beta = y\gamma$ *(*$y \in \mathcal{X}$*).*

By considering the (Dung) style AF, $\langle\tilde{\mathcal{X}}, \tilde{\mathcal{R}}\rangle$ constructed from an AFRA $\langle\mathcal{X}, \mathcal{R}\rangle$ by $\tilde{\mathcal{X}} = \mathcal{X} \cup \mathcal{R}$ and $\tilde{\mathcal{R}} = \{\langle\alpha, \beta\rangle \ : \ \alpha \rightarrow \beta\}$ [2], a correspondence between semantics structures (e.g. the basic notions of conflict-free and admissible sets and the extensions of various semantics) in an AFRA $\langle\mathcal{X}, \mathcal{R}\rangle$ and the analogous (Dung style) structures within $\langle\tilde{\mathcal{X}}, \tilde{\mathcal{R}}\rangle$ is obtained. In particular we will exploit the fact that the grounded extension of an AFRA denoted as $GE^{\text{AFRA}}(\langle\mathcal{X}, \mathcal{R}\rangle)$ coincides with the (Dung style) grounded extension of the corresponding AF $GE(\langle\tilde{\mathcal{X}}, \tilde{\mathcal{R}}\rangle)$.

3 An Example: Moral Dilemmas

The recursive form of $\mathcal{R}$ in an AFRA, $\langle\mathcal{X}, \mathcal{R}\rangle$, in principle, admits the capability of describing infinite attack structures even though $\mathcal{X}$ is a finite set. To exemplify the potential utility of this kind of structures as a modelling tool we consider a case of moral dilemma.

Fred is the network administrator of a large company and among his duties he has to release emails, addressed to staff members, that have been accidentally blocked by the security filters. One day he gets a helpdesk request from Eve, a staff member and his best friend's wife, requesting the release of an email. As part of the procedure he has to ensure that the email is safe by scanning its contents. He finds out that it's actually an email addressed to Eve from her lover. He releases the email, and his initial reaction is to call his friend up and tell him about the affair. However, the law forbids him to reveal the information. This is a case of conflict of obligations, and, following [4, 1], we can model this situation with abstract argumentation[3].

First of all, the reasons for the alternative actions can be represented as practical arguments [11]. Indeed, since Fred wants to be a good friend, then he should tell his friend what he knows (**T**), but since Fred wants to be a good citizen, then he should not (**D**). These two arguments are obviously attacking each other. Moreover, both **D** and **T** are related to values [4], respectively Legality and Friendship. These values can be represented as arguments (**L** and **F**) [11, 1] which affect the evaluation of the two practical arguments. For instance, in the case at hand, the value of Friendship would resolve the dilemma by making **T**

[3] A detailed comparison of alternative argumentation-based approaches to practical reasoning is beyond the scope of this paper. The interested reader may refer to [2] for a comparison between AFRA and Modgil's EAF, or to [1] for an illustration of the modelling approach adopted in the example.

prevail over **D**. From an argumentation point of view, this means that **F** would allow **T** to defeat **D**. This can be modelled by making ineffective the attack from **D** to **T** (α in Fig. 1) by attacking it through an attack (β) whose source is the value of Friendship **F**. This can be read as: even if the attack from **D** to **T** (α) holds because **D** and **T** support conflicting actions, nevertheless, in the case at hand, α is undermined by the moral commitment of **F**. Obviously, **L** states a similar moral commitment, namely making **D** prevail over **T**. This requires **L** to undermine both the attack (γ) from **T** to **D** and the attack (β) from **F** against α. In turn **F** should again make ineffective the latter attacks (δ and η) and **L** and **F** will continue attacking each other's attacks forever. This infinite construction reveals an unresolved dilemma.

Finally, let us suppose that Fred chooses to pursue Legality rather than Friendship. This can be represented by another argument (**M**) (a "must" argument in the terminology of [1]). The argument **M** represents a choice between values in the case at hand. Therefore, **M** will undermine any moral commitment of **F** over the two actions **T** and **D** by attacking the (infinite number of) attacks whose source is **F**.

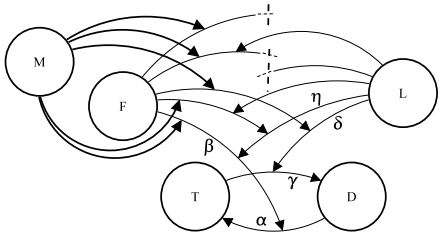


Fig. 1. Fred's dilemma

An AFRA representing Fred's dilemma is shown in Fig. 1. It consists of a finite set of arguments $\mathcal{X}_F = \{\mathbf{D}, \mathbf{T}, \mathbf{L}, \mathbf{F}, \mathbf{M}\}$ and of an infinite set of attacks $\mathcal{R}_F = \{\mathbf{DT}, \mathbf{TD}, \mathbf{LTD}, \mathbf{FDT}, \mathbf{FLTD}, \mathbf{LFDT}, \mathbf{LFLTD}, \mathbf{FLFDT}, \ldots, \mathbf{MFDT}, \mathbf{MFLTD}, \mathbf{MFLFDT}, \ldots\}$.

4 Representing $\mathcal{R}$ in AFRAs

Given the potential practical interest in AFRAs with infinite attacks, the following question arises.

> When $\mathcal{R}$ is infinite what characterises suitable specification mechanisms for describing $\mathcal{R}$?

In order to pursue this question, we need some terminology.

Definition 3 *For $\mathcal{X}$ a finite set of arguments, we denote by $\mathcal{X}^*$ the set of all finite length* sequences *(or* words*) that can be formed using arguments in $\mathcal{X}$*

(noting this includes ε the so-called empty sequence *comprising no arguments). Given $w \in \mathcal{X}^*$, $|w|$ denotes its* length*, i.e. the number of arguments occurring in its definition. Note that* repetitions *of the same arguments contribute to $|w|$ so that, e.g. $|x_1x_2x_1| = 3$ (and not 2). Given $w \in \mathcal{X}^*$ we will denote as $\bar{w}$ the sequence obtained by reversing the order of the symbols in w, namely, given $w = x_1x_2 \dots x_n$, $\bar{w} = x_n \dots x_2x_1$.*

Given $u = u_1u_2 \dots u_r$ and $v = v_1v_2 \dots v_k \in \Sigma^*$ we denote by $u \cdot v$ (or simply uv) the word w of length $k + r$ defined by $u_1u_2 \dots u_rv_1v_2 \dots v_k$. We note that $w \cdot \varepsilon = \varepsilon \cdot w = w$. We say that $\mathcal{L} \subseteq \mathcal{X}^*$ is an attack language over $\mathcal{X}$ *if $\mathcal{L}$ satisfies, $\forall\ w \in \mathcal{L}$ $w = xu$ with $x \in \mathcal{X}$ and either $|u| = 1$ or $u \in \mathcal{L}$.*

If $\mathcal{L}$ is an attack language over $\mathcal{X}$, then the pair $\langle \mathcal{X}, \mathcal{L} \rangle$ certainly describes an AFRA. Classical formal language and computability theory, see e.g. [9], provides a means of capturing the vague concept of "specification mechanism" via *Formal Grammars* and their associated machine models. As well known, given a set of symbols Σ a *formal grammar* G specifies the derivation of a language $L(G) \subseteq \Sigma^*$ called *language generated by G*. A language, $L \subseteq \Sigma^*$, is *recognisable* if there is a formal grammar G for which $w \in L$ if and only if $w \in L(G)$.

As a starting point for "specification mechanisms" for attack languages we can consider descriptions which are formal grammars (so that $\Sigma = \mathcal{X}$ in such cases).

Unsurprisingly, arbitrary attack languages have unhelpful computational properties.

Proposition 1 *Let $\mathcal{X} = \{x, y\}$. There are attack languages, L, over $\mathcal{X}$ which are not recognisable, i.e. for which there is no formal grammar G for which $L(G) = L$.*

Proof. In view of the correpondence from the fact that $L \subseteq \Sigma^*$ is *recursively enumerable* if and only if there is an unrestricted grammar, G such that $L(G) = L$, it suffices to show that there are attack languages which fail to be r.e. First recall that any TM program, M, can be associated with a finite length *codeword*, $\beta(M)$, (over the alphabet $\{0, 1\}$) in such a way that given $\beta(M)$ the behaviour of M can be reproduced by another TM program. Furthermore, the language corresponding to the set of valid encodings, i.e. $CODE = \{w \in \{0,1\}^* : w = \beta(M)$ for some TM program, $M\}$ is recursive.[4] With such encodings it is known that the language $L^{\varepsilon}_{\neg HALT} \subset \{0,1\}^*$ given by $\{\beta(M)$: The TM program, M, **fails to halt** given the empty word as input$\}$ is not r.e.

Now since $CODE \subset \{0,1\}^*$ we can *order* the set of all TM programs simply by ordering words[5] within $\{0,1\}^*$, so that the "first" TM program (M_1) is the first word, w_1 in this ordering of $\{0,1\}^*$ for which $w_1 \in CODE$, the "second" program (M_2) the second word, w_2 in the ordering for which $w_2 \in CODE$, and so on.

[4] See e.g. [6, Ch. 4] or any standard introductory text on computability, such as [9, Ch. 8.3].

[5] For example using the standard lexicographic ordering under which $0 <_{\text{lex}} 1$ and $u <_{\text{lex}} w$ whenever $|u| < |w|$.

We are now ready to define a suitable attack language, $\mathcal{R} \subset \{x, y\}^*$ establishing the proposition's claim: $\mathcal{R} = \{ xy^k : k \geq 2 \text{ and } M_k \in L^{\varepsilon}_{\neg HALT} \} \cup \{y^n : n \geq 2\}$. This is easily seen to be an attack language[6] and, furthermore, cannot be r.e. For suppose, $\mathcal{R}$ is r.e. with AL a TM accepting exactly the words in $\mathcal{R}$ then $L^{\varepsilon}_{\neg HALT}$ could be shown r.e. as follows: given $\beta(M)$ determine the index k for which M is the k'th TM program. Then $\beta(M) \in L^{\varepsilon}_{\neg HALT}$ if and only if xy^k is accepted by AL. □

As a consequence of Propn. 1 there will be attack languages for which it is not possible to present any specification (as a formal grammar). Of course the nature of such languages is unlikely to be of practical concern: Propn. 1 merely establishes a technical limitation affecting attack languages but certainly does not invalidate their use. In practice we would wish to consider only attack languages that are presented in some "verifiable form". What is the notion of "verifiable form" intended to capture here? Presented with a formal grammar G, there are two immediate issues which we would like to ensure can be addressed.

Q1. How easily can it be verified that $L(G)$ *does* describe an attack language?
Q2. Assuming $L(G)$ is verified as describing *some* attack language, $\mathcal{R}$ over $\mathcal{X}$, given $\alpha \in \mathcal{X}^*$ how easily can it be decided whether α *is* an attack in $\langle \mathcal{X}, \mathcal{R} \rangle$, i.e. whether $\alpha \in L(G)$?

It can be easily derived from Rice's Theorem (see, e.g. [9, pp. 185–195]) that unrestricted grammars face problems with respect to Q1.

Proposition 2 *Given an unrestricted grammar G, the problem of determining if $L(G)$ is an attack language is undecidable.*

On the other hand the family of *regular languages* [9] provides the basis for a positive result, using automata as representation mechanism.

Definition 4 *A* deterministic finite automaton *(*DFA*) is defined via a* 5*-tuple, $M = \langle \Sigma, Q, q_0, F, \delta \rangle$ where $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$ is a finite set of input symbols, $Q = \{q_0, q_1, \ldots, q_m\}$ a finite set of* states*; $q_0 \in Q$ the* initial *state; $F \subseteq Q$ the set of* accepting *states; and $\delta : Q \times \Sigma \rightarrow Q$ the* state transition *function. For $q \in Q$ and $w \in \Sigma^*$, the* reachable state *from q on input w is*

$$\rho(q, w) = \begin{cases} q \text{ if } w = \varepsilon \\ \delta(q, w) \text{ if } |w| = 1 \\ \delta(\rho(q, u), x) \text{ if } w = u \cdot x \end{cases}$$

A sequence $w = w_1 w_2 \ldots w_n \in \Sigma^*$ is accepted by the DFA *$\langle \Sigma, Q, q_0, F, \delta \rangle$ if $\rho(q_0, \bar{w}) = \rho(q_0, w_n w_{n-1} \ldots w_1) \in F$, i.e. the sequence of states (consistent with the state transition function δ) which processes every symbol in w* in reverse order *ends in an accepting state. For a* DFA*, $M = \langle \Sigma, Q, q_0, F, \delta \rangle$, $L(M)$ is the subset of Σ^* accepted by M.*

[6] The reader concerned by the fact that this includes a self-attacking argument (y) may note that we may use $xy^k x$ and $y^n x$ ($n \geq 1$) to achieve the same effect without self-attacking arguments.

Fact 1 *The language $L \subseteq \Sigma$ is regular if and only if there is a* DFA $M = \langle \Sigma, Q, q_0, F, \delta \rangle$ *for which* $L(M) = L$.

The following lemma shows that the conditions for an automaton to recognize an attack language are relatively simple.

Lemma 1. *Let $M = \langle \mathcal{X}, Q, q_0, F, \delta \rangle$ be a* DFA. *Then $L(M)$ is an attack language if and only if both the following conditions hold:*

C1. $\forall\ w \in \{\varepsilon\} \cup \mathcal{X}$, $\rho(q_0, w) \notin F$.
C2. $\forall\ q \in (Q \setminus \{q_0\})$, $\forall x\ \in \mathcal{X}$ *if* $q' = \delta(q, x) \notin F$ *then* $\forall w \in \mathcal{X}^*$ *it holds that* $\rho(q', \bar{w}) \notin F$.

Proof. Suppose first that $L(M)$ is an attack language. Since every $w \in L(M)$ satisfies $|w| \geq 2$ it is immediate that M satisfies C1. To see that C2 must hold, consider any $q \in (Q \setminus \{q_0\})$ and $x \in \mathcal{X}$ such that $q' = \delta(q, x) \notin F$. Furthermore consider any $u \in \mathcal{X}^*$ such that $q = \rho(q_0, \bar{u})$. Since $q \neq q_0$, $|u| \geq 1$ and, since $q' = \delta(q, x) \notin F$, $xu \notin L(M)$ and $|xu| \geq 2$. Since $L(M)$ is an attack language $\nexists p \in \mathcal{X}^*$ such that $p = vxu \in L(M)$, i.e. it is not possible to reach an accepting state from $q' = \delta(q, x)$.

For the converse direction, we show that if M satisfies both C1 and C2 then $L(M)$ is an attack language, i.e. $\forall\ w = xu \in L(M)$ either $|u| = 1$ or $u \in L(M)$. Since C1 holds, it is immediate that $|w| \geq 2$ for every $w \in L(M)$. Suppose now $w = yu \in L(M)$ with $|u| > 1$. Assume by contradiction $u \notin L(M)$, i.e. letting $q' = \rho(q_0, \bar{u})$ it holds $q' \notin F$. Since $|u| > 1$ it must be the case that $q' = \delta(q, x)$ for some x with $q \neq q_0$. By C2, this implies $\forall w \in \mathcal{X}^*\ \rho(q', \bar{w}) \notin F$ which contradicts $w = yu \in L(M)$, as this would entail $\delta(q', y) \in F$. □

The desired result in Theorem 1 follows directly from Fact 1 and Lemma 1.

Theorem 1 *Let $M = \langle \mathcal{X}, Q, q_0, F, \delta \rangle$ be a* DFA *defining the regular* language, *$L(M) \subseteq \mathcal{X}^*$. The problem of verifying that $L(M)$ is an attack language is polynomial time decidable.*

Proof. (outline) Given a DFA, $M = \langle \mathcal{X}, Q, q_0, \delta, F \rangle$ from Lemma 1, in order to verify that $L(M)$ is an attack language it suffices to confirm that M satisfies the conditions C1 and C2 and that these can tested in time polynomial in $|Q|$. □

5 The DFA$^+$ Representation of AFRAs

Expressing $\mathcal{R}$ within an AFRA, $\langle \mathcal{X}, \mathcal{R} \rangle$ via a DFA, M for which $L(M) = \mathcal{R}$ turns out to have some useful computational benefits in addition to verifiability and deciding whether a specified attack is present. We will demonstrate these advantages as far as the problem of computing the grounded extension is concerned. To this purpose we have first to introduce a representation of a whole AFRA (not just the attack relation) as an automaton and analyze its properties. Given an AFRA $\langle \mathcal{X}, \mathcal{R} \rangle$ where $\mathcal{R} \subset \mathcal{X}^*$ is a regular language represented as a DFA

$\mathcal{M} = \langle \mathcal{X}, Q_{\mathcal{M}}, q_0, F_{\mathcal{M}}, \delta \rangle$, it is easy to obtain a representation of $\langle \mathcal{X}, \mathcal{R} \rangle$ as a single DFA $\mathcal{M}^+ = \langle \mathcal{X}, Q_{\mathcal{M}^+}, q_0, F_{\mathcal{M}^+}, \delta^+ \rangle$ (indicated for the sake of brevity as DFA$^+$ in the following) such that for any $w \in \mathcal{X}^*$ it holds $w \in L(\mathcal{M}^+)$ if and only if $w \in \mathcal{X} \cup \mathcal{R}$. Let us notice that, in general, there are infinite DFA$^+$s representing a single AFRA. This may raise the problem of defining a canonical DFA$^+$ representation for each AFRA. This problem, not considered in the paper, is left for future work. In the following we will provide some general results that hold for any DFA$^+$ representing an AFRA.

Figure 2 shows $\mathcal{M}_F^+$, a DFA$^+$ which accepts all the words of the regular language $\mathcal{R}_F$ describing Fred's dilemma.

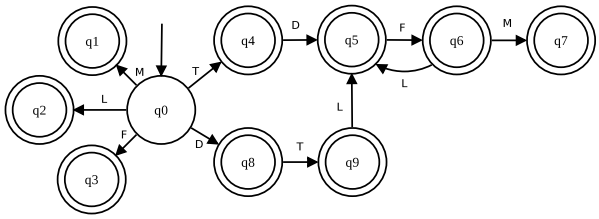


Fig. 2. A DFA$^+$ for Fred's dilemma (double circles represent accepting states)

We introduce also some handy notation concerning neighbor states and "input" symbols for a given state. For $p \in Q_{\mathcal{M}^+}$ we define $state-out(p) = \{ q \in Q_{\mathcal{M}^+} : \exists\, x \in \mathcal{X}$ for which $q = \delta^+(p,x)\}$. For instance, in $\mathcal{M}_F^+$, $state-out(q_0) = \{q_1, q_2, q_3, q_4, q_8\}$. For $p \in F_{\mathcal{M}^+}$ we define $sym-in(p) = \{x \in \mathcal{X} : \exists\, q \in Q_{\mathcal{M}^+}$ for which $p = \delta^+(q,x)\}$ and $state-in(p) = \{ q \in Q_{\mathcal{M}^+} : \exists\, x \in \mathcal{X}$ for which $p = \delta^+(q,x)\}$. In $\mathcal{M}_F^+$, $sym-in(q_5) = \{D, L\}$ and $state-in(q_5) = \{q_4, q_9, q_6\}$.

It is now useful to point out several properties of the DFA$^+$ representation (we will implicitly assume that each accepting state is reachable from q_0, as it should be in order to avoid useless parts in the automaton).

First we can partition the accepting states in $F_{\mathcal{M}^+}$ into two sets: *argument states* and *attack states*.

Argument states are in one-to-one correspondence with the elements of $\mathcal{X}$ and are reachable in one step from the initial state q_0: they represent the "additional part" of the DFA$^+$ w.r.t. the DFA representation. Formally $\forall x \in \mathcal{X}$ $\exists q \in F_{\mathcal{M}^+}$ such that $\delta^+(q_0, x) = q$ and $sym-in(q) = \{x\}$. For each $x \in \mathcal{X}$ we will denote the corresponding argument state as $argst(x)$ and, conversely, if $q = argst(x)$ we will say that $x = reparg(q)$. For the whole set of arguments $\mathcal{X}$ in a DFA$^+$ representation we define $ArgS(\mathcal{M}^+) \triangleq \{argst(x) \mid x \in \mathcal{X}\}$. Hence, $ArgS(\mathcal{M}_F^+) = \{q_1, q_2, q_3, q_4, q_8\}$.

In AFRA an argument can receive only direct defeats from other arguments: an argument x is defeated by an argument y if and only if $\langle x, y \rangle \in \mathcal{R}$ namely if the corresponding two-length string in $\mathcal{X}^*$ is accepted by the DFA$^+$ (and of

course by the original DFA). Formally we can identify the set of direct defeaters of an argument x as $dirdef(x) \triangleq \{y \in \mathcal{X} \mid \delta^+(argst(x), y) \in F_{\mathcal{M}^+}\}$. Of course an argument x is *unattacked* in AFRA if and only if $dirdef(x) = \emptyset$. The set of unattacked arguments will be denoted as $unatt - args(\mathcal{M}^+)$. The above definitions can be extended from arguments to argument states in the obvious way.

Attack states are all the accepting states which are not argument states and are defined as $AttS(\mathcal{M}^+) \triangleq F_{\mathcal{M}^+} \setminus ArgS(\mathcal{M}^+)$. Hence, $AttS(\mathcal{M}^+_F) = \{q_5, q_6, q_7, q_9\}$. Every attack state q in a DFA$^+$ (and in the original DFA) corresponds to a (possibly infinite) subset of $\mathcal{R}$, namely to a (nonempty) set of elements of the corresponding attack language, denoted as $AttL(q)$. Formally, for any $q \in AttS(\mathcal{M}^+)$ $AttL(q) \triangleq \{r \in \mathcal{R} \mid \rho(q_0, \bar{r}) = q\}$. Given $r \in AttL(q)$ we will say that q is the *representative state* of r, denoted as $q = repst(r)$. Of course, $\forall r \in \mathcal{R}\ \exists! q \in AttS(\mathcal{M}^+) \mid q = repst(r)$.

An element r of $\mathcal{R}$ can have both direct defeaters and indirect defeaters (see 1. and 2. in Def. 2). A direct defeater is any argument x which is the source of an attack whose target is r, it follows that $xr \in \mathcal{R}$. It can then be observed that given an attack state q all elements of $AttL(q)$ have the same direct defeaters. Formally, for any $q \in AttS(\mathcal{M}^+)$ we define $dirdef(q) \triangleq \{x \in \mathcal{X} \mid \delta^+(q, x) \in F_{\mathcal{M}^+}\}$ and for any $r \in \mathcal{R}$ $dirdef(r) \triangleq dirdef(repst(r))$.

An indirect defeater is any argument x which is the source of an attack whose target is the source of r: $indirdef(r) \triangleq dirdef(src(r))$.

Given an attack state q it can be noted that the source of any attack represented by q corresponds to one of the elements of $sym{-}in(q)$: in fact any element of $sym - in(q)$ is the first symbol of some of the elements of the attack language accepted by q. By extension, we can hence define the indirect defeaters of any $q \in AttS(\mathcal{M}^+)$: $indirdef(q) \triangleq \bigcup_{r \in AttL(q)} indirdef(r) = \bigcup_{x \in sym-in(q)} dirdef(x)$.

The whole set of defeaters of an element r of $\mathcal{R}$ will be denoted as $totdef(r) \triangleq dirdef(r) \cup indirdef(r)$. Analogously, for a state q, $totdef(q) \triangleq dirdef(q) \cup indirdef(q)$. We say that an attack state q is unattacked if $totdef(q) = \emptyset$. For instance in Fig. 2 q_7 is unattacked while $totdef(q_5) = \{\mathbf{F}, \mathbf{T}\}$. In the following we will use the term unattacked states to refer collectively to both unattacked argument states and unattacked attack states. It can be noted that if an attack state q is unattacked then all elements of $AttL(q)$ are unattacked, but it does not hold that if $r \in \mathcal{R}$ is unattacked then $repst(r)$ is unattacked. In fact $totdef(r) = \emptyset$ implies $dirdef(repst(r)) = \emptyset$ but does not imply $indirdef(repst(r)) = \emptyset$ since $repst(r)$ might have indirect defeaters due to other elements of $AttL(q)$. On the other hand it can easily be observed that $totdef(r) = \emptyset$ implies also $indirdef(repst(r)) = \emptyset$ if $|sym - in(repst(r))| = 1$. Under this condition $r \in \mathcal{R}$ is unattacked if and only if $repst(r)$ is unattacked.

Since this is a desirable property, we need to introduce a transformation of DFA$^+$ aimed at ensuring the above condition while leaving unmodified the accepted language. This will be achieved by *splitting* some attack states of the DFA$^+$.

Definition 5 *An attack state p is* splittable *if $|sym - in(p)| > 1$. The set of splittable states of a* DFA$^+$ *$\mathcal{M}^+$ will be denoted as $split - states(\mathcal{M}^+)$.*

In $\mathcal{M}^+_F$, q_5 is splittable since $sym-in(q_5) = \{L, D\}$.

As explained above we need a *complete split* (*csplit* in the following) operator whose goal is transforming a DFA$^+$ (without affecting the language it accepts) so that in the resulting DFA$^+$ there are no splittable states. This is achieved by adding, for each splittable state p, a number $|sym - in(p)| - 1$ new accepting states. Accordingly a *split* operation w.r.t a splittable state can be defined as follows:

Definition 6 *For $\mathcal{M}^+ = \langle \mathcal{X}, Q_{\mathcal{M}^+}, q_0, F_{\mathcal{M}^+}, \delta^+ \rangle$ let p be a splittable state with $sym - in(p) = \{x_1, \ldots, x_n\}$, $(n > 1)$. The* DFA$^+$ *resulting by* splitting *p, $split(\mathcal{M}^+, p) = \langle \mathcal{X}, Q^{\text{spl}}_{\mathcal{M}^+}, q_0, F^{\text{spl}}_{\mathcal{M}^+}, \delta^{+\text{spl}} \rangle$ is obtained by:*

S1. $Q^{\text{spl}}_{\mathcal{M}^+} = Q_{\mathcal{M}^+} \cup \{p_2, \ldots, p_n\}$ *where $p_2, \ldots, p_n$ are new (accepting) states hence included also in $F^{\text{spl}}_{\mathcal{M}^+}$.*

S2. Letting $p_1 = p$ the transition function $\delta^{+\text{spl}}$ has, for $i = 1 \ldots n$:
$\delta^{\text{spl}}(q', x_i) = p_i$ *if* $q' \in state - in(p) \wedge \delta(q', x_i) = p$
$\delta^{\text{spl}}(p_i, y) = \delta(p, y)$
$\delta^{\text{spl}}(q, y) = \delta(q, y)$ *if* $q \in Q_{\mathcal{M}^+} \setminus state - in(p)$

In words, a splittable state p is partitioned into several states p_i each with $sym-in(p_i) = \{x_i\}$ and the transitions from p to other states are replicated from each p_i to them. It can be observed that the application of the split operation:

- does not affect the language accepted by the DFA$^+$: for any splittable state p $L(\mathcal{M}^+) = L(split(\mathcal{M}^+, p))$;
- does not affect the cardinality of $sym - in(q)$ for any state $q \neq p$: in fact q may have additional incoming transitions from the elements p_i but they all correspond to elements already present in $sym - in(q)$;
- for each state p_i, letting x_i be the only element of $sym-in(p_i)$, in $split(\mathcal{M}^+, q)$ it holds that $dirdef(p_i) = dirdef(q)$ and $indirdef(p_i) = indirdef(x_i)$.

In virtue of the second point above, it can be noted that it is possible to extend the definition of the split operation to a set of splittable states: given a set P of splittable states of a DFA$^+$ $\mathcal{M}^+$, the result of the operation $split(\mathcal{M}^+, P)$ is the DFA$^+$ resulting from the application of $split(\mathcal{M}^+, p)$ for each $p \in P$ (the order of application of the operations $split(\mathcal{M}^+, p)$ does not matter).

Of course the *csplit* operation is obtained by applying the split operation to all splittable states of a DFA$^+$ $\mathcal{M}^+$: $csplit(\mathcal{M}^+) \triangleq split(\mathcal{M}^+, split{-}states(\mathcal{M}^+))$. It is easy to observe that the number of states of $split(\mathcal{M}^+, split - states(\mathcal{M}^+))$ is upper bounded by $|Q_{\mathcal{M}^+}| * |\mathcal{X}|$ hence the *csplit* operation can be carried out in polynomial time with respect to the number of states and arguments of $\mathcal{M}^+$.

Figure 3 depicts the result of the application of the *csplit* operator to $\mathcal{M}^+_F$. As we noticed before, q_5 is a splittable state (and it is the only one in $\mathcal{M}^+_F$). $csplit(\mathcal{M}^+_F)$ has an additional state w.r.t. $\mathcal{M}^+_F$, namely q'_5 with $sym - in(q'_5) = \{L\}$ while after splitting $sym - in(q_5) = \{D\}$. Moreover, as required by Def. 6, any outgoing transitions from the split state is replicated, giving rise to the transitions from q_5 to q_6 and from q'_5 to q_6, both triggered by F.

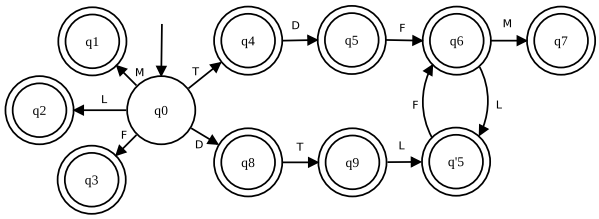


Fig. 3. Graphical representation of $csplit(\mathcal{M}_F^+)$

6 Computing the Grounded Extension with the DFA$^+$ Representation

In this section we show that the grounded extension of AFRAs with DFA$^+$ representation can be computed in polynomial time. Observe that, since the grounded extension of an AFRA includes both arguments and attacks, it may be infinite and therefore will, in turn, be expressed through a DFA$^+$ representation, algorithmically derived from the one of the AFRA.

Before illustrating the algorithm we need to consider some properties of AFRAs and of the grounded extension.

First recall a characterization of the grounded extension for finitary argumentation frameworks [5].

Definition 7 *An argumentation framework $\langle \mathcal{X}, \mathcal{A} \rangle$ is* finitary *iff for each argument x there are only finitely many arguments in $\mathcal{X}$ which attack x.*

Proposition 3 *If an argumentation framework* AF *is finitary then $GE(\text{AF}) = \bigcup_{i=1\ldots\infty} \mathcal{F}^i(\emptyset)$ where $\mathcal{F}$ is the characteristic function of* AF *(Def. 1).*

It is now easy to see that, for any AFRA, the corresponding AF $\langle \tilde{\mathcal{X}}, \tilde{\mathcal{R}} \rangle$ (see Sect. 2) is finitary:

- the attackers of each element x of $\tilde{\mathcal{X}} \cap \mathcal{X}$ correspond to the direct defeaters of x in AFRA, which are at most $|\mathcal{X}|$;
- the attackers of each element r of $\tilde{\mathcal{X}} \cap \mathcal{R}$ correspond to the direct and indirect defeaters of r in AFRA, which are at most $2 * |\mathcal{X}|$.

On this basis we can now state some relatively straightforward conditions concerning the membership of AFRA arguments and attacks to $GE(\langle \tilde{\mathcal{X}}, \tilde{\mathcal{R}} \rangle) = GE^{\text{AFRA}}(\langle \mathcal{X}, \mathcal{R} \rangle)$, drawing relations between the characteristic function and defeaters in the DFA$^+$ representation.

Proposition 4 *Let $\langle \mathcal{X}, \mathcal{R} \rangle$ be an* AFRA *with* DFA$^+$ *representation and $\langle \tilde{\mathcal{X}}, \tilde{\mathcal{R}} \rangle$ be its corresponding* AF *with characteristic function $\mathcal{F}$, x be an element of $\tilde{\mathcal{X}} \cap \mathcal{X}$, r be an element of $\tilde{\mathcal{X}} \cap \mathcal{R}$. It holds that:*

1. *$x \in \mathcal{F}^1(\emptyset)$ iff $dirdef(x) = \emptyset$*

2. $r \in \mathcal{F}^1(\emptyset)$ *iff* $totdef(r) = \emptyset$
3. *for* $i \geq 2, x \in \mathcal{F}^i(\emptyset) \setminus \mathcal{F}^{i-1}(\emptyset)$ *iff* $\forall y \in dirdef(x)\ (totdef(yx) \cap \mathcal{F}^{i-1}(\emptyset)) \neq \emptyset \wedge \exists y \in dirdef(x) \mid (totdef(yx) \cap \mathcal{F}^{i-2}(\emptyset)) = \emptyset$
4. *for* $i \geq 2, r \in \mathcal{F}^i(\emptyset) \setminus \mathcal{F}^{i-1}(\emptyset)$ *iff* $\forall y \in totdef(r)\ (totdef(yr) \cap \mathcal{F}^{i-1}(\emptyset)) \neq \emptyset \wedge \exists y \in totdef(r) \mid (totdef(yr) \cap \mathcal{F}^{i-2}(\emptyset)) = \emptyset$

We can now introduce Algorithm 1 which builds a DFA accepting the grounded extension of $\langle \mathcal{X}, \mathcal{R} \rangle$.

Algorithm 1 Determining $GE(\langle \mathcal{X}, \mathcal{R} \rangle)$ in AFRAs

1: **Input:** DFA$^+$ $\mathcal{M}^+ = \langle \mathcal{X}, Q_{\mathcal{M}^+}, q_0, F_{\mathcal{M}^+}, \delta^+ \rangle$ with $\alpha \in L(\mathcal{M}^+) \Leftrightarrow \alpha \in \mathcal{X} \cup \mathcal{R}$.
2: **Output:** DFA $\mathcal{M}_G = \langle \mathcal{X}, Q_G, q_0, F_G, \delta_G \rangle$ with $\alpha \in L(\mathcal{M}_G) \Leftrightarrow \alpha \in GE(\langle \tilde{\mathcal{X}}, \tilde{\mathcal{R}} \rangle)$
3: $i := 0$
4: $\mathcal{M}_i := csplit(\mathcal{M}^+)$; with $\mathcal{M}_i = \langle \mathcal{X}, Q_i, q_0, F_i, \delta_i \rangle$
5: **repeat**
6: $\quad i := i + 1$; $\mathcal{M}_i := \mathcal{M}_{i-1}$;
7: $\quad$ For each (unmarked) unattacked state q of $\mathcal{M}_i$ mark q as **in**(i).
8: $\quad$ **for** each unattacked state q and every $q' \in state - in(q) \cap F_i$ **do**
9: $\quad\quad$ Mark q' as **out** and remove q' from F_i.
10: $\quad$ **end for**
11: $\quad$ **for** each $x \in \mathcal{X}$ s.t. $argst(x)$ is marked **out do**
12: $\quad\quad$ For each state $q \in F_i$ with $x \in sym - in(q)$ mark q as **out** and remove q from F_i.
13: $\quad$ **end for**
14: **until** $\mathcal{M}_i = \mathcal{M}_{i-1}$
15: **for** any $q \in F_i$ which is not marked **in**() **do**
16: $\quad$ remove q from F_i
17: **end for**
18: **return** $\langle \mathcal{X}, Q_i, q_0, F_i, \delta_i \rangle$

Figure 4 shows the result of the execution of Alg. 1 on $\mathcal{M}_F^+$. After splitting, in the first iteration of the **repeat** cycle the unattacked states q_1, q_2, q_3, q_7 are marked **in**(1) (note that q_7 has no indirect defeaters since q_1 is unattacked). Then, since $state - in(q_7) = \{q_6\}$, q_6 is marked **out** and removed from the set of accepting states. As a consequence, during the second iteration, ${q'}_5$ is unattacked and is marked **in**(2). Then, q_9 is marked **out** at line 9 of Alg. 1 and removed from the set of accepting states. Finally, in the third iteration, both q_5 and q_8 are unattacked (note in particular that q_5 is unattacked since $argst(\mathbf{D}) = q_8$ is unattacked). As a consequence they are marked **in**(3) and q_4 is marked **out** at line 9 of Alg. 1. The algorithm will then terminate at the following iteration.

From an argumentation point of view, this result means that the arguments **M**, **L**, **F** and **D** are in the AFRA grounded extension, along with any attack whose source is one of **M**, **L**, and **D**. Therefore, the dilemma's solution is that Fred should not tell his friend what he knows, because in this situation the value of legality prevails over the value of friendship.

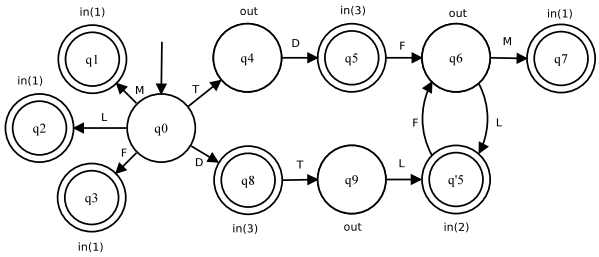


Fig. 4. The DFA$^+$ after the execution of Alg. 1 on $\mathcal{M}_F^+$

Turning back to technical results, correctness of Algorithm 1 follows from the following proposition.

Proposition 5 *Let $\mathcal{M}^+ = \langle \mathcal{X}, Q_{\mathcal{M}^+}, q_0, F_{\mathcal{M}^+}, \delta^+ \rangle$ with $\alpha \in L(\mathcal{M}^+) \Leftrightarrow \alpha \in \mathcal{X} \cup \mathcal{R}$ be a* DFA$^+$ *describing the* AFRA *$\langle \mathcal{X}, \mathcal{R} \rangle$, with corresponding* AF *$\langle \tilde{\mathcal{X}}, \tilde{\mathcal{R}} \rangle$, and $\mathcal{M}_i = \langle \mathcal{X}, Q_i, q_0, F_i, \delta_i \rangle$ the automaton produced by Algorithm 1 at the i-th iteration of the **repeat** cycle. For $i \geq 0$, let $T_i \subseteq F_i$ be the set of states*

$$T_i \quad = \quad \bigcup_{k=1}^{i} \quad \{ \; q \in F_i \; : \; q \textit{ is labelled } \textbf{\textit{in}}(k) \textit{ by Algorithm 1} \}$$

and $L_i = \{\alpha \in \mathcal{X}^* \; : \; \rho(q_0, \bar{\alpha}) \in T_i\}$. For every $i \geq 1$ $\alpha \in \mathcal{X} \cup \mathcal{R}$ is in L_i if and only if $\alpha \in \mathcal{F}_i(\emptyset)$, i.e. α is acceptable w.r.t. $\mathcal{F}_{i-1}(\emptyset)$ in $\langle \tilde{\mathcal{X}}, \tilde{\mathcal{R}} \rangle$.

On this basis we obtain one of the main results of the paper.

Theorem 2 *Let $\mathcal{M}^+ = \langle \mathcal{X}, Q_{\mathcal{M}^+}, q_0, F_{\mathcal{M}^+}, \delta^+ \rangle$ with $\alpha \in L(\mathcal{M}^+) \Leftrightarrow \alpha \in \mathcal{X} \cup \mathcal{R}$ be a* DFA$^+$ *describing the* AFRA*, $\langle \mathcal{X}, \mathcal{R} \rangle$ with corresponding* AF *$\langle \tilde{\mathcal{X}}, \tilde{\mathcal{R}} \rangle$. It is possible to construct in polynomial time a* DFA *$\mathcal{M}_G = \langle \mathcal{X}, Q_G, q_0, F_G, \delta_G \rangle$ with $\alpha \in L(\mathcal{M}_G) \Leftrightarrow \alpha \in GE(\langle \tilde{\mathcal{X}}, \tilde{\mathcal{R}} \rangle)$*

Proof. Given Prop. 5, we have only to show that Alg. 1 terminates in polynomial time. We have already commented that the *csplit* operation (l. 4) can be carried out in polynomial time and gives rise to a total number of states $\#Q \leq |Q_{\mathcal{M}^+}| * |\mathcal{X}|$. The **repeat** cycle terminates when $\mathcal{M}_i = \mathcal{M}_{i-1}$, which occurs when no unmarked unattacked states are detected at iteration i. Identifying whether a state q is unattacked requires the following checks (check (ii) only applies to attack states): (i) for any state $p \in state-out(q)$ is p in F_i? (ii) for any argument $x \in sym - in(q)$ is any defeater of x in F_i?

Check (i) requires at most $\#Q$ constant time operations for each state q, so its complexity in a single iteration of the **repeat** cycle is $O(\#Q^2)$. Check (ii) requires at most $|\mathcal{X}|^2$ constant time operations for each state q, so its complexity in a single iteration of the **repeat** cycle is $O(\#Q * |\mathcal{X}|^2)$.

Given the identification of unattacked states for granted, in a single iteration of the **repeat** cycle:

- at most $\#Q$ mark operation are executed at l. 7;
- at most $\#Q$ checks on membership to $state - in(q) \cap F_i$ are carried out at l. 8 and at most the same number of marking and removal operations are executed at l. 9;
- the **for** cycle at l. 11 is executed at most $|\mathcal{X}|$ times and for each of these iterations at most $\#Q$ marking and removal operations are executed at l. 12.

Noting that the algorithm never adds accepting states, it follows that the number of removals and, hence, the number of iterations of the **repeat** cycle is bounded by $\#Q$. Finally the **for** cycle at l. 15 is executed at most $\#Q$ times.

Summing up, the order of magnitude of the computational complexity of Alg. 1 is determined by checks (i) and (ii) within the **repeat** cycle, which turn out to be respectively $O(\#Q^3) = O(|Q_{\mathcal{M}^+}|^3 * |\mathcal{X}|^3)$ and $O(\#Q * \#Q * |\mathcal{X}|^2) = O(|Q_{\mathcal{M}^+}|^2 * |\mathcal{X}|^4)$. □

7 Conclusions

This paper proposes a methodology and provides some initial results in the largely unexplored field of computing with infinite argumentation frameworks, using as a starting point the possible existence of infinite attacks in the recently introduced AFRA formalism, exemplified by a case of moral dilemma. While other approaches (for instance, Modgil's EAF [10]) may provide a different formalization of this specific example, from a general point of view it is worth noting that the notion of unlimited recursive attacks, as in the AFRA formalism, may encompass infinite attack sequences even with a finite set of arguments. This can be easily seen as a finite alphabet able to describe infinite attack structures.

In fact, the proposal is built on the main idea of drawing correspondences between the specification of argumentation frameworks and well-known notions and results in formal language theory. While there are cases of infinite attacks which can not be represented with formal grammars, deterministic finite automata provide a convenient way to represent infinite attack relations with potential practical use. In particular we show that, with this representation, the problem of computing the grounded extension, which is tractable in the finite case, preserves its tractability in the infinite case. We are already extending this kind of analysis to other "standard" computational problems in abstract argumentation, like checking whether a set is conflict-free, is admissible or is a stable extension. The representation of special reasoning cases, like dilemmas, is an example of motivation for this kind of studies. In a similar spirit, one might consider the representation of dialogues where the repetition of previous moves is allowed: while this is normally forbidden, in order to ensure dialogue termination, the proposed approach might be used to define a sound semantics for some kinds of non-terminating dialogues, which represent the formal counterpart of situations where dialogue participants decide to keep (some of) their positions forever.

In the perspective of enlarging its applicability domain, the proposed methodology and techniques could also be applied to other cases of infinite frameworks,

either in the context of traditional Dung's AF or in some of its extended versions. In particular, it can be noted that the proposed approach implicitly deals with a family of infinite Dung's AFs since any AFRA with infinite attacks can be translated into a traditional AF with infinite arguments (see Sect. 2). From a more general perspective, one can consider using the DFA representation to specify an infinite set of arguments (so that each accepted word corresponds to an argument) complemented by a compact definition of the attack relation. Just to give an example, one simple option is to state that if both words xw and w are accepted (i.e. both of them represent arguments) then xw attacks w. In this way it is possible, for instance, to represent an infinite chain of attacks with a simple DFA, accepting the words x, xx, xxx, ... A variants of Algorithm 1 could then be devised to compute the grounded extension of this kind of frameworks.

A deep investigation of these issues represents the main direction of future work.

References

1. Baroni, P., Cerutti, F., Giacomin, M., Guida, G.: An argumentation-based approach to modeling decision support contexts with what-if capabilities. In: AAAI Fall Symposium. Technical Report SS-09-06. pp. 2–7 (2009)
2. Baroni, P., Cerutti, F., Giacomin, M., Guida, G.: AFRA: argumentation framework with recursive attacks. International Journal of Approximate Reasoning 51(1), 19–37 (2011)
3. Barringer, H., Gabbay, D., Woods, J.: Temporal dynamics of support and attack networks: From argumentation to zoology. In: Hutter, D., Stephan, W. (eds.) Mechanizing Mathematical Reasoning, Lecture Notes in Computer Science, vol. 2605, pp. 59–98. Springer Berlin / Heidelberg (2005)
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. 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)
6. Dunne, P.E.: Computability Theory – concepts and applications. Ellis–Horwood (1991)
7. Dunne, P.E., Modgil, S., Bench-Capon, T.: Computation in extended argumentation frameworks. In: Proceedings of 19th European Conference on Artificial Inteligence (ECAI 2010). pp. 119–124. Lisbon, P (2010)
8. Gabbay, D.: Semantics for higher level attacks in extended argumentation frames part 1: Overview. Studia Logica 93, 357–381 (2009)
9. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley (1979)
10. Modgil, S.: Reasoning about preferences in argumentation frameworks. Artificial Intelligence 173(9–10), 901–934 (2009)
11. Walton, D., Reed, C., Macagno, F.: Argumentation Schemes. Cambridge University Press, NY (2008)

Splitting Argumentation Frameworks: An Empirical Evaluation

Ringo Baumann, Gerhard Brewka, Renata Wong

Universität Leipzig, Johannisgasse 26, 04103 Leipzig, Germany,
baumann@informatik.uni-leipzig.de

Abstract. In a recent paper Baumann [1] has shown that splitting results, similar to those known for logic programs under answer set semantics and default logic, can also be obtained for Dung argumentation frameworks (AFs). Under certain conditions a given AF A can be split into subparts A_1 and A_2 such that extensions of A can be computed by (1) computing an extension E_1 of A_1, (2) modifying A_2 based on E_1, and (3) combining E_1 and an extension E_2 of the modified variant of A_2. In this paper we perform a systematic empirical evaluation of the effects of splitting on the computation of extensions. Our study shows that the performance of algorithms may drastically improve when splitting is applied.

1 Introduction

Dung's abstract argumentation frameworks (AFs) [3] are widely used in formal approaches to argumentation. They provide several standard semantics, each capturing different intuitions about how to handle conflicts among (abstract) arguments. This makes them a highly useful tool in argumentation (see for instance Prakken's ASPIC [6] for a typical way of using AFs) and algorithms for computing extensions have received considerable interest.

In a recent paper, Baumann [1] has shown that splitting results, similar to those known for logic programs under answer set semantics [4] and default logic [8], can also be obtained for Dung argumentation frameworks. It turns out that under certain conditions a given AF A can be split into subparts A_1 and A_2 such that the computation of extensions of A can be divided into smaller subproblems: to compute an extension of A one has to (1) compute an extension E_1 of A_1, (2) modify A_2 based on E_1, and (3) combine E_1 and an extension E_2 of the modified variant of A_2.

Given these results, the obvious question is: does splitting an AF really pay off in practice? In this paper we aim to give an empirical answer to this question. We do this by systematically comparing the behavior of an algorithm for computing extensions with and without splitting. Our study shows that the performance of the algorithm indeed may drastically improve when splitting is applied.

Our evaluation is based on an implementation of Caminada's labelling algorithm [5], arguably the standard genuine algorithm for computing extensions.

We focus on preferred and stable semantics. We also include results for grounded semantics, but as this semantics is known to be polynomial an improvement of performance here was never expected, and our results confirm this.

The paper is organized as follows: we start in Sect. 2 with the necessary background on AFs, labellings, splitting and strongly connected components. We then describe in Sect. 3 the algorithms used in our evaluation. Sect. 4 contains the empirical evaluation and thus the main results of the paper. Sect. 5 concludes.

2 Background

2.1 Argumentation frameworks

An *argumentation framework* $\mathcal{A}$ is a pair (A, R), where A is a non-empty finite set whose elements are called *arguments* and $R \subseteq A \times A$ a binary relation, called the *attack relation*.

In the following we consider a fixed countable set $\mathcal{U}$ of arguments, called the *universe*. Quantified formulae refer to this universe and all denoted sets are finite subsets of $\mathcal{U}$ or $\mathcal{U} \times \mathcal{U}$ respectively. Furthermore we will use the following abbreviations. Let $\mathcal{A} = (A, R)$ be an AF, B and B' subsets of A and $a \in A$. Then

1. $(B, B') \bar{\in} R \Leftrightarrow_{def} \exists b \exists b' : b \in B \wedge b' \in B' \wedge (b, b') \in R$,
2. a is defended by B in $\mathcal{A} \Leftrightarrow_{def} \forall a' : a' \in A \wedge (a', a) \in R \rightarrow (B, \{a'\}) \bar{\in} R$,
3. B is conflict-free in $\mathcal{A} \Leftrightarrow_{def} (B, B) \bar{\notin} R$,
4. $cf(\mathcal{A}) = \{C \,|\, C \subseteq A, C \text{ conflict-free in } \mathcal{A}\}$.

The set of all extensions of $\mathcal{A}$ under semantics $\mathcal{S}$ is denoted by $\mathcal{E}_{\mathcal{S}}(A)$. We consider the classical semantics introduced by Dung, namely stable, preferred, complete and grounded (compare [3]).

Definition 1. *Let $\mathcal{A} = (A, R)$ be an AF and $E \subseteq A$. E is a*

1. *admissible extension*[1] *($E \in \mathcal{E}_{ad}(\mathcal{A})$) iff*
   $E \in cf(\mathcal{A})$ and each $a \in E$ is defended by E in $\mathcal{A}$,
2. *complete extension ($E \in \mathcal{E}_{co}(\mathcal{A})$) iff*
   $E \in \mathcal{E}_{ad}(\mathcal{A})$ and for each $a \in A$ defended by E in $\mathcal{A}$, $a \in E$ holds,
3. *stable extension ($E \in \mathcal{E}_{st}(\mathcal{A})$) iff*
   $E \in \mathcal{E}_{co}(\mathcal{A})$ and for every $a \in A \backslash E$, $(E, \{a\}) \bar{\in} R$ holds,
4. *preferred extension (i.e. $E \in \mathcal{E}_{pr}(\mathcal{A})$) iff*
   $E \in \mathcal{E}_{co}(\mathcal{A})$ and for each $E' \in \mathcal{E}_{co}(\mathcal{A})$, $E \not\subset E'$ holds,
5. *grounded extension ($E \in \mathcal{E}_{gr}(\mathcal{A})$) iff*
   $E \in \mathcal{E}_{co}(\mathcal{A})$ and for each $E' \in \mathcal{E}_{co}(\mathcal{A})$, $E' \not\subset E$ holds.

[1] Note that it is more common to speak about admissible sets instead of the admissible semantics. For reasons of unified notation we used the less common version.

2.2 Labelling-based Semantics

The labelling approach [2, 5] provides an alternative possibility to describe extensions. Given an AF $\mathcal{A} = (A, R)$, a labelling is a total function $L : A \to \{in, out, undec\}$. We use $x(L)$ for $L^{-1}(\{x\})$, i.e. $x(L) = \{a \in A \mid L(a) = x\}$. This allows to rewrite a labelling L as a triple $(in(L), out(L), undec(L))$ which is frequently used. Analogously to $\mathcal{E}_{\mathcal{S}}(\mathcal{A})$ we write $\mathcal{L}_{\mathcal{S}}(\mathcal{A})$ for the set of all labellings prescribed by semantics $\mathcal{S}$ for an AF $\mathcal{A}$.

Definition 2. *Given an AF $\mathcal{A} = (A, R)$ and a labelling L of $\mathcal{A}$. L is called a complete labelling ($L \in \mathcal{L}_{co}(\mathcal{A})$) iff for any $a \in A$ the following holds:*

1. *If $a \in in(L)$, then for each $b \in A$ s.t. $(b, a) \in R$, $b \in out(L)$,*
2. *If $a \in out(L)$, then there is a $b \in A$ s.t. $(b, a) \in R$ and $b \in in(L)$,*
3. *If $a \in undec(L)$, then there is a $b \in A$ s.t. $(b, a) \in R$ and $b \in undec(L)$ and there is no $b \in A$ s.t. $(b, a) \in R$ and $b \in in(L)$.*

Now we are ready to define the remaining counterparts of the extension-based semantics in terms of complete labellings.

Definition 3. *Given an AF $\mathcal{A}$ = (A,R) and a labelling $L \in \mathcal{L}_{co}(\mathcal{A})$. L is a*

1. *stable labelling ($L \in \mathcal{L}_{st}(\mathcal{A})$) iff $undec(L) = \emptyset$,*
2. *preferred labelling ($L \in \mathcal{L}_{pr}(\mathcal{A})$) iff for each $L' \in \mathcal{L}_{co}(\mathcal{A})$, $in(L) \not\subset in(L')$,*
3. *grounded labelling ($L \in \mathcal{L}_{gr}(\mathcal{A})$) iff for each $L' \in \mathcal{L}_{co}(\mathcal{A})$, $in(L') \not\subset in(L)$.*

Theorem 1. *[5] Given an AF $\mathcal{A}$. For each $\sigma \in \{co, st, pr, gr\}$,*

1. *$E \in \mathcal{E}_{\sigma}(\mathcal{A})$ iff $\exists L \in \mathcal{L}_{\sigma}(\mathcal{A}) : in(L) = E$ and*
2. *$|\mathcal{E}_{\sigma}(\mathcal{A})| = |\mathcal{L}_{\sigma}(\mathcal{A})|$ holds.*

This theorem will be used to make the splitting results applicable for our algorithm.

2.3 Splitting Results

Baumann [1][2] showed that, under certain conditions, the computation of the extensions of an AF $\mathcal{A}$ can be considerably simplified: one splits the AF $\mathcal{A}$ into two subframeworks $\mathcal{A}_1$ and $\mathcal{A}_2$, computes an extension E_1 of $\mathcal{A}_1$, uses E_1 to reduce and modify $\mathcal{A}_2$, computes an extension E_2 of the modified and reduced version of $\mathcal{A}_2$ and then simply combines E_1 and E_2. We briefly recall the relevant definitions as they are crucial for the algorithms to be discussed later.

Definition 4. *Let $\mathcal{A}_1 = (A_1, R_1)$ and $\mathcal{A}_2 = (A_2, R_2)$ be AFs such that $A_1 \cap A_2 = \emptyset$. Let $R_3 \subseteq A_1 \times A_2$. We call the tuple $(\mathcal{A}_1, \mathcal{A}_2, R_3)$ a splitting of the argumentation framework $\mathcal{A} = (A_1 \cup A_2, R_1 \cup R_2 \cup R_3)$.*

[2] The full version is available at http://www.informatik.uni-leipzig.de/~baumann/.

Definition 5. *Let $\mathcal{A} = (A, R)$ be an AF, A' a set disjoint from A, $S \subseteq A'$ and $L \subseteq A' \times A$. The (S, L)-reduct of $\mathcal{A}$, denoted $\mathcal{A}^{S,L}$ is the AF*

$$\mathcal{A}^{S,L} = (A^{S,L}, R^{S,L})$$

where

$$A^{S,L} = \{a \in A \mid (S, \{a\}) \not\in L)\}$$

and

$$R^{S,L} = \{(a, b) \in R \mid a, b \in A^{S,L}\}.$$

Definition 6. *Let $\mathcal{A} = (A, R)$ be an AF, E an extension of $\mathcal{A}$. The set of arguments undefined with respect to E is*

$$U_E = \{a \in A \mid a \notin E, (E, \{a\}) \not\in R\}.$$

It can be checked that in case of $\sigma \in \{co, st, pr, gr\}$, U_E equals $undec(L)$, where L is the unique σ - labelling s.t. $in(L) = E$ holds (compare Theorem 1).

Definition 7. *Let $\mathcal{A} = (A, R)$ be an AF, A' a set disjoint from A, $S \subseteq A'$ and $L \subseteq A' \times A$. The (S, L)-modification of $\mathcal{A}$, denoted $mod_{S,L}(\mathcal{A})$, is the AF*

$$mod_{S,L}(\mathcal{A}) = (A, R \cup \{(b, b) \mid a \in S, (a, b) \in L\}).$$

We now present the splitting theorem in both extension-based and labelling-based semantics style. The labelling-based notation can be easily obtained by using the original extension-based splitting results (1.(a), 2.(a)), Theorem 1 and the observation below Def. 6.

Theorem 2. *($\sigma \in \{st, pr, co, gr\}$) Let $\mathcal{A} = (A, R)$ be an AF which possesses a splitting $(\mathcal{A}_1, \mathcal{A}_2, R_3)$ with $\mathcal{A}_1 = (A_1, R_1)$ and $\mathcal{A}_2 = (A_2, R_2)$.*

1. *(a)* $E_1 \in \mathcal{E}_\sigma(\mathcal{A}_1) \wedge E_2 \in \mathcal{E}_\sigma(mod_{U_{E_1}, R_3}(\mathcal{A}_2^{E_1, R_3})) \Rightarrow E_1 \cup E_2 \in \mathcal{E}_\sigma(\mathcal{A})$
   (b) $L_1 \in \mathcal{L}_\sigma(\mathcal{A}_1) \wedge L_2 \in \mathcal{L}_\sigma(mod_{undec(L_1), R_3}(\mathcal{A}_2^{in(L_1), R_3}) \Rightarrow$
   $\exists!\ L \in \mathcal{L}_\sigma(\mathcal{A}) : in(L) = in(L_1) \cup in(L_2)$
2. *(a)* $E \in \mathcal{E}_\sigma(\mathcal{A}) \Rightarrow E \cap A_1 \in \mathcal{E}_\sigma(\mathcal{A}_1) \wedge E \cap A_2 \in \mathcal{E}_\sigma(mod_{U_{E \cap A_1}, R_3}(\mathcal{A}_2^{E \cap A_1, R_3}))$
   (b) $L \in \mathcal{L}_\sigma(\mathcal{A}) \Rightarrow \exists!\ L_1 \in \mathcal{L}_\sigma(\mathcal{A}_1) : in(L_1) = in(L) \cap A_1 \wedge$
   $\exists!\ L_2 \in \mathcal{L}_\sigma(mod_{undec(L) \cap A_1, R_3}(\mathcal{A}_2^{in(L) \cap A_1, R_3})) : in(L_2) = in(L) \cap A_2$

2.4 Splittings and Strongly Connected Components (SCC)

To generate a splitting we use the related graph-theoretic concept of *strongly connected components*. A directed graph is strongly connected if there is a path from each vertex to every other vertex. The SCCs of a graph $\mathcal{A}$ ($SCC(\mathcal{A})$ for short) are its maximal strongly connected subgraphs. Contracting every SCC to a single vertex leads to an acyclic graph. It is well-known that an acyclic graph induces a partial order on the set of vertices. Based on this order every SCC-decomposition can be easily transformed into a splitting. The most obvious

possibility is to take the union of the initial nodes of the decomposition (= $\mathcal{A}_1$) and the union of the remaining subgraph (= $\mathcal{A}_2$).
The following figure exemplifies the idea. We sketch three different splittings, namely S_1, S_2 and S_3. Note that these are not all possible splittings.

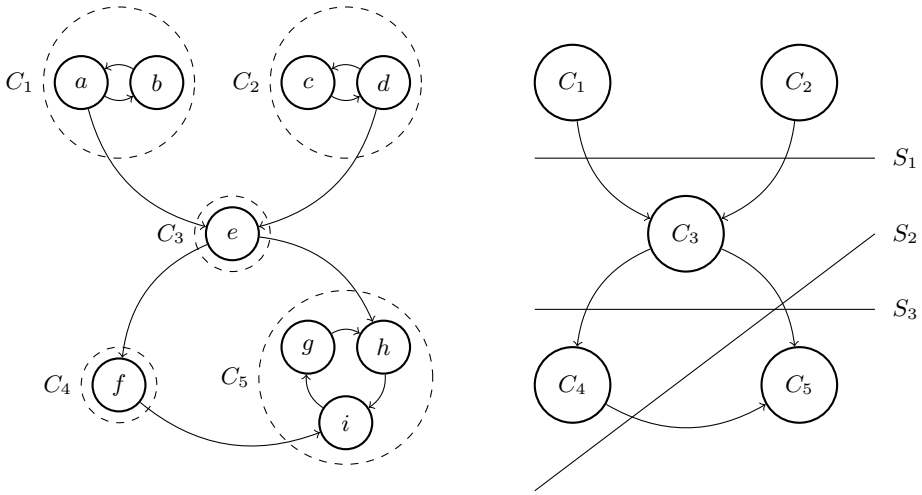


Fig. 1. SCCs and Splittings

3 Algorithms

Our implementation is available at wwwstud.rz.uni-leipzig.de/∼bss01gsc/. It is based on the labelling algorithms for grounded, preferred and stable semantics in [5] and on the standard Tarjan algorithm for computing strongly connected components from [7]. For the latter we refer the reader to the original paper. We briefly describe the former to make the paper more self-contained.

3.1 Labelling Algorithms

The grounded labelling (L_{gr}) is generated as follows: all arguments which are not attacked are assigned the label *IN*. The next step is to assign the label *OUT* to all those arguments that are attacked by at least one of the arguments just labeled *IN*. We continue assigning the label *IN* to any argument having all of its attackers labeled *OUT*. The iteration stops when no further assignment can be made. The set $undec(L_{gr})$ is the set of arguments from A which were not labeled during the iteration.

In order to present the algorithms for preferred and stable labellings, further terminological explanations are in place.

Definition 8. *Given an AF $\mathcal{A} = (A, R)$ and a labelling $L \in \mathcal{L}_\sigma(\mathcal{A})$, an argument $a \in A$ is*

1. *legally IN iff x is labeled IN and $\forall b : (b, a) \in R$, b is labeled OUT,*
2. *legally OUT iff x is labeled OUT and $\exists b : (b, a) \in R$ and b is labeled IN,*
3. *illegally IN iff it is not legally IN,*
4. *illegally OUT iff it is not legally OUT,*
5. *super-illegally IN iff it is illegally IN and $\exists b : (b, a) \in R$ and b is legally IN or UNDEC.*

The algorithm for computing all preferred labellings (Algorithm 1) starts by assigning to all arguments the label *IN* (labelling L_{IN}), and initializing an empty set in which candidate labellings are to be stored. Then, by way of the main procedure *find_labellings* arguments that are *illegally IN* in L_{IN} are identified. To each of these arguments a procedure called *transition_step* is applied, by which the label of the given argument is changed from *IN* to *OUT*. If such an argument whose label has been changed from *IN* to *OUT* or if any argument(s) it attacks is *illegally OUT*, it will be relabeled as *UNDEC*. Thus we have obtained a new labelling which contains one less *IN*-argument. Then the entire process repeats again by passing any new labelling onto the main procedure, and the process continues until an acceptance or rejection condition is met. A labelling which does not have any argument which is *illegally IN* will be added to the candidate labellings, unless at any previous stage in the recursion it is detected that a better labelling has been found, i.e. a labelling with a larger *in*-set is already contained in *candidate_labellings*. If such a labelling with a larger cardinality of the *in*-set exists, the current labelling will not be processed.

In order to avoid the situation in which incomplete labellings are being generated by any incorrect assignment of labels, the algorithm is designed to always extract first those arguments that are *super-illegally IN*, i.e. arguments having at least one attacker *legally IN* or *UNDEC*, whenever we try to extract arguments that are *illegally IN*.

The algorithm for computing all stable labellings is obtained by rewriting line 1.5 of the algorithm for preferred labellings to read "**if** $undec(L) \neq \emptyset$ **then return**". If the set of arguments labeled *UNDEC* in a labelling is not empty, i.e. it violates the requirement for a stable labelling, the labelling will not be further processed.

3.2 Computation of Splitting

Our splitting algorithm consists of two parts: The first part (Algorithm 2) is executed prior to the first call of a labelling algorithm for a semantics and computes $\mathcal{A}_1$, $\mathcal{A}_2$ and the set R_3. The second part (Algorithm 3) is executed after receiving an extension from the labelling algorithm. The tuple $\mathcal{A}_2$ is then modified in accordance with the extension.

The first task is set to look for all the initial arguments (A_1) of our framework ($\mathcal{A}$). We use the set of *strongly connected components* returned by the Tarjan algorithm. The algorithm starts by introducing a Boolean variable *scc_attacked* which will be initialized to *false* for every SCC in $SCC(\mathcal{A})$. Given an SCC, once an argument in this SCC is attacked by some argument in another SCC, the

Algorithm 1: Computation of Preferred Labellings

input : $L_{IN} = (in(L_{IN}) = A, out(L_{IN}) = \emptyset, undec(L_{IN}) = \emptyset)$

```
1.1  candidate_labellings := ∅
1.2  find_labellings(L_IN)

1.3  PROCEDURE find_labellings(L)
1.4  begin
1.5      if ∃L' ∈ candidate_labellings : in(L) ⊂ in(L') then return;
1.6      if L does not contain an argument illegally IN then
1.7          foreach L' ∈ candidate_labellings do
1.8              if in(L') ⊂ in(L) then
1.9                  candidate_labellings := candidate_labellings − {L'}
1.10         candidate_labellings := candidate_labellings ∪ {L}
1.11         return;
1.12     else
1.13         if L has an argument that is super-illegally IN then
1.14             x := some argument that is super-illegally IN in L
1.15             find_labellings(transition_step(L, x))
1.16         else
1.17             foreach x that is illegally IN in L do
1.18                 find_labellings(transition_step(L, x))
```

variable will be set to *true* and the execution of the algorithm for this SCC stops. Then the algorithm starts processing the next SCC. Only if *scc_attacked* remains *false*, which means that the corresponding SCC is not attacked, will all the arguments of this SCC be added to A_1. This way of splitting corresponds to S_1 in Fig. 1.

The splitting operation described above may result in subframeworks which differ a lot in size. We also provide a possibility to equalize the cardinalities along the partial ordering dictated by $SCC(\mathcal{A})$. The algorithm, called *optimize*, accepts the already computed arguments of A_1 and adds new ones under certain conditions. The first criterion used is the cardinality of A_1. Since the addition of new arguments relies on the partial order, it may not always be possible. Therefore, choosing 45% as a starting condition for equalization was an attempt to optimally equalize the numbers of arguments on the one hand, and on the other not to slow down the splitting process unnecessarily. Another condition limits the number of arguments added to A_1 by imposing a relative restriction on the added SCC's cardinality, i.e. if $|SCC| + |A_1| > |A| * 60\%$, the SCC will not be accepted. The algorithm runs recursively until no further arguments can be added (i.e. when $|optimal\_set| = |A_1|$). This way of splitting corresponds to S_3 in Fig. 1.

On the basis of the set A_1 we can then compute the sets R_1, A_2, R_2 as well as the set of attacks along which the framework is split (R_3). The pseudo code for these operations is not included here due to their obvious simplicity.

The processing of the tuple $\mathcal{A}_1$ by a labelling algorithm may return an extension as part of a labelling, if it exists. This extension (E_1) will in turn be used for modifying the AF $\mathcal{A}_2$ in the second part of the splitting algorithm. We start with the set A_2' which is A_2 minus all the arguments in A_2 that are attacked by E_1, and we call it the modified set of A_2.

Next we apply the second algorithm on E_1, starting with an empty set, in order to compute a reduced set of undefined arguments (U_{E_1}). Note that we are not concerned with all the undefined arguments as stated in Definition 6, but only with those that are sources of an attack in R_3. Whenever an argument is a source of an attack in R_3, if it neither is an element of the extension E_1 nor is attacked by E_1, it will be added to the set U_{E_1}.

We then proceed to the final step in the modification of the AF $\mathcal{A}_2$. Given U_{E_1}, for every argument of A_2' which is attacked by U_{E_1}, a loop is added. By this addition, we have modified the set R_2. We call this modified set R_2', and now we can define $\mathcal{A}_2'$ as the tuple (A_2', R_2'). With the given definition, $\mathcal{A}_2'$ is to be processed by a labelling algorithm.

4 Experimental Results

Our evaluation of the runtime for grounded, preferred and stable semantics is based on the sampling of 100 random frameworks[3]. The tests were performed on a Samsung P510 notebook with a Pentium Dual Core Processor, CPU speed: 2.0 GHz, CPU Caches: 32 KB (L1) and 1024 KB (L2), RAM: 2 GB. We focused in our experiments on frameworks where the number of attacks (n) exceeds the number of arguments (m) by a factor between 1.5 and 3.

The reasons for this restriction are as follows. First of all, even leaving execution times aside[4], by further increasing the number of attacks the probability of generating frameworks consisting of a single SCC grows, thus rendering the experiment inconclusive as splitting has no effect on AFs with a single SCC. For example, initial tests showed that if 500 or more attacks (n) are given for 100 arguments (m), then almost all of the randomly generated frameworks will consist of only a single SCC and no effect of splitting is to be expected.

On the other hand, choosing an n smaller than m would not lead to significant differences in execution time between AFs with and without splitting as execution times tend to be fast under such conditions anyway.

With the above limitations in mind, a total of 100 examples were collected, with 20 examples extracted from each of the following m/n combinations: 10/30,

[3] The attacks were created by randomly selecting the source and the target argument for frameworks with given number of arguments and attacks.

[4] For example, our preliminary testing showed that for AFs with 100 arguments, if 200 attacks are specified, the percentage of frameworks with runtime over 3 *min* for preferred semantics without splitting was about 70%.

Algorithm 2: Computation of Splitting, part 1

input : set of strongly connected components ($SCC(\mathcal{A})$)
output: A_1, R_1, A_2, R_2, R_3

```
2.1  PROCEDURE compute_A1(SCC(A))
2.2  begin
2.3      foreach SCC ∈ SCC(A) do
2.4          scc_attacked := false
2.5          loop:
2.6          foreach a ∈ SCC do
2.7              foreach b s.t. (b, a) ∈ R do
2.8                  if b ∉ SCC then
2.9                      scc_attacked = true
2.10                     break loop;
2.11         if scc_attacked = false then add SCC to A1
2.12     return A1

2.13 PROCEDURE optimize(SCC(A), A1)
2.14 begin
2.15     optimal_set := A1
2.16     illegal_attacks := false
2.17     foreach SCC ∈ SCC(A) do
2.18         if |A1| < |A| * 0.45 then
2.19             pick an a ∈ SCC
2.20             if a ∉ A1 and |A1| + |SCC| < |A| * 0.6 then
2.21                 illegal_attacks = false
2.22                 loop:
2.23                 foreach a ∈ SCC do
2.24                     foreach (b, a) ∈ R do
2.25                         if b ∉ A1 and b ∉ SCC then
2.26                             illegal_attacks = true
2.27                             break loop;
2.28                 if illegal_attacks = false then add SCC to A1
2.29     if |A1| < |A| * 0.45 and |optimal_set| ≠ |A1| then
2.30         optimize(SCC(A), A1)
2.31     return A1
```

Algorithm 3: Computation of Splitting, part 2

input : an extension of $\mathcal{A}_1$ (E_1), A_2, R_1, R_2, R_3
output: $\mathcal{A}'_2 = (A'_2, R'_2)$

3.1 $compute\_modified\_A_2(E_1, A_2, R_3)$
3.2 $compute\_U_{E_1}(E_1, R_1, R_3)$
3.3 $compute\_modified\_R_2(U_{E_1}, R_2, R_3)$

3.4 **PROCEDURE** $compute\_modified\_A_2(E_1, A_2, R_3)$
3.5 **begin**
3.6 $A'_2 := A_2$
3.7 **foreach** $a \in E_1$ **do**
3.8 **foreach** $(a, b) \in R_3$ **do**
3.9 **if** $b \in A'_2$ **then** remove b from A'_2
3.10 **return** A'_2

3.11 **PROCEDURE** $compute\_U_{E_1}(E_1, R_1, R_3)$
3.12 **begin**
3.13 $U_{E_1} := \emptyset$
3.14 **foreach** $(a, b) \in R_3$ **do**
3.15 **if** $a \notin E_1$ *and* $(E_1, \{a\}) \bar{\notin} R_1$ **then** add a to U_{E_1}
3.16 **return** U_{E_1}

3.17 **PROCEDURE** $compute\_modified\_R_2(U_{E_1}, R_2, R_3)$
3.18 **begin**
3.19 $R'_2 := R_2 - \{(x, y) | (x, y) \in R_2$ and $(x \notin A'_2$ or $y \notin A'_2)\}$
3.20 **foreach** $a \in U_{E_1}$ **do**
3.21 **foreach** $(a, b) \in R_3$ **do** add (b, b) to R'_2
3.22 **return** R'_2

50/100, 100/175, 200/375 and 500/750. A brief description of the results obtained will be presented below together with a tabular summary of statistical data for each combination. Each table contains average-runtime results (in milliseconds) and gain-in-time results (in %)[5] for the grounded, preferred and stable semantics. Under "average runtime", the first column contains results from executing without splitting, the second from executing with non-optimized splitting and the third from executing with optimized splitting. Under "gain in time", minimal, maximal and average gain results, each in relation to non-optimized and optimized splitting, are distinguished.

The 10/30 combination was the only case in which we experienced no runtime that was over 3 *min*.[6] Thanks to the low number of arguments we were given a possibility of structural analysis. Although 20 examples is a small sample size, we were able to distinguish 4 characteristics based on the structure of the framework and the corresponding difference in runtime between executions without and with splitting. The analysis below applies to the preferred and stable semantics as the execution of the grounded semantics did not show any difference.

First, in 3 cases out of 20 a single SCC was generated. As splitting has no effect on AFs consisting of a single SCC, there was no runtime improvement for all 3 semantics. However, no noticeable runtime delay in relation to the splitting process was recorded either.

Second, 3 further examples had the form of a single argument SCC attacking a large SCC. Here we recorded no improvement or only a slight improvement in the runtime when splitting was applied: 0-20%.

Third, yet 3 further cases consisted of a single argument SCC with a self-loop attacking a large SCC. The only difference regarding the single argument between this form and the previous one was that we now had a loop attack. However in terms of runtime the gap was significant. In the second case it was between 68-71% for preferred semantics and between 99-100% for stable semantics.

And last, 11 of the random AFs had the form of a larger SCC attacking a single argument SCC, a single argument SCC with a self-loop or two SCCs; or the form of two SCCs, with at least one attack each, attacking a single SCC. The difference in execution without and with splitting ranged here between 80-99% for preferred semantics and between 59-100% for stable semantics.

The limited data suggest that splitting can render computation significantly faster for frameworks with certain characteristics. It seems that the most relevant are those AFs having one or more SCCs, each with at least one attack (i.e. a single argument SCC with a loop or an SCC with at least 2 arguments), attacking one or more SCCs whose structure in itself is not relevant.[7]

[5] For convenience, in the presented data we use "0 *ms*" to mean "close to 0 *ms*" and "100%" to mean "close to 100%".

[6] It comes as no surprise since the computation of preferred labellings for an AF with 10 arguments and 100 attacks takes around 260,000 *ms*

[7] An additional test on an AF of 10 arguments, of which 9 constituted an SCC with 81 attacks and all 9 attacked the 10th argument, recorded a 90% runtime difference for both preferred and stable semantics. This additional result lies nicely within the

In general we obtained an average acceleration of 60% for both types of splitting in comparison to an execution without splitting. It is partly due to the fact that for the 10/30 combination both non-optimized splitting and optimized splitting usually overlap, which in turn is a result of the existence of large SCCs that limits the possibility of having different splittings. In no case was the execution with splitting slower than the one without.

Table 1. Evaluation results for 10 arguments and 30 attacks

m = 10	**average runtime (in ms)**			**gain in time (in %)**					
n = 30	**w/o spl.**	**w/ spl.**	**opt. spl.**	**min**	**min/op**	**max**	**max/op**	**avg.**	**avg./op**
grounded	1	1	1	0	0	0	0	0	0
pref.	3871	886	890	0	0	99	99	60	61
stable	1040	267	262	0	0	100	100	59	60

The runtimes for the 50/100 combination were very diversified: from 1 *ms* (for stable) and 2 *ms* (for preferred semantics) to 381,512 *ms* (preferred)[8] and 4,456 *ms* (stable). The grounded labelling was computed at the speed of 1-3 *ms* in each case, no improvement nor delay was recorded for executions with splitting in comparison to those without.

In 9 out of the 20 cases, the computation time for preferred and stable labellings without and with splitting was very short (below 20 *ms*). No significant difference was observed. The time gain for these cases was given as 0%, which had a negative effect on the average gain in time as shown in Table 2: it dropped to only 26-29%. Note that the maximal gain in time for both semantics was at 99%.

For the stable semantics we observed dramatic improvements in cases where no labellings existed. Through splitting of the framework, the time needed to find the first argument of the *undec* set, hence breaking the execution of the labelling algorithm, was at times very short. In 8 out of 15 cases where no labelling existed, the execution times lay below 20 *ms* which as mentioned above had 0% gain. Among the remaining 7 cases, 2 recorded an improvement of 99%, the rest between 17-75%. In none of the 20 examples was the execution without splitting faster than the one with splitting. Neither significant improvement nor delay was found for optimized splitting as compared to regular splitting.

ranges of the previously obtained 80-99% and 59-100% respectively. A further test of a single argument with a self-loop attacking each argument of an SCC with 9 arguments and 81 attacks showed a 90% runtime difference for preferred semantics and 100% for stable semantics. The performance was evidently better than the previously obtained result for preferred semantics (68-71%). Having removed the loop attack we obtained a runtime of 1 *ms* for preferred and stable semantics, both with and without splitting. Again, these results are also in compliance with the ones obtained in the sample test using 20 examples.

[8] This example had already been included in the data before the imposition of the 3-minute limit, and so this is the only example with a runtime above 3 *mins*.

Table 2. Evaluation results for 50 arguments and 100 attacks

m = 50	**average runtime (in ms)**			**gain in time (in %)**					
n = 100	**w/o spl.**	**w/ spl.**	**opt. spl.**	**min**	**min/op**	**max**	**max/op**	**avg.**	**avg./op**
grounded	2	2	2	0	0	0	0	0	0
pref.	35860	23237	23352	0	0	99	99	29	26
stable	663	487	480	0	0	99	99	29	29

Some 40% of the frameworks generated with 100 arguments and 175 attacks had a computation time of at least 3 *min* for the preferred semantics without splitting. They were not taken into consideration for the reason stated at the beginning of this section. In the collected examples, the runtimes varied from around 20 *ms* to slightly below 40,000 *ms*. No stable labelling existed in 19 out of the 20 examples. In 9 out of these 19 examples, we obtained an improvement of 90-100% for the stable semantics and 0-50% for the remaining 10. No slow down due to the process of splitting was noticeable.

Here, for the first time, we recorded a significant improvement in runtime when the optimized version of splitting was applied. It was 13% for the preferred semantics and 5% for the stable semantics, both of which were better than the non-optimized variant. On average, an execution with splitting was better than one without splitting by 56-69% for the preferred semantics and by 60-65% for the stable semantics.

Table 3. Evaluation results for 100 arguments and 175 attacks

m = 100	**average runtime (in ms)**			**gain in time (in %)**					
n = 175	**w/o spl.**	**w/ spl.**	**opt. spl.**	**min**	**min/op**	**max**	**max/op**	**avg.**	**avg./op**
grounded	2	2	2	0	0	0	0	0	0
pref.	8335	3701	2502	0	0	93	99	56	69
stable	499	297	262	0	0	100	99	60	65

The computation time for preferred and stable labellings without splitting in frameworks of 200 arguments and 375 attacks was in general above 15 *ms*, thus making a more precise comparison possible. All the generated AFs showed a runtime improvement of at least 14% (pref.) and 26% (stable) when the execution with splitting is compared to the execution without splitting. Here too the gain in time reached in some cases 99% for the preferred labellings and 96% for the stable labellings.

With an average runtime of 3 *ms* for the grounded semantics, no difference between execution without and with splitting was found. The computation of stable labellings with applied splitting took on average 56% less time than that without. For the preferred semantics, the gain was somewhat less, it was 45% with optimized splitting and 47% with non-optimized splitting.

Table 4. Evaluation results for 200 arguments and 375 attacks

m = 200	**average runtime (in ms)**			**gain in time (in %)**					
n = 375	**w/o spl.**	**w/ spl.**	**opt. spl.**	**min**	**min/op**	**max**	**max/op**	**avg.**	**avg./op**
grounded	3	3	3	0	0	0	0	0	0
pref.	9333	6531	6296	14	16	99	98	47	45
stable	352	236	222	26	26	96	93	56	56

It was relatively comfortable testing the 500/750 combination since only about 20% of the randomly generated frameworks had a runtime above 3 *min* for preferred labellings without splitting. The execution time was quite steady. The lowest runtime for preferred semantics without splitting was 53 *ms* and 58 *ms* for stable semantics without splitting. The absence of drastic highs and lows was mirrored in all the average runtimes for preferred semantics, which were much lower than the average runtimes measured for 200/375. Here we observed also a steady improvement after splitting was applied. The lowest of which was 35% for preferred semantics and 33% for stable. The upper range was also less drastic with up to 86% for preferred and 97% for stable. The average differences were quite high with 57-61% for preferred labellings and 62-66% for stable. There was a drop in efficiency for the optimized type of splitting as compared to the non-optimized type (by 4% for both preferred and stable labellings). However, in AFs with a runtime above 700 *ms*, the optimized type ran faster than the one without optimization. In no case though was an execution with splitting slower than the one without splitting.

While in frameworks with 200 arguments and lower the grounded semantics did not perform worse after splitting, here we observed a visible slowdown. There was an average loss of 2% in the case of the non-optimized variant and an average loss of 36% in the case of the optimized variant.

Table 5. Evaluation results for 500 arguments and 750 attacks

m = 500	**average runtime (in ms)**			**gain in time (in %)**					
n = 750	**w/o spl.**	**w/ spl.**	**opt. spl.**	**min**	**min/op**	**max**	**max/op**	**avg.**	**avg./op**
grounded	10	10	13	-12	-60	15	-15	-2	-36
pref.	2785	1697	1168	36	35	86	78	61	57
stable	232	120	99	33	47	97	89	66	62

5 Conclusions

Based on our evaluations of 100 randomly generated AFs, we have made the following observations:

1. Among the 100 AFs, we observed an average improvement by 50-51% and by 54% for preferred and stable semantics respectively. The data contained some inconclusive examples which had “marred” the results to some extent.

2. No instance, neither for preferred semantics nor for stable, was found in which the execution with splitting lasted longer than the one without. This shows that the additional overhead introduced by splitting is negligible.
3. The optimized type of splitting did better than the non-optimized type in cases when the AF without splitting had a relatively long runtime. When the runtime was relatively short, the type without optimization usually performed better.
4. Splitting may significantly improve runtime for stable semantics in frameworks where no stable labellings exist. By splitting the framework, we were able to complete the execution of the algorithm a lot faster because it took less time to find a labelling with the *undec* set that was not empty.
5. It seems that there exist certain regularities between the structure of frameworks and the corresponding runtime. Having an SCC with at least one attack (or several SCCs with at least one attack each) attacking the rest of the framework can improve runtime significantly. We especially hope that this will greatly affect computation of large frameworks with large SCCs, which so far we were unable to test due to the required long computation time.

In future work we plan not only to extend our evaluation to larger AFs, we would also like to see whether there is an impact of moving from randomly generated to "natural" argumentation frameworks arising in realistic argumentation scenarios. Moreover, our results together with the theoretical considerations from the beginning of Sect. 4 suggest an advanced version of the algorithm where splitting is (1) performed iteratively on the identified subparts and (2) conditioned on the number of arguments and ratio between arguments and attacks, that is, only if the number of arguments is above a threshold and this ratio is in the "interesting" range splitting is performed.

References

1. Ringo Baumann. Splitting an argumentation framework. In *Proc. LPNMR-11*, pages 40–53. Springer, 2011.
2. Martin Caminada. On the issue of reinstatement in argumentation. In *Proc. JELIA-06*, pages 111–123, 2006.
3. Phan Minh Dung. 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.
4. Vladimir Lifschitz and Hudson Turner. Splitting a logic program. In *ICLP*, pages 23–37, 1994.
5. Sanjay Modgil and Martin Caminada. Proof theories and algorithms for abstract argumentation frameworks. In Iyad Rahwan and Guillermo R. Simari, editors, *Argumentation in Artificial Intelligence*, pages 105–132. Springer, 2009.
6. Henry Prakken. An abstract framework for argumentation with structured arguments. *Argument and Computation*, 1:93–124, 2010.
7. Robert Endre Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.
8. Hudson Turner. Splitting a default theory. In *Proc. AAAI-96*, pages 645–651, 1996.

A Constraint-based Computational Framework for Argumentation Systems

Stefano Bistarelli[1,2] and Francesco Santini[1]

[1] Dipartimento di Matematica e Informatica, Università di Perugia, Italy
[bista,francesco.santini]@dmi.unipg.it
[2] Istituto di Informatica e Telematica (CNR), Pisa, Italy
stefano.bistarelli@iit.cnr.it

Abstract. We propose a framework based on *Constraint Programming* in order to model and solve hard problems related to Argumentation Frameworks (AFs). *Constraint Satisfaction Problems* or CSPs are mathematical problems defined as a set of variables whose value must satisfy a number of constraints or limitations. CSPs offer a wide number of efficient techniques (as inference and search algorithms) that can tackle the complexity in *a)* finding all the possible Dung's conflict-free, admissible, complete and stable extensions in AFs and *b)* important problems related to weighted Dung's grounded extensions. We use JaCoP, a Java library which provides the user with a *Finite Domain Constraint Programming* paradigm, to model and solve these two problems. We test our implementation on small-world networks in order to show the performance and feasibility of such tool.

1 Introduction

Interactions are a core part of all multi-party systems (e.g. multi-agent systems). *Argumentation* [9] is based on the exchange and the evaluation of interacting arguments which may represent information of various kinds, especially beliefs or goals. Argumentation can be used for modeling some aspects of reasoning, decision making, and dialogue. For instance, when an agent has conflicting beliefs (viewed as arguments), a (nontrivial) set of plausible consequences can be derived through argumentation from the most acceptable arguments for the agent. Argumentation has become an important subject of research in Artificial Intelligence and it is also of interest in several disciplines, such as Logic, Philosophy and Communication Theory [16].

Many theoretical and practical developments build on Dung's seminal theory of argumentation. A *Dung Argumentation Framework* (AF) is a directed graph consisting of a set of arguments and a binary conflict based *attack relation* among them. The sets of arguments to be considered are then defined under different semantics, where the choice of semantics equates with varying degrees of scepticism or credulousness. The main issue for any theory of argumentation is the selection of acceptable sets of arguments, based on the way arguments interact.

Intuitively, an acceptable set of arguments must be in some sense coherent and strong enough (e.g. able to defend itself against all attacking arguments).

Constraint Programming (CP) is a powerful paradigm for solving combinatorial search problems that draws on a wide range of techniques from artificial intelligence, computer science, databases, programming languages, and operations research. Constraint programming is currently applied with success to many domains, such as scheduling, planning, vehicle routing, configuration, networks, and bioinformatics [22]. The basic idea in constraint programming is that the user states the constraints and a general purpose constraint solver is used to solve them. Constraints are just relations, and a constraint satisfaction problem (CSP) states which relations should hold among the given decision variables.

Constraint solvers search the solution space either systematically, as with backtracking or branch and bound algorithms, or use forms of local search which may be incomplete. Systematic method often interleave search and inference, where inference consists of propagating the information contained in one constraint to the neighboring constraints.

In this paper we implement and tests the constraint-based framework presented in [5] for weighted AFs. In this paper we test only the classical (i.e. not weighted) AF definition over randomly generated small- world graphs of arguments. We program the constraints in JaCoP [15] to automatically obtain all the Dung's conflict-free, admissible, complete and stable extensions. Then, we report the performance to show the feasibility of such tool applied to AFs; since the number of these extensions, which in practice are subsets of the $\mathcal{A}_{rgs}$ set of arguments, may explode for large $\mathcal{A}_{rgs}$ (the powerset of $\mathcal{A}_{rgs}$ has $2^{|\mathcal{A}_{rgs}|}$ elements), it is important to use techniques to tackle this inherent complexity, as CP. This is particularly true for conflict-free extensions, which represents the least constrained extension.

Moreover, we can find in literature problems related to weighted AFs (see Sec. 6) that are hard to solve. We consider weighted AFs as AFs where attacks are associated with a weight capturing their relative strength [10, 6]. An example, which we represent in this paper is, given a weighted argument system, a set of arguments and inconsistency budget β, checking whether β is minimal is co-NP-complete. This and other problems have been highlighted in [10].

The remainder of this paper is organized as follows. In Sec. 2 we report the theory behind Dung Argumentation (and weighted AFs [10]), while in Sec. 3 we summarize the background (weighted) constraints. In Sec. 4 we propose a mapping from AFs to CSPs. In Sec. 5 we show the tests that prove the feasibility of a constraint-based framework to solve AFs, while in Sec. 6 we model hard problems related to AFs (e.g. preferred extensions) in JaCoP. A comparison with related work is given in Sec. 7. Finally, Sec. 8 presents our conclusions.

2 Dung Argumentation

In [9], the author has proposed an abstract framework for argumentation in which he focuses on the definition of the status of arguments. For that purpose, it can

be assumed that a set of arguments is given, as well as the different conflicts among them. An argument is an abstract entity whose role is solely determined by its relations to other arguments.

Definition 1. *An Argumentation Framework (AF) is a pair* $\langle \mathcal{A}_{rgs}, R \rangle$ *of a set* $\mathcal{A}_{rgs}$ *of arguments and a binary relation* R *on* $\mathcal{A}_{rgs}$ *called the attack relation.* $\forall a_i, a_j \in \mathcal{A}_{rgs}$, $a_i R\, a_j$ *means that* a_i *attacks* a_j*. An AF may be represented by a directed graph (the interaction graph) whose nodes are arguments and edges represent the attack relation. A set of arguments* $\mathcal{B}$ *attacks an argument* a *if* a *is attacked by an argument of* $\mathcal{B}$*. A set of arguments* $\mathcal{B}$ *attacks a set of arguments* $\mathcal{C}$ *if there is an argument* $b \in \mathcal{B}$ *which attacks an argument* $c \in \mathcal{C}$*.*

The "acceptability" of an argument [9] depends on its membership to some sets, called *extensions*. These extensions characterize collective "acceptability".

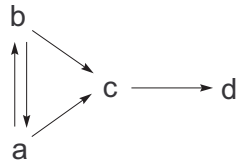


Fig. 1. An example of Dung Argumentation Framework; e.g. c attacks d.

In Fig. 1 we show an example of AF represented as an *interaction graph*: the nodes represent the arguments and the directed arrow from c to d represents the attack of c towards d, that is $c\, R\, d$. Dung [9] gave several semantics to "acceptability". These various semantics produce none, one or several acceptable sets of arguments, called extensions. In Def. 2 we define the concepts of conflict-free and stable extensions:

Definition 2. *A set* $\mathcal{B} \subseteq \mathcal{A}_{rgs}$ *is conflict-free iff no two arguments* a *and* b *in* $\mathcal{B}$ *exist such that* a *attacks* b*. A conflict-free set* $\mathcal{B} \subseteq \mathcal{A}_{rgs}$ *is a stable extension iff for each argument which is not in* $\mathcal{B}$*, there exists an argument in* $\mathcal{B}$ *that attacks it.*

The other semantics for "acceptability" rely upon the concept of defense:

Definition 3. *An argument* b *is defended by a set* $\mathcal{B} \subseteq \mathcal{A}_{rgs}$ *(or* $\mathcal{B}$ *defends* b*) iff for any argument* $a \in \mathcal{A}_{rgs}$*, if* a *attacks* b *then* $\mathcal{B}$ *attacks* a*.*

An admissible set of arguments according to Dung must be a conflict-free set which defends all its elements. Formally:

Definition 4. *A conflict-free set* $\mathcal{B} \subseteq \mathcal{A}_{rgs}$ *is admissible iff each argument in* $\mathcal{B}$ *is defended by* $\mathcal{B}$*.*

Besides the stable semantics, three semantics refining admissibility have been introduced by Dung [9]:

Definition 5. *A preferred extension is a maximal (w.r.t. set inclusion) admissible subset of $\mathcal{A}_{rgs}$. An admissible $\mathcal{B} \subseteq \mathcal{A}_{rgs}$ is a complete extension iff each argument which is defended by $\mathcal{B}$ is in $\mathcal{B}$. The least (w.r.t. set inclusion) complete extension is the grounded extension.*

A stable extension is also a preferred extension and a preferred extension is also a complete extension. Stable, preferred and complete semantics admit multiple extensions whereas the grounded semantics ascribes a single extension to a given argument system.

In the paper we want to to deal also with hard problems related to *weighted AFs* [10]. Formally, a weighted AF is a triple $\langle \mathcal{A}_{rgs}, R, w \rangle$ where $\langle \mathcal{A}_{rgs}, R \rangle$ is a Dung-style abstract argument system, and $w : \mathcal{A}_{rgs} \rightarrow \mathbb{R}^+$ is a function assigning real valued weights to attacks.

A key idea presented in [10] is the *inconsistency budget*, $\beta \in \mathbb{R}^+$, which the authors use to characterise how much inconsistency they are prepared to tolerate. The intended interpretation is that, given an inconsistency budget β, *we would be prepared to disregard attacks up to a total weight of* β [10]. Conventional AFs implicitly assume an inconsistency budget of 0. Weighted AFs have been already modeled also in [6], by considering a semiring-based constraint programming framework.

3 Constraint Programming

The classic definition of a *Constraint Satisfaction Problem* (*CSP*) is as follows. A *CSP* P is a triple $P = \langle X, D, C \rangle$ where X is an n-tuple of variables $X = \langle x_1, x_2, \ldots, x_n \rangle$, D is a corresponding n-tuple of domains $D = \langle D_1, D_2, \ldots, D_n \rangle$ such that $x_i \in D_i$, C is a t-tuple of constraints $C = \langle C_1, C_2, \ldots, C_t \rangle$. A constraint C_j is a pair $\langle R_{S_j}, S_j \rangle$ where R_{S_j} is a relation on the variables in $S_i = scope(C_i)$. In other words, R_i is a subset of the Cartesian product of the domains of the variables in S_i. A solution to the CSP P is an n-tuple $A = \langle a_1, a_2, \ldots, a_n \rangle$ where $a_i \in D_i$ and each C_j is satisfied in that R_{S_j} holds on the projection of A onto the scope S_j. In a given task one may be required to find the set of all solutions, $sol(P)$, to determine if that set is non-empty or just to find any solution, if one exists. If the set of solutions is empty the CSP is unsatisfiable. This simple but powerful framework captures a wide range of significant applications in fields as diverse as artificial intelligence, operations research, scheduling, supply chain management, graph algorithms, computer vision and computational linguistics [22].

One of the main reasons why constraint programming quickly found its way into applications has been the early availability of usable constraint programming systems, as JaCoP, which we will use in the implementation and solution of the AFs [15].

Various generalizations of the classic CSP model have been developed subsequently. One of the most significant is the *Constraint Optimization Problem* (*COP*) for which there are several significantly different formulations, and the nomenclature is not always consistent [22]. Perhaps the simplest COP formulation retains the CSP limitation of allowing only hard Boolean-valued constraints but adds a cost function over the variables, that must be minimized. A *weighted constraint* $\langle c, w\rangle$ is just a classical constraint c, plus a weight w (over natural, integer, or real numbers). The cost of an assignment t of the variable is the sum of all $w(c)$, for all constraints c which are violated by t [22].

Clearly, we can use weighted constraints to model and solve weighted AFs, as presented in [6] and the related hard problems presented in [10]. These last problems are presented in Sec. 6.

4 Mapping AFs to CSPs

In this section we propose a mapping from AFs to CSPs. Given an $AF = \langle \mathcal{A}_{rgs}, R\rangle$, we define a variable for each argument $a_i \in \mathcal{A}_{rgs}$ ($V = \{a_1, a_2, \ldots, a_n\}$) and each of these argument can be taken or not, i.e. the domain of each variable is $D = \{1, 0\}$.

In the following explanation, notice that b attacks a means that b is a parent of a in the interaction graph, and c attacks b attacks a means that c is a grandparent of a. To compute the (weighted) extensions of Dung we need to define specific sets of constraints:

1. **Conflict-free constraints.** Since we want to find the conflict-free sets, if $R(a_i, a_j)$ is in the graph we need to prevent the solution to include both a_i and a_j in the considered extension: $\neg(a_i = 1 \wedge a_j = 1)$. The other possible assignment of the variables ($a = 0 \wedge b = 1$), ($a = 1 \wedge b = 0$) and ($a = 0 \wedge b = 0$) are permitted: in these cases we are choosing only one argument between the two (or none of the two) and thus, we have no conflict.
2. **Admissible constraints.** For the admissibility, we need that, if child argument a_i has a parent node a_f but a_i has no grandparent node a_g (parent of a_f), then we must avoid to take a_i in the extension because it is attacked and cannot be defended by any ancestor: expressed with a unary constraint, $a_i = 0$.
   Moreover, if a_i has several grandparents $a_{g1}, a_{g2}, \ldots, a_{gk}$ and only one parents a_f (child of $a_{g1}, a_{g2}, \ldots, a_{gk}$), we need to add a $k+1$-ary constraint $\neg(a_i = 1 \wedge a_{g1} = 0 \wedge \cdots \wedge a_{gk} = 0)$. The explanation is that at least a grandparent must be taken in the admissible set, in order to defend a_i from one of his parents a_f. Notice that, if a node is not attacked (i.e. he has no parents), he can be taken or not in the admissible set.
3. **Complete constraints.** To compute a complete extension $\mathcal{B}$, we impose that each argument a_i which is defended by $\mathcal{B}$ is in $\mathcal{B}$, except those a_i that, in such case, would be attacked by $\mathcal{B}$ itself [4]. This can be enforced by imposing that for each a_i taken in the extension, also all its $a_{s1}, a_{s2}, \ldots, a_{sk}$

grandchildren (i.e. all the arguments defended by a_i), whose fathers are not taken in the extension, must be in $\mathcal{B}$. Formally, $(a_i = 1 \wedge a_{s1} = 1 \wedge \cdots \wedge a_{sk} = 1)$ only for those a_{si} for which it stands that $(a_{fs_1} = 0 \wedge a_{fs_2} = 0 \wedge \cdots \wedge a_{fs_z} = 0)$, where $a_{fs_1}, a_{fs_2}, \ldots, a_{fs_z}$ are the fathers of a_{si}.

4. **Stable constraints.** If we have a child node a_i with multiple parents $a_{f1}, a_{f2}, \ldots, a_{fk}$, we need to add the constraint $\neg(a_i = 0 \wedge a_{f1} = 0 \wedge \cdots \wedge a_{fk} = 0)$. In words, if a node is not taken in the extension (i.e. $a_i = 0$), then it must be attacked by at least one of the taken nodes, that is at least a parent of a_i needs to be taken in the stable extension (that is, $a_{fj} = 1$). Moreover, if a node a_i has no parent in the graph, it has to be included in the stable extension (notice a_i cannot be attacked by nodes inside the extension, since he has no parent). The corresponding unary constraint is $\neg(a_i = 0)$.

The following proposition states the equivalence between solving an AF_S and its related CSP.

Proposition 1 (Solution equivalence [5]). *Given an $AF = \langle \mathcal{A}_{rgs}, R \rangle$, the solutions of the related CSP obtained with the mapping corresponds to find over AF all the*

- *conflict-free extensions by using conflict-free constraint classes.*
- *admissible extensions by using conflict-free and admissible constraint classes.*
- *complete extensions by using conflict-free, admissible and complete constraint classes.*
- *stable extensions by using conflict and stable constraint classes.*

Notice that the presented soft constraint framework can be easily used to solve argumentation problems with additional constraints, as proposed in [8] only for boolean constraints. We can find further requirements on the sets of arguments which are expected as extensions, like "extensions must contain argument a when they contain b" or "extensions must not contain one of c or d when they contain a but do not contain b".

5 Computing Extensions with Constraint Programming

In this section we program in practice the constraints proposed in Sec. 4 and we test how it efficient to generate all the possible conflict-free, admissible, complete and stable extensions over a randomly generated small-world network. For the implementation we use two Java libraries, the *Java Constraint Programming* solver [15] (JaCoP) and the *Java Universal Network/Graph Framework* (*JUNG*) [19].

JaCoP [15] is a Java library which provides the Java user with *Finite Domain Constraint Programming* paradigm [22]. It provides different type of constraints: most commonly used primitive constraints, such as arithmetical constraints, equalities and inequalities, logical, reified and conditional constraints, combinatorial (global) constraints. It provides a significant number of (global)

constraints to facilitate an efficient modeling. It also provides a modular design of search to help the user on specific characteristics of the problem being addressed.

To practically develop and test our model, we also adopted JUNG [19], a software library for the modeling, generation, analysis and visualization of graphs. We suppose that interaction graphs, where nodes are arguments and edges are attacks (see Sec. 2), represent in this case a kind of social network and consequently show the related small-world properties [21]. A practical example can be the study of discussion fora, where the users post their arguments that can attack other users' arguments [21, 12].

Therefore, for the following tests we use the *BarabasiAlbertGenerator* class [19, 2], which is an evolving scale-free random graph generator. At each time step, a new vertex is created and is connected to existing vertices according to the principle of "preferential attachment", whereby vertices with higher degree have a higher probability of being selected for attachment. At a given step, the probability p of creating an edge between an existing vertex v and the newly added vertex is $p = (degree(v) + 1)/(|E| + |V|)$. where $|E|$ and $|V|$ are, respectively, the number of edges and vertices currently in the network (counting neither the new vertex nor the other edges that are being attached to it). An example of such random graphs with 40 nodes is shown in Fig. 2.

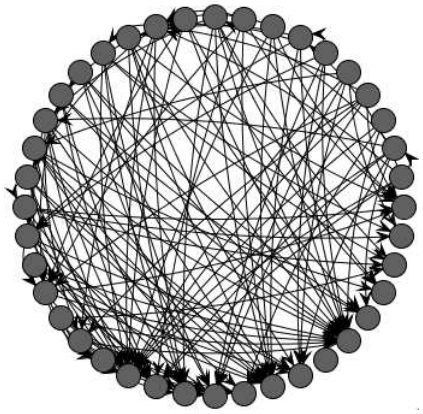


Fig. 2. A small-world network with 40 nodes generated with JUNG by using the *BarabasiAlbertGenerator* class [19, 2].

In the first tests we use a *Depth First Search* (*DFS*) algorithm [15, 22]: this algorithm searches for a possible solution by organizing the search space as a search tree. In every node of this tree a value is assigned to a domain variable and a decision whether the node will be extended or the search will be cut in this node is made. The search is cut if the assignment to the selected domain variable does not fulfill all constraints. Each time during the search, we select the

variable which has most constraints assigned to it and we assign to it a random value from its current domain: we use *MostConstrainedStatic()* as the variable selection method and *IndomainSimpleRandom()* as the value selection method, offered by JaCoP. Moreover, we set a timeout of 180 sec. to interrupt the search procedure and to report the number of solutions found only in that interval; we run our experiments over 4 different sets of random graphs with 10, 40, 60 and 100 nodes. The performance in Tab. 1 reports the average results for each set of 5 different random graphs each. Each row in Tab. 1 shows the number of nodes and edges for the graphs and the average number of found conflict-free, admissible, complete and stable extensions; the time is measured in milliseconds.

Nodes	Edges	Conf-free (time)	Admissible (time)	Compl. (time)	Stable (time)
10	23	73 (317)	32 (307)	29 (99)	1 (161)
40	142	421.697 (≃3min)	30.720 (3.193)	3.108 (938)	1 (64)
60	240	320.828 (≃3min)	320.466 (≃3min)	3.104 (1.035)	2 (98)
100	451	219.194 (≃3min)	220.528 (≃3min)	377.610 (≃3min)	1 (112)

Table 1. The test small-world network generated with JUNG [19] and the corresponding statistics; time is in milliseconds.

The implementation easily finds all the admissible extensions up to 40*nodes* and complete extensions (which do not scale between 40 and 60 nodes) up to 60 nodes. Stable extensions, due to the characteristics of this kind of random scale-free network, are very few. The main problem is represented by finding all the conflict-free extensions, since we have problems already with 40 nodes: in fact, they represent the "less" constrained extension w.r.t. the others and, therefore, we have a higher number of them. However, we remind that these AFs contain at most $2^{|\mathcal{A}_{rgs}|}$ extensions, thus the problem explodes very quickly given $\mathcal{A}_{rgs}$.

However, the constraint framework comes with different performant solving techniques: to show how the performance can be improved, we also used a partial method, the *Limited Discrepancy Search* (*LDS*), which is a kind of *Depth First Search* procedure adopting the method proposed in [13]. If a given number of different decisions along a search path is exhausted, then backtracking is initiated [15, 13]. In Tab. 2 we show the improved results only for conflict-free and admissible extensions. With this method we can find up to five times more the number of extensions w.r.t. DFS, except for conflict-free extensions for graphs with 100 nodes: in this case the number of extensions is so huge that the LDS method performs as plain DFS within 3 minutes.

In order to study our implementation on different networks, we have repeated the same tests for another kind of small-world network, the *KleinbergSmall-WorldGenerator* [15]. This is another graph generator that produces a random graph with small world properties: the model is an $m \times n$ (optionally toroidal) lattice. Each node u has four local connections, one to each of its neighbors, and in addition one long range connection to some node v, where v is chosen ran-

Nodes	LDS n	Conf-free (time)	Admissible (time)
40	30	2.455.079 (≃3min)	30.720 (3.120)
60	40	1.716.880 (≃3min)	1.562.289 (≃3min)
100	451	215.362 (≃3min)	843.927 (≃3min)

Table 2. The test small-world network generated with JUNG [19] and the corresponding statistics: time is in milliseconds.

domly according to probability proportional to d^{α} where d is the lattice distance between u and v and α is the clustering exponent. An example of such graph with 36 nodes is shown in Fig. 3.

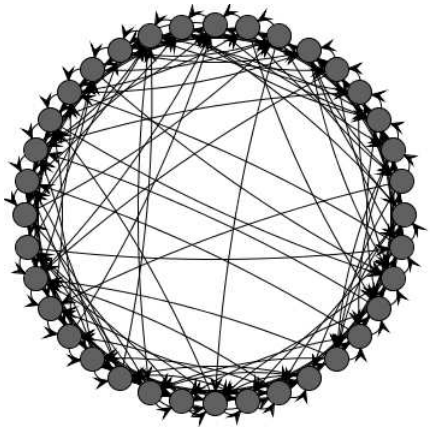


Fig. 3. A small-world network with 36 nodes generated with JUNG by using the *KleinbergSmallWorldGenerator* class [19, 2].

In Tab. 3 we report the performance collected with the same methodology as for Tab 1. The higher number of found stable extensions w.r.t. complete ones in case of 100 nodes can be justified with the fact that stable extensions are more constrained and therefore they are easier to find within the timeout of the search procedure (i.e. still 180 sec.).

The performance in this section have been collected using a MacBook with 2.4Ghz Core Duo and 4Gb 1067Mhz DDR3 of RAM. Notice that the coalition structure generation problem is extremely challenging due to the number of possible solutions that need to be examined. Other works in literature, finding different kinds of constrained (and optimized according to some criteria) coalitions are usually tested over networks of 15-30 nodes [20].

Nodes	Edges	Conf-free (time)	Admissible (time)	Compl. (time)	Stable (time)
9	45	22 (83)	12 (71)	8 (80)	6 (73)
36	180	411.317 (121.371)	5.1412 (8.511)	525 (1.249)	449 (717)
64	320	290.910 (≃3min)	292.480 (≃3min)	92.725 (134.896)	63.878 (15.601)
100	500	219.194 (≃3min)	215.273 (≃3min)	49.787 (≃3min)	58.728 (≃3min)

Table 3. The test small-world network generated with JUNG [19] and the corresponding statistics: time is in milliseconds.

6 Hard Problems in AFs

In this section we propose some hard problems related to AFs; in particular, on finding preferred extensions and weighted grounded extensions. Then we show how to implement the corresponding constraints in JaCoP.

Preferred Extensions. The first interesting problem is determining whether a set of arguments is a preferred extension, which is a co-NP-complete [4] problem. Since a preferred extension S^* is a maximal (w.r.t. set inclusion) admissible subset of $\mathcal{A}_{rgs}$ (see Sec. 2), we can implement the search by finding an admissible extension S' such that $S^* \subset S'$; if S' exists, then S^* is not a preferred extension, otherwise it is. The problem can be solved by assigning to 1 the variables representing the arguments in S^*, given as the input of the decidability problem. This ensures that the search procedure will try to find a superset of S^*. Then we can impose the constraints that we use to find all the admissible extensions.

```
// find all solutions; if not set false
label.getSolutionListener().searchAll(true);
```

Weighted Grounded Extensions. As shown in [10], while the the problem of finding the weighted version of the classical extensions (e.g. stable or admissible) is not computationally harder than the original problem, there are some important problems related to weighted grounded extensions that are very difficult to solve. The weighted AFs and the concept of inconsistency budget β have been introduced in Sec. 2. In the following propositions, i.e. Prop. 2, Prop. 3 and Prop. 4, we show three complex problems proposed in [10].

As for preferred extensions, we say an argument is credulously accepted if it forms a member of at least one weighted grounded extension, and sceptically accepted if it is a member of every weighted grounded extensions [10]. Since there are multiple β-grounded extensions [10], we can consider credulous and sceptical variations of the problem, as with preferred extensions. In Prop. 2 we consider the credulous case first:

Proposition 2 ([10]). *Given weighted argument system $\langle X, A, w\rangle$, inconsistency budget β, and argument $a \in X$, the problem of checking whether $\exists S \in wge(X, A, w, \beta)$ such that $a \in S$ is NP-complete.*

In Prop. 3 we consider the sceptical version of the problem.

Proposition 3 ([10]). *Given weighted argument system $\langle X, A, w\rangle$, inconsistency budget β, and argument $a \in X$, the problem of checking whether, $\forall Y \in wge(X, A, w, \beta)$, we have $a \in Y$ is co-NP-complete.*

Suppose now we have a weighted argument system $\langle X, A, w\rangle$ and a set of arguments S. Then what is the smallest amount of inconsistency would we need to tolerate in order to make S a solution? [10] When considering conflict-free and admissible extensions, the answer is easy: we know exactly which attacks we would have to disregard to make a set of arguments admissible or consistent. However, when considering weighted grounded extensions, the answer is not so easy. There may be multiple ways of getting a set of arguments into a weighted extension, each with potentially different costs; we are thus typically interested in solving the problem expressed by Prop. 4:

Proposition 4 ([10]). *Given a weighted argument system $\langle X, A, w\rangle$, set of arguments $S \subseteq X$, and inconsistency budget β, checking whether β is minimal w.r.t. $\langle X, A, w\rangle$ and S is* co-NP-complete*.*

Implementation in JaCoP. To compute all the weighted grounded extensions, first of all we need to extend our representation to include weighted constraints, as defined in Sec. 3. The weighted constraints represent the costs associated to the attacks among the arguments in weighted AFs. To do so, we need to redefine the conflict-free constraints as proposed also in [6], thus leading to a weighted AF as defined in Sec. 2, i.e. $\langle \mathcal{A}_{rgs}, R, w\rangle$:

Conflict-free constraints. Since we want to find the conflict-free sets, if $w(R(a_i, a_j)) = s$ is in the graph we need assign a s consistency budget to the solution that includes both a_i and a_j in the considered conflict-free extension: $(a_i = 1 \wedge a_j = 1) = s$. For the other possible assignment of the variables we have $(a = 0 \wedge b = 1) = 0, (a = 1 \wedge b = 0) = 0$ and $(a = 0 \wedge b = 0) = 0$, since these assignments are permitted with no cost also in the classical semantics: in these cases we are choosing only one argument between the two (or none of the two) and thus, we have no conflict.

The easy way (if memory is not an issue) is to use *ExtensionalConflict* constraints [15]. In our implementation we can specify, for example, an assignment $[1, 1, 10]$ for three variables: the first two values states that if we take a_1 and a_2 in the same extension (i.e. $a_1 = 1$ and $a_2 = 1$), then the cost to be paid is represented by the third value (i.e. 10). In our model, this cost represents the cost associated to the attack between a_1 and a_2.

Since with this representation we need to specify different costs for any assignment of the considered variables, in general this extensional form could cause the system to run out of memory. However, for the weighted AF case it is easy and not memory-consuming to express all the costs of attacks in this way, since the variables can be assigned only to 0,1 and therefore there are four cases to

define for each attack. In the following we show how to fully express a constraint that defines an attack with cost 10 between a_1 and a_2.

```
store.impose(new ExtensionalConflictVA(new IntVar[]{a1, a2, cost},
new int[][]
{{1, 1, 10},
{0, 1, 0},
{1, 0, 0},
{0, 0, 0}}));
```

So far, to check Prop. 2 we impose $a = 1$ (i.e. a must be present in the extension) and we have also to constrain the sum of the consistency budget β to be equal to the given β (i.e. $cost = \beta$).

To check Prop. 3 we can simply impose a constraint on the sum of the consistency budgets to be equal to the given β (i.e. $cost = \beta$), and then we solve the problem with a self-implemented solution listener [15] (a plug-in in JaCoP that is called by search when a solution is found) that exits from the search when $a \notin X$, that is when the given a does not belong to a solution. At this point we can print it on the screen that a does not belong to every solution with an inconsistency budget equal to β.

To solve the problem expressed by Prop. 4, we simply solve the CSP by minimizing *cost* and then to check if the found solution(s) have a *cost* value equal to the desired β consistency budget. JaCoP offers methods for finding a solution that minimizes a given cost function. A very simple way is to find all the solutions by invoking the following method.

```
IntVar cost;
...
boolean result = label.labeling(store, select, cost);
```

7 Related Work

As far as we know, in literature there is no example of practically finding Dung's classical extensions over a small-world network. Therefore, the work presented in this paper is new, and proposes a bridge towards studying the features of interaction graphs, also from the computational point of view, tied to small-world graphs. Moreover, we propose the constraint framework, as done also in [5, 6] as an ideal framework where to solve AFs: in this paper we also model in JaCoP some hard problems presented in [10]. The framework proposed in this paper is able to solve classical [9] (as shown in Sec. 4) and weighted [10] AFs (as shown in Sec. 6).

In [10] no solving mechanism is proposed to solve the problems presented in the paper, but the authors define the hard problems and propose their computational complexity proofs that inspired our work in this paper.

In [14] crisp constraint have been used to model argumentation as constraint propagation in *Distributed Constraint Satisfaction Problem* (*DSCP*). Different agents represent the distributed points in the problem. The paper shows the appropriateness of constraints in solving large-scale argumentation systems. However, it seems to only solve classical problems, that is no weighted extensions.

In [4] the authors associates to each subset S of arguments a formula in propositional logic; then, S is an extension under a given semantics if and only if the formula is satisfiable (i.e. they solve the problem with SAT [7]). An extensive survey of the difference between SAT and CP can be found in [7]: summarizing, CP is more expressive for the modeling phase in order to find more complex semantics and further user-defined constraints on classical semantics [8]. In addition, in CP the user has the possibility to inform the solver about problem specific information and then to appropriately tune it, while in SAT there is usually little room and need for this parametrization. Moreover the encoding presented in [4] has no practical implementation.

The are some frameworks based on Logic Programming-like languages. For example, the system *ASPARTIX* [11] is a tool for computing acceptable extensions for a broad range of formalizations of Dung's argumentation framework and generalizations thereof, e.g. value-based AFs [3] or preference-based [1]. *ASPARTIX* relies on a fixed disjunctive datalog program which takes an instance of an argumentation framework as input, and uses the answer-set solver DLV for computing the type of extension specified by the user. However, *ASPARTIX* does not solve any quantitative argumentation case, as well as other Answer Set Programming systems [17]. As far as we know, all these implemented systems are not tested on large interaction graphs as we do in this paper, thus it is not possible to have an idea on how these systems scale over the number of arguments (for example).

In [5] the authors have proposed a semiring-based constraint framework to model AFs with weights on arguments, i.e. withe the nodes of the interaction graph. In [6] the authors have extended [5] in order to solve over-constrained weighted AF problems, where weights are instead associated with arcs and represent the cost of the attack between two arguments. to relax the notion of conflict-free extensions to α-conflict-free ones (and also for the other extensions of Dung), in order to include in the same set also attacking arguments, whose attack costs are not worse than a threshold α.

The conventional model of Coalition Formation considers every possible subset of agents as a potential coalition. However, in many real-world applications, there are inherent constraints on feasible coalitions: for instance, certain agents may be prohibited from being in the same coalition, or the coalition structure may be required to consist of coalitions of the same size. In this paper, we present the first systematic study of Constrained Coalition Formation (CCF) [20, 18]. In [20] the authors use ad-hoc algorithms and no CSP formulation. In [18] the authors use constraints to represent the problem, but they do not represent AFs in their discussion.

8 Conclusions and Future Work

In the paper we have tested how Dung's classical extensions (conflict-free, admissible, complete and stable ones) [9] can be found by using a constraint-based framework. We have presented a mapping from AFs to CSPs and solved the obtained CSP with JaCoP, thus practically finding the solution of the related AF. We have proposed an unifying computational framework with strong mathematical foundations and solving techniques. We have tested the implementation over two different kinds of small-world networks and we have reported the performance in Sec. 5. The performance show the efficiency of our framework and the paper proposes the first computational tests of Dung's AFs applied to interaction graphs with small-world properties. The generation of coalitions given a set of entities and their relationships, in this case according to the constraints defined by Dung, is a challenging problem in literature, due to the rapid explosion of the solutions [20].

As a second result we have shown how to define and program in JaCoP the constraints to model and solve hard problems related to AFs (e.g. the preferred extension) and weighted AFs (e.g. weighted ground extensions). Constraint solving techniques prove to be able to deal with large scale problems, even if the treated problems are difficult [22]. Practical applications of our framework may consist, for example, in automatically study discussion fora [12] where arguments are rated by users, thus, with large sets of arguments.

For the future we have many open issues. First of all, we would like to investigate the properties of interaction graphs, in order to reproduce the tests we have presented in this paper on more appropriate graphs (not randomly generated). In particular we would like to find the small-world properties in real interaction graphs. Moreover, we would like to compare the performance with other systems, as *ASPARTIX* [11].

A further intent is to optimize the performance in order to speed up the search of conflict-free and admissible extensions. We would like to further investigate the search plug-ins provided by JaCoP: the search-plugin is an object, which is informed about the current state of the search and may influence the behavior of the search [15]. Moreover, we would like to directly program ad-hoc methods that can improve the performance during the search in the JaCoP solver .

At last, we want to propose some temporal results also for the implementation of the hard problems shown in Sec. 6.

References

1. L. Amgoud and C. Cayrol. Inferring from inconsistency in preference-based argumentation frameworks. *J. Autom. Reasoning*, 29(2):125–169, 2002.
2. A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.
3. T. J. M. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *J. Log. Comput.*, 13(3):429–448, 2003.

4. P. Besnard and S. Doutre. Checking the acceptability of a set of arguments. In *Workshop on Non-Monotonic Reasoning*, pages 59–64, 2004.
5. S. Bistarelli, D. Pirolandi, and F. Santini. Solving weighted argumentation frameworks with soft constraints. In *Recent Advances in Constraints, 14th Annual ERCIM International Workshop on Constraint Solving and Constraint Logic Programming, CSCLP 2009, Revised Selected Papers*, Lecture Notes in Computer Science, 2009.
6. S. Bistarelli and F. Santini. A common computational framework for semiring-based argumentation systems. In *European Conference on Artificial Intelligence (ECAI10)*, 2010.
7. L. Bordeaux, Y. Hamadi, and L. Zhang. Propositional satisfiability and constraint programming: A comparative survey. *ACM Comput. Surv.*, 38, December 2006.
8. S. Coste-Marquis, C. Devred, and P. Marquis. Constrained argumentation frameworks. In *Knowledge Representation and Reasoning (KR)*, pages 112–122. AAAI Press, 2006.
9. P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–357, 1995.
10. P. E. Dunne, A. Hunter, P. McBurney, S. Parsons, and M. Wooldridge. Inconsistency tolerance in weighted argument systems. In *Conf. on Autonomous Agents and Multiagent Systems*, pages 851–858. IFAAMS, 2009.
11. U. Egly, S. Alice Gaggl, and S. Woltran. ASPARTIX: Implementing argumentation frameworks using answer-set programming. In *International Conference on Logic Programming (ICLP)*, pages 734–738. LNCS, Springer, 2008.
12. T. F. Gordon and N. I. Karacapilidis. The zeno argumentation framework. *KI*, 13(3):20–29, 1999.
13. W. D. Harvey and M. L. Ginsberg. Limited discrepancy search. In *IJCAI (1)*, pages 607–615, 1995.
14. H. Jung, M. Tambe, and S. Kulkarni. Argumentation as distributed constraint satisfaction: applications and results. In *Conference on Autonomous agents (AGENTS)*, pages 324–331, New York, NY, USA, 2001. ACM.
15. K. Kuchcinski and R. Szymanek. Jacop - java constraint programming solver, 2001. http://jacop.osolpro.com/.
16. S. Modgil. Reasoning about preferences in argumentation frameworks. *Artif. Intell.*, 173(9-10):901–934, 2009.
17. J. C. Nieves, U. Cortés, and M. Osorio. Possibilistic-based argumentation: An answer set programming approach. In *Mexican International Conference on Computer Science(ENC)*, pages 249–260. IEEE Computer Society, 2008.
18. N. Ohta, V. Conitzer, R. Ichimura, Y. Sakurai, A. Iwasaki, and M. Yokoo. Coalition structure generation utilizing compact characteristic function representations. In *Principles and Practice of Constraint Programming - CP*, Lecture Notes in Computer Science, pages 623–638. Springer, 2009.
19. J. O'Madadhain, D. Fisher, S. White, and Y. Boey. The JUNG (Java Universal Network/Graph) framework. Technical report, UC Irvine, 2003.
20. T. Rahwan, S. D. Ramchurn, N. R. Jennings, and A. Giovannucci. An anytime algorithm for optimal coalition structure generation. *J. Artif. Int. Res.*, 34:521–567, April 2009.
21. G. Ravid and S. Rafaeli. Asynchronous discussion groups as small world and scale free networks. *First Monday*, 9(9), 2004.
22. F. Rossi, P. van Beek, and T. Walsh. *Handbook of Constraint Programming (Foundations of Artificial Intelligence)*. Elsevier Science Inc., New York, NY, USA, 2006.

An empirical study of a deliberation dialogue system

Elizabeth Black[1] and Katie Bentley[2]

[1] Intelligent Systems Group, Universiteit Utrecht, De Uithof, 3584 CC Utrecht, NL
lizblack@cs.uu.nl
[2] Vascular Biology Lab, London Research Institute, Cancer Research UK, Lincoln's Inn Fields, WC2A 3LY, UK
katie.bentley@cancer.org.uk

Abstract. We present an empirical simulation-based study of the use of value-based argumentation in two-party deliberation dialogues, investigating the impact that argumentation can have on the quality of the outcome reached. Our simulation allows us to vary the number of values, actions and arguments that appear in the system; we investigate how the behaviour of the system changes as these parameters vary. This parameter sensitivity analysis tells us whether a value-based deliberation dialogue system may be useful for a particular real-world application. We measure the quality of the dialogue outcome (i.e. the action that the agents agree to) against a global view of whether that action would be agreeable to each agent if all of the agents' knowledge were taken into account. We compare the deliberation outcome with a simple consensus forming procedure (where no arguments are exchanged). Our results show that the deliberation dialogue system we present outperforms consensus forming.

ACM Category: I.2.11 Multiagent systems. **General terms:** performance, experimentation

Keywords: dialogue, value-based argumentation, simulation, agreement, deliberation.

1 Introduction

There is little work on evaluating whether an argumentation-based approach to a problem is a good approach to take. Most works assume that the decision to use argumentation has already been made and disregard the question of whether there is a better approach to take. We present what we believe to be the first simulation-based study of an argumentation-based deliberation dialogue system, which allows us to start addressing this question and allows us to investigate the effect of varying the parameters of the system.

Simulation is an imperative next step for bridging the gap between argumentation theory and real-world agent applications. Given the complexity of argumentation-based dialogue systems, it is very hard to theoretically investigate their properties without making many restrictive assumptions. In order to gain a full understanding of the behaviour of such systems, theoretical investigations need to be complemented with empirical simulation-based studies. Simulation provides a unique opportunity to generate

large, complex scenarios and analyse their results across thousands of iterations and permutations.

There are few existing works that take a simulation approach to investigating the performance of argumentation-based dialogue systems. Two notable examples are [1, 2]. Each of these focus on a form of argumentation-based negotiation (ABN), where arguments providing reasons for an agent's position are shared; this exchange of information allows the negotiation space to change. In [1], the information exchanged relates to the influence of social commitments between roles, whilst [2] focusses on interest-based negotiation where agents exchange information about their underlying goals and different ways to achieve these.

In [3], ABN is used to address the distributed constraint satisfaction problem. Importantly, the authors have performed experiments with their model to investigate the performance of their argument-based approach. Agents in the system use arguments in the sense that they put forward a proposal and provide a justification for this by giving their local constraints, these constraints are propagated by the receiving agent.

Our deliberation context differs from ABN (which is generally concerned with the allocation of scarce resources), as agents in our system have a shared goal and wish to come to an agreement on how to act in order to achieve that goal. Similarly to the systems discussed above, our agents also share arguments regarding actions to achieve the goal and this allows the set of actions that an agent finds agreeable to change. In our system, however, these arguments are value-based (relating to various social values that may be promoted or demoted by performing an action) and, unlike in [1–3], our agents also use argumentation as the reasoning mechanism with which they determine which actions they find agreeable.

We specifically investigate two questions:

- Do our deliberation dialogues perform better across the entire parameter space than a simple consensus forming approach, where agents try to find an action they each find agreeable without sharing any arguments?
- How does the behaviour of both the dialogue system and the consensus forming mechanism change as the number of arguments, actions and values present in the system varies?

Our results clearly show that the deliberation dialogue system outperforms consensus forming across all parameter combinations. Further, we have identified particular parameter settings that optimise dialogue performance in terms of quality of outcome and length of dialogue. This detailed parameter sensitivity analysis allows a designer of an agent system to evaluate whether value-based deliberation dialogues are useful for their particular application domain.

2 Model

In this section we describe the model that we are simulating. We give details of the value-based argumentation model, the dialogue system, the consensus forming mechanism, the evaluation metric that we use and our experimental set up. The model was written in c++ on a standard workstation. A complete parameter sensitivity analysis of 1.8 million runs took less than an hour to complete.

2.1 Argumentation model

We are investigating the performance of the system formally specified in [4], which is based on the popular argument scheme and critical question approach [5]. Arguments are generated by an agent instantiating a **scheme for practical reasoning** [6]: In the current circumstances R, we should perform action A, which will result in new circumstances S, which will achieve goal G, which will promote value V.

The scheme is associated with a set of characteristic critical questions (CQs) that can be used to identify challenges to proposals for action that instantiate the scheme. An unfavourable answer to a CQ will identify a potential flaw in the argument. Since the scheme makes use of what are termed as 'values', this caters for arguments based on subjective preferences as well as more objective facts. Such values represent qualitative social interests that an agent wishes (or does not wish) to uphold by realising the goal stated [7].

An agent has a **Value-based Transition System** (VATS), that it uses to instantiate the scheme for practical reasoning. This transition system represents the agent's knowledge about the effect of actions and the values that are promoted or demoted. (For brevity, we omit the definition here; the reader is referred to [4].) Given its VATS, an agent can instantiate the practical reasoning argument scheme in order to construct arguments for (or against) actions to achieve a particular goal because they promote (or demote) a particular value. Note that here we are focussing on the **choice of action** stage (as defined in [6]), we assume that any discrepancies between the agents in either the problem formulation or epistemic reasoning stages have been resolved (perhaps with some other type of dialogue); thus, for example, agents do not need to question here whether an action in question does achieve the desired goal or whether a certain set of circumstances hold.

Definition 1: *An* **argument** *constructed by an agent x from its VATS is a 4-tuple $A = \langle a, p, v, s\rangle$ where:*
$s = +$ iff a is an **action** *that will achieve* **goal** *p and will* **promote** *value v;*
$s = -$ iff a is an **action** *that will achieve* **goal** *p but will* **demote** *value v.*
We define the functions: $\mathsf{Act}(A) = a$; $\mathsf{Goal}(A) = p$; $\mathsf{Val}(A) = v$; $\mathsf{Sign}(A) = s$. *If* $\mathsf{Sign}(A) = +(-resp.)$*, then we say A is a* **positive** *(***negative** *resp.) argument* **for** *(***against** *resp.) action a. We denote the* **set of all arguments an agent** *x* **can construct from its VATS** *as $Args^x$; we let $Args^x_p = \{A \in Args^x \mid \mathsf{Goal}(A) = p\}$. The set of* **values** *for a set of arguments $\mathcal{X}$ is defined as* $\mathsf{Vals}(\mathcal{X}) = \{v \mid A \in \mathcal{X}$ *and* $\mathsf{Val}(A) = v\}$.

If we take a particular argument for an action, it is possible to generate attacks on that argument by posing the various CQs related to the practical reasoning argument scheme. The relevant CQs are used to generate a set of arguments for and against different actions to achieve a particular goal, where each argument is associated with a motivating value. To evaluate the status of these arguments we use a Value Based Argumentation Framework (VAF) (introduced in [7]), an extension of the argumentation frameworks (AF) of Dung [8]. In an AF an argument is admissible with respect to a set of arguments S if all of its attackers are attacked by some argument in S, and no argument in S attacks an argument in S. In a VAF an argument succeeds in defeating an argument it attacks if its value is ranked higher than or at least as high as the value of the

argument attacked; a particular ordering of the values is characterised as an **audience**. Arguments in a VAF are admissible with respect to an audience A and a set of arguments S if they are admissible with respect to S in the AF which results from removing all the attacks which are unsuccessful given the audience A. A maximal admissible set of a VAF is known as a **preferred extension**.

Although VAFs are often considered abstractly, here we give an instantiation in which we define the attack relation between the arguments. This attack relation is derived from the CQs, for details the reader is referred to [4].

Definition 2: *An* **instantiated value-based argumentation framework (iVAF)** *is defined by a tuple $\langle \mathcal{X}, \mathcal{A} \rangle$ s.t. $\mathcal{X}$ is a finite set of arguments and $\mathcal{A} \subset \mathcal{X} \times \mathcal{X}$ is the* **attack relation**. *A pair $(A_i, A_j) \in \mathcal{A}$ is referred to as "A_i attacks A_j" or "A_j is attacked by A_i". For two arguments $A_i = \langle a, p, v, s \rangle$, $A_j = \langle a', p', v', s' \rangle \in \mathcal{X}$, $(A_i, A_j) \in \mathcal{A}$ iff $p = p'$ and either: (1) $a = a'$, $s = -$ and $s' = +$; or (2) $a = a'$, $v \neq v'$ and $s = s' = +$; or (3) $a \neq a'$ and $s = s' = +$.*

An **audience** *for an agent x over the values V is a binary relation $\mathcal{R}^x \subset V \times V$ that defines a* total order *over V where exactly one of (v, v'), (v', v) are members of $\mathcal{R}^x$ for any distinct v, $v' \in V$. If $(v, v') \in \mathcal{R}^x$ we say that v is* **preferred to** *v', denoted $v \succ_{\mathcal{R}^x} v'$.We say that an argument A_i is* **preferred to** *the argument A_j in the audience $\mathcal{R}^x$, denoted $A_i \succ_{\mathcal{R}^x} A_j$, iff $\texttt{Val}(A_i) \succ_{\mathcal{R}^x} \texttt{Val}(A_j)$. If R^x is an audience over the values V for the iVAF $\langle \mathcal{X}, \mathcal{A} \rangle$, then $\texttt{Vals}(\mathcal{X}) \subseteq V$.*

We use the term 'audience' to be consistent with the literature. Note, however, audience does not refer to the preference of a *set* of agents; rather, it represents a particular agent's preference over values.

Given an iVAF and a particular agent's audience, we can determine acceptability of an argument as follows. Note that (as in [4]) if an attack is symmetric, then an attack only succeeds in defeat if the attacker is more preferred than the argument being attacked; however, if an attack is asymmetric, then an attack succeeds in defeat if the attacker is at least as preferred as the argument being attacked. Asymmetric attacks occur only when an argument against an action attacks another argument for that action; in this case, if both arguments are equally preferred then we do not wish the argument for the action to withstand the attack. If we have a symmetric attack where the arguments attacking one another are equally preferred, then we must have arguments for two different actions that promote the same value; here, the defeat is not successful, since it is reasonable to choose either action.

Definition 3: *Let $\mathcal{R}^x$ be an audience and let $\langle \mathcal{X}, \mathcal{A} \rangle$ be an iVAF.*
For $(A_i, A_j) \in \mathcal{A}$ s.t. $(A_j, A_i) \notin \mathcal{A}$, A_i **defeats** *A_j under $\mathcal{R}^x$ if $A_j \not\succ_{\mathcal{R}^x} A_i$.*
For $(A_i, A_j) \in \mathcal{A}$ s.t. $(A_j, A_i) \in \mathcal{A}$, A_i **defeats** *A_j under $\mathcal{R}^x$ if $A_i \succ_{\mathcal{R}^x} A_j$.*
An argument $A_i \in \mathcal{X}$ is **acceptable w.r.t** *S under $\mathcal{R}^x$ ($S \subseteq \mathcal{X}$) if: for every $A_j \in \mathcal{X}$ that defeats A_i under $\mathcal{R}^x$, there is some $A_k \in S$ that defeats A_j under $\mathcal{R}^x$.*
A subset S of $\mathcal{X}$ is **conflict-free** *under $\mathcal{R}^x$ if no argument $A_i \in S$ defeats another argument $A_j \in S$ under $\mathcal{R}^x$.*
A subset S of $\mathcal{X}$ is **admissible** *under $\mathcal{R}^x$ if: S is conflict-free in $\mathcal{R}^x$ and every $A \in S$ is acceptable w.r.t S under $\mathcal{R}^x$.*
A subset S of $\mathcal{X}$ is a **preferred extension** *under $\mathcal{R}^x$ if it is a maximal admissible set under $\mathcal{R}^x$.*

An argument A is **acceptable** *in the iVAF $\langle \mathcal{X}, \mathcal{A} \rangle$ under audience $\mathcal{R}^x$ if there is* some *preferred extension containing it.*

We have defined a mechanism with which an agent can determine attacks between arguments for and against actions; it can then use an ordering over the values that motivate such arguments (its audience) in order to determine their acceptability. Next, we define our dialogue system.

2.2 Dialogue system

The dialogue system investigated here is formally defined in [4]. For readability and brevity, we omit the formal definitions here but informally describe the dialogue system. The communicative acts in a dialogue are called **moves**. We assume that there are always exactly two agents (**participants**) taking part in a dialogue, each with its own identifier taken from the set $\mathcal{I} = \{Ag1, Ag2\}$ and each with a knowledge base of arguments that it knows about (those it can construct from its VATS). Each participant takes it in turn to make a move to the other participant. We refer to participants using the variables x and $\overline{x}$ such that: x is $Ag1$ if and only if $\overline{x}$ is $Ag2$; x is $Ag2$ if and only if $\overline{x}$ is $Ag1$.

We assume that the participants have agreed to partake in a deliberation dialogue whose **topic** is the joint goal in question. During the dialogue, agents can either:

- `assert` a positive argument (an argument *for* an action);
- `assert` a negative argument (an argument *against* an action);
- `agree` to an action;
- indicate that they have no arguments that they wish to assert (with a `pass`).

The agents take it in turn to make a single move. A dialogue terminates under one of two conditions: **failure**, when two `pass` moves appear one immediately followed by the other in the dialogue; **success** with **outcome** a, when two moves each agreeing to the action a appear one immediately followed by the other in the dialogue.

In order to evaluate which actions it finds agreeable at a point in a dialogue with topic p, an agent x considers the iVAF that it constructs from all the arguments that it currently has available to it relating to p; this consists of the arguments from its own VATS, as well as the arguments that the other agent has asserted thus far. We call this agent x's **dialogue iVAF**, which is the iVAF $\langle \mathcal{X}, \mathcal{A} \rangle$ where $\mathcal{X} = Args_p^x \cup \{A \mid \overline{x}$ has previously asserted A during the dialogue$\}$. An action is **agreeable** to an agent x if and only if there is some argument *for* that action that is acceptable in x's dialogue iVAF under the audience that represents x's preference over values. Note that the set of actions that are agreeable to an agent may change over the course of the dialogue, due to its dialogue iVAF changing as arguments asserted by $\overline{x}$ are added to it.

The **protocol** defines which moves an agent x (whose turn it is) is allowed to make at any point in a deliberation dialogue with topic p as follows:

- It is permissible to `assert` an argument A iff $\texttt{Goal}(A) = p$ (i.e. the argument is for or against an action to achieve the topic of the dialogue) and A has not been asserted previously during the dialogue.
- It is permissible to `agree` to an action a iff either:
  - the immediately preceding move was an `agree` to the action a, or

    - the other participant $\overline{x}$ has at some point previously in the dialogue asserted a positive argument A for the action a.
- It is always permissible to `pass`.

We have thus defined a protocol that determines which moves it is permissible to make during a dialogue; however, an agent still has considerable choice when selecting which of these permissible moves to make. In order to select one of the permissible moves, an agent uses a particular strategy. The **strategy** that our agents use is as follows:

- If it is permissible to `agree` to an action that the agent finds *agreeable*, then make such an `agree` move; else
- if it is permissible to `assert` a positive argument *for* an action that the agent finds *agreeable*, then assert some such argument; else
- if it is permissible to `assert` a negative argument *against* an action and the agent finds that action *not agreeable* then assert some such argument; else
- make a `pass` move.

We have now defined how our dialogue system regulates the moves that agents may make, and the strategy that the agents use to select one of the permissible moves to make. (For an example of a dialogue produced by this system, please refer to [4].) Next, we define a method with which two agents may form a consensus without exchanging any arguments.

2.3 Consensus forming

In order to start investigating the question of whether it is worth using argumentation-based deliberation dialogues to decide how to act to achieve a shared goal, we compare outcomes produced by our dialogue system with those produced by a simple consensus forming method. For two agents $x, \overline{x}$ who are about to enter into a deliberation dialogue with topic p, the outcome produced by **consensus forming** is simply the *intersection* of the following two sets:

- the set of actions to achieve p that agent x finds agreeable at the start of the dialogue;
- the set of actions to achieve p that agent $\overline{x}$ finds agreeable at the start of the dialogue.

That is to say, the consensus set contains all the actions that each agent finds agreeable, given the arguments they can construct from their VATS and without any exchange of arguments. If consensus forming returns a non-empty set, then we say that a **consensus was found** and that the consensus forming was **successful**.

This gives us a non-argumentative approach to which we can compare our dialogue system. We next discuss how we compare these systems, namely on the quality of outcome.

2.4 Measuring quality of outcome

Unless they exchange all arguments, agents in our system only ever have a partial view of all of the available knowledge. We can, however, take a global view of which potential outcomes are best for each of the agents. For this purpose, we define for a dialogue the **omniscient argumentation framework** (OAF), which is the iVAF constructed from the union of the arguments that each participant can construct from its

VATS that relate to the topic of the dialogue. For a dialogue with participants $x, \overline{x}$ and topic p, the associated OAF is thus the iVAF $\langle \mathcal{X}, \mathcal{A} \rangle$ where $\mathcal{X} = Args_p^x \cup Args_p^{\overline{x}}$. We say that an action is **globally agreeable** to an agent x if and only if there is some *positive* argument *for* that action that is acceptable in the OAF under the audience that represents x's value preference.

We can now measure the quality of a particular outcome (i.e. an action to achieve the goal p) by considering whether it is globally agreeable to each agent. Such a quality measure can be applied to both the outcome produced by a dialogue and the outcome produced by consensus forming.

For a particular outcome a, we assign an **outcome quality score** as follows:

- if a is **globally agreeable to both** x and $\overline{x}$, **score 3**;
- if a is **globally agreeable to only one** of x or $\overline{x}$, **score 2**;
- if a is **not globally agreeable to either** x or $\overline{x}$, **score 1**.

If there is **no successful outcome** (i.e. dialogue terminates in failure or consensus forming returns an empty set) then the outcome quality **score is 0**. Where the consensus forming returns a set of more than one action, we assign the outcome quality score to be that of the action from the set which receives the lowest score (since this is the best that the consensus forming method can guarantee to do, given that only one action can be selected).

Our simple scoring metric reflects the intuition that any outcome is better than no outcome, but an outcome that is globally agreeable to an agent is better than one that is not. We plan to study more sophisticated scoring metrics in future work.

2.5 Experimental set up

The dialogue system and consensus forming mechanism were implemented as described in the previous sections. We also implemented a **random scenario generator**; this generates **scenarios** that initialise the agents' knowledge bases (i.e. the arguments known to each agent at the start of the dialogue, which all relate to the joint goal which the agents wish to achieve) and their audiences. The generator takes three parameters $(\mathsf{Args}, \mathsf{Vals}, \mathsf{Acts})$, where

- Args is the number of distinct arguments to appear in the union of the agents' knowledge bases;
- Vals is the number of distinct values that may motivate those arguments;
- Acts is the number of distinct actions that the arguments may relate to.

The generator randomly constructs without replacement (i.e. does not allow duplicate arguments) the required number of arguments from the allowed values and actions and the symbols $\{+, -\}$ (where each combination is equally likely). For example, when given parameters $(8, 2, 2)$, the generator will construct the following set of arguments:
$\{\langle a1, p, v1, +\rangle, \langle a1, p, v1, -\rangle, \langle a1, p, v2, +\rangle, \langle a1, p, v2, -\rangle,$
$\langle a2, p, v1, +\rangle, \langle a2, p, v1, -\rangle, \langle a2, p, v2, +\rangle, \langle a2, p, v2, -\rangle\}$.

(Note, it is not possible for the generator to construct a set of arguments from parameters $(\mathsf{Args}, \mathsf{Vals}, \mathsf{Acts})$ if $\mathsf{Args} > \mathsf{Vals} \times \mathsf{Acts} \times 2$. For a particular number of values and a particular number of actions, the total **possible arguments** is $\mathsf{Vals} \times \mathsf{Acts} \times 2$.)

The generator randomly assigns each agent an audience over the allowed values and it randomly allocates exactly half of the constructed arguments to one agent, and the other half to the other agent. Our generator is therefore simulating the construction of arguments from the agents' VATS. It allows us to run experiments over all possible combinations of the parameters $(\mathsf{Args}, \mathsf{Vals}, \mathsf{Acts})$. In the experiments reported here we consider all possible parameter combinations where:

- $\mathsf{Vals} \in \{2, 4, 6, 8, 10\}$,
- $\mathsf{Acts} \in \{2, 4, 6, 8, 10\}$,
- $2 \leq \mathsf{Args} \leq \mathsf{Vals} \times \mathsf{Acts} \times 2$.

Our experiments investigate how the outcome quality scores of the dialogue system and the consensus forming mechanism compare across the space of possible parameter combinations. We performed 1000 runs of our simulation for each possible parameter combination. In each run, a random scenario is generated. We first calculate the consensus set of the scenario and then simulate a dialogue from the same scenario; we compare the quality scores assigned to the outcomes produced by these two approaches.

3 Results

3.1 Dialogue is significantly more likely to be successful than consensus forming

Figure 1 shows strikingly across all parameter combinations that the frequency of successful consensuses is never as great as the frequency of successful dialogues. There is a significant difference between these two frequencies: across all parameters, consensus forming fails more than 50% of the time, whilst up to 90% of dialogues are successful.

We also found that, across all runs for each possible parameter combination (a total of 1.8 million runs), for every run in which a consensus was found the dialogue produced was also successful. It is not immediately clear whether the converse situation (i.e. a consensus is found but the dialogue produced is not successful) is theoretically impossible, but this result strongly suggests that this may be the case and so identifies a property worthy of theoretical investigation.

Consensus forming is relatively robust to the number of values present in the system; however there is a marked difference when $\mathsf{Vals} = 2$, in which case the frequency of successful consensuses is approximately half that of when $\mathsf{Vals} \in \{4, 6, 8, 10\}$.

When $\mathsf{Acts} = 2$, the highest frequency of consensuses found is seen when Args is equal to approximately 50% of the total arguments possible. A higher number of arguments present in the system leads to a higher frequency of successful consensuses; in contrast, the frequency of successful dialogues drops as the number of arguments present in the system increases (although the number of successful dialogues is still greater than the number of consensuses found).

3.2 Successful dialogues are more likely with higher numbers of actions and values

Looking at the top of Figure 1 in depth, we can see how sensitive the dialogue system is to the parameters. The dialogue system appears to be most sensitive to the parameter settings $\mathsf{Acts} = 2$ and $\mathsf{Vals} = 2$.

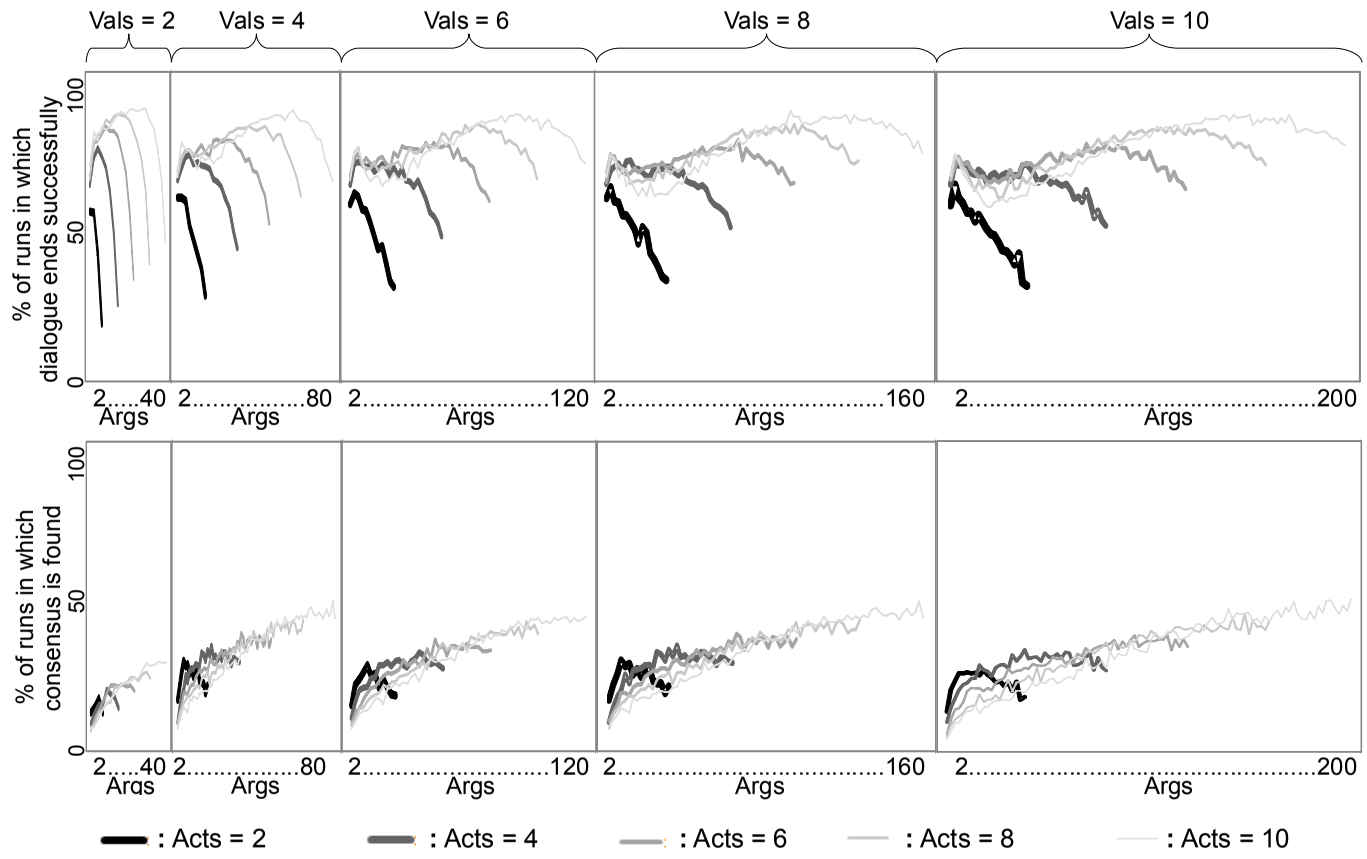


Fig. 1. Top: Percentage of dialogues that ended successfully out of 1000 runs across each possible parameter combination. Bottom: Percentage of 1000 runs across each possible parameter combination in which a consensus was found.

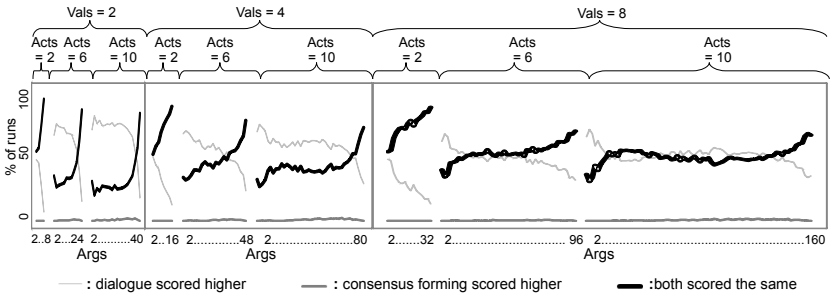


Fig. 2. Percentage of 1000 runs across each possible parameter combination where $\mathsf{Acts} \in \{2, 6, 10\}$ and $\mathsf{Vals} \in \{2, 4, 8\}$ in which: dialogue outcome quality score was higher than consensus outcome quality score; consensus outcome quality score was higher than dialogue outcome quality score; dialogue outcome quality score was the same as consensus outcome quality score.

Across all parameter settings, the frequency of successful dialogues is closely related to the percentage of the total possible arguments present in the system: if $\mathsf{Acts} \in \{4, 6, 8, 10\}$ and $\mathsf{Vals} \in \{4, 6, 8, 10\}$, this frequency peaks when Args is around 75% of the total possible; if $\mathsf{Acts} = 2$, this frequency peaks when $\mathsf{Args} \approx 4$; if $\mathsf{Acts} \in \{4, 6, 8, 10\}$ and $\mathsf{Vals} = 2$, this frequency peaks when Args is around 50% of the total possible.

When $\mathsf{Acts} = 2$, the highest frequency of successful dialogues seen is lower than the highest frequencies seen for the other settings of Acts. Both the maximum and the minimum frequency of successful dialogues recorded is greater when more actions are under consideration, and the minimum frequency of successful dialogues is greater when more values are present in the system.

Generalising these results, we can say that the dialogue system performs better (i.e. reaches agreement more often) when $\mathsf{Acts} \neq 2$. The more values and the more actions present in the system the better the system performs, with the frequency of successful dialogues dependent on the percentage of the total possible arguments present in the system.

3.3 Quality of dialogue outcome is very rarely worse than quality of consensus outcome

We next consider for each run whether the dialogue system or consensus forming resulted in a higher outcome quality score. We investigated this across all possible parameter combinations; since we found a trend that repeats across the whole parameter space, we present in Figure 2 only the results for when $\mathsf{Acts} \in \{2, 6, 10\}$ and $\mathsf{Vals} \in \{2, 4, 8\}$.

Figure 2 shows clearly that only very rarely (in less than 3% of the runs across all possible parameter settings) does consensus forming produce a higher quality outcome than the dialogue system. However, if there are only two actions, then the two methods produce the same quality outcome more often than the dialogue system produces a higher quality outcome. This is a useful observation, particularly considering the higher computational overheads associated with the dialogue system.

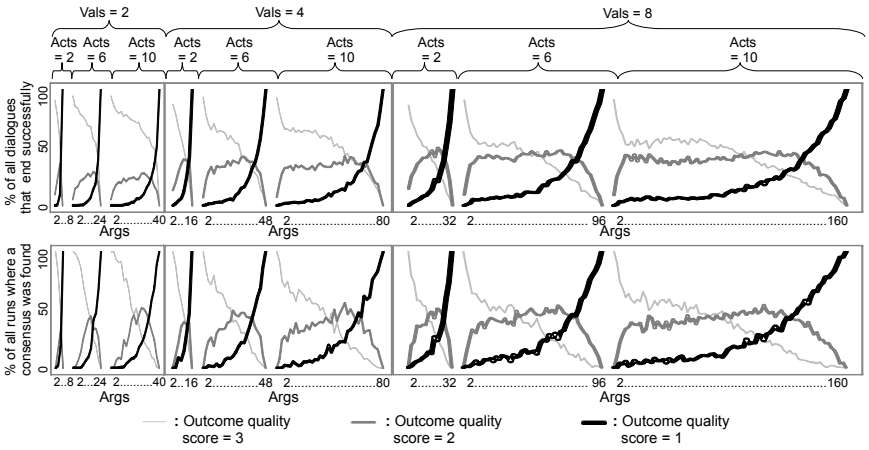


Fig. 3. Top: across all possible parameter settings where Acts $\in \{2, 6, 10\}$ and Vals $\in \{2, 4, 8\}$, percentage *of the dialogues that ended successfully* that received each outcome quality score. Bottom: across all possible parameter settings where Acts $\in \{2, 6, 10\}$ and Vals $\in \{2, 4, 8\}$, percentage of the runs *in which a consensus was found* that received each outcome quality score.

3.4 Successful dialogue outcomes are more likely to be globally agreeable to both agents than successful consensus outcomes

We now consider how the outcome quality score varies for successful outcomes produced by both the dialogue system and consensus forming across the parameter space. We performed this analysis across all possible parameter settings and found a trend that occurs across the entire parameter space; hence we present in Figure 3 only those results where where Acts $\in \{2, 6, 10\}$ and Vals $\in \{2, 4, 8\}$. The top of this figure shows what percentage *of the dialogues that ended in agree* received which outcome quality score. The bottom of this figure shows what percentage *of the runs in which a consensus was found* received which outcome quality score. (Recall the outcome quality score metric: 3 - outcome is globally agreeable to both agents; 2 - outcome is globally agreeable to only of the agents; 1 - outcome is not globally agreeable to either of the agents.)

As discussed earlier, Figure 1 shows that the frequency of dialogues that end successfully is considerably higher than the frequency of consensuses found, and that each of these frequencies vary as the parameters change; thus, it is important to bear in mind here that the percentages denoted on the y-axes of the graphs in Figure 3 relate to different sized sets depending on the particular parameter settings and on whether dialogue outcome or consensus outcome is being considered. Considering only the proportion of *successful* dialogues and consensuses that receive the different outcome quality scores (as seen in Figure 3) allows us to clearly see the following points.

Of the successful outcomes produced by both methods (consensus forming and the dialogue system), a higher proportion of those produced by the dialogue system are globally agreeable to each agent (i.e. outcome quality score = 3). The difference between the proportion of successful dialogues that receive outcome quality score 3 and

the proportion of consensuses that receive outcome quality score 3 is bigger the more actions and the fewer values that are present in the system.

It is interesting to note that the points on the graphs in Figure 3 where the green line (i.e. outcome quality score = 1) and the red line (i.e. outcome quality score = 2) intersect occur at the same position on the x-axis for both the dialogue outcome and the consensus outcome. If $\mathsf{Vals} = 2$, this occurs when Args is equal to approximately 95% of the total possible arguments, otherwise this occurs when Args is equal to approximately 80% of the total possible arguments. Thus, if a successful outcome is produced either by consensus forming or by the dialogue system and there are more than 80% of the total possible arguments present in the system (95% if $\mathsf{Vals} = 2$), it is likely that this outcome is not globally agreeable to either agent.

The quality of successful outcomes produced by both the dialogue system and consensus forming is most sensitive to the number of arguments present in the system, and is little affected by changes to the number of values or actions under consideration. Consensus forming is more sensitive than the dialogue system to the number of arguments.

3.5 Average dialogue outcome quality score is higher than average consensus outcome quality score

Figure 4 shows the average outcome quality score produced by both the dialogue system and the consensus forming mechanism across all parameter settings where $\mathsf{Args} = 25\%$, 50% and 75% of the total possible arguments. It is very clear from these results that, on average, the dialogue system outperforms consensus forming.

Looking at Figure 4 in more depth, we see that the highest outcome quality score averages for the dialogue system are seen when $\mathsf{Vals} = 2$, whilst this parameter setting produces the lowest outcome quality score averages for consensus forming. For all settings of Acts and Vals, the smallest difference between the outcome quality score averages of the two methods is seen when $\mathsf{Args} = 75\%$ of the total possible arguments. For all settings of Vals and Args, the smallest difference between the two outcome quality score averages is seen when $\mathsf{Acts} = 2$. We can conclude that if $\mathsf{Vals} = 2$ and $\mathsf{Acts} \neq 2$, it is likely that the outcome produced by the dialogue system will be higher quality than that produced by consensus forming.

3.6 Dialogue length grows exponentially with increasing arguments

Figure 5 shows that the time it takes to complete dialogues increases exponentially with the number of arguments. However as the number of values increases this trend flattens and increases are more linear. Indeed as values and actions increase the curve becomes almost sigmoidal. This indicates that if speed is a key factor for an applied dialogue system, deliberation dialogues are most useful when either the number of arguments is low or the number of values and actions is high.

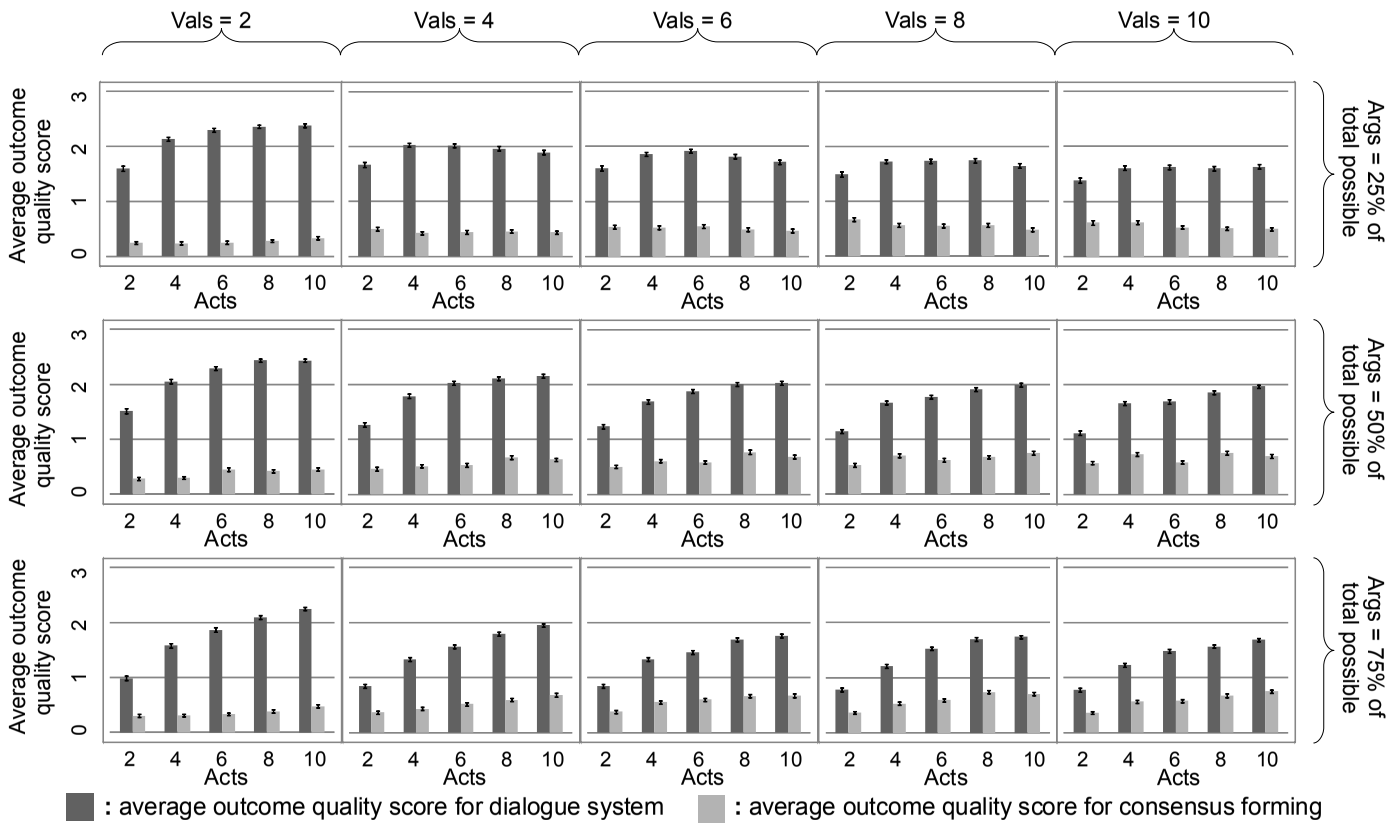


Fig. 4. Average quality outcome score over 1000 runs for both the dialogue system and consensus forming, across every parameter combination where Args = 25%, 50% or 75% of the total possible arguments. The error bars show the standard errors of the means.

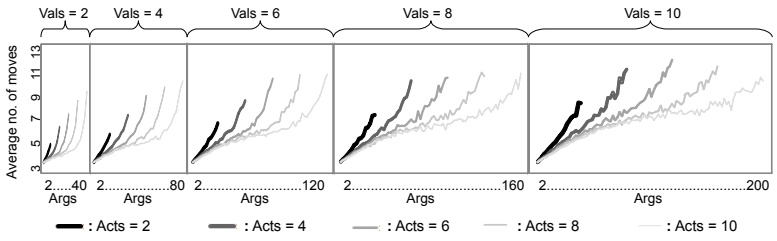


Fig. 5. Average number of moves in a terminated dialogue.

4 Discussion

We have presented empirical results from what we believe is the first simulation-based study of a deliberation dialogue system, where the agents involved used value-based argumentation to determine agreeable actions. Our results show that the dialogue system we present outperforms a simple consensus forming mechanism. We provide an in-depth analysis of the behaviour that can be expected from the system based on the number of actions, values and arguments that are present. For instance, the dialogue system reaches agreement more frequently when there is a higher number of actions and values under consideration; the quality of a successful dialogue outcome is more likely to be higher when there are less than 80% of the total possible arguments present.

These results take a significant step towards demonstrating the applicability of value-based deliberation dialogue systems, as well as demonstrating the importance of complementing theoretical evaluations with simulation-based studies. Our specific quantitative results can be compared against the parameters derived from a particular domain in order to determine the suitability of value-based deliberation dialogues.

Our simulation facilitates many avenues of future work, for example it is simple to adapt it to allow multiple agents and we are particularly interested in investigating different strategies that the agents might use and seeing how these compare with one another. Our next step is to analyse why the system behaves as it does. We have already begun to investigate how the topology of the OAF (which is itself determined by the combination of parameters) affects the dialogue behaviour, and it is clear that they are closely linked. Here, we have restricted the system so that the agents each get exactly half of the arguments present in the system; certainly altering this split will have a marked effect of the behaviour of the system and this is something we are keen to investigate. We also intend to extend our dialogue model to take into account the other stages of practical reasoning (problem formulation and epistemic reasoning [6]).

It would be very interesting to see how an argumentative agent would perform against a non-argumentative agent, such as one that uses classical decision theory to determine the actions it finds agreeable. There is a large body of work on computational social choice (see e.g. [9]), which considers mechanisms with which group decisions can be made. Although beyond the scope of this paper, we plan to compare deliberation dialogues with social choice mechanisms (more sophisticated that the simple consensus forming method presented here). Such comparisons of an argumentation-based

approach with approaches from other fields are of vital importance if we are to demonstrate the value of argumentation theory to the wider field of Artificial Intelligence.

Our investigation here takes a fundamental first step towards evaluating the potential benefit of a value-based deliberation dialogue system; however, it is not clear whether the scenarios that our simulation randomly generates are reflected in any real world setting. For example: Are there any real applications where more than 75% of all possible arguments are present in the system? Is it realistic that negative arguments are as likely to appear within the system as positive arguments? In order to be sure that the results are useful beyond a randomised setting, it is important to test argumentation-based approaches using real world data. This presents a challenge for the community, since it is hard to get access to such data that can be represented as arguments. We plan to collaborate with researchers working on real applications in order to validate our approach.

This simulation has been invaluable in identifying areas of future work that have the potential to be of benefit to real world applications, and in providing us with an implemented framework that we can adapt to investigate these areas.

Acknowledgements E. Black funded by the European Union Seventh Framework Programme (FP7/2007-2011) under grant agreement 253911. K. Bentley funded by the Fondation Leducq.

References

1. Karunatillake, N.C., Jennings, N.R., Rahwan, I., McBurney, P.: Dialogue games that agents play within a society. Artificial Intelligence **173**(9-10) (2009) 935–981
2. Pasquier, P., Hollands, R., Rahwan, I., Dignum, F., Sonenberg, L.: An empirical study of interest-based negotiation. Autonomous Agents and Multi-Agent Systems **22**(2) (2011) 249–288
3. Jung, H., Tambe, M.: Towards argumentation as distributed constraint satisfaction. In: Proc. of AAAI Fall Symposium on Negotiation Methods for Autonomous Cooperative Systems. (2001)
4. Black, E., Atkinson, K.: Choosing persuasive arguments for action. In: Proc. of the 10th Int. Conf. on Autonomous Agents and Multi-Agent Systems. (2011) 905–912
5. Walton, D.N.: Argumentation Schemes for Presumptive Reasoning. Lawrence Erlbaum Associates, Mahwah, NJ, USA (1996)
6. Atkinson, K., Bench-Capon, T.J.M.: Practical reasoning as presumptive argumentation using action based alternating transition systems. Artificial Intelligence **171**(10–15) (2007) 855–874
7. Bench-Capon, T.J.M.: Agreeing to differ: Modelling persuasive dialogue between parties without a consensus about values. Informal Logic **22**(3) (2002) 231–245
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** (1995) 321–357
9. Chevaleyre, Y., Endriss, U., Lang, J., Maudet., N.: A short introduction to computational social choice. In: Proc. of the 33rd Conf. on Current Trends in Theory and Practice of Computer Science, LNCS 4362. (2007) 51–69

Quantifying Disagreement in Argument-based Reasoning

Richard Booth[1], Martin Caminada[1], Mikołaj Podlaszewski[1], and Iyad Rahwan[2]

[1] University of Luxembourg
{richard.booth, martin.caminada, mikolaj.podlaszewski}@uni.lu
[2] Masdar Institute of Science and Technology, Abu Dhabi, UAE, and
University of Edinburgh, UK
irahwan@acm.org

Abstract. An argumentation framework can be seen as expressing, in an abstract way, the conflicting information of an underlying logical knowledge base. This conflicting information often allows for the presence of more than one possible reasonable position (extension/labelling) which one can take. A relevant question, therefore, is how much these positions differ from each other. In the current paper, we will examine the issue of how to define meaningful measures of distance between the (complete) labellings of a given argumentation framework. We provide concrete distance measures based on argument-wise label difference, as well as based on the notion of critical sets, and examine their properties.

1 Introduction

Given a conflicting logical theory, an agent is faced with the problem of deciding what it could reasonably believe. As advocated in various nonmonotonic inference formalisms such as default logic [26], it is often possible to identify *multiple* reasonable positions, or so-called *extensions*. This idea has been adopted in abstract argumentation theory [15], which attempts to analyze possible extensions while abstracting away from the underlying logic. In particular, this theory views logical derivations as abstract arguments (nodes in a graph), and conflicts as defeat relations (directed arcs) over these arguments.

The presence of multiple reasonable positions raises a fundamental question: *how different are two given evaluations of a conflicting logical theory?* We attempt to answer this question in the context of abstract argumentation theory.

This question is relevant to two fundamental problems. The first problem is *argument-based belief revision*. Suppose a diplomat receives instructions to switch his position on one particular argument (e.g. "Mubarak should stay in power until September, because it is good for stability"). To maintain a consistent viewpoint, the diplomat must revise his evaluation of other related arguments. Faced with multiple possibilities, the diplomat may wish to choose the one that differs the least from his initial position (e.g. to maintain credibility).

The issue of distance is also relevant to the problem of *judgement aggregation* over how a given set of arguments should be evaluated collectively by a group of agents with different opinions [9, 10, 25]. For instance it is very well possible that the members

of a jury in a criminal trial all share the same information on the case (and hence have the same argumentation framework) but still have different opinions on what the verdict should be. Hence, these differences of opinion are consequences not of differences in the knowledge base but of the nature of nonmonotonic reasoning, which allows for various reasonable positions (extensions). In the context of judgement aggregation one may examine the extent to which the collective position differs from the various positions of the individual participants. Ideally, one would like to have a collective position that is closest to the collection of individual positions, for example such that the sum of its distance to each individual position is minimal.

In this paper, we examine a number of possible candidates for measuring the *distance* between different labellings (evaluations) of an argumentation graph. The paper advances the state-of-the-art in argument-based reasoning in three ways: (1) We provide the first systematic investigation of quantifying the distance between two evaluations of an argument graph; (2) We propose a number of intuitive measures and show that they fail to satisfy basic desirable postulates; (3) we come up with a measure that satisfies them all. In addition to providing many answers, our paper also raises many interesting questions to the community at the intersection between argumentation and social choice.

2 Abstract Argumentation and Labellings

In this section, we briefly restate some preliminaries regarding argumentation theory. For simplicity, we only consider finite argumentation frameworks.

Definition 1. *An* argumentation framework *(AF for short) is a pair* $\mathcal{A} = (Ar, \rightharpoonup)$*, where* Ar *is a finite set of arguments and* $\rightharpoonup \subseteq Ar \times Ar$*.*

We say that argument A *attacks* argument B iff $(A, B) \in \rightharpoonup$. An AF can be represented as a directed graph in which the arguments are represented as nodes and the attack relation is represented as arrows.

In the current paper, we follow the approach of [3, 8] in which the semantics of abstract argumentation is expressed in terms of *argument labellings*. The idea is to distinguish between the arguments that one accepts (that are labelled `in`), the arguments that one rejects (that are labelled `out`) and the arguments which one abstains from having an opinion about (that are labelled `undec` for "undecided").

Definition 2. *Given an AF* $\mathcal{A} = (Ar, \rightharpoonup)$*, a labelling for* $\mathcal{A}$ *is a function* $\mathcal{L} : Ar \rightarrow \{\texttt{in}, \texttt{out}, \texttt{undec}\}$.

Since a labelling is a function, it can be represented as a set of pairs, each consisting of an argument and a label (`in`, `out`, or `undec`).

We are now ready to state the concept of *complete labelling* [3, 8].

Definition 3 ([3]). *Let* $\mathcal{L}$ *be a labelling of AF* $\mathcal{A} = (Ar, \rightharpoonup)$*.* $\mathcal{L}$ *is a complete labelling (over* $\mathcal{A}$*) iff for each* $A \in Ar$ *it holds that:*

1. $\mathcal{L}(A) = \texttt{in}$ *iff* $\forall B \in Ar : (B \rightharpoonup A \supset \mathcal{L}(B) = \texttt{out})$
2. $\mathcal{L}(A) = \texttt{out}$ *iff* $\exists B \in Ar : (B \rightharpoonup A \wedge \mathcal{L}(B) = \texttt{in})$.

We denote the set of all complete labellings of $\mathcal{A}$ by $Comp_{\mathcal{A}}$.

Defn. 3 enables argumentation to be described to end-users in a non-technical way. An argument is accepted iff all attackers are rejected, and an argument is rejected iff at least one attacker is accepted.

As stated in [3, 8], complete labellings coincide with complete extensions in the sense of [15]. Moreover, the relationship between them is one-to-one. In essence, a complete extension is simply the `in`-labelled part of a complete labelling [3, 8].

The labelling approach has also been defined for other semantics, such as grounded, preferred, stable and semi-stable semantics [3, 8], as well as for ideal semantics [9]. In this paper, however, we will focus on the case of complete semantics and the associated complete labellings, not only because of their relative simplicity, but also because complete labellings serve as the basis for defining labellings for various other semantics [3, 8]. That is, semantics like grounded [15], preferred [15], stable [15], semi-stable [4], ideal [14] and eager [5] in essence select subsets of the set of all complete labellings [8, 3, 6]. Since the approach in the current paper is to compare any arbitrary pair of complete labellings, our results are directly applicable also to the aforementioned semantics.[3]

Example 1. Consider a simple argumentation framework $\mathcal{A} = (Ar, \rightharpoonup)$ with $Ar = \{A, B, C\}$ and $\rightharpoonup = \{(A, B), (B, A), (B, C)\}$. Then $Comp_{\mathcal{A}} = \{\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3\}$, where each $\mathcal{L}_i$ may be visualised in Fig. 1. In this and subsequent diagrams, a node with a solid line indicates an `in` label, a dotted line indicates `out` and a grey node indicates `undec`. Thus for example the first labelling $\mathcal{L}_1 = \{(A, \texttt{in}), (B, \texttt{out}), (C, \texttt{in})\}$.

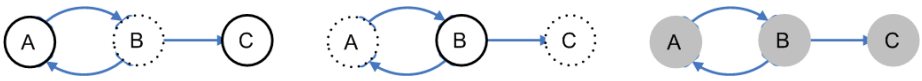


Fig. 1. Three possible complete labellings $\mathcal{L}_1, \mathcal{L}_2$ and $\mathcal{L}_3$

3 Distance between Complete Labellings

The problem we are interested in in this paper is the following:

> Given an AF $\mathcal{A}$, and given two complete labellings $\mathcal{S}$ (the *source* labelling) and $\mathcal{T}$ (the *target* labelling) over $\mathcal{A}$, how can we quantify the *distance* from $\mathcal{S}$ to $\mathcal{T}$, denoted $d(\mathcal{S}, \mathcal{T})$?

Of course we don't just want a method which applies to only one AF, we want a method to be able to do this for *any* given $\mathcal{A}$.

[3] Another point to mention is that it has been proved that complete-based semantics (that is, semantics whose sets of extensions/labellings are subsets of the set of all complete extensions/labellings), when used for the purpose of logical inference, tend to produce fully instantiated argumentation formalisms that satisfy reasonable properties in the sense of [7, 24].

Definition 4. *A* labelling distance *(for AF $\mathcal{A}$) is a function $d : Comp_{\mathcal{A}} \times Comp_{\mathcal{A}} \to \mathbb{N}$. A* labelling distance method *is a function which assigns to every AF $\mathcal{A}$ a labelling distance for $\mathcal{A}$.*

In the following sections we will provide a few concrete definitions of distance functions. But first, are there any properties which we should expect such a function to satisfy? In mathematics, when formalising the notion of distance it is common to require that d be a *metric* distance. Recall that a metric distance over a set Θ is a function $m : \Theta^2 \to \mathbb{N}$ which satisfies, for all $a, b, c \in \Theta$:

(m1) $m(a, a) = 0$
(m2) $m(a, b) > 0$ if $a \neq b$
(m3) $m(a, b) = m(b, a)$ (Symmetry)
(m4) $m(a, c) \leqslant m(a, b) + m(b, c)$ (Triangle Inequality)

Also, let's define the following binary relation on the set of all complete labellings, given a fixed source complete labelling $\mathcal{S}$:

$$\mathcal{T}_1 \leqslant_{\mathcal{S}} \mathcal{T}_2 \text{ iff } \forall A \, (\mathcal{T}_1(A) \neq \mathcal{S}(A) \supset \mathcal{T}_2(A) = \mathcal{T}_1(A))$$

$\mathcal{T}_1 \leqslant_{\mathcal{S}} \mathcal{T}_2$ means that every argument that $\mathcal{T}_1$ labels differently to $\mathcal{S}$, is labelled equally differently by $\mathcal{T}_2$. Thus $\mathcal{T}_2$ differs from $\mathcal{S}$ at least as much as $\mathcal{T}_1$ does. It can be shown that $\leqslant_{\mathcal{S}}$ is a partial order over $Comp_{\mathcal{A}}$ with minimum element $\mathcal{S}$, i.e., $\mathcal{S} \leqslant_{\mathcal{S}} \mathcal{T}$ for all $\mathcal{T} \in Comp_{\mathcal{A}}$. Let $<_{\mathcal{S}}$ denote the strict version of $\leqslant_{\mathcal{S}}$, i.e., $\mathcal{T}_1 <_{\mathcal{S}} \mathcal{T}_2$ iff both $\mathcal{T}_1 \leqslant_{\mathcal{S}} \mathcal{T}_2$ and $\mathcal{T}_2 \not\leqslant_{\mathcal{S}} \mathcal{T}_1$. Thus the following might seem to be a reasonable requirement on a distance function d:

(m5) If $\mathcal{T}_1 <_{\mathcal{S}} \mathcal{T}_2$ then $d(\mathcal{S}, \mathcal{T}_1) < d(\mathcal{S}, \mathcal{T}_2)$ (Disagreement monotony)

To see why this might be reasonable, note that $\mathcal{T}_1 <_{\mathcal{S}} \mathcal{T}_2$ means that for every argument on which $\mathcal{T}_1$ disagrees with $\mathcal{S}$, the labelling $\mathcal{T}_2$ disagrees with $\mathcal{S}$ in exactly the same way, but that there exists at least one argument on which $\mathcal{T}_2$ disagrees with $\mathcal{S}$, but for which $\mathcal{T}_1$ and $\mathcal{S}$ agree. In this case it seems as though $\mathcal{T}_2$ is making strictly more changes to $\mathcal{S}$ than $\mathcal{T}_1$ is, and so d should also endorse this conclusion. It is not difficult to show that if d satisfies both **(m1)** and **(m5)** then it satisfies **(m2)**.

4 Distance via argument-wise label difference

Our first family of distance functions is about simply finding the raw quantity of disagreement between two complete labellings. We can do this in terms of difference between labels, that is, we assume we have some measure of disagreement $diff(\mathtt{x}, \mathtt{y})$ for any $\mathtt{x}, \mathtt{y} \in \{\mathtt{in}, \mathtt{out}, \mathtt{undec}\}$ between the different labels, and then obtain the distance between two labellings by summing the differences between *all* arguments in the AF $(Ar, \rightharpoonup)$ under consideration, i.e., take

$$d(\mathcal{S}, \mathcal{T}) = \sum_{A \in Ar} diff(\mathcal{S}(A), \mathcal{T}(A)). \tag{1}$$

Definition 5. *If the function d can be defined from some function $diff : \{\mathtt{in}, \mathtt{out}, \mathtt{undec}\}^2 \to \mathbb{N}$ as in (1) then we say d is a* simple diff-based distance.

Given this general definition, can we already say which postulates from Section 3 will be satisfied by such a d? It is not difficult to prove that if *diff* is itself a metric distance over $\{\texttt{in}, \texttt{out}, \texttt{undec}\}$ then d will be a metric distance over $Comp_{\mathcal{A}}$, and furthermore **(m5)** will be satisfied. More precisely we have the following result, which is actually a corollary of a more general result (Prop. 5) given in Section 5.1.

Proposition 1. *Let d be defined via diff as in (1). For each $i = 1, 2, 3, 4$, if diff satisfies* **(m**i**)** *then d satisfies* **(m**i**)** *too. If diff satisfies* **(m1)** *and* **(m2)** *then d satisfies* **(m5)**.

How can we define *diff*? In the rest of this section we will present three possibilities.

4.1 First idea: Hamming distance

The first idea is to just count the number of arguments which get different labels. This idea is familiar from the Hamming distance, which has already been employed in propositional belief revision [12].

$$diff_H(\mathtt{x}, \mathtt{y}) \stackrel{\text{def}}{=} \begin{cases} 1 \text{ if } \mathtt{x} \neq \mathtt{y} \\ 0 \text{ otherwise} \end{cases}$$

We denote by d_H the distance obtained from $diff_H$ via (1). For instance if we look at Ex. 1 the distance d_H between any pair of labellings from $\mathcal{L}_1$, $\mathcal{L}_2$ and $\mathcal{L}_3$ is equal to 3. It is commonly known that the Hamming distance forms a metric distance, and so, by Prop. 1:

Proposition 2. *d_H satisfies* **(m1)**-**(m5)**.

Although applying the Hamming distance is a relatively simple and straightforward way of satisfying **(m1)**-**(m5)**, it also has its disadvantages. Consider for instance the results for Ex. 1, where we see that $d_H(\mathcal{L}_1, \mathcal{L}_2) = 3 = d_H(\mathcal{L}_1, \mathcal{L}_3)$. Thus, according to d_H, labellings $\mathcal{L}_2$ and $\mathcal{L}_3$ are equidistant from $\mathcal{L}_1$. However it might be thought that the change between $\mathcal{L}_1$ and $\mathcal{L}_2$ is more "drastic" than that between $\mathcal{L}_1$ and $\mathcal{L}_3$, since it involves a complete swing in the status of its arguments from `in` (resp. `out`) to `out` (resp. `in`). Shouldn't the difference between `in` and `out` be strictly greater than the difference between `in` (or `out`) to `undec`? In other words we might expect the following property to hold for *diff*:

(diff +) If at least one of $\mathtt{x}, \mathtt{y}$ is `undec` then
$diff(\mathtt{x}, \mathtt{y}) < diff(\mathtt{in}, \mathtt{out})$ and $diff(\mathtt{x}, \mathtt{y}) < diff(\mathtt{out}, \mathtt{in})$

If d is a diff-based distance defined from a *diff* satisfying **(diff+)** then d will satisfy a property which is stronger than **(m5)**. To express this property we first define the following ordering over $Comp_{\mathcal{A}}$, given any source labelling $\mathcal{S}$ and target labellings $\mathcal{T}_1, \mathcal{T}_2$:

$$\begin{aligned} \mathcal{T}_1 \leqslant^b_{\mathcal{S}} \mathcal{T}_2 \text{ iff either } & \mathcal{T}_1(A) = \mathcal{S}(A) \\ \text{or } & \mathcal{T}_1(A) = \mathcal{T}_2(A) \\ \text{or } & [\mathcal{T}_1(A) = \mathtt{undec} \text{ and } \mathcal{S}(A) \neq \mathcal{T}_2(A)]. \end{aligned}$$

The superscript "b" on $\leqslant^b_{\mathcal{S}}$ may be thought of as standing for "*between*", since $\mathcal{T}_1 \leqslant^b_{\mathcal{S}} \mathcal{T}_2$ is merely expressing that, for all $A \in Ar$, $\mathcal{T}_1(A)$ lies on a path *between* $\mathcal{S}(A)$ and $\mathcal{T}_2(A)$, assuming the natural neighbourhood graph $\mathtt{in} - \mathtt{undec} - \mathtt{out}$ over the labels.

Proposition 3. *Let d be defined via diff as in (1). If diff satisfies* **(m1)**, **(m2)** *and* **(diff+)** *then d satisfies the following property:*

(m5+) *If $\mathcal{T}_1 <^b_{\mathcal{S}} \mathcal{T}_2$ then $d(\mathcal{S},\mathcal{T}_1) < d(\mathcal{S},\mathcal{T}_2)$* *(Betweenness monotony)*

where $<^b_{\mathcal{S}}$ is the strict part of the relation $\leqslant^b_{\mathcal{S}}$

As with Prop. 1, this result is actually a corollary of Prop. 5 in Section 5.1. It is straightforward to see that $<_{\mathcal{S}} \subseteq <^b_{\mathcal{S}}$, and so **(m5+)** is indeed a strengthening of **(m5)**.

Clearly the distance d_H does not satisfy **(m5+)**, since in Ex. 1 we have $\mathcal{L}_3 <^b_{\mathcal{L}_1} \mathcal{L}_2$ but $d_H(\mathcal{L}_1,\mathcal{L}_3) = d_H(\mathcal{L}_1,\mathcal{L}_2)$. Our next distance measure aims to respect this intuition.

4.2 Second idea: Refined Hamming distance

We first assign the following numeric values to the three possible argument labels.

$$val(\texttt{in}) = 1, val(\texttt{undec}) = 0, \text{and } val(\texttt{out}) = -1.$$

and then take the absolute value of the difference of the labels, i.e., take $diff_{rh}(\texttt{x},\texttt{y}) = |val(\texttt{x}) - val(\texttt{y})|$ (where $diff_{rh}$ can be read as **r**efined **h**amming distance), so:

$diff_{rh}(\texttt{x},\texttt{x}) = 0$ for all $\texttt{x} \in \{\texttt{in},\texttt{undec},\texttt{out}\}$
$diff_{rh}(\texttt{in},\texttt{out}) = diff_{rh}(\texttt{out},\texttt{in}) = 2$
$diff_{rh}(\texttt{in},\texttt{undec}) = diff_{rh}(\texttt{undec},\texttt{in}) = 1$
$diff_{rh}(\texttt{out},\texttt{undec}) = diff_{rh}(\texttt{undec},\texttt{out}) = 1$

Alternatively $diff_{rh}(\texttt{x},\texttt{y})$ may be thought of as the shortest path between $\texttt{x}$ and $\texttt{y}$ in the neighbourhood graph $\texttt{in} - \texttt{undec} - \texttt{out}$ over the labels. We denote by d_{rh} the distance obtained by plugging in $diff_{rh}$ into (1). Going back to Ex. 1, we have $d_{rh}(\mathcal{L}_1,\mathcal{L}_2) = diff_{rh}(\texttt{in},\texttt{out}) + diff_{rh}(\texttt{out},\texttt{in}) + diff_{rh}(\texttt{in},\texttt{out}) = 2+2+2 = 6$ and $d_{rh}(\mathcal{L}_1,\mathcal{L}_3) = diff_{rh}(\texttt{in},\texttt{undec}) + diff_{rh}(\texttt{out},\texttt{undec}) + diff_{rh}(\texttt{in},\texttt{undec}) = 1+1+1 = 3$. Thus we obtain $d_{rh}(\mathcal{L}_1,\mathcal{L}_2) > d_{rh}(\mathcal{L}_1,\mathcal{L}_3)$, i.e., $\mathcal{L}_3$ is strictly d_{rh}-closer to $\mathcal{L}_1$ than $\mathcal{L}_2$ is. It is quite easy to check that $diff_{rh}$ itself forms a metric distance over $\{\texttt{in},\texttt{out},\texttt{undec}\}$ which satisfies **(diff+)**, so by Props. 1 and 3:

Proposition 4. *d_{rh} satisfies* **(m1)**-**(m4)** *and* **(m5+)**.

Note that there is an incompatibility between d_H and d_{rh}, in the sense that there exist examples in which d_H and d_{rh} yield opposite conclusions regarding the relative proximity of two labellings $\mathcal{T}_1, \mathcal{T}_2$ to a given $\mathcal{S}$. For consider the three complete labellings of the AF containing three pairs of mutually attacking arguments in Fig. 2.
Here we have $d_H(\mathcal{S},\mathcal{T}_1) = 4 < 6 = d_H(\mathcal{S},\mathcal{T}_2)$ and $d_{rh}(\mathcal{S},\mathcal{T}_1) = 8 > 6 = d_{rh}(\mathcal{S},\mathcal{T}_2)$.

4.3 Third idea: Measuring incompatibility

The third idea for measuring the difference between labels is inspired by the work of [9]. A labelling $\mathcal{L}_1$ is *compatible* with labelling $\mathcal{L}_2$ (written as $\mathcal{L}_1 \approx \mathcal{L}_2$) iff there is no argument A such that either $[\mathcal{L}_1(A) = \texttt{in}$ and $\mathcal{L}_2(A) = \texttt{out}]$ or $[\mathcal{L}_1(A) = \texttt{out}$ and $\mathcal{L}_2(A) = \texttt{in}]$. The idea behind compatibility is to give a rough impression of how difficult it is to publicly defend a position (labelling) that is not one's own. Although it

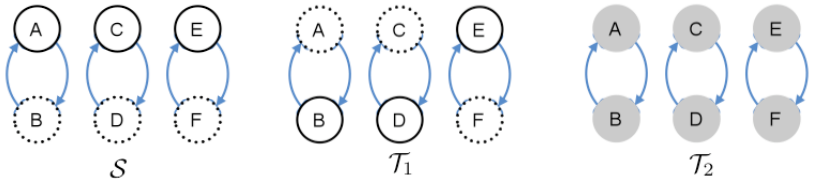


Fig. 2. Example showing opposite results of d_H and d_{rh}.

might be possible to publicly accept or reject an argument which one privately has no opinion about (undec), or to remain silent about an argument that one privately accepts or rejects, it is significantly more difficult to publicly accept an argument which one privately rejects (and vice versa). The third measure of distance makes the distance zero if the two labellings are compatible, and measures the "degree of incompatibility" if they are not.

$$\mathit{diff}_{\approx}(\mathtt{in},\mathtt{out}) = \mathit{diff}_{\approx}(\mathtt{out},\mathtt{in}) = 1$$
$$\mathit{diff}_{\approx}(\mathtt{x},\mathtt{y}) = 0 \text{ for all other pairs of labels}$$

So using this *diff* measure only in/out and out/in have a non-zero difference. This leads to a function $d_{\approx}$ (defined using $\mathit{diff}_{\approx}$ via (1)) which is more like a "measure of conflict" between $\mathcal{S}$ and $\mathcal{T}$. This distance is quite a departure from the previous two, since $\mathit{diff}_{\approx}$ does not form a metric distance over the labels. Although it clearly satisfies **(m1)**, **(m3)** and **(diff+)**, it fails to satisfy **(m2)** and **(m4)**. As a result, the resulting $d_{\approx}$ also fails to satisfy **(m2)** (and hence **(m5)** and **(m5+)**), as can be seen in Fig. 1 where $d_{\approx}(\mathcal{L}_1,\mathcal{L}_3) = 0$. It also fails **(m4)** (since, e.g., $d_{\approx}(\mathcal{L}_1,\mathcal{L}_2) = 3 > 0 = d_{\approx}(\mathcal{L}_1,\mathcal{L}_3) + d_{\approx}(\mathcal{L}_3,\mathcal{L}_2)$). The only properties satisfied from Section 3 are **(m1)** and **(m3)** (by Prop. 1, since clearly $\mathit{diff}_{\approx}$ satisfies the corresponding two properties).

5 Approaches based on critical sets

In the rest of the paper we will assume we work with some difference measure *diff* between labels which satisfies **(m1)** and **(m3)**, as well as the following two properties:

(diff A) $\mathit{diff}(\mathtt{in},\mathtt{undec}) = \mathit{diff}(\mathtt{out},\mathtt{undec})$
(diff B) $\mathit{diff}(\mathtt{in},\mathtt{out}) > 0$

(Note that all three *diff* measures from the previous section satisfy these properties).

Suppose we have the complete labelling $\mathcal{S}$ shown on the left of Fig. 3 over an AF containing eight arguments $\{A,B,C,D,E,F,G,H\}$. As usual a node with a solid line denotes the argument is in, while a dotted line denotes out. Now consider the two target labellings $\mathcal{T}_1$ and $\mathcal{T}_2$ shown to the right of it. $\mathcal{T}_1$ is obtained from $\mathcal{S}$ by leaving the labels of A,B,C,D as they are and inverting the labels of the four arguments E,F,G,H. For $\mathcal{T}_2$ we leave E,F,G,H untouched and invert the labels of the four arguments A,B,C,D. The question is: which of $\mathcal{T}_1,\mathcal{T}_2$ is closer to $\mathcal{S}$? Or are they both equally close?

Let's consider what a diff-based distance function d has to say about this. Assuming *diff* satisfies **(m1)** and **(m3)**, one can see that we will get $d(\mathcal{S},\mathcal{T}_1) = 4 \times \mathit{diff}(\mathtt{in},\mathtt{out}) =$

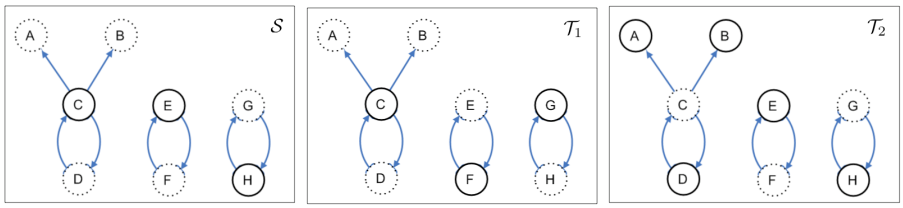


Fig. 3. Source labelling $\mathcal{S}$ and 2 target labellings $\mathcal{T}_1, \mathcal{T}_2$

$d(\mathcal{S}, \mathcal{T}_2)$. Thus any diff-based distance generated from a *diff* satisfying **(m1)** and **(m3)** will judge $\mathcal{T}_1$ and $\mathcal{T}_2$ as equidistant from $\mathcal{S}$.

However, on reflection it seems we can be more discriminating and say that $\mathcal{T}_2$ is closer to $\mathcal{S}$. Intuitively the reason is based on the observation that disagreement between $\mathcal{S}$ and $\mathcal{T}_2$ involves a higher degree of "contagion". If two agents only differ in their opinions on argument C (or only on D), this would suffice to determine their disagreement over all other arguments in that connected component (namely A, B, C, and D). On the other hand, when comparing $\mathcal{S}$ with $\mathcal{T}_2$, two agents would have to at least disagree (fundamentally let's say) on two arguments in order for this emerge.[4]

How can we make this intuition precise? We now investigate two ways in which the diff-based approach can be refined in order to take this into account. We will see that the first one, although intuitive, is flawed.

5.1 Critical subsets approach

The first idea comes from a concept introduced by Gabbay [16]. Instead of looking at all arguments, one specifically focuses on the *critical subsets*.

Definition 6 ([16]). *Given an AF $\mathcal{A} = (Ar, \rightharpoonup)$, a subset $X \subseteq Ar$ is* critical *iff for any $\mathcal{L}_1, \mathcal{L}_2 \in Comp_{\mathcal{A}}$ we get $\mathcal{L}_1 = \mathcal{L}_2$ whenever $\mathcal{L}_1$ and $\mathcal{L}_2$ agree on the arguments in X. We denote the set of critical subsets for $\mathcal{A}$ by critical$(\mathcal{A})$.*

In other words a critical subset for $\mathcal{A}$ is a set of arguments whose status is enough to determine the status of *all* the arguments in Ar. Clearly at least one critical subset will always exist, for Ar is obviously critical. We are interested in the *minimal* critical subsets.[5]

[4] A similar intuition to this can be found in [2] in the context of reasoning about action and belief update. The idea there is that there might exist some *causal* links between the value of one literal and that of another, which should be taken into account when calculating how much one possible world, i.e., conjunction of literals, differs from another. If the change in value of one literal is caused by another, then this change should not count towards calculating the difference.

[5] This is slightly reminiscent of the notion of *kernel* from belief revision [18], although the setting there is quite different. A kernel of a given set of formulas B with respect to a given formula α is a minimal subset of B which logically implies α.

Definition 7. *We denote the collection of set-theoretically minimal subsets of critical*$(\mathcal{A})$ *by mincritical*$(\mathcal{A})$*, i.e., mincritical*$(\mathcal{A}) \stackrel{\text{def}}{=} \{X \in \mathit{critical}(\mathcal{A}) \mid \nexists Y(Y \in \mathit{critical}(\mathcal{A}) \wedge Y \subset X)\}$.

If we look at the AF of Fig. 3 one can check that one critical subset is $X_1 = \{C, E, G\}$, since, the label of E (respectively G) determines the label of F (respectively H), while the label of C determines the labels of A, B and D. Indeed if C is `in` then A, B and D must all be `out`, if C is `out` then A, B, D must all be `in`, while if C is `undec` then A, B, D must all be `undec` too.

So, the first idea would be, given *diff*, to pick some minimal critical subset X and then just define, for all $\mathcal{S}, \mathcal{T} \in \mathit{Comp}_{\mathcal{A}}$, $d'(\mathcal{S}, \mathcal{T}) = d_X(\mathcal{S}, \mathcal{T})$, where

$$d_X(\mathcal{S}, \mathcal{T}) \stackrel{\text{def}}{=} \sum_{A \in X} \mathit{diff}(\mathcal{S}(A), \mathcal{T}(A)). \tag{2}$$

Formally, the *critical sets distance method cd* is defined via a function $\mathbb{C}$ which selects for each $\mathcal{A}$ an element of *mincritical*$(\mathcal{A})$ and then sets $cd(\mathcal{S}, \mathcal{T}) = d_{\mathbb{C}(\mathcal{A})}(\mathcal{S}, \mathcal{T})$ for any $\mathcal{S}, \mathcal{T} \in \mathit{Comp}_{\mathcal{A}}$.

Example 2. Taking the complete labellings $\mathcal{S}$ and $\mathcal{T}_1, \mathcal{T}_2$ in Fig. 3, and taking $\mathbb{C}(\mathcal{A}) = \{C, E, G\}$ we get $cd(\mathcal{S}, \mathcal{T}_1) = 2 \times \mathit{diff}(\mathtt{in}, \mathtt{out})$ and $cd(\mathcal{S}, \mathcal{T}_2) = \mathit{diff}(\mathtt{in}, \mathtt{out})$. Thus $\mathcal{T}_2$ is deemed closer to $\mathcal{S}$ than $\mathcal{T}_1$ is.

The distance function d_X in (2) actually fares rather well when measured against the properties for distance functions from earlier, provided *diff* is sufficiently well-behaved:

Proposition 5. *Let $X \in \mathit{critical}(\mathcal{A})$ and let d_X be defined from diff as in (2). For each $i = 1, 2, 3, 4$, if diff satisfies* **(mi)** *then d_X satisfies* **(mi)** *too. Furthermore if diff satisfies* **(m1)** *and* **(m2)** *then d_X satisfies* **(m5)***, while if diff satisfies* **(m1)***,* **(m2)** *and* **(diff+)** *then d_X satisfies* **(m5+)**.

Note the above result holds taking X to be *any* critical subset, not only the minimal ones. By taking $X = Ar$ we thus obtain Props. 1 and 3 from Section 4 as corollaries.

One problem is that more than one minimal critical subset may exist. For example in the above example one can check that another minimal critical subset can be obtained by exchanging A for D to obtain $X_2 = \{A, E, G\}$. Indeed one can exchange any argument in the leftmost component. One could also replace E by F or G by H. There are $4 \times 2 \times 2 = 16$ possible minimal critical subsets in this example. We would like the distance (or at least the similarity ordering induced by it) to be independent of the particular minimal critical subset we use. Is it possible that we might get $d_{X_1}(\mathcal{S}, \mathcal{T}_1) \neq d_{X_2}(\mathcal{S}, \mathcal{T}_2)$ for different minimal critical subsets X_1, X_2? In the above example the answer is no, but unfortunately this does not always hold in general, as the next example shows.

Example 3. Assume we have an AF $\mathcal{A}_1$ consisting of three arguments $\{A, B, C\}$ all mutually attacking each other, as depicted in Fig. 4. There are four possible complete labellings for $\mathcal{A}_1$, which are also depicted. It is not the case that by knowing the label of one argument we know the full complete labelling, however, one can check that if we know the label of any *pair* of arguments, we automatically know the label of the third. Thus we have $\mathit{mincritical}(\mathcal{A}_1) = \big\{\{A, B\}, \{A, C\}, \{B, C\}\big\}$. We have

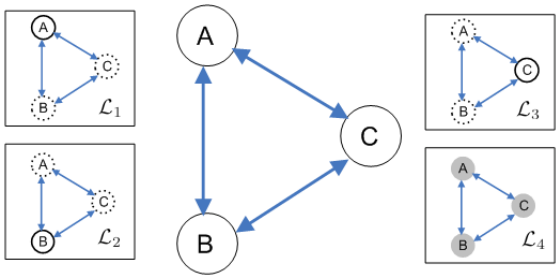


Fig. 4. The AF $\mathcal{A}_1$ and its four possible complete labellings $\mathcal{L}_1$-$\mathcal{L}_4$

$d_{\{A,B\}}(\mathcal{L}_1, \mathcal{L}_2) = 2 \times diff(\texttt{in}, \texttt{out})$ and $d_{\{A,B\}}(\mathcal{L}_1, \mathcal{L}_3) = diff(\texttt{in}, \texttt{out})$. Thus if we focus on the critical subset $\{A, B\}$ we obtain that $\mathcal{L}_3$ is closer to $\mathcal{L}_1$ than $\mathcal{L}_2$ is. But if instead we focus on critical subset $\{A, C\}$ we obtain the opposite conclusion, for $d_{\{A,C\}}(\mathcal{L}_1, \mathcal{L}_2) = diff(\texttt{in}, \texttt{out})$ and $d_{\{A,C\}}(\mathcal{L}_1, \mathcal{L}_3) = 2 \times diff(\texttt{in}, \texttt{out})$.

This sensitivity to the choice of critical subset is somewhat undesirable. Our next distance function aims to avoid this.

5.2 Distance via issue-wise label difference

We want to capture the idea that the labels of two arguments are "tied together". For example in a simple 2-argument AF consisting of two arguments A and B mutually attacking each other, there may be two arguments but to all intents and purposes there is really only one "issue" at stake, and that is whether A or B should be accepted. We want to isolate these different issues which are being argued over. Given an AF $\mathcal{A} = (Ar, \rightharpoonup)$, let us define the following two binary relations over Ar. For any $A, B \in Ar$:

- $A \equiv_1 B$ iff $\forall \mathcal{L} \in Comp_{\mathcal{A}} : \mathcal{L}(A) = \mathcal{L}(B)$
- $A \equiv_2 B$ iff $\forall \mathcal{L} \in Comp_{\mathcal{A}} : (\mathcal{L}(A) = \texttt{in} \Leftrightarrow \mathcal{L}(B) = \texttt{out}) \wedge (\mathcal{L}(A) = \texttt{out} \Leftrightarrow \mathcal{L}(B) = \texttt{in})$.

$A \equiv_1 B$ means that the labels assigned to A and B are exactly the same in all complete labellings, i.e., that A and B are in a sense logically equivalent, while $A \equiv_2 B$ means that A and B always receive "opposite" labels: whenever A is labelled in then B is labelled out, and vice versa. It is easy to see that if $A \equiv_2 B$ then we also have $\mathcal{L}(A) = \texttt{undec}$ iff $\mathcal{L}(B) = \texttt{undec}$. From these two relations we define

$$A \equiv B \text{ iff } (A \equiv_1 B \vee A \equiv_2 B).$$

Thus if $A \equiv B$ then intuitively the labels of A and B are "in sync", in that the label of one cannot be changed without causing a change of equal magnitude to the label of the other.

Proposition 6. *$\equiv$ is an equivalence relation over Ar.*

Within each $\equiv$-equivalence class, there are at most 3 possible labellings which can occur: either *(i)* all its elements are labelled `undec`, or *(ii)* all its elements are set to `in` or `out`, or *(iii)* the "inverse" labelling to *(ii)* occurs, in which those arguments labelled `in` become `out` and those labelled `out` are now `in`. Essentially each equivalence class acts as a single 3-valued argument. We call each such class an *issue* of the given AF.

Definition 8. *Given an AF $\mathcal{A} = (Ar, \rightharpoonup)$, the set $\mathbb{I}(\mathcal{A})$ of* issues *of $\mathcal{A}$ is defined as $\mathbb{I}(\mathcal{A}) = Ar/\equiv$. For $A \in Ar$ we will denote the $\equiv$-equivalence class of A by $[A]$.*

For example, it can be checked that the issues for the AF in Fig. 3 are $\{A, B, C, D\}$, $\{E, F\}$ and $\{G, H\}$. In the AF of Fig. 4 there are 3 issues $\{A\}$, $\{B\}$ and $\{C\}$.

Now, rather than calculate distance via argument-wise label difference as we did in Section 4 , we can instead do it via *issue-wise* label difference. For this we need to define the measure of disagreement $DIFF(\mathcal{S}, \mathcal{T}, [A])$ between two labellings $\mathcal{S}$ and $\mathcal{T}$ on a single issue $[A]$. We do this using our distance-between-labels measure *diff*:

$$DIFF(\mathcal{S}, \mathcal{T}, [A]) \stackrel{\text{def}}{=} diff(\mathcal{S}(A), \mathcal{T}(A)).$$

Proposition 7. *DIFF is well-defined, i.e., if $[A] = [B]$ then $diff(\mathcal{S}(A), \mathcal{T}(A)) = diff(\mathcal{S}(B), \mathcal{T}(B))$.*

Note this result depends on the assumptions that *diff* satisfies **(m1)**, **(m3)** and **(diff A)**.

Finally the issue-based distance measure *id* is defined by setting, for any $\mathcal{S}, \mathcal{T} \in Comp_{\mathcal{A}}$,

$$id(\mathcal{S}, \mathcal{T}) = \sum_{[A] \in \mathbb{I}(\mathcal{A})} DIFF(\mathcal{S}, \mathcal{T}, [A])$$

In the example in Fig. 3 we have $id(\mathcal{S}, \mathcal{T}_1) = 2 \times diff(\texttt{in}, \texttt{out})$ and $id(\mathcal{S}, \mathcal{T}_2) = diff(\texttt{in}, \texttt{out})$, as with the critical subsets approach of Section 5.1. For the example in Fig. 4 we get $id(\mathcal{L}_1, \mathcal{L}_2) = 2 \times diff(\texttt{in}, \texttt{out}) = id(\mathcal{L}_1, \mathcal{L}_3)$. Thus according to the issue-based distance $\mathcal{L}_2$ and $\mathcal{L}_3$ are equidistant from $\mathcal{L}_1$.

The issue-wise distance measure can be related to the preceding critical subsets approach. Clearly we have $id(\mathcal{S}, \mathcal{T}) = d_X(\mathcal{S}, \mathcal{T})$ (see equation (2) in Section 5.1), where X is any set formed by taking a representative of each $\equiv$-equivalence class. It turns out that we have the following:

Proposition 8. *Let X be any set obtained by taking one element of each issue in $\mathbb{I}(\mathcal{A})$. Then $X \in critical(\mathcal{A})$.*

Thus *id* can be thought of as a critical-set based distance which chooses from among a particular class of critical sets, viz. those which contain one argument from each issue. Furthermore, unlike the critical-set based distance the precise choice of these elements is irrelevant. However the critical set chosen need not be a minimal one, i.e., an element of $mincritical(\mathcal{A})$, as can be seen already in the AF of Fig. 4. We may deduce from this and Prop. 5 the following (recall that we are already assuming *diff* to satisfy **(m1)**, **(m2)** and **(diff A)**) :

Proposition 9. *id satisfies* **(m1)** *and* **(m3)**. *If diff satisfies* **(m4)** *then so does id. If diff satisfies* **(m2)** *then id satisfies* **(m5)** *(and hence also* **(m2)***). If diff satisfies* **(diff+)** *as well as* **(m2)** *then id satisfies* **(m5+)**.

5.3 More properties of *id*

Prop. 9 says that *id* satisfies the same postulates which were shown to be satisfied by d_{rh}. Can we formalise any properties which differentiate *id* from d_{rh}? We end this section with 2 such properties. The first is formulated with the help of the notion of issue described above. Consider the following definitions:

- $HCIssues(\mathcal{S},\mathcal{T}) \stackrel{\text{def}}{=} \{[A] \in \mathbb{I}(\mathcal{A}) \mid \mathcal{S}(A) \perp \mathcal{T}(A)\}$
- $SCIssues(\mathcal{S},\mathcal{T}) \stackrel{\text{def}}{=} \{[A] \in \mathbb{I}(\mathcal{A}) \mid \mathcal{S}(A) \amalg \mathcal{T}(A)\}$

Where $\mathcal{S}(A) \perp \mathcal{T}(A)$ means that either $\mathcal{S}(A) = \texttt{in}$ and $\mathcal{T}(A) = \texttt{out}$ or $\mathcal{S}(A) = \texttt{out}$ and $\mathcal{T}(A) = \texttt{in}$ (*hard conflict*), and $\mathcal{S}(A) \amalg \mathcal{T}(A)$ means $\mathcal{S}(A) \neq \mathcal{T}(A)$ and at least one of the two labels is `undec` (*soft conflict*). Note that if $A \equiv A'$ then $\mathcal{S}(A) \perp \mathcal{T}(A)$ iff $\mathcal{S}(A') \perp \mathcal{T}(A')$ for all $\mathcal{S},\mathcal{T} \in Comp_{\mathcal{A}}$ and similarly for $\mathcal{S}(\cdot) \amalg \mathcal{T}(\cdot)$.

(m6) If $|HCIssues(\mathcal{S},\mathcal{T}_1)| < |HCIssues(\mathcal{S},\mathcal{T}_1)|$ and $|SCIssues(\mathcal{S},\mathcal{T}_1)| \leqslant |SCIssues(\mathcal{S},\mathcal{T}_1)|$, then $d(\mathcal{S},\mathcal{T}_1) < d(\mathcal{S},\mathcal{T}_2)$ (Issue disagreement cardinality)

So **(m6)** says if moving from $\mathcal{T}_1$ to $\mathcal{T}_2$ causes both the number of hard conflicting issues and the number of soft conflicting issues with $\mathcal{S}$ to increase, with a *strict* increase in the former, then $\mathcal{T}_2$ should be considered further away from $\mathcal{S}$.

Our second property differs from all our previous ones in that whereas they dealt with a *fixed* AF $\mathcal{A}$ as given, this rule relates distance between labellings over *different*, but related argumentation frameworks. Technically speaking, while **(m1)**-**(m6)** are properties purely of the *labelling distance* $d_{\mathcal{A}}$ for fixed $\mathcal{A}$, the next is a property of the *distance method*, i.e., the mapping $\mathcal{A} \mapsto d_{\mathcal{A}}$ (Defn. 4). Let $\mathcal{A} = (Ar, \rightharpoonup)$ be an arbitrary AF and let $\mathcal{A}^+$ be any framework obtained from $\mathcal{A}$ by adding a single new argument $B \notin Ar$ along with a **single** attack $A \rightharpoonup B$ from some $A \in Ar$. Then for any two complete labellings $\mathcal{S},\mathcal{T}$ over the expanded AF $\mathcal{A}^+$ the following should hold for any distance method:

(m7) $d_{\mathcal{A}^+}(\mathcal{S},\mathcal{T}) = d_{\mathcal{A}}(\mathcal{S}\downarrow_{Ar}, \mathcal{T}\downarrow_{Ar})$. (Indifference to peripheral issues)

Here, in the right-hand side, $\mathcal{S}\downarrow_{Ar}$ denotes $\mathcal{S}$ restricted to the arguments in Ar, i.e., ignoring B (and similarly for $\mathcal{T}\downarrow_{Ar}$). It is easy to check that $\mathcal{S}\downarrow_{Ar}, \mathcal{T}\downarrow_{Ar} \in Comp_{\mathcal{A}}$. The property essentially says that adding $A \rightharpoonup B$ to $\mathcal{A}$ should not make any difference to the distance, intuitively because the label of B is in any case determined by that of A, and so the introduction of B does not introduce any new *issues*, in our sense of the word. The next result confirms the issue-based distance satisfies both of these last 2 postulates:

Proposition 10. *For any AF $\mathcal{A}$, $id_{\mathcal{A}}$ satisfies* **(m6)** *(if diff satisfies* **(m2)***), and the distance method $\mathcal{A} \mapsto id_{\mathcal{A}}$ satisfies* **(m7)**.

d_{rh} fails to satisfy both **(m6)** and **(m7)** in general, as can be seen in Fig. 3. For **(m6)** observe that $\mathcal{T}_1$ has strong conflict with $\mathcal{S}$ on strictly more issues than $\mathcal{T}_2$ does, while neither has any soft conflicts, and yet they are judged equidistant from $\mathcal{S}$. For **(m7)** note that $d_{rh}(\mathcal{S},\mathcal{T}_2) = 8$, but if we take the same labellings restricted to the AF obtained by deleting B then we get $d_{rh}(\mathcal{S},\mathcal{T}_2) = 6$. It is also easy to see that d_H fails these 2 rules as well, while *cd* can be shown to falsify **(m6)**, but validates **(m7)** under the assumption that $\mathbb{C}$ selects the same minimal critical set in both $\mathcal{A}$ and $\mathcal{A}^+$.

6 Related Work and Conclusion

We have initiated the investigation of the notion of distance between two reasonable evaluations of an argument graph. While this issue has been investigated in non-argument based accounts of both belief revision [13, 20, 22], in judgement aggregation [21, 23], and in abstract preferences [1], to our knowledge we are the first to study it in the context of formal argumentation theory.

We presented several different distance functions, all defined on top of a difference function on the space of possible labels $\{\texttt{in}, \texttt{out}, \texttt{undec}\}$. These functions fall into two groups: those which sum the difference between the labels of *all* arguments Ar in the framework, and those which single out various subsets of Ar as being in some sense the *critical* ones. We gave some postulates for such distance functions and, even though we saw that most of our suggested distances suffer from some problem or another, developed some intuitions via several examples about what a distance function between complete labellings should be like.

Property	d_H	d_{rh}	$d_{\approx}$	cd	id
(m1) Distance to itself is zero	✓	✓	✓	✓	✓
(m2) Non-zero distance to others	✓	✓	×	✓	✓
(m3) Symmetry	✓	✓	✓	✓	✓
(m4) Triangle inequality	✓	✓	×	✓	✓
(m5) Disagreement monotony	✓	✓	×	✓	✓
(m5+) Betweenness monotony	×	✓	×	✓	✓
(m6) Issue disagreement cardinality	×	×	×	×	✓
(m7) Indifference to peripheral issues	×	×	×	✓	✓

Table 1. Summary of Properties of Distance Measures. Results for *cd*, *id* assume *diff* satisfying **(m1)**, **(m3)**, **(diff A)** and **(diff B)**. The satisfaction of **(m5+)** for *cd* and *id* assumes *diff* satisfies **(diff +)**.

Coste-Marquis et al presented an approach for merging multiple Dung-style argumentation graphs presented by multiple agents [11]. The authors use a combination of graph expansion, distance calculation and voting in order to arrive at a single argumentation framework. This work addresses a fundamentally different problem, since agents may differ over which arguments exist, or which arguments attack which other arguments. In contrast, in our work, we assume that all arguments are available to all agents (e.g. as in a jury hearing), and that the attack relation is not a subjective matter (i.e. it is objectively determined by the underlying logical system, as is for instance the case in [17, 24]). In other words, the distance measures introduced by Coste-Marquis et al are between different graphs, and thus address a fundamentally different problem (e.g. the edit distance, which measures the number of insertions/deletions of attacks needed to turn one entire argument graph into another). Our notion of distance, on the other hand, is aimed at quantifying disagreement over the evaluation of the *given* graph; i.e. it's distance between different evaluations of the given evidence, not between different perspectives on what the evidence is.

For future work we would like to apply these new distances to the problems of revision and judgement aggregation in argumentation. In revision we want to choose the closest labelling to the current one which extends an input new partial labelling. In questions of judgement aggregation we want to choose the labellings which are closest to the group as a whole. Similar considerations have been applied in propositional contexts (e.g. [12, 19]), while a first exploration of the use of Hamming-like distances (see Sect. 4) in labelling-aggregation has been carried out by Caminada et al. in [10], where it is used to check the manipulability and Pareto optimality of certain aggregation operators. They assume that each member of a group of agents provides a complete labelling, and then that each agent's preference relation over the set of all complete labellings is given by Hamming set or Hamming distance from its given labelling.

It would also be interesting to examine distances from a computational viewpoint via the construction of algorithms for computing the various distance measures we proposed. Finally, here we focused on complete labellings. This is reasonable since they correspond to rational, coherent standpoints. But the definitions will also work for other families of labellings too, like preferred and stable [15], and semi-stable [4].

Acknowledgements

Thanks are due to the reviewers for their encouraging remarks. Thanks also to Ringo Baumann, Gerhard Brewka and the Individual and Collective Reasoning group at the University of Luxembourg for some useful comments.

References

1. N Baigent. Preference proximity and anonymous social choice. *The Quarterly Journal of Economics*, 102(1):161–169, 1987.
2. G Brewka and J Hertzberg. How to do things with worlds: On formalizing actions and plans. *Journal of Logic and Computation*, 3(5):517, 1993.
3. M Caminada. On the issue of reinstatement in argumentation. *Lecture Notes in Computer Science*, 4160:111, 2006.
4. M Caminada. Semi-stable semantics. In *COMMA 2006*, pages 121–130, 2006.
5. M Caminada. An algorithm for computing semi-stable semantics. *Lecture Notes in Computer Science*, 4724:222, 2007.
6. M Caminada. A labelling approach for ideal and stage semantics. *Argument & Computation*, 2:1–21, 2011.
7. M Caminada and L Amgoud. On the evaluation of argumentation formalisms. *Artificial Intelligence*, 171(5-6):286–310, 2007.
8. M Caminada and D Gabbay. A logical account of formal argumentation. *Stud Logica*, 93(2):109–145, 2009.
9. M Caminada and G Pigozzi. On judgment aggregation in abstract argumentation. *Autonomous Agents and Multi-Agent Systems*, pages 1–39, 2009.
10. M Caminada, G Pigozzi, and M Podlaszewski. Manipulation in group argument evaluation. In *IJCAI-11*, 2011.
11. S Coste-Marquis, C Devred, S Konieczny, MC Lagasquie-Schiex, and P Marquis. On the merging of dung's argumentation systems. *Artificial Intelligence*, 171(10-15):730–753, 2007.

12. M Dalal. Investigations into a theory of knowledge base revision: Preliminary report. In *AAAI-88*, volume 2, pages 475–479, 1988.
13. JP Delgrande. Preliminary considerations on the modelling of belief change operators by metric spaces. In *Proc. NMR*, pages 118–125, 2004.
14. P. M. Dung, P. Mancarella, and F. Toni. Computing ideal sceptical argumentation. *Artificial Intelligence*, 171(10-15):642–674, 2007.
15. PM Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games* 1. *Artificial Intelligence*, 77(2):321–357, 1995.
16. D Gabbay. Fibring argumentation frames. *Stud Logica*, 93(2):231–295, 2009.
17. N Gorogiannis and A Hunter. Instantiating abstract argumentation with classical logic arguments: Postulates and properties. *Artificial Intelligence*, pages 1–30, 2010.
18. S.O. Hansson. Kernel contraction. *Journal of Symbolic Logic*, 59(3):845–859, 1994.
19. S Konieczny, J Lang, and P Marquis. DA^2 merging operators. *Artificial Intelligence*, 157(1-2):49–79, 2004.
20. D Lehmann, M Magidor, and K Schlechta. Distance semantics for belief revision. *Journal of Symbolic Logic*, pages 295–317, 2001.
21. M Miller and D Osherson. Methods for distance-based judgment aggregation. *Social Choice and Welfare*, 32(4):575–601, 2009.
22. P Peppas, S Chopra, and N Foo. Distance semantics for relevance-sensitive belief revision. *Proc. KR*, 4:319–328, 2004.
23. G Pigozzi. Belief merging and the discursive dilemma: an argument-based account to paradoxes of judgment aggregation. *Synthese*, 152(2):285–298, 2006.
24. H. Prakken. An abstract framework for argumentation with structured arguments. *Argument and Computation*, 1(2):93–124, 2010.
25. I Rahwan and F Tohmé. Collective argument evaluation as judgement aggregation. In *Proc. AAMAS*, pages 417–424, 2010.
26. R Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

On the Complexity of Computing the Justification Status of an Argument*

Wolfgang Dvořák

Technische Universität Wien, Institute for Information Systems 184/2
Favoritenstrasse 9-11, 1040 Vienna, Austria
dvorak@dbai.tuwien.ac.at

Abstract. We address the problem of determining the acceptance status of an argument w.r.t. labeling-based semantics. Wu and Caminada recently proposed a labeling-based justification status of arguments to distinguish different levels of acceptability for arguments. We generalize their approach, which was originally restricted to complete semantics, to arbitrary argumentation semantics and provide a comprehensive study of the computational properties.

1 Introduction

We study the problem of computing the acceptance status of an argument in abstract argumentation frameworks (AFs) [9], following the approach of labeling-based justification status by Wu and Caminada [18].

Dung [9] introduced abstract argumentation frameworks together with semantics which specify subsets of arguments, so called extensions, distinguishing the arguments which are accepted from the arguments which are not. Towards a more fine-grainted distinction of arguments several kind of argumentation labelings have been proposed either for algorithmic or logical purposes (see, e.g. [6, 16, 17]). In this work we use three-valued labelings as proposed by Caminada and Gabbay [6]. Roughly speaking such labelings partition the arguments of an framework into three sets. Similar as in the concept of extensions, there are the acceptable arguments (which are labeled by in) and labelings distinguish two kinds of arguments which are not accepted: those which are attacked by an accepted argument (and labeled out) and those which are neither accepted nor attacked (and labeled $undec$).

Traditional extension-based approaches for deciding the acceptance status of arguments distinguish between skeptically accepted arguments, i.e. arguments contained in each extension, credulously accepted arguments, i.e. arguments contained in at least one extension and arguments which are in no extension at all. However such a characterisation completely ignores the additional information provided by labelings. To take this information into account, Wu and Caminada [18] proposed their labeling-based justification status of an argument, which allows to distinguish different levels of acceptance (and rejection) for arguments based on the labelings of an argumentation framework.

* This work was supported by the Vienna Science and Technology Fund (WWTF) under grant ICT08-028.

The main idea of this justification status for an argument a is to consider all labels l such that at least one complete labeling of the AF assigns l to a.

As mentioned, Wu and Caminada only consider the justification status concerning complete semantics. In this paper we first generalize the concept of justification status to arbitrary argumentation semantics and then consider instantiations for several important semantics, namely the semantics defined in [9], semi-stable [3] and stage [16] semantics. We provide a detailed complexity analysis for the concept of justification status w.r.t the afore mentioned semantics (which has not been done in [18]). Further we show general properties of these justification statuses as well as relations between justification statuses for different semantics.

The structure of the remainder of the paper is as follows: In Section 2 we introduce abstract argumentation frameworks, the semantics we consider in the paper and the concept of labelings. We also highlight known complexity results for these semantics. Section 3 gives the definition of the justification status of an argument, as well as basic results about the properties of such justification statuses for different semantics. In Section 4 we provide a comprehensive complexity analysis for the decision problems associated to the justification status of an argument. Finally, in Section 5 we conclude the paper with a summary of the presented results.

2 Preliminaries

In this section we introduce abstract argumentation frameworks [9] and the concept of labelings [6]. Further we recall some of the most important semantics for abstract argumentation (see [2]). Finally, we highlight complexity results for typical decision problems associated to such frameworks.

Definition 1. *An* argumentation framework (AF) *is a pair $F = (A, R)$ where A is a finite set of arguments and $R \subseteq A \times A$ is the attack relation. For a given AF $F = (A, R)$ we use A_F to denote the set A of its arguments and R_F to denote its attack relation R. We sometimes use $a \rightarrowtail^R b$ instead of $(a, b) \in R$. For $S \subseteq A$ and $a \in A$, we also write $S \rightarrowtail^R a$ (resp. $a \rightarrowtail^R S$) in case there exists $b \in S$, such that $b \rightarrowtail^R a$ (resp. $a \rightarrowtail^R b$). In case no ambiguity arises, we use $\rightarrowtail$ instead of $\rightarrowtail^R$.*

Using the extension-based approach, one assigns a set $\sigma(F) \subseteq 2^A$ of extensions to each AF $F = (A, R)$. For σ consider the functions stb, adm, prf, com, grd, stg, and sem which stand for stable, admissible, preferred, complete, ground stage, and semi-stable semantics. Before actually defining the semantics, we have to introduce a few more formal concepts.

Definition 2. *Given an AF $F = (A, R)$, an argument $a \in A$ is* defended *(in F) $S \subseteq A$ if for each $b \in A$, such that $b \rightarrowtail a$, also $S \rightarrowtail b$ holds. The characteristic function $\mathcal{F}_F : 2^A \rightarrow 2^A$, is defined as $\mathcal{F}_F(S) = \{x \in A_F \mid x \text{ is defended by } S\}$. Moreover, for a set $S \subseteq A$, we denote by S_R^+ the set $S \cup \{b \mid \exists a \in S, \text{such that } (a, b) \in R\}$.*

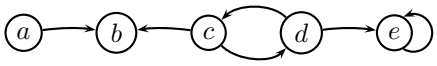
We are now ready to define the semantics.

Definition 3. *Let $F = (A, R)$ be an AF. A set $S \subseteq A$ is* conflict-free (in F) *(denoted as $S \in cf(F)$), iff there are no $a, b \in S$, such that $(a, b) \in R$. For $S \in cf(F)$, we define:*

- $S \in stb(F)$, *if* $S_R^+ = A$;
- $S \in adm(F)$, *if* $S \subseteq \mathcal{F}_F(S)$;
- $S \in com(F)$, *if* $S = \mathcal{F}_F(S)$;
- $S \in prf(F)$, *if* $S \in adm(F)$ *and there is no* $T \in adm(F)$ *with* $T \supset S$;
- $S \in grd(F)$, *if* $S \in com(F)$ *and there is no* $T \in com(F)$ *with* $T \subset S$;
- $S \in sem(F)$, *if* $S \in adm(F)$ *and there is no* $T \in adm(F)$ *with* $T_R^+ \supset S_R^+$.
- $S \in stg(F)$, *if there is no* $T \in cf(F)$ *in* F, *such that* $T_R^+ \supset S_R^+$;

We recall that for each AF F, $stb(F) \subseteq sem(F) \subseteq prf(F) \subseteq com(F) \subseteq adm(F)$, and that for each of the considered semantics σ (except stable) $\sigma(F) \neq \emptyset$ holds. Moreover we have that for each AF F there is an unique grounded extension, which is the least fixed-point of $\mathcal{F}_F$. Further in case that an AF has at least one stable extension then its stable, semi-stable, and stage extensions coincide.

Example 1. Consider the AF $F = (A, R)$, with $A = \{a, b, c, d, e\}$ and $R = \{(a, b), (c, b), (c, d), (d, c), (d, e), (e, e)\}$. The graph representation of F is given as follows.



We have $stb(F) = stg(F) = sem(F) = \{\{a, d\}\}$. Further we have the admissible sets $\{\}$,$\{a\}$,$\{c\}$,$\{d\}$,$\{a, c\}$,$\{a, d\}$,and thus $prf(F) = \{\{a, c\}$,$\{a, d\}\}$. Finally the complete extensions of F are $\{a\}$, $\{a, c\}$ and $\{a, d\}$, with $\{a\}$ being the grounded extension of F. $\Diamond$

An extension separates the accepted arguments from the non-accepted arguments, but there may be a substantial difference whether an argument is rejected because of a conflict with the arguments in the extension or because the extension does not defend it. The more fine-grained concept of labelings, as introduced by Caminada [4], generalizes extensions and captures the above observation.

Definition 4. *Let $F = (A, R)$ be an AF. A labeling for F is a function $\mathcal{L} : A \to \{in, out, undec\}$. We will denote labelings $\mathcal{L}$ also by triples $(\mathcal{L}_{in}, \mathcal{L}_{out}, \mathcal{L}_{undec})$, where $\mathcal{L}_l = \{a \in A \mid \mathcal{L}(a) = l\}$.*

The intuition behind these labels is the following. An argument is labeled with: in if it is accepted; out if there are strong reasons to reject it, i.e. its attacked by an accepted argument; and by $undec$ if the argument is undecided, i.e. neither accepted nor attacked by accepted arguments. In [5,6], the authors define labeling-based semantics for AFs independent of the extension-based semantics and then showed a strong correspondence to extension based semantics. For simplicity we use this correspondence to directly define labeling-based semantics via extensions. To this end, for each AF we define a function mapping sets of arguments to labelings.

Definition 5. *Let $F = (A, R)$ be an AF. We define the function $Ext2Lab_F : 2^A \to \{in, out, undec\}^A$ such that $Ext2Lab_F(E) = (E, E_R^+ \setminus E, A \setminus E_R^+)$ for $E \subseteq A$.*

Now we can interpret an extension-based semantics σ as labeling-based semantics, by mapping each σ-extension to a labeling using the function $Ext2Lab$. In particular $Ext2Lab$ is a one-to-one mapping (see [6]) and thus different extensions yield different labelings.

Definition 6. *Let $F = (A, R)$ be an AF and σ an extension-based semantics. The corresponding labeling-based semantics $\sigma_{\mathcal{L}}$ is defined as follows $\sigma_{\mathcal{L}}(F) = \{Ext2Lab(E) \mid E \in \sigma(F)\}$. If no ambiguity arises we will use $\sigma(F)$ instead of $\sigma_{\mathcal{L}}(F)$.*

We mention that our definition of $adm_{\mathcal{L}}$ doesn't match the definition of admissible labelings by Caminada and Gabbay [6], but what they call JV-labelings. [1]

We now turn to the complexity of reasoning in AFs. We assume the reader has knowledge about standard complexity classes, i.e. P, NP and LOGSPACE (L). Nevertheless we briefly recapitulate the concept of oracle machines and some related complexity classes. Let $\mathcal{C}$ notate some complexity class. By a $\mathcal{C}$-oracle machine we mean a (polynomial time) Turing machine which can access an oracle that decides a given (sub)-problem in $\mathcal{C}$ within one step. We denote the class of decision problems, that can be solved by such machines, as $\mathrm{P}^{\mathcal{C}}$ if the underlying Turing machine is deterministic and $\mathrm{NP}^{\mathcal{C}}$ if the underlying Turing machine is nondeterministic. Moreover we consider deterministic oracle machines where the number of allowed oracle calls is bounded by a constant k, and denote the corresponding complexity classes as $\mathrm{P}^{\mathcal{C}[k]}$. We now turn to concrete complexity classes. The class $\Sigma_2^P = \mathrm{NP}^{\mathrm{NP}}$, denotes the problems which can be decided by a nondeterministic polynomial time algorithm that has access to an NP-oracle. The class $\Pi_2^P = \mathrm{coNP}^{\mathrm{NP}}$ is defined as the complementary class of Σ_2^P, i.e. $\Pi_2^P = \mathrm{co}\Sigma_2^P$. Finally we define the classes D^{P} and $\mathrm{D}_2^{\mathrm{P}}$. A decision problem L is in the class D^{P} iff L can be characterised as $L_1 \cap L_2$ for decision problems $L_1 \in \mathrm{NP}$ and $L_2 \in \mathrm{coNP}$. Similarly $L \in \mathrm{D}_2^{\mathrm{P}}$ iff L can be characterised as $L_1 \cap L_2$ for $L_1 \in \Sigma_2^P$ and $L_2 \in \Pi_2^P$. The following gives an overview of the relations between the complexity classes used in this paper.

$$\mathrm{L} \subseteq \mathrm{P} \subseteq \begin{matrix}\mathrm{NP}\\ \mathrm{coNP}\end{matrix} \subseteq \mathrm{D}^{\mathrm{P}} \subseteq \begin{matrix}\Sigma_2^P\\ \Pi_2^P\end{matrix} \subseteq \mathrm{P}^{\Sigma_2^P[1]} \subseteq \mathrm{D}_2^{\mathrm{P}}$$

Next we mention the typical decision problems for argumentation frameworks. To this end let σ be one of the semantics introduced in Definition 3.

- Cred_σ: Given AF $F = (A, R)$ and $a \in A$. Is a contained in some $S \in \sigma(F)$?
- Skept_σ: Given AF $F = (A, R)$ and $a \in A$. Is a contained in each $S \in \sigma(F)$?
- Skept'_σ: Given AF $F = (A, R)$ and $a \in A$. Is a contained in each $S \in \sigma(F)$ and $\sigma(F) \neq \emptyset$?
- Ver_σ: Given AF $F = (A, R)$ and $S \subseteq A$. Is $S \in \sigma(F)$?
- Exists_σ: Given AF $F = (A, R)$. Is $\sigma(F) \neq \emptyset$?
- $\mathsf{Exists}_\sigma^{\neg\emptyset}$: Given AF $F = (A, R)$. Does there exist a set $S \neq \emptyset$ such that $S \in \sigma(F)$?

From the literature [7–9, 11–15], we obtain the complexity-landscape of abstract argumentation as given in Table 1. We mention that most of the semantics (except stable) always provide at least one extension. For such a semantics σ the problem Exists_σ can be trivially answered with yes and further the problems Skept'_σ and Skept_σ coincide.

[1] However for each AF both semantics propose the same justification statuses for the arguments.

σ	Cred_σ	Skept_σ	Skept'_σ	Ver_σ	Exists_σ	$\mathsf{Exists}_\sigma^{\neg\emptyset}$
grd	P-c	P-c	P-c	P-c	trivial	in L
stb	NP-c	coNP-c	D^{P}-c	in L	NP-c	NP-c
adm	NP-c	trivial	trivial	in L	trivial	NP-c
com	NP-c	P-c	P-c	in L	trivial	NP-c
prf	NP-c	Π_2^P-c	Π_2^P-c	coNP-c	trivial	NP-c
sem	Σ_2^P-c	Π_2^P-c	Π_2^P-c	coNP-c	trivial	NP-c
stg	Σ_2^P-c	Π_2^P-c	Π_2^P-c	coNP-c	trivial	in L

Table 1. Complexity of abstract argumentation ($\mathcal{C}$-c denotes completeness for class $\mathcal{C}$)

3 Justification Status of Arguments

Here we consider the task of reasoning in AFs with labeling-based semantics. When using extensions the usual reasoning modes are skeptical acceptance, accepting an argument if it is in all extensions, and credulous acceptance, i.e. accepting an argument if it is an least one extension. Applying these acceptance criterion to labelings would not take the labels $out, undec$ in to account, hence one seeks for a more sophisticated reasoning mode. To overcome this, Wu and Caminada [18] recently proposed a more elaborate concept, the so called *justification status* of an argument, to reason with complete labelings. The main idea is to define the acceptance status of an argument a, by the set of labels l such that there exists an complete labeling mapping a to l. In the following definition we generalize this concept to arbitrary semantics.

Definition 7. *Let $F = (A, R)$ be an AF and σ a semantic. The* justification status *of an argument $a \in A$ in F is defined as $\mathcal{JS}_\sigma(F, a) = \{\mathcal{L}(a) \mid \mathcal{L} \in \sigma(F)\}$.*

Definition 7 in general proposes eight different justification statuses for arguments. Following [18], we call $\{in\}$ strong accept; $\{in, undec\}$ weak accept; $\{out\}$ strong reject; $\{out, undec\}$ weak reject; and consider the remaining as borderline cases.

The justification status of an argument generalizes the idea of skeptical and credulous reasoning in the sense that one can encode skeptical and credulous acceptance as a property of the justification status of the argument. An argument a, is skeptically accepted iff $\mathcal{JS}_\sigma(F, a) = \{in\}$, whereas it is credulously accepted iff $in \in \mathcal{JS}_\sigma(F, a)$.

As mentioned above in general there are eight possible justification statuses for each argument, but none of the semantics in Definition 3 is able to generate all of them. In the following we give a compact analysis of the possible justification statuses for each semantics. We start with a general observation for unique status semantics:

Lemma 1. *Let σ be an unique status semantics, $F = (A, R)$ and $a \in A$. Then $\mathcal{JS}_\sigma(F, a) \in \{\{in\}, \{out\}, \{undec\}\}$.*

Proof. Immediate by the fact that there is exactly one labeling $\mathcal{L} \in \sigma(F)$. □

The following proposition fully characterises the possible justification statuses for the semantics from Definition 3.

Proposition 1. *Let $F = (A, R)$ be an AF and $a \in A$. Then we have that:*

1. $\mathcal{JS}_{grd}(F,a) \in \{\{in\},\{out\},\{undec\}\}$
2. $\mathcal{JS}_{adm}(F,a) \in \{\{undec\},\{in,undec\},\{out,undec\},\{in,out,undec\}\}$
3. $\mathcal{JS}_{com}(F,a) \in \{\{in\},\{out\},\{undec\},\{in,undec\},\{out,undec\},$
   $\{in,out,undec\}\}$
4. $\mathcal{JS}_{stb}(F,a) \in \{\{in\},\{out\},\{in,out\},\{\}\}$
5. $\mathcal{JS}_{prf}(F,a) \in \{\{in\},\{out\},\{undec\},\{in,undec\},\{out,undec\},\{in,out\},$
   $\{in,out,undec\}\}$
6. $\mathcal{JS}_{sem}(F,a) \in \{\{in\},\{out\},\{undec\},\{in,undec\},\{out,undec\},\{in,out\},$
   $\{in,out,undec\}\}$
7. $\mathcal{JS}_{stg}(F,a) \in \{\{in\},\{out\},\{undec\},\{in,undec\},\{out,undec\},\{in,out\},$
   $\{in,out,undec\}\}$

Proof. We have that (1) holds because grd is a unique status semantics. For (2) we use that $(\emptyset,\emptyset,A)$ is always an admissible labeling and thus $undec \in \mathcal{JS}_{adm}(F,a)$. The case of complete semantics has already been studied by Wu and Caminada [18]. The restricted domain steams from the fact that the grounded labelling $\mathcal{L}_{grd}$ is the unique $\subseteq$-minimal w.r.t. both $\mathcal{L}_{in}$ and $\mathcal{L}_{out}$ complete labeling. So if $\mathcal{L}_{grd}(a) = in$ or $\mathcal{L}_{grd}(a) = out$ we have that also $\mathcal{JS}_{com}(F,a) = \{in\}$ or resp. $\mathcal{JS}_{com}(F,a) = \{out\}$. Otherwise we have that $\mathcal{L}_{grd}(a) = undec$ and therefore $undec \in \mathcal{JS}_{com}(F,a)$. To show (4) we use that $\mathcal{L}_{undec} = \emptyset$ for every stable labeling and thus $undec \notin \mathcal{JS}_{stb}(F,a)$. The remaining follows from the fact that for each semantic $\sigma \in \{prf, sem, stg\}$ it is the case that $\sigma(F) \neq \emptyset$ and thus also that $\mathcal{JS}_{\sigma}(F,a) \neq \{\}$. □

Proposition 1 shows that several justification statuses are not possible w.r.t. some semantics. It remains to show that the remaining ones are possible. To this end, we give examples for each of the remaining justification statuses.

Example 2. Here we use the AFs illustrated in Figure 1. First we consider the justification statuses $\{in\}$, and $\{out\}$. For semantics $\sigma \in \{grd, com, stb, prf, sem, stg\}$ we have that $\mathcal{JS}_{\sigma}(F_1,a) = \{in\}$ and $\mathcal{JS}_{\sigma}(F_1,b) = \{out\}$. Next let us consider $\{undec\}$ and $\sigma \in \{grd, adm, com, prf, sem, stg\}$, we have that $\mathcal{JS}_{\sigma}(F_2,a) = \{undec\}$. The following examples complete the picture for admissible and stable semantics: $\mathcal{JS}_{adm}(F_1,a) = \{in,undec\}$, $\mathcal{JS}_{adm}(F_1,b) = \{out,undec\}$, $\mathcal{JS}_{stb}(F_2,a) = \{\}$ and. $\mathcal{JS}_{\sigma}(F_3,a) = \{in,out,undec\}$ for $\sigma \in \{adm, com\}$.

For complete semantics it remains to give examples for the justification statuses $\{out,undec\}$ and $\{in,undec\}$. To this end let us consider AF_4 where we have that $\mathcal{JS}_{\sigma}(F_4,b) = \{out,undec\}$ and $\mathcal{JS}_{\sigma}(F_4,c) = \{in,undec\}$ for $\sigma \in \{com, prf, sem\}$.

Now let us consider the remaining justification statuses for stable, preferred, semi-stable and stage semantics. First to get the justification status $\{in,out\}$ we use the AF F_3 and obtain that $\mathcal{JS}_{\sigma}(F_3,a) = \{in,out\}$ for $\sigma \in \{stb, prf, sem, stg\}$. The picture is completed by the following observations on F_5 and F_6: $\mathcal{JS}_{stg}(F_5,a) = \{in,undec\}$, $\mathcal{JS}_{stg}(F_5,e) = \{out,undec\}$, $\mathcal{JS}_{stg}(F_5,c) = \{in,out,undec\}$ and $\mathcal{JS}_{prf}(F_6,e) = \mathcal{JS}_{sem}(F_6,e) = \{in,out,undec\}$. ◊

Wu and Caminada [18] mention that the complete labelings of an AF can be seen as *subjective but reasonable positions* given the conflicting information of the AF. We are now interested how the justification statuses are affected if we change our opinion about what are reasonable positions, i.e. we change semantics.

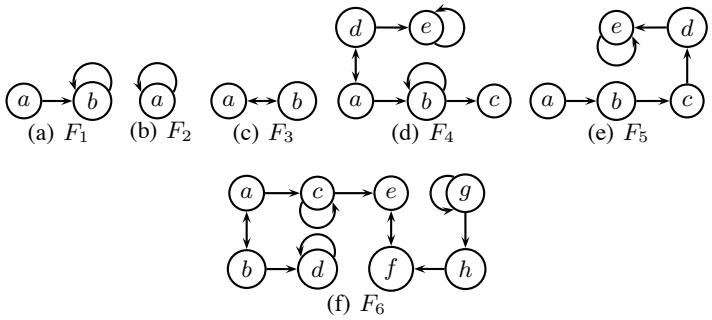


Fig. 1. Example argumentation frameworks F_1-F_7

Proposition 2. *Let $F = (A, R)$ be an AF and $a \in A$ then:*

$$\mathcal{JS}_{stb}(F,a) \subseteq \mathcal{JS}_{sem}(F,a) \subseteq \mathcal{JS}_{prf}(F,a) \subseteq \mathcal{JS}_{com}(F,a) \subseteq \mathcal{JS}_{adm}(F,a)$$
$$\mathcal{JS}_{grd}(F,a) \subseteq \mathcal{JS}_{com}(F,a) \subseteq \mathcal{JS}_{adm}(F,a)$$

Proof. The above $\subseteq$ follows from well-known relations between the semantics, namely that: each stable extensions is also a semi-stable extension; each semi-stable extensions is also a preferred extension; each preferred extension is a complete extension; each complete extension is an admissible set; and the grounded extension is a complete extension. □

Proposition 2 indicates that there are two kinds of scepticism. First there are different levels of scepticism, about what is an reasonable positions in an argument, that is captured by the semantics. Second depending on ones scepticism one tolerates different kinds of dispute about an argument. In an extension based setting this is captured be the reasoning modes, i.e. skeptical and credulous acceptance. In a labeling based setting this is captured by the justification statuses one is willing to accept. Mention that this kinds of skepticism differ conceptual from those introduced in [1], which are about relations between the extensions of different semantics whereas we are only interested in the status of an argument.

We can show that the justification statuses w.r.t admissible, complete and preferred semantics only differ on the $undec$ labels.

Proposition 3. *Let $F = (A, R)$ be an AF and $a \in A$.*

1. $\mathcal{JS}_{adm}(F,a) = \mathcal{JS}_{com}(F,a) \cup \{undec\}$
2. $\mathcal{JS}_{adm}(F,a) = \mathcal{JS}_{prf}(F,a) \cup \{undec\}$
3. $\mathcal{JS}_{com}(F,a) = \begin{cases} \mathcal{JS}_{grd}(F,a) & \text{if } a \in grd(F)^+ \\ \mathcal{JS}_{adm}(F,a) & \text{otherwise} \end{cases}$
4. $\mathcal{JS}_{com}(F,a) = \begin{cases} \mathcal{JS}_{grd}(F,a) & \text{if } a \in grd(F)^+ \\ \mathcal{JS}_{prf}(F,a) \cup \{undec\} & \text{otherwise} \end{cases}$

Proof. For (1) & (2) we first use the fact that $(\emptyset, \emptyset, A) \in adm(F)$ and hence adding $undec$ is always valid. Further by the definitions of the semantics we have that for

each $\mathcal{L} \in adm(F)$ there exists $\mathcal{L}' \in com(F)$ (resp. $\mathcal{L}' \in prf(F)$) such that $\mathcal{L}_{in} \subseteq \mathcal{L}'_{in}$ and $\mathcal{L}_{out} \subseteq \mathcal{L}'_{out}$. Hence $\mathcal{JS}_{adm}(F,a) \subseteq \mathcal{JS}_{com}(F,a) \cup \{undec\}$. Further as each complete (resp. preferred) labeling is also an admissible labeling we also get $\mathcal{JS}_{adm}(F,a) \supseteq \mathcal{JS}_{com}(F,a) \cup \{undec\}$. (3) In [18] they observe that if $a \in grd(F)^+$ then $\mathcal{JS}_{com}(F,a) = \mathcal{JS}_{grd}(F,a)$. Further they observe that if $a \notin grd(F)^+$ then $undec \in \mathcal{JS}_{com}(F,a)$ and by (1) then $\mathcal{JS}_{adm}(F,a) = \mathcal{JS}_{com}(F,a)$. Finally we obtain (4) by combining (2) and (3). □

4 Complexity Analysis

In this section we provide a formal analysis of the computational properties of the justification status. To this end we have to define an appropriate problem that concerns the justification status of an argument. The canonical decision problem is the problem of deciding if a given argument has a given justification status.

Definition 8. *The* justification status decision problem JS_σ*: Given an AF* $F = (A, R)$*,* $L \subseteq \{in, out, undec\}$ *and an argument* $a \in A$*. Is* $\mathcal{JS}_\sigma(F,a) = L$*.*

We can express skeptical acceptance as instance of JS_σ by choosing $L = \{in\}$, but we can't neither express credulous acceptance nor if an argument is at least weakly accepted. Thus we generalize the above definition to capture these cases:

Definition 9. *The* generalized justification status decision problem GJS_σ*: Given an AF* $F = (A, R)$*,* $L, M \subseteq \{in, out, undec\}$ *and an argument* $a \in A$*. Is* $L \subseteq \mathcal{JS}_\sigma(F,a)$ *as well as* $\mathcal{JS}_\sigma(F,a) \cap M = \emptyset$*.*

One can encode a problem instance of JS_σ as GJS_σ instance, by using $M = \{in, out, undec\} \setminus L$. Thus we have that GJS_σ is at least as computationally hard as JS_σ and as we will show, for each semantics σ from Definition 3 the problems JS_σ as GJS_σ have the same complexity. In the following we give hardness proofs for JS_σ, and membership proofs, i.e .algorithms, for GJS_σ which then implies the completeness of both problems.

First we prove some general observations about the problems JS_σ and GJS_σ and then present our results for specific $\sigma \in \{grd, adm, com, stb, prf, sem, stg\}$. Our first result allows us to propagate hardness results from skeptical acceptance to JS_σ.

Proposition 4. *If one of the problems* $\mathsf{coExists}_\sigma$, Skept'_σ *is* $\mathcal{C}$*-hard then* JS_σ *is* $\mathcal{C}$*-hard.*

Proof. Immediate by the fact that both $\mathsf{coExists}_\sigma$ and Skept'_σ can be easily encoded as instances of JS_σ. For the first, simply set $M = \{\}$ and for the latter set $M = \{in\}$. □

Next we provide a (generic) algorithm to decide GJS_σ and the corresponding complexity bounds:

Theorem 1. *If for semantics* σ *the problem* Ver_σ *is in a complexity class* $\mathcal{C}$ *then the problem* GJS_σ *is in the complexity class* $\mathrm{NP}^{\mathcal{C}} \wedge \mathrm{coNP}^{\mathcal{C}}$*.*

Proof. To prove $GJS_\sigma \in \text{NP}^{\mathcal{C}} \wedge \text{coNP}^{\mathcal{C}}$ we give an appropriate algorithm. Thus let us consider an arbitrary instance of GJS_σ, with AF $F = (A, R)$, an argument $a \in A$ and sets $L, M \subseteq \{in, out, undec\}$. First we give an $\text{NP}^{\mathcal{C}}$ algorithm to decide $L \subseteq \mathcal{JS}_\sigma(F, a)$:

- For each $l \in L$ guess a labeling $\mathcal{L}_l$ with $\mathcal{L}_l(a) = l$
- Test whether $\mathcal{L}_l \in \sigma(F)$ or not, using the $\mathcal{C}$-oracle.
- Accept if for each $l \in L$, $\mathcal{L}_l \in \sigma(F)$

As there are at most three labels $l \in L$ the guess is polynomial in the size of the input AF and thus we have an $\text{NP}^{\mathcal{C}}$ algorithm.

Next we show that there exits an $\text{coNP}^{\mathcal{C}}$ algorithm deciding $\mathcal{JS}_\sigma(F, a) \cap M = \emptyset$. For simplicity we give an $\text{NP}^{\mathcal{C}}$ algorithm that decides the complementary problem $\mathcal{JS}_\sigma(F, a) \cap M \neq \emptyset$.

- For each $l \in M$ guess a labeling $\mathcal{L}_l$ with $\mathcal{L}_l(a) = l$
- Test whether $\mathcal{L}_l \in \sigma(F)$ or not
- Accept if there exists an $l \in M$ such that $\mathcal{L}_l \in \sigma(F)$

Now as $GJS_\sigma(F, a)$ iff $L \subseteq \mathcal{JS}_\sigma(F, a)$ and $\mathcal{JS}_\sigma(F, a) \cap M = \emptyset$ we finally observe that $GJS_\sigma \in \text{NP}^{\mathcal{C}} \wedge \text{coNP}^{\mathcal{C}}$. □

The algorithm in the above proof is (worst case) optimal for semantics $\sigma \in \{adm, com, stb, sem, stg\}$, but, as we will show, not for $\sigma \in \{grd, prf\}$.

We now turn to study the complexity of the problems JS_σ, GJS_σ for specific semantics and start with grounded semantics.

Proposition 5. *The problems JS_{grd}, GJS_{grd} are* P*-complete.*

Proof. The membership in the class P follows by the fact that the unique grounded labelling can be computed in polynomial time (see Table 2). The hardness follows from the fact that Skept'_{grd} is P-hard and Proposition 4. □

Next we consider admissible and complete semantics. By Proposition 3 the justification statuses of admissible and complete semantics are closely related, thus it is not surprising that we can use the same construction to show hardness for both semantics.

Proposition 6. *The problems JS_{com}, GJS_{com},JS_{adm}, GJS_{adm} are* D^{P}*-complete.*

Proof. The membership is an immediate consequence of Theorem 1 and the fact that Ver_{com}, Ver_{adm} are in L.

We prove hardness by reducing the (D^{P}-hard) SAT-UNSAT problem to JS_{com} (resp. JS_{adm}). We assume that the two CNF formulas are given as a set of clauses, where each clause is a set over atoms and negated atoms (denoted by $\bar{x}$). For such CNFs $\varphi(X) = \bigwedge_{c \in C_\varphi} c$, $\psi(Y) = \bigwedge_{c \in C_\psi} c$ over variables X resp. Y with $X \cap Y = \emptyset$, define the AF $F_{\varphi,\psi} = (A, R)$ [2] with $A = X \cup \bar{X} \cup Y \cup \bar{Y} \cup C_\varphi \cup C_\psi \cup \{t, t'\}$ and

$$R = \{(z, \bar{z}), (\bar{z}, z) \mid z \in X \cup Y\} \cup \{(l, c) \mid l \in c, c \in C_\varphi \cup C_\psi\} \cup$$
$$\{(c, t), (c, c) \mid c \in C_\varphi\} \cup \{(c, t'), (c, c) \mid c \in C_\psi\} \cup \{(t, t'), (t', t)\}$$

[2] This reduction builds slightly modified standard translations (as defined in [13]) of both formulas and adds a mutual attack between them.

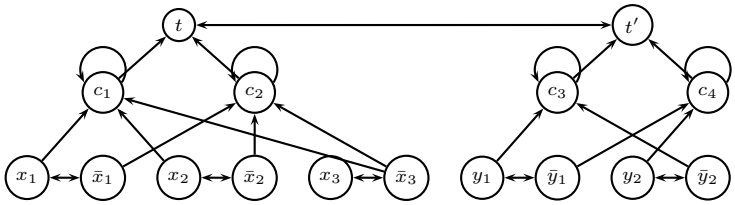


Fig. 2. AF $F_{\varphi,\psi}$ with $c_{\varphi,1} = \{x_1, x_2, \bar{x}_3\}$, $c_{\varphi,2} = \{\bar{x}_1, \bar{x}_2, \bar{x}_3\}$, $c_{\psi,3} = \{y_1, \bar{y}_2\}$, $c_{\psi,4} = \{\bar{y}_1, y_2\}$.

where $\bar{X} = \{\bar{x} \mid x \in X\}$, $\bar{Y} = \{\bar{y} \mid y \in y\}$ and t, t' are fresh arguments. See Figure 2 for an illustrating example. Let us first mention that $grd(F) = \emptyset$ and thus for each $a \in A$ it holds that $JS_{com}(F, a) = JS_{adm}(F, a)$. Hence it suffices to show that $(\varphi, \psi) \in$ SAT-UNSAT iff $JS_{com}(F, t) = \{in, undec\}$.

As $grd(F) = \emptyset$ we have that $undec \in JS_{com}(F, t)$, independently of the instance (φ, ψ). It is easy to see that[3] $in \in JS_{com}(F, t)$ iff φ is satisfiable and that $in \in JS_{com}(F, t')$ iff ψ is satisfiable. Further t' is the only not self-conflicting argument attacking t. Thus we have that $out \in JS_{com}(F, t)$ iff $in \in JS_{com}(F, t')$ iff ψ is satisfiable. To sum up, $JS_{com}(F, t) = \{in, undec\}$ iff φ is satisfiable and ψ is unsatisfiable, i.e. $(\varphi, \psi) \in$ SAT-UNSAT. □

Next we obtain completeness for stable semantics.

Proposition 7. *The problems JS_{stb}, GJS_{stb} are D^{P}-complete.*

Proof. The membership follows from Theorem 1 and the fact that Ver_{stb} is in L. Further as Skept'_{stb} is D^{P}-hard [13] we can use Proposition 4 to obtain D^{P}-hardness for JS_{stb} and thus D^{P}-completeness. □

In the case of preferred semantics Proposition 1 does not yield the optimal membership, i.e. it yields $\mathrm{D}_2^{\mathrm{P}}$ membership instead of $\mathrm{P}^{\Sigma_2^P[1]}$-membership. Hence we have to provide an algorithm for preferred semantics to show membership.

Proposition 8. *The problems JS_{prf}, GJS_{prf} are $\mathrm{P}^{\Sigma_2^P[1]}$-complete.*

Proof. To show *membership* we give a $\mathrm{P}^{\Sigma_2^P[1]}$ algorithm for GJS_{prf}. To this end, let us consider an arbitrary instance with AF $F = (A, R)$, an argument $a \in A$ and sets $L, M \subseteq \{in, out, undec\}$. First we observe that testing whether $in \in \mathcal{JS}_\sigma(F, a)$ (resp. $out \in \mathcal{JS}_\sigma(F, a)$) can be done with an NP-algorithm. Such an algorithm simple guesses an labeling $\mathcal{L} = (\mathcal{L}_{in}, \mathcal{L}_{out}, \mathcal{L}_{undec})$ and then tests if $\mathcal{L} \in \mathcal{L}_{adm}(F)$ and $a \in \mathcal{L}_{in}$ (resp. $a \in \mathcal{L}_{out}$). But testing for $undec \in \mathcal{JS}_\sigma(F, a)$ is not that easy, as it doesn't suffice to consider admissible labelings. Our algorithm proceeds as follows:
If $undec \in L \cup M$ (w.l.o.g. we assume $L \cap M = \emptyset$) we start the following Σ_2^P algorithm.

1. Guess labeling $\mathcal{L}$ with $\mathcal{L}(a) = undec$.
2. Test whether $\mathcal{L}$ is preferred or not, using a coNP-oracle.
3. Test whether $L \setminus \{undec\} \subseteq \mathcal{JS}_\sigma(F, a)$, using a NP-oracle.

[3] following the argument for the standard translation in [13].

4. Test whether, $(M \setminus \{undec\}) \cap \mathcal{JS}_\sigma(F, a) = \emptyset$, using a coNP-oracle.
5. If $undec \in L$ accept iff 2,3 and 4 hold;
   If $undec \in M$ accept iff 2 holds, or either 3 or 4 fails.

If $undec \in L$ we use the answer of the Σ_2^P algorithm as answer for the problem GJS_{prf}. Otherwise, i.e. $undec \in M$, we negate the answer of the Σ_2^P algorithm. In the case where $undec \notin L \cup M$ we omit steps 1 & 2 of the above algorithm, which leads to an $\mathrm{D^P}$-algorithm deciding GJS_{prf}.

We show *hardness* by reducing the QBF_2-problem, i.e. the problem of deciding if a QBF with one quantifier alternation is valid, to JS_{prf}. Let Φ be an instance of QBF_2, then Φ is either of the form (i) $\Phi = \forall X \exists Y \varphi(X, Y)$ or (ii) $\Phi = \exists X \forall Y \varphi(X, Y)$. In case (i) we can use the reduction from the Π_2^P-hardness proof of Skept'_{prf} [11] to encode ϕ as Skept'-problem, which itself can be encoded as instance of JS_{prf}. (see proof of Proposition 4). In case (ii) we can encode $\neg\Phi$ as coSkept'-problem in the same way as in (i), but it remains to show that the problem coSkept'_{prf} can be reduced to a JS_{prf}-instance. To this end consider an arbitrary AF $F = (A, R)$ with argument $t \in A$. If $(t, t) \in R$ then coSkept' is trivially true, thus for the remainder of the proof we assume $(t, t) \notin R$. We build the AF $F' = (A \cup \{g, u, v, w\}, R \cup \{(t, g), (g, g), (g, u), (v, w), (w, v)\} \cup \{(v, a) \mid a \in A \setminus \{t\}\})$ and claim that $JS_{prf}(F', u) = \{in, undec\}$ iff the argument t is not skeptical accepted in F.

As $\{v, t, u\}$ is conflict-free and attacks all the other arguments it is a preferred extension of F' and thus $in \in JS_{prf}(F', u)$. Further as g is the only attacker of u and $(g, g) \in R'$ we also have that $out \notin JS_{prf}(F', u)$. It remains to show that $undec \in JS_{prf}(F', u)$ iff t is not skeptical accepted in F.

To show the "if"-part, consider $E \in prf(F)$ with $t \notin E$. It holds that $E \cup \{w\} \in adm(F')$ and thus there exists $E' \in prf(F')$ such that $E \cup \{w\} \subseteq E'$. As $w \in E'$ we have that $v \notin E'$ and as w, u, g do not attack any attacker of t, we also have that $t \notin E'$. Hence also $u \notin E'$ and thus $undec \in JS_{prf}(F', u)$.

Next, to show the "only if"-part, assume $undec \in JS_{prf}(F', u)$. Then there exists an $E' \in prf(F')$ such that $u \notin E'$, thus by construction F' also $t \notin E'$ and $v \notin E'$. Moreover we have that $E = E' \cap A \in adm(F)$ and $E' = E \cup \{w\}$. Next we show that also $E \in prf(F)$. Let us assume, towards a contradiction, that $E \notin prf(F)$ then there exists $S \in prf(F)$ with $E \subset F$. But by construction $S' = S \cup \{w\} \in adm(F')$ and as $E \subset S$ also $E' \subset S'$, a contradiction to our assumption $E' \in prf(F')$. We have that E is a preferred extension of F with $t \notin E$, i.e. t is not skeptical accepted in F. □

Finally we consider semi-stable and stage semantics. To this end we first reclaim some concepts and results from the literature.

Definition 10. *Given a $QBF^2_\forall$ formula $\Phi = \forall Y \exists Z \bigwedge_{c \in C} c$, with C a set of clauses, we define the AF $\mathcal{S}_\Phi = (A, R)$, where*

$$
\begin{aligned}
A = {} & \{t, \bar{t}, b\} \cup C \cup Y \cup \bar{Y} \cup Y' \cup \bar{Y}' \cup Z \cup \bar{Z} \\
R = {} & \{(c, t) \mid c \in C\} \cup \{(t, \bar{t}), (\bar{t}, t), (t, b), (b, b)\} \cup \\
& \{(x, \bar{x}), (\bar{x}, x) \mid x \in Y \cup Z\} \cup \\
& \{(y, y'), (\bar{y}, \bar{y}'), (y', y'), (\bar{y}', \bar{y}') \mid y \in Y\} \cup \\
& \{(l, c) \mid \textit{literal } l \textit{ occurs in } c \in C\}.
\end{aligned}
$$

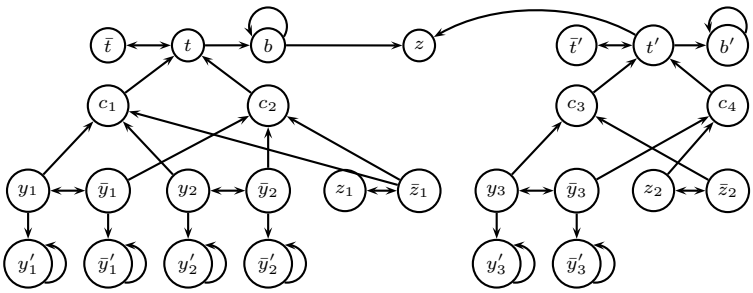


Fig. 3. AF $F_{\varphi,\psi}$ with $c_{\varphi,1} = \{y_1, y_2, \bar{z}_1\}, c_{\varphi,2} = \{\bar{y}_1, \bar{y}_2, \bar{z}_1\}, c_{\psi,3} = \{y_3, \bar{y}_3\}, c_{\psi,4} = \{\bar{z}_1, z_2\}$.

Theorem 2 ([14]). *For a semantics $\sigma \in \{sem, stg\}$, a $QBF^2_\forall$ formula Φ is a valid iff t is skeptically accepted in F_Φ w.r.t. σ iff $\bar{t}$ is not credulously accepted in F_Φ w.r.t. σ.*

Using this we obtain the exact complexity of JS_{sem}, GJS_{sem},JS_{stg} and GJS_{stg}.

Proposition 9. *The problems JS_{sem}, GJS_{sem} are D_2^P-complete.*

Proof. The membership follows from Theorem 1 and the fact that $\mathsf{Ver}_{sem} \in$ coNP. To show hardness we reducing the (D_2^P-hard) $QBF^2_\forall$-co$QBF^2_\forall$ problem to JS_{sem}. The $QBF^2_\forall$-co$QBF^2_\forall$ problem remains hard when the $QBFs$ Φ, Ψ are in CNF with satisfiable clause sets ϕ, ψ. In the following we will tacitly assume Φ, Ψ are of that form.

We define the AF $F_{\Phi,\Psi} = \mathcal{S}_\Phi \dot\cup \mathcal{S}_\Psi \cup (\{z\}, \{(b,z),(t',z)\})$. Here the symbol "$\dot\cup$" denotes the disjoint union, i.e. we rename each argument of $\mathcal{S}_\Psi$ that also occurs in $\mathcal{S}_\Phi$, and denote the renamed argument $t \in \mathcal{S}_\Psi$ by t'. One can see that $E_1 \in sem(\mathcal{S}_\Phi)$, $E_2 \in sem(\mathcal{S}_\Psi)$ iff either $E_1 \dot\cup E_2 \in sem(\mathcal{S}_\Psi)$ or $E_1 \dot\cup E_2 \cup \{z\} \in sem(F_{\Phi,\Psi})$. We have to show that $(\Phi, \Psi) \in QBF^2_\forall$-co$QBF^2_\forall$ iff $\mathcal{JS}_{sem}(F_{\Phi,\Psi}, z) = \{in, out\}$.

For the "only if"-part, let us assume (Φ, Ψ) is a valid instance of $QBF^2_\forall$-co$QBF^2_\forall$. Then by Theorem 2 for each $E_1 \in sem(\mathcal{S}_\Phi)$ holds that $t \in E_1$ and thus also for each $E \in sem(F_{\Phi,\Psi})$ holds $t \in E$. As Ψ is not valid, by Theorem 2, there exists $E_2 \in sem(\mathcal{S}_\Psi)$ such that $t' \notin E_2$, hence $\bar{t}' \in E_2$. Thus there exist an $E \in sem(F_{\Phi,\Psi})$ such that $t \in E$ and $\bar{t}' \in E$. But then E defends z and by the maximality of semi-stable extension we have $z \in E$ and therefore $in \in \mathcal{JS}_{sem}(F_{\Phi,\Psi}, z)$. Further as ψ is satisfiable there also exists an $E_2 \in sem(\mathcal{S}_\Psi)$ such that $t' \in E_2$, hence an $E \in sem(F_{\Phi,\Psi})$ with $t' \in E$ and thus $out \in \mathcal{JS}_{sem}(F_{\Phi,\Psi}, z)$. As for each $E \in sem(F_{\Phi,\Psi})$ clearly either $t' \in E$ or $\bar{t}' \in E$ we have $undec \notin \mathcal{JS}_{sem}(F_{\Phi,\Psi}, z)$.

For the "if"-part, let us assume $(\Phi, \Psi) \notin QBF^2_\forall$-co$QBF^2_\forall$. Then either (i) Ψ is valid or (ii) Φ and Ψ are invalid. We have to show that $\mathcal{JS}_{sem}(F_{\Phi,\Psi}, z) \neq \{in, out\}$.

(i) If Ψ is valid then, by Theorem 2, for each $E_2 \in sem(\mathcal{S}_\Psi)$ it holds that $t' \in E_2$ and thus also for each $E \in sem(F_{\Phi,\Psi})$. Hence $\mathcal{JS}_{sem}(F_{\Phi,\Psi}, z) = \{out\}$.
(ii) If Φ is not valid then, by Theorem 2, there exists $E_1 \in sem(\mathcal{S}_\Phi)$ such that $t \notin E_1$. If also Ψ is not valid then there exists an $E \in sem(F_{\Phi,\Psi})$ with $t \notin E$ and $t' \notin E$. Thus z is neither attacked by E nor defended E, hence $undec \in \mathcal{JS}_{sem}(F_{\Phi,\Psi}, z)$.

□

Proposition 10. *The problems JS_{stg}, GJS_{stg} are D_2^P-complete.*

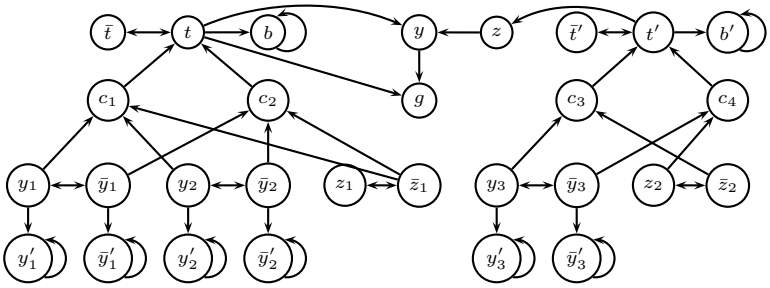


Fig. 4. AF $F_{\varphi,\psi}$ with $c_{\varphi,1} = \{y_1, y_2, \bar{z}_1\}$, $c_{\varphi,2} = \{\bar{y}_1, \bar{y}_2, \bar{z}_1\}$, $c_{\psi,3} = \{y_3, \bar{y}_3\}$, $c_{\psi,4} = \{\bar{z}_1, z_2\}$.

Proof. The membership is immediate by Theorem 1 and the fact that $\mathsf{Ver}_{stg} \in$ coNP.

To prove hardness we reducing the ($\mathrm{D}_2^{\mathrm{P}}$-hard) $QBF^2_\forall$-co$QBF^2_\forall$ problem to JS_{stg}. and as before we assume that the clause sets of the QBFs are satisfiable. For our reduction we define the AF $F_{\Phi,\Psi} = \mathcal{S}_\Phi \dot\cup \mathcal{S}_\Psi \cup (\{y, z, g\}, \{(t, y), (t, g), (t', z), (g, g), (y, g), (z, y)\})$ (again $\dot\cup$ denotes the disjoint union). One can see that $E_1 \in stg(\mathcal{S}_\Phi)$, $E_2 \in stg(\mathcal{S}_\Psi)$ iff $E_1 \dot\cup E_2 \cup T \in stg(F_{\Phi,\Psi})$ with $T \subset \{y, z\}$. We have to show that (Φ, Ψ) is an valid instance of $QBF^2_\forall$-co$QBF^2_\forall$ iff $\mathcal{JS}_{stg}(F_{\Phi,\Psi}, z) = \{in, out\}$.

For the "only if"-part, let us assume $(\Phi, \Psi) \in QBF^2_\forall$-co$QBF^2_\forall$. Then by Theorem 2 for each $E_1 \in stg(\mathcal{S}_\Phi)$ holds that $t \in E_1$ and also for each $E \in stg(F_{\Phi,\Psi})$ holds $t \in E$. As Ψ is not valid, by Theorem 2, there exists $E_2 \in stg(\mathcal{S}_\Psi)$ such that $t' \notin E_2$, hence $\bar{t}' \in E_2$. Thus there exist an extension $E \in stg(F_{\Phi,\Psi})$ such that $t \in E$ and $\bar{t}' \in E$. But then E attacks y and $E \cup \{z\}$ is conflict-free. By the maximality of stage extension we have $z \in E$ and thus $in \in \mathcal{JS}_{stg}(F_{\Phi,\Psi}, z)$. Further as ψ is satisfiable there also exists an $E_2 \in stg(\mathcal{S}_\Psi)$ such that $t' \in E_2$, hence an $E \in stg(F_{\Phi,\Psi})$ with $t' \in E$ and thus $out \in \mathcal{JS}_{stg}(F_{\Phi,\Psi}, z)$. As for each $E \in stg(F_{\Phi,\Psi})$ clearly either $t' \in E$ or $\bar{t}' \in E$ we have $undec \notin \mathcal{JS}_{stg}(F_{\Phi,\Psi}, z)$.

For the "if"-part, let us assume that $(\Phi, \Psi) \notin QBF^2_\forall$-co$QBF^2_\forall$. Then either (i) Ψ is valid or (ii) Φ and Ψ are invalid. We have to show that $\mathcal{JS}_{stg}(F_{\Phi,\Psi}, z) \neq \{in, out\}$.

(i) If Ψ is valid then, by Theorem 2, for each $E_2 \in stg(\mathcal{S}_\Psi)$ it holds that $t' \in E_2$ and thus also for each $E \in stg(F_{\Phi,\Psi})$. Hence $\mathcal{JS}_{sem}(F, z) = \{out\}$.

(ii) If Φ is not valid then, by Theorem 2, there exists $E_1 \in sem(\mathcal{S}_\Phi)$ such that $t \notin E_1$. If also Ψ is not valid then there exists an $E_2 \in sem(\mathcal{S}_\Psi)$ such that $t' \notin E_2$. We have that $E = E_1 \cup E_2 \cup \{y\} \in stg(F_{\Phi,\Psi})$, as it is the only conflict free superset of $E_1 \cup E_2$ such that $g \in E \cup E^+$. As neither $z \in E$ nor $E \rightarrowtail z$ we have that $undec \in \mathcal{JS}_{sem}(F, z)$.

□

We showed that for all semantics σ under our consideration, except grounded semantics, the problems JS_σ, GJS_σ are even harder than NP. Thus one might be interested in tractable fragments, i.e. classes of problem instances that can be solved in polynomial time. First, there are AFs having a special graph structure [10]. It is easy to see that the known tractable fragments for Cred_σ and Skept_σ (i.e. acyclic AFs, symmetric AFs, bipartite AFs, AFs of bounded tree-width / clique-width) are also tractable fragments for JS_σ, GJS_σ. Second, one can consider instances that test for a fixed justification status, e.g. weak acceptance. For instance consider $JS_{com}(F, a) = \{in\}$, then

we can use that $JS_{com}(F, a) = \{in\}$ iff a is in the grounded extension [18]. As the grounded extension can be computed in P, we can decide $JS_{com}(F, a) = \{in\}$ in P instead of using a $\mathrm{D^P}$-algorithm. However, a full analysis of such fragments is beyond the scope of this paper.

σ	*grd*	*adm*	*com*	*stb*	*prf*	*sem*	*stg*
Cred_σ	P-c	NP-c	NP-c	NP-c	NP-c	Σ_2^P-c	Σ_2^P-c
Skept'_σ	P-c	trivial	P-c	$\mathrm{D^P}$-c	Π_2^P-c	Π_2^P-c	Π_2^P-c
JS_σ	P-c	$\mathrm{D^P}$-c	$\mathrm{D^P}$-c	$\mathrm{D^P}$-c	$\mathrm{P}^{\Sigma_2^P[1]}$-c	D_2^P-c	D_2^P-c
GJS_σ	P-c	$\mathrm{D^P}$-c	$\mathrm{D^P}$-c	$\mathrm{D^P}$-c	$\mathrm{P}^{\Sigma_2^P[1]}$-c	D_2^P-c	D_2^P-c

Table 2. Overview of Complexity results ($\mathcal{C}$-c denotes completeness for class $\mathcal{C}$)

5 Conclusion

In this paper we generalized the labeling-based justification-status, introduced by Wu and Caminada [18], to arbitrary argumentation semantics. In particular we considered the justification-status w.r.t. grounded, admissible, complete, preferred, semi-stable and stage semantics and provided a comparison between different semantics.

We have studied the computational complexity of decision problems associated to the justification status of an argument w.r.t different semantics (see Table 2). The overall picture is that complexity has slightly increased compared to the complexity of credulous and skeptical acceptance. They main reason for this is that, in contrary to credulous / skeptical acceptance, we have to do provide both, witnesses for labels to be in the justification status and proofs for other labels to be not in the justification status. In general this causes two orthogonal sources of complexity, with two notable exceptions. First, when considering grounded semantics, we have just one labeling which can be computed in polynomial time and thus reasoning problems remains simple. Second, for preferred semantics we have that one can decide the labels in, out using admissible sets which is much easier than using preferred extensions. However for deciding the label $undec$ admissible sets are not sufficient and we have to use (proof procedures for) preferred extensions. Thus the complexity in case of preferred semantics is dominated by the complexity of deciding whether $undec$ is in the justification status or not.

Moreover let us mention that the complexity results in Table 2 strongly correlate with expressibility results presented in [15]. There the authors study (faithful) translations between different argumentation semantics and the results there indicate an expressibility hierarchy of admissibility-based semantics. That is we have four levels of expressibility: grounded semantics are on the first level; admissible, complete and stable semantics on the second level; preferred semantics on the third level; and semi-stable semantics are on the fourth level. Ordering the semantics w.r.t. complexity of JS_σ (resp. GJS_σ) leads exactly to the same hierarchy of semantics[4], whereas this nei-

[4] The relation between stage and admissibility-based semantics is not fully determined in [15], but putting it at the fourth level would be consistent with the results there.

ther holds for credulous nor for skeptical acceptance. Thus it seems that the problems JS_σ, GJS_σ capture more of the nature of argumentation semantic than credulous or skeptical acceptance.

Acknowledgments

The author is grateful to Martin Caminada, Stefan Woltran and the anonymous referees for fruitful discussions and valueable comments which helped to improve the paper.

References

1. Pietro Baroni and Massimiliano Giacomin. Comparing argumentation semantics with respect to skepticism. In Proc. *ECSQARU 2007*, pages 210–221.
2. Pietro Baroni and Massimiliano Giacomin. Semantics of abstract argument systems. In *Argumentation in Artificial Intelligence*, pages 25–44. Springer, 2009.
3. Martin Caminada. Semi-stable semantics. In Proc. *COMMA 2006*, pages 121–130.
4. Martin Caminada. On the Issue of Reinstatement in Argumentation. In Proc. *JELIA 2006*, pages 111–123.
5. Martin Caminada. A labelling approach for ideal and stage semantics. *Argument & Computation*, 2:1–21, 2011.
6. Martin Caminada and Dov M. Gabbay. A logical account of formal argumentation. *Studia Logica*, 93(2):109–145, 2009.
7. Sylvie Coste-Marquis, Caroline Devred, and Pierre Marquis. Symmetric argumentation frameworks. In Proc. *ECSQARU 2005*, pages 317–328.
8. Yannis Dimopoulos and Alberto Torres. Graph theoretical structures in logic programs and default theories. *Theor. Comput. Sci.*, 170(1-2):209–244, 1996.
9. Phan Minh Dung. 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.
10. Paul E. Dunne. Computational properties of argument systems satisfying graph-theoretic constraints. *Artif. Intell.*, 171(10-15):701–729, 2007.
11. Paul E. Dunne and Trevor J. M. Bench-Capon. Coherence in finite argument systems. *Artif. Intell.*, 141(1/2):187–203, 2002.
12. Paul E. Dunne and Martin Caminada. Computational complexity of semi-stable semantics in abstract argumentation frameworks. In Proc. *JELIA 2008*, pages 153–165.
13. Paul E. Dunne and Michael Wooldridge. Complexity of abstract argumentation. In Guillermo Simari and Iyad Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 85–104. Springer US, 2009.
14. Wolfgang Dvořák and Stefan Woltran. Complexity of semi-stable and stage semantics in argumentation frameworks. *Inf. Process. Lett.*, 110(11):425–430, 2010.
15. Wolfgang Dvořák and Stefan Woltran. On the intertranslatability of argumentation semantics. In Proc. *NonMon@30*, October 22-25 2010.
16. Bart Verheij. Two approaches to dialectical argumentation: admissible sets and argumentation stages. In Proc. *NAIC'96*, pages 357–368.
17. Bart Verheij. A Labeling Approach to the Computation of Credulous Acceptance in Argumentation. In Proc. *IJCAI 2007*, pages 623-628.
18. Yining Wu and Martin Caminada. A labelling-based justification status of arguments. *Studies in Logic*, 3(4):12–29, 2010.

A Goal-Oriented Dynamical Computation of Preference of Strategy

Jenny Eriksson Lundström[1]

Dept. of Informatics and Media, Uppsala University, Sweden
jenny.eriksson@im.uu.se

Abstract. We introduce a mechanism for goal-oriented deliberation of strategies in argumentation games that does not rest on a probability distribution for the different actions or goals. Our mechanism is intended to investigate the compatibility of a given argument to an agent's superior (preferred goals), and to dynamically propose new lines of reasoning in light of new information from the adversary. We show how to construct a theory of acceptable strategies that, according to some preference order of the goals, maximizes the preferred goal-fulfilment while it acknowledges the influence of perspectives and context of the specific case. In this paper, we illustrate the approach by modeling the strategy for winning an adversarial argumentation game as achievable by several strategies that may or may not include the realization of intermediate (less preferred) goals. We show how the choice of pursuing or passing on such intermediate goals will influence a player's chance of reaching the superior goal.

1 Introduction

In this paper we approach strategies of adversarial argumentation games, in which two agents debate over one topic. An adversarial argumentation game can be viewed as a proof-theoretical process where the agents construct a proof for a proposition by taking turns in presenting arguments. The strategies for successful argumentation could be perceived as plans that express how to commit one's opponent to a certain standpoint, cf. the persuasion dialogue schema by Walton [30]. Analogously knowledge for optimizing an agent's debating performance can be defined as plans that express how a certain goal can be accomplished.

Planning understood as construction of plans how to achieve one's goals is called teleological planning cf. *e.g.* [7]. As stated in [15] p1.:

> Goals provide us with guidance when we must decide what to do. An action that promotes a goal should pro tanto be performed, while an action that detracts from a goal should pro tanto be avoided. Similarly, a rule that promotes a goal should pro tanto be adopted, while a rule that detracts from a goal should pro tanto be rejected.

However, agents, in a non-trivial setting, often hold several goals. For these situations, we note that sometimes the optimum strategy for achieving one goal

may lead to the inability to accomplish another goal cf. eg. [14]. Thus, in a particular dispute, such strategies cannot be specified and enacted without taking into account the setting and particular circumstances of the dispute. In a dynamic setting with complex goals or repeated interaction amongst the agents, sophisticated plans may be needed. In other words, the success of an agent to fulfil its goals, becomes dependent on the ability of the agent to reflect on how the valid chains of inference found established between the the available actions and a particular goal of the agent, effects the possibility to fulfil other goals.

[25] presents a probability distribution approach to strategic argumentation games. It adds a game theoretical aspect to the strategy of argumentation games, and it bases its approach on the assumption of perfect information to calculate the optimum strategy for the argumentation games.

Unfortunately, as discussed by [25], an *a priori* computation of the preferred expected utility cannot be readily accounted for in most disputes.

Our focus in this paper is to present a metalogic computational approach for addressing such teleological planning for agents' strategies in adversarial argumentation games. In our approach the preferences are set by the game specific circumstances and the dynamical progression of the game, rather than via the general probabilities generated from strong assumptions on the available knowledge of the adversary. The preferences of arguments are expressed without reference to strength of warrant, cf. eg. [10]. Our approach draws on the stratified representation of legal knowledge of [16], hence the logical computation of optimum strategies is accomplished by the formalization of provability as presented below. In order to establish a mechanism equipped to reason on the strategies, we will utilize the methodology of [16] inspired by Kleene's [17] description of formalization. This methodological approach draws on interpretation to assess the strategies available in the argumentation game setting. In this way we avoid the need for an *a priori* known formal theory that is "isomorphic" to the informal theory it characterizes. Instead, the formal theory is constructed by means of schemata.

In the following Sections 2 and 3, we elaborate on argumentation games and we introduce the notion of complex argumentation games and the metalogical toolbox for establishing the mechanism for addressing optimized teleological planning for agents in adversarial argumentation games. In the next Sections 4, 5 and 6 we present an example known from the literature on strategic argumentation games, and its formalization by means of metalogic. In Section 7 we briefly discuss related research on teleological planning. The paper is concluded in Section 8.

2 Argumentation Games

We consider an adversarial argumentation game as a game where we have two agents called the *Proponent* and the *Opponent*. Each agent is equipped with a set of arguments, a subset of which the agents *move* as an exchange of utterances or speech acts, *i.e.*, take turns in putting forward. Expressed at the level

of information analysis and modeling, utterances as well as movement of physical objects are described as speech acts put forward according to protocols for the game. In the simple case presented in this paper, the speech acts consists of propositions expressing facts, legal statutes, evidence etc. in some formal language. As the goals of the agents are conflicting [1], each agent strives to justify its conclusion and refute the adversary's claim, while adhering to the particular protocol scheme that governs the particular game. The protocol of the game directs the moves of the agents according to the intended rules for the game. It establishes the criteria for termination of the game, and the meaning of the terminal state *i.e.* outcome for a particular game.

A basic protocol for the admissible moves of the players could be, for the proponent, that the current move attacks the previous move of the opponent, and that the main claim (the content of the dispute) follows from the arguments assessed as currently valid. For the opponent we have that the arguments of the move attack the previous move, and the main claim is not derivable. Even though more complex winning conditions are possible, by a basic protocol an agent wins the dialogue game when the adversary is out of admissible moves. For many cases these argumentation games are expected to come out as a draw. That is, the proponent can defend her proposition against attack, but on the other hand she is not able to justify it by logical means given the available evidence and presented counterarguments.

In addition we need a mechanism for deriving logical consequences of the claims given the available background knowledge. Our focus are adversarial argumentation games, hence we formally have to accommodate the opposing claims and arguments put forward by the agents. Hence, we have to accommodate that arguments may be defeated by new information. By a monotonic consequence relation, if one agent claims p and the adversary claims $\neg p$ these claims together would constitute an unacceptable contradiction. Therefore in formal argumentation one often resorts to defeasible reasoning, where new information can be introduced without causing logical contradictions. Here defeasible propositions are often used for expressing assertions which are to be rejected by stronger assertions, in particular by classical indisputable propositions.

In contrast to the approach presented in the defeasible reasoning framework of [11], and the defeasible argumentation system of [23, 22, 21, 24], we consider a key aspect of argumentation to be that the agents in general advance their available knowledge and claims stepwise, in full knowledge of the previously presented knowledge by their adversary. Hence, a salient feature of this approach is that the knowledge of an agent is partitioned into private knowledge and common knowledge. Knowledge becomes available as an agent expresses the knowledge as arguments, whereas the private knowledge, which is unknown to

[1] Naturally this does not imply that agents in an adversarial scenario cannot cooperate in parts of their interaction. Conflicting sub-goals often are found amongst agents in a cooperative setting, while all the same in an adversarial discussion one agent may partially accept a proposal of her adversary as it provides for a stronger justification of her case [29].

the adversary, may be withheld for some time or during the whole dispute with the intent to optimize the own chances of success. Thus, a salient feature of the framework of [9] is the formalization of the interaction between the logical layer of defeasible argumentation and the dynamic progression in the argumentation dialogue.

3 Complex Argumentation Games

We consider a complex argumentation game an adversarial argumentation game where several, possibly conflicting, goals are dependent on the strategy of actions displayed in the argumentation game. A preference order of these goals, or a hierarchical ordering, by which a particular goal g_i is an intermediate goal of its superior goal g_{i+1}, are assumed.

For complex argumentation games, we note that winning an intermediate game or realizing an intermediate goal might lead to a loss of a superior goal, while concession or withholding an argument leading to a loss of an intermediate game may strengthen a particular agent's position and enable a win of the superior goal.

3.1 Strategies for Complex Argumentation Games

Problem-solving knowledge can be defined as plans that express how a certain task can be accomplished. During problem-solving an agent could use poor strategies, hence resulting in poor or non-optimal solutions. This could be the result from using incorrect strategies, slips or inefficient problem-solving techniques for examining too much information [12]. We initially stated that a strategy can be understood as a relation between the available actions of the strategy and the goal to be reached. We now consider an ideal strategy for a particular argumentation game as an ideal mapping from the available set of arguments to the set of goals of the agent. For a single goal of the agent, often expressed as proving the keyclaim or its refutation in a particular argumentation game, the agent simply moves to try to win the game by the available arguments. In a complex argumentation game however, a strategy to fulfil an intermediate goal may very well lead to that a superior goal cannot be accounted for. From the perspective of the agent, a preferred strategy to realize an intermediate goal has to represent a partial description on realizing the superior goal. For a particular agent involved in a dispute, a strategy signifies the connection between the agent's environment, a preference for a particular situation in the environment and such a particular situation occurring in that environment. Hence, only after deliberation that takes into account all the goals of a higher order of priority to the agent, the context and dialectical setting, the appropriate strategy for pursuing the intermediate goal can be determined.

It means that in a complex argumentation game the value obtained by ideally exploiting the available arguments for accomplishing a certain goal depends on or may even be countered by a strategy for achieving an intermediate goal.

Thus, the set of available strategies constitutes a canon of sometimes incoherent fragments of an ideal strategy for achieving the agent's superior goals. As such it allows for multiple interpretations of what to rule as an acceptable strategy for a particular argumentation game played by a particular agent in a given situation. By a goal-oriented approach, only after establishing the effect of the particular strategy on the agent's superior goal, the appropriateness of a certain strategy could be determined for the agent in a particular argumentation game. Thus, an ideal strategy needs to determine the acceptable and meaningful strategies of the intermediate goals of the argumentation game and dynamically compute for each such intermediate goal the strategies acceptable and meaningful for accomplishing every superior goal of the agent.

3.2 Stratification for Adaptation of the Strategy

We consider the structure of the 'ideal' strategy of an agent as a hierarchy of theories where the language of each level i consists of all strategies for achieving a goal. These are expressed as schemata and conditions for their specialization on the level $i + 1$. Each level T_i is characterized in the adjacent level T_{i+1} by schematic descriptions of its sentences and rules for deciding which specializations of these are meaningful and acceptable for the above superior goal.

We draw on metalogic augmented with reflection to formalize provability between an object theory T and an object language formula A named $n(T)$ and $n(A)$ respectively in the metalanguage. Thus, we assume the two-place predicate $Demo(n(T), n(A))$ to represent $T \vdash A$ i.e. logic provability encoding in the metalanguage. Hence, *Demo* is expressed as a non-ground metainterpreter (cf. [18]) augmented with the following upward reflection rule:

$$Demo(n(T_i), n(A)) \leftarrow$$
$$Demo(n(T_{i+1}), n(Demo(n(T_i), n(A)))).$$

The upward reflection rule allows the construction and assessment of which strategies for a given intermediate goal that are coherent with more superior goals. Below we let a formula A name itself autonomously at the metalevel.

3.3 A Methodology for the Interpretation of a Formal Theory of Strategies

For the study of the mechanisms of reasoning here adopted, we will utilize the methodology of [16] inspired by Kleene's [17] description of formalization.

Kleene [17] describes the process of formalization as involving three separate and distinct theories:

IT the informal theory to be formalized in the formal system
OT the formal system or object theory
MT the metatheory, which describes, analyses or even constructs the formal system

Theory OT is not a theory in the common sense, but a system of symbols and objects built from symbols described from MT. Theories IT and MT are informal and do not have an exactly determined structure as does OT.

In MT the formal system OT can be investigated by methods without making use of IT, a.k.a. as metamathematical methods if finitary.

Alternatively, in MT an interpretation of OT can be exploited under which it constitutes a formalization of IT. That is, IT is analyzed by "selecting and stereotyping fundamental concepts, presuppositions and deductive connections, and thus eventually arrive at a formal system" [17], i.e., OT.

In [16], MT is understood in this latter sense and is itself subjected to formalization and subsequent implementation as a metalogic program. The aim is to mirror (legal) reasoning as the stepwise construction of a formal theory by instantiating schemata into meaningful rules which will be assessed for acceptability by metarules. The metarules are obtained in the same way, etc.

MT is not to be confused with the metalevels in the object theory OT itself. MT takes the whole multilayered OT as its object theory.

Thus, OT is constructed to fit the circumstances of the case by recognizing an interpretation under which it constitutes a formalization of IT. The construction process is subject to two constraints. Rule candidates must be meaningful, i.e., instances of predefined rule schemata, and accepted by the rules of interpretation at the next level in OT.

This methodological approach should be contrasted to addressing the formal theory without making use of interpretation. However as pointed out by [16] this latter approach would require the access to an *a priori* known formal theory which could constitute an "isomorphic" characterization of the informal theory.

This architecture is useful for domains where an infinite regression may be an issue. As [16], we note that by allowing the object theory OT to be stratified into the (hierarchical) layers corresponding to the domain, pragmatical reasons would make infinite regression a theoretical issue, cf. the discussion of open-texture and infinite regression in [3].

In other words, as soon as there are no more rule-schemata to be instantiated by a adjacent higher level, the upward reflection process is to stop. The use of MT as a control mechanism for this purpose, fits well with the above described Kleene-structure.

3.4 Reflection for Establishing Optimum Strategies

Our approach to establish optimum strategies requires all strategies of an intermediate goal at level i to be assessed by a theory of what constitutes acceptable strategies also in the light of higher level concerns, *i.e.* metarules, situated at the adjacent higher level *i+1*. It means that our metatheory characterizes the object theory. As such it provides the upward reflective feature of our model, cf. [18, 8].

In order to allow for collated strategic concerns to be included in the theory, the resulting object level is stratified. It is important that the object level merely constructs the theory (deliberative reasoning) rather than also controlling and

monitoring it (metamanagement). Therefore our model is augmented by the introduction of a second intelligent layer that is to handle the metamanagement, cf. the structure of Kleene described in Section 3.3. This metalayer controls and evaluates the object level. Hence, the overall goal of the metalayer is to monitor and adapt the object layer processes. The metalayer is modeled as a deliberative layer integrated on top of the object level. The symbolic representation of deliberative layer allows for the design of intelligent behavior needed to reason on ideal strategies of the subordinate layer. Incorporating a proactive feature, the knowledge on possible erroneous strategies is augmented by the knowledge available from the different strategies in the second object level.

As the availability of the strategies and their performance is to depend on the adversary's strategies, only in an ideal environment, the metalevel of one agent is fully able to extract all necessary data for strategic analysis. As means of facilitating the metamanagement by enabling efficient and comprehensive metalevel reasoning, we introduce a stratified theory OL_b that provides a more comprehensive representation of the domain than the object level theory OL_a.

Occasionally, for efficiency purposes, the metalevel representation of the object level formulas might be insufficient for some reasoning required in establishing logic provability between an object theory T and an object language formula A. We accommodate the need for computations on the formula of the object level rather than its representations *n(T)* and *n(A)* at the metalevel, as *Demo* is augmented with a downward reflection rule that caters for simulations of the object theory reasoning to be conducted in a full scale model T_{i_b}, although not executed in the actual object layer.

$$\begin{aligned} &Demo(T_{i+1}, Demo(T_i, A)) \leftarrow \\ &\quad Demo(T_{i_b}, A) \wedge \\ &\quad not(Demo(T_{i+1}, not(Demo(T_i, A)))). \end{aligned}$$

The downward reflection rule facilitates the superior goal strategy optimization of the agent as the metatheory has precedence over the object theory at each level of the stratified system. $not(Demo(T_{i+1}, not(Demo(T_i, A))))$ of the above described downward reflection rule. This results in circumventing the problem of non-cooperativeness and fragmentary as well as missing knowledge on the adversary's future moves and preferences, as we, for efficient reasoning purposes, deal with different levels of granularity in the metalevel representation of the object theory. To prevent the metalevel deliberations from being constrained by insufficiencies due to the level of abstraction of the metarepresentation of the object theory, some computations in the metatheory may be based on actual formulas in the object theory OT_b, *i.e.* drawing on the discrepancies found in comparing its own deliberations with simulations of its reasoning in the theory OT_b.

Below we consider an abstract hierarchical architecture of an agent where the metalevel control is handled by the metalayer as it monitors and dynamically adapts the agent's strategies generated by the underlying object level theory

OL_a. A key feature in this model is that the agent is augmented by the introduction of a second object-layer OL_b. As the metalevel representation of the object theory is adapted for efficient metareasoning purposes, the deliberations in this second object layer OL_b serve the metalevel with computations on a full representation of the ground level interaction. Thus, during the game, the metatheory is allowed a more comprehensive, and yet cooperative, base for its reasoning.

4 A Legal Example

Let us turn to the legal example provided in [25] to illustrate our approach in a comparative setting:

> "The proponent Pro, John Prator, is the new owner of a flat in Rome. The previous owner sold the flat to John Prator, for the symbolic amount of 10 euros, through a notary deed. The previous owner had signed with Rex Roll, the opponent Opp, a rental agreement for the flat, which should apply for two more years. John has an interest in kicking out Rex, since he received an offer from another party, who intends to buy the flat paying 300 000 euros upon the condition that Rex leaves the flat. Rex Roll needs to stay two more years in Rome and refuses to leave the flat, as the rental fee was 500 euros per month, which is a convenient price (other flats in Rome have a rental fee of at least 600 euros per month). Hence, John sues Rex and asks the judge to impose to Rex to leave the flat. We assume that legislation states that any previously signed rental agreement still applies to the new owner when **(1)** a flat is donated, or **(2)** flat value is less than 150 000 euros. John's main argument **A** runs as follows: we do not have a donation since the flat was paid, thus the rental agreement is no longer valid and so Rex Roll has no right to stay. The opponent may present argument **C** that paying a symbolic amount of money indeed configures a case of donation, and John may reply with argument **E** that it is not the case because the property transfer was a sale formalized by a notary deed. Alternatively, Rex may present the argument **B** that the market value of the flat is of 120 000 euros and so the rental agreement is valid, whereas John may reply with **D** saying that he will pay within 10 days 210 000 euros to the previous owner, thus amending the transfer deed in order that it indisputably be a sale concerning a good of a value greater than 150000 euros. [...] So the problem is the following. According to the analysis of the case, which strategy to adopt?"

In order to illustrate our approach, we also add that John wants to secure a deal with Rex Roll, who indisputably is a very talented young person that could contribute to John Prator's already blossoming business.

We use this example for illustrating the construction of a theory of strategies adapted for a particular adversarial argumentation game. Here, the overall goal is to win the dispute.

However, when it comes to coordinating the strategy for achieving this goal, at the lowest level, winning the argumentation game, for Prator in the initial

state of the game, the optimal strategy is to put forward the argument most advantageous for winning the argumentation game, here the above argument D. However, even though winning the argumentation game is important, the victory should be accomplished by means of which the consequences do not take out the benefits of the purchase. Hence the strategy most advantageous for winning the argumentation game might be less optimal for achieving the goal. This is because the successful victory of the argumentation game by this means would commit Prator to accepting that the purchase price of the flat increases to at least 150 000 euros. As an additional overall preferred goal of gaining an upswing for John Prator's business by signing Rex Roll to his company, it might be the best strategy for fulfilling this goal to accept a deal that maintains the good relations with Rex. The underlying assumption is that the chances of signing Rex are lost by kicking Rex out of the flat [2]. As such, for the initial state of the game, the best strategy is to make a move advantageous for winning the argumentation game by which neither a good income for selling the flat nor signing Rex Roll for the business are inhibited.

5 A Metalogic Formalization of Optimum Strategy Computation

We define *Demo* to serve as a rule constructor for any object level formula A [3]. We establish the formula A at the object level T_i through admissibility by provability in T_{i+1} of the formula A. Here we draw on the metarules of the language in the metametatheory at level T_{i+2}, etc.

If the available knowledge, *i.e.* the common knowledge and the agent's own repository, at the object level T_i do not suffice to establish provability of the formula A, the reasoning is reflected upwards to its adjacent metalevel T_{i+1}.

$$Demo(T_i, S, A) \leftarrow$$
$$Demo(T_{i+1}, S, Demo(T_i, S, A)).$$

As the metalevel model of the object layer model's computational processes is represented in a form suitable for the deliberations at the metalevel, and hence, the knowledge about a particular strategy of a lower level may be of too little detail. Thus, occasionally the metalevel cannot resolve the acceptability of the strategy without utilizing a more expressive representation of the strategy. To force a formula (strategy) to be accepted in a theory T_i due to its established provability using the 'imperfect strategic knowledge' of the lower level T_{i-1} is unsound given the hierarchical system, as it might give rise to contradictions or unjustly constrain the strategies ascribed at the metalevel. Thus, if the object theory simulations of the metatheory reasoning need to be conducted in a full scale model, although not executed in the actual object layer, we define our downward reflection rule as:

[2] This assumption also excludes any substitute living arrangements for the benefit of Rex Roll

[3] The formula could represent an arbitrary strategy of the object level

$Demo(T_{i+1}, S, Demo(T_i, S, A)) \leftarrow$
$Demo(T_{i_b}, S, A).$

To be noted is that the downward reflection mechanism of our system is 'artificial', as the metatheory makes its judgment of acceptability on deliberations with simulations of its reasoning in the object theory T_{i_b}.

Moreover, we define *Demo* for AND-introduction for strategies at level T_i, establishing any formula $(A \wedge Q)$ at the level T_i through provability in the adjacent level T_{i+1} of the formula A and the formula Q given the metarules of the language in the metametatheory at level T_{i+2}, etc. We accommodate the precedence of the metatheory reasoning on T_i over the inferences on T_i carried out at the object level through requiring non-provability in the metatheory T_{i+1} of the non-provability of both formulas A and Q at the object level T_i.

$Demo(T_i, S, A \wedge Q) \leftarrow$
$Demo(T_i, S, A),$
$Demo(T_i, S, Q),$

We define *Demo* to serve as a rule constructor for an object level formula A. We establish the formula A at the object level T_i through admissibility by provability in T_{i+1}

$Demo(T_i, S, A) \leftarrow$
$Demo(T_i, S, A \subset Q),$
$Demo(T_i, S, Q).$

Whether an object level formula A is acceptable is given by the query:

$\leftarrow Demo(T_i, S, A)$

6 A Particular Argumentation Game - Flat War

Consider the legal example of [25] provided in Section 4. The example concerns a litigation of eviction of a tenant, Rex, however, with three goals of the proponent Prator. Accomplished by the strategy ς_1 we have the goal of securing the win of the litigation. Influencing this strategy is strategy ς_2 for achieving the goal of preserving the good deal of the flat purchase. At the topmost level the strategy ς_3, where the long term economical interest of signing Rex Roll to augment the already successful Prator-business, is the issue.

Here, the goal is to win the litigation in the optimum way given the three goals. Accommodating the first level goal the optimal strategy, given the goal, is to find the most solid argument for a win, *e.g.* that Prator will pay the additional 149990 euros to the previous owner, which would under all circumstances allow for the legal eviction of Rex Roll.

In the example, each level T_i consists of distinct subtheories $\varsigma_j \subseteq T_i$, $j \in 0..n$. We accommodate the precedence of subtheory inferencing such that

ς_l has precedence over the inferences on ς_{l-1}, $1 \leq l \leq n$. Hence, we add to the definition of *Demo* the following unit definite clauses. In the following clauses $\subset$, $and(\ldots,\ldots)$ are term level descriptions of implication and conjunction respectively, without any meaning to a theorem prover.

$$Demo(T_i, S, (D \in \varsigma_1 \subset \\ and(included(S, D), not(D \in \varsigma_1) \notin \varsigma_2))).$$

$$Demo(T_i, S, (E \in \varsigma_1 \subset \\ and(included(S, E), \\ and(not(E \in \varsigma_1) \notin \varsigma_2, not(D \in \varsigma_1))))).$$

$$Demo(T_i, S, (resting \in \varsigma_1 \subset \\ and(included(S, resting), \\ and(not(resting \in \varsigma_1) \notin \varsigma_2, \\ and(not(D \in \varsigma_1), not(E \in \varsigma_1))))).$$

Here the binary predicate *included* is an auxiliary predicate that checks the membership of the second argument in the shared knowledgebase or the agent repository in the given state S.

For the goal of winning without severe consequences for the revenue of the purchase that strategy might be less optimal. In order to achieve the goal it is required that no arguments committing Prator to pay additionally for the purchase are used to win the litigation. Therefore, given this preference Prator entertains the following strategy:

$$Demo(T_i, S, (E \in \varsigma_1) \in \varsigma_2 \subset \\ and(included(S, cannot\_secure\_revenue), \\ (not((E \in \varsigma_1) \notin \varsigma_3)))).$$

$$Demo(T_{i+1}, S, \\ (Demo(T_i, S, \\ included(S, cannot\_secure\_revenue) \subset \\ and((Demo(T_i, S, \\ (included(S, cannot\_secure\_revenue) \subset \\ included(S, committed\_to\_pay\_higher\_price)))), \\ Demo(T_i, S, \\ included(S, committed\_to\_pay\_higher\_price)))))).$$

Finally, at the topmost-level, for the long term goal of Prator, it might be the best strategy to withdraw from that particular litigation. This is if the chances of either retaining the revenue and signing Rex Roll is dependent on the withdrawal of the litigation.

$$Demo(T_i, S, ((resting \in \varsigma_x) \in \varsigma_3 \subset \\ included(S, risk\_not\_signing\_rex))).$$

$$Demo(T_{i+1}, S,$$
$$(Demo(T_i, S, included(S, risk\_not\_signing\_rex) \subset$$
$$and((Demo(T_i, S,$$
$$(included(S, risk\_not\_signing\_rex) \subset$$
$$included(S, leave\_rome\_in\_anger\_rex)))),$$
$$Demo(T_i, S,$$
$$included(S, leave\_rome\_in\_anger\_rex))))))).$$

In this particular dispute from $S_{T_{i+1}}$ follows:

$$claim\_in\_court\_higher\_price \rightarrow$$
$$committed\_to\_pay\_higher\_price.$$

$$evicted\_rex \rightarrow leave\_rome\_in\_anger\_rex.$$

Hence, the current strategy for the argumentation game is to play the speech act *resting* as the best strategy, according to the below:

$$Demo(T_i, S, (resting \in \varsigma_1) \subset$$
$$and(included(S, resting),$$
$$and((not(D \in \varsigma_1) \notin \varsigma_2), (not(E \in \varsigma_1) \notin \varsigma_2)))).$$

Due to the current:

$$Demo(T_i, S, ((resting \in \varsigma_x) \in \varsigma_3 \subset$$
$$included(S, risk\_not\_signing\_rex))).$$

During the game, additional information becomes available:

$$Demo(T_i, S,$$
$$included(S, no\_risk\_not\_signing\_rex)).$$

$$Demo(T_i, S, ((resting \in \varsigma_1) \notin \varsigma_x \subset$$
$$Demo(T_{i_b}, S, included(S, Rotolo\_already\_signed\_rex)))).$$

7 Related Work

In this paper we approach strategies for adversarial argumentation, where two agents debate over one topic. As our point of departure, work on teleological planning and argumentation are of interest. [26] are using strategic games to measure the probabilities of a claimed conclusion being defeasibly provable. The approach builds on defeasible logic and probability calculus. The game is played between a proponent, *Prop*, and an opponent, *Opp*, that take turns in presenting their arguments. The aim for the proponent is to maximize the probability that the claimed conclusion will be defeasibly provable in the game, while the opponent is to minimize this probability. In our approach, the preferences are set by the game specific circumstances, and the dynamical progression of the

game, rather than the general probabilities generated from strong assumptions on the available knowledge of the other agent. Our approach draws on the stratified representation of legal knowledge of [16], hence the logical computation of the strategy is accomplished by the metalogical formalization of provability and reflection. Our choice of settling the preference of arguments is inspired by [10]. They present Zeno as a mediating system, intended as a tool for conceding differences in opinion between different agents. The dynamical model of argumentation presented in [10] focuses on the type of speech acts possible to put forward and their locution, as a protocol of the mediation. As either agent could put forward a speech act labeled as pro or con some issue, as either an argument or a counterargument of the issue. The aim is to resolve the open issues. The model is recursively defined by drawing on dialectical graphs. In our approach, the preferences are set by the game specific circumstances for adversarial argumentation games. Advocating the teleological tradition for the construction of legal theory as a coherent whole, Hage [14] makes reference to the works of Dworkin, MacCormick, Alezy and Pecenik as based on the *Interessenjurisprudenz* of Marx and Larentz. [14] provides the formulation of standards for evaluation of competing sets of rules and cases given the principles they are to instantiate. These standards are expressing part of Dworkin's model of principles in the framework of Reason-based Logic [13], aiming to apply logical methods for the evaluation of legal regulations given their goals. Hage distinguishes the goals that are promoted by a regulation from those that are detracted from the regulation and those goals that remain unaffected. By weighing these subsets against each other the acceptability of a regulation can be determined. As illustration, the Lebach-case [1] is used. Here the judge has to decide in this dispute, while balancing the goals of respect for privacy of others and the goal of protecting free press. Closely related to the approaches of [28] and [5], a model of Dworkin's principles is set in the Reason-Based Logic, and subsequently used for the comparison of regulations given the goals they are to realize. An extension of Dworkin's theory of legal theory construction is presented in [15]. Hage and Sartor [15] present an approach on how to adapt legal theory construction as tinkering with goals, their relative importance, and with cases to construct a coherent theory of law. Our approach draws on the stratified representation of legal knowledge of [16], hence the logical computation of the optimum strategy is accomplished by the metalogical formalization of provability and reflection in a dynamical game setting. A related area of interest to our research focus comes from the acknowledgement that decisions normally are influenced by the interests of the adjudicator in the form of a judge, jury or audience [6]. This aspect is acknowledged by [4] and is stated to reflect the social climate of the social setting of the dispute settlement [2]. Widening the perspective to more general domains of computational argumentation, we find work on teleological planning for agents, drawing on value considerations as an alternative characterization of the preferences influencing the argumentation [4], but also for hierarchical argumentation in which argumentation is used for resolving conflicts between arguments at adjacent levels [19], [20].

8 Conclusion

The ideal strategy for achieving an agent's superior goal could be characterized as the coordination of the acceptable and meaningful strategies for accomplishing intermediate goals of the argumentation game, according to some preference. In this paper we have shown how metalogic stratification and interpretation of formal theories may be used as a mechanism for goal-oriented deliberation that does not presuppose available a probability distribution to compute an adapted theory of interpretation of an ideal strategy. It is achieved by mapping the strategies as meaningful and acceptable given the intermediate goals, the progression of the game and the available arguments.

The common way of comparing strategies for achieving goals is by a expected utility function. In our approach the preferences are set by the game specific circumstances, rather than the general probabilities generated from strong assumptions on the available knowledge of the adversary. We do not assume that the preferences of acceptance of the agents' strategies are known *a priori*. Besides from the concerns discussed in [25], it may reduce an argumentation game to a blackboard computation. For these reasons, an *a priori* computation of the preferred expected utility cannot be readily accounted for in most disputes. This is mainly due to the unavailability of the future moves to be played by the adversary, cf. our key objectives of privacy of the agent's knowledge, as discussed in Section 2. For the purpose of winning, a rational agent choose the goal that best will enable overall success, by taking into account its available means [27]. In an argumentation game this situation is dependent on the dialectical process, i.e. the available knowledge that could be derived from the common knowledge and the private knowledge put forward by the agents during the dispute. However, in a particular litigation, such aspects cannot be specified and enacted without taking into account the dynamical progression of the argumentation game. We are to pursue our research along these lines.

References

1. R. Alexy and A. Peczenik. The concept of coherence and its significance for discursive rationality. *Ratio Juris*, 1990.
2. K. Atkinson and T. J. M. Bench-Capon. Argumentation and standards of proof. In *ICAIL*, pp. 107–116, 2007.
3. T. Bench-Capon and M. Sergot. Towards a rule-based representation of open texture in law. *In Charles Walter, editor, Computing Power and Legal Language*, pp. 39–60, 1988.
4. T. J. M. Bench-Capon. Persuasion in practical argument using value based argumentation frameworks. *Logic and Computation*, 13(3):429–448, 2003.
5. T. J. M. Bench-Capon and G. Sartor. Theory based explanation of case law domains. In *ICAIL '01: Proceedings of the 8th international conference on Artificial intelligence and law*, pp. 12–21, New York, NY, USA, 2001. ACM.
6. D. H. Berman and C. D. Hafner. Representing teleological structure in case-based legal reasoning: the missing link. *Proceedings of the Fourth International Conference on Artificial Intelligence and Law*, pp. 50–59, 1993.

7. M. Bratman. *Intention Plans and Practical Reason*. Harward University Press, 1987.
8. S. Constatini. Meta-reasoning: a survey. In *Computational Logic: Logic Programming and Beyond Essays in Honour of Robert A. Kowalski (LNAI Volumes 2408*, pp. 253–288. Springer-Verlag, 2002.
9. J. Eriksson Lundström, A. Hamfelt, and J. F. Nilsson. Legal rules and argumentation in a metalogic framework. In *JURIX 2007 Leiden*, pp. 39–48. ACM Press, 2007.
10. T. Gordon and N. Karacapilidis. The zeno argumentation framework. In *Proceedings of ICAIL-97*, 1997.
11. G. Governatori, G. Antoniou, D. Billington, and M. J. Maher. Argumentation semantics for defeasible logics. *Journal of Logic and Computation*, 14(5), 2004.
12. S. A. Guerlain, P. J. Smith, J. H. Obradovich, S. Rudmann, P. Strohm, J. W. Smith, J. Svirbely, and L. Sachs. Interactive critiquing as a form of decision support - an empirical evaluation. *Human Factors*, 41(1):72–89, 1999.
13. J. Hage. A model of legal reasoning and a logic to match. *Artificial Intelligence and Law*, 4(3-4):199–273, 1996.
14. J. Hage. Goal-based thory evaluation. In *Proceedings of the Thirteenth Jurix Conference*, pp. 59–72. IOS Press Amsterdam, 2000.
15. J. Hage and G. Sartor. Legal theory construction. *Associations*, 2003.
16. A. Hamfelt. Formalizing multiple interpretation of legal knowledge. *Journal of Artificial Intelligence and Law*, 3(4):221–265, 1995-1996.
17. S. C. Kleene. *Mathematical Logic*. John Wiley. Dover reprint, 1967/2002.
18. R. A. Kowalski. Problems and promises of computational logic. In *eds. J.W. Lloyd Computational Logic*, pp. 1–36. Springer Verlag Berlin, 1990.
19. S. Modgil. Value based argumentation in hierarchical argumentation frameworks. In *1st International Conference on Computational Models of Argument (COMMA 06)*, pp. 297–308, 2006.
20. S. Modgil and T. J. M. Bench-Capon. Metalevel argumentation. *Journal of Logic and Computation (JLC)*, pp. 1–45, September 2010.
21. H. Prakken. Modelling defeasibility in law:logic or procedure? *Fundamenta informaticae*, pp. 253–271, 2001.
22. H. Prakken and G. Sartor. Presumptions and burdens of proof. In *Jurix 2006*, pp. 21–30. IOS Press, 1996.
23. H. Prakken and G. Sartor. Rules about rules: Assessing conflicting arguments in legal reasoning. *Artificial Intelligence and Law*, pp. 331–368, 1996.
24. H. Prakken and G. Sartor. Formalising arguments about the burden of persuasion. In *ICAIL 2007*, pp. 97–106. ACM Press, 2007.
25. R. Riveret, H. Prakken, A. Rotolo, and G. Sartor. Heuristics in argumentation - a game-theoretical investigation. *COMMA 2008*, pp. 324–335, 2008.
26. B. Roth, R. Riveret, A. Rotolo, and G. Governatori. Strategic argumentation: A game theoretical investigation. In *Proc. ICAIL 2007*, pp. 81–90. ACM Press, 2007.
27. S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition, 2008.
28. G. Sartor. Teleological arguments and theory-based dialectics. *Artificial Intelligence and Law*, 10(1-3):95112, 2002.
29. S. Thakur, G. Governatori, V. Nair, and J. Eriksson Lundström. Dialogue games in defeasible logic. In *the Proceedings of the Twentieth Australian Joint Conference on Artificial Intelligence, the Gold Coast of Australia*, December 2007.
30. D. N. Walton. *The New Dialectic: Conversational Contexts of Argument*. Buffalo University of Toronto Press, 1998.

A First Step towards Argumentation Dialogues for Discovery

Xiuyi Fan and Francesca Toni

Imperial College London, London, SW7 2AZ, UK,
{xf309,ft}@imperial.ac.uk

Abstract. We present a formal model for two-agent discovery dialogues. The model allows agents to collectively discover a realization for a shared goal, using argumentation dialogues to exchange information. This information is in the form of rules, assumptions, and contraries of assumptions as in Assumption-based Argumentation (ABA). With dialogues, agents jointly build arguments and construct shared ABA frameworks. We define successful discovery dialogues as those giving admissible arguments that realize the shared goal. The main novelty of this paper is the modelling of the *buttom-up* relation between utterances. This new relation helps building "higher level" arguments from existing "lower level" supports, which we deem essential for discovery.

1 Introduction

Argumentation dialogues have been studied by a number of researchers [1, 8, 12]. Walton & Krabbe [14] introduce a dialogue taxonomy that categorizes dialogues into six types: *persuasion, inquiry, information seeking, negotiation, deliberation*, and *eristic*. McBurney and Parsons [8] introduce *discovery* as an additional type of dialogue, different from other types in that "[discovery dialogues] discover something not previously known; the question whose truth is to be ascertained may only emerge in the course of the dialogue." In this paper, we present a two-agent dialogue framework that supports a special type of discovery dialogues.

Most previous work on argumentation dialogues, e.g. [1, 11], define dialogue models as dialogues starting from a known proposition; through dialogues, the acceptability of this proposition is examined. However, in discovery dialogues, there may be no such known proposition to start with. Instead, the dialogue participants face an open problem; they must decide on an abstract description of the goal of the dialogue and proceed by putting forward information that may contribute to identify the proposition and determine its acceptability.

In this paper, we focus on a particular type of discovery dialogue, in which the two participating agents start from the same abstract description of the proposition. We call this abstract description the *goal*. None of the two agents have sufficient information to produce an acceptable concrete realization of this goal. The agents' task is then to discover an acceptable goal realization.

In this work, the two agents communicate to each other using Assumption-Based Argumentation (ABA) [3] frameworks. ABA is a general-purpose, widely

applicable form of argumentation where arguments are built from *rules* and supported by *assumptions*, and attacks against arguments are directed at the assumptions supporting the arguments, and are provided by arguments for the *contrary* of assumptions. ABA is applicable in several settings, e.g. see [3].

During the dialogue, agents communicate with ABA by putting forward rules, assumptions, and contraries of assumptions as their utterances. In order to perform the joint discovery, the dialogue starts by one agent putting forward a sentence. Then the two agents can either expand this sentence in a *top-down* manner to explore its supports and attacks or *bottom-up* inference to identify any "higher level" arguments that are supported by it. Utterances jointly form the ABA framework *drawn* from the dialogue. A discovery dialogue is successful if it produces a goal realization. We justify the soundness of our approach by showing that the produced goal realization is admissible with respect to the ABA framework drawn from the dialogue.

This work extends [6], which present a conflict resolution framework in a two-agent system. The two agents in [6] share the same goal but each agent has its own realization to start with. Hence, in [6] the two agents examine each realization solely in (what we call) the *top-down* fashion to determine its admissibility. Since [6] is an application of the dialogue framework in [5], this paper also extends [5]. The main novelty of our work here is the recognition of the *bottom-up* relation, which we believe is essential for modelling discovery dialogues.

2 Background

An ABA framework [3, 4] is a tuple $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \mathcal{C} \rangle$ where

- $\langle \mathcal{L}, \mathcal{R} \rangle$ is a deductive system, with $\mathcal{L}$ the *language* and $\mathcal{R}$ a set of *rules* of the form $s_0 \leftarrow s_1, \ldots, s_m (m \geq 0)$;
- $\mathcal{A} \subseteq \mathcal{L}$ is a (non-empty) set, referred to as *assumptions*;
- $\mathcal{C}$ is a total mapping from $\mathcal{A}$ into $\mathcal{L}$, where $\mathcal{C}(\alpha)$ is the *contrary* of $\alpha \in \mathcal{A}$.

Given a rule $s_0 \leftarrow s_1, \ldots, s_m$, s_0 is referred to as the *head* and $s_1, \ldots, s_m$ as the *body*. We use the following notation: $Head(s_0 \leftarrow s_1, \ldots, s_m) = s_0$ and $Body(s_0 \leftarrow s_1, \ldots, s_m) = \{s_1, \ldots, s_m\}$. As in [4], we enforce that ABA frameworks are *flat*, namely assumptions do not occur in the head of rules. Moreover, without loss of generality, we enforce that no two assumptions may have the same contrary.

In ABA, *arguments* are deductions of claims using rules and supported by assumptions, and *attacks* are directed at assumptions. Informally, following [3]:

- *an argument for (the claim)* $c \in \mathcal{L}$ *supported by* $S \subseteq \mathcal{A}$ ($S \vdash c$ in short) is a (finite) tree with nodes labelled by sentences in $\mathcal{L}$ or by the symbol τ[1], such that the root is labelled by c, leaves are either τ or assumptions in S, and non-leaves s have as many children as elements in the body of a rule with head s, in a one-to-one correspondence with the elements of this body.
- *an argument* $S_1 \vdash c_1$ *attacks an argument* $S_2 \vdash c_2$ iff $c_1 = \mathcal{C}(\alpha)$ for $\alpha \in S_2$.

[1] As in [3], $\tau \notin \mathcal{L}$ represents "true". It is used to represent the empty body of a rule.

Attacks between arguments correspond in ABA to attacks between sets of assumptions, where *a set of assumptions A attacks a set of assumptions B* iff an argument supported by $A' \subseteq A$ attacks an argument supported by $B' \subseteq B$.

With argument and attack defined, standard argumentation semantics can be applied in ABA [3]. We focus on the admissibility semantics:

- *a set of assumptions is admissible* (w.r.t. $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \mathcal{C} \rangle$) iff it does not attack itself and it attacks all $A \subseteq \mathcal{A}$ that attack it;
- *an argument $S \vdash c$ belongs to an admissible extension supported by $\Delta \subseteq \mathcal{A}$* (w.r.t. $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \mathcal{C} \rangle$) iff $S \subseteq \Delta$ and Δ is admissible;
- *a sentence is admissible* iff it is the claim of an argument that belongs to an admissible extension supported by some $\Delta \subseteq \mathcal{A}$.

Our main result will be proven using the abstract dispute trees of [4], with an *abstract dispute tree* for an argument γ a (possibly infinite) tree $\mathcal{T}^a$ such that:

1. every node of $\mathcal{T}^a$ is labelled by an argument and is assigned the status of either a *proponent* (P) node or an *opponent* (O) node, but not both;
2. the root of $\mathcal{T}^a$ is a P node labelled by γ;
3. for every P node n labelled by an argument b, and for every argument c that attacks b, there exists a child of n, which is an O node labelled by c;
4. for every O node n labelled by an argument b, there exists exactly one child of n which is a P node labelled by an argument which attacks some assumption α in the set supporting b. α is said to be the *culprit* in b;
5. there are no other nodes in $\mathcal{T}^a$ except those given by 1-4 above.

The set of all assumptions in (the support of arguments of) the proponent nodes in $\mathcal{T}^a$ is called the *defence set* of $\mathcal{T}^a$. An abstract dispute tree is *admissible* iff no culprit in the argument of an opponent node belongs to the defence set of $\mathcal{T}^a$. The defence set of an admissible abstract dispute tree $\mathcal{T}^a$ for an argument γ is admissible (Theorem 5.1 in [4]), and thus γ belongs to an admissible extension and the sentence supported by γ is admissible.

3 Motivating Example

Two agents, Jenny and Amy, are planning a film night. They would like to jointly decide on a movie. Jenny wants to pick a fun movie. She finds action movies fun. Jenny also worries about going home late so she prefers a movie that finishes by 10 o'clock. Amy knows that *Terminator* is screening and is an action movie. She also knows that *Terminator* finishes by 10 o'clock. Amy does not have any preference in selecting a movie. In order to reach agreement, the two agents may conduct a dialogue as follows[2].

[2] A variant of this dialogue example is in [6]. There, however, the dialogue starts with an initial concrete realization of the goal watchMovie(X).

Jenny: Let's find a movie to watch.
Amy: Sure, I know *Terminator* is an action movie.
Jenny: That's interesting. I think action movies are pretty fun.
Amy: We can watch *Terminator*, as long as it has the right screening time.
Jenny: Agreed. I think *Terminator* starts at the right time.
Amy: Are you sure it won't be too late?
Jenny: Why?
Amy: I don't know. I am just afraid so.
Jenny: It won't be too late if it finishes by 10 o'clock.
Amy: I see. Indeed *Terminator* finishes by 10 o'clock.
Jenny: OK.
Amy: OK.

Jenny starts the dialogue by putting forward the goal of determining some movie to watch. Then Amy supplies one possibly relevant fact, that *Terminator* is an action movie. This is just a guess, in the sense that the agent does not know whether a goal realization can be found by exploring information related to *Terminator*. From this utterance, agents reason buttom-up until Amy's second utterance. Then they start top-down. From the inital guess, the dialogue constructs arguments both for and against watching *Terminator*. After examining the arguments, the agents decide that *Terminator* is a good movie to watch.

4 Goals and Discovery Dialogues

We define *goal* and *goal realization* w.r.t. a given ABA framework, as in [6].

Definition 1. *A* goal *(w.r.t. $\mathcal{L}$) is of the form $\exists\ X\ \mathcal{G}$ such that*

- *X is a tuple of variables;*
- *there exists $\sigma = \{X/t\}$ for t a tuple of terms such that $\mathcal{G}\sigma \in \mathcal{L}$* [3]*.*

A (goal) realization *(w.r.t. an ABA framework $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \mathcal{C} \rangle$) is $\mathcal{G}\sigma \in \mathcal{L}$ such that $\sigma = \{X/t\}$ and $\mathcal{G}\sigma$ is admissible (w.r.t. $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \mathcal{C} \rangle$).*

In our example, watchMovie(X) is the goal and watchMovie(Ter) is a realization (for $\{X/Ter\}$, where Ter stands for $Terminator$) w.r.t. the ABA framework:

R_δ:	watchMovie(X) $\leftarrow$ fun(X), goodScreenTime(X)
	fun(X) $\leftarrow$ actionMovie(X)
	actionMovie(Ter)
	finishByTen(Ter)
A_δ:	goodScreenTime(X)
	late(X)
C_δ:	$\mathcal{C}_\delta$(goodScreenTime(X)) = late(X)
	$\mathcal{C}_\delta$(late(X)) = finishByTen(X)

[3] $\mathcal{G}\sigma$ stands for $\mathcal{G}$ where all occurrences of (elements of) X are replaced by (the corresponding elements of) t. We often leave the existential quantifier of goals implicit.

In the remainder we consider two generic agents a_1 and a_2. In our example these are Jenny (J) and Amy (A). We assume that a_1, a_2 share a language $\mathcal{L}$[4].

We define dialogues as sequences of utterances between a_1 and a_2. Formally:

Definition 2. *An* utterance *from agent a_i to agent a_j ($i, j \in \{1, 2\}$, $i \neq j$) w.r.t. $\mathcal{L}$ is a tuple $\langle a_i, a_j, Target, C, ID, R\rangle$, where:*

- *C (the* content*) is of one of the following forms: (1) $goal(\mathcal{G})$ for some $\mathcal{G}$ such that $\exists X\mathcal{G}$ is a goal; (2) $rl(s_0 \leftarrow s_1, \ldots, s_m)$ for some $s_0, \ldots, s_m \in \mathcal{L}$ (a* rule*), and if $m > 0$ then $s_i \neq s_j$ for $0 \leq i, j \leq m, i \neq j$; (3) $asm(a)$ for some $a \in \mathcal{L}$ (an* assumption*); (4) $ctr(a, s)$ for some $a, s \in \mathcal{L}$ (a* contrary*); (5) a* pass *sentence π, such that $\pi \notin \mathcal{L}$.*
- *$ID \in \mathbb{N} \cup \{0\}$ (the* identifier*).*
- *$Target \in \mathbb{N} \cup \{0\}$ (the* target*); $Target \leq ID$.*
- *R is either td* (top-down)*, bu* (bottom-up) *or nr* (not-related)*.*

We refer to an utterance with content π as a pass-utterance*, and to an utterance with content other than π and $goal(\_)$ as* regular-utterance*.*

Intuitively, a pass indicates that the agent does not have or want to contribute information at that point. This definition is based on Definition 1 of [5], but (1) this definition adds the new *related* field (R) to indicate different utterance relations (td, bu or nr); and (2) there is no "claim" used in this definition.

Definition 3. *For two utterances $u_i \neq u_j$, u_j is* top-down related to *u_i iff $u_i = \langle \_, \_, \_, C_i, ID, \_\rangle$, $u_j = \langle \_, \_, ID, C_j, \_, td\rangle$*[5]*, and one of the following holds:*

1. *$C_j = rl(\rho_j)$, $Head(\rho_j) = h$ and either $C_i = rl(\rho_i)$ with $h \in Body(\rho_i)$, or $C_i = ctr(\_, h)$;*
2. *$C_j = asm(a)$ and either $C_i = rl(\rho)$ with $a \in Body(\rho)$, or $C_i = ctr(\_, a)$;*
3. *$C_j = ctr(a, \_)$ and $C_i = asm(a)$.*

This definition is based on Definition 3 of [5], but without considering claim utterances (as there are not allowed here). Intuitively, an utterance is top-down related to another if its target is the identifier of the latter and it contributes to expanding an argument (cases 1), identifies an assumption in the support of an argument (cases 2) or starts the construction of a counter-argument (case 3).

Definition 4. *For two utterances $u_i \neq u_j$, u_j is* bottom-up related to *u_i iff $u_i = \langle \_, \_, \_, C_i, ID, \_\rangle$ and $u_j = \langle \_, \_, ID, C_j, \_, bu\rangle$, and one of the following holds:*

1. *$C_i = rl(\rho_i)$, $C_j = rl(\rho_j)$, and $Head(\rho_i) \in Body(\rho_j)$;*
2. *$C_i = asm(a)$, $C_j = rl(\rho)$, and $a \in Body(\rho)$.*

[4] Throughout all ABA frameworks in the paper, we leave the language component implicit.

[5] Throughout, _ stands for an an anonymous variable.

Intuitively, an utterance is buttom-up related to another if its target is the identifier of the latter and it forms a "higher level" argument supported by its target. We say that an utterance u_j is *related to* u_i if u_j is either top-down or bottom-up related to u_i. Also, no pass-utterance can be related to a regular-utterance and no utterance can be related to a pass-utterance.

Definition 5. *A* dialogue $D^{a_i}_{a_j}(\mathcal{G})$ *(*between a_i and a_j, for goal $\mathcal{G}$ w.r.t. $\mathcal{L}$*),* $i, j \in \{1, 2\}$*,* $i \neq j$*, is a finite sequence* $\langle u_0, \ldots, u_n \rangle$*,* $n \geq 0$*, where each* u_l*,* $l = 0, \ldots, n$*, is an utterance from* a_i *or* a_j *(w.r.t.*$\mathcal{L}$*),* u_0 *is an utterance from* a_i*, and:*

1. *the content of* u_l *is* $goal(\mathcal{G})$ *iff* $l = 0$*;*
2. *the content of* u_1 *is either* $rl(\_)$ *or* $asm(\_)$*;*
3. u_0 *and* u_1 *are of the form* $\langle \_, \_, 0, \_, \_, nr \rangle$*;*
4. *the target of pass-utterances is* 0*;*
5. *each regular-utterance* u_l*,* $l > 1$*, is related to its target utterance;*
6. *no two consecutive utterances are pass-utterances, other than possibly the last two utterances,* u_{n-1} *and* u_n*;*
7. *the identifier of* u_i *is* i*.*

This definition requires dialogues to start with a goal (the first utterance). The second utterance must be a rule or an assumption. Agents make this utterance with a "wild guess" in the hope that a goal realization can be found by exploring around this guess. All subsequent regular-utterance must be related to some earlier utterance in the dialogue. This definition is a variant of Definition 3 in [5] with dialogues starting with a goal rather than a claim and the second utterance being not related to the first. An example dialogue is given below.

$\langle J, A, 0, goal(\text{watchMovie(X)}), 0, nr \rangle$
$\langle A, J, 0, rl(\text{actionMovie}(Ter) \leftarrow), 1, nr \rangle$
$\langle J, A, 1, rl(\text{fun}(Ter) \leftarrow \text{actionMovie}(Ter)), 2, bu \rangle$
$\langle A, J, 2, rl(\text{watchMovie}(Ter) \leftarrow \text{fun}(Ter), \text{goodScreenTime}(Ter)), 3, bu \rangle$
$\langle J, A, 3, asm(\text{goodScreenTime}(Ter)), 4, td \rangle$
$\langle A, J, 4, ctr(\text{goodScreenTime}(Ter), \text{late}(Ter)), 5, td \rangle$
$\langle J, A, 0, \pi, 6, nr \rangle$
$\langle A, J, 5, asm(\text{late}(Ter)), 7, td \rangle$
$\langle J, A, 7, ctr(\text{late}(Ter), \text{finishByTen}(Ter)), 8, td \rangle$
$\langle A, J, 8, rl(\text{finishByTen}(Ter)), 9, td \rangle$
$\langle J, A, 0, \pi, 10, nr \rangle$
$\langle A, J, 0, \pi, 11, nr \rangle$

An informal reading of this dialogue is given in Section 3. Below, $\mathcal{U}$ and $\mathcal{D}$ stand for the sets, respectively, of all utterances as in Definition 2 and of all dialogues as in Definition 5.

By means of dialogues, agents exchange information and build a shared framework, as defined in [5]:

Definition 6. *[5] The* framework drawn from a dialogue $\delta = \langle u_0, \ldots, u_n \rangle$ *is* $\langle \mathcal{L}, \mathcal{R}_\delta, \mathcal{A}_\delta, \mathcal{C}_\delta \rangle$ *where*

- $\mathcal{R}_\delta = \{r | rl(\rho)$ *is the content of some* u_i *in* $\delta\}$;
- $\mathcal{A}_\delta = \{a | asm(a)$ *is the content of some* u_i *in* $\delta\}$;
- $\mathcal{C}_\delta$ *is a mapping such that, for any* $a \in \mathcal{A}_\delta$, $\mathcal{C}_\delta(a) = c$ *such that* $ctr(a, c)$ *is the content of some* u_i *in* δ, *if one exists, and is undefined otherwise.*

The framework drawn from the example dialogue is $F_\delta = \langle \mathcal{L}, \mathcal{R}_\delta, \mathcal{A}_\delta, \mathcal{C}_\delta \rangle$ shown earlier.

Note that F_δ in this example is a flat ABA framework, but, in general as discussed in [5], the framework drawn from a dialogue may not be an ABA framework, since $\mathcal{C}_\delta$ may not be total. We follow [5] and impose that flat ABA frameworks by using suitable *legal-move functions* and *outcome functions.*

Definition 7. *A* legal-move function *is a mapping* $\lambda : \mathcal{D} \mapsto \mathcal{U}$ *such that, given* $\delta = \langle u_0, \ldots, u_n \rangle$ *between* a_i, a_j, $i, j \in \{1, 2\}, i \neq j$, $\lambda(\delta) = \langle a_i, a_j, t, C, id, r \rangle$ *and*

1. $\delta \circ \lambda(\delta) = \langle u_0, \ldots, u_n, \lambda(\delta) \rangle$ *is a dialogue;*
2. *if* $C \neq \pi$, *then there exists no* $k, 1 \leq k \leq n$, *such that* $u_k = \langle \_, \_, t, C, \_, \_ \rangle$.
3. *if* $C = ctr(a, c)$ *then there exists no* $u_k = \langle \_, \_, \_, ctr(a, c'), k, \_ \rangle$, *for* $1 \leq k \leq n$, *and* $c' \neq c$.

Given $\delta = \langle u_0, \ldots, u_n \rangle$, *if* $\lambda(\langle u_0, \ldots, u_m \rangle) = u_{m+1}$ *for all* m $(1 \leq m < n)$, *we say that* δ *is constructed with* λ. *We use* Λ *to denote the set of all legal-move functions.*

This definition is similar to Definition 5 in [5] but has an extra field r to indicate how two utterances are related. This definition imposes that a sequence of utterances form a dialogue (condition 1); there is no repeated utterance to the same target (condition 2); and that assumptions have a single contrary (condition 3). However, the definition of legal-move function does not impose any "mentalistic" requirement on agents, such as that they utter information they hold true within their ABA framework.

Definition 8. *A* flat legal-move function *is such that if, for* $n \geq 0$, $\lambda(\langle u_0, \ldots, u_n \rangle) = \langle a_x, a_y, t, C, n + 1, r \rangle$, *then*

- $C = asm(a)$ *only if* $\nexists\ u_i$ *for* $1 \leq i \leq n$ *with* $C = rl(\rho)$ *and* $Head(\rho) = a$;
- $C = rl(\rho)$ *only if* $\nexists\ u_i$ *for* $1 \leq i \leq n$ *with* $C = asm(a)$ *and* $Head(\rho) = a$.

This definition is similar to Definition 6 with the extra r field. Trivially, the framework drawn from a dialogue generated by a flat legal-move function, if an ABA framework, is flat, in the sense of [2].

Definition 9. *A (flat)* one-way expansion legal-move function *is such that, if for* $n \geq 0$, $\lambda(\langle u_0, \ldots, u_n \rangle) = \langle a_x, a_y, t, C, n + 1, r \rangle$, *then* $C = rl(\rho)$ *only if* $\nexists\ u_i = \langle \_, \_, t, rl(\rho'), i, \_ \rangle$ *for* $1 \leq i \leq n$ *and* $Head(\rho) = Head(\rho')$.

This definition imposes that, in a dialogue there is only one unique way of expanding a rule for the same target. Notice that this definition enforces uniqueness

solely w.r.t. expanding the same target. Hence it allows that two rules $p \leftarrow q$, $p \leftarrow q'$ co-exist in a dialogue, as long as they are uttered to different targets.

Legal-move functions provide some guidance as to what agents should utter during dialogues. In order to guarantee that the contrary mapping in the framework drawn from a dialogue is total, we use the notion of *outcome function*, checking specific properties in a generated dialogue:

Definition 10. *An* outcome function *is a mapping* $\omega : \mathcal{D} \times \Lambda \mapsto \{true, false\}$. *The* ABA outcome function*,* ω_{ABA}*, is such that given a dialogue* δ *and a legal-move function* λ*,* $\omega_{ABA}(\delta, \lambda) = true$ *iff* δ *is constructed with* λ *and the framework* $\langle \mathcal{L}, \mathcal{R}_\delta, \mathcal{A}_\delta, \mathcal{C}_\delta \rangle$ *drawn from* δ *is such that for all* $\alpha \in \mathcal{A}_\delta, \mathcal{C}_\delta(\alpha)$ *is not undefined.*[6]

We focus on dialogues where each utterance results from applying a one-way expansion legal-move function to the dialogue prior to that utterance, and for which ω_{ABA} is true. We refer to these dialogues as *ABA dialogues*.

We use *debate trees* to refine the notions of legal-move and outcome functions to guarantee that dialogues compute arguments in admissible extensions.

5 Debate trees

Nodes of debate trees are labelled either *proponent* or *opponent* as in the abstruct dispute trees in [4]. However, differently from [4], in debate trees each node (1) contains one sentence, (2) is *tagged* as either *unmarked (um), marked-rule (mr)* or *marked-assumption (ma)*, and (3) has an ID to identify its corresponding utterance in a dialogue.

When constructing a debate tree from a dialogue, we use a subset of utterances presented in the dialogue. This extraction ignores the goal- and pass-utterances, i.e. a debate tree is extracted from the *goal-/π-pruned sequence obtained from a dialogue*, consisting of all regular-utterances. Note that, since no regular-utterance has a pass-utterance as its target (see definition 5), the target of every utterance in a goal-/π-pruned sequence is guaranteed to be in this sequence. Furthermore, by Definition 5, for all utterances $u = \langle \_, \_, \_, \_, i, R \rangle$ in a goal-/π-pruned sequence, if $i > 1$, then R is either td or bu.

The sentence of each node in a debate tree represents an argument's claim or an element of its support. A node is tagged *unmarked* if its sentence is only mentioned in the body of a rule or contrary of an assumption, but without any further examination, *marked-rule* if it is the head of a rule, and *marked-assumption* if it has been explicitly declared as an assumption. Formally:

Definition 11. *A* debate *drawn from a dialogue* $\delta = \langle u_0, \ldots, u_n \rangle$ *(*$n \geq 0$*) is a graph* $\mathcal{T}(\delta)$ *whose nodes are tuples* $(S, F : \mathrm{L}[U])$ *where*

- S *(the sentence) is a sentence in* $\mathcal{L}$*,*

[6] Definition 10,18, and 20 are variations of Definition 7, 13, and 15, respectively, in [5] with the extra parameter legal-move function Λ to enforce that a dialogue is properly constructed.

– *F (the tag) is either um* (unmarked), *mr* (marked-rule) *or ma* (marked-assumption),
– *L (the label) is either P* (proponent) *or O* (opponent),
– $U \in \mathbb{N}$ *(the ID).*

$\mathcal{T}(\delta)$ is $\mathcal{T}^m(\delta)$ in the sequence $\mathcal{T}^0(\delta), \mathcal{T}^1(\delta), \ldots, \mathcal{T}^m(\delta)$ constructed inductively from the goal-/π-pruned sequence $\delta' = \langle u'_1 \ldots, u'_m \rangle$ obtained from δ, as follows:

– *$\mathcal{T}^0(\delta)$ is empty;*
– *$\mathcal{T}^1(\delta)$ is constructed as follows,*
  - *if the content of u'_1 is $asm(\alpha)$, then $\mathcal{T}^1(\delta)$ consists only of $(\alpha, ma{:}\mathrm{P}[1])$;*
  - *if the content of u'_1 is $rl(h \leftarrow b_1, \ldots, b_l)$, then $\mathcal{T}^1(\delta)$ consists of $l+1$ nodes, where $(h, mr{:}\mathrm{P}[1])$ is a new node and*

$$(b_1, um{:}\mathrm{P}[1]), \ldots, (b_l, um{:}\mathrm{P}[1])$$

  are children of $(h, mr{:}\mathrm{P}[1])$.
– *Let $\mathcal{T}^i(\delta)$ be the i-th tree, for $0 < i < m$, let $u'_{i+1} = \langle \_, \_, t, C, id, R \rangle$, and let $u'_t = \langle \_, \_, \_, C_t, t, \_ \rangle$ be the target utterance of u'_{i+1}; then $\mathcal{T}^{i+1}(\delta)$ is obtained according to one of the following cases:*
  - *If $R = td$, then*
    * *if $C{=}rl(h \leftarrow b_1, \ldots, b_l)$ then $\mathcal{T}^{i+1}(\delta)$ is $\mathcal{T}^i(\delta)$ with additional l nodes:*

$$(b_1, um{:}\mathrm{L}[id]), \ldots, (b_l, um{:}\mathrm{L}[id])$$

      as children of the node $(h, um : \mathrm{L}[t])$, and this node is replaced by $(h, mr{:}\mathrm{L}[id])$.
    * *if $C = asm(\alpha)$ then $\mathcal{T}^{i+1}(\delta)$ is $\mathcal{T}^i(\delta)$ with the node $(\alpha, um{:}\mathrm{L}[t])$ replaced by $(\alpha, ma{:}\mathrm{L}[id])$.*
    * *if $C = ctr(\alpha, c)$ then $\mathcal{T}^{i+1}(\delta)$ is $\mathcal{T}^i(\delta)$ with an additional node: $(c, um{:}\mathrm{L}[id])$ child of $(\alpha, ma{:}\mathrm{L}'[t])$, where $L, L' \in \{P, O\}, L \neq L'$.*
  - *If $R = bu$, then*
    * *$C = rl(h \leftarrow b_1, \ldots, b_l)$, $\mathcal{T}^{i+1}(\delta)$ is $\mathcal{T}^i(\delta)$ with l additional nodes, in which there is a node $(h, mr{:}\mathrm{L}[id])$, parent of $(b_t, F{:}\mathrm{L}[t])$, such that*
      · *if $C_t = rl(h' \leftarrow b'_1, \ldots, b'_k)$, then $b_t = h'$, $F = mr$;*
      · *if $C_t = asm(\alpha)$, then $b_t = \alpha$, $F = ma$;*

      and the remaining $l-1$ nodes are

$$(b''_1, um{:}\mathrm{L}[id]), \ldots, (b''_{l-1}, um{:}\mathrm{L}[id]),$$

      children of $(h, mr{:}\mathrm{L}[id])$, where $\{b''_1, \ldots, b''_{l-1}\} = \{b_1, \ldots, b_l\} \setminus \{b_t\}$.

Figure 1 gives the construction of the debate drawn from the dialogue in our example[7]. Note that this is a tree but in general it may not be.

To ensure a debate as a tree, we give the following definition.

[7] ↑ represents expanding a rule within an argument. ⇑ represents the attack relation between arguments. Here, wM, aM, gST, fBT, and T are a shorthand for watchMovie, actionMovie, goodScreenTime, finishByTen, and Terminator, respectively.

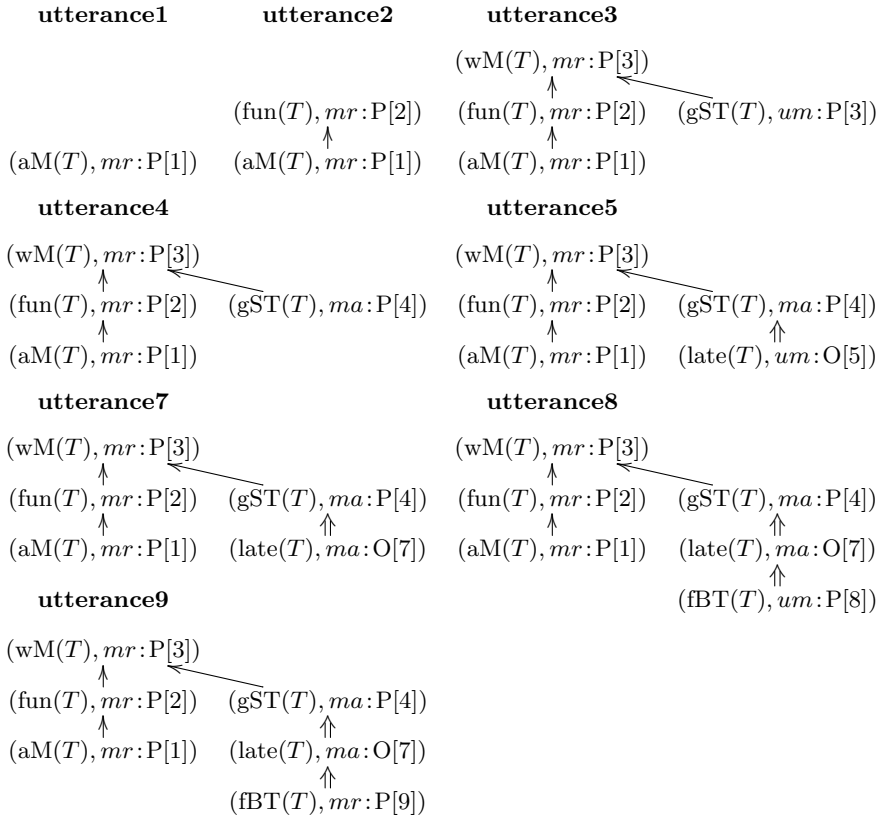


Fig. 1. The construction of the debate tree drawn from the dialogue in our example.

Definition 12. *A debate $\mathcal{T}(\delta)$ is* properly-structured *iff it is a tree.*

This definition is needed as we need to rule out the case of related utterances with contents of the form $p \leftarrow q, q \leftarrow r$ and $p' \leftarrow q$, where the second utterance is top-down related to the first and the third is buttom-up related to the second. Clearly, such situiation prevents a properly-structured debate tree being built.

We ensure debate trees drawn from our dialogues are properly-structured with a legal-move function, as follows.

Definition 13. *A (one-way expansion) legal-move function $\lambda : \mathcal{D} \mapsto \mathcal{U}$ is a* properly-structured legal-move function *if and only if for every dialogue $\delta \in \mathcal{D}$ such that $\mathcal{T}(\delta)$ is properly-structured, then $\mathcal{T}(\delta \circ \lambda(\delta))$ is also properly-structured.*

Thus, when an agent decides what to utter, it needs to keep the current debate tree into account and make sure that its new utterance keeps the tree properly-structured. As a result, the debate tree can be seen as a *commitment store*.

Definition 14. *Given a debate tree $\mathcal{T}(\delta)$,*

- *the* defence set $\mathcal{D}ef(\mathcal{T}(\delta))$ *is the union of all assumptions s in proponent nodes of the form $(s, ma\!:\!\mathrm{P}[\_])$;*
- *the* culprits $\mathcal{C}ul(\mathcal{T}(\delta))$ *are given by the set of all assumptions s in opponent nodes $n = (s, ma\!:\!\mathrm{O}[\_])$ such that the child of n in $\mathcal{T}(\delta)$ is $(\_, \_\!:\!\mathrm{P}[\_])$.*[8]

Arguments can be drawn from a debate tree, as follows:

Definition 15. *An* argument drawn from a debate tree $\mathcal{T}(\delta)$ *is a subtree $\mathcal{T}$ of $\mathcal{T}(\delta)$, such that:*

- *all nodes in $\mathcal{T}$ have the same label (either P or O),*
- *$\nexists$ a node n' in $\mathcal{T}(\delta)$ such that n' is parent or a child of some node n_i in $\mathcal{T}$ and n_i, n' have the same label (either P or O).*

An argument $\mathcal{T}$ is actual *if for all nodes $(\_, F\!:\!\_[\_])$ in $\mathcal{T}$, F is either mr or ma; if there is at least one node of the form $(\_, um\!:\!\_[\_])$ in $\mathcal{T}$, then $\mathcal{T}$ is* potential.

The sentence c in the root of $\mathcal{T}$ is the claim of $\mathcal{T}$. If $\mathcal{T}$ is actual, $\mathcal{T}$ is written as $S \vdash c$, where S is the union of all s such that $(s, ma\!:\!\_[\_])$ is a node in $\mathcal{T}$.

Proposition 1. *Actual arguments can be mapped to equivalent ABA arguments.*

This is trivially true as a node in an actual argument can be mapped to a node in an ABA argument by dropping the tag and the ID. Then add node τ as the child of each leaf node (as each represents a rule with an empty body).

The actual arguments drawn from the dialogue in our example are: (1) $\{\text{goodScreenTime}(Ter)\} \vdash \text{watchMovie}(Ter)$, (2) $\{\text{late}(Ter)\} \vdash \text{late}(Ter)$, and (3) $\{\} \vdash \text{finishByTen}(Ter)$. There is no potential argument in our example.

We consider now restricted forms of debate trees, that we then use below to refine our notion of legal-move function.

Definition 16. *A debate tree $\mathcal{T}(\delta)$ is*

- patient *if and only if for all nodes $n = (\_, ma\!:\!\_[\_])$ in $\mathcal{T}(\delta)$ such that n has a child, then n is in an actual argument drawn from $\mathcal{T}(\delta)$.*
- focused *if and only if for all arguments Γ drawn from $\mathcal{T}(\delta)$, if Γ contains a node $(\_, ma\!:\!\mathrm{O}[\_])$, then there is at most one node c in the form of $(\_, ma\!:\!\mathrm{O}[\_])$ in Γ such that c has a child.*

Arguments in a patient tree are fully expanded (cf. actual) before being attacked. In focused trees, no alternative ways to defend claims are considered simultaneously, i.e., an opponent argument is only attacked by a single proponent argument whereas a proponent argument can be attacked in as many ways as the number of its assumptions. The tree in Figure 1 is both patient and focused.

The restricted form of legal-move function we consider is guaranteed to generate patient, focused trees, as follows.

[8] Definition 14, 15, 16, 19 are adaptations to the case of debate trees of definitions 10, 11, 12, 15, respectively, in [5], whereas they are given for the dialectical trees defined therein.

Definition 17. *A (properly-structured) legal-move function $\lambda : \mathcal{D} \mapsto \mathcal{U}$ is a* patient and focused legal-move function *iff for every dialogue $\delta \in \mathcal{D}$ such that $\mathcal{T}(\delta)$ is patient and focused, $\mathcal{T}(\delta \circ \lambda(\delta))$ is still patient and focused.*

This definition is in the same spirit as Definition 13 presented earlier as it requires agents to consult the debate tree before making utterances. All dialogues in discussion later are constructed with patient and focused legal-move functions.

Definition 18. *The* exhaustive outcome function ω_{ex} *is such that, given $\delta \in \mathcal{D}$, $\lambda \in \Lambda$ and $\langle \mathcal{L}, \mathcal{R}_\delta, \mathcal{A}_\delta, \mathcal{C}_\delta \rangle$ is the framework drawn from δ, $\omega_{ex}(\delta, \lambda) = true$ iff $\omega_{ABA}(\delta, \lambda) = true$ and $\nexists u' = \lambda(\delta)$ with content either $asm(a)$, for $a \in \mathcal{A}_\delta$, or $rl(\rho)$, for $\rho \in \mathcal{R}_\delta$, or $ctr(a, c)$, for $c = \mathcal{C}_\delta(a)$, such that $\omega_{ABA}(\delta \circ u', \lambda) = true$.*

Note that exhaustiveness does not force agents contribute to dialogues with all relevant information they hold. Rather, it enforces agents to make all utterances that are compliant with the given λ.

6 Formal results

In this section we link dialogues and the admissible argumentation semantics. First we refine the outcome function so that if a dialogue has a true outcome then the (fictional) proponent has the last word in the dialogue, namely all leaves in the debate tree are proponent nodes or "dead-end" opponent nodes (not corresponding to any actual argument). Formally:

Definition 19. *The* last word outcome function ω_{lw} *is such that, given $\delta \in \mathcal{D}$, $\lambda \in \Lambda$, and the debate tree drawn from $\mathcal{T}(\delta)$, then $\omega_{lw}(\delta, \lambda) = true$ iff $\omega_{ex}(\delta, \lambda) = true$ and one of the following two cases holds:*

1. *for all leaf nodes n in $\mathcal{T}(\delta)$, n is either $(\_, mr : \mathrm{P}[\_])$ or $(\_, ma : \mathrm{P}[\_])$;*
2. *if a leaf node n is in the form $(\_, \_ : \mathrm{O}[\_])$, then n is in an argument that contains at least one node in the form $(\_, um : \mathrm{O}[\_])$.*

We refer to exhaustive dialogues for which ω_{lw} is true as *positive*. The dialogue in our example is positive. Positive dialogues give debate trees corresponding to *abstract dispute tree* with the same defence set and culprits. Formally:

Lemma 1. *Given a positive dialogue δ constructed with a legal move function λ, let $\mathcal{T}(\delta)$ be the debate tree drawn from δ and s be the sentence in the root node of $\mathcal{T}(\delta)$. Then there is an abstract dispute tree $\mathcal{T}^a$ for $S \vdash s$ for some S, such that $\mathcal{D}ef(\mathcal{T}(\delta)) = \mathcal{D}ef(\mathcal{T}^a)$ and $\mathcal{C}ul(\mathcal{T}(\delta)) = \mathcal{C}ul(\mathcal{T}^a)$.*

Proof. We can transform debate trees into abstract dispute trees. Given a debate tree $\mathcal{T}(\delta)$, its equivalent abstract dispute tree $\mathcal{T}^a$ is constructed as follows.

1. Modify $\mathcal{T}(\delta)$ by appending a new *flag* field $Z = \{0, 1\}$ to each node in $\mathcal{T}(\delta)$ and initialize Z to 0 for all nodes, i.e., a node now looks like $(\_, \_ : \_[\_])[0]$.
2. Delete all nodes n from $\mathcal{T}(\delta)$ where n is in a potential argument.

3. $\mathcal{T}^a$ is $\mathcal{T}^a_m$ in the sequence $\mathcal{T}^a_1, \ldots, \mathcal{T}^a_m$ constructed inductively as follows:
   - $\mathcal{T}^a_0$ is empty;
   - obtain the argument Γ that contains the root of $\mathcal{T}(\delta)$, set the flags of all nodes in $\mathcal{T}(\delta)$ that are also in Γ to 1; then $\mathcal{T}^a_1$ contains a single node that is labelled by Γ and is a P node;
   - let $\mathcal{T}^a_i$ be the i-th tree, for $0 < i < m$, then $\mathcal{T}^a_{i+1}$ is $\mathcal{T}^a_i$ with an additional node (Θ, L), where Θ is an argument drawn from $\mathcal{T}(\delta)$, child of Θ', another argument drawn from $\mathcal{T}(\delta)$, such that
     - the flag of nodes in Θ are 0;
     - the root node of Θ has a parent node p in $\mathcal{T}(\delta)$ which has its flag equal to 1 and p is in Θ';
     - L is P if the root node of Θ is a proponent node, otherwise L is O;
     - set the flags of all nodes in $\mathcal{T}(\delta)$ that are also in Θ to 1.
   - $\mathcal{T}^a_m$ is constructed when there is no node in $\mathcal{T}(\delta)$ with its flag equal to 0.

Trivially, $\mathcal{T}^a$ constructed above is an abstract dispute tree. Since $\mathcal{T}^a$ contains the same arguments as $\mathcal{T}(\delta)$ and arguments have the same P/O labelling in both $\mathcal{T}^a$ and $\mathcal{T}(\delta)$, we have $\mathcal{D}ef(\mathcal{T}(\delta)) = \mathcal{D}ef(\mathcal{T}^a)$ and $\mathcal{C}ul(\mathcal{T}(\delta)) = \mathcal{C}ul(\mathcal{T}^a)$.

As in the case of abstract dispute trees, the defence set of a debate tree may not be admissible, as it may attack itself. We refine the notion of outcome function by enforcing that this set does not attack itself, as follows:

Definition 20. *The* admissible outcome function ω_{ADM} *is such that, given* $\delta \in \mathcal{D}$, $\lambda \in \Lambda$, $\omega_{ADM}(\delta, \lambda) = true$ *iff* $\omega_{lw}(\delta, \lambda) = true$ *and* $\mathcal{D}ef(\mathcal{T}(\delta)) \cap \mathcal{C}ul(\mathcal{T}(\delta)) = \{\}$. *If* δ *is positive and* $\omega_{ADM}(\delta) = true$, *we say that* δ *is* successful.

Theorem 1. *Given a* successful *dialogue* $D^{a_i}_{a_j}(\mathcal{G}) = \delta$ *constructed with* λ, *let* s *be the sentence in the root node of* $\mathcal{T}(\delta)$. *If* $\omega_{ADM}(\delta, \lambda) = true$, *then there exists an argument* $S \vdash s$ *that belongs to an* admissible *extension supported by* $\mathcal{D}ef(\mathcal{T}(\delta))$ *w.r.t. the ABA framework drawn from* δ.

Proof. If $\omega_{ADM}(\delta, \lambda) = true$, by Lemma 1 there exists an abstract dispute tree $\mathcal{T}^a$ such that $\mathcal{D}ef(\mathcal{T}(\delta)) = \mathcal{D}ef(\mathcal{T}^a)$ and $\mathcal{C}ul(\mathcal{T}(\delta)) = \mathcal{C}ul(\mathcal{T}^a)$. By Theorem 5.1 of [4] (see Section 2), the theorem holds.

Theorem 1 connects dialogues with argumentation semantics. Thus, our dialogues can be seen as a distributed mechanism for computing admissible extensions. With this, we prove a result specifically for discovery dialogues.

Theorem 2. *Given a successful dialogue* $\delta = D^{a_i}_{a_j}(\mathcal{G})$, *let* $(\mathcal{G}\sigma, \_ : \mathrm{P}[\_])$ *be the root node of* $\mathcal{T}(\delta)$, *where* $\sigma = \{X/t\}$. *Then* $\mathcal{G}\sigma$ *is a goal realization for* $\mathcal{G}$ *w.r.t. to the ABA framework drawn from* δ.

Proof. If $\mathcal{G}\sigma$ is the sentence in the root node of $\mathcal{T}(\delta)$ and δ is successful, then $\mathcal{G}\sigma$ is admissible w.r.t. AF_δ, the ABA framework drawn from δ, by Theorem 1. Hence $\mathcal{G}\sigma$ is a goal realization with respect to AF_δ.

7 Related Work

McBurney and Parsons in [9] give an overview of dialogue games for argumentation. Our work can be seen as providing a novel dialogue game for ABA.

McBurney and Parsons [8] present a modelling for chance discovery dialogue. The formal system in that work is defined with locutions and rules without linking to any argumentation framework, whereas our work is based on ABA. There is no argumentation semantics used in examining the result of their dialogues, whereas our work makes the connection to the admissibility semantics. Moreover, [8] focuses on chance discovery, whereas our work is applicable to any discoveries as long as the desired outcome can be qualified, essentially, by a predicate.

Rybakov [13] presents a logic modelling of chance discovery. Our work differs from that as is focuses on a dialogue system for discovery whereas his is mainly concerned with constructing a modal/temporal modelling for chance discovery.

Fisher [7] presents a mechanism for concurrent theorem-proving. In his setting, the knowledge base (a set of formulae) is distributed at different objects and each object continuously broadcasts messages about its formulae. Upon receiving messages, an object makes inferences, transforms its formulae and sends out further messages. Even though similarity exists, this work is vastly different from ours as (1) it does not focus on discovering any particular information; (2) it is not concerned with either agents or dialogues; (3) it requires formulae to be represented in Horn Clauses. No formal results have been shown in [7].

Fan and Toni [5] introduce a formal modelling for argumentation dialogues. Though the modelling presented in [5] is generic, it starts a dialogue with a claim and it only uses top-down reasoning, whereas our work here allows agent to start the dialogue with any information and reason buttom-up. Moreover, the result of [5] is based on the concrete dispute trees of [4] whereas our work here uses the abstract dispute trees of [4]. Our work can be viewed as an extension of [5], apart from the fact that the framework introduced there does not require a dialogue to be built with a one-way expansion legal-move function.

[1] present a formal system for inquiry dialogues based on DeLP as the underlying argumentation framework. Our work differs from theirs as (1) our work does not start the dialogue with a claim; (2) it does not focus on inquiry dialogues; (3) it does not force full disclosure of all relevant knowledge.

[11] defines a formal system for persuasion. The main differences with our work are: (1) their work starts the dialogue with a claim (2) proponent and opponent roles are pre-assigned to agents before the dialogue whereas in our work agents can play both roles within the same dialogue; (3) he considers the grounded semantics, whereas we use admissibility; (4) his set of utterances refer to arguments and attacks, as in the case of [10]; (5) he forces the support of arguments to be minimal, whereas we do not.

8 Conclusion

In this paper, we introduce a formal modelling for a simple form of discovery dialogue for two agents, in which the desired outcomes are only partially known

when dialogues start. In our setting, the two agents share the same discovery goal but neither of the two agents is capable of discovering a justified goal realization that fulfills the shared goal. A discovery dialogue is successful if a goal realization is found through the dialogue.

In our model, the dialogue effectively starts by one agent putting forward a piece of information that might be related to the goal realization. Through dialogues, more information that is related to the initial utterance is gathered and examined. This process is defined with various legal-move functions with the help of constructing a debate tree. We examine the acceptability of the goal realization with the admissibility semantics.

Future work includes investigating other forms of discovery dialogues, e.g., where agents can change their abstract goals with some form of buttom-up reasoning and studying cases in which results about completeness can be obtained, namely, conditions of a goal realization discovery can be guaranteed. We are also interested in allowing more than one "wild guess" in a dialogue and the possibility of agents to have preferences over realizations.

References

1. E. Black and A. Hunter. An inquiry dialogue system. *JAAMAS*, 19:173–209, 2009.
2. A. Bondarenko, P. M. Dung, R. Kowalski, and F. Toni. An abstract, argumentation-theoretic approach to default reasoning. *AIJ*, 93(1-2):63–101, 1997.
3. P. M. Dung, R. A. Kowalski, and F. Toni. Assumption-based argumentation. In *Argumentation in Artificial Intelligence*, pages 25–44. Springer, 2009.
4. P.M. Dung, R.A. Kowalski, and F. Toni. Dialectic proof procedures for assumption-based, admissible argumentation. *AIJ*, 170:114–159, 2006.
5. X. Fan and F. Toni. Assumption-based argumentation dialogues. In *Proc. IJCAI*, 2011. To appear.
6. X. Fan and F. Toni. Conflict resolution with argumentation dialogues – Extended abstract. In *Proc. AAMAS*, 2011.
7. Michael Fisher. An open approach to concurrent theorem-proving. In *Parallel Processing for Artificial Intelligence*, pages 209–230. Elsevier/North, 1997.
8. P. McBurney and S. Parsons. Chance discovery using dialectical argumentation. In *Proc. Joint JSAI 2001 Workshop on New Frontiers in AI*, pages 414–424, 2001.
9. P. McBurney and S. Parsons. Dialogue games for agent argumentation. In *Argumentation in Artificial Intelligence*, pages 261–280. Springer, 2009.
10. S. Parsons, P. McBurney, E. Sklar, and M. Wooldridge. On the relevance of utterances in formal inter-agent dialogues. In *Proc. AAMAS*, pages 47–62, 2007.
11. H. Prakken. Coherence and flexibility in dialogue games for argumentation. *JLC*, 15:1009–1040, 2005.
12. H. Prakken. Formal systems for persuasion dialogue. *Knowledge Eng. Review*, 21(2):163–188, 2006.
13. V. Rybakov. Logic of knowledge and discovery via interacting agents - decision algorithm for true and satisfiable statements. *Inf. Science*, 179:1608–1614, 2009.
14. D. Walton and E. Krabbe. *Commitment in Dialogue: Basic concept of interpersonal reasoning*. State University of New York Press, Albany NY, 1995.

Argument Types and Typed Argumentation Frameworks

Sebastian Gottifredi, Alejandro J. García, and Guillermo R. Simari

Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)
Artificial Intelligence Research and Development Laboratory,
Department of Computer Science and Engineering - Universidad Nacional del Sur,
Bahía Blanca, ARGENTINA,
e-mail: {sg, ajg, grs}@cs.uns.edu.ar

Abstract. The concept of argument type is inherent to the computational study of argumentation. However, this concept has always been underlying and fixed to the problem for which the argumentation is used. In this work we will present a concrete representation of argument types based on abstract argumentation frameworks. Here, each type will be identified by a set of abstract arguments and relations among them. In addition, we will present a set of type relations to establish possible conflicts, preferences and inheritances among the argument types. Individual types and their relation will characterize a Multi-Typed Argumentation Framework. This framework will allow to evaluate the acceptability of its arguments taking into account the type information and relations. Examples and an instantiation in an agent programming language for the proposed framework will be shown.

1 Introduction

In the past years several approaches were proposed to model argumentation on an abstract basis [11], using classical logics [8], or using logic programming [12, 16]. In general, this area concerns the study of argument structures, the interplay of arguments, and the outcome of argumentation on a formal basis. Argumentation frameworks can be used in practical reasoning problems, decision-making problems, negotiation problems and multi-agent dialogues (see e. g. [6, 9]).

In the literature, some formalisms use different kinds of arguments for representing diverse kinds of information, or for distinguishing arguments that are used for distinct purposes. For instance, in [4] *explanatory*, *rewards* and *threats* arguments are used for negotiation dialogues, and in practical reasoning approaches [1, 18, 13] different types of arguments are used to represent categorized domain information, like *belief*, *goals* or *plans*.

The idea of argument type involves the characterization of a group of arguments that have some common features (*e.g.*, its structure, conflict or preference relations) and therefore, these features distinguish them from other groups of arguments. In the mentioned approaches, argument types and their features are usually defined specifically for the particular domain or particular problem that

the formalism is conceived for (*e.g.*, dialogue systems, practical reasoning, agent programming, etc.). That is, the concept of argument type is underlying to the formalism, and hence, the approach does not provide a general or a formal characterization of the notion of argument type.

In this work we aim to formally address the concepts of Argument Type and Multi-Type Argumentation Framework in the context of abstract argumentation. Thus, the contribution of this work is a formalization that allows to specify individual argument types (called Single Type Argumentation Framework or STAF) where special features for a group of arguments can be defined (*e.g.,* conflict and preferences), and relations among argument types can be introduced (*e.g.*, inheritance, preference or conflicts) in a Multi-Type Argumentation Framework (or MTAF). For instance, consider the following scenario where the set of arguments $\{\mathcal{A}_1 \dots \mathcal{A}_n\}$ is such that every argument shares certain features, and therefore are considered as arguments of type T_A. Let us also consider that there is another set of arguments $S_B = \{\mathcal{B}_1 \dots \mathcal{B}_k\}$ of type T_B, and in the domain being modelled arguments of T_A are always preferable to arguments of type T_B. However, there is also a specialization of T_B (called T_C), that is, a subset $S_C \subset S_B$ that inherits some properties of T_B, but all arguments of type T_C are preferable to arguments of type T_A. As we will show next, this kind of relations between arguments and types of arguments can be represented in a Multi-Type Argumentation Framework.

The rest of this paper is organized as follows. In Section 2, we will present the Multi-Typed Argumentation Frameworks. First, we will describe how to represent individual types using abstract argumentation frameworks with preferences, and then we will show how to establish conflicts, preferences and inheritances among the types. Later, we will show how to determine attacks, preferences, and defeats in the context of the multi typed frameworks. In section 3 we will show how to evaluate arguments in these frameworks. Then, in Section 4, we will show the reasoning model of an agent programming language that will be based on the presented framework. Finally, we conclude this work with a discussion of related and future work.

2 Typed Argumentation

A classical abstract argumentation framework is composed by a set of arguments and a defeat relation among them. When an argument defeats another, they can not be accepted simultaneously. Therefore, an analysis of the defeat relation should be carried out to determine which arguments are finally acceptable.

The Multi-Typed Abstract Argumentation Frameworks that we propose in this work will also contain arguments and a defeat relation. However, arguments will belong to individual frameworks representing single types and, following the ideas of [2], these single types will establish attacks and preferences between their arguments. The single types will relate to each other by type preferences, conflicts and inheritances. Therefore, the defeat relation will be built considering

all these elements and will be used to determine which arguments are finally acceptable.

Next, we introduce a representation for an individual type of a Multi-Typed Argumentation Framework. These single types are referred as Single Typed Argumentation Frameworks, and are based on the preference based argumentation frameworks [2, 15]. Basically, a single type is formed of three elements: a set of arguments and two relations among them.

Definition 1 (Single Typed Argumentation Framework). *A Single Typed Argumentation Framework (STAF) is a tuple $ST = (AR, \rightsquigarrow, \succeq)$, where AR is a finite set of arguments, $\rightsquigarrow$ is an attack relation between arguments $\rightsquigarrow \subseteq AR \times AR$, and $\succeq$ is an anti-reflexive preference relation $\succeq \subseteq AR \times AR$.*
The functions $\mathsf{Args}(ST), \mathsf{Atts}(ST), \mathsf{Prefs}(ST)$ will respectively return the arguments, the attacks and the preferences of ST.

Here, arguments are abstract entities that will be denoted using calligraphic uppercase letters. No reference to the underlying logic is needed since we are abstracting the structure of the arguments. The attack relation between two arguments $(\mathcal{A}, \mathcal{B}) \in \rightsquigarrow$, noted $\mathcal{A} \rightsquigarrow \mathcal{B}$, represents that the argument $\mathcal{A}$ attacks the argument $\mathcal{B}$, *i.e.* that $\mathcal{A}$ is in conflict with $\mathcal{B}$. Note that there can be symmetrical attacks. The relation $\succeq$ is introduced in the framework to evaluate arguments, modelling a preference criterion based on a measure of strength.

Example 1. An agent is considering buying a house in the suburb A of the city CT. To make a decision the agent has to consider arguments of different types:

T_1) arguments from newspapers of CT that have some news about safety in the city, and
T_2) arguments from a web site of the city CT that publish people's comments about suburbs

Type T_1 has the following arguments:

- a newspaper has published an article N saying that crime has increased in the suburb A of city CT (argument $\mathcal{C}$),
- that newspaper articles are usually based on rumors (argument $\mathcal{R}$),
- the article N is based on an interview with the chief of the police department (argument $\mathcal{P}$).

These arguments will be characterized by the argument type represented with the STAF $T_1 = (\{\mathcal{C}, \mathcal{R}, \mathcal{P}\}, \{\mathcal{R} \rightsquigarrow \mathcal{C}, \mathcal{P} \rightsquigarrow \mathcal{R}\}, \{\mathcal{P} \succeq \mathcal{R}\})$. Note that the preference relation in this STAF denotes that the agent favours the information source of the article N to the general knowledge that news can be based on rumours.
Type T_2 has the following arguments:

- suburb B is the best investment option for city CT. (argument $\mathcal{B}$),
- houses of suburb A are expensive (argument $\mathcal{A}$),
- suburb A is considered a safe place, thus, it is desirable to buy a house there (argument $\mathcal{S}$)

Using the following STAF we will characterize this argument type: $T_2 = (\{\mathcal{B}, \mathcal{A}, \mathcal{S}\}, \{\mathcal{B} \rightsquigarrow \mathcal{S}, \mathcal{A} \rightsquigarrow \mathcal{S}\}, \{\mathcal{S} \succeq \mathcal{B}, \mathcal{S} \succeq \mathcal{A}\})$, where arguments $\mathcal{A}$ and $\mathcal{B}$ both attack $\mathcal{S}$ but the argument regarding security is preferred over the other two.

In a classic argumentation framework, like in [15], the preference relation and the attack relation are used to determine if an attack is either effective or ineffective. That is, if an attack is either a *defeat* or not. Therefore, intuitively, in Example 1 $\mathcal{P}$ would *defeat* $\mathcal{R}$. However, in the context of multi-typed frameworks, it is not sufficient to consider only the relations of a single type, since there can be several interacting types. Below, we will present the concept of defeat after introducing the formalisation of a Multi-Typed Argumentation Framework.

2.1 Type Relations over STAFs

In the context of multi-typed argumentation, individual argument types are expected to interact with each other through different relations. In this work, we will allow to define attacks, preferences and inheritance among argument types. These type-relations will determine the dependence among individual types, and will be used to figure out which arguments will be ultimately accepted in the multi-typed framework.

Usually, when we have different argument types it is desirable to allow attacks among arguments of these individual types. To represent this concept we will use a function which returns the pairs of arguments in conflict from two STAFs.

Definition 2 (Type Attack Function). *Given two STAFs T_1 and T_2, a Type Attack Function $\mathsf{TAtt}(T_1, T_2)$ will return a set C of pairs $(\mathcal{A}, \mathcal{B})$, such that $\forall(\mathcal{A}, \mathcal{B}) \in C$ either $\mathcal{A} \in \mathsf{Args}(T_1)$ and $\mathcal{B} \in \mathsf{Args}(T_2)$, or $\mathcal{A} \in \mathsf{Args}(T_2)$ and $\mathcal{B} \in \mathsf{Args}(T_1)$.*

Example 2. Consider the scenario presented in Example 1. In this case, argument $\mathcal{C}$ of T_1, expressing that there is crime in the suburb A, attacks argument $\mathcal{S}$ of T_2, expressing that the suburb A is safe, and *vice versa*. This will be represented by $\mathsf{TAtt}(T_1, T_2) = \{(\mathcal{C}, \mathcal{S}), (\mathcal{S}, \mathcal{C})\}$.

The motivation behind this function is to capture those conflicts that emerge from the integration of different argument types, and may have some semantical connotation regarding to these argument types. For instance, as we will see in Section 4, it is usual that a belief argument supporting a conclusion attacks a goal argument supporting the same conclusion. The TAtt function provides a mechanism to explicitly represent these kind of situations.

Looking into the situation depicted in the previous examples, it will be desirable to express a preference among argument types. This preference will help to determine if the attacks involving arguments of the different STAFs are effective or not. To model these notions we will present the type preference relation.

Definition 3 (Type Preference Relation). *Let S_T be a set of STAFs. A Type Preference Relation $>>$ is a binary, non-reflexive, non-symmetric relation defined over the elements of S_T.*

Example 3. Consider the situation described in examples 1 and 2. Now suppose that the agent wants to establish that prefers arguments from T_1 to arguments of T_2, because newspaper arguments are more reliable than web site arguments. This will be represented by the type preference relation $T_1 >> T_2$.

Intuitively, introducing these relations among the argument types of our examples, we will have that $\mathcal{C}$ of type T_1 *defeats* $\mathcal{S}$ of type T_2. However, to formally define the notion of *defeat* in multi-typed argumentation we have to analyse all possible relations among argument types.

Another interesting relation among argument types that will be introduced in this work is the *Inheritance Relation*. This relation will be used to determine an "is-a" relation among argument types, where we will identify a base type and a subtype. The subtype (or inheriting type) will contain a subset of the arguments of the base type and may extend or override the internal argument relations (*i.e.* attacks and preferences).

Definition 4 (Inheritance Relation). *Let S_T be a set of STAFs, $\rightarrowtriangle$ is an Inheritance Relation over S_T iff $\rightarrowtriangle$ is an anti-reflexive, non-symmetric and transitive relation and $\forall T_i, T_j \in S_T$ such that if $T_i \rightarrowtriangle T_j$ it holds that $\mathsf{Args}(T_i) \subseteq \mathsf{Args}(T_j)$.*

A main motivation for presenting this relation is to allow the representation of type specialization. That is, using inheritance and preferences among types, it will be possible to make arguments of a specialized type S preferred over arguments of other types, even though the base type of S was not.

Example 4. Consider the situation described in examples 1, 2, and 3, and suppose that the argument $\mathcal{S}$, expressing that suburb A is safe, was obtained from a very reliable friend that lives in there. Then, the agent should prefer these arguments to any other arguments. However, they are still arguments of type T_2 which were not preferred to arguments of type T_1. Therefore, the arguments from the friend will constitute a specialized type which inherits from type T_2 and is preferred to type T_1. This specialized type will be represented by the STAF $T_3 = (\{\mathcal{S}\}, \emptyset, \emptyset)$, where $T_3 \rightarrowtriangle T_2$ and $T_3 >> T_1$.

Note that in Definition 4 we did not included any restriction on the sets of conflicts and preferences. This is because the subtype will be able to override or establish newer relations among the inherited arguments. In addition, observe that now $\mathcal{S}$ belongs to two STAFs T_2 and T_3. In the following subsection, we will show the impact of this situation when we define the defeat relation.

2.2 Multi-Typed Argumentation Frameworks

Now that we have presented a representational model for individual types of arguments and relations among them, we can formally define the multi-typed argumentation frameworks. Basically, these frameworks will consist on a set of single typed argumentation frameworks and three type relations previously introduced.

Definition 5 (Multi-Typed Argumentation Framework). *A Multi-Typed Argumentation Framework (MTAF) is a tuple* $MT = (S_T, \rightarrowtriangle\!\!\leftarrow, \mathsf{TAtt}, >>)$*, where* S_T *is a set of single typed argumentation frameworks,* $\rightarrowtriangle\!\!\leftarrow$ *is an Inheritance Relation defined over the elements of* T_S*,* TAtt *is a Type Attack Function, and* $>>$ *is a Type Preference Relation among the elements of* S_T*.*
The function $\mathsf{MArgs}(MT)$ *will return the arguments of all the STAFs in* MT*, that is,* $\mathsf{MArgs}(MT) = \bigcup \mathsf{Args}(T_i), T_i \in S_T$*.*

Example 5. To model the complete situation described in previous examples a $MTAF$ will be used. This MTAF will be $MT_B = (S_T, \rightarrowtriangle\!\!\leftarrow, \mathsf{TAtt}, >>)$, where $S_T = \{T_1, T_2, T_3\}$ with T_1, T_2 and T_3 the STAFs from examples 1, 2 and 4 respectively, $\rightarrowtriangle\!\!\leftarrow$ is the inheritance relation of example 4, TAtt is the type attack function of example 2, and $>>$ the type pref. relation of examples 3 and 4.

Since we have inheritance among the argument types of a MTAF it is possible for an argument to belong to more than one type. The Types function will return the types associated to an argument.

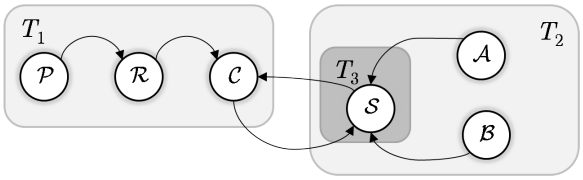
Definition 6 (Types Function). *Let* MT *be a MTAF and* $\mathcal{A}$ *an argument such that* $\mathcal{A} \in \mathsf{MArgs}(MT)$*. The Types Function* $\mathsf{Types}(\mathcal{A}, MT)$ *will return a set* A_T *of STAFs such that* $T_i \in A_T$ *iff* $\mathcal{A} \in \mathsf{Args}(T_i)$*.*

As explained in the previous subsections, using a MTAF will allow to determine to determine the status of the arguments from the different types involved in the framework. To achieve this, we will need to determine defeats among the arguments of the whole framework. Recall that to establish defeat we need to find which attacks are effective regarding to the preference among arguments. Therefore, to determine if an argument $\mathcal{A}$ from an arbitrary type defeats an argument $\mathcal{B}$ of possibly another type, there should be an attack from $\mathcal{A}$ to $\mathcal{B}$, and $\mathcal{A}$ should be preferred to $\mathcal{B}$. Thus, we should define how to establish attacks and preferences among the arguments of a MTAF.

An attack among two arguments of a MTAF will occur if there is an attack between these arguments in one of the STAFs they belong to, or if the attack is established by the type attack function.

Definition 7 (MTAF - Attack). *Let* $MT = (S_T, \rightarrowtriangle\!\!\leftarrow, \mathsf{TAtt}, >>)$ *be a MTAF and* $\mathcal{A}, \mathcal{B} \in \mathsf{MArgs}(MT)$*. We will say that A attacks B in MT, noted* $\mathcal{A} \rightharpoonup \mathcal{B}$*, iff* $\exists T_i \in S_T$ *such that* $\mathcal{A} \rightsquigarrow \mathcal{B} \in \mathsf{Atts}(T_i)$*, or* $\exists T_A, T_B$ *such that* $T_A \in \mathsf{Types}(A, MT)$ *and* $T_B \in \mathsf{Types}(B, MT)$*, and* $(\mathcal{A}, \mathcal{B}) \in \mathsf{TAtt}(T_A, T_B)$*.*

For instance, recall the MTAF MT_B of Example 5, representing the scenario of an agent considering buying a house. Below we present a graph depicting the attacks in MT_B, where the rectangles are the STAFs, the circles are the arguments, and the light arrows are the obtained attacks.



The argument preference relation in the context of a MTAF requires a deeper analysis compared to attacks. This is manly because preferences are related to the inheritance dependence among individual types. Slightly, we will say that an argument will be preferred to another argument if there is a preference established in (between) some type, and this preference is not overridden by a more specific type. Therefore, we should analyse the inheritance relation among types.

The inheritance relation in the MTAFs will denote an inheritance network [14]. These networks are acyclic directed graphs, where nodes are types and directed edges are inheritance relations among those types. To show that this kind of networks are feasible with our inheritance relation, we will demonstrate that the inheritance relation is non-circular.

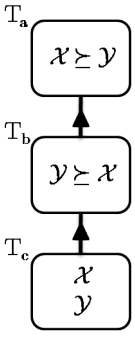
Proposition 1 (Non-Circular Inheritance). *Let* $MT=(S_T, \rightarrow\!\triangleright\!\leftarrow, \mathsf{TAtt}, >>)$ *be a MTAF, and* $T_i, T_j \in S_T$ *such that* $T_i \rightarrow\!\triangleright\!\leftarrow T_j$. *Then* $\nexists T_k \in S_T$ *such that* $T_k \rightarrow\!\triangleright\!\leftarrow T_i$ *and* $T_j \rightarrow\!\triangleright\!\leftarrow T_k$.

Proof. Let us assume that $\exists T_i, T_j, T_k \in S_T$ *such that* $T_i \rightarrow\!\triangleright\!\leftarrow T_j$, $T_k \rightarrow\!\triangleright\!\leftarrow T_i$ *and* $T_j \rightarrow\!\triangleright\!\leftarrow T_k$. *Then by transitivity* $T_i \rightarrow\!\triangleright\!\leftarrow T_i$, *which is a contradiction because the inheritance relation must be non-reflexive.* □

Therefore, it is possible to build a directed acyclic graph using the $\rightarrow\!\triangleright\!\leftarrow$ relation of a MTAF. This graph will be called a $\rightarrow\!\triangleright\!\leftarrow$ graph, an inheritance network, where its nodes will be the STAFs and the edges will be determined by the $\rightarrow\!\triangleright\!\leftarrow$ relation of the associated MTAF. These graphs will be used to explain how preferences among arguments in a MTAF are defined.

In order to determine argument preferences in the context of MTAFs, we should consider two cases: when preferences are established within a type, and when preferences are established outside a type with the type preference relation. Next, we will analyse both situations and provide definitions for each partial preference.

As it was previously explained, an STAF can define preferences among its arguments. However, type inheritance in a MTAF allows a subtype to redefine the preference relations of its ancestors. For instance, consider the $\rightarrow\!\triangleright\!\leftarrow$ graph on the right, where $T_c \rightarrow\!\triangleright\!\leftarrow T_b \rightarrow\!\triangleright\!\leftarrow T_a$, and the arguments $\mathcal{X}$ and $\mathcal{Y}$ belong to the three types. Let us analyse the preference relation between $\mathcal{X}$ and $\mathcal{Y}$. The STAF T_a establishes that $\mathcal{X} \succeq \mathcal{Y}$, but this preference is overridden by the preference specified in T_b, since T_b is a more specialized type than T_a. Even though T_c is more specialized than T_b, T_c does not establish a preference between $\mathcal{X}$ and $\mathcal{Y}$. Therefore, in this MTAF $\mathcal{Y}$ will be preferred to $\mathcal{X}$.

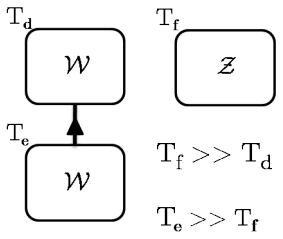


This illustrates the notion of *internal preference*, where the preference between two arguments is established by a STAF, but can be overriden by more specialized STAFs. Then, we will say that an argument $\mathcal{N}$ is internally preferred to other argument $\mathcal{M}$, if for every STAF that established that $\mathcal{M}$ is preferred to $\mathcal{N}$, there is a more specialized STAF establishing the contrary. Formally:

Definition 8 (Internal Preference). *Let $MT = (S_T, \rightarrow\!\!\triangleright\!\!\leftarrow, \mathsf{TAtt}, >>)$ be a MTAF, and $\mathcal{A}, \mathcal{B} \in \mathsf{MArgs}(MT)$. The argument $\mathcal{A}$ is internally prefered to the argument argument $\mathcal{B}$, noted $\mathcal{A} >_I \mathcal{B}$, iff $\exists T_i \in S_T$ such that $\mathcal{A} \succeq \mathcal{B} \in T_i$, and $\forall T_j \in S_T$ such that $\mathcal{B} \succeq \mathcal{A} \in T_j$ it holds that $\exists T_k \in S_T$ such that $T_k \rightarrow\!\!\triangleright\!\!\leftarrow T_j$ and $\mathcal{A} \succeq \mathcal{B} \in T_k$.*

Note that this definition follows the spirit of preference aggregation of [3]. In that approach, there are several preferences for a set of arguments, and there is a complete ordering on those preferences used to determine which arguments are finally preferred. Here, each STAF specifies a preference relation among its arguments and the inheritance relation implicitly imposes an order among these preference relations. However, differing from [3], this order is not a complete order since there can be several inheritance paths in the $\rightarrow\!\!\triangleright\!\!\leftarrow$ graph where a preference for two arguments is defined.

Another situation that we must consider is when there are inheritance and preferences among types. For instance, consider the situation depicted on the right, and let us analyse the preference relation between $\mathcal{W}$ and $\mathcal{Z}$ in the context of the MTAF. There are three STAFs T_d, T_e and T_f, such that $T_e \rightarrow\!\!\triangleright\!\!\leftarrow T_d$, and the preference relation depicted in the figure. Even though T_f is preferred to T_d, the argument $\mathcal{W}$ also belongs to a more specialized type (T_e) which is preferred to T_f. Therefore, in this case $\mathcal{W}$ will be preferred to $\mathcal{Z}$. Note that the situation described in Example 4 is similar.



This example depicts the other form of preference in a MTAF, the *external preference*. Here the preference determined by the type preference relation can be overridden by a type preference relation involving a more specialized type. Then, we will say that an argument $\mathcal{N}$ is externally preferred to an argument $\mathcal{M}$, if for every type T_M of $\mathcal{M}$ preferred to a type of $\mathcal{N}$, there is a more specialized type (wrt. the less preferred) of $\mathcal{N}$ which is preferred to T_M. Formally:

Definition 9 (External Preference). *Let MT be a MTAF, and $\mathcal{A}, \mathcal{B} \in \mathsf{MArgs}(MT)$ such that $\mathsf{Types}(\mathcal{A}, MT) = Ts_A$ and $\mathsf{Types}(\mathcal{B}, MT) = Ts_B$. The argument $\mathcal{A}$ is externally preferred to the argument argument $\mathcal{B}$, noted $\mathcal{A} >_E \mathcal{B}$, iff $\exists T_1 \in Ts_A$ and $\exists T_2 \in Ts_B$ such that $T_1 >> T_2$, and $\forall T_i \in Ts_B$ such that $T_i >> T_j$ with $T_j \in Ts_A$ it holds that $\exists T_k \in Ts_A$ such that $T_k \rightarrow\!\!\triangleright\!\!\leftarrow T_j$ and $T_k >> T_i$.*

The notions formalized in definitions 8 and 9 follow the principle of preemption from the inheritance network literature [14]. The intuition behind this

concept is that relations based on more specific types override relations based on less specific types. Both definitions follow this principle requiring that the preference established in a type is not overriden by preferences established in more specific types.

In order to define a preference relation considering the whole MTAF we have to relate the external and internal preferences. If an argument $\mathcal{A}$ is externally and internally preferred to an argument $\mathcal{B}$, clearly, it will be preferred to $\mathcal{B}$ in the MTAF. In the completely opposite case, where $\mathcal{A}$ is neither internally nor externally preferred to $\mathcal{B}$, $\mathcal{A}$ will not be preferred to $\mathcal{B}$ in MTAF. In the case that $\mathcal{A}$ is externally, but not internally, preferred to $\mathcal{B}$, we have two possibilities: either there is not an internal preference established between $\mathcal{A}$ and $\mathcal{B}$, or $\mathcal{B}$ is internally preferred to $\mathcal{A}$. In the former, clearly, $\mathcal{A}$ will be preferred to $\mathcal{B}$ in the MTAF. In the latter, no preference will be established regarding $\mathcal{A}$ and $\mathcal{B}$ in the MTAF. There is an analogous situation if $\mathcal{A}$ is internally preferred to $\mathcal{B}$, but not externally preferred. In the following definition we capture these intuitions.

Definition 10 (Peferred Argument). *Let $MT = (S_T, \rightarrowtriangle\!\!\!\!-, \mathsf{TAtt}, >>)$ be a MTAF. An argument $\mathcal{A}$ is MT-preferred to an argument $\mathcal{B}$, noted $\mathcal{A} >_{MT} \mathcal{B}$ iff:*

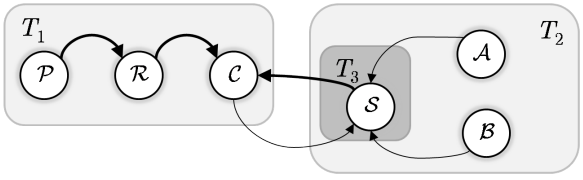
- *$\mathcal{A} >_E \mathcal{B}$ and $\mathcal{B} \not>_I \mathcal{A}$, or*
- *$\mathcal{A} >_I \mathcal{B}$ and $\mathcal{B} \not>_E \mathcal{A}$.*

For instance, in the MTAF of Example 5 we have: $\mathcal{S} >_{MT} \mathcal{A}$, $\mathcal{S} >_{MT} \mathcal{B}$, $\mathcal{P} >_{MT} \mathcal{R}$, $\mathcal{R} >_{MT} \mathcal{C}$, $\mathcal{S} >_{MT} \mathcal{C}$. Now that we have defined the notions of attack and preference for MTAFs we can define the defeat relation similarly to [15].

Note that we have taken a cautious approach when defining MTAF preference: when internal and external preferences disagree no preference is established between the arguments. Another alternative would be to favor either the internal or the external preference between the compared arguments.

Definition 11 (Defeat). *Let MT be a MTAF. An argument $\mathcal{A}$ defeats another argument $\mathcal{B}$ ($\mathcal{A}$ is a defeater for $\mathcal{B}$), noted $\mathcal{A}{\rightarrow}\mathcal{B}$, iff $\mathcal{A}{\rightharpoonup}\mathcal{B}$ and $\mathcal{A} >_{MT} \mathcal{B}$, or $\mathcal{A} \not>_{MT} \mathcal{B}$ and $\mathcal{B} \not>_{MT} \mathcal{A}$.*

Below we present a graph showing which attacks result in defeat for the MTAF of Example 5. The broad black arrows represent defeats, and light arrows represent attacks that did not result in defeat.



In next section, the classic acceptability notion is applied to MTAFs, in order to analyse which arguments are accepted regarding to the notion of defeat in multi-typed argumentation.

3 Typed Acceptability Semantics

Since arguments in a MTAF can defeat each other, they can not be accepted simultaneously. Therefore, the arguments status will be subject to a status evaluation. An argument will be accepted if it somehow "survives" the defeats it receives, or rejected otherwise. This evaluation process is defined by the arguments acceptability semantics.

In this work we will follow the extensional based semantics (see [5] for more details). An extension is a set of arguments that intuitively are jointly acceptable. In the literature a wide variety of different types of extensions has been proposed. All of these are based on the notions of conflict-freeness and acceptability. Next, we will define these basic notions in the context of MTAFs.

Definition 12 (MTAF Acceptability). *Let MT be a MTAF. An argument $\mathcal{A} \in \mathsf{MArgs}(MT)$ is acceptable with respect to a set of arguments $S \subseteq \mathsf{MArgs}(MT)$ iff every defeater $\mathcal{B}$ of $\mathcal{A}$ has a defeater in S.*

Definition 13 (Conflict Free). *Let MT be a MTAF. $CF \subseteq \mathsf{MArgs}(MT)$ is a conflict free set of MT iff $\forall \mathcal{A}, \mathcal{B} \in CF$ $\mathcal{A} \nrightarrow \mathcal{B}$.*

Using these basic concepts we can define the admissible, complete, ground, preferred and stable extensions in the same way as classical abstract argumentation frameworks do [11].

Definition 14 (Acceptability Semantics). *Let MT be a MTAF and S a conflict-free set of MT. Then S is:*

- *An admissible extension of MT iff all arguments in S are acceptable wrt. S.*
- *A complete extension of MT iff $S = \{\mathcal{A} | \mathcal{A}$ is acceptable wrt. $S\}$ (each argument that is acceptable wrt. S is in S).*
- *The grounded extension of MT iff S is the smallest (wrt. set inclusion) complete extension of MT.*
- *A preferred extension of MT iff S is a maximal (wrt. set inclusion) admissible extension of MT.*
- *A stable extension of MT iff $\forall \mathcal{A} \in S$, $\forall \mathcal{B} \in \mathsf{MArgs}(MT) \setminus S$, $\mathcal{A} \rightarrow \mathcal{B}$.*

Given a MTAF and a semantics s, an argument $\mathcal{A}$ is skeptically accepted if it belongs to all s extensions; $\mathcal{A}$ is credulously accepted if it belong to a least one s extension; and $\mathcal{A}$ is rejected if does not belong to any s extension.

In the example we have been using along the paper, from the MTAF MT_B we will have a single complete, preferred and stable extension which is also the grounded extension $\{\mathcal{P}, \mathcal{S}, \mathcal{A}, \mathcal{B}\}$. Therefore, arguments $\mathcal{P}, \mathcal{S}, \mathcal{A}, \mathcal{B}$ will be accepted and arguments $\mathcal{R}, \mathcal{C}$ will be rejected.

Definitions 12, 13, and 14 correspond to those presented for classical argumentation frameworks (see [11]). Recall that a classical argumentation framework is represented through a set of arguments and a defeat relation among them. Thus, using the defeat relation and the arguments of a MTAF we can characterize a classical argumentation framework which accepts the same arguments as the MTAF under a given semantics.

Proposition 2. *For each MTAF* $MT = (S_T, \rightarrowtriangle\!\!\!\!-, \mathsf{TAtt}, >>)$ *there is a classical abstract argumentation framework* $AF = (\mathsf{MArgs}(MT), \rightarrow)$*, such that AF has the same set of extensions as MT under a given semantics.*

Proof. Straightforward from definitions 12, 13, and 14 □

Therefore, by Proposition 2 a MTAF will inherit all properties from classical argumentation frameworks (for more details about these properties refer to [11]). Moreover, it will be possible to evaluate the acceptability of arguments in a MTAF using the classical argumentation framework associated to this MTAF. Then, acceptability semantics are applied to this associated framework.

4 An Agent Programming Approach using Typed-Arguments

In this section we will present the reasoning model of an abstract agent programming language (APL) using a MTAF. The purpose is to show how the features of the typed argumentation systems can benefit the representation on concrete domains. The use of argumentation for deliberation in agent specification and agent programming is not new. Works such as [1, 18, 13] present important contributions showing how BDI agent mental components can be benefited with the use of argumentation for reasoning with conflicting information.

For this purpose, we will define a simple abstract agent programming language called SAPL. Like in agent programming [10] agents are represented through a set of mental components such as beliefs and goals, used in the deliberative process to determine what the agent will do. In SAPL the mental compoments are specified by two bases: the belief base, used to represent agent's knowledge about the world and its perceptions, and the goal base denoting situations of the world that the agent wants to realize or maintain. However, we will abstract from the concrete specification of these bases.

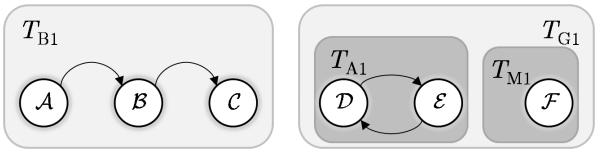
From these bases the agent will build arguments representing the information that can be tentatively inferred from each mental component. This information is tentative because there can be attacks among arguments, therefore, an argumentative acceptability analysis should be carried out. In particular, in this paper we are not concerned with how the arguments are built from a knowledge base (for specific approaches see [1, 18, 13]). Similarly, we will not concern how the internal conflicts and preferences of each base are obtained.

We will use different STAFs to represent each mental component: T_B for beliefs, T_G for goals, T_A for achievement goals, and T_M for maintenance goals. We will also assume that every argument supports a conclusion. Therefore, given an argument $\mathcal{A}$ the function $\mathsf{Conc}\ (\mathcal{A})$ will return the conclusion supported by $\mathcal{A}$. These STAFs will compose a MTAF whose acceptable arguments will determine what the agent believes and wants.

Note that we will use three different STAFs for representing goals. This is because we will distinguish two types of goals: achievement goals, characterized

with T_A; and maintenance goals, characterized with T_M. The STAF T_G characterizes all agent goals, and it will capture the general relation among goal arguments regardless of its type Then, since both T_A and T_M are goal types they will inherit from T_G.

Example 6. Let us consider the STAFs T_{B1}, T_{G1}, T_{A1} and T_{M1} built from the knowledge bases of an SAPL agent, where $T_{B1} = (\{\mathcal{A}, \mathcal{B}, \mathcal{C}\}, \{\mathcal{A} \rightsquigarrow \mathcal{B}, \mathcal{B} \rightsquigarrow \mathcal{C}\}, \emptyset)$, $T_{G1} = (\{\mathcal{D}, \mathcal{E}, \mathcal{F}\}, \{\mathcal{D} \rightsquigarrow \mathcal{E}, \mathcal{E} \rightsquigarrow \mathcal{D}\}, \emptyset)$, $T_{A1} = (\{\mathcal{D}, \mathcal{E}\}, \emptyset, \{\mathcal{D} \succeq \mathcal{E}\})$, and $T_{M1} = (\{\mathcal{F}\}, \emptyset, \emptyset)$. A graphical representation of this situation is included below.



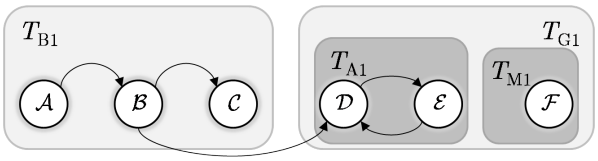
In particular, in the above depicted scenario the agent's arguments have that $\mathsf{Conc}(\mathcal{B}) = \mathsf{Conc}(\mathcal{D})$ and $\mathsf{Conc}(\mathcal{A}) = \mathsf{Conc}(\mathcal{F})$.

Achievement goals [17] are situations of the world that are not currently believed, which the agent wants to achieve. Following this concept, a belief argument supporting a conclusion will defeat an achievement argument supporting the same conclusion. Formally, we will have that $T_B >> T_A$ and that:

$$\mathsf{TAtt}(T_B, T_A) = \{(\mathcal{X}, \mathcal{Y}) | \mathcal{X} \in \mathsf{Args}(T_B), \mathcal{Y} \in \mathsf{Args}(T_A), \mathsf{Conc}(\mathcal{X}) = \mathsf{Conc}(\mathcal{Y})\}$$

To achieve these kind of goals the agent will perform a set of actions that modify the belief base. Mainly, since we are abstracting from the representation, these actions will add or remove belief arguments. Given an achievement goal argument $\mathcal{A}$ from T_A the functions $\mathsf{Add}\ (\mathcal{A})$ and $\mathsf{Remove}\ (\mathcal{A})$ will return the set of arguments that will be added and removed form the belief base.

Example 7. Continuing with Example 6, we have that $\mathsf{TAtt}(T_{B1}, T_{A1}) = \{(\mathcal{B}, \mathcal{D})\}$, since $\mathcal{D}$ and $\mathcal{B}$ share the same conclusion.



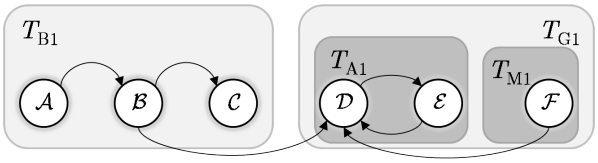
In addition, we will consider that $\mathsf{Remove}(\mathcal{D}) = \{\mathcal{A}\}$.

On the other hand, maintenance goals [17] are situations of the world that the agent wants to hold. Generally, these are important situations that the agent wants to maintain unaffected while performing actions. Therefore, an argument for a maintenance goal will defeat any achievement goal argument that jeopardizes the condition to maintain. That is, any achievement argument that removes the belief argument intended to maintain or introduces an attacker to that belief argument.[1] Formally, we will have that $T_M >> T_A$ and that:

[1] Note that this is a form of weak maintenance. Stronger notions of maintenance are out of the scope of this work.

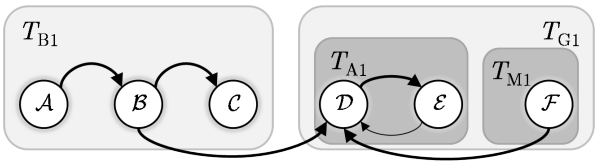
$$\mathsf{TAtt}(T_M, T_A) = \{(\mathcal{X}, \mathcal{Y}) | \mathcal{X} \in \mathsf{Args}(T_M), \mathcal{Y} \in \mathsf{Args}(T_A), \exists \mathcal{Z} \in \mathsf{Args}(T_B), \mathsf{Conc}(\mathcal{X}) = \mathsf{Conc}(\mathcal{Z}),\ \textit{either}\ \mathcal{Z} \in \mathsf{Remove}(\mathcal{Y})\ \textit{or}\ \mathcal{W} \in \mathsf{Add}(\mathcal{Y})\ \textit{such that}\ \mathcal{W} \rightsquigarrow \mathcal{Z}\}$$

Example 8. Continuing Example 7, we will have that $\mathsf{TAtt}(T_{M1}, T_{A1}) = \{(\mathcal{F}, \mathcal{D})\}$ because $\mathcal{D}$ removes the argument $\mathcal{A}$ whose conclusion is being maintained by the argument $\mathcal{F}$.



Therefore, the MTAF MT_{SAPL} used to represent the reasoning model of SAPL will be $MT_{SAPL} = (\{T_B, T_G, T_M, T_A\}, \{T_A \mathrel{-\!\!\triangleright\!\!-} T_G, T_M \mathrel{-\!\!\triangleright\!\!-} T_G\}, \mathsf{TAtt}, \{T_B >> T_A, T_M >> T_A\})$. Then, the SAPL agent will construct a MTAF MT_{SAPL} from its knowledge bases. Like in many APLs [10], these bases can change in each deliberative cycle, thus, modifying MT_{SAPL} from cycle to cycle.

Example 9. Considering the MTAF $MT_{SAPL1} = (\{T_{B1}, T_{G1}, T_{M1}, T_{A1}\}, \{T_{A1} \mathrel{-\!\!\triangleright\!\!-} T_{G1}, T_{M1} \mathrel{-\!\!\triangleright\!\!-} T_{G1}\}, \mathsf{TAtt}, \{T_{B1} >> T_{A1}, T_{M1} >> T_{A1}\})$ based on Example 8, we will obtain the following defeats:



Using MT_{SAPL}, the agent will be able to determine what currently, believes and wants. Basically, skeptically acceptable arguments under a given semantics of each STAF in MT_{SAPL} will determine what the agent believes and which are its goals. In our example, we will have that the agent believes in arguments $\mathcal{A}$ and $\mathcal{C}$, and its goals will be determined by the achievement argument $\mathcal{E}$ and the maintenance argument $\mathcal{F}$.

5 Conclusions and Related Work

In this work we initiated the study for a formalization of the argument type concept in the context of abstract argumentation frameworks. We introduced an approach where argument types are represented using abstract argumentation frameworks with preferences, and are related among each other through attacks, preferences and inheritance. These individual types together with their relations compose a Multi-Typed Argumentation Framework. We have decided to distinguish STAFs and MTAFs as separated entities because we believe that this separation provides clarity to the representation. Moreover, this model contributes to a better type modularization and an easier inheritance relation specification.

To define a defeat relation suited for these frameworks we had to consider relations among types and inside types. In particular, we had to pay special attention to the inheritance between types when determining preferences among arguments. Using the defeat relation we have shown how to determine which arguments will be acceptable using a classical acceptability semantics. As an instantiation of the presented approach we have shown how MTAFs can be used for the reasoning model of agent programming languages.

There are several things left open for future study. One of such topics is the consideration of the sub-argument relation among types that might lead to arguments of mixed types. Another interesting topic is how to integrate concepts of argumentation dynamics, since it would be useful for the agent programming domain. Finally, a formalization with a concrete rule based argumentation formalism such as [16, 12] will also be studied in order to take advantage of the concept of type.

In terms of the specific formalism we are presenting here, there is no other work addressing the formalization of the argument type concept explicitly. As we said in previous sections, several works in the literature use the idea of argument type, but their representation is fixed to the domain or problem they are addressing. In contrast, in our approach we provide a general and modular argumentation framework to explicitly represent multiple types. Therefore, here, types representation and types relation are not restricted to a specific domain.

The approach presented in [7] for value based argumentation can be related with our framework. In value based argumentation each argument can represent multiple values, and these values can be used to determine preferences among arguments. Their approach is presented using an abstract argumentation framework containing arguments, attacks, values, a function that maps arguments to values and an audience that determines a preference over these values. In sake of comparison, even if it is not the spirit of the values, it is possible to consider these values as argument types. Therefore, an argument will have certain types associated and evaluated with respect to them, like we do in our framework. However, that approach does not provides a general way to model internal conflicts and properties for arguments of a certain type, and it does not provide the tools for type specialization using inheritance. Finally, this comparison is clearly unfair since the objective of value base argumentation frameworks is completely different from ours.

Clearly, the examples and applications shown for the MTAFs can be modelled with other systems, like Dung's abstract framework or preference based argumentation frameworks. However, besides aiming to formalize the concept of argument type, MTAFs are a more concrete approach to represent argumentative or non-monotonic scenarios where information can be typified. Nevertheless, it would be very interesting to show the formal relation between the MTAFs and other argumentation frameworks such as [2, 7], besides Dungs abstract frameworks [11].

References

1. Amgoud, L., Devred, C., Lagasquie, M.: A constrained argumentation system for practical reasoning. In: Seventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS'08). pp. 429–436 (2008)
2. Amgoud, L., Cayrol, C.: Inferring from inconsistency in preference-based argumentation frameworks. Journal Autonomous Reasoning 29(2), 125–169 (2002)
3. Amgoud, L., Parsons, S., Perrussel, L.: An argumentation framework based on contextual preferences. In: In International Conference on Formal and Applied and Practical Reasoning (FAPR'2000). pp. 59–67 (2000)
4. Amgoud, L., Prade, H.: Handling threats, rewards, and explanatory arguments in a unified setting. International Journal of Intelligent Systems 20(12), 1195–1218 (2005)
5. Baroni, P., Giacomin, M.: Semantics of abstract argumentation systems plans. In: Rahwan, I., Simari, G. (eds.) Argumentation in Artificial Intelligence, pp. 25–44. Springer (2009)
6. Beierle, C., Freund, B., Kern-Isberner, G., Thimm, M.: Using defeasible logic programming for argumentation-based decision support in private law. In: Third International Conference on Computational Models of Argument (COMMA'10). Desenzano del Garda, Italy (September 2010)
7. Bench-Capon, T.J.M.: Persuasion in practical argument using value-based argumentation frameworks. Journal of Logic and Computation 13(3), 429–448 (2003)
8. Besnard, P., Hunter, A.: A logic-based theory of deductive arguments. Artificial Intelligence 128(1-2), 203–235 (2001)
9. Black, E., Hunter, A.: An inquiry dialogue system. Autonomous Agents and Multi-Agent Systems (JAAMAS 19(2), 173–209 (October 2009)
10. Bordini, R., Braubach, L., Dastani, M., Seghrouchni, A.E.F., Gomez-Sanz, J., Leite, J., O'Hare, G., Pokahr, A., Ricci, A.: A survey of programming languages and platforms for multi-agent systems. In: Informatica 30. pp. 33–44 (2006)
11. 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)
12. Garcia, A., Simari, G.: Defeasible logic programming: An argumentative approach. Theory and Practice of Logic Programming (TPLP) 4, 95–138 (2004)
13. Gottifredi, S., Garcia, A.J., Simari, G.R.: Query-based argumentation in agent programming. In: Twelfth Ibero-American Conference on Artificial Intelligence (IBERAMIA'2010). pp. 284–295 (2010)
14. Horty, J.F., Thomason, R.H., Touretzky, D.S.: A skeptical theory of inheritance in nonmonotonic semantic networks. Artificial Intelligence 42(2-3), 311–348 (1990)
15. Kaci, S., van der Torre, L.: Preference-based argumentation: Arguments supporting multiple values. International Journal of Approximate Reasoning 48(3), 730–751 (2008)
16. Prakken, H.: An abstract framework for argumentation with structured arguments. Argument and Computation 1(1), 93–124 (2010)
17. van Riemsdijk, M.B., Dastani, M., Winikoff, M.: Goals in agent systems: a unifying framework. In: Seventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS'08). pp. 713–720 (2008)
18. Rotstein, N.D., Garcia, A.J., Simari, G.R.: Reasoning from desires to intentions: A dialectical framework. In: AAAI Conference on Artificial Intelligence. pp. 136–141 (2007)

Selective Revision by Deductive Argumentation

Patrick Krümpelmann[1], Matthias Thimm[1],
Marcelo A. Falappa[2], Alejandro J. García[2],
Gabriele Kern-Isberner[1], and Guillermo R. Simari[2]

[1] Technische Universität Dortmund, Germany
[2] Universidad Nacional del Sur, Bahía Blanca, Argentina

Abstract. The *success* postulate of classic belief revision theory demands that after revising some beliefs with by information the new information is believed. However, this form of *prioritized* belief revision is not apt under many circumstances. Research in *non-prioritized* belief revision investigates forms of belief revision where *success* is not a desirable property. Herein, *selective revision* uses a two step approach, first applying a transformation function to decide if and which part of the new information shall be accepted, and second, incorporating the result using a prioritized revision operator. In this paper, we implement a transformation function by employing deductive argumentation to assess the value of new information. Hereby we obtain a non-prioritized revision operator that only accepts new information if believing in the information is justifiable with respect to the beliefs. By making use of previous results on selective revision we prove that our revision operator satisfies several desirable properties. We illustrate the use of the revision operator by means of examples and compare it with related work.

1 Introduction

Belief revision [4, 12] is concerned with changing beliefs in the light of new information. Usually, the beliefs of an agent are not static but change when new information is available. In order to be able to act reasonably in a changing environment the agent has to integrate new information and give up outdated beliefs. In particular, if the agent learns that some beliefs have been misleadingly assumed to be true its beliefs have to be *revised*. The research field of belief revision distinguishes between *prioritized* and *non-prioritized* belief revision. In prioritized belief revision [12] new information is always assumed to represent the most reliable and correct information available and revising the agent's beliefs by the new information is expected to result in believing the new information. This is a reasonable assumption for many imaginable situations and there are many technical challenges in realizing prioritized belief revision, cf. e. g. [12]. However, many circumstances demand that new information is not blindly accepted but weighted against the current beliefs. The field of non-prioritized belief revision [11] investigates change operations where revising some beliefs by new information may not result in believing the new information. Imagine a

multi-agent system where agents exchange information. In general, agents may be cooperative or competitive. Information that is passed from one agent to another may be intentionally wrong, mistakenly wrong, or correct. It is up to the receiver of the information to evaluate whether it should be integrated into the beliefs or not. In particular, in non-prioritized belief revision the satisfaction of the *success* postulate—which demands that new information is believed after revision—is not desirable. In [9] a specific class of non-prioritized belief revision operators is investigated. A *selective revision* is a two-step revision that consists of 1.) filtering new information using a *transformation function* and 2.) revising the beliefs with the result of the filtering in a prioritized way. In [9], no concrete implementations of the transformation function are given but several results are proven that show how specific properties for the transformation function and the *inner* prioritized revision translate to specific properties for the *outer* non-prioritized revision.

In this paper we propose a specific implementation of a transformation function that makes use of *deductive argumentation* [2]. A deductive argumentation theory is a set of propositional sentences and an argument for some sentence ϕ is a minimal proof for ϕ. If the theory is inconsistent there may also be proofs for the complement of a sentence $\neg\phi$ and in order to decide whether ϕ or $\neg\phi$ is to be believed, an argumentative evaluation is performed that compares arguments with counterarguments. We use the framework of [2] to implement a transformation function for selective revision that decides for each individual piece of information whether to accept it for revision or not, based on its argumentative evaluation. In particular, we consider the case that revision is to be performed based on a set of pieces of information instead of just a single piece of information. By doing so, we allow new information to contain arguments. As a result, an agent decides whether to accept some new information on the basis of its own evaluation of the information and the arguments that may be contained in this information. Consider the following example.

Example 1. Imagine the agent *Anna* wants to spend her holidays on Hawaii. She is aware of the fact that there has been some volcano activity on Hawaii recently but is convinced there is no immediate danger. Anna's boss *Bob* doesn't want Anna to go on vacation at this time of the year and tells her that she has to do some work here and should not go to Hawaii. However, Anna wants to go surfing and this is a much better argument for her to go to Hawaii instead of staying at work. As a consequence she rejects Bob's argument to stay and does not revise her beliefs. Consider now that *Carl*, a good friend of *Anna*, is a vulcanologist and tells Anna that there is actually an immediate danger of an eruption. Anna knows that Carl is clearly knowledgable about volcanos and finds his argument convincing. Consequently, she accepts the new information and revises her beliefs accordingly. ■

In the previous example the decisions of the agent Anna resulted in either accepting or rejecting the new information completely. However, it may also be the case that some of the new information is accepted and some is rejected. Consider the following example.

Example 2. Imagine Bob tells Anna that she has to stay for work because all her colleagues are having a vacation at the same time and she has to fill in for them. Suppose Anna knows that there is no work to do during her planned vacation as all clients of her company are on vacation as well. Then Anna would reject the conclusion of Bob's argument that she has to stay, but might very well accept that all her colleagues will be on vacation as well. ■

In this paper we develop an approach for selective revision that is capable of deciding whether to accept, reject, or partially accept some new information, based on deductive argumentation. In order to do so we also extend the notions of selective revision to the problem of multiple base revision, i. e., the problem of revising a belief base (instead of a belief set) by a set of sentences.

The rest of this paper is organized as follows. In Section 2 we introduce some necessary technical preliminaries. We go on in Section 3 with providing an overview on the notions of belief revision and extending the approach of selective revision to selective multiple base revision. We continue in Section 4 with presenting the framework of deductive argumentation. In Section 5 we propose our implementation of selective multiple base revision via deductive argumentation and investigate its properties. In Section 6 we review some related work and in Section 7 we conclude.

2 Preliminaries

In this paper we suppose that the beliefs of an agent are given in the form of propositional sentences. Let At be a propositional signature, i. e. a set of propositional atoms. Let $\mathcal{L}(\mathsf{At})$ be the corresponding propositional language generated by the atoms in At and the connectives $\wedge$ (*and*), $\vee$ (*or*), $\Rightarrow$ (*implication*), and $\neg$ (*negation*). As a notational convenience we assume some arbitrary total order $\succ$ on the elements of $\mathcal{L}(\mathsf{At})$ which is used to enumerate elements of each finite $\Phi \subseteq \mathcal{L}(\mathsf{At})$ in a unique way, cf. [2]. For a finite subset $\Phi \subseteq \mathcal{L}(\mathsf{At})$ the *canonical enumeration* of Φ is the vector $\langle \phi_1, \ldots, \phi_n \rangle$ such that $\{\phi_1, \ldots, \phi_n\} = \Phi$ and $\phi_i \succ \phi_j$ for every $i < j$ with $i, j = 1, \ldots, n$. As $\succ$ is total the canonical enumeration of every finite subset $\Phi \subseteq \mathcal{L}(\mathsf{At})$ is uniquely defined.

We use the operator $\vdash$ to denote classical entailment, i. e., for sets of propositional sentences $\Phi_1, \Phi_2 \subseteq \mathcal{L}(\mathsf{At})$ we say that Φ_2 *follows* from Φ_1, denoted by $\Phi_1 \vdash \Phi_2$, if and only if Φ_2 is entailed by Φ_1 in the classical logical sense. For sentences $\phi, \phi' \in \mathcal{L}(\mathsf{At})$ we write $\phi \vdash \phi'$ instead of $\{\phi\} \vdash \{\phi'\}$. We define the deductive closure $Cn(\cdot)$ of a set of sentences Φ as $Cn(\Phi) = \{\phi \in \mathcal{L}(\mathsf{At}) \mid \Phi \vdash \phi\}$. Two sets of sentences $\Phi, \Phi' \subseteq \mathcal{L}(\mathsf{At})$ are *equivalent*, denoted by $\Phi \equiv^p \Phi'$, if and only if it holds that $\Phi \vdash \Phi'$ and $\Phi' \vdash \Phi$. We also use the equivalence relation $\cong^p$ which is defined as $\Phi \cong^p \Phi'$ if and only if there is a bijection $\sigma : \Phi \to \Phi'$ such that for every $\phi \in \Phi$ it holds that $\phi \equiv^p \sigma(\phi)$. This means that $\Phi \cong^p \Phi'$ if Φ and Φ' are *element-wise* equivalent. Note that $\Phi \cong^p \Phi'$ implies $\Phi \equiv^p \Phi'$ but not vice versa. In particular, it holds that e. g $\{a \wedge b\} \equiv^p \{a, b\}$ but $\{a \wedge b\} \ncong^p \{a, b\}$. For sentences $\phi, \phi' \in \mathcal{L}(\mathsf{At})$ we write $\phi \equiv \phi'$ instead of $\{\phi\} \equiv \{\phi'\}$ if $\equiv \in \{\equiv^p, \cong^p\}$. If $\Phi \vdash \bot$ we say that Φ is *inconsistent*.

For a set S let $\mathfrak{P}(S)$ denote the power set of S, i. e. the set of all subsets of S. For a set S let $\mathfrak{PP}(S)$ denote the set of multi-sets of S, i. e. the set of all subsets of S where an element may occur more than once. To distinguish sets from multi-sets we use brackets "$\langle$" and "$\rangle$" for the latter.

3 Selective Multiple Base Revision

The field of belief revision is concerned with the change of beliefs when more recent or more reliable information is at hand. The most important description of properties of *prioritized* belief change operators is given by Alchourrón, Gärdenfors and Makinson in their seminal paper [4]. The usual framework for representing beliefs considered for belief revision is that of *belief sets* which are revised by a single sentence. A *belief set* S is a subset of $\mathcal{L}(\mathsf{At})$ that is deductively closed, i. e., $S = Cn(S)$. Working with belief sets in practice is unmanageable due to their infinite size. The more practical representation form are *belief bases* which are finite sets of sentences. These also come with the advantage of making it possible to differentiate between explicit and inferred beliefs, cf. [12]. In this work we consider the problem of *multiple base revision*. That is, we employ belief bases for knowledge representation and we consider revising a belief base by a set of sentences, cf. the notion of *multiple revision* in [12].

Let $\mathcal{K} \subseteq \mathcal{L}(\mathsf{At})$ be a belief base, $\Phi \subseteq \mathcal{L}(\mathsf{At})$ be some set of sentences, and consider the problem of changing $\mathcal{K}$ in order to entail Φ. If $\mathcal{K} \cup \Phi$ is consistent then there is no need for contracting the existing beliefs and the problem can be solved via *expansion* $\mathcal{K} + \Phi$ which is characterized via $\mathcal{K} \cup \Phi$. If $\mathcal{K} \cup \Phi$ is inconsistent, conflicts arising from the addition of Φ to $\mathcal{K}$ have to be resolved. In general, this means that some of the current beliefs have to be given up in order to come up with a consistent belief base. The AGM framework [4] proposes several basic postulates a revision operator should obey. As we consider belief bases for knowledge representation we start with the corresponding postulates for belief base revision [12] adapted to revision by sets of sentences [8]. Let $*$ be a multiple base revision operator—i. e., if $\mathcal{K}$ and Φ are sets of sentences so is $\mathcal{K} * \Phi$—and consider the following postulates.

Success. $\mathcal{K} * \Phi \vdash \Phi$.
Inclusion. $\mathcal{K} * \Phi \subseteq \mathcal{K} + \Phi$.
Vacuity. If $\mathcal{K} \cup \Phi \nvdash \bot$ then $\mathcal{K} + \Phi \subseteq \mathcal{K} * \Phi$.
Consistency. If Φ is consistent then $\mathcal{K} * \Phi$ is consistent.
Relevance. If $\alpha \in (\mathcal{K} \cup \Phi) \setminus (K * \Phi)$ then there is a set H such that $\mathcal{K} * \Phi \subseteq H \subseteq K \cup \Phi$ and H is consistent but $H \cup \{\alpha\}$ is inconsistent.

Another important property for the framework of [4] is *extensionality* which can be phrased for multiple base revision as follows.

Extensionality. If $\Phi \equiv^p \Psi$, then $\mathcal{K} * \Phi \equiv^p \mathcal{K} * \Psi$.

The above property is usually not considered for the problem of base revision as base revision is motivated by observing syntax and not (only) semantic contents. In particular, for the problem of multiple base revision, satisfaction of *extensionality* imposes that $\mathcal{K} * \{a, b\} \equiv^p \mathcal{K} * \{a \wedge b\}$ as $\{a, b\} \equiv^p \{a \wedge b\}$. Identifying the "comma"-operator with the logical "AND"-operator is not always a reasonable thing to do, see e. g. [5] for a discussion. However, we consider the following weakened form of *extensionality*.

Weak Extensionality. If $\Phi \cong^p \Phi'$ then $\mathcal{K} * \Phi \equiv^p \mathcal{K} * \Phi'$.

The property *weak extensionality* only demands that the outcomes of the revisions $\mathcal{K} * \Phi$ and $\mathcal{K} * \Phi'$ are equivalent if Φ and Φ' are element-wise equivalent.

Definition 1. *A revision operator $*$ is called a* prioritized multiple base revision operator *if $*$ satisfies* success*,* inclusion*,* vacuity*,* consistency*,* relevance*, and* weak extensionality*.*

For non-prioritized multiple base revision the properties *inclusion*, *vacuity*, *consistency*, *relevance*, and *weak extensionality* can also be regarded as desirable. This is not the case for *success* is general but we can replace *success* by weakened versions, cf. [11]. We denote with $\circ$ a non-prioritized belief revision operator, i. e., $\mathcal{K} \circ \Phi$ is the non-prioritized revision of $\mathcal{K}$ by Φ. Then consider the following properties for $\circ$, cf. [9].

Weak Success. If $\mathcal{K} \cup \Phi \nvdash \bot$ then $\mathcal{K} \circ \Phi \vdash \Phi$.
Consistent Expansion. If $\mathcal{K} \not\subseteq \mathcal{K} \circ \Phi$ then $\mathcal{K} \cup (\mathcal{K} \circ \Phi) \vdash \bot$.

Note that *weak success* follows from *vacuity*, and *consistent expansion* follows from *vacuity* and *success*, cf. [9].

Definition 2. *A revision operator $\circ$ is called* non-prioritized multiple base revision operator *if $\circ$ satisfies* inclusion*,* consistency*,* weak extensionality*,* weak success*, and* consistent expansion*.*

We do not require *relevance* to be satisfied by non-prioritized multiple base revisions as it is hardly achievable in the context of selective revision, see below. For the following, bear in mind that the main difference between a prioritized multiple base revision operator $*$ and a non-prioritized multiple base revision operator $\circ$ is that $\mathcal{K} * \Phi \vdash \Phi$ is required but $\mathcal{K} \circ \Phi \vdash \Phi$ is not.

A specific approach to non-prioritized belief revision is *selective revision* [9]. There, the problem of revising a belief set S with a single sentence α is realized by applying a *transformation function* f to α, obtaining a new sentence α', and then revising S by α' in a prioritized way. The transformation function f is supposed to determine whether α should be accepted as a whole or whether it should be somewhat weakened. We adopt the notions of [9] for the problem of *selective multiple belief base revision* and still consider the problem of revising

a belief base $\mathcal{K}$ by some set Φ of sentences. Following the ideas of [9] we define the selective multiple base revision $\mathcal{K} \circ \Phi$ via

$$\mathcal{K} \circ \Phi = \mathcal{K} * f_{\mathcal{K}}(\Phi) \tag{1}$$

with a transformation function $f_{\mathcal{K}} : \mathfrak{P}(\mathcal{L}(\mathsf{At})) \to \mathfrak{P}(\mathcal{L}(\mathsf{At}))$ and some (prioritized) multiple base revision $*$. In [9] several properties for transformation functions in the context of belief set revision are discussed. We rephrase some of them here slightly to fit the framework of multiple base revision. Let $\mathcal{K} \subseteq \mathcal{L}(\mathsf{At})$ be consistent and let $\Phi, \Phi' \subseteq \mathcal{L}(\mathsf{At})$.

Inclusion. $f_{\mathcal{K}}(\Phi) \subseteq \Phi$
Weak Inclusion. If $\mathcal{K} \cup \Phi$ is consistent then $f_{\mathcal{K}}(\Phi) \subseteq \Phi$
Extensionality. If $\Phi \equiv^p \Phi'$ then $f_{\mathcal{K}}(\Phi) \equiv^p f_{\mathcal{K}}(\Phi')$
Consistency Preservation. If Φ is consistent then $f_{\mathcal{K}}(\Phi)$ is consistent
Consistency. $f_{\mathcal{K}}(\Phi)$ is consistent
Maximality. $f_{\mathcal{K}}(\Phi) = \Phi$
Weak Maximality. If $\mathcal{K} \cup \Phi$ is consistent then $f_{\mathcal{K}}(\Phi) = \Phi$

We also consider the following novel property.

Weak Extensionality. If $\Phi \cong^p \Phi'$ then $f_{\mathcal{K}}(\Phi) \cong^p f_{\mathcal{K}}(\Phi')$

Not all of the above properties may be desirable for a transformation function that is to be used for selective revision. For example, the property *maximality* states that $f_{\mathcal{K}}$ should not modify the set Φ. Satisfaction of this property makes (1) equivalent to $\mathcal{K} * \Phi$. As $*$ is meant to be a prioritized revision function we lose the possibility for non-prioritized revision.

Note that for *weak extensionality* we demand $f_{\mathcal{K}}(\Phi)$ and $f_{\mathcal{K}}(\Phi')$ to be elementwise equivalent instead of just equivalent (in contrast to the property *weak extensionality* for revision). We do this because $f_{\mathcal{K}}$ is supposed to be applied in the context of base revision which is sensitive to syntactic variants. We introduce the postulate *weak extensionality* for transformation functions with the same motivation as we do for multiple base revision. However, for the case of transformation functions the problem with satisfaction of *extensionality* is more apparent. Consider again $\Phi = \{a, b\}$ and $\Phi' = \{a \wedge b\}$. It follows that $\Phi \equiv^p \Phi'$ and if $f_{\mathcal{K}}$ satisfies *extensionality* this results in $f_{\mathcal{K}}(\{a, b\}) \equiv^p f_{\mathcal{K}}(\{a \wedge b\})$. If $f_{\mathcal{K}}$ also satisfies *inclusion* it follows that $f_{\mathcal{K}}(\{a \wedge b\}) \in \{\emptyset, \{a \wedge b\}\}$ and therefore $f_{\mathcal{K}}(\{a, b\}) \in \{\emptyset, \{a, b\}\}$. In general, if $f_{\mathcal{K}}$ satisfies both *inclusion* and *extensionality* it follows that either $f_{\mathcal{K}}(\Phi) = \emptyset$ or $f_{\mathcal{K}}(\Phi) = \Phi$ for every $\Phi \subseteq \mathcal{L}(\mathsf{At})$ (as Φ is equivalent to a Φ' that consists of a single formula that is the conjunction of the formulas in Φ and $f_{\mathcal{K}}(\Phi') = \emptyset$ or $f_{\mathcal{K}}(\Phi') = \Phi'$ due to *inclusion*). As we are interested in a more graded approach to belief revision we want to be able to accept or reject specific pieces of Φ and not just Φ as a whole. Consequently, we consider *weak extensionality* as a desirable property instead of *extensionality*. Note that *extensionality* implies *weak extensionality* as $\Phi \cong^p \Phi'$ implies $\Phi \equiv^p \Phi'$.

In [9] several representation theorems are given that characterize non-prioritized belief revision by selective revision via (1) and specific properties of $*$

and $f_{\mathcal{K}}$. In particular, it is shown that a reasonable non-prioritized belief revision operator $\circ$ can be characterized by an AGM revision $*$ and a transformation function $f_{\mathcal{K}}$ that satisfies *extensionality*, *consistency preservation*, and *weak maximality*. Note, however, that [9] deals with the problem of revising a belief set by a single sentence. Nonetheless, we can carry over the results of [9] to the problem of multiple base revision and obtain the following result.

Proposition 1. *Let $*$ be a prioritized multiple base revision operator and let $f_{\mathcal{K}}$ satisfy* inclusion*,* weak extensionality*,* consistency preservation*, and* weak maximality*. Then $\circ$ defined via (1) is a non-prioritized multiple base revision operator.*

Proof. We have to show that $\circ$ satisfies inclusion*,* consistency*,* weak extensionality*,* weak success*, and* consistent expansion*.*

Inclusion. *It holds that $f_{\mathcal{K}}(\Phi) \subseteq \Phi$ as $f_{\mathcal{K}}$ satisfies* inclusion*. Also, $*$ satisfies* inclusion *and it follows $\mathcal{K} * f_{\mathcal{K}}(\Phi) \subseteq \mathcal{K} \cup f_{\mathcal{K}}(\Phi) \subseteq \mathcal{K} \cup \Phi$.*

Consistency. *If Φ is consistent so is $f_{\mathcal{K}}(\Phi)$ as $f_{\mathcal{K}}$ satisfies* consistency preservation*. As $*$ satisfies* consistency *it follows that $\mathcal{K} * f_{\mathcal{K}}(\Phi)$ is consistent.*

Weak Extensionality. *If $\Phi \cong^p \Phi'$ then $f_{\mathcal{K}}(\Phi) \cong^p f_{\mathcal{K}}(\Phi')$ as $f_{\mathcal{K}}$ satisfies* weak extensionality*. It follows that $\mathcal{K} * f_{\mathcal{K}}(\Phi) \equiv^p \mathcal{K} * f_{\mathcal{K}}(\Phi')$ as $*$ satisfies* weak extensionality*.*

Weak Success. *If $\mathcal{K} \cup \Phi$ is consistent it follows that $f_{\mathcal{K}}(\Phi) = \Phi$ as $f_{\mathcal{K}}$ satisfies* weak maximality*. As $*$ satisfies* vacuity *it follows $\mathcal{K} + \Phi \subseteq \mathcal{K} * f_{\mathcal{K}}(\Phi)$. Hence, $\circ$ satisfies vacuity as well and therefore* weak success*.*

Consistent Expansion. *Suppose $\mathcal{K} \not\subseteq \mathcal{K} * f_{\mathcal{K}}(\Phi)$. Note that $*$ satisfies* consistent expansion *as $*$ satisfies* vacuity *and* success*, cf. [9]. It follows that $\mathcal{K} \cup \{\mathcal{K} * f_{\mathcal{K}}(\Phi)\}$ is inconsistent.* □

Note that *relevance* does not hold for $\mathcal{K} \circ \Phi$ defined via (1) in general. Consider for example the transformation function $f^0_{\mathcal{K}}$ defined via $f^0_{\mathcal{K}}(\Phi) = \Phi$ if $\mathcal{K} \cup \Phi$ is consistent and $f^0_{\mathcal{K}}(\Phi) = \emptyset$ otherwise. Then $f^0_{\mathcal{K}}$ satisfies all properties for transformation functions except *maximality*. But it is easy to see that $\mathcal{K} \circ \Phi$ defined via (1) using $f^0_{\mathcal{K}}$ and a prioritized multiple base revision operator $*$ fails to satisfy *relevance*. We leave it to future work to investigate further properties for transformation functions that may enable *relevance* to hold in general.

In the following we aim at implementing a selective multiple base revision using deductive argumentation and go on with introducing the latter.

4 Deductive Argumentation

Argumentation frameworks [1] allow for reasoning with inconsistent information based on the notions of arguments, counterarguments and their relationships. Since the seminal paper [6] interest has grown in research in computational models for argumentation that allow for a coherent procedure for consistent reasoning in the presence of inconsistency. In this paper we use the framework of *deductive argumentation* as proposed by Besnard and Hunter [2]. This framework

bases on classical propositional logic and is therefore apt for our aim to use argumentation to realize a transformation function f. The central notion of the framework of deductive argumentation is that of an *argument*.

Definition 3 (Argument). *Let $\Phi \subseteq \mathcal{L}(\mathsf{At})$ be a set of sentences. An* argument *$\mathcal{A}$ for a sentence $\alpha \in \mathcal{L}(\mathsf{At})$ in Φ is a tuple $\mathcal{A} = \langle \Psi, \alpha \rangle$ with $\Psi \subseteq \Phi$ that satisfies 1.) $\Psi \nvdash \bot$, 2.) $\Psi \vdash \alpha$, and 3.) there is no $\Psi' \subsetneq \Psi$ with $\Psi' \vdash \alpha$. For an argument $\mathcal{A} = \langle \Psi, \alpha \rangle$ we say that α is the* claim *of $\mathcal{A}$ and Ψ is the* support *of $\mathcal{A}$.*

Hence, an argument $\mathcal{A} = \langle \Psi, \alpha \rangle$ for α is a minimal proof for entailing α. Given a set $\Phi \subseteq \mathcal{L}(\mathsf{At})$ of sentences there may be multiple arguments for α. As in [2] we are interested in arguments that are most cautious.

Definition 4 (Conservativeness). *An argument $\mathcal{A} = \langle \Psi, \alpha \rangle$ is more* conservative *than an argument $\mathcal{B} = \langle \Phi, \beta \rangle$ if and only if $\Psi \subseteq \Phi$ and $\beta \vdash \alpha$.*

In other words, an $\mathcal{A}$ is more conservative than an argument $\mathcal{B}$ if $\mathcal{B}$ has a smaller support (with respect to set inclusion) and a more general conclusion. An argument $\mathcal{A}$ is *strictly more conservative* than an argument $\mathcal{B}$ if and only if $\mathcal{A}$ is more conservative than $\mathcal{B}$ but $\mathcal{B}$ is not more conservative than $\mathcal{A}$. If $\Phi \subseteq \mathcal{L}(\mathsf{At})$ is inconsistent there are arguments with contradictory claims.

Definition 5 (Undercut). *An argument $\mathcal{A} = \langle \Psi, \alpha \rangle$ is an* undercut *for an argument $\mathcal{B} = \langle \Phi, \beta \rangle$ if and only if $\alpha = \neg(\phi_1 \wedge \ldots \wedge \phi_n)$ for some $\phi_1, \ldots, \phi_n \subseteq \Phi$.*

If $\mathcal{A}$ is an undercut for $\mathcal{B}$ then we also say that $\mathcal{A}$ *attacks* $\mathcal{B}$. In order to consider only those undercuts for an argument that are most general we restrain the notion of undercut as follows.

Definition 6 (Maximally conservative undercut). *An argument $\mathcal{A} = \langle \Psi, \alpha \rangle$ is a* maximally conservative undercut *for an argument $\mathcal{B} = \langle \Phi, \beta \rangle$ if and only if $\mathcal{A}$ is an undercut of $\mathcal{B}$ and there is no undercut $\mathcal{A}'$ for $\mathcal{B}$ that is strictly more conservative than $\mathcal{A}$.*

Definition 7 (Canonical undercut). *An argument $\mathcal{A} = \langle \Psi, \neg(\phi_1 \wedge \ldots \wedge \phi_n) \rangle$ is a* canonical undercut *for an argument $\mathcal{B} = \langle \Phi, \beta \rangle$ if and only if $\mathcal{A}$ is a maximally conservative undercut for $\mathcal{B}$ and $\langle \phi_1, \ldots, \phi_n \rangle$ is the canonical enumeration of Φ.*

It can be shown that it suffices to consider only the canonical undercuts for an argument in order to come up with a reasonable argumentative evaluation of some claim α [2]. Having an undercut $\mathcal{B}$ for an argument $\mathcal{A}$ there may also be an undercut $\mathcal{C}$ for $\mathcal{B}$ which *defends* $\mathcal{A}$. In order to give a proper evaluation of some argument $\mathcal{A}$ we have to consider all undercuts for its undercuts as well, and so on. This leads to the notion of an *argument tree*.

Definition 8 (Argument tree). *Let $\alpha \in \mathcal{L}(\mathsf{At})$ be some sentence and let $\Phi \subseteq \mathcal{L}(\mathsf{At})$ be a set of sentences. An* argument tree *$\tau_\Phi(\alpha)$ for α in Φ is a tree where the nodes are arguments and that satisfies*

1. *the root is an argument for* α *in* Φ,
2. *for every path* $[\langle\Phi_1, \alpha_1\rangle, \ldots, \langle\Phi_n, \alpha_n\rangle]$ *in* $\tau_\Phi(\alpha)$ *it holds that* $\Phi_n \nsubseteq \Phi_1 \cup \ldots \cup \Phi_{n-1}$, *and*
3. *the children* $\mathcal{B}_1, \ldots, \mathcal{B}_m$ *of a node* $\mathcal{A}$ *consist of all canonical undercuts for* $\mathcal{A}$ *that obey 2.).*

Let $\mathcal{T}(\mathsf{At})$ *be the set of all argument trees.*

An argument tree is a concise representation of the relationships between different arguments that favor or reject some argument $\mathcal{A}$. In order to evaluate whether a claim α can be justified we have to consider all argument trees for α and all argument trees for $\neg\alpha$. For an argument tree τ let $\mathsf{root}(\tau)$ denote the root node of τ. Furthermore, for a node $\mathcal{A} \in \tau$ let $\mathsf{ch}_\tau(\mathcal{A})$ denote the children of $\mathcal{A}$ in τ and $\mathsf{ch}_\tau^{\mathcal{T}}(\mathcal{A})$ denote the set of sub-trees rooted at a child of $\mathcal{A}$.

Definition 9 (Argument structure). *Let* $\alpha \in \mathcal{L}(\mathsf{At})$ *be some sentence and let* $\Phi \subseteq \mathcal{L}(\mathsf{At})$ *be a set of sentences. The* argument structure $\Gamma_\Phi(\alpha)$ *for* α *with respect to* Φ *is the tuple* $\Gamma_\Phi(\alpha) = (\mathcal{P}, \mathcal{C})$ *such that* $\mathcal{P}$ *is the set of argument trees for* α *in* Φ *and* $\mathcal{C}$ *is the set of arguments trees for* $\neg\alpha$ *in* Φ.

The argument structure $\Gamma_\Phi(\alpha)$ of a $\alpha \in \mathcal{L}(\mathsf{At})$ gives a complete picture of the reasons for and against α. In order to evaluate those reasons we use the following notation, cf. [2].

Definition 10 (Categorizer). *A* categorizer γ *is a function* $\gamma : \mathcal{T}(\mathsf{At}) \to \mathbb{R}$.

A categorizer is meant to assign a value to an argument tree τ depending on how strongly this argument tree favors the root argument. In particular, the larger the value of $\gamma(\tau)$ the better the justification in believing in the claim of the root argument. For an argument structure $\Gamma_\Phi(\alpha) = (\{\tau_1^p, \ldots, \tau_n^p\}, \{\tau_1^c, \ldots, \tau_m^c\})$ and a categorizer γ we abbreviate

$$\gamma(\Gamma_\Phi(\alpha)) = (\langle\gamma(\tau_1^p), \ldots, \gamma(\tau_n^p)\rangle, \langle\gamma(\tau_1^c), \ldots, \gamma(\tau_m^c)\rangle) \in \mathfrak{PP}(\mathbb{R}) \times \mathfrak{PP}(\mathbb{R})\,.$$

Definition 11 (Accumulator). *An* accumulator κ *is a function* $\kappa : \mathfrak{PP}(\mathbb{R}) \times \mathfrak{PP}(\mathbb{R}) \to \mathbb{R}$.

An accumulator is meant to evaluate the categorization of argument trees for or against some sentence α. We say that a set of sentences $\Phi \subseteq \mathcal{L}(\mathsf{At})$ *accepts* a sentence α with respect to a categorizer γ and an accumulator κ, denoted by $\Phi \mathrel{|\!\sim}_{\kappa,\gamma} \alpha$ if and and only if

$$\kappa(\gamma(\Gamma_\Phi(\alpha))) > 0$$

A set of sentences $\Phi \subseteq \mathcal{L}(\mathsf{At})$ *rejects* a sentence α with respect to a categorizer γ and an accumulator κ, denoted by $\Phi \not|\!\sim_{\kappa,\gamma} \alpha$ if and and only if

$$\kappa(\gamma(\Gamma_\Phi(\alpha))) < 0$$

If Φ neither accepts nor rejects α with respect to γ and κ we say that Φ is *undecided* about α with respect to γ and κ. Some simple instances of categorizers and accumulators are as follows.

Example 3. Let τ be some argument tree. The classical evaluation of an argument tree—as e. g. employed in *Defeasible Logic Programming* [10]—is that each leaf of the tree is considered "undefeated" and an inner node is "undefeated" if all its children are "defeated" and "defeated" if there is at least one child that is "undefeated". This intuition can be formalized by defining the *classical categorizer* γ_0 recursively via

$$\gamma_0(\tau) = \begin{cases} 1 & \text{if } \mathsf{ch}_\tau(\mathsf{root}(\tau)) = \emptyset \\ 1 - \max\{\gamma_0(\tau') \mid \tau' \in \mathsf{ch}_\tau^{\mathcal{T}}(\mathsf{root}(\tau))\} & \text{otherwise} \end{cases}$$

Furthermore, a simple accumulator κ_0 can be defined via

$$\kappa_0(\langle N_1, \ldots, N_n\rangle, \langle M_1, \ldots, M_m\rangle) = N_1 + \ldots + N_n - M_1 - \ldots - M_m\,.$$

For example, a set of sentences $\Phi \subseteq \mathcal{L}(\mathsf{At})$ accepts a sentence α with respect to γ_0 and κ_0 if and only if there are more argument trees for α where the root argument is undefeated than argument trees for $\neg\alpha$ where the root argument is undefeated. ■

More examples of categorizers and accumulators can be found in [2]. Using those notions we are able to state for every sentence $\phi \in \Phi$ whether ϕ is accepted in Φ or not, depending on the arguments that favor α and those that reject α.

5 Selective Revision by Deductive Argumentation

Using the deductive argumentation framework presented in the previous section one is able to decide for each sentence $\alpha \in \Phi$ whether α is justifiable with respect to Φ. Note that the framework of deductive argumentation heavily depends on the actual instances of categorizer and accumulator. In the following we only consider categorizer and accumulator that comply with the following minimal requirements.

Definition 12 (Well-behaving categorizer). *A categorizer γ is called* well-behaving *if $\gamma(\tau) > \gamma(\tau')$ whenever τ consists only of one single node and τ' consists of at least two nodes.*

In other words, a categorizer γ is well-behaving if the argument tree that has no undercuts for its root is considered the best justification for the root.

Definition 13 (Well-behaving accumulator). *An accumulator κ is called* well-behaving *if and only if $\kappa((\mathcal{P}, \mathcal{C})) > 0$ whenever $\mathcal{P} \neq \emptyset$ and $\mathcal{C} = \emptyset$.*

This means, that if there are no arguments against a claim α and at least one argument for α in Φ then α should be accepted in Φ. Note that both γ_0 and κ_0 are well-behaving as well as all categorizers and accumulators considered in [2]. Note further that if Φ is consistent then every sentence $\alpha \in \Phi$ is accepted by Φ with respect to every well-behaving categorizer and well-behaving accumulator.

Let $\mathcal{K} \subseteq \mathcal{L}(\mathsf{At})$ be a consistent set of sentences, and let γ be some well-behaving categorizer and κ be some well-behaving accumulator. We consider again a selective revision $\circ$ of the form (1). In order to determine the outcome of the non-prioritized revision $\mathcal{K} \circ \Phi$ for some $\Phi \subseteq \mathcal{L}(\mathsf{At})$ we implement a transformation function f that checks for every sentence $\alpha \in \Phi$ whether α is accepted in $\mathcal{K} \cup \Phi$. Note that although $\mathcal{K}$ is consistent the union $\mathcal{K} \cup \Phi$ is not necessarily consistent which gives rise to an argumentative evaluation. In the following, we consider *two* different transformation functions based on deductive argumentation. We define the *skeptical transformation function* $\mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}$ via

$$\mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}(\Phi) = \{\alpha \in \Phi \mid \mathcal{K} \cup \Phi \mathrel{|\!\sim}_{\kappa,\gamma} \alpha\}$$

for every $\Phi \subseteq \mathcal{L}(\mathsf{At})$ and the *credulous transformation function* $\mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}$ via

$$\mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}(\Phi) = \{\alpha \in \Phi \mid \mathcal{K} \cup \Phi \not\mathrel{|\!\sim}_{\kappa,\gamma} \neg\alpha\}$$

for every $\Phi \subseteq \mathcal{L}(\mathsf{At})$. In other words, the value of $\mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}(\Phi)$ consists of those sentences of Φ that are accepted in $\mathcal{K} \cup \Phi$ and the value of $\mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}(\Phi)$ consists of those sentences of Φ that are not rejected in $\mathcal{K} \cup \Phi$. There is a subtle difference in the behavior of those two transformation functions as the following example shows.

Example 4. Let $\mathcal{K}_1 = \{a\}$ and $\Phi_1 = \{\neg a\}$. Note that there is exactly one argument tree τ_1 for $\neg a$ and one argument tree τ_2 for a in $\mathcal{K}_1 \cup \Phi$. In τ_1 the root is the argument $\mathcal{A} = \langle\{\neg a\}, \neg a\rangle$ which has the single canonical undercut $\mathcal{B} = \langle\{a\}, a\rangle$. In τ_2 the situation is reversed and the root of τ_2 is the argument $\mathcal{B}$ which has the single canonical undercut $\mathcal{A}$. Therefore, the argument structure for $\neg a$ is given via $\Gamma_{\mathcal{K}\cup\Phi}(\neg a) = (\{\tau_1\}, \{\tau_2\})$. It follows that $\gamma_0(\tau_1) = \gamma_0(\tau_2) = 0$ and $\kappa_0(\gamma_0(\Gamma_{\mathcal{K}\cup\Phi}(a))) = \kappa_0(\langle 0, 0\rangle) = 0$. It follows that $\mathcal{K} \cup \Phi$ is undecided about both $\neg a$ and a. Consequently, it follows that

$$\mathsf{S}_{\mathcal{K}_1}^{\gamma_0,\kappa_0}(\Phi_1) = \emptyset \qquad\qquad \mathsf{C}_{\mathcal{K}_1}^{\gamma_0,\kappa_0}(\Phi_1) = \{\neg a\}\,.$$

■

Let $*$ be some (prioritized) multiple base revision operator, γ some categorizer, and κ some accumulator. Using the skeptical transformation function we can define the *skeptical argumentative revision* $\circ_S^{\gamma,\kappa}$ following (1) via

$$\mathcal{K} \circ_S^{\gamma,\kappa} \Phi = \mathcal{K} * \mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}(\Phi) \tag{2}$$

for every $\Phi \subseteq \mathcal{L}(\mathsf{At})$ and using the credulous transformation function we can define the *credulous argumentative revision* $\circ_C^{\gamma,\kappa}$ via

$$\mathcal{K} \circ_C^{\gamma,\kappa} \Phi = \mathcal{K} * \mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}(\Phi) \tag{3}$$

for every $\Phi \subseteq \mathcal{L}(\mathsf{At})$.

Example 5. We continue Example 4. Let $*$ be some prioritized multiple base revision. Then it follows that $\mathcal{K}_1 \circ_S^{\gamma_0,\kappa_0} \Phi_1 = \{a\}$ and $\mathcal{K}_1 \circ_C^{\gamma_0,\kappa_0} \Phi_1 = \{\neg a\}$. ■

We now investigate the formal properties of the transformation functions $\mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}$ and $\mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}$ and the resulting revision operators $\circ_S^{\gamma,\kappa}$ and $\circ_C^{\gamma,\kappa}$.

Proposition 2. *Let γ be a well-behaving categorizer and κ be a well-behaving accumulator. Then the transformation functions $\mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}$ and $\mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}$ satisfy* inclusion*,* weak inclusion*,* weak extensionality*,* consistency preservation *and* weak maximality*.*

Proof.

Inclusion. *This is satisfied by definition as for $\alpha \in \mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}(\Phi)$ and each $\alpha \in \mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}(\Phi)$ it follows $\alpha \in \Phi$.*

Weak Inclusion. *This follows directly from the satisfaction of* inclusion*.*

Weak Extensionality. *Let $\Phi \cong^p \Phi'$ and let $\sigma : \Phi \to \Phi'$ be a bijection such that for every $\phi \in \Phi$ it holds that $\phi \equiv^p \sigma(\phi)$. We extend σ to $\mathcal{K}$ via $\sigma(\psi) = \psi$ for every $\psi \in \mathcal{K}$. If $\Psi \subseteq \mathcal{K} \cup \Phi$ we abbreviate*

$$\sigma(\Psi) = \bigcup_{\psi \in \Psi} \{\sigma(\psi)\}.$$

Let $\langle \Psi, \phi \rangle$ be an argument for some $\phi \in \Phi$ with respect to $\mathcal{K} \cup \Phi$. Then $\langle \sigma(\Psi), \sigma(\phi) \rangle$ is an argument for $\sigma(\phi)$ in $\mathcal{K} \cup \Phi'$. It follows that if τ is an argument tree for $\langle \Psi, \phi \rangle$ in $\mathcal{K} \cup \Phi$ then τ' is an argument tree for $\langle \sigma(\Psi), \sigma(\phi) \rangle$ in $\mathcal{K} \cup \Phi'$ where τ' is obtained from τ by replacing each sentence ϕ with $\sigma(\phi)$. This generalizes also to argument structures and it follows that

$$\kappa(\gamma(\Gamma_{\mathcal{K} \cup \Phi}(\phi))) = \kappa(\gamma(\Gamma_{\mathcal{K} \cup \Phi'}(\sigma(\phi)))).$$

Hence, $\phi \in \mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}(\Phi)$ if and only if $\sigma(\phi) \in \mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}(\Phi')$ for every $\phi \in \Phi$. It follows that $\mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}(\Phi) \cong^p \mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}(\Phi')$. The same is true for $\mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}$.

Consistency Preservation. *Every subset of a consistent set of sentences is consistent and, due to* inclusion*, it holds that $\mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}(\Phi), \mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}(\Phi) \subseteq \Phi$ with consistent Φ.*

Weak Maximality. *If $\mathcal{K} \cup \Phi$ is consistent then for all arguments for a sentence $\alpha \in \Phi$ there do not exist any undercuts as these would have to entail the negation of some sentence of the argument for α which implies inconsistency of $\mathcal{K} \cup \Phi$. The argument structure $\Gamma_\Phi(\alpha) = (\mathcal{P}, \mathcal{C})$ consists of one or more single node trees $\mathcal{P}$ and $\mathcal{C} = \emptyset$. As both γ and κ are well-behaving it follows that $\kappa(\gamma(\Gamma_\Phi(\alpha))) > 0$ for each $\alpha \in \Phi$ and therefore $\mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}(\Phi) = \Phi$ and $\mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}(\Phi) = \Phi$.* □

In particular, note that both $\mathsf{S}_{\mathcal{K}}^{\gamma,\kappa}$ and $\mathsf{C}_{\mathcal{K}}^{\gamma,\kappa}$ do not satisfy either *consistency* or *maximality* in general.

Corollary 1. *Let γ be a well-behaving categorizer and κ be a well-behaving accumulator. Then both $\circ_S^{\gamma,\kappa}$ and $\circ_C^{\gamma,\kappa}$ are non-prioritized multiple base revision operators.*

Proof. This follows directly from Propositions 1 and 2. □

Example 6. We continue Examples 1 and 2 and consider $\mathsf{At} = \{c, a, q, b, r, s\}$ with the following informal interprations.

c : Anna has financial problems
a : Anna travels to Hawaii
q : there is volcano activity on Hawaii
b : Anna has a lot of money
r : Anna is a surf fanatic
s : Anna takes a loan

Now consider Anna's belief base $\mathcal{K}_2$ given via

$$\mathcal{K}_2 = \{r, \quad r \Rightarrow a, \quad s, \quad s \Rightarrow b, \quad b \Rightarrow a, \quad b \Rightarrow \neg c\}\,.$$

This means that Anna believes that she is a surf fanatic (r), that a surf fanatic should travel to Hawaii ($r \Rightarrow a$), that she takes a loan (s), that taking a loan means that she has a lot of money ($s \Rightarrow b$), that having a lot of money implies she should travel to Hawaii ($b \Rightarrow a$), and that having a lot of money she does not have financial problems. Note that $\mathcal{K} \vdash a$, i. e. Anna intends to got to Hawaii. Now consider the new information $\Phi_2 = \{c, \quad c \Rightarrow \neg a, \quad q, \quad q \Rightarrow \neg a\}$ stemming from communication with Anna's mother. In Φ_2 the mother of Anna tries to convince her not to travel to Hawaii. In particular, Φ_2 states that Anna has financial problems (c), that having financial problems Anna should not travel to Hawaii ($c \Rightarrow \neg a$), that there is also volcano activity on Hawaii (q), and that given volcano activity Anna should not travel to Hawaii ($q \Rightarrow \neg a$).

As one can see there a several arguments for and against a in $\mathcal{K}_2 \cup \Phi_2$, e. g., $\langle r, r \Rightarrow a, a\rangle$, $\langle c, c \Rightarrow \neg a, \neg a\rangle$. We do not go into details regarding the argumentative evaluation of the sentences in Φ_2. We only note that $\mathcal{K}_2 \cup \Phi_2$ is undecided about c but accepts $c \Rightarrow \neg a$, q, and $q \Rightarrow \neg a$ with respect to γ_0 and κ_0. Consequently, the values of $\mathsf{S}_{\mathcal{K}_2}^{\gamma_0,\kappa_0}(\Phi_2)$ and $\mathsf{C}_{\mathcal{K}_2}^{\gamma_0,\kappa_0}(\Phi_2)$ are given via

$$\mathsf{S}_{\mathcal{K}_2}^{\gamma_0,\kappa_0}(\Phi_2) = \Phi_2 \setminus \{c\} \quad \text{ and } \quad \mathsf{C}_{\mathcal{K}_2}^{\gamma_0,\kappa_0}(\Phi_2) = \Phi_2\,.$$

Let $*$ be some prioritized multiple base revision operator and define $\circ_S^{\gamma_0,\kappa_0}$ and $\circ_C^{\gamma_0,\kappa_0}$ via (2) and (3), respectively. Then some possible revisions of $\mathcal{K}_2$ with Φ_2 are given via

$$\mathcal{K}_2 \circ_S^{\gamma_0,\kappa_0} \Phi_2 = \{r,\ s \Rightarrow b,\ b \Rightarrow a,\ b \Rightarrow \neg c,\ c \Rightarrow \neg a,\ q \Rightarrow \neg a,\ q\}$$
$$\mathcal{K}_2 \circ_C^{\gamma_0,\kappa_0} \Phi_2 = \{r,\ s \Rightarrow b,\ b \Rightarrow a,\ b \Rightarrow \neg c,\ c \Rightarrow \neg a,\ c,\ q \Rightarrow \neg a,\ q\}\,.$$

Note that it holds $\mathcal{K}_2 \circ_S^{\gamma_0,\kappa_0} \Phi_2 \vdash \neg a$ and $\mathcal{K}_2 \circ_C^{\gamma_0,\kappa_0} \Phi_2 \vdash \neg a$. Hence, Anna accepts the conclusion of her mother's arguments not to travel Hawaii. However, if she revises her beliefs in a skeptical way she does not accept that she has financial problems. ■

6 Related Work

In terms of related work there are mainly two areas that are related to the work presented here. On the one hand, non-prioritized belief revision and on the other hand belief revision by argumentation. In the former area we instantiate and ex-

tended the non-prioritized revision operator of selective revision presented in [9] towards multiple revision and to revision of belief bases. Selective revision is one of the most general non-prioritized revision operator of the type *decision+revision* [11]. Moreover it allows for partial acceptance of the input, in contrast to most other approaches. Apart from *decision+revision* approaches there are *expansion+consolidation* approaches to non-prioritized belief revision. These perform a simple expansion by the new information, i.e. $\mathcal{K} \cup \Phi$, and then apply a consolidation operator ! that restores consistency, i.e. $\mathcal{K} * \Phi = (\mathcal{K} \cup \Phi)!$. This approach is limited to belief bases since all inconsistent belief sets are equal, i. e. $Cn(\bot) = \mathcal{L}(\mathsf{At})$. An instantiation of such an operator that is similar to the setup used in this work has been presented in [8]. The considered input to the revision consists of a set of sentences that form an explanation of some claim in the same form as the argument definition used here. However, as with all approaches of the type *expansion+consolidation*, new and old information are completely equal to the consolidation operator. In contrast, the approach presented here which makes use of two different mechanisms to first decide about if, and which part, of the input shall be accepted just considering the new information, and then performing prioritized belief revision of the old information. Also, there are *integrated choice* approaches that do not feature a two step process but a single step process applying the same technique for the selection and revision process. Mostly these approaches need some meta information, e. g. an epistemic entrenchment relation, and thus differ on the basic process as well as on the information needed.

While there has been some work on the revision of argumentation systems, very little work on the application of argumentation techniques for the revision process has been done so far, cf. [7]. In fact, the work most related to the work presented here makes use of negotiation techniques for belief revision [3, 13], without argumentation. In the general setup of [3] a symmetric merging of information from two sources is performed by means of a negotiation procedure that determines which source has to reduce its information in each round. The information to be given up is determined by another function. The negotiation ends when a consistent union of information is reached. While this can be seen as a one step process of merging or consolidation in general, the formalism also allows to differentiate between the information given up from the first source and the second source. In [3], this setting is then successively biased towards prioritizing the second source which leads to representation theorems for operations equivalent to selective revision satisfying *consistent expansion* and for classic AGM operators. While those results are interesting, the negotiation framework used in [3] is very different from the argumentation formalism used here and also very different from the setup of selective revision. Moreover, the functions for the negotiation and concession are left abstract. In [13] mutual belief revision is considered where two agents revise their respective belief state by information of the other agent. Both agents agree in a negotiation on the information that is accepted by each agent. The revisions of the agents are split into a selection function and two iterated revision functions which leads to operators satisfying

consistent expansion. The selection function is then a negotiation function on two sets of beliefs that represent the sets of belief that each agent is willing to accept from the other agent that might obey game theoretic principles. This setting has a very different focus as ours and also does not specify the selection function.

7 Conclusion

In this paper we combined the research strains of selective revision and deductive argumentation in order to implement non-prioritized multiple base revision operators that only revise by those portions of the new information that are justified. We only took some first steps in investigating the properties of those revision operators but were able to show that those comply with many desirable properties for non-prioritized revision. We discussed the performance of our operators by examples and briefly compared our approach to related work.

Future work includes a deeper analysis of the revisions $\circ_S^{\gamma,\kappa}$ and $\circ_C^{\gamma,\kappa}$ and a more thorough comparison with related work.

References

1. T. J. M. Bench-Capon and P. E. Dunne. Argumentation in artificial intelligence. *Artificial Intelligence*, 171:619–641, 2007.
2. P. Besnard and A. Hunter. A logic-based theory of deductive arguments. *Artificial Intelligence*, 128(1-2):203–235, 2001.
3. R. Booth. A negotiation-style framework for non-prioritised revision. In *Proceedings of TARK'01*, pages 137–150, 2001.
4. D. Makinson C. E. Alchourron, P. Gärdenfors. On the logic of theory change: Partial meet contraction and revision functions. *The Journal of Symbolic Logic*, Vol. 50, No. 2 (Jun., 1985):510–530, 1985.
5. J. P. Delgrande and Yi Jin. Parallel belief revision. In *Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI-08)*, 2008.
6. P. M. Dung. 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.
7. M. A. Falappa, G. Kern-Isberner, and G. R.Simari. Belief revision and argumentation theory. In *Argumentation in Artificial Intelligence*, pages 341–360. Springer, 2009.
8. M. A. Falappa, G. Kern-Isberner, and G. R. Simari. Explanations, belief revision and defeasible reasoning. *Artificial Intelligence*, 141(1):1–28, 2002.
9. E. Fermé and S. O. Hansson. Selective revision. *Studia Logica*, 63:331–342, 1999.
10. A. J. García and G. R. Simari. Defeasible Logic Programming: An Argumentative Approach. *Theory and Practice of Logic Programming*, 4(1–2):95–138, 2004.
11. S. O. Hansson. A survey of non-prioritized belief revision. *Erkenntnis*, 50(2-3):413–427, 1999.
12. S. O. Hansson. *A Textbook of Belief Dynamics*. Kluwer Academic Publishers, Norwell, MA, USA, 2001.
13. D. Zhang, N. Foo, T. Meyer, and R. Kwok. Negotiation as mutual belief revision. In *Proceedings of AAAI'04*, pages 317–322, 2004.

Stable extensions in timed argumentation frameworks

Maria Laura Cobo, Diego C. Martinez, and Guillermo R. Simari

Artificial Intelligence Research and Development Laboratory (LIDIA)
Department of Computer Science and Engineering, Universidad Nacional del Sur
Av. Alem 1253 - (8000) Bahía Blanca - Bs. As. - Argentina
{mlc,dcm,grs}@cs.uns.edu.ar
http://www.cs.uns.edu.ar/lidia

Abstract. A Timed Abstract Argumentation Framework is a novel formalism where arguments are only valid for consideration in a given period of time, which is defined for every individual argument. Thus, the attainability of attacks and defenses is related to time, and the outcome of the framework may vary accordingly. In this work we study the notion of stable extensions applied to timed-arguments. The framework is extended to include intermittent arguments, which are available with some repeated interruptions in time.

1 Introduction

One of the main concerns in Argumentation Theory is the search for rationally based positions of acceptance in a given scenario of arguments and their relationships. This task requires some level of abstraction in order to study pure semantic notions. Abstract argumentation systems [10, 15, 3, 2] are formalisms for argumentation where some components remain unspecified, being the structure of an argument the main abstraction. In this kind of system, the emphasis is put on the semantic notion of finding the set of accepted arguments. Most of these systems are based on the single abstract concept of *attack* represented as an abstract relation, and extensions are defined as sets of possibly accepted arguments. For two arguments $\mathcal{A}$ and $\mathcal{B}$, if $(\mathcal{A}, \mathcal{B})$ is in the attack relation, then the acceptance of $\mathcal{B}$ is conditioned by the acceptance of $\mathcal{A}$, but not the other way around. It is said that argument $\mathcal{A}$ *attacks* $\mathcal{B}$, and it implies a priority between conflicting arguments.

The simplest abstract framework is defined by Dung in [10]. It only includes a set of abstract arguments and a binary relation of attack between arguments. Several semantics notions are defined and the Dung's argument extensions became the foundation of further research. Other proposals extends Dung's framework by the addition of new elements, such as preferences between arguments [3, 6] or subarguments [13]. Other authors use the original framework to elaborate new extensions [11, 5]. All of these proposals are based on varied abstract formalizations of arguments and attacks.

In this scenario, the combination of time and argumentation is a novel research line. In [12] a calculus for representing temporal knowledge is proposed, and defined in terms of propositional logic. This calculus is then considered with respect to argumentation, where an argument is defined in the standard way: an argument is a pair constituted by

a minimally consistent subset of a database entailing its conclusion. This work is thus related to [4].

In [7, 8] a novel framework is proposed, called *Timed Abstract Framework* (TAF), combining arguments and temporal notions. In this formalism, arguments are relevant only in a period of time, called its *availability interval*. This framework mantains a high abstract level in an effort to capture intuitions related with the dynamic interplay of arguments as they become available and cease to be so. The notion of *availability interval* refers to an interval of time in which the argument can be legally used for the particular purpose of an argumentation process. Thus, this kind of timed-argument has a limited influence in the system, given by the temporal context in which these arguments are taken into account. For example, consider the following argument:

My client committed no crime, since he was drafted for the ongoing war

In order to argue about an alleged crime, this argument can only be used when there is an actual war in which the defended is involved. The same argument cannot be used as a defense for crimes committed after the soldier was discharged. Thus, this argument has a temporal relevance. Timed abstract frameworks capture the previous argument model by assigning arguments to an availability interval of time. In [8] a skeptical, timed interval-based semantics is proposed, using admissibility notions. As arguments may get attacked during a certain period of time, defense is also time-dependant, requiring a proper adaptation of classical acceptability. In [7], algorithms for the characterization of defenses between timed arguments are presented.

In this work we formalize a natural expansion of timed argumentation frameworks by considering arguments with more than one availability interval. These are called *intermittent arguments*. These arguments are available with (possibly) some repeated interruptions in time. For instance, a traveling salesman may have a set of arguments to be used in negotiations, but maybe some of them are irrelevant (or politically incorrect) in different cultures, and then these arguments can only be used while staying in certain cities. In some legal procedures, a lawyer can use some arguments depending on the actual stage of the process (initial disclosure of evidence, the trial, the appeal). Some countries apply sets of specific legal and economic rules during certains periods of time, like financial crises, natural disasters or wartime. This has an impact on, for instance, political argumentation. In all of these scenarios arguments may become relevant, or cease to be so, depending on time-related factors. Using this extended timed argumentation framework, we analyze the notion of *stable extension*. A stable extension is a set of arguments attacking every other argument not in the same set. Since in a timed context arguments are attacked sporadically, the characterization of a stable set of arguments requires a deeper analysis.

This paper is organized as follows. In the next section we recall classic argumentation semantic notions. Thereafter, time-intervals and the terminology used in this work are defined, towards the presentation of our Timed Abstract Argumentation Framework with intermittent arguments in Section 4. The notion of stable extension is presented in Section 5. Finally, conclusions and future work are discussed.

2 Classic abstract argumentation

Dung defines several argument extensions that are used as a reference for many authors. The formal definition of the classic argumentation framework follows.

Definition 1 [10] *An argumentation framework is a pair $AF = \langle AR, attacks \rangle$ where AR is a set of arguments, and $attacks$ is a binary relation on AR, i.e. $attacks \subseteq AR \times AR$.*

Arguments are denoted by labels starting with an uppercase letter, leaving the underlying logic unspecified. A set of accepted arguments is characterized in [10] using the concept of *acceptability*, which is a central notion in argumentation, formalized by Dung in the following definition.

Definition 2 [10] *An argument $\mathcal{A} \in AR$ is acceptable with respect to a set of arguments S if and only if every argument $\mathcal{B}$ attacking $\mathcal{A}$ is attacked by an argument in S.*

If an argument $\mathcal{A}$ is acceptable with respect to a set of arguments S then it is also said that S *defends* $\mathcal{A}$. Also, the attackers of the attackers of $\mathcal{A}$ are called *defenders* of $\mathcal{A}$. We will use these terms throughout this paper.

Acceptability is the main property of Dung's semantic notions, which are summarized in the following definition.

Definition 3 *A set of arguments S is said to be*
– conflict-free if there are no arguments $\mathcal{A}, \mathcal{B}$ in S such that $\mathcal{A}$ attacks $\mathcal{B}$.
– admissible if it is conflict-free and defends all its elements.
– a preferred extension if S is a maximal (for set inclusion) admissible set.
– a complete extension if S is admissible and it includes every acceptable argument w.r.t. S.
– a grounded extension if and only if it is the least (for set inclusion) complete extension.
– a stable extension if S is conflict-free and it attacks each argument not belonging to S.

The grounded extension is also the least fixpoint of a simple monotonic *characteristic* function:

$$F_{AF}(S) = \{\mathcal{A} : \mathcal{A} \text{ is acceptable with respect to } S\}.$$

In [10], theorems stating conditions of existence and equivalence between these extensions are also introduced.

Example 1 *Consider the argumentation framework $AF_1 = \langle AR, attacks \rangle$, where $AR = \{\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}, \mathcal{E}, \mathcal{F}, \mathcal{G}, \mathcal{H}\}$ and $attacks = \{(\mathcal{B}, \mathcal{A}), (\mathcal{C}, \mathcal{B}), (\mathcal{D}, \mathcal{A}), (\mathcal{E}, \mathcal{D}), (\mathcal{G}, \mathcal{H}), (\mathcal{H}, \mathcal{G})\}$. Then*
– $\{\mathcal{A}, \mathcal{C}, \mathcal{E}\}$ is an admissible set of arguments.
– $\{\mathcal{A}, \mathcal{C}, \mathcal{E}, \mathcal{F}, \mathcal{G}\}$ is a preferred extension. It is also a complete extension.
– $\{\mathcal{A}, \mathcal{C}, \mathcal{E}, \mathcal{F}\}$ is the grounded extension.

Dung's abstract formalism is sufficient to define some basic extensions on arguments. In this work we study the formalization of intermittent timed-arguments in an abstract framework, and we present an argument extension inspired by the stable semantics. In the following section we prepare the road to timed argumentation by introducing several time-related concepts.

3 Time representation

In order to capture a time-based model of argumentation, we enrich the classical abstract frameworks with temporal information regarding arguments. The problem of representing temporal knowledge and temporal reasoning arises in a lot of disciplines, including Artificial Intelligence. There are many ways of representing temporal knowledge. A usual way to do this is to determine a *primitive* to represent time, and its corresponding *metric relations* [1, 9, 14]. In this work we will use *temporal intervals of discrete time* as primitives for time representation, and thus only metric relations for intervals are applied.

Definition 4 *[Temporal Interval] An interval is a pair build from* $a, b \in \mathbb{Z} \cup \{-\infty, \infty\}$*, in one of the following ways:*

- $[a, a]$ *denotes a set of time moments formed only by moment* a*.*
- $[a, \infty)$ *denotes a set of moments formed by all the numbers in* $\mathbb{Z}$ *since* a *(including* a*).*
- $(-\infty, b]$ *denotes a set of moments formed by all the numbers in* $\mathbb{Z}$ *until moment* i *(including* b*).*
- $[a, b]$ *denotes a set of moments formed by all the numbers in* $\mathbb{Z}$ *moment* i *until moment* j *(including both* a *and* b*).*
- $(-\infty, \infty)$ *a set of moments formed by all the numbers in* $\mathbb{Z}$*.*

The moments a, b *are called endpoints.*

The set of all the intervals defined over $\mathbb{Z} \cup \{-\infty, \infty\}$ *is denoted* Υ*.*

For example, $[5, 12]$ and $[1, 200]$ are intervals. If X is an interval then X^-, X^+ are the corresponding endpoints ($i.e.$, $X = [X^-, X^+]$). And endpoint may be a point of discrete time, identified by an integer number, or infinite.

There are thirteen possible relations between intervals [1]. Seven of them are the basics while the remaining six are defined as their inverses. In the context of this work it is unnecessary to keep track of the difference between some of Allen relations, particularly *starts*, *during* and *finishes*, so we redefine the *during*. On Table 1 we present the relation we are going to use in the paper. The 'x's and 'y's represents the interval X and Y respectively. The table shows the relation between *endpoints*.

We will usually work with sets of intervals (as they will be somehow related to arguments). Thus, we introduce several definitions and properties needed for semantic elaborations.

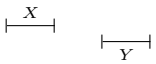
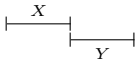
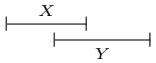
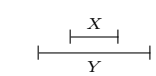
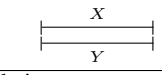
Relation	*Symb*	*Inverse*	*e.g.*	*Relation on Endpoints*
X Before Y	(b)	(bi)		$X^+ < Y^-$
X Meets Y	(m)	(mi)		$X^+ = Y^-$
X Overlaps Y	(o)	(oi)		$X^- < Y^-, X^+ > Y^-$
X During Y	(d)	(di)		$X^- \geq Y^-, X^+ < Y^+$ or $X^- > Y^-, X^+ \leq Y^+$
X Equal Y	(e)			$X^- = Y^-, X^+ = Y^+$

Table 1. Qualitative relations among arguments, based on [1]

.

Definition 5 *Let S be a set of intervals and let i be a moment of time. The exclusion of i from S, denoted $S \ominus i$, is defined as follows:*

$$\begin{aligned} S \ominus i = &\{I : I \in S \wedge i \notin I\} \cup \\ &\{[I^-, i-1] : I \in S \wedge i \in I, I \neq I^-\} \cup \\ &\{[i+1, I^+] : I \in S \wedge i \in I, I \neq I^+\} \end{aligned}$$

The exclusion of the interval I from S, noted as $S \textcircled{\scriptsize -} i$, can be recursively defined as follows:

$$\begin{aligned} &i.\ S \textcircled{\scriptsize -} I = S \ominus I^- && \text{si } I^- = I^+ \\ &ii.\ S \textcircled{\scriptsize -} I = (S \ominus I^-) \textcircled{\scriptsize -} [I^- + 1, I^+] && \text{if } I^- \neq I^+ \end{aligned}$$

The operation needed is the difference among set of intervals, $i.e. S_1 - S_2$ being S_1 and S_2 sets of intervals. This operation can be defined recursively using $\textcircled{\scriptsize -}$.

Intersection is another relevant operation on intervals. The intersection of two intervals is the interval formed by all the common points of both of them. Its endpoints are the minimal and maximal time points in common.

Definition 6 *Let I_1 and I_2 be two intervals. The intersection is defined as: $I_1 \cap I_2 = [x, y]$ with $x, y \in I_1$ and $x, y \in I_2$ such that there are no $w, z : w, z \in I_1$ and $w, z \in I_2$ with $w < x$ or $y < z$.*

Definition 7 *Let S_1 and S_2 be two sets of intervals. The intersection of these sets, noted as $S_1 \Cap S_2$, is: $S_1 \Cap S_2 = \{I : I = I_1 \cap I_2 \neq [\,], \forall I_1 \in S_1, I_2 \in S_2\}$*

Definition 8 *Let S be a set of intervals. The partition of S, denoted* $\mathsf{Part(S)}$ *is defined as:*

- $\mathsf{Part(S)} = S$ *if* $\forall I_1, I_2 \in S, I_1 \cap I_2 = \emptyset$.
- $\mathsf{Part(S)} = \mathsf{Part(S} - \{\mathsf{I}_1, \mathsf{I}_2\} \cup \{\mathsf{I}_1 - (\mathsf{I}_1 \cap \mathsf{I}_2), \mathsf{I}_2 - (\mathsf{I}_1 \cap \mathsf{I}_2), \mathsf{I}_1 \cap \mathsf{I}_2\})$, *with* $I_1, I_2 \in S$ *and* $I_1 \cap I_2 \neq \emptyset$

The partition of a set of argument's availability *breaks* overlapped intervals in smaller intervals. This notion simplifies semantic elaborations, since it discretizes the evolution of the framework according to moments where arguments start or cease to be available.

Any set of intervals, either fragmented or not, has several non-fragmented subsets. In fact, any singleton subset is trivially non-fragmented.

Although the previous definition can be given in terms of interval calculus relations (Table 1), we use a different notation to improve their readability.

In the following section we present Timed Abstract Argumentation Frameworks with intermittent arguments.

4 Timed Argumentation Framework

As remarked before, in Timed Argumentation Frameworks the consideration of time restrictions for arguments is formalized through an *availability function*, which defines a temporal interval for each argument in the framework. This interval states the period of time in which an argument is available for consideration in the argumentation scenario. The formal definition of our timed abstract argumentation framework follows.

Definition 9 *A timed abstract argumentation framework (TAF) is a 3-tuple* $\langle Args, Atts, \mathcal{A}v\rangle$ *where* $Args$ *is a set of arguments,* $Atts$ *is a binary relation defined over* $Args$ *and* $\mathcal{A}v$ *is the availability function for timed arguments, defined as* $\mathcal{A}v : Args \rightarrow \wp(\upsilon)$.

Example 2 *The triplet* $\langle Args, Atts, \mathcal{A}v\rangle$, *where* $Args = \{\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}, \mathcal{E}\}$, $Atts = \{(\mathcal{B}, \mathcal{A}), (\mathcal{C}, \mathcal{B}), (\mathcal{D}, \mathcal{A}), (\mathcal{E}, \mathcal{D})\}$ *and the availability function is defined as*

$Args$	$\mathcal{A}v$	$Args$	$\mathcal{A}v$
$\mathcal{A}$	$\{[10, 40], [60, 75]\}$	$\mathcal{B}$	$\{[30, 50]\}$
$\mathcal{C}$	$\{[20, 40], [45, 55], [60, 70]\}$	$\mathcal{D}$	$\{[47, 65]\}$
$\mathcal{E}$	$\{(-\infty, 44]\}$		

is a timed abstract argumentation framework.

The framework of Example 2 can be depicted as in Figure 1, using a digraph where nodes are arguments and arcs are attack relations. An arc from argument $\mathcal{X}$ to argument $\mathcal{Y}$ exists if $(\mathcal{X}, \mathcal{Y}) \in Atts$. Figure 1 also shows the time availability of every argument, as a graphical reference of the $\mathcal{A}v$ function. It is basically the framework's evolution in time. Endpoints are marked with a vertical line, except for $-\infty$ and ∞. For space reasons, only some relevant time points are numbered in the figure. As stated before, the availability of arguments is tied to a temporal restriction. Thus, an attack to an argument may actually occur only if both the attacker and the attacked argument are available. In other words, an attack between two arguments may be *attainable*, under certain conditions. *Attainable attacks* are attacks that will eventually occur in some period of time. In order to formalize this, we need to compare time intervals, using the previously defined metric relations.

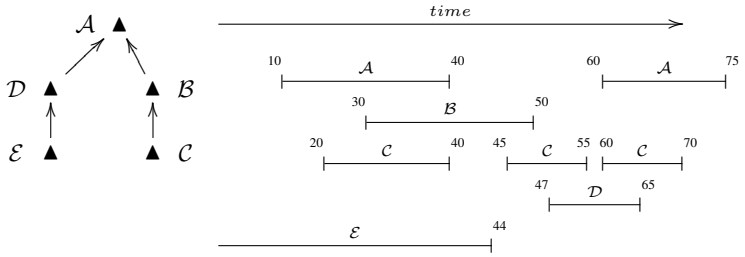


Fig. 1. Framework of Example 2

Definition 10 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF, and let $\{\mathcal{A}, \mathcal{B}\} \subseteq Args$ such that $(\mathcal{B}, \mathcal{A}) \in Atts$. The attack $(\mathcal{B}, \mathcal{A})$ is said to be attainable if the following conditions holds: $I_{\mathcal{A}}$ R $I_{\mathcal{B}}$, where $R \in \{$ⓓ, ⓓⓘ, ⓞ,ⓞⓘ ⓔ$\}$ for some $I_{\mathcal{A}} \in \mathcal{A}v(\mathcal{A})$ and $I_{\mathcal{B}} \in \mathcal{A}v(\mathcal{B})$. The attack is said to be attainable in $\mathcal{A}v(\mathcal{A}) \Cap \mathcal{A}v(\mathcal{B})$. The set of intervals where an attack $(\mathcal{B}, \mathcal{A})$ is attainable will be noted as IntSet$((\mathcal{B}, \mathcal{A}))$*

Note that an attack is attainable if the availability of both the attacker and the attacked argument eventually overlaps.

Example 3 *Consider the timed argumentation framework of example 2. The attacks $(\mathcal{D}, \mathcal{A})$ and $(\mathcal{B}, \mathcal{A})$ are both attainable in the framework. Attack $(\mathcal{D}, \mathcal{A})$ is attainable since $[47, 65]$ⓞ $[60, 75]$ with $[47, 65] \in \mathcal{A}v(\mathcal{D})$ and $[60, 75] \in \mathcal{A}v(\mathcal{A})$. Attack $(\mathcal{B}, \mathcal{A})$ is attainable since $[30, 50]$ⓞⓘ $[10, 40]$, in $[30, 40]$. Recall that $[30, 50] \in \mathcal{A}v(\mathcal{B})$, $[10, 40] \in \mathcal{A}v(\mathcal{A})$. The attack $(\mathcal{C}, \mathcal{B})$ is also attainable. Since $\mathcal{A}v(\mathcal{C}) = \{[20, 40], [30, 50]\}$ and $\mathcal{A}v(\mathcal{B}) = \{[30, 50]\}$ then we can assure the attainability of the attack by one of the following relations:$ $[20, 40]$ⓞ $[30, 50]$, $[45, 55]$ⓞⓘ $[30, 50]$. The attack is then attainable at $\{[30, 40], [45, 50]\}$, i.e.$\mathcal{A}v(\mathcal{C}) \Cap \mathcal{A}v(\mathcal{B})$. The attack $(\mathcal{E}, \mathcal{D})$ is not attainable, since $(-\infty, 45]$ⓑ $[47, 65]$. The arguments involved in this attack are never available at the same time.*

The set of all attainable attacks in the framework Φ is denoted $AttAtts_{\Phi}$. It is also possible to define the attainability of attacks at a particular timed intervals, as shown next.

Definition 11 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF, and let $\{\mathcal{A}, \mathcal{B}\} \subseteq Args$ such that $(\mathcal{B}, \mathcal{A}) \in Atts$. The attack $(\mathcal{B}, \mathcal{A})$ is said to be attainable at I if: $I \cap \mathcal{A}v(\mathcal{A}) \neq [\,]$ and the following condition holds: $I \cap I_{\mathcal{A}}$ R $I_{\mathcal{B}}$, where $R \in \{$ⓓ, ⓓⓘ, ⓞ, ⓞⓘ, ⓔ$\}$, for some $I_{\mathcal{A}} \in \mathcal{A}v(\mathcal{A})$ and $I_{\mathcal{B}} \in \mathcal{A}v(\mathcal{B})$.*

The set of attainable attacks of Φ at interval I is denoted $AttAtts^{I}_{\Phi}$.

Example 4 *Consider the timed argumentation framework of Example 2. The set $AttAtts_{\Phi}$ is: $\{(\mathcal{D}, \mathcal{A}), (\mathcal{B}, \mathcal{A}), (\mathcal{C}, \mathcal{B})\}$. The set $AttAtts^{[35,40]}_{\Phi}$ is $\{(\mathcal{B}, \mathcal{A}), (\mathcal{C}, \mathcal{B})\}$. The attack $(\mathcal{D}, \mathcal{A})$ is in $AttAtts_{\Phi}$ but it is not in $AttAtts^{[35,40]}_{\Phi}$, since $[35, 40] \cap [47, 65]$ is the emptyset. The attack $(\mathcal{B}, \mathcal{A})$ is in $AttAtts_{\Phi}$ and is also in $AttAtts^{[35,40]}_{\Phi}$, since $[35, 40] \cap [10, 40] = [35, 40]$ and $[35, 40]$ⓓ $[30, 50]$. Note that $[10, 40] \in \mathcal{A}v(\mathcal{A})$ and $[30, 50] \in \mathcal{A}v(\mathcal{B})$.*

The definition of attainability of attacks can be attached to particular time points too. The set of attainable attacks of Φ at moment i is denoted $AttAtts_{\Phi}(i)$ and is defined as $AttAtts_{\Phi}(i) = AttAtts_{\Phi}^{[i,i]}$.

5 Semantics for Timed Argumentation

As attacks may occur only on a period of time (that in which the participants are available), argument defense is also occasional. In [8] a skeptical, timed interval-based semantics is proposed, using admissibility notions. The classical definition of acceptability is adapted to a timed context. The complexity of this adaptation lies on the fact that defenses may occur sporadically and hence the focus is put on finding *when* the defense takes place. For example, an argument $\mathcal{A}$ may be defended by $\mathcal{X}$ in the first half of its time interval, and later by an argument $\mathcal{Y}$ in the second half. Although $\mathcal{X}$ is not capable of providing a full defense, argument $\mathcal{A}$ is defended while $\mathcal{A}$ is available. In other words, defenders *take turns* to provide a defense.

In this paper we are mainly interested in the notion of stable set of arguments. In the classical sense, a stable extension of an argumentation framework is a set of arguments S that attacks each argument not belonging to S. A set of arguments S is said to attack an argument $\mathcal{A}$ if at least one argument in S is an attacker of $\mathcal{A}$.

In timed argumentation frameworks, an attack may be attainable only in a restricted period of time, and so the exitance of an attacker in the set is not enough now.

Definition 12 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF, and let $\mathcal{A} \in Args$. Argument $\mathcal{A}$ is a threatened argument if there is at least one argument $\mathcal{B}$, such that $(\mathcal{B}, \mathcal{A}) \in AttAtts_{\Phi}$.*

A threat interval is a period of time in which an argument attacks another. Naturally, it is possible for an argument to have more than one threat interval. Consider again the framework of Example 2. Argument $\mathcal{A}$ has two threat intervals since there are two attainable attacks, $(\mathcal{B}, \mathcal{A})$ and $(\mathcal{D}, \mathcal{A})$. The set of intervals $\tau_{\mathcal{B}}(\mathcal{A})$ is $\{[30, 40]\}$ while $\tau_{\mathcal{D}}(\mathcal{A})$ is $\{[60, 65]\}$. Notice that $\tau_{\mathcal{X}}(\mathcal{Y})$ is a set in general since it is possible that $\mathcal{X}$ threats $\mathcal{Y}$ in more than one interval, that is the case of the attack $(\mathcal{C}, \mathcal{B})$, $\tau_{\mathcal{C}}(\mathcal{B}) = \{[30, 40], [54, 50]\}$.

Since attacks are sporadic we need to know in which subintervals of its availability interval an argument is threatened. We need a general structure associating arguments and intervals. This is captured by the notion of *t-profile* as defined next.

Definition 13 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF. A t-profile is general structure $\langle arg, set\rangle$ where $arg \in Args$ and $set \in \wp(\Upsilon)$ and for each $I \in set$, $I \subseteq I_{arg}$ with $I_{arg} \in \mathcal{A}v(arg)$. The set of all the t-profiles definable from Φ will be noted as $\mathfrak{P}$.*

A t-profile is a record of an argument $\mathcal{A}$ associated with a set of intervals with only one restriction: every interval in the set is a subinterval of an availability interval of $\mathcal{A}$. In other words, t-profiles reflect a particular view of the availability of an argument. The associated set of intervals may denote moments of attacks, or moments of defense, or any other special consideration of an argument regarding a set of intervals. In particular, a set of t-profiles is the consideration of a set of arguments in several moments of time.

For instance, the set $S = \{\langle \mathcal{A}, \{I_1\}\rangle, \langle \mathcal{B}, \{I_2\}\rangle\}$ includes two arguments $\mathcal{A}$ and $\mathcal{B}$, but it is possible that these arguments never co-exist if I_1 and I_2 are not overlapping intervals. The set S may denote a special status for arguments, and *when* this status is assigned.

Remember that a stable extension attacks every single external argument. Since we are dealing with arguments restricted to intervals, it is necessary to define what it means for a t-profile to be attacked by a set of t-profiles. This is substantially different than in classic frameworks. In order to figure out why, consider an argument $\mathcal{A}$ attacked by two arguments $\mathcal{B}$ and $\mathcal{C}$. Suppose that $\mathcal{A}$ is attacked by $\mathcal{B}$ and $\mathcal{C}$ in two different, non-overlapping intervals I_1 and I_2, such that $\mathcal{B}$ attacks $\mathcal{A}$ during I_1 and $\mathcal{C}$ attacks $\mathcal{A}$ during I_2. There is no moment of time in which $\mathcal{A}$ is free of an attainable attack, either from $\mathcal{B}$ or $\mathcal{C}$ since both attackers take turns to attack $\mathcal{A}$. Is the set $S_1 = \{\mathcal{B}\}$ attacking $\mathcal{A}$? In the classical sense of stable semantics it does, since an attack finally occurs. However, when time dimension is considered, S_1 does not attack $\mathcal{A}$ in every moment in which $\mathcal{A}$ is considered: in I_2 argument $\mathcal{A}$ is not attacked by any argument in S_1. On the other hand, $\mathcal{A}$ *always* has an attacker in the set $S_2 = \{\mathcal{B}, \mathcal{C}\}$.

Clearly, any further analysis of attacks between timed arguments must take intervals into account. Thus, following definition formalizes an attack between a set of t-profiled and a single t-profile.

Definition 14 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF, let $S \subseteq \mathfrak{P}$ be a set of t-profiles and let $\mathcal{X}$ be an argument in $Args$. The set S attacks a t-profile $\langle \mathcal{X}, \mathfrak{D}_\mathcal{X}\rangle$ if $\mathfrak{D}_\mathcal{X} \subseteq \mathfrak{T}$ with*

$$\mathfrak{T} = \{\tau_\mathcal{Y}(\mathcal{X}) \cap \mathfrak{D}_\mathcal{Y} : \langle \mathcal{Y}, \mathfrak{D}_\mathcal{Y}\rangle \in S \text{ and } \tau_\mathcal{Y}(\mathcal{X}) \cap \mathfrak{D}_\mathcal{Y} \neq \emptyset\}$$

A t-profile $t = \langle \mathcal{X}, \mathfrak{D}_\mathcal{X}\rangle$ is attacked by a set S of t-profiles if every interval of t overlaps an interval of an attacker in S. In order to define a stable extension, a formal definition of conflict-freeness for sets of t-profile is needed.

Definition 15 *Let $\langle Args, Atts, \mathcal{A}v\rangle$ be a TAF, and let $S \subseteq \mathfrak{P}$. The set S is said to be conflict-free if there are not two profiles $\langle \mathcal{A}, \mathfrak{D}_\mathcal{A}\rangle$ and $\langle \mathcal{B}, \mathfrak{D}_\mathcal{B}\rangle$ such that*

$$\exists i \in \mathfrak{D}_\mathcal{B}, i \in \mathfrak{D}_\mathcal{B} : (\mathcal{A}, \mathcal{B}) \in AttAtts_\Phi^{[i,i]}$$

Now we are in conditions to define the stable semantics. In order for a set S of t-profiles to be considered a stable extension, two main situations must be considered. First, the set S must attack an argument $\mathcal{X}$ not considered in any t-profile of S. This is consistent with the classical notion of stable extension: if it is not in S, then it must be attacked. Second, if an argument $\mathcal{X}$ has a t-profile in S, then it must be attacked by S in any moment of time in which $\mathcal{X}$ is not considered in S. The formal definition follows.

Definition 16 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF and let $S \subseteq \mathfrak{P}$ a set of t-profiles. The set S is an stable extension if and only if:*

1. *S is conflict-free.*
2. *$\forall \mathcal{X} \in Args$ such that $\langle \mathcal{X}, \mathfrak{D}_\mathcal{X}\rangle \notin S$, S attacks $\langle \mathcal{X}, \mathcal{A}v(\mathcal{X})\rangle$.*
3. *$\forall \mathcal{X} \in Args$ such that $\langle \mathcal{X}, \mathfrak{D}_\mathcal{X}\rangle \in S$, S attacks $\langle \mathcal{X}, \mathcal{A}v(\mathcal{X}) - \mathfrak{D}_\mathcal{X}\rangle$.*

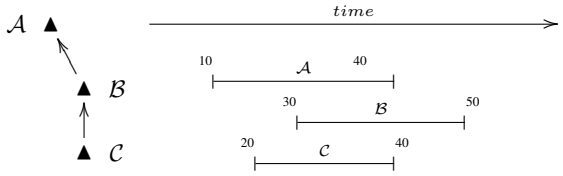


Fig. 2. Framework $\Phi_{\mathcal{E}}$

Example 5 *Let $\Phi_{\mathcal{E}} = \langle Args, Atts, \mathcal{A}v\rangle$, be the TAF depicted on Figure 2. The set $S = \{\langle \mathcal{A}, [10, 40]\rangle, \langle \mathcal{B}, (40, 50]\rangle, \langle \mathcal{C}, [20, 40]\rangle\}$ is a stable extension of $\Phi_{\mathcal{E}}$. The set S is clearly conflict-free and there are no arguments in $Args$ not considered in S. Thus, only condition (3) of Definition 16 is relevant. In this particular case, the only argument in such condition is $\mathcal{B}$ for which the availabily in $\Phi_{\mathcal{E}}$ is $[30, 50]$. However the t-profile of $\mathcal{B}$ in S considers only the sub-interval $(40, 50]$. The rest of the original availability interval is $[30, 40]$, in which $\mathcal{B}$ is attacked by $\mathcal{C}$. Argument $\mathcal{C}$ is in the set S, and according to the corresponding profile it is associated with all the availability interval $\{[20, 40]\}$.*

The set $S_2 = \{\langle \mathcal{A}, [10, 20]\rangle, \langle \mathcal{C}, [20, 40]\rangle\}$ is not a stable extension, since there is argument $\mathcal{B}$, that is not included in a t-profile in S_2 and it is not attacked by an t-profile in the set. The set S_2 should attack $\langle \mathcal{B}, \mathcal{A}v(\mathcal{B})\rangle$ in order to be a stable extension, but this is not the case. It can be observed that there is a period of time in which $\mathcal{B}$ is not attacked by any other argument.

Finally the set $S_3 = \{\langle \mathcal{A}, [10, 30)\rangle, \langle \mathcal{B}, [30, 50]\rangle, \langle \mathcal{C}, [20, 30)\rangle\}$ is not stable extension. In this case the problem is related to $\mathfrak{D}_{\mathcal{C}} = \{[20, 30)\}$, the argument $\mathcal{C}$ has $\{[20, 40]\}$ as its availability period in the framework and has no attackers there. Then the third condition fails, the t-profile $\langle \mathcal{C}, \mathcal{A}v(\mathcal{C}) - \{[10, 30]\}\rangle = \langle \mathcal{C}, \mathcal{A}v(\mathcal{C}) - \{[10, 19)\}\rangle$ is not attacked by S_3.

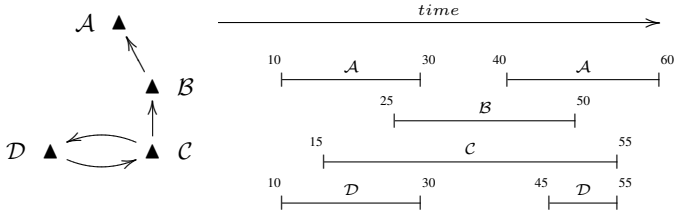


Fig. 3. Framework $\Phi_{\mathcal{S}}$

Example 6 *Let $\Phi_{\mathcal{S}} = \langle Args, Atts, \mathcal{A}v\rangle$, be the TAF depicted on Figura 3. In this framework we have two stable extensions:*

$$S_1 = \{\ \langle \mathcal{A}, \{[10, 24], [51, 50]\}\rangle, \langle \mathcal{B}, \{[25, 30], [45, 50]\}\rangle, \langle \mathcal{C}, \{[31, 44]\}\rangle, \langle \mathcal{D}, \mathcal{A}v\mathcal{D}\rangle \quad \}$$

$$S_2 = \{\ \langle \mathcal{A}, \mathcal{A}v(\mathcal{A})\rangle, \langle \mathcal{C}, \mathcal{A}v(\mathcal{C})\rangle, \langle \mathcal{D}, \{[10, 14]\}\rangle\ \}$$

This example shows shows that the presence of cycles requires a proper consideration of time. In non-temporal, classical frameworks whenever cycles like the one in $\Phi_{\mathcal{S}}$ are present, only one of the participant appears in the stable extension. In timed argumentation frameworks, the analysis requires an examination of the associated time intervals. In this case both arguments $\mathcal{C}$ and $\mathcal{D}$ appear in the extension, but with different time restrictions. Note that the set $S_3 = \{\langle \mathcal{A}, \mathcal{A}v(\mathcal{A})\rangle, \langle \mathcal{C}, \mathcal{A}v(\mathcal{C})\rangle\}$ is not a stable extension since it is not true that $\forall \mathcal{X} \in Args$ such that $\langle \mathcal{X}, \mathfrak{D}_{\mathcal{X}}\rangle \notin S$, S attacks $\langle \mathcal{X}, \mathcal{A}v(\mathcal{X})\rangle$, since S_3 does not attack $\langle \mathcal{D}, \mathcal{A}v(\mathcal{D})\rangle$. The condition that $\mathcal{A}v(\mathcal{D}) = \{[10, 30], [45, 55]\}$ must be a subset of $\mathfrak{T}$ does not hold. Since $\mathcal{C}$ is the only attacker of $\mathcal{D}$, the set $\mathfrak{T}$ is $\tau_{\mathcal{C}}(\mathcal{D}) \cap \mathcal{A}v(\mathcal{C})$ $i.e.\{[15, 30], [45, 55]\} \Cap \{15, 55\} = \{[15, 30], [45, 55]\}$. It is clear that $\{[10, 30], [45, 55]\} \not\subseteq \{[15, 30], [45, 55]\}$.

A set of t-profiles is a set of arguments that are put in specific contextual time restrictions. Example 6 shows that a set of t-profiles may have arguments with potential conflicts, and this is legal as far as conflictive arguments are considered in different intervals of time.

As shown in previous section, Dung defines the notion of admissibility. This notion can be naturally extended to timed argumentation frameworks. Again, we use t-profiles to associate arguments with sets of intervals. An addmissible set is a set of t-profiles that is conlict-free and defends all of its elements.

Definition 17 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF and $S \subseteq \mathfrak{P}$ a set of t-profiles. The set of defense intervals for $\mathcal{A}$ against $\mathcal{B}$, denoted $\delta^{\mathcal{B}}_{\mathcal{A}}(S)$, is defined as:*

$$\bigcup\{\mathit{IntSet}((\mathcal{X}, \mathcal{B})) \Cap \mathit{IntSet}((\mathcal{B}, \mathcal{A})) \Cap \mathfrak{D}_{\mathcal{X}} : \langle \mathcal{X}, \mathfrak{D}_{\mathcal{X}}\rangle \in S\}$$

The set of defense intervals is the set of all the intervals in which an attacker is attacked by another argument. This is obtained by intersection of availability intervals.

Example 7 *Consider the timed argumentation framework of Example 2 and the set $S = \{\langle \mathcal{C}, \{[35, 40], [48, 52]\}\rangle\}$. The set of defense intervals for $\mathcal{A}$ against $\mathcal{B}$ is*

$$\begin{aligned}\delta^{\mathcal{B}}_{\mathcal{A}}(S) &= \{\mathit{IntSet}((\mathcal{C}, \mathcal{B})) \Cap \mathit{IntSet}((\mathcal{B}, \mathcal{A})) \Cap \mathfrak{D}_{\mathcal{X}} \\ &= \{[30, 40], [45, 50]\} \Cap \{[30, 40]\} \Cap \{[35, 40], [48, 52]\} \\ &= \{[35, 40]\}\end{aligned}$$

The following definition considers every attainable attacks at a certain period I, in order to grant defense. If there are multiple attackers, then the defense takes place in those moments where the argument has defenders for all of the attainable attacks, that is, no attack succeeds.

Definition 18 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF and $S \subseteq \mathfrak{P}$ a set of t-profiles. Let $I \in \mathsf{Part}(\tau_{\Phi}(\mathcal{A}))$ The set of defended intervals for $\mathcal{A}$ at I, denoted $\Delta^{S}_{\mathcal{A}}(I)$, is defined as:*

$$\Delta^{S}_{\mathcal{A}}(I) = \Cap\delta^{\mathcal{X}}_{\mathcal{A}}(S) \Cap \{I\} : \mathcal{X} \in AttAtts^{I}_{\Phi}$$

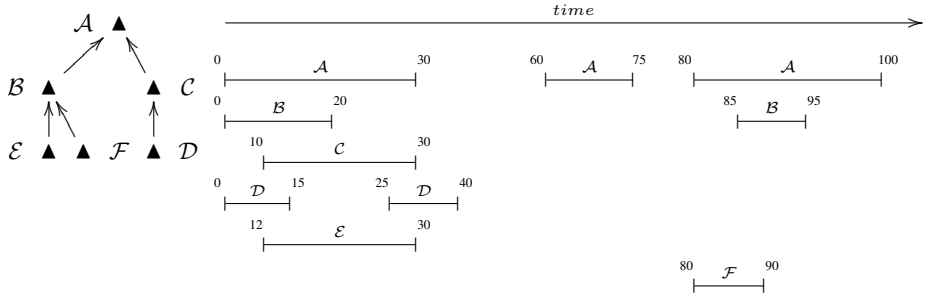


Fig. 4. Timed Framework to ilustrate defenses

Example 8 *Consider the timed argumentation framework of Figure 4. If we consider the argument $\mathcal{A}$ is clear that it is threatened by $\mathcal{B}$ in $\{[0,20],[85,95]\}$ and in $\{[10,30]\}$. The partition of the union of this sets is $\{[0,9],[10,20],[21,30]\}$. This is important because in these small intervals the argument $\mathcal{A}$ needs defense against different attackers, in particular in $[10,20]$ it requieres defense against $\mathcal{B}$ and $\mathcal{C}$.*

Consider the set $S = \{\langle \mathcal{D}, \{[10,15],[25,30]\}\rangle, \langle \mathcal{E}, \{[16,30]\}\rangle\}$. The set $\Delta^S_{\mathcal{A}}([0,9])$ is the emptyset since $\delta^{\mathcal{B}}_{\mathcal{A}}(S) = \{[16,20]\}$ but $\delta^{\mathcal{B}}_{\mathcal{A}}(S) \Cap \{[0,9]\} = \emptyset$. The set $\Delta^S_{\mathcal{A}}([21,30])$ is $\{[25,30]\}$ since $\delta^{\mathcal{C}}_{\mathcal{A}}(S) = \{[10,15],[25,30]\}$ so $\delta^{\mathcal{C}}_{\mathcal{A}}(S) \Cap \{[21,30]\} = \{[25,30]\}$. Finally $\Delta^S_{\mathcal{A}}([10,20]) = \emptyset$ since S fails in providing defense for both attacker at the same time, so the intersection is empty. You can see that

$\delta^{\mathcal{C}}_{\mathcal{A}}(S) \Cap \{[10,20]\} = \{[10,15],[25,30]\} \Cap \{[10,20]\} = \{[10,15]\}$ *and* $\delta^{\mathcal{B}}_{\mathcal{A}}(S) \Cap \{[10,20]\} = \{[16,20]\} \Cap \{[10,20]\} = \{[16,20]\}$.

The set $\Delta^S_{\mathcal{A}}([10,20])$ is defined as the intersection of this last sets, i.e.$\{[10,15]\} \Cap \{[16,20]\}$*, which is clearly the emptyset.*

Finally the argument is defended in the union of the moments defined in every interval of the partition of its threat-intervals.

Definition 19 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF and $S \subseteq \mathfrak{P}$ a set of t-profiles. The set of defended intervals for $\mathcal{A}$, denoted $\Delta^S_{\mathcal{A}}$, is:*

$\Delta^{\emptyset}_{\mathcal{A}} = \mathcal{A}v(\mathcal{A}) \stackrel{I}{-} \tau_{\Phi}(\mathcal{A})$
$\Delta^S_{\mathcal{A}} = \Delta^{\emptyset}_{\mathcal{A}} \cup \bigcup_{I \in \mathsf{Part}(\tau_{\Phi}(\mathcal{A}))} \Delta^S_{\mathcal{A}}(I)$ *when* $S \neq \emptyset$.

Following the analysis made in Example 8 we can determine $\Delta^S_{\mathcal{A}}$ which in this case is $\{[60,75],[80,100],[25,30]\}$. The set $\{[60,75],[80,100]\}$ represents the periods in which $\mathcal{A}$ is not attacked ($\Delta^{\emptyset}_{\mathcal{A}}$), and the set $\{[25,30]\}$ is the union of $\Delta^S_{\mathcal{A}}(I)$, for all $I \in \mathsf{Part}(\tau_{\Phi}(\mathcal{A}))$.

Definition 20 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF and $S \subseteq \mathfrak{P}$ be a set of t-profiles. An argument $\mathcal{A} \in Args$ is acceptable with respect to S if $\Delta^S_{\mathcal{A}} \neq \emptyset$. If $\mathcal{A}$ is acceptable, then it is acceptable at $\Delta^S_{\mathcal{A}}$. Its t-profile of acceptability is $\langle \mathcal{A}, \Delta^S_{\mathcal{A}}\rangle$*

Once this point is reached we can determine if some t-profile is defended or not.

Definition 21 *Let $\Phi = \langle Args, Atts, \mathcal{A}v\rangle$ be a TAF, $S \subseteq \mathfrak{P}$ be a set of t-profiles and a t-profile $\langle \mathcal{X}, \mathfrak{D}_{\mathcal{X}}\rangle$. The t-profile $\langle \mathcal{X}, \mathfrak{D}_{\mathcal{X}}\rangle$ is defended by S if $\mathfrak{D}_{\mathcal{X}}$ is included in $\Delta^{S}_{\mathcal{X}}$.*

Consider the timed argumentation framework of Figure 4 and the set

$S = \{\langle \mathcal{D}, \{[10, 15], [25, 30]\}\rangle, \langle \mathcal{E}, \{[16, 30]\}\rangle\}$.

Argument $\mathcal{A}$ is acceptable with respect to S and its t-profile of acceptability is

$\langle \mathcal{A}, \{[60, 75], [80, 100], [25, 30]\}\rangle$.

As a consequence the t-profile $\langle \mathcal{A}, \{[80, 90], [28, 30]\}\rangle$ is defended by S while the t-profile $\langle \mathcal{A}, \mathcal{A}v\mathcal{A}\rangle$ is not.

Proposition 1 *If an argument $\mathcal{A}$ belongs to a stable extension S with t-profile $\langle \mathcal{C}, \mathfrak{D}_{\mathcal{X}}\rangle$, then $\langle \mathcal{C}, \mathfrak{D}_{\mathcal{X}}\rangle$ is defended by S.*

Proof: If $\langle \mathcal{X}, \mathfrak{D}_{\mathcal{X}}\rangle \in S$ then either

- *$\mathcal{X}$ has no attackers in $\mathfrak{D}_{\mathcal{X}}$, and then $\langle \mathcal{X}, \mathfrak{D}_{\mathcal{X}}\rangle$ is acceptable with respect to S.*
- *$\mathcal{X}$ has an attacker $\mathcal{Y}$ in $\mathfrak{D}_{\mathcal{X}}$. Clearly, since S is conflict free, then no t-profile $\langle \mathcal{Y}, \mathfrak{D}_{\mathcal{Y}}\rangle$ is included in S such that $\mathfrak{D}_{\mathcal{X}}$ and $\mathfrak{D}_{\mathcal{Y}}$ have moments in common. Then, there is a t-profile $t = \langle \mathcal{Y}, \mathfrak{D}_{\mathcal{Y}}\rangle$ outside S that attacks $\mathcal{X}$, $i.e.\mathfrak{D}_{\mathcal{X}}$ and $\mathfrak{D}_{\mathcal{Y}}$ have moments in common. Since t is not in S, then S attacks t. This means that every interval in $\mathfrak{D}_{\mathcal{Y}}$ has moments in common ($i.e.$the intersection is not empty) t-profiles in S. But then, the same time $\mathcal{X}$ is attacked by $\mathcal{Y}$, it is defended by another argument in S. Thus, S defends $\mathcal{X}$ in $\mathfrak{D}_{\mathcal{X}}$.* □

Since arguments in a stable extension are defended by the extension, the following proposition is induced.

Proposition 2 *A stable extension S of a timed argumentation framework is admissible.*

It is important to preserve the rationality behind classical semantics for argumentation frameworks and the new semantics for timed argumentation. This means that whenever a relation between classical semantic notions is established, the same relation is expected to be found in a timed context. This is perhaps the most difficult aspect of the task of elaborating semantic notions for new timed argumentation frameworks, and it is an active part of this line of research. In the following section conclusions and future work are discussed.

6 Conclusions and future work

In this work we presented an extension of previously defined Timed Argumentation Frameworks in which arguments with more than one availability interval are considered. These arguments are called *intermittent arguments*, and are temporaly available with some repeated interruptions in time. Using this extended timed argumentation framework, we studied the notion of *stable extension*, which requires the consideration of time as a new dimension, leading to the definition of *t-profiles* of timed arguments.

Future work has several directions. The relation between different timed semantics needs to be addressed. In classical argumentation there are conditions for which several semantics coincide. For instance, in well-formed argumentation frameworks [10] there is only one extension that is grounded, preferred and stable. The notion of *well-formed* applied to timed argumentation frameworks is being analyzed. We are also interested in the evolution of the framework through time. For a given semantic notion $\mathcal{S}$, such as stable as presented in this paper, there may be intervals of time in which the extensions induced by $\mathcal{S}$ do not change, even when some arguments become or cease to be available during these intervals. These are called *steady periods* of the framework and are also an interesting topic. It may be used to model *eras of thinking* for a rational agent or a society, and the impact of including new arguments. New semantics elaborations based in this notion are being studied.

References

1. Allen, J.: Maintaining knowledge about temporal intervals. Communications of the ACM (26), 832–843 (1983)
2. Amgoud, L., Cayrol, C.: A reasoning model based on the production of acceptable arguments. In: Annals of Mathematics and Artificial Intelligence
3. Amgoud, L., Cayrol, C.: On the acceptability of arguments in preference-based argumentation. In: 14th Conference on Uncertainty in Artificial Intelligence (UAI'98). pp. 1–7. Morgan Kaufmann (1998)
4. Augusto, J.C., Simari, G.R.: Temporal defeasible reasoning. Knowl. Inf. Syst. 3(3), 287–318 (2001)
5. Baroni, P., Giacomin, M.: Resolution-based argumentation semantics. In: Proc. of 2nd International Conf. on Computational Models of Argument (COMMA 2008). pp. 25–36 (2008)
6. Bench-Capon, T.: Value-based argumentation frameworks. In: Proc. of Nonmonotonic Reasoning. pp. 444–453 (2002)
7. Cobo, M., Martinez, D., Simari, G.: An approach to timed abstract argumentation. In: Proc. of Int. Workshop of Non-monotonic Reasoning 2010 (2010)
8. Cobo, M., Martinez, D., Simari, G.: On admissibility in timed abstract argumentation frameworks. In: Coelho, H., Studer, R., Wooldridge, M. (eds.) ECAI. Frontiers in Artificial Intelligence and Applications, vol. 215, pp. 1007–1008. IOS Press (2010)
9. Dechter, R., Meiri, I., Pearl, J.: Temporal constaints networks. In: Proceedings KR '89. pp. 83–93 (1989)
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
11. Jakobovits, H.: Robust semantics for argumentation frameworks. Journal of Logic and Computation 9(2), 215–261 (1999)
12. Mann, N., Hunter, A.: Argumentation using temporal knowledge. In: Proc. of 2nd International Conf. on Computational Models of Argument (COMMA 2008). pp. 204–215 (2008)
13. Martínez, D.C., García, A.J., Simari, G.R.: Modelling well-structured argumentation lines. In: Proc. of XX IJCAI-2007. pp. 465–470 (2007)
14. Meiri, I.: Combining qualitative and quantitative contraints in temporal reasoning. In: Proceedings of AAAI '92. pp. 260–267 (1992)
15. Vreeswijk, G.A.W.: Abstract argumentation systems. Artificial Intelligence 90(1–2), 225–279 (1997)

Probabilistic Argumentation Frameworks

Hengfei Li[1], Nir Oren[1], and Timothy J. Norman[1]

Department of Computing Science, University of Aberdeen, Aberdeen, AB24 3UE, Scotland
{h.li,n.oren,t.j.norman}@abdn.ac.uk

Abstract. In this paper, we extend Dung's seminal argument framework to form a probabilistic argument framework by associating probabilities with arguments and defeats. We then compute the likelihood of some set of arguments appearing within an arbitrary argument framework induced from this probabilistic framework. We show that the complexity of computing this likelihood precisely is exponential in the number of arguments and defeats, and thus describe an approximate approach to computing these likelihoods based on Monte-Carlo simulation. Evaluating the latter approach against the exact approach shows significant computational savings. Our probabilistic argument framework is applicable to a number of real world problems; we show its utility by applying it to the problem of coalition formation.

1 Introduction

Likelihoods and probabilities form a cornerstone of reasoning in complex domains. When argumentation is used as a form of defeasible reasoning, uncertainty can affect the decisions reached during the reasoning process [1]. Uncertainty can also affect applications of argumentation technologies in other ways. For example, in the context of a dialogue, uncertainty regarding the knowledge of participants can affect both the dialogue outcome, and the utterances the participants choose to make. Furthermore, if uncertainty is viewed as a proxy for argument strength, questions immediately arise regarding argument interaction and the strength of conclusions given an argument system.

In this paper we examine the role of probabilities in an abstract argument framework. Within such a framework, an argumentation semantics defines a method by which a set of justified arguments can be deduced. As a reasoning approach, a semantics takes an argumentation framework as its knowledge base and produces a set of justified arguments as its output. The problem we address thus involves identifying the effects of probabilities on argument justification.

At the intuitive level, our approach is relatively simple. Starting with Dung's abstract argumentation framework[2] as its base[1], we assign probabilities to arguments and defeats. These probabilities represent the likelihood of existence of a specific argument or defeat, and thus capture the uncertainties inherent in the argument system. Within such a *probabilistic argument framework* (abbreviated PrAF), all possible arguments neither definitely exist, nor completely disappear. Instead, all elements of the

[1] Though as discussed in Section 6, our techniques are applicable to nearly any other argumentation framework.

framework have a different chance of existing. The semantics of such a framework then identify the likelihood of different sets of arguments being justified according to different types of extensions.

Now, since we are interested in the likelihood of a set of arguments being justified we are, in a sense, reversing the standard semantics of argumentation. Rather than identifying *which* arguments are in some sense compatible, we are instead identifying a set of arguments and asking what their likelihood of being compatible is (with respect to the other arguments, defeats and probabilities which make up the framework). Answering this type of question has a number of real world applications, including to the domains of trust and reputation [3] and coalition formation [4].

As we show, a naïve approach to computing the likelihood of some set of arguments being justified within a probabilistic argumentation framework based on the standard laws of probability has exponential computational complexity with respect to the number of arguments even in situations where the underlying semantics has linear complexity. Given that this is impractical for most real-life scenarios we propose, and evaluate, an approximation method based on the idea of Monte-Carlo simulation for calculating the likelihood of a set of arguments being justified.

The remainder of this paper is structured as follows. In the next section, we describe and formally define probabilistic argumentation frameworks, and explain the naïve method for performing computations over such PrAFs. Section 3 then details the Monte-Carlo simulation based approximation method. In Section 4, we empirically evaluate the performance of both of our techniques. An illustrative application for which PrAFs are particularly applicable is detailed in Section 5, following which Section 6 provides a more general discussion together with suggestions for future work. We then summarise our results and conclude the paper in Section 7.

2 Probabilistic Argumentation Frameworks

In this section, we extend Dung's argumentation framework to include uncertainty with respect to arguments and defeats. Essentially, we assign a probability to all elements of the argument framework, namely to every argument and defeat relation. Note that our approach can be trivially extended to other frameworks such as bipolar [5], evidential [6] and value based argumentation frameworks [7] as probabilities can be also assigned to other elements such as support relations (in the case of bipolar frameworks) or preferences. We intend to investigate such extensions in future work, and deal only with Dung argument frameworks in this paper. We therefore begin by briefly describing Dung's system, following which we discuss our extensions and methods for reasoning about probabilistic frameworks.

Definition 1. ***(Dung Argumentation Framework)*** *A Dung argumentation framework DAF is a pair (Arg, Def) where Arg is a set of arguments, and $Def \subseteq Arg \times Arg$ is a defeats relation.*

A set of arguments S is conflict-free *if $\nexists a, b \in S$ such that $(a, b) \in Def$. An argument a is* acceptable *with respect to a set of arguments S iff $\forall b \in Arg$ such that $(b, a) \in Def$, $\exists c \in Arg$ such that $(c, b) \in Def$. A set of arguments S is* admissible *iff it is conflict free and all its arguments are acceptable with respect to S.*

From these definitions, different semantics have been defined [8]. The purpose of these semantics is to identify sets of arguments which are, in some intuitive sense, compatible with each other. For example, the grounded semantics yield a single extension which is the least fixed point of the characteristic function $F_{AF}(S) = \{a | a \in Arg \text{ is acceptable w.r.t } S\}$. In the remainder of this paper, we will concentrate on the grounded semantics due to its computational tractability [9].

2.1 Formalising Probabilistic Argumentation Frameworks

Now a probabilistic argumentation framework extends Dung's argument framework by associating a likelihood with each argument and defeat in the original system. Intuitively, a PrAF represents an entire set of DAFs that exist *in potentia*. A specific DAF can then has a certain likelihood of being *induced* from the PrAF.

Definition 2. ***(Probabilistic Argumentation Framework)*** *A Probabilistic Argumentation framework $PrAF$ is a tuple (A, P_A, D, P_D) where (A, D) is a DAF, $P_A : A \rightarrow (0 : 1]$ and $P_D : D \rightarrow (0 : 1]$.*

The functions P_A and P_D map individual arguments, and defeats to likelihood values. These represent the likelihood of existence of an argument within an arbitrary DAF induced from the PrAF. As discussed below, P_D is, implicitly, a conditional probability. It should be noted that the lower bound of these probabilities is not 0 (but approaches it in the limit). This requirement exists because any argument or defeat with a likelihood of 0 cannot ever appear within a DAF induced from the PrAF, and is thus redundant.

A PrAF represents the set of all DAFs that can potentially be created from it. We call this creation process the *inducement* of a DAF from the PrAF. All arguments and defeats with a likelihood of 1 will be found in the induced DAF, which can then contain additional arguments and defeats, as specified by the following definition.

Definition 3. ***(Inducing a DAF from a PrAF)*** *A Dung argument framework $AF = (Arg, Def)$ is said to be induced from a probabilistic argumentation framework $PrAF = (A, P_A, D, P_D)$ iff all of the following hold:*

- $Arg \subseteq A$
- $Def \subseteq D \cap (Arg \times Arg)$
- $\forall a \in A$ *such that* $P_A(a) = 1$, $a \in Arg$
- $\forall (f, t) \in D$ *such that* $P_D((f, t)) = 1$ *and* $P_A(f) = P_A(t) = 1$, $(f, t) \in Def$

We write $I(PrAF)$to represent the set of all DAFs that can be induced from $PrAF$.

A DAF induced from a PrAF thus contains a subset of the arguments found in the PrAF, together with a subset of the defeats found in the PrAF, subject to these defeats containing only arguments found within the induced DAF. The process of inducing a DAF eliminates information regarding likelihoods found in the original PrAF.

Now, consider a situation where a number of entities are participating in a dialogue, and one of them (labelled α) would like to compute what conclusions might be drawn at the end of this interaction. Let us assume that α has arguments a and b in its knowledge base, and it believes that the other dialogue participants have arguments c and d in

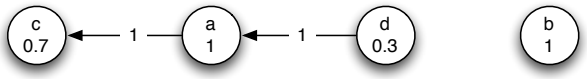


Fig. 1. A graphical depiction of a PrAF.

their knowledge base. This belief is however uncertain; c is believed to be known by the others with a likelihood of 0.7, and d with a likelihood of 0.3. Now let us assume that argument a defeats c and d defeats a. For simplicity, we assume that these defeat relations have no uncertainty associated with them (i.e. $P_D = 1$ for each of them). Formally, this can be represented by the following PrAF, illustrated in Figure 1:

$$(\{a, b, c, d\}, \{(a, 1), (b, 1), (c, 0.7), (d, 0.3)\}, \{(a, c), (d, a)\}, \{((a, c), 1), ((d, a), 1)\})$$

Given this PrAF, we can induce the following DAFs:

$$(\{a, b\}, \{\}), \qquad (\{a, b, c\}, \{(a, c)\}),$$
$$(\{a, b, d\}, \{(d, a)\}), (\{a, b, c, d\}, \{(a, c), (d, a)\})$$

Clearly, b appears in the grounded extension of all of these DAFs, while a appears in the grounded extension of 3 out of 4 induced DAFs. Now, α might want to identify the likelihood of a being justified (i.e. in the grounded extension) at the end of the dialogue, perhaps to decide whether to advance it or not (assuming that advancing an argument has some associated utility cost [10]).

2.2 Probabilistic Justification

Our goal is to compute the likelihood that some set of arguments exists and is justified according to some semantics within the DAFs induced from a PrAF. This likelihood can be obtained from the basic laws of probability, and we detail this procedure next. We make one critical simplifying assumption, namely that the likelihood of one argument (defeat) appearing in an induced DAF is independent of the likelihood of some other argument (defeat) appearing[2]. With this assumption in hand, we begin by computing the likelihood of some DAF being induced from the PrAF.

As mentioned earlier, the P_D relation associates a *conditional* probability with each possible defeat. That is, for some arguments a, b

$$P_D(a, b) = P((a, b) \in Def | a, b \in Arg) \text{ for the induced DAF } (Arg, Def)$$

Informally, the probability of some DAF AF being induced from a PrAF can be computed via the joint probabilities of the arguments and defeat relations appearing in AF. In order to formalise this concept compactly, we must identify the set of defeats that *may* appear in an induced DAF. We label this set as $DefA$. Given a DAF with arguments $Args$, and a $PrAF$ containing defeats D

$$DefA = \{(a, b) | a, b \in Args \text{ and } (a, b) \in D\}$$

[2] Relaxing this assumption will be left to future work.

This allows us to compute the probability of some DAF AF being induced from a PrAF, written $P^I_{PrAF}(AF)$, by computing the joint probabilities of independent variables as follows:

$$P^I_{PrAF}(AF) = \prod_{a \in Arg} P_A(a) \prod_{a \in A \setminus Arg} (1 - P_A(a)) \prod_{d \in Def} P_D(d) \prod_{d \in DefA \setminus Def} (1 - P_D(d)) \tag{1}$$

Applying this to our earlier example, $P^I_{PrAF}((\{a, b\}, \{\})) = 0.21$.

Proposition 1. *The sum of probabilities of all DAFs that can be induced from an arbitrary $PrAF$ is 1. That is, $\sum_{a \in I(PrAF)} P^I_{PrAF}(a) = 1$.*

Now our goal is to identify the likelihood of some set of arguments being consistent with respect to some set of argumentation semantics. Such a semantics may return one or many extensions for a given argument framework, and we formalise our notion of consistency through the definition of a *semantic evaluation function*, $\xi^S(AF, X)$ which returns $true$ if and only if the set of arguments X is deemed consistent using the semantics S when evaluated over the argument framework AF. Thus, for example $\xi^G(AF, X)$ could return true if the set of arguments X appears as a subset of the grounded extension of AF.

Then, following on from Proposition 1, given some $PrAF$, the likelihood of X being consistent according to the semantics S is defined as follows:

$$P_{PrAF}(X) = \sum_{AF \in I(PrAF)} P^I_{PrAF}(a) \text{ where } \xi^S(AF, X) = true \tag{2}$$

Referring again to our earlier example, $P_{PrAF}(\{a, b\}) = 0.7$.

While we can utilise Equations 1 and 2 to compute the exact likelihood of a set of arguments being justified with regards to some semantics, the size of the set of possible DAFs which can be induced from a PrAF grows exponentially with regards to the number of arguments and defeats within the PrAF, resulting in exponential time complexity (not including the computational costs associated with computing the results of ξ^S). This is clearly impractical for a large set of arguments, and in the next section, we examine an approximate method for determining these likelihoods.

3 Approximate Solutions in Probabilistic Argumentation Frameworks

In this section we describe a Monte-Carlo simulation based approach to computing $P_{PrAF}(X)$ for an arbitrary set of arguments X. At an abstract level, a Monte-Carlo simulation operates by repeatedly sampling a distribution many times in order to approximate it. More specifically, such a simulation has three basic steps. First, given a possible set of inputs, a subset of these inputs is selected according to some probability distribution. Second, some computation is performed using the selected inputs. Finally, the results of repeating the first two steps multiple times is aggregated. Monte-Carlo

Algorithm 1 An algorithm to approximate $P_{PrAF}(X)$.

Require: A Probabilistic Argumentation Framework $PrAF = (A, P_A, D, P_D)$
Require: A set of arguments $X \subseteq A$
Require: A number of trials $N \in \mathbb{N}$
Require: A semantic evaluation function, ξ^S
1: $Count = 0$
2: **for** $I = 0$ to N **do**
3: $Arg = Def = \{\}$
4: **for all** $a \in A$ **do**
5: Generate a random number r such that $r \in [0, 1]$
6: **if** $P_A(a) \geq r$ **then**
7: $Arg = Arg \cup \{a\}$
8: **end if**
9: **end for**
10: **for all** $(f, t) \in D$ such that $f, t \in Arg$ **do**
11: Generate a random number r such that $r \in [0, 1]$
12: **if** $P_D((f, t)) \geq r$ **then**
13: $Def = Def \cup \{(f, t)\}$
14: **end if**
15: **end for**
16: **if** $\xi^S((Arg, Def), X) = true$ **then**
17: $Count = Count + 1$
18: **end if**
19: **end for**
20: **return** $Count/N$

simulation has a long history, and has been applied to a variety of computationally difficult problems including inference in Bayesian Networks [11], reinforcement learning [12] and computer game playing [13].

In this context of probabilistic argumentation frameworks, this process involves randomly inducing DAFs from a PrAF, with the likelihood of an arbitrary DAF being induced being dependant on the underlying probability distribution of its individual members. We thus sample the space of possible DAFs in a way that approximates the DAFs true distribution in the probability space.

The only source of uncertainty in Equation 2 lies in P^I_{PrAF} which in turn depends only on the probabilities found in the underlying PrAF. Therefore, in order to approximate $P_{PrAF}(X)$ we need only sample the space of arguments and defeats found in the PrAF. Algorithm 1 describes this process more precisely.

The algorithm samples N DAFs from the set of inducible DAFs. A single DAF is generated by randomly selecting arguments and defeats according to their likelihood of appearance (Lines 4-7 and 10-14 respectively). This resultant DAF is then evaluated for the presence of X through the ξ^S function (Line 16), and if this function holds, the DAF is counted. $P_{PrAF}(X)$ is finally approximated as the ratio of the total number of DAFs in which $\xi^S(X)$ holds to the number DAFs sampled (Line 20).

The following proposition states that as our number of trials increases, the error in our approximation of $P_{PrAF}(X)$ shrinks.

Proposition 2. *If we denote the output of Algorithm 1 as $P'_{PrAF}(X)$, then as $N \to \infty$, $P_{PrAF}(X) - P'_{PrAF}(X) \to 0$. More specifically, there is some $N \in \mathbb{Z}^+$ and $\epsilon \in \mathbb{R}^+$ such that for all $M > N$, if M trials are run, $|P_{PrAF}(X) - P'_{PrAF}(X)| < \epsilon$.*

This proposition means that our algorithm has an anytime property: it may be terminated at any time, and earlier terminations will still provide an approximation to the true probability, albeit with a greater error than would be provided from a later termination.

While this proposition provides some guarantees regarding the accuracy of our results given enough trials, it does not answer one critical question: how many trials must be run to ensure (with some level of confidence) that our approximation has only a small level of error?

In order to answer this question, we note that the results of a Monte-Carlo simulation can be seen as a normal distribution over possible values for $P_{PrAF}(X)$, and with $P'_{PrAF}(X)$ as its mean. Given this, we can make use of the notion of a *confidence interval* in order to answer our question. In statistics, a confidence level of l for a given a confidence interval CI and a mean p' can be read as stating that the true mean lies within $p' \pm CI$ with a likelihood of l. Such a confidence interval is dependant on the observed likelihood of an event and the number of trials used to make the observations. We can thus recast our problem to ask how many trials need to be run in order to ensure that the confidence interval around $P'_{PrAF}(X)$ (i.e. its error) is smaller than some value ϵ with some specific confidence level (e.g. 95%).

Probably the most common approach to computing such an interval is the normal approximation interval [14], which is defined as follows:

$$p' \pm z_{1-\alpha/2}\sqrt{\frac{p'(1-p')}{n}} \tag{3}$$

Here, p' is the observed mean, n is the number of trials, and $z_{1-\alpha/2}$ the $1-(\alpha/2)$ percentile of the normal distribution. In the experiments described in Section 4, we required a 95% confidence level, resulting in $z_{1-\alpha/2} = 1.96$. Inserting this value into 3, we obtain the following equation to compute the number of trials required to achieve an error level below ϵ:

$$N > \frac{p'(1-p')}{\epsilon^2}(1.96)^2 \tag{4}$$

However, this approximation is problematic in our situation as p' is either 0 or 1 after a single trail, which will break down the calculation. To overcome this problem, we utilise the Agresti-Coull interval [15] instead. The general form of this interval is the same as that of Equation 3. However, the values of n and p' are computed differently:

$$n = N + z^2_{1-\alpha/2} \qquad p' = \frac{X + (z^2_{1-\alpha/2}/2)}{n}$$

Here, N is the number of trials and X is the number of "successes" observed. The Agresti-Coull method thus perturbs the true number of trials and probability of success slightly, and ensures that p' will not be 0. For the 95% confidence level, we can approximate $z_{1-\alpha/2} = 1.96$ with the value 2, leading to the following equation for computing the number of trials required to achieve an error level below ϵ:

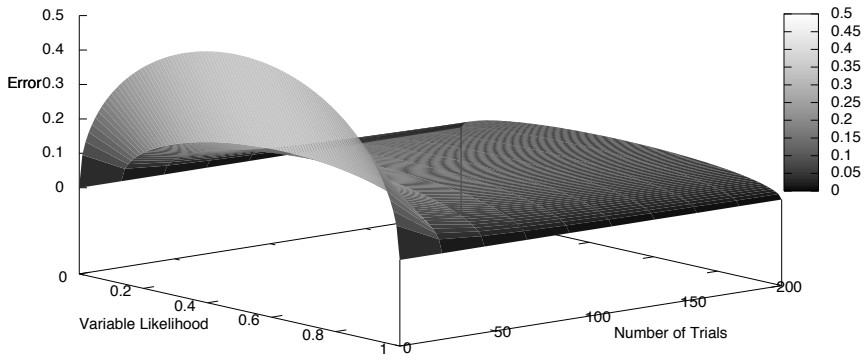


Fig. 2. The relationship between likelihood of a variable, the number of observations made and the error in the observed likelihood.

$$N > \frac{4p'(1-p')}{\epsilon^2} - 4 \tag{5}$$

Figure 2 provides a plot of this function. As seen here, initially, as the number of trials increase, the error falls off rapidly. However, this shrinking of the error quickly ceases, and additional trials serve to reduce the error by only a small amount. It should also be noted that the likelihoods of variables with extreme values (i.e. near 0 or 1) can be approximated far more quickly than variables with values near 0.5.

Given a desired error level ϵ and confidence level, Equation 5 provides us with a new stopping condition for Algorithm 1. The `for` loop of Line 2 can be substituted for a `while` loop which computes whether the expected error level falls below ϵ given the number of iterations that have been run so far. If this is the case, the loop can end, and the algorithm will terminate.

4 Evaluation

We have described, given some PrAF, two approaches to computing the likelihood of a chosen set of arguments being justified with respect to some semantics. While it is clear that the exact approach is exponential in complexity, it is useful to identify the approximate number of arguments in a PrAF at which point this becomes impractical. Similarly, in order to use it in real world settings, the approximate running time of the Monte-Carlo based approach must also be evaluated.

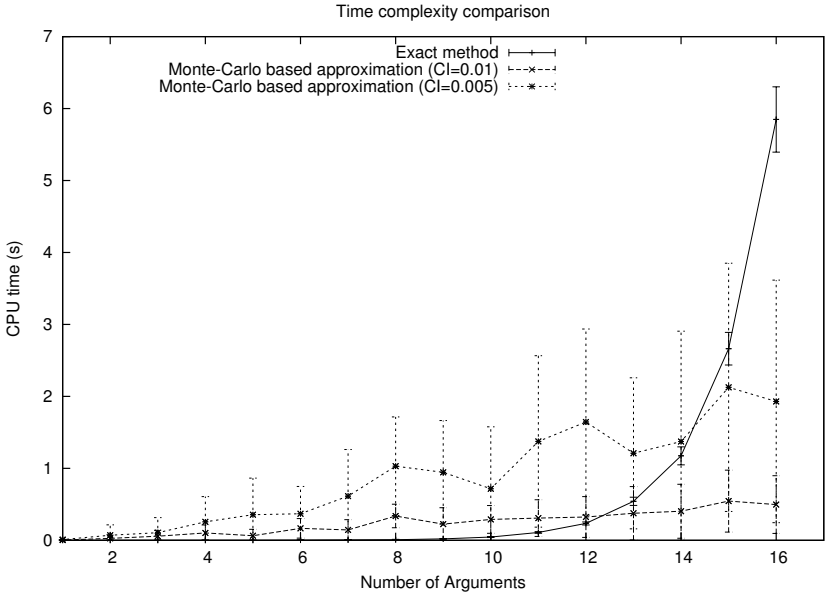


Fig. 3. Comparison of runtimes between the exact and Monte-Carlo based approaches. Error bars indicate 1 standard deviation.

We implemented both of the approaches described in the paper using SWI-Prolog[3]. For simplicity, we associated likelihood values only with arguments within the PrAF; all defeats had a likelihood of 1. The goal of our first experiment was to identify the effects of differently sized PrAFs on the runtimes of the exact approach, and of the Monte-Carlo based approach with different error tolerances ($\epsilon = 0.01$ and $\epsilon = 0.005$). In order to do so, we evaluated the approaches on identical PrAFs with each PrAF containing between 1 and 16 arguments. Our semantic evaluation function $\xi^S(X)$ computed whether X formed a subset of the grounded extension. We ran our experiment 20 times for each unique number of arguments, and Figure 3 shows our results. As expected, the time taken by the exact approach increases exponentially; the Monte-Carlo based approaches overtake the exact approach at around 13 (when $\epsilon = 0.01$) and 15 (when $\epsilon = 0.005$) arguments. The introduction of uncertainty into the defeats relation would increase the number of DAFs that can be induced from the PrAF meaning that our results, in a sense, represent the best case for the exact approach.

In order to more closely examine the effect of ϵ and the size of the PrAF on the performance of our approximate algorithm, Figure 4 compares the average number of iterations, and runtime, required to achieve the desired level of accuracy against the number of arguments found in the PrAF. As expected, an increase in the size of the PrAF has only a linear effect on the runtime of our algorithm. This increase occurs due

[3] http://www.swi-prolog.org

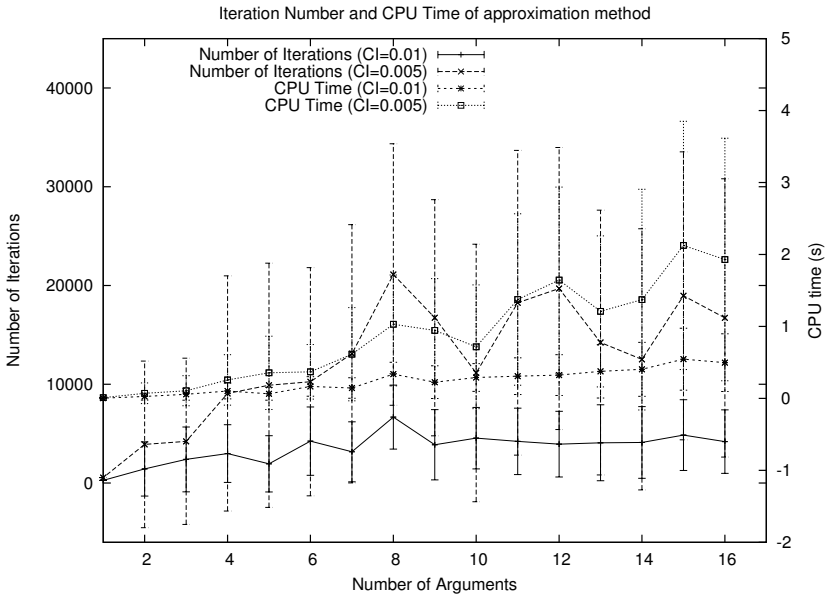


Fig. 4. Comparison of runtimes and number of iterations between the Monte-Carlo based approaches with different ϵ values. Error bars indicate 1 standard deviation.

to an increase in the time required to compute the grounded extension (as computing this has linear complexity) rather than additional iterations[4]. This result can clearly be seen from Figure 2; the number of iterations required to obtain a certain error level do not depend on the number of arguments and defeats in the PrAF, but only on the joint probabilities obtained from the PrAF. Figure 2 also predicts another result clearly seen in Figure 4, namely that as the permitted error shrinks, the standard deviation of the number of iterations that must be executed grows. This is because the number of iterations required to obtain an error ϵ when the joint probability in question is close to 0 or 1 grows much more slowly than when the probability is close to 0.5.

Finally, it can also be seen that there exists some variability between the number of iterations required and the time to execute these iterations; this arises due to the underlying Prolog implementation, and the number of iterations is thus a better indicator of algorithm performance.

[4] Of course, if preferred rather than grounded semantics were used, in the worst case, the graph would reflect an exponential increase in running time. Ultimately, it should be noted that utilising Monte-Carlo based approximation only increases time complexity by a multiplicative constant.

5 Applying PrAFs to Coalition Formation

In this section, we describe an application of our approach to a real world problem, namely coalition formation. According to [4], "Coalition formation is a fundamental form of interaction that allows the creation of coherent groupings of distinct, autonomous, agents in order to efficiently achieve their individual or collective goals". Coalition formation is applicable to both virtual domains such as e-commerce (where virtual organisations can form in order to satisfy a customer's requirements [16]), and physical domains where, for example, a search and rescue team must be composed of agents with specific capabilities in order to be able to undertake some mission [17].

Most approaches to coalition formation treat the problem as one of utility maximisation; agents will join a coalition if being in the coalition will yield a greater utility than not. Here, we show how to address the problem of coalition formation from a very different perspective. This different perspective allows us to explore an aspect of the social dimensions involved in coalition formation; i.e. the notion of whether or not an individual's presence in a coalition may influence another's membership. More specifically, we model a system containing agents with different capabilities, each of which has a prior probability of joining the coalition, and a probability of preventing other potential coalition members from joining the coalition. We would then like to determine what the probability of a coalition forming which is capable of achieving some task.

We can model the coalition formation problem using PrAFs as follows: we associate agents with nodes (arguments in the PrAF). Each node's P_A is associated to the agent's prior probability of joining the coalition. Defeats represent the likelihood of the presence of one member in the coalition preventing another member from joining. Computing the likelihood of a coalition containing specific members can then be computed by computing P_{PrAF}.

As an illustrative example[5], consider a small mercenary team consisting of a leader h, a pilot m, a mechanic b and an expert in persuasion f. Now assume that the presence of the pilot cannot be tolerated by the mechanic, and that f is generally disliked by other team members (to varying degrees); f's presence in a coalition will increase the risk that others will not join. Finally, assume that both f and h are often busy, and occasionally cannot join the team. This situation can be represented by the PrAF shown in Figure 5.

The techniques presented in this paper can then be used to compute the likelihood of a specific team being formed, for example consisting of h, m and b (this would be 0.016128+0.056=0.072128. The first value is the likelihood of the full team forming, and the second, the probability of the team forming without f). Given this likelihood, the user might decide to change their goals, or add new agents to the system to increase the chances of success.

The discussion thus far has concentrated on determining whether a coalition can be formed containing some specific set of agents. However, in the context of coalition formation, the goal is often to form a coalition consisting of agents taking on some set of specific roles (e.g. a coalition requires two mechanics and a pilot). One approach to determining the likelihood of forming such a coalition involves identifying all possible

[5] This example is based on the characters from a 1980's television series.

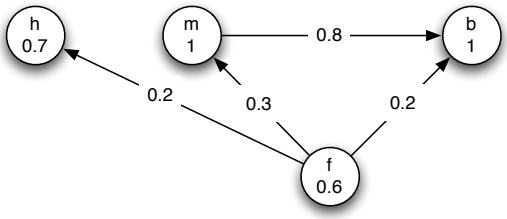


Fig. 5. A PrAF representing the coalition formation example.

ways in which such a coalition can form, and combining the probabilities of each individual coalition to obtain an aggregate probability. However, this approach does not scale well as the size of the system increases. We intend to investigate techniques for dealing with this issue in future work, and discuss it further in the next section.

6 Discussion and Future Work

The use of likelihood in different facets of argumentation for modelling strength or uncertainty of arguments has a long and rich history. Most commonly, such likelihood measures have served as a proxy for argument strength [1, 18], and have been used to compute the likelihood of some conclusion holding using a variety of different methods. [19, 20] consider probability in the construction of argument, they derive a probability of argument from the probabilities of its premises by different methods. In the context of abstract argumentation frameworks, some approaches for modelling uncertainties such as assigning a numerical values [21] or preference ordering [22] to attacks have been developed. Another approach to computing argument strength involves counting the number of subsets which meet the requirements of some (multiple status) semantics [23], and in which the argument under question appears. The ratio of this number to the total number of extensions then serves to act as a measure of strength for the argument. Our approach is similar in spirit to this latter work as we compute the likelihood of some subset of arguments appearing. However, the introduction of probabilities, through the definition of a PrAF, makes the approach applicable to both single and multiple status semantics, with the distinct advantage of the former's tractability. Another similar work is by Janssen et al [24] in which they use a value between 0 and 1 to model the strength of arguments and attacks. By utilising and underlying fuzzy logic semantics, they then describe to what degree an argument holds or attacks another argument.

In Section 5, we discussed one possible application for PrAFs, namely answering questions about the likelihood of a coalition with certain characteristics being formed. We discussed one shortcoming, namely the inability of the basic approach to deal with the notion of roles in coalition formation, and suggested one method for overcoming this shortcoming. Another more nuanced approach involves the use of resource bounded argumentation frameworks [25], which would allow us to place requirements on team composition via constraints, and thus also allow for more nuanced team formation.

Another shortcoming involves our underlying Dung based model wherein only defeats between arguments are modelled. Constructing a PrAF on top of a bipolar framework (e.g. [5, 6]) would allow us to cater for situations where one agent is more likely to enter into a coalition if some other agent will be present. Another way of achieving this would be to lift the independence assumption regarding the likelihood of argument and defeat relation likelihoods, and all of these form enticing possibilities for future work.

PrAFs and the techniques described in this paper can be applied to other argument frameworks and domains. For example, a value based argumentation framework (VAF) [7] provides a model of determining whether some set of arguments will be accepted by audiences containing agents with different preferences over the defeat relation. Constructing a PrAF on top of such a VAF can allow us to answer questions such as "what is the likelihood of all members in the audience accepting this argument". Clear applications of this include opponent modelling [10] and heuristics for argument [26–28]. Another interesting possibility lies in associating a probability distribution with the preferences of the audience within the VAF, allowing us to model the persuasive force of some set of arguments.

Apart from the coalition formation and argument strategy domains, the ideas associated with constructing and evaluating PrAFs can also play a role in other domains where the notion of the strength of an argument is relevant. For example, in the area of trust and reputation [3], PrAFs can be used to associate reputation information with individual agents. Distrust relationships (following [29]) or biases in trust relationships (following [30]) can then be constructed through the defeats relation , and, by using a bipolar framework, trust relationships can be created through support links. The resultant PrAF can then be used to compute the likelihood of some set of agents considering one another trustworthy.

7 Conclusions

In this paper we introduced *probabilistic argumentation frameworks*. These frameworks add the notion of likelihood to all elements of an abstract argument framework (in this paper, we concentrated on Dung argument frameworks, and thus associated likelihoods with arguments and defeats), and are used to determine the likelihood of some subset of arguments appearing within an extension. The exact method for determining this likelihood has exponential complexity, and is thus impractical for use with anything other than a small argumentation system. To overcome this limitation, we introduced a Monte-Carlo simulation based approach to approximate the likelihood. This latter technique scales up well, providing good results in a reasonable period of time, and has anytime properties, making it ideal for use in almost all situations.

PrAFs have applications to a myriad of domains. In this paper, we focused on one such domain, namely coalition formation, and described how PrAFs can be used to assist a system designer. While we have touched on the applications of PrAFs to other domains, and suggested a number of extensions to their basic representation, we intend to further explore their potential applicability to additional argumentation frameworks and application domains.

References

1. Pollock, J.L.: Cognitive Carpentry. Bradford/MIT Press (1995)
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) (1995) 321–357
3. Teacy, W.T.L., Patel, J., Jennings, N.R., Luck, M.: Travos: Trust and reputation in the context of inaccurate information sources. Autonomous Agents and Multi-Agent Systems **12**(2) (2006) 183–198
4. Rahwan, T.: Algorithms for Coalition Formation in Multi-Agent Systems. PhD thesis, University of Southampton (2007)
5. Cayrol, C., Lagasquie-Schiex, M.C.: Bipolar Abstract Argumentation Systems. In Rahwan, I., Simari, G., eds.: Argumentation in Artificial Intelligence. Springer, http://www.springerlink.com (2009) 65–84
6. Oren, N., Norman, T.J.: Semantics for evidence-based argumentation. In: Computational Models of Argument: Proceedings of COMMA 2008, Toulouse, France, May 28-30, 2008. (2008) 276–284
7. Bench-Capon, T.: Value based argumentation frameworks. In: Proceedings of the 9th International Workshop on Nonmonotonic Reasoning, Toulouse, France (2002) 444–453
8. Baroni, P., Giacomin, M.: Semantics of abstract argument systems. In Simari, G., Rahwan, I., eds.: Argumentation in Artificial Intelligence. Springer US (2009) 25–44
9. Dunne, P.E., Wooldridge, M.: Complexity of abstract argumentation. In Simari, G., Rahwan, I., eds.: Argumentation in Artificial Intelligence. Springer US (2009) 85–104
10. Oren, N., Norman, T.J.: Arguing using opponent models. In: Proceedings of the Sixth International Workshop on Argumentation in Multi-agent Systems, Budapest, Hungary (2009)
11. Mitchell, T.M.: Machine Learning. McGraw-Hill Higher Education (1997)
12. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning). The MIT Press (1998)
13. Coulom, R.: Efficient selectivity and backup operators in Monte-Carlo tree search. In van den Herik, J.H., Ciancarini, P., Donkers, j.H.H.L.M., eds.: Proceedings of the 5th International Conference on Computer and Games. Volume 4630/2007 of Lecture Notes in Computer Science., Turin, Italy, Springer (2006) 72–83
14. Lewicki, P., Hill, T.: Statistics: Methods and Applications. StatSoft Inc (2005)
15. Agresti, A., Coull, B.A.: Approximate is better than "exact" for interval estimation of binomial proportions. The American Statistician **52**(2) (1998) 119–126
16. Patel, J., Teacy, W.T.L., Jennings, N.R., Luck, M., Chalmers, S., Oren, N., Norman, T.J., Preece, A., Gray, P.M.D., Shercliff, G., Stockreisser, P.J., Shao, J., Gray, W.A., Fiddian, N.J., Thompson, S.: Agent-based virtual organisations for the grid. Multiagent and Grid Systems **1**(4) (2006) 237–249
17. Pechoucek, M., Marík, V., Bárta, J.: A knowledge-based approach to coalition formation. IEEE Intelligent Systems **17** (2002) 17–25
18. Gómez Lucero, M., Chesñevar, C., Simari, G.: Modelling argument accrual in possibilistic defeasible logic programming. In Sossai, C., Chemello, G., eds.: Symbolic and Quantitative Approaches to Reasoning with Uncertainty. Volume 5590 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2009) 131–143
19. Krause, P., Ambler, S., Elvang-Goransson, M., Fox, J.: A logic of argumentation for reasoning under uncertainty. Computational Intelligence **11**(1) (1995) 113–131
20. Kohlas, J., Haenni, R.: Assumption-based reasoning and probabilistic argumentation systems. Technical Report 96–07, Institute of Informatics, University of Fribourg, Switzerland (1996)

21. Dunne, P.E., Hunter, A., McBurney, P., Parsons, S., Wooldridge, M.: Weighted argument systems: Basic definitions, algorithms, and complexity results. Artificial Intelligence **175**(2) (2011) 457 – 486
22. Amgoud, L., Cayrol, C.: Inferring from inconsistency in preference-based argumentation frameworks. Journal of Automated Reasoning **29** (2002) 125–169
23. Baroni, P., Dunne, P.E., Giacomin, M.: On extension counting problems in argumentation frameworks. In: Proceeding of the 2010 conference on Computational Models of Argument: Proceedings of COMMA 2010, Amsterdam, The Netherlands, The Netherlands, IOS Press (2010) 63–74
24. Janssen, J., Cock, M.D., Vermeir, D.: Fuzzy argumentation frameworks. In: Proceedings of IMPU2008 (12th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems). (2008) 513–520
25. Rotstein, N., Oren, N., Norman, T.J.: Resource bounded argumentation frameworks. In: Proceedings of the First International Workshop on the Theory and Applications of Formal Argumentation. (2011) to appear
26. Oren, N., Norman, T.J., Preece, A.: Arguing with confidential information. In: Proceedings of the 18th European Conference on Artificial Intelligence, Riva del Garda, Italy (2006) 280–284
27. Riveret, R., Prakken, H., Rotolo, A., Sartor, G.: Heuristics in argumentation: A game theory investigation. In: Computational Models of Argument: Proceedings of COMMA 2008, Toulouse, France, May 28-30, 2008. (2008) 324–335
28. Emele, C.D., Norman, T.J., Parsons, S.: Argumentation strategies for plan resourcing. In: Proceedings of the Tenth International Conference on Autonomous Agents and Multiagent Systems. (2011)
29. Erriquez, E., van der Hoek, W., Wooldridge, M.: An abstract framework for reasoning about trust. In: Proceedings of the Tenth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-2011). (2011) 1085–1086
30. Burnett, C., Norman, T.J., Sycara, K.: Stereotypical trust and bias in dynamic multi-agent systems. ACM Transactions on Intelligent Systems and Technology (in press)

A Three-Layer Argumentation Framework

Paulo Maio and Nuno Silva

GECAD – School of Engineering – Polytechnic of Porto
Rua Dr. Bernardino de Almeida 431, 4200-072 Porto, Portugal
{pam, nps}@isep.ipp.pt

Abstract. Argumentation frameworks which are abstract are suitable for the study of independent properties of any specific aspect (e.g. arguments sceptical and credulous admissible) that are relevant for any argumentation context. However, its direct adoption on specific application contexts requires dealing with questions such as the argument structure, the argument categories, the conditions under which an attack/support is established between arguments, etc. This paper presents a generic argumentation framework which comprehends a conceptualization layer to capture the expressivity and semantics of the argumentation data employed in a specific context and simplifies its adoption by applications.

Keywords: Argumentation Frameworks, Bipolar Argumentation, Agents, MAS

1 Introduction

A crucial problem on BDI agents as described by Wooldridge [1] concerns what should be the agent beliefs and how those beliefs are used (i) to form new intentions, or (ii) to redraw/revise current intentions. On this matter, contributions of the argumentation research field may be exploited internally by BDI agents since argumentation can be used either for reasoning about what to believe (i.e. theoretical reasoning) and/or for deciding what to do (i.e. practical reasoning). Despite existing differences between both, according to [2], from a standpoint of first-personal reflection, a set of considerations for and against a particular conclusion are drawn on both. Yet, agents in multi-agent systems (MAS) may apply argumentation externally during interactions between agents, i.e. agents' dialogues (cf. [3] for details). Within this context, argumentation is seen as an activity where each participant tries to increase (or decrease) the acceptability of a given standpoint for the other participants by presenting arguments. Therefore, argumentation is foreseen as an adequate modeling formalism to reduce the gap between models governing the internal and external agent behavior.

In which concerns to argumentation, there is an abundance of relevant literature in argumentation and argumentation systems. With regards to argumentation modeling formalisms, the abstract argumentation frameworks AF [4], BAF [5] and VAF [6] are suitable to represent many different situations without being committed to any domain

of application. Due to their abstract nature they are also suitable for the study of independent properties of any specific aspect that are relevant for any argumentation context that can be captured and formalized accordingly. On the other hand, this abstract nature represents an expressiveness limitation to the direct adoption of specific application contexts [7]. To overcome this limitation, argumentation systems usually adopt an abstract argumentation framework and extend it in order to get a less abstract formalism, dealing in particular with (i) the construction of arguments and their structure, (ii) the conditions under which argument-relations (i.e. attack and/or support) are established, (iii) categories of arguments, etc. Nevertheless, these argumentation frameworks do not provide any machinery facilitating and governing how applications should instantiate the framework. As a result, a significant gap between abstract argumentation frameworks and applications exist.

This paper proposes a less abstract argumentation framework whose purpose is to reduce existing gaps between abstract argumentation frameworks and applications. For that, the proposed framework includes a conceptual layer for the specification of the semantics of argumentation data applied in a specific domain of application (e.g. e-commerce, legal reasoning and decision making) and defines a general and intuitive argument structure. In addition, the proposed framework exploits the conceptual information and the defined argument structure to automatically detect the existing argument-relations (i.e. attack and/or support). Despite having these new features, the proposed argumentation framework remains general, but less abstract than AF [4], BAF [5] and VAF [6]. Yet, to profit from the inherent suitability of abstract argumentation frameworks on the study of independent properties, information represented according to the proposed argumentation framework is easily transformed (or converted) to BAF [5].

The rest of the paper is organized as follows. The next section introduces background concepts about abstract argumentation frameworks. Section 3 presents the proposed argumentation framework, which is further complemented in section 0 with a process to automatically derive the attack and support relationships between arguments. Finally, Section 5 draws conclusions and discusses future work.

2 Abstract Argumentation Frameworks

This section briefly describes the main concepts of the most referenced abstract argumentation frameworks found in the literature: the Argumentation Framework proposed by Dung (AF) [4], the Value Argumentation Framework (VAF) [6] and the Bipolar Argumentation Framework (BAF) [5].

As proposed by Dung [4], the AF core entities are *Argument*, and a binary relation between arguments (R_{att}) as depicted in Fig. 1a. The R_{att} relation is known as the attack relation. An AF can be defined as a tuple $AF = (A, R_{att})$ where A is a set of arguments and R_{att} is a relation on A such that $R_{att} \subseteq A \times A$.

An AF instance may be represented by a directed graph whose nodes are arguments and edges represent the attack relation. For any two arguments, say a_1 and a_2, such that $a_1, a_2 \in A$, one says that a_1 attacks a_2 iif $(a_1, a_2) \in R_{att}$.

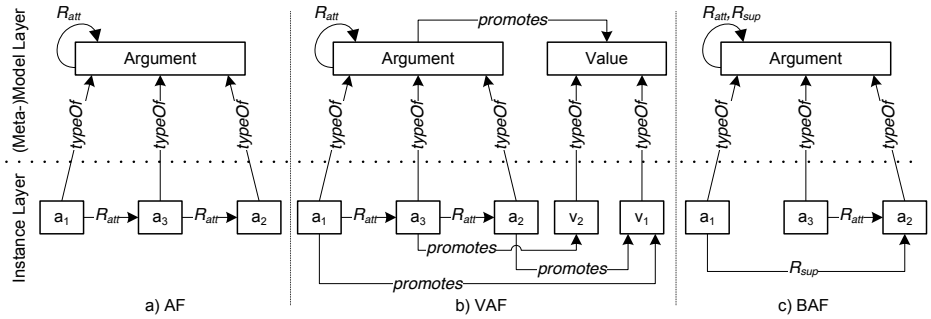


Fig. 1. The main concepts of abstract argumentation frameworks

In Dung's work attacks always succeed (i.e. it defeats the attacked arguments). Yet, one says that an argument y is attacked by a set of arguments S such that $S \subseteq A$ if S contains at least one argument attacking y. Grounded on that, the following notions were defined:

- An argument $a \in A$ is *acceptable* with respect to a set of arguments S, i.e. $acceptable(a, S)$, iif $\forall x: x \in A \ \wedge (x, a) \in R_{att} \rightarrow \exists y: y \in S \wedge (y, x) \in R_{att}$;
- A set of arguments S if *conflict-free* iif $\nexists x, y: x, y \in S \wedge (x, y) \in R_{att}$;
- A *conflict-free* set of arguments S is admissible iif $\forall x: x \in S \rightarrow acceptable(x, S)$;
- A set of arguments S is a *preferred extension* iif it is maximal (with respect to set inclusion) *admissible* set of A.

A *preferred extension* represents a consistent position within an AF instance, which is defensible against all attacks and cannot be further extended without introducing a conflict. Yet, multiple *preferred extensions* can exist in an AF instance due to the presence of cycles of *even* length in the graph. Given that, one considers that (i) an argument is *sceptical admissible* if it belongs to any *preferred extension* and (ii) an argument is *credulous admissible* if it belongs to at least one *preferred extension*.

While it is reasonable that attacks always succeed when dealing with deductive arguments, in domains where arguments lack this coercive force, arguments provide reasons which may be more or less persuasive and their persuasiveness may vary according to their audience. Accordingly, it is necessary to distinguish between attacks and successful attacks (i.e. defeats) prescribing different strengths to arguments on the basis of the values they promote and/or their motivation in order to accommodate the different interests and preferences of an audience. With that purpose, the VAF [6] extended the AF [4] with (i) the concept of *Value* and (ii) the function *promotes* relating an *Argument* with a single *Value* (depicted in Fig. 1b). Therefore, a VAF can be defined as 4-uple $VAF = (A, R_{att}, V, promotes)$ where A and R_{att} means the same as in the AF, a non-empty set of values V and the function $promotes: A \rightarrow V$ to map elements from A to elements of V. Consequently, an *audience* for a VAF instance corresponds to a binary preference relation $P \subseteq V \times V$ which is transitive, irreflexive and asymmetric. If a pair $(v_1, v_2) \in P$ means that value v_1 is preferred to v_2 in the audience P. An attack between two arguments (i.e. $(a_1, a_2) \in R_{att}$) where a_1 promotes a value v_1 and a_2 promotes a value v_2 succeeds (i.e. a_1 defeats a_2) iif the adopted audience prefers v_1 to v_2 otherwise the attack fails.

As a result, previous notions (i.e. *acceptable*, *admissible*, *conflict-free* and *preferred extension*) were redefined accordingly (cf. [6] for details). Notice that for the same audience multiple preferred extensions are possible and different audiences may also lead to a unique preferred extension. In this way, different agents (each one represented by one audience) can have different perspectives (i.e. preferred extensions) over the same arguments.

The AF and the VAF assume that an argument a_1 supports an argument a_2 if a_1 attacks and therefore defeats an argument a_3 that attacks argument a_2. Thus, these frameworks only explicitly represent the negative interaction (i.e. attack), while the positive interaction (i.e. defense/support) of an argument a_1 to another argument a_2 is implicitly represented by the attack of a_1 to a_3. Since support and attack are related notions, this modeling approach adopts a parsimonious strategy, which is neither a complete nor a correct modeling of argumentation [8]. Conversely, the BAF [5] assumes the attack relation is independent of the support relation and both have a diametrically opposed nature and represent repellent forces. As a result, BAF [5] extended the AF [4] with the support relation (R_{sup}) in order to be explicitly represented (depicted in Fig. 1c). Thus, an BAF can be defined as a 3-uple $BAF = (A, R_{att}, R_{sup})$ where A and R_{att} means the same as in the AF and R_{sup} is a binary relation on A such that $R_{sup} \subseteq A \times A$. Given that, for any two arguments, say a_1 and a_2, such that $a_1, a_2 \in A$, one says that a_1 supports a_2 iif $(a_1, a_2) \in R_{sup}$. Consequently, the notions of *acceptable* and *conflict-free* arguments as well as the notion of a *preferred extension* were redefined accordingly (cf. [5] for details).

For all of these frameworks, an argument is anything that may attack/support or be attacked/supported by another argument. The absence of an argument structure and semantics enables the study of independent properties of any specific aspect that are relevant for any argumentation context that can be captured and formalized accordingly. On the other hand, this emphasizes the limited semantics for direct adoption in specific application contexts [7]. Indeed, a given application context requires a less abstract formalism to deal with (i) the construction of arguments and their structure, (ii) the conditions for an argument attack/support another, (iii) categories of arguments, etc.

3 Three-Layer Argumentation Framework

This section presents the proposed argumentation framework, which is denominated as Three-Layer Argumentation Framework (TLAF). First, we give an informal overview of the framework main concepts and their relations. Further, the framework is formally defined. Finally, an example is provided.

3.1 Informal Overview

Unlike the abstract argumentation frameworks described, the TLAF features three modeling layers as depicted in Fig. 2 (the line ending with a hollow triangle means specialization/generalization).

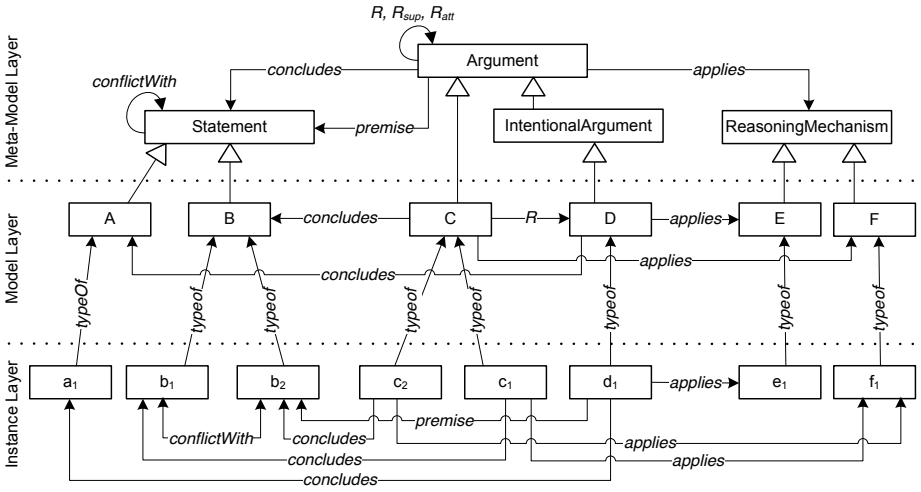


Fig. 2. The three modeling layers of the proposed argumentation framework

Despite existing differences, the TLAF Meta-Model Layer and the TLAF Instance Layer have the same purpose as those of AF [4], BAF [5] and VAF [6] layers with the same name. The TLAF Model Layer intends to capture the semantics of argumentation data (e.g. argument types/schemes) applied in a specific domain of application (e.g. e-commerce, legal reasoning and decision making) and the relations existing between them. In that sense, the model layer is important for the purpose of enabling knowledge sharing and reuse. A model is in this context a specification used for making model commitments. Practically, a model commitment is an agreement to use a vocabulary in a way that is consistent (but not complete) with respect to the theory specified by a model. Agents then commit to models and models are designed so that the knowledge can be shared among these agents. Accordingly, the content of this layer directly depends on (i) the domain of application to be captured and (ii) the perception one (e.g. a community of agents) has about that domain. Due to this, we adopt the vocabulary of (i) argument (or statement)-instance as an instance of an (ii) argument (or statement)-type defined at the Model Layer. Similarly, we adopt the vocabulary of (i) relation between types, and (ii) relationship between instances.

In TLAF, the meta-model layer defines an argument which is made of three parts: (i) a set of premise-statements, (ii) a conclusion-statement and (iii) an inference from premises to the conclusion enabled by a reasoning mechanism. This argument structure is very intuitive and corresponds to the minimal definition presented by Walton in [9]. For that, the meta-model layer defines the notion of *Argument*, *Statement* and *Reasoning Mechanism*, and a set of relations between these concepts. Following the notion of the BDI model [10, 11], an *IntentionalArgument* is the type of argument whose content corresponds to an intention. Domain data and its meaning are captured by the notion of *Statement*. This mandatorily includes the domain intentions, but also the desires and beliefs. The distinction between arguments and statements allows the application of the same domain data (i.e. statement) in and by different means to arguments. Also the same statement can be concluded by different

arguments, and serve as the premise of several arguments. The notion of *Reasoning Mechanism* captures the rules, methods, or processes applied by arguments.

At the instance layer, an argument-instance *applies* a reasoning mechanism to *conclude* a conclusion-statement from a set of *premise*-statements. The relation *conflictWith* is established between two statement-instances only. A statement-instance b_1 is said to be in conflict with another statement-instance b_2 when b_1 states something that implies or suggests that b_2 is not true or do not holds. The *conflictWith* relation is asymmetric (in Fig. 2 b_2 conflicts with b_1 too). In this case, for example, b_1 may represent the statement "Peter is an expert on PCs." and b_2 may represent the statement "Peter is not an expert on PCs". While the R_{att} and R_{sup} relations are established between argument-instances as in BAF [5], these relationships are automatically inferred in TLAF exploiting (i) the argument statements (i.e. conclusion and premises), (ii) the existing conflicts between statement-instances and (iii) based on the R relations defined at the model layer (cf. section 0 for details).

At the model layer, an argument-type (or argument scheme) is characterized by the statement-type it concludes, the applied reasoning mechanism and the R relations it has. The R relation is an abstraction of R_{att} and R_{sup} relations. The purpose of R is to define at the conceptual level that argument-instances of an argument-type may affect (either positively or negatively) instances of another argument-type. For example, according to the model layer of Fig. 2, $(C, D) \in R$ means instances of argument-type C may attack or may support instances of argument-type D depending on the instances content. On the other hand, if $(X, Y) \notin R$ it means that instances of argument-type X cannot (in any circumstance) attack/support instances of argument-type Y. Yet, the R relation is also used to determine the types of statements that are admissible as premises of an argument-instance. So, an argument-instance of type X can only have as premises statements of type S iif S is concluded by an argument-type Y and Y affects X (i.e. $(Y, X) \in R$). For example, considering again the model layer of Fig. 2, instances of argument-type D can only have as premises statements of type B because D is affected by argument-type C only.

It is worth noticing that all instances existing in the instance layer must have an existing type in the model layer and according to the type characterization.

3.2 Formal Definition

The TLAF is formally described as follows.

Definition 1 (TLAF). A TLAF structure is a singleton $TLAF = (E)$, where E is the set of entities of a TLAF.

A TLAF represents a self-contained unit of structured information. Elements in an TLAF are called argumentation entities.

Definition 2 (TLAF Model Layer). A model layer associated with a TLAF is a 6-tuple $ML(TLAF) = (A, IA, S, M, R, \sigma)$ where:

- $A \subseteq E$ is a set of argument-types;
- $IA \subseteq A$ is the sub-set of argument-types whose instances will play the role of intentional arguments, i.e. the arguments corresponding to the intentions ([10, 11]);
- $S \subseteq E$ is the a set of statement-types;

- $M \subseteq E$ is the set of reasoning mechanisms;
- $R \subseteq A \times A$ establishes a reflexive relation between two argument-types called arguments' affectation. If a pair $(a_1, a_2) \in R$ then argument-instances of type a_1 may affect (positively or negatively) argument-instances of type a_2;
- σ is a function that assigns to every argument-type (i) the concluded statement-type and (ii) the reasoning mechanism applied, such as $\sigma: A \to S \times M$ where:
  - function $concl: A \to S$;
  - function $reason: A \to M$.

Each TLAF has a model layer associated with it. Information captured within the model layer plays an important role by conducting and governing the instantiation process of the framework by an application, namely which concerns the construction and semantics of instances and existing relations between them. In that sense, the model layer can also be used to validate the TLAF Instance Layer.

Arguments may be used with two purposes: (i) to represent and communicate their intentions (i.e. intentional arguments) and (ii) to provide considerations (i.e. beliefs, desires) for and against those intentions. Intentional arguments may be also used as beliefs with respect to others intentions.

Notice that argument-types do not define their statement-types used as premises. Instead, these are derived from the R relation established between arguments.

Definition 3 (TLAF Instance Layer). An instance layer associated with a TLAF is a 6-tuple $IP(TLAF) = (I, instA, instS, instM, \Sigma, sconflict)$ where:

- $I \subseteq E$, is a set of instances;
- function $instA: A \to 2^I$ relates an argument-type with a set of instances. Consequently, the set of all argument instances AI is defined according to equation 1 (see below). Furthermore, we define the inverse function as $instA^-: AI \to A$;
- function $instS: S \to 2^I$ relates a statement-type with a set of instances. Consequently, the set of all statement instances SI is defined according to equation 1. Furthermore, we define the inverse function as $instS^-: SI \to S$;
- function $instM: M \to 2^I$ relates a reasoning mechanism with a set of instances. Consequently, the set of all reasoning mechanism instances MI is defined according to equation 1. Furthermore, we define the inverse function as $instM^-: MI \to M$;
- function $\Sigma: AI \to SI \times MI \times 2^{SI}$, defines for every argument-instance (i) the statement-instance concluded, (ii) the reasoning mechanism instance used to infer the conclusion and (iii) the set of statement-instances used as premises, where:
  - function $iconcl: AI \to SI$, defines the statement-instance that plays the role of conclusion on an argument-instance. Indeed, an argument-instance has only one statement-instance as conclusion while a statement-instance is concluded by at least one argument-instance;
  - function $ireason: AI \to MI$, defines the reasoning mechanism instance that is used by an argument-instance.
  - function $ipremise: AI \to 2^{SI}$, defines the statement-instances used as premises on an argument-instance. Moreover, statement-instances used as premises are also concluded by other arguments;
- function $sconflict: SI \to 2^{SI}$, defines the statement-instances that are in conflict with a statement-instance.

$$AI = \bigcup_{\forall x: x \in A} instA(x), \qquad SI = \bigcup_{\forall x: x \in S} instS(x), \qquad MI = \bigcup_{\forall x: x \in M} instM(x) \tag{1}$$

As the reader might have noticed, the instance layer definition is concerned with the generation of argument-instances, statement-instances and their inter-relationships (Σ and $sconflict$). Despite the fact that this is a domain dependent process, it profits from the subjacent TLAF model, namely due to the rules complementing the $iconcl$, $ipremise$ (see next definition) and $sconflict$ (see section 0), that have the ability to conduct and simplify the process.

Definition 4 (TLAF Interpretation). An interpretation of a TLAF is a structure $\mathfrak{I} = (\Delta^{\mathfrak{I}}, A^{\mathfrak{I}}, S^{\mathfrak{I}}, M^{\mathfrak{I}}, I^{\mathfrak{I}})$ where:

- $\Delta^{\mathfrak{I}}$ is the domain set;
- $A^{\mathfrak{I}}: A \to 2^{\Delta^{\mathfrak{I}}}$ is an argument interpretation function that maps each argument-type to a subset of the domain set;
- $S^{\mathfrak{I}}: S \to 2^{\Delta^{\mathfrak{I}}}$ is a statement interpretation function that maps each statement-type to a subset of the domain set;
- $M^{\mathfrak{I}}: M \to 2^{\Delta^{\mathfrak{I}}}$ is a reasoning mechanism interpretation function that maps each reasoning mechanism to a subset of the domain set;
- $I^{\mathfrak{I}}: I \to \Delta^{\mathfrak{I}}$ is an instance interpretation function that maps each instance to a single element in the domain set;

An interpretation is a model of TLAF if it satisfies the following properties:

- $\forall a, i: a \in A \wedge i \in instA(a) \Rightarrow I^{\mathfrak{I}}(i) \in A^{\mathfrak{I}}(a)$;
- $\forall s, i: s \in S \wedge i \in instS(s) \Rightarrow I^{\mathfrak{I}}(i) \in S^{\mathfrak{I}}(s)$;
- $\forall m, i: m \in M \wedge i \in instM(m) \Rightarrow I^{\mathfrak{I}}(i) \in M^{\mathfrak{I}}(m)$;
- $\forall a: a \in IA \Rightarrow A^{\mathfrak{I}}(a)\ are\ intentions$
- $\forall a, i: a \in A \wedge i \in instA(a) \Rightarrow I^{\mathfrak{I}}\big(iconcl(i)\big) \in S^{\mathfrak{I}}\big(concl(a)\big) \wedge I^{\mathfrak{I}}\big(ireason(i)\big) \in M^{\mathfrak{I}}\big(reason(a)\big)$;
- $\forall a, i, p: a \in A \wedge i \in instA(a) \wedge p \in ipremise(i) \Rightarrow \exists x, y: I^{\mathfrak{I}}(y) \in A^{\mathfrak{I}}(x) \wedge p = iconcl(y) \wedge (x, a) \in R$
- $\forall a, s: a \in A \wedge s \in S \Rightarrow A^{\mathfrak{I}}(a) \cap S^{\mathfrak{I}}(s) = \emptyset$;
- $\forall a, m: a \in A \wedge m \in M \Rightarrow A^{\mathfrak{I}}(a) \cap M^{\mathfrak{I}}(m) = \emptyset$;
- $\forall s, m: s \in S \wedge m \in M \Rightarrow S^{\mathfrak{I}}(s) \cap M^{\mathfrak{I}}(m) = \emptyset$.

3.3 Example

This section provides an example whose purpose is to show the application of TLAF. For that, we decide on a common and simple scenario such as buying digital cameras. Fig. 3 graphically depicts a partial TLAF model layer for such scenario[1].

The intention of buying a camera is captured by the argument-type *BuyCamera* which is affected by considerations about (i) the *Requirement* to buy a camera, (ii) the

[1] Instead of a formal definition, we present a partial graphical view of the model layer because we consider it to be more informative to the reader.

general trend of received *Reviews*, (iii) the general perspective about the cameras' *Features* and (iv) the *PriceRelation* (i.e. expensive vs. cheap). The *PriceRelation* grounds on considerations about the *CurrentPrice* and the *PastPrice*.

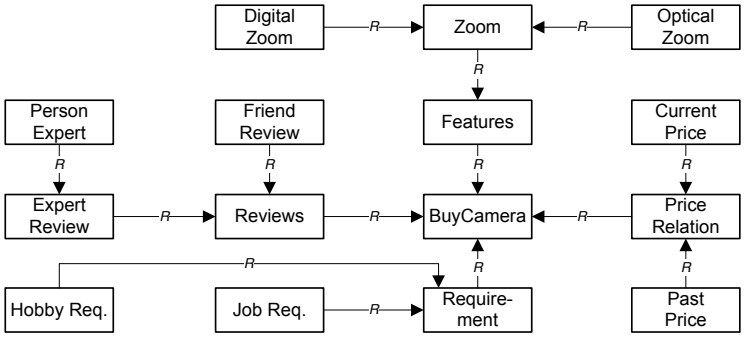


Fig. 3. A partial view over a TLAF model layer for buying cameras

The *Requirement* is affected by two types of considerations: (i) *HobbyReq* (i.e. a hobby requirement) or (ii) a *JobReq* (i.e. job requirements). *Reviews* are affected by each individual opinion of friends (*FriendsReview*) and experts (*ExpertReview*). The latter requires that the reviewer is considered an expert (*PersonExpert*). The *Features* are affected by considerations about the *Zoom* which is made based on the *DigitalZoom* and *OpticalZoom*. Additionally, for the sake of brevity, consider that each of these arguments concludes a statement-type with a similar name (e.g. argument *OpticalZoom* concludes *OpticalZoomStmt*) and applies a heuristic or presumptive reasoning mechanism. Notice that the provided conceptualization do not intends to be neither complete nor the most accurate approach for the scenario in hands.

Now, let us address the instantiation of the TLAF. Consider the following dialogue takes place between husband (H) and wife (W), where the relevant statements are marked as st_i (with $i > 0$).

H. I am looking forward to buy camera X (st_1).
W. Why? We don't need it (st_2).
H. That is not true (st_3). I need a camera to perform the task that Sam assigned to me (st_4). Besides that, the camera received several good reviews on a website (st_5).
W. Susan and Mary bought that camera and they told me that they regret their option (st_6 and st_7).
H. Oh, come on Honey. Peter Noble is an expert on the matter (st_8) and he says great things about the camera (st_9).
W. How much it costs? Is it expensive?
H. No! Currently, there is a great opportunity in the city mall (st_{10}). It only costs 100€ (st_{11}). Last week, the price was 150€ (st_{12}).
W. That camera is a discontinued product.
H. I don't care about that.

W. I am reading in this magazine that it lacks some minimal features (st_{13}) such as zoom (st_{14}).
H. Nonsense! Camera X has a digital zoom of "80x" (st_{15});
W. Yeah! But, the optical zoom is only of "4x" (st_{16}).

Even though this is a short and somewhat contrived dialogue, it already may be difficult to keep track of all argument-instances and their relationships. Consider the arguments, the statements and the relationships between arguments and statements presented in Table 1 as the result of an instantiation process.

Table 1. Instances of arguments and statements constructed and their relationships

Argument		**Premise**	**Conclusion-Statement**	
ID	**Type**	**Statements**	**Statement**	**conflictWith**
a_1	BuyCamera	st_3, st_5, st_{10}	st_1	
a_2	Requirement		st_2	st_1, st_3
a_3	Requirement	st_4	st_3	st_2
a_4	JobReq		st_4	
a_5	Reviews	st_9	st_5	
a_6	FriendReview		st_6	st_5
a_7	FriendReview		st_7	st_5
a_8	PersonExpert		st_8	
a_9	ExpertReview	st_8	st_9	
a_{10}	PriceRelation	st_{11}, st_{12}	st_{10}	
a_{11}	CurrentPrice		st_{11}	
a_{12}	PastPrice		st_{12}	
a_{13}	Features	st_{14}	st_{13}	st_1
a_{14}	Zoom	st_{16}	st_{14}	
a_{15}	DigitalZoom		st_{15}	st_{14}
a_{16}	OpticalZoom		st_{16}	

Each statement identified in the dialogue gives raise to one argument-instance concluding that statement. Since the semantics of the argument-type *FriendReview* corresponds to a single opinion and Susan and Mary have the same opinion then two argument-instances (a_6 and a_7) were generated. During the argumentation dialogue the premises of arguments are not always explicit (e.g. st_{10} as premise of a_1). In those circumstances the premises of arguments are inferred through the information captured in the model layer that is governing the instantiation (e.g. arguments of type *PriceRelation* directly affect arguments of type *BuyCamera*). On the other hand, notice that the information stating camera X is a discontinued product did not give raise any statement or argument because it was not envisioned in the scenario conceptualization. This example is further developed in section 4.3.

4 Deriving Arguments Relationships

According to the formal definitions introduced above, the R_{att} and R_{sup} relationships between argument-instances of an $IP(TLAF)$ are not explicitly defined. Instead, these

relationships are derived based on two distinct kinds of information:

- extensional information (existing at the instance layer):
  - the premises and conclusions of the argument-instances;
  - the conflicts between statement-instances, and;
- conceptual information (existing at the model layer), namely the R relations defined between argument-types.

4.1 Deriving Support Relationships

A support relationship between two argument-instances (say x and y) is established (i.e. $(x, y) \in R_{sup}$) when the argument-type of x (say a) affects the argument-type of y (say b), i.e. $(a, b) \in R$, and either (i) the conclusion of x is a premise of y or (ii) both argument-instances have the same conclusion. The following rules (graphically depicted in Fig. 4) capture the conditions required to establish support relationships between argument-instances:

R1. $\forall a, b, x, y: a, b \in A \wedge (a, b) \in R \wedge x \in instA(a) \wedge y \in instA(b) \wedge x \neq y \wedge iconcl(x) \in ipremise(y) \Rightarrow (x, y) \in R_{sup}$ (Fig. 4a);

R2. $\forall a, b, x, y: a, b \in A \wedge (a, b) \in R \wedge x \in instA(a) \wedge y \in instA(b) \wedge x \neq y \wedge iconcl(x) = iconcl(y) \Rightarrow (x, y) \in R_{sup}$ (Fig. 4b).

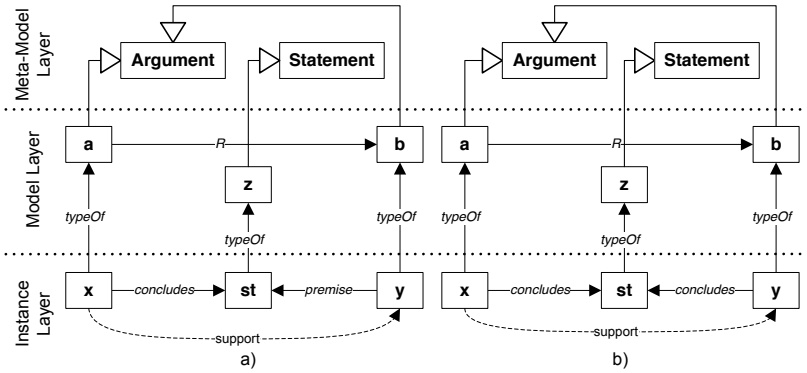


Fig. 4. Conditions to derive a support relationship between two argument-instances

Notice that two argument-instances might achieve the same conclusion starting from a different set of premises and/or reasoning mechanisms. In those circumstances, a support relation between argument-instances exists if there is a R relation between both (depicted in Fig. 4b). For a mutual support, two R relationships are required: one from a to b (i.e. $(a, b) \in R$) and another one from b to a (i.e. $(b, a) \in R$).

4.2 Deriving Attack Relationships

An attack relationship between two argument-instances (say x and y) is established (i.e. $(x, y) \in R_{att}$) when the argument-type of x (say a) affects the argument-type of

y (say b), i.e. $(a, b) \in R$, and either (i) the conclusion of x is in conflict with any premise of y or (ii) the conclusion of x is in conflict with the conclusion of y. The following rules (graphically depicted in Fig. 5) capture the conditions required to establish attack relationships between argument-instances:

R3. $\forall a, b, x, y, s: a, b \in A \wedge (a, b) \in R \wedge x \in instA(a) \wedge y \in instA(b) \wedge x \neq y \wedge s \in ipremise(y) \wedge s \in sconflict(iconcl(x)) \Rightarrow (x, y) \in R_{att}$ (Fig. 5a);

R4. $\forall a, b, x, y: a, b \in A \wedge (a, b) \in R \wedge x \in instA(a) \wedge y \in instA(b) \wedge x \neq y \wedge iconcl(y) \in sconflict\big(iconcl(x)\big) \Rightarrow (x, y) \in R_{att}$ (Fig. 5b).

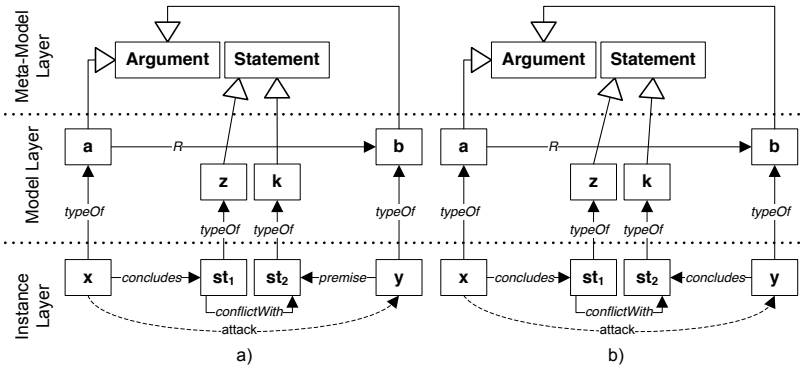


Fig. 5. Conditions to derive an attack relationship between two argument-instances

According to the rule/scenario depicted in Fig. 5b, one cannot say that argument y also attacks argument x because the conflict relation between statements is asymmetric. However, that would happen iif statement st_2 is also in conflict with statement st_1 (i.e. $st_1 \in sconflict(st_2)$) and a R relationship between b and a (i.e. $(b, a) \in R$) exists too.

The application process used to identify and establish conflicts between statement-instances may exploit the knowledge embedded in these rules to reduce and drive the search/combination space between statements. Indeed, it is worth establishing a conflict relationship between two statement-instances (say st_1 and st_2) iif their statement-types (say z and k respectively) satisfy at least one of the following conditions:

- There is an argument-type (say a) concluding z that affects any other argument-type (say b), i.e. $(a, b) \in R$, where statement-instances of type k can be used as premises of argument-instances of type b;
- There is an argument-type (say a) concluding z that affects any other argument-type (say b), i.e. $(a, b) \in R$, where k is concluded by b.

Notice that, these conditions can be verified using the information captured at the model layer only. On the other hand, if a conflict relationship is established between two statement-instances and none of these conditions apply then it has no impact on derived attack relationships between arguments.

4.3 Example

Considering the example introduced in section 3.3, previous rules would establish the support and attack relationships. For example, applying rule R1 to the argument types *CurrentPrice* and *PriceRelation*, it is instantiated as follows:

$$(CurrentPrice, PriceRelation) \in R \wedge a_{11} \in instA(CurrentPrice) \wedge a_{10} \in instA(PriceRelation) \wedge a_{10} \neq a_{11} \wedge iconcl(a_{11}) \in ipremise(a_{10}) \Rightarrow (a_{11}, a_{10}) \in R_{sup}$$

Applying rule R3 to the argument-types *Requirement* and *BuyCamera*, it is instantiated as follows:

$$(Requirement, BuyCamenra) \in R \wedge a_2 \in instA(Requirement) \wedge a_1 \in instA(BuyCamera) \wedge a_1 \neq a_2 \wedge st_3 \in ipremise(a_1) \wedge \mathrm{st}_3 \in sconflict(iconcl(a_2)) \Rightarrow (a_2, a_1) \in R_{att}$$

All derived R_{sup} and R_{att} relationships are graphically depicted in Fig. 6.

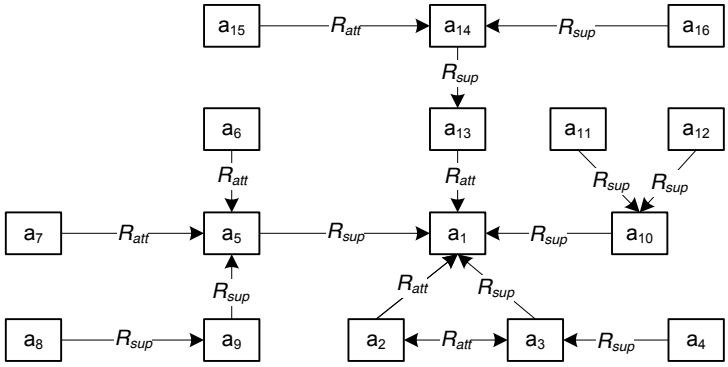


Fig. 6. Derived support and attack relationships between argument-instances of the example

5 Conclusions and Future Work

This paper describes the Three-Layer Argumentation Framework (TLAF) that reduces the existing gap between the most referenced abstract argumentation frameworks and its adoption by applications. The main novelty of the proposed argumentation framework relies on its conceptualization layer (i.e. model layer), namely the *R* relation. This layer captures the structure and semantics of the argumentation data employed in a specific context constraining and conducting the modeling process of the argumentation specific scenario. Even though, for the same scenario very different modeling approaches are possible. The TLAF also adopts an intuitive argument structure encompassing a set of premise-statements, a reasoning mechanism and a conclusion-statement, which matches the Walton's perspective presented in [9]. Similarly, several works such as the AIF-based ontology proposed in [12] are also adopting the Walton's perspective. Unlike the AIF-based ontology, TLAF models (direct) conflicts between statements only. Moreover, TLAF explicitly distinguishes between argument-type (argument scheme) and the reasoning

mechanisms, while the reasoning mechanisms in the AIF-based ontology are implicit in the name of the argument-scheme. TLAF is then more flexible than the AIF-based ontology.

Despite being generic, TLAF is mainly targeted to be adopted by autonomous agents. In relation to that, the TLAF adopts and follows some terminology from the BDI model, namely by distinguishing between intentional arguments and non-intentional arguments. Based on the conceptual relations captured by the framework and the defined argument structure, a clear and minimal set of conditions was established for an argument-instance to attack/support another one. Given that, the support and attack relations between argument-instances are automatically derived according to the subjacent TLAF model. Despite the fact that the argument-instances generation process, and the Σ and $sconflict$ functions are fully domain dependent, their definition profits from the established TLAF model. An OWL ontology capturing the TLAF concepts is available in [13].

While not directly addressed in this paper, the TLAF has the following advantages: (i) when generating statements it constrains the scope in which it is valuable to establish a conflict relationship between statements (i.e. $sconflict$), and therefore simplifies the automation of the process that discovers or instantiates the $sconflict$ relation, by reducing and driving the search/combination space between statements; (ii) when generating arguments upon existing statements, it constraints the type of conclusion and premises, and the reasoning mechanism associated with an argument-instance, therefore simplifying the automation of the process that instantiates arguments, that establishes the premises and conclusion relationships with statements and establishes the $R_{att}$and R_{sup} relationships between arguments.

Besides the new features provided by TLAF, it is generic enough to be adopted by different domain applications. Moreover, a TLAF instance can be easily represented in a more abstract formalism such as BAF, where the AI set corresponds to the set of arguments of BAF and the derived argument-instances relationships, i.e. R_{sup} and R_{att}, correspond to the BAF binary relations with the same name respectively. Therefore, one can use this feature to benefit from the BAF capabilities already identified in the literature, namely those concerned with arguments acceptability and the study of independent properties such as the notion of internal and external coherence with respect to the preferred extensions. Notice that, while a valid BAF instance is always achieved from a TLAF instance, the inverse operation is not possible because a TLAF instance requires the existence of a model layer that does not exist in BAF.

TLAF does not impose any particular argument evaluation process. For the moment it is envisaged the adoption of an existing argument evaluation process capable to deal with bipolarity such as the ones proposed in [8, 14-16]. However, because none of these processes is able to take advantage of the TLAF Model Layer we are working to propose one as well. The multi-classification of arguments allowed in AIF-based ontology raises acceptability problems not completely understood.

The authors consider that no experiences would be relevant for the evaluation of the proposed framework, as its application depends on the modeling approaches of the domain, and less of the framework. This suggests the need for further development of methods and methodologies for argument modeling.

In order to simplify the modeling process and profit from experience, for example, in the software engineering and ontology development fields, the authors envisage the need to provide modularity and extensibility modeling features to TLAF. These new features potentially promote TLAF in the scope of heterogeneous, ill-specified, emergent multi-agent systems as it provides the mechanisms to model private argumentation models in respect (specializing) to other argumentation models, thus inheriting a common model.

Acknowledgments. This work is partially supported by the Portuguese MCTES-FCT project COALESCE (PTDC/EIA/74417/2006). The authors would like to acknowledge Jorge Santos, Maria João Viamonte, Jorge Coelho and Besik Dundua for their useful counsels and Owen Gilson for his revisions of the document.

References

1. Wooldridge, M.: Reasoning about rational agents. The MIT press, Cambridge (MA US) (2000).
2. Moran, R.: Authority and Estrangement: An Essay on Self-Knowledge. Princeton University Press (2001).
3. Walton, D.N., Krabbe, E.C.W.: Commitment in dialogue. SUNY Press (1995).
4. 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).
5. Cayrol, C., Lagasquie-Schiex, M.C.: On the Acceptability of Arguments in Bipolar Argumentation Frameworks. Symbolic and Quantitative Approaches to Reasoning with Uncertainty. pp. 378-389 (2005).
6. Bench-Capon, T.J.M.: Persuasion in Practical Argument Using Value-based Argumentation Frameworks. J Logic Computation. 13, 429-448 (2003).
7. Baroni, P., Giacomin, M.: Semantics of Abstract Argument Systems. Argumentation in Artificial Intelligence. pp. 25-44 (2009).
8. Cayrol, C., Lagasquie-Schiex, M.C.: Gradual Valuation for Bipolar Argumentation Frameworks. Symbolic and Quantitative Approaches to Reasoning with Uncertainty. pp. 366-377 (2005).
9. Walton, D.: Argumentation Theory: A Very Short Introduction. Argumentation in Artificial Intelligence. Springer Publishing Company, Incorporated (2009).
10. Bratman, M.: Intention, Plans and Practical Reason. Cambridge, MA: Harvard University Press (1987).
11. Wooldridge, M.: An Introduction to MultiAgent Systems. Wiley (2009).
12. Rahwan, I., Banihashemi, B., Reed, C., Walton, D., Abdallah, S.: Representing and classifying arguments on the semantic web. The Knowledge Engineering Review (to appear). (2011).
13. Maio, P., Silva, N.: TLAF Meta-Model Layer as an Ontology, http://www.dei.isep.ipp.pt/~pmaio/TLAF/Ontology/TLAF_Ontology.owl.
14. Amgoud, L., Cayrol, C., Lagasquie-Schiex, M.C., Livet, P.: On bipolarity in argumentation frameworks. Int. J. Intell. Syst. 23, 1062-1093 (2008).
15. Karacapilidis, N., Papadias, D.: Computer Supported Argumentation And Collaborative Decision Making: The Hermes System. Information Systems. 26, 259--277 (2001).
16. Verheij, B.: On the existence and multiplicity of extensions in dialectical argumentation. cs/0207067. (2002).

Arguments over Co-operative Plans

Rolando Medellin-Gasque[1], Katie Atkinson[1], Peter McBurney[2], and Trevor Bench-Capon[1]

[1] University of Liverpool, Department of Computer Science, Liverpool UK
{medellin,katie,tbc}@liverpool.ac.uk
[2] King's College London, Department of Informatics, London UK
peter.mcburney@kcl.ac.uk

Abstract. Autonomous planning agents that share a common goal should be able to propose, justify and share information about plans. To reach an agreement on the best plan, strategies for persuasion and negotiation can be used by agents in order to share their beliefs about the world and resolve conflicts between the agents. We present an argumentation scheme and associated critical questions to create and justify plan proposals where plans are combinations of actions requiring several agents for their execution. An analysis of different ways in which actions can combine is presented and then associated with the argumentation scheme and the critical questions. We believe these elements are necessary to enable agents to engage in rational debate over co-operative plan proposals.

Key words: plan proposal, argumentation schemes, critical questions, co-operation.

1 Introduction

Planning in Artificial Intelligence is concerned with the automatic synthesis of action strategies from a description of actions, sensors and goals [11]. The planning literature has been focusing in recent years on overcoming strong assumptions about plan generation. The complexity of distributed systems restricts the application of single-agent planning strategies to distributed problems usually because a local agent view is not sufficient. A common assumption in AI planning is that the planner has accurate and complete knowledge of the world and the capabilities of other agents. We want to overcome this assumption and provide strategies where agents with different views of the world are able to propose and justify complex plans. Our goal is to provide autonomous agents with tools to justify plans in terms of acceptable arguments and enable them to critique and defend plans to refine them and choose between them. The dialogue is intended to support planning tasks such as plan modification, choosing the best plan and even establishing coordination strategies for the execution of a plan.

In this paper we present an argumentation scheme to propose and justify plans based on the argumentation scheme for action proposals of Atkinson *et al.* in [3]. We extend the concept of action used in [3] with action-elements taken from the PDDL 2.1 Planning Specification [3] presented in [10] such as time constraints and invariant conditions.

[3] Planning Domain Description Language (PDDL) is an attempt to standardize planning domain and problem description languages developed for the International Planning Competitions.

Thus, this work extends the action proposal model of [3] to more complex types of action-proposals involving several durative actions performed by several agents. An analysis over different ways to combine actions to form plans is also presented in order to create more specific critical questions. The analysis is based on interval algebra proposed by Allen in [1]. Allen's interval relations define the basic relations between time intervals. We relate time intervals with the action duration in order to define ways in which actions may be combined in a co-operative plan. Furthermore, we present critical questions grouped in 6 categories that address specific elements of the plan-proposal. As a basis to formalize our argumentation scheme we will use a formal model developed in [17], an Action-based Alternating Transition System (AATS). This transition system defines actions that may be performed by agents through the states from which these could be performed and the states that will result, with a particular focus on the simultaneous action of a group of agents. This makes AATSs especially suitable for situations where co-operation is important.

The paper is structured as follows: Section 2 presents the action representation and proposal including action combinations. Sub-section 2.3 introduces the AATS notation to formalize the action proposal in sub-section 2.4. In section 3 we present the plan proposal as an argumentation scheme of AATS models together with critical questions associated to the extended action and the ways in which actions can combine. In section 4 we develop an example using the elements presented in this paper. Section 5 comments on related work and finally, in section 6, we conclude the paper and discuss future research work.

2 Action Representation and proposal

2.1 Action representation

Actions usually are represented as operations an agent is able to perform from a state where some preconditions hold. We want to extend this action definition and incorporate elements useful when representing and reasoning about temporal plans. We use actions as presented in the PDDL 2.1 specification [10] which have elements to express temporal domain descriptions over plans. In the PDDL 2.1 specification, instead of having an action with preconditions and effects, actions are represented as durative actions with elements to express more precisely temporal conditions and effects. The durative action representation is as follows: initially, the action can start at a point in time when a set of preconditions hold, once the action start, " start effects" become true. Action has a duration and "invariant conditions" (distinct from preconditions) and are accessible though the duration of the action. Actions are not black-boxes and access to start effects is available during the performance of the action. The end of the action is given by "termination conditions" where upon, end effects become true [4]. The planning community is still developing ways to create planners that handle temporally extended actions. Our intention in presenting this durative action representation is to consider all the elements

[4] This model still represents a simplified model of time; durative actions could be extended to allow effects to be asserted at arbitrary points during the interval of execution, or to be a function of duration ("until" actions).

needed by agents to engage in argumentative dialogues over co-operative plans. In the next sub-section we will define different ways in which actions may combine to form a plan.

2.2 Action combinations

We now define the way in which actions can be combined to form plans. By action combinations we mean the different ways in which atomic actions could be combined in a plan in terms of concurrency, repetition and temporal aspects. Even if there is just one action there could be variants such as its periodicity or whether the action execution is optional. Two or more actions could be defined in a plan as a sequence (as in classical AI planning) or as a set of actions with no particular order (partial-order planning) that could, but need not, overlap. We want to cover both cases and others focusing on aspects such as the order of the actions and their periodicity.

The analysis is based on the interval algebra proposed by Allen in [1]. Interval algebra is based on the 13 possible primitive relationships (6 of which are inverses) between two time intervals (Figure 1). We apply a similar model to combinations of actions. Most of the interest in Allen's representation for time intervals comes from a mechanism by which the time relationships between the pairs of intervals can be propagated through the collection of all intervals. The notion of disjunction of interval relationships can be used to declare multiple paths and interactions. This idea gives us reason to think this analysis could be extended for larger plans. We add to the Allen list cases focusing on specific properties such as the periodicity, optionality and interleaving of actions. The 14 cases are presented in the following list, for arbitrary actions $alpha$ and $beta$:

AC1.- Action α occurs exactly *k* times, where *k* is a non-negative integer ($\alpha(k)$).
AC2.- Optionally, action α occurs exactly *k* times ($\alpha(k, o)$).
AC3.- Action α occurs from 1 to *k* times, where $k > 1$ ($\alpha((1 - k)$).
AC4.- Optionally, action α occurs from 1 to *k* times ($\alpha(1 - k, o)$).
AC5.- Action α precedes β ($precedes(\alpha, \beta)$) (Figure 1a).
AC6.- Action α meets β ($meets(\alpha, \beta)$) (Figure 1b).
AC7.- Action α overlaps β ($overlaps(\alpha, \beta)$) (Figure 1c).
AC8.- Action α starts β. ($starts(\alpha, \beta)$) (Figure 1d).
AC9.- Action α is entirely in action β ($entirely(\alpha, \beta)$ (Figure 1e).
AC10.- Action α finishes β. ($finishes(\alpha, \beta)$ (Figure 1f).
AC11.- Action α equals β. ($equals(\alpha, \beta)$ (Figure 1g).
AC12.- Action α or action β but not both ($\alpha|\beta$))
AC13.- Both actions interleaving concurrently (overlapping) over periods of time until completion of both ($iC(\alpha, \beta)$.
AC14.- Both actions executed not concurrently over periods of time until completion of both. $i(\alpha, \beta)$.

The purpose of this analysis is to cover all of the ways in which actions may be combined in a plan with questions that match the specific action combinations. This analysis covers cases where plans are formed of one or two atomic actions. Perhaps,

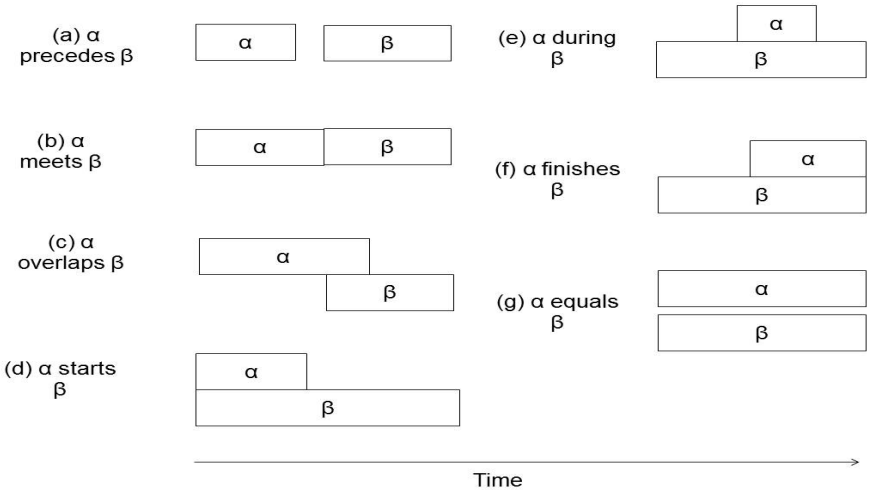


Fig. 1. Allen's possible primitive time relationships between two intervals labelled α and β. Time is represented by the horizontal axis. (a) to (f) have inverses.

plans comprising one or two actions seem too simple for the purposes of creating real-world plans, but nevertheless, we want to identify basic cases in which actions may be combined before extending this to larger plans.

2.3 Action-based Alternating Transition Systems

We use Action-based Alternating Transition Systems (AATS) as introduced in [17] as a basis for our formalism to represent action and plan proposals. AATS models define joint-actions that may be performed by agents in a state and the effects of these actions. In particular, an AATS model defines semantic structures useful to represent joint-actions for multiple agents, their preconditions and the states that will result from the transition. An *AATS* is an (n+7)-tuple of the form:

$$S = \langle Q, q_0, A_g, Ac_1, ..., Ac_n, \rho, \tau, \Phi, \pi \rangle$$

where:

- Q is a finite non-empty set of states;
- $q_0 \in Q$ is the initial state;
- $A_g = \{1, ..., n\}$ is a finite non-empty set of agents;
- Ac_i is a finite, non-empty set of actions, for each $i \in Ag$, where $Ac_i \cap Ac_j = \emptyset$ for all $i \neq j \in Ag$;
  Now we can say that a joint action j_{Ag} for the set of agents Ag is a tuple $(\alpha_i, .., \alpha_n)$ where for each $\alpha_j (j \leq n)$ there is some $i \in Ag$ such that $\alpha_j \in Ac_i$. We denote the set of all joint-actions J_{AG}. Given an element j of J_{AG} and an agent $i \in Ag$, $i's$ action in j is denoted by j_i .

- $\rho : Ac_{Ag} \rightarrow 2^Q$ is an action precondition function, which for each action $\alpha \in Ac_{Ag}$ defines the set of states $\rho(\alpha)$ from which α may be executed;
- $\tau : Q \times J_{Ag} \rightarrow Q$ is a partial system transition function, which defines the state $\tau(q, j)$ that would result by the performance of j from state q, note that, as this function is partial, not all joint actions are possible in all states (cf. the pre-condition function above);
- Φ is a finite, non-empty set of atomic propositions; and
- $\pi : Q \rightarrow 2^\Phi$ is an interpretation function, which gives the set of primitive propositions satisfied in each state: if $p \in \pi(q)$, then this means that the propositional variable p is satisfied (equivalently, true) in state q.

In [2] Atkinson and Bench-Capon extended this transition system to enable representation of a theory of practical reasoning related to arguments about action through which values [5] were added to the system. The extensions are:

- Av_i, is a finite, non-empty set of values $Av_i \subseteq V$, for each $i \in Ag$.
- δ: $Q \times Q \times Av_{Ag} \rightarrow \{+, -, =\}$ is a valuation function which defines the status (promoted(+), demoted(-) or neutral (=)) of a value $v_u \in Av_{Ag}$ ascribed by the agent to the transition between two states: $\delta(q_x, q_y, v_u)$ labels the transition between q_x and q_y with one of $\{+, -, =\}$ with respect to the value $v_u \in Av_{Ag}$.

2.4 Proposals for Action

Argumentation schemes are stereotypical patterns of defeasible reasoning used in everyday conversational argumentation. In an argumentation scheme, arguments are presented as general inference rules where under a given set of premises a conclusion can be presumptively drawn [19]. Artificial Intelligence has become increasingly interested in argumentation schemes due to their potential for making significant improvements in the reasoning capabilities of artificial agents [7] and for automation of agent interactions. In [20], Walton explains: "*...arguments need to be examined within the context of an ongoing investigation in dialogue in which questions are being asked and answered*". Critical questions are a way to examine the acceptability of arguments. Depending on the nature of the critical question, they can be used to critique several aspects of the argument. Usually, critical questions provide pointers which would make the argumentation scheme inapplicable or could lead to a valid way to attack the argument, either defeating the argument on one of its premises or on its presumptive conclusion.

The action proposal presented in [3] is as follows: In the current circumstances R, we should perform action A to achieve new circumstances S which will realize some goal G which will promote some value v. Furthermore, in [2] the authors re-stated the argumentation scheme in terms of the extended AATS. Figure 2 presents an action as in the PDDL 2.1 specification (presented in section 2.1). So, we can extend the action proposal from [3] with elements from the PDDL 2.1 specification. The extended action proposal and AATS representation are presented in Table 1.

For the purpose of this paper, time is discrete and actions take a single time step, thus we will not represent durative actions elements from section 2.1 in the plan proposal in the next section. Nevertheless, in the critical questions' section, time elements

[5] Our use of the term values follows [4] where values are qualitative social interests of agents.

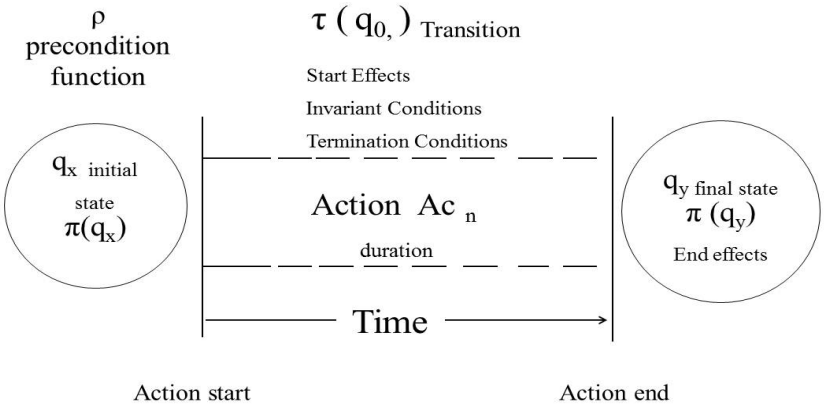


Fig. 2. Action Proposal Representation.

Table 1. Argumentation scheme for actions

Action Proposal	**as an AATS model** $AS2$
In the current circumstances R	In the initial state $q_0 = q_x \in Q$
we should perform action A at time t with duration d	agent $i \in Ag$ should participate in joint action $j_n \in J_{Ag}$
to achieve start effects from point t	where $j_{n^i} = \alpha_i$
given invariant conditions	such that $\tau(q_x, j_n)$ is q_y
action finishing by termination conditions	such that $p_a \in \pi(q_y)$ and $p_a \notin \pi(q_x)$
to achieve new circumstances S	or $p_a \notin \pi(q_y)$ and $p_a \in \pi(q_x)$
which will realize some goal G	such that for some $v_u \in Av_i$, $\delta(q_x, q_y, v_u)$ is +.
which will promote some value v	

are considered. Future work will be focused on representing durative actions within the action and plan proposal and the representation of action elements such as the propositions satisfied during the transition, which do not arise in [3].

3 Plan Proposal and Critical Questions

We now present our argumentation scheme in terms of the action elements presented above. Our plan proposal ASP is as follows: Given a social context X [6] in the current circumstances q_0 holding preconditions $\pi(q_0)$, plan PL should be performed to achieve

[6] The "social context" was an extension to the argumentation scheme presented in [5] where agents use a social structure to issue valid commands between them.

new circumstances q_x , that will hold postconditions $\pi(q_x)$ which will realize the plan-goal G which will promote value(s) V_G.

The valid instantiation of the scheme pre-supposes the existence of a regulatory environment or a social context X in which the proponent has some rights to engage in a dialogue with the co-operating agent. Current circumstances are represented by an initial state q_0. The agent acting as the proponent propose plan PL as a finite set of linked action-combinations. The plan leads to a state in which propositions $\pi(q_x)$ and the plan-goal G is achieved (where G is an assignment of truth values to a set of propositions $p \subseteq \Phi$) and a non-empty set of values associated with the plan is promoted. Table 2 presents the plan proposal as an AATS model.

Table 2. Plan Proposal ASP

Plan Proposal	**as an AATS model**
Given a social context X, in the current circumstances q_x holding preconditions $\pi(q_x)$ plan PL should be performed to achieve new circumstances q_y that will hold postconditions $\pi(q_y)$ which will realize the plan-goal G which will promote value(s) V_G.	Given social context Δ , In the initial state q_0 = $q_x \in Q$, where $\pi(q_0)$, agents $i, j \in Ag$ should execute plan PL, where PL is a finite set of joint-actions j_n such that $PL = \{j_0, .., j_n\}$ and $\{j_0, .., j_n\} \in J_{Ag}$ and $j_n = \{\alpha_i, .., \alpha_j\}$ with transition given by $\tau(q_x, PL)$ is q_y , where $\tau(q_0, \{j_1, .., j_n\}) = \tau(\tau(q_0, j_1), (j_2, ..j_n))$ and $\tau(q_x, \{\}) = q_x$ such that $p_a \in \pi(q_x)$ and $p_a \notin \pi(q_y)$ where $G = p$ and ($V_G \subseteq V$ such that $v_1 \in V_G$ iff $\delta(q_x, q_y, v_1)$ is $+$) and $V_G \neq \emptyset$

3.1 Critical Questions for plan proposals

A benefit of having critical questions associated with an argument scheme is that the questions enable dialogue participants to identify points of challenge in a debate or locate premises in an instantiation of the argument scheme that can be recognized as questionable. Most of the critical questions are created from argumentation scheme elements and represent a valid way to challenge proposals that could identify sources of disagreement about a particular element of the argumentation scheme. A question can be seen as a weak form of attack on a particular element of the argument scheme given different beliefs about the world of the agent posing the question. Critical questions then could be used to create Dialogue Games for agents where the participants put forward arguments instantiating the argumentation scheme and opponents to the argument challenge it through objections based on critical questions. Argumentation-based dialogues are used to formalize dialogues between autonomous agents based on theories of argument exchange. In [18] a classification is given based on the role the question plays in the context of the argumentation scheme. A question could be used to: criticize a scheme premise, point to exceptional situations in which the scheme should not

be used, set conditions for the proper use of the scheme, or point to other arguments that might be used to attack the scheme. Furthermore, questions could argue for an incompatible conclusion like : Are there (better) reasons not to do plan A or plan B? We classify our set of critical questions into 6 layers (also presented in Figure 3).

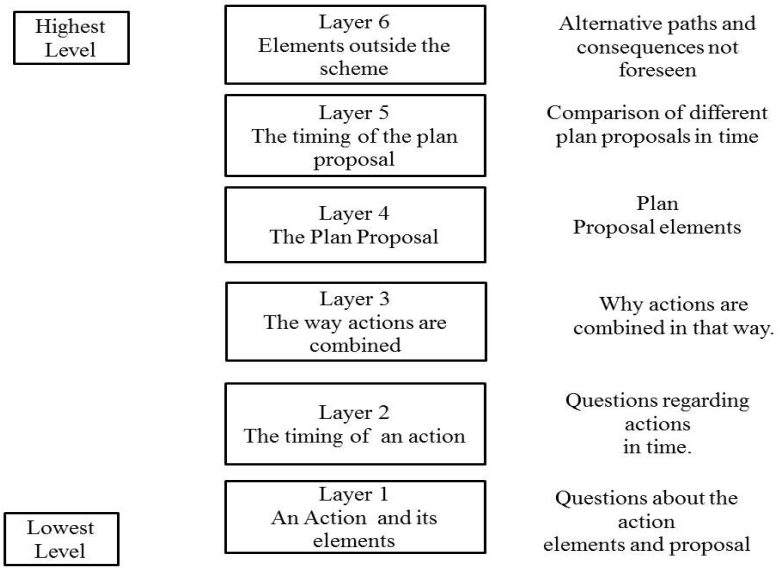


Fig. 3. Critical Question Layers.

- Layer 1.- An action and its elements (Lowest level).
- Layer 2.- The timing of a particular action.
- Layer 3.- The way actions are combined.
- Layer 4.- The plan proposal overall.
- Layer 5.- The timing of the plan proposal.
- Layer 6.- Elements outside the scheme (alternative paths or consequences not foreseen) (Highest Level).

The layers are derived from the different categories of critical questions that relate to the different elements of the argumentation scheme. Each layer groups questions according to the level of detail on which they focus. At the plan proposal level, for example, the critical questions are all those which are independent of the way in which actions are composed inside the plan i.e. the way in which actions are combined. This classification allows us then, to consider questions at each layer separately. Furthermore, this classification gives us elements to create a strategy to select critical questions in a dialogue. Having the critical questions classified an agent could pick a layer and narrow the scope of available questions. An agent then could focus on a specific

level of the proposal *e.g.* either the plan proposal or specific sequences of actions. A strategy like this involves a dialogue-protocol where rules to issue such questions are specified. It could be that the answer to a critical question in one layer imposes constraints within another layer, so this may affect the optimal ordering in which the layers are addressed. Appropriate participant strategies, and their possible relationships with the dialogue protocol, are left for future work.

Our set of critical questions is based on the set developed for action proposals in command dialogues presented in [5]. The complete list of 66 critical questions necessary to comprehensively question all relevant aspects of the plan proposals is presented in [14]. We believe this analysis enables plan proposals to be questioned in a comprehensive way in order to be justified. We present here example questions for each layer.

Layer 1. An action α and its elements (9 questions).
- Is the action α possible?
- Are the action invariants conditions as stated by proponent?
- Are the termination conditions as described as possible?

Layer 2. The timing of an action (10 questions).
- What is the earliest time the action α can start?
- Is the action α possible to finish at the specified time?
- What is the earliest time the action α can end?

Layer 3. The way actions are combined (7 questions).
- (For sequential actions) Can the order of the actions be changed?
- (For concurrent actions) Is there a conflict in any of the invariants conditions of the actions?

Layer 4. The plan proposal (18 questions).
- Are the current circumstances R as stated by proponent?
- Is the value v indeed a legitimate value?
- Can the desired goal G be realized?

Layer 5. The timing of the plan proposal (11 questions).
- Is the starting point for the plan PL fixed? If not, what is the range allowed?
- Is the plan PL possible at the specified time?

Layer 6. Elements outside the scheme (11 questions).
- Does performing the plan PL have a side effect which demotes some other value v?
- Is there an alternative plan PL to realize the same goal G?

4 Example

To illustrate our approach we will use our argumentation scheme in the context of agents representing organizations in a conflict zone. The example was first introduced in [8] and also used in [16] to illustrate a similar problem regarding planning and dialogues for autonomous agents. The situation is the following: two agents, one representing an Non-Governmental Organization (NGO) and one representing a peace keeping force (KF), are working in a conflict zone.

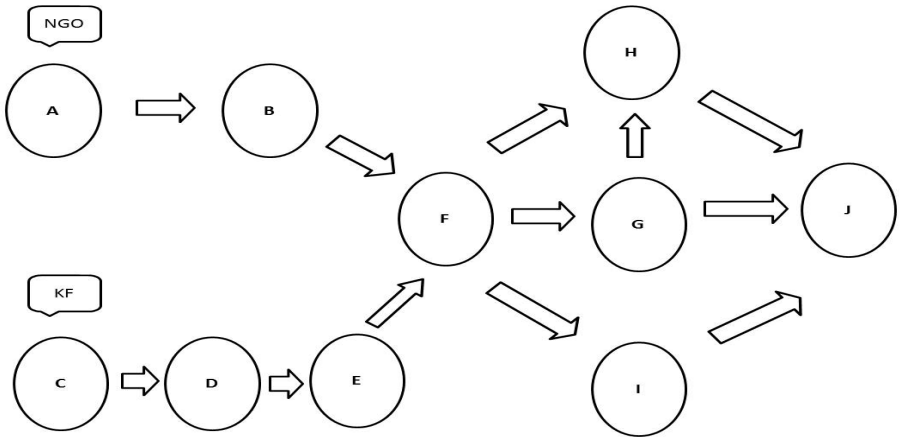


Fig. 4. Example NGO.

The initial conditions are: Agent NGO is based at zone A and agent KF is based at zone C. The joint-goal is that agent NGO reaches zone J safely to help the villagers there. A initial sub-goal is to meet in zone F. The values involved are: v_1 representing humanitarian help and v_2 representing NGO security. The restrictions are: NGO can traverse the routes (A,B),(B,F),(F,H),(I,J) independently, but for all the other routes it needs to be accompanied by KF. KF can traverse any route. At any time, some disruption may flare up at zone G. If this happens, only the KF agent has the surveillance data to know this is happening, and must go to zone G to suppress the disturbance. Furthermore, NGO cannot traverse the routes where zone G is involved if there is a conflict. Finally agent NGO is able to see all the zones and routes only when in zone F. The routes between zones are shown as arcs in Figure 4. The list of possible actions and joint-actions is presented in Table 3.

Table 3. Actions and joint-actions.

Actions	**Joint-actions**
$\alpha_0 = move_{NGO}(X, Y)$	$j_0 = (idle_{NGO}, idle_{KF})$
$\alpha_1 = idle_{NGO}(X)$	$j_1 = (idle_{NGO}, control_{KF})$
$\alpha_2 = move_{KF}(X)$	$j_2 = (idle_{NGO}, move_{KF})$
$\alpha_3 = move_{KF}(X, Y)$	$j_3 = (move_{NGO}, idle_{KF})$
$\alpha_4 = control_{KF}(X)$	$j_4 = (move_{NGO}, control_{KF})$
	$j_5 = (move_{NGO}, move_{KF})$

Our strategy to coordinate the agents is based on a persuasion dialogue where the NGO agent propose a plan and engages in a dialogue where KF need to accept all the actions in the plan to execute it. Another strategy could involve agents creating a

plan from the top following a deliberation dialogue. As mentioned in section 3.1, a dialogue-protocol for engaging in such dialogues is left for future work.

We present now a possible dialogue between 2 agents based on the scenario presented above (possible plans PL_1, PL_2, PL_3 are presented in table 4).

1. NGO presents a proposal with plan PL_1 to reach zone F and promote v_1 humanitarian help (the generation and agreement on sub-goals is out of the scope of this paper).
2. KF present a question: Does performing the plan PL_1 have a side effect which demotes the value v_2 NGO security?
3. NGO provides justification asserting none of the effects demote v_2.
   (The value of having a set of critical questions related to a plan proposal is that an agent has several options to questions and/or attack the elements presented. In this example we only present one question for space reasons. We believe the exchange of arguments using critical questions in a dialogue allows agents to choose the best possible plan).
4. KF accepts plan PL_1. (From this point we assume the plan PL_1 is executed and agents have a new view of the world. Agents are now in zone F).
5. NGO presents plan PL_2 to reach zone J (goal) and promote values v_1 and v_2 (Different sets of values involved may lead to different plans).
6. KF rejects PL_2 and provides evidence that shows demotion of value v_2 (Agent KF detects a conflict in zone G).
7. NGO accepts KF rejection.
8. KF presents plan PL_3 to reach zone J.
9. NGO rejects joint-action $j_5 = (move_{NGO}(H, J), (sodfe), control_{KF}(G))$ [7] referring to a constraint in NGO domain not evident to KF.(Agent cannot travel alone on route (H-J)). The plan is then partially accepted by NGO. From here agents exchange arguments in the action level, assuming the first action of PL_3 : $j_5 = (move_{NGO}(F, H), (e), move_{KF}(F, G))$ was accepted.
10. KF proposes a modification to plan PL_3. We assume the modification is based on a replanning process based on new information. The new sequence proposed is (j_1, j_2, j_5).
11. NGO accepts modification to plan PL_3. and dialogue-goal is reached.

A detailed dialogue in AATS terms is presented in table 5. This detailed example-dialogue represents joint-states with a sub-index $(q_0, .., q_{18})$. Each joint-state represent the state of agent NGO, the state of agent KF and the conflict status. For example, the initial state q_0 is given by the function $\pi(q_0) = \{In(A)_{NGO}, In(C)_{KF}, conflict(0)\}$.

5 Related Work

Our approach is influenced by work on argumentation for practical reasoning [2] and dialogues about plans [6, 15, 16]. Regarding dialogues and plans, Tang, Norman and

[7] Action sequences are specified using the action combination analysis of section 2.2. $(sodfe)(j_0, j_1)$ means: joint-action j_0 Starts, Overlaps, During, Finishes or Equals joint-action j_1.

Table 4. Plans.

Time	Plan
	Plan PL_1 to reach zone F
1	$j_5 = equals(move_{NGO}(A,B), move_{KF}(C,D))$
2	$j_5 = equals(move_{NGO}(B,F), move_{KF}(D,E))$
3	$j_2 = equals(idle_{NGO}(F), move_{KF}(E,F))$
	Plan PL_2 to reach zone J
4	$j_5 = equals(move_{NGO}(F,G), move_{KF}(F,G))$
5	$j_5 = equals(move_{NGO}(G,J), move_{KF}(G,J))$
	Plan PL_3 to reach zone J
4	$j_5 = equals(move_{NGO}(F,H), move_{KF}(F,G))$
5	$j_4 = sodfe(move_{NGO}(H,J), control_{KF}(G))$
	Modified plan PL_3 to reach zone J
4	$j_5 = equals(move_{NGO}(F,H), move_{KF}(F,G))$
5	$j_1 = sodfe(idle_{NGO}(H), control_{KF}(G))$
6	$j_2 = sodfe(idle_{NGO}(H), move_{KF}(G,H))$
7	$j_5 = sodfe(move_{NGO}(H,J), move_{KF}(H,J))$

Parsons in [16] establish a model for individual and joint agents' actions suitable for describing the behaviour of a multi-agent team, including communication actions. Tang *et al's* work has been focused on setting a basis for implementing multi-agent planning dialogues based on argumentation that take into account the communication needs for the plan to be executed successfully. The model uses policies to generate plans and the communication needs are embedded in the policy algorithm generation. From the work of Tang *et al.* we are particularly interested in the techniques used to combine planning and dialogue models using policies. In our approach agents propose and justify previously created plans and then engage in a dialogue to justify the actions and possibly modify the plan. The approach in Tang *et al.* embeds the communication policy in the planning algorithm.

In [6] Belesiotis, Rovatsos and Rahwan develop an argumentation mechanism for reconciling conflicts between agents over plan proposals. The authors extend a protocol where argument-moves enable discussion about planning steps in iterated dispute dialogues as presented in [9]. The authors then introduce a logic for arguments about plans based on the situation calculus. From this approach we are interested in their protocol based on iterated disputes. We plan to modify the approach extending the way a plan proposal makes use of critical questions in the dispute tree.

Another approach is presented in [15]. Onaindia *et al.* present the problem of solving cooperative distributed planning tasks through an argumentation-based model. The model allows agents to exchange partial solutions, express opinions on the adequacy of candidate solutions and adapt their own proposals for the benefit of the overall task. The argumentation-based model is designed in terms of argumentation schemes and critical questions whose interpretation is given through the semantic structure of a partial order planning paradigm. The approach assumes a lack of uncertainty and deterministic planning actions, thus, focuses only on questions concerned with the choice of actions.

Table 5. Possible Dialogue for the NGO example.

	Agent	Locution	Variables	Comments
1	NGO	ProposePlan(PL_1)	(q_0, PL_1) is q_{11}	Initial state NGO in zone A, KF in zone C.
2			v_1+	Final state NGO and KF in zone F.
3				Promotion of value "humanitarian aid".
4	KF	Question()	$\delta(q_0, q_1, v_2)$ is $-$	Is there a side effect that demotes v_2?
5	NGO	Provide()	$\delta(q_0, q_1, v_2)$ is $=$	Value v_2 is not demoted.
6	KF	AcceptProposal(PL_1)		Proposal accepted for plan PL_1
7	NGO	ProposePlan(PL_2)	(q_{11}, PL_2) is q_{16}	Initial state NGO and KF in zone F.
8			$G \in \pi(q_{16})$	Final state NGO and KF in zone J
9			v_1+ and v_2+	goal reached in q_{16}, values promoted
10	KF	RejectProposal(PL_2)	PL_2	
11		Provide()	$\delta(q_{11}, q_{16}, v_2)$ is $-$	Demotion of v_2
12		Provide()	$conflict \in \pi(q_2)$	
13	NGO	AcceptRejection()		
14	KF	ProposePlan(PL_3)	(q_{11}, PL_3) is q_{18}	NGO and KF in zone F.
15			v_1+ and v_2+	NGO and KF in zone J.
16	NGO	AcceptAction()	j_5	
17		RejectAction()	j_4	
18		Provide()	$(q_7, j_4) \notin Q$	NGO cannot travel alone proposed route.
19	KF	ProposeActions()	j_1, j_2, j_5	Goal reached in q_{17}
20	NGO	AcceptActions()	j_1, j_2, j_5	

The argumentation scheme, based on the scheme for action proposal from [3] is of the form:

In the current circumstances and considering the current base plan Π_i, agent ag_i should perform the refinement step $\Pi^{'}$, which will result in a new partial plan Π_j, which will realise some sub-goals *G*, which will promote some values *V*.

Our work is very similar in approach to this work in the sense that plans should be entities treated at a detailed level when arguing about them. We go further and consider plan proposals in more detail referring to action elements and combination of actions. Furthermore, our argumentation scheme is related to a more comprehensive set of critical questions, giving an agent more options to critique and enhance a proposal. We believe these elements allow an agent to question and/or attack the argument in a more targeted fashion, facilitating the modification of more types of plans and faster identification of differences between participants.

6 Conclusion

Our research aims at contributing to solving problems related to multi-agent planning, where agents need to agree on plans given different views of the world and of other agents' capabilities. We believe our main contribution in this paper is that we have articulated a novel list of critical questions related to an argumentation scheme for plan proposals as different combination of actions including temporal aspects. The critical questions address each element of a proposed plan and so they are comprehensive with

respect to the representation we have chosen for plan proposals. We believe every component and every interaction of components in our representation of a proposal for plan is subject to a possible critical question

The importance of this work is that it enables a proposal for plan execution to be considered rationally and automatically by software agents engaged in deliberation over the plan of action. The critical questions enable the proposed plan to be questioned/challenged in a comprehensive and organized manner, and to be clarified or defended in response, as appropriate. Indeed, it is possible to use the critical questions as the basis for an agent dialogue game protocol in which one participating agent may propose, and then clarify or defend a plan of action, while other agents question or challenge this proposal. For example, Atkinson and colleagues in [3] develop such a dialogue protocol for proposals for single actions. Whether the proposed plan of action survives such questions and attacks in the dialogue will depend upon the facts about the world underlying the proposal, and the ability of the proponent agent to defend his proposal from attack. Consequently, the acceptability or otherwise of the proposed plan will depend upon the outcome of the multi-agent dialogue based upon the critical questions, and vice-versa.

The multi-agent dialogue is a form of game-theoretic semantics for the statement of a plan of action in the same way as Hintikka's game-theoretic semantics for first-order logic [12] interprets well-formed formulae involving existential and universal quantifiers as equivalent to two-party games between a proponent and an opponent of some proposition. Our approach will interpret proposals for plans in terms of dialogue games between agents defending and attacking the proposal. Our work in this paper can therefore be seen as part of a larger effort to develop computational semantics for plans of actions between interacting software agents [13].

Future work includes analysis on how to represent formally action elements not yet accounted for in the formalization, such as the duration of actions and action invariants. One limitation in this work is that we only considered plans comprising two actions, effectively a plan for each agent; how to decompose these plans into a number of actions and the issues that arise from the interaction of their components is something we will consider in the next phase of our research. To support this theory we will also implement a prototype where agents use a protocol that allows them to engage in dialogues about plan proposals in a single solution.

Acknowledgements Katie Atkinson and Trevor Bench-Capon were partially supported by the FP7-ICT-2009-4 Programme, IMPACT Project, Grant Agreement Number 247228. The views expressed are those of the authors and are not necessarily representative of the project.

References

1. J. F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154, 1984.
2. K. Atkinson and T. Bench-Capon. Practical reasoning as presumptive argumentation using action based alternating transition systems. *Artificial Intelligence*, 171(10-15):855–874, 2007.

3. K. Atkinson, T. Bench-Capon, and P. McBurney. A dialogue game protocol for multi-agent argument over proposals for action. *Autonomous Agents and Multi-Agent Systems*, 11(2):153–171, 2005.
4. K. Atkinson, T. Bench-Capon, and P. McBurney. Computational representation of practical argument. *Synthese*, 152(2):157–206, 2006.
5. K. Atkinson, R. Girle, P. McBurney, and S. Parsons. Command Dialogues. In I. Rahwan and P. Moraitis, editors, *Argumentation in Multi-Agent Systems*, Fifth International Workshop, pages 93–106, Berlin, Heidelberg, 2009. Springer-Verlag.
6. A. Belesiotis, M. Rovatsos, and I. Rahwan. A generative dialogue system for arguing about plans in situation calculus. In P. McBurney, I. Rahwan, S. Parsons, and N. Maudet, editors, *Argumentation in Multi-Agent Systems*, volume 6057 of *Lecture Notes in Computer Science*, pages 23–41. Springer Berlin / Heidelberg, 2010.
7. F. Bex, H. Prakken, C. Reed, and Douglas Walton. Towards a formal account of reasoning about evidence: argumentation schemes and generalisations. *Artificial Intelligence Law*, 11(2-3):125–165, 2003.
8. C. Burnett, D. Masato, M. Mccallum, T. J. Norman, J. Giampapa, M. J. Kollingbaum, and K. Sycara. Agent support for mission planning under policy constraints. In Proceedings of the Second Annual Conference of the ITA, Imperial College, 2008.
9. P. E. Dunne and T. Bench-Capon. Two party immediate response disputes: properties and efficiency. *Artificial Intelligence*, 149(2):221–250, 2003.
10. M. Fox and D. Long. PDDL2.1: An extension to PDDL for Expressing Temporal Planning Domains. *Journal Artificial Intelligence Research (JAIR)*, 20:61–124, 2003.
11. H. Geffner. Perspectives on Artificial Intelligence Planning. In *Eighteenth National Conference on Artificial intelligence*, pages 1013–1023, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.
12. J. Hintikka. *The Game of Language: Studies in Game-Theoretical Semantics and Its Applications*, volume 22 of *Synthese Language Library*. D. Reidel, Dordrecht, The Netherlands, 1983.
13. P. McBurney and S. Parsons. Dialogue games for agent argumentation. In I. Rahwan and G. Simari, editors, *Argumentation in Artificial Intelligence*, chapter 13, pages 261–280. Springer, Berlin, Germany, 2009.
14. R. Medellin-Gasque, K. Atkinson, P. McBurney, and T. Bench-Capon. Critical Questions for Plan Proposals. Technical Report ULCS-11-003, Department of Computer Science, University of Liverpool, UK, March 2011.
15. E. Onaindía, O. Sapena, and A. Torreño. Cooperative Distributed Planning through Argumentation. In *International Journal of Artificial Intelligence*, volume 4, pages 118–136. CESER Publications, 2010.
16. Y. Tang, T. J. Norman, and S. Parsons. A model for integrating dialogue and the execution of joint plans. In *AAMAS '09: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 883–890, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.
17. W. van der Hoek, M. Roberts, and M. Wooldridge. Social laws in alternating time: Effectiveness, feasibility, and synthesis. *Synthese*, 156:1–19, 2007.
18. B. Verheij. Dialectical argumentation with argumentation schemes: an approach to legal logic. *Artificial Intelligence and Law*, 11(2-3):167–195, 2003.
19. D. N. Walton. *Argumentation Schemes for Presumptive Reasoning*. Lawrence Erlbaum Associates, Mahwah, NJ, USA, 1996.
20. D. N. Walton. Justification of argumentation schemes. *Australasian Journal of Logic*, 3, 2005.

Conditional Labelling for Abstract Argumentation

Guido Boella[1], Dov M. Gabbay[2], Alan Perotti[1],
Leendert van der Torre[3], Serena Villata[4]

[1] Dipartimento di Informatica, Università di Torino {guido,perotti}@di.unito.it
[2] King's College London dov.gabbay@kcl.ac.uk
[3] ICR, University of Luxembourg leon.vandertorre@uni.lu
[4] INRIA, Sophia Antipolis serena.villata@inria.fr

Abstract. Agents engage in dialogues having as goals to make some arguments acceptable or unacceptable. To do so they may put forward arguments, adding them to the argumentation framework. Argumentation semantics can relate a change in the framework to the resulting extensions but it is not clear, given an argumentation framework and a desired acceptance state for a given set of arguments, which further arguments should be added in order to achieve those justification statuses. Our methodology, called *conditional labelling*, is based on argument labelling and assigns to each argument three propositional formulae. These formulae describe which arguments should be attacked by the agent in order to get a particular argument *in*, *out*, or *undecided*, respectively. Given a conditional labelling, the agents have a full knowledge about the consequences of the attacks they may raise on the acceptability of each argument without having to recompute the overall labelling of the framework for each possible set of attack they may raise.

1 Introduction

Agents engage in dialogues having as goals to make some arguments acceptable or unacceptable: for instance, *agent A wins the auction* or *agent B is proven guilty*. At each turn, an agent owns a set of possible arguments she can add to the framework: each addition of further arguments to the framework is called a *move*. Argumentation semantics allow us to relate the introduction of a new argument (a *move*) to the resulting justification status of an argument (the *goal*): for instance, *if you defeat argument α then argument β will be labeled undec*. What is missing is a mechanism for making inferences from goals to moves: suppose an agent wants to make an argument β *undec*. How can she compute which arguments to add in order to achieve this goal? What she can do is to try and simulate the introduction of every possible argument she owns to the framework and then compute β's resulting label, comparing it to her goal. Beside this exhaustive approach there is no way, so far, for an agent to know which move to make in order to achieve her goal. Since reaching a goal may require the insertion of several arguments, the complexity of the exhaustive approach is exponential

(cardinality of the powerset) over the number of arguments an agent can add to the framework.

The research question of the paper is:

– How to change an abstract argumentation framework, by introducing new arguments and their associated attacks, in order to have one or more arguments accepted or rejected?

Suppose that two agents, Ag_1 and Ag_2, initiate a dialogue. Ag_1 proposes argument a, as depicted in Figure 1.1. Assume that Ag_2 wants to defeat Ag_1's argument but we have that argument a is *in*, and the only way to have it labelled *out* is to attack it. Thus, Ag_2 attacks a with her new argument b, defeating it. At this turn, as shown in Figure 1.2, it is up to Ag_1 to decide how to proceed in the dialogue. She wants to have her argument a accepted, so she puts forward argument c which attacks b, obtaining the framework in Figure 1.3. In this basic framework, it is straightforward to see which arguments the agents should attack in order to get their arguments accepted. In more complex argumentation frameworks, where also cycles are involved, it is less simple to detect these arguments. Consider the framework depicted in Figure 2: it contains loops and multiple attacks. Suppose that an agent wants to defend argument i: it is not intuitive at all to see which potential modifications of the framework allow her to do that. Moreover, if she has a set A^{ag} of arguments she may add to the framework, she may have to run $2^{|A^{ag}|}$ tests in order to find out whether she can defend i, thus making this process' complexity dependent on the number of possible moves she has.

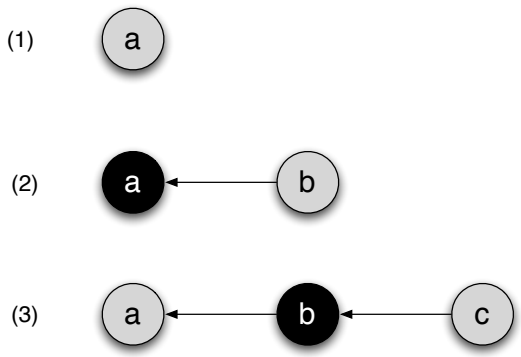


Fig. 1: An argumentation framework with a basic reinstatement.
Arguments labelled *in* are depicted as grey nodes, arguments with label *out* are depicted as black nodes.

Thus, the research question breaks down into the following subquestions:

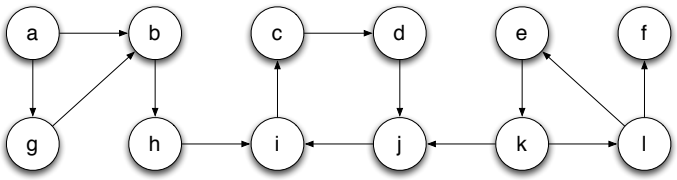


Fig. 2: A more complex argumentation framework.

1. What kind of information can we associate to each argument concerning its possible justification statuses depending on the acceptability of other arguments in the framework?
2. How to compute this information in an efficient way?

We deal with abstract argumentation frameworks [3], where the internal structure of the arguments is left unspecified. We are inspired by Caminada's labelling [2], which assigns to each argument a label *in*, *out*, *undec*, and we extend this idea by assigning a triple of propositional formulae, called *conditional lables*, to every argument in the framework. These formulae are a guide in the dialogic process and suggest which move should be made next. Note that these formulae (and the algorithmic process to compute them) are in no way related to the number of agents: our approach does not depend on the number of arguments and we apply it to a two-agent scenario for the sake of explanation. Consider the framework of Figure 1.3, the conditional label of a for making it accepted is the emptyset because a is already *in* and no "move" is needed to get it accepted. The conditional label, instead, for making a unaccepted is $a \vee c$, because a can be defeated by defeating a itself or c. Depending on the further arguments at her disposal, an agent may not be able to directly defeat an argument and therefore giving all alternatives is required. Conditional labelling assigns a conditional label to each abstract argument in the framework, even if the framework involves one or more cycles.

The implementation of the algorithm of conditional labelling deals with a number of complexity issues, mostly due to loops in the argumentation frameworks: some preprocessing techniques allow to speed up the performances displayed by a straightforward implementation of the conditional labels' theoretical definition.

In this paper, we are interested in introducing the basic ideas of the conditional labelling and explain it using a number of examples. We do not treat belief revision, and we restrict our examples to grounded semantics.

The paper is organized as follows: Section 2 provides the basic concepts of argumentation theory, Section 3 introduces the conditional evaluation of arguments, Section 4 discusses an algorithmical definition of the conditional labelling and some possible optimizations for the implementation. Finally, some conclusions are drawn.

2 Background

We provide the basic concepts and insights of Dung's abstract argumentation [3].

Definition 1. *(Abstract argumentation framework) An abstract argumentation framework is a pair* $\langle \mathcal{A}, \rightarrow \rangle$. *$\mathcal{A}$ is a set of elements called arguments and* $\rightarrow \subseteq \mathcal{A} \times \mathcal{A}$ *is a binary relation called attack. We say that an argument* A_i *attacks an argument* A_j *if and only if* $(A_i, A_j) \in \rightarrow$.

Definition 2. *(Conflict-free, Defence) Let* $C \subseteq \mathcal{A}$. *A set C is conflict-free if and only if there exist no* $A_i, A_j \in C$ *such that* $A_i \rightarrow A_j$. *A set C defends an argument* A_i *if and only if for each argument* $A_j \in A$ *if* A_j *attacks* A_i *then there exists* $A_k \in C$ *such that* A_k *attacks* A_j.

Definition 3. *(Acceptability semantics) Let C be a conflict-free set of arguments, and let* $\mathcal{D} : 2^{\mathcal{A}} \mapsto 2^{\mathcal{A}}$ *be a function such that* $\mathcal{D}(C) = \{A | C\ defends\ A\}$.

- *C is admissible if and only if* $C \subseteq \mathcal{D}(C)$.
- *C is a complete extension if and only if* $C = \mathcal{D}(C)$.
- *C is a grounded extension if and only if it is the smallest (w.r.t. set inclusion) complete extension.*
- *C is a preferred extension if and only if it is a maximal (w.r.t. set inclusion) complete extension.*
- *C is a stable extension if and only if it is a preferred extension that attacks all arguments in* $\mathcal{A} \setminus C$.

The concepts of admissibility, as well as those of Dung's semantics are originally stated in terms of sets of arguments. It is equal to express these concepts using argument *labeling*. This approach has been proposed firstly by Jakobovits and Vermeir [4] and then by Caminada [2] with the aim to provide quality postulates for dealing with the reinstatement of arguments. The simplest example of reinstatement is: argument A_1 attacks argument A_2 and argument A_2 attacks argument A_3. We have that argument A_1 reinstates argument A_3, i.e., it makes argument A_3 accepted by attacking the attacker of A_3. In a reinstatement labeling [2], an argument is labeled *in* if all its attackers are labeled *out* and it is labeled *out* if it has at least an attacker which is labeled *in*.

Definition 4. *(AF-labeling) Let* $\langle \mathcal{A}, \rightarrow \rangle$ *be an abstract argumentation framework. An AF-labeling is a total function* $lab : \mathcal{A} \rightarrow \{in,\ out,\ undec\}$. *We define* $in(lab) = \{A_i \in \mathcal{A} | lab(A_i) = in\}$, $out(lab) = \{A_i \in \mathcal{A} | lab(A_i) = out\}$, $undec(lab) = \{A_i \in \mathcal{A} | lab(A_i) = undec\}$.

Definition 5. *(Reinstatement labeling) Let lab be an AF-labeling. We say that lab is a reinstatement labeling if and only if it satisfies the following:*

- $\forall A_i \in \mathcal{A} : (lab(A_i) = out \equiv \exists A_j \in \mathcal{A} : (A_j \rightarrow A_i \wedge lab(A_j) = in))$ *and*
- $\forall A_i \in \mathcal{A} : (lab(A_i) = in \equiv \forall A_j \in \mathcal{A} : (A_j \rightarrow A_i \supset lab(A_j) = out))$ *and*
- $\forall A_i \in \mathcal{A} : (lab(A_i) = undec \equiv \exists A_j \in \mathcal{A} : (A_j \rightarrow A_i \wedge \neg(lab(A_j) = out)) \wedge \nexists A_k \in \mathcal{A} : (A_k \rightarrow A_i \wedge lab(A_k) = in)$.

3 Conditional labels

Our goal is to enrich each argument with some information about his *vulnerability*, i.e., we want to know how this argument could be successfully (even if indirectly) attacked, defended or made undecided. We purposely restrict our attention to argument defeating, due to two considerations: first of all, attacks are not resources but consequences of the insertion of the arguments and given a couple of arguments the existence of attacks between them is determined and not subject to strategic moves of agents. In second place, the building of an argumentation framework is a monotonic process and arguments can be defeated with new arguments rather than removed from the framework. Hence our proposal is to attach **three** formulae to each argument, meaning respectively

- Which arguments should be attacked in order to have this argument labelled *in*?
- Which arguments should be attacked in order to have this argument labelled *out*?
- Which arguments should be attacked in order to have this argument labelled *undec*?

Given an argumentation framework $\langle A, R\rangle$, we associate to each argument α three formulae: α^+, α^-, $\alpha^?$. We indicate a generic formula associated to argument α as α^*. The language of the formulae is the same:

Definition 6. *(Language of conditional labels)*

- *if* $\beta \in A$, β° *is a formula.*
- $\top$ *and* $\bot$ *are formulae*
- *if* α_1^* *and* α_2^* *are formulae, also* $\alpha_1^* \wedge \alpha_2^*$ *and* $\alpha_1^* \vee \alpha_2^*$ *are.*

We will refer to α^+ (respectively: α^-, $\alpha^?$) formulae as *green* (*red*, *grey*) formulae.

The interpretation of the formulae is: a *green* formula α^+, if satisfied, guarantees that the related argument α is accepted (labelled *in*). The same holds for *red* formulae for *out* labels and *grey* formulae for *undec* labels respectively. The atoms of those formulae are argument names β° or the special values $\top, \bot$.

- β° means *you have to defeat argument* β (to reach your goal)
- $\top$ means *you do not need to do anything* (to reach your goal)
- $\bot$ means *you can not do anything* (to reach your goal)

Figure 3 provides a simple example of a framework with conditional labels.

- Figure 3.1: There is no need to modify the framework in order to achieve a's acceptability (it is already labelled *in*) ($a^+ : \top$); to defeat a you have to defeat a ($a^- : a^\circ$), you can not make a undecidable by defeating any combination of the arguments of the framework ($a^? : \bot$).

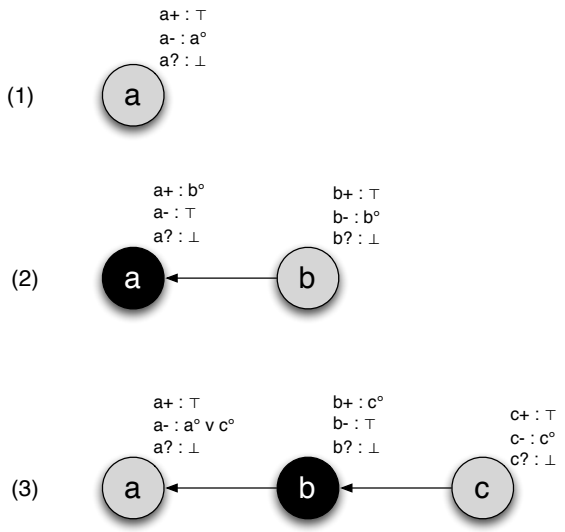


Fig. 3: An argumentation framework with a basic reinstatement and conditional labels

- Figure 3.2: a can be reinstated defeating b ($a^+ : b^\circ$), a is already *out* ($a^- : \top$); b is already *in* ($b^+ : \top$) and can only be defeated by being directly defeated ($b^- : b^\circ$); no argument can be made undecidable by defeating any combination of the arguments of the framework ($a^?, b^? : \bot$).
- Figure 3.3: a is *in* ($a^+ : \top$) and can be defeated by defeating a itself **or** c ($a^- : a^\circ \vee c^\circ$); b is *out* ($b^- : \top$) and can be reinstated defeating c ($b^+ : c^\circ$); c is in ($c^+ : \top$) and can only be defeated by direct (and successful) attack ($c^- : c^\circ$); no argument can be made undecidable by defeating any combination of the arguments of the framework ($a^?, b^?, c^? : \bot$).

Now we can introduce a more formal definition of what conditional labels are and what can they be used for.

Definition 7. *(Disjunctive Normal Form, makeset)*
Let Γ be a propositional formula. dnf (Γ) is the normalization of Γ in Disjunctive Normal Form.
Let makeset$(\bigvee_i \bigwedge_j \alpha_j^i) = \{\{\alpha_1^1, \alpha_2^1, .., \alpha_p^1\}, \{\alpha_1^2, \alpha_2^2, .., \alpha_q^2\}, .., \{\alpha_1^n, \alpha_2^n, .., \alpha_m^n\}\}$.

The *makeset* function translates a dnf-formula in a set of sets of atoms, where each set corresponds to a conjunctive subformula of the dnf-formula in input. For instance, $dnf(a^\circ \wedge (b^\circ \vee c^\circ)) = (a^\circ \wedge b^\circ) \vee (a^\circ \wedge c^\circ)$ and $makeset((a^\circ \wedge b^\circ) \vee (a^\circ \wedge c^\circ)) = \{\{a, b\}, \{a, c\}\}$.

Definition 8. *(Label of an argument in a framework)*
Let $\mathbb{L}(\alpha, \langle A, R\rangle)$ *be the label of argument* α *in the framework* $\langle A, R\rangle$.

Definition 9. *(Defeat)*
Let U *be the universe of arguments and* $A \subset U$*, let* $AF = \langle A, R\rangle$ *be an abstract argumentation framework and let* $\alpha \in U \backslash A$. $defeat(\alpha) = \{\beta \mid \beta \in A, \mathbb{L}(\beta, \langle A \cup \{\alpha\}, R\rangle) = out\}$.

The $defeat$ function gives information about which arguments β of a framework are defeated inserting a new argument α into it. For instance, considering the framework in Figure 1.c with $A = \{a\}$, $defeat(b)$=$\{a\}$, $defeat(c)$=$\{\emptyset\}$. The definition of *defeat* can be easily extended for sets of arguments: it will point out which arguments of a framework are defeated by inserting a set of new arguments. For instance, considering the framework in Figure 1.c with $A = \{a\}$, $defeat(\{b, c\})$=$\{\emptyset\}$

A *move* M is the insertion of a set of arguments into the framework: in the previous example, $\emptyset$, $\{b\}$, $\{c\}$ and $\{b, c\}$ are (possible) moves. Applying a move $M = \{\alpha_1, .., \alpha_n\}$ to a framework $AF = \langle A, R\rangle$ transforms it into a new framework $AF^M = \langle\{A \cup M\}, R\rangle$.

Definition 10. *(Conditional labels' structure)*
A conditional label $\alpha^i : body^i_\alpha$ *(where* α *is an argument,* $i \in \{+, -, ?\}$ *and* $body^i_\alpha$ *is a propositional formula) is a relation between a justification status and a set of targets.*

Let js (for *justification status*) be this function: $js(+) : in$, $js(-) : out$, $js(?) : undec$. js maps label symbols to the acceptability of arguments. The justification status is expressed by the head of the label: α^i means that α is labelled $js(i)$. The set of targets is expressed by the body of the label, and it consists in a set of sets of argument to defeat.

Definition 11. *(Conditional labels' use)*
Given a framework $AF = \langle A, R\rangle$*, an argument* $\alpha \in A$ *with label* $\alpha^i : body^i_\alpha$ *and a move* M*,*

$$(defeat(M) \in makeset(dnf(body^i_\alpha))) \Rightarrow \mathbb{L}(\alpha, AF^M) = js(i)$$

This means: when we modify a framework via a move M we can defeat a set of arguments $defeat(M)$. If this set is one of the allowed target sets for the conditional label of an argument α (that is, if this set belongs to $makeset(dnf(body^i_\alpha))$ for some α, i), then the labelling of α in the resulting framework will be the one expressed by the head of the label α^i (that is, $js(i)$).

In the next sections we will explain how to associate labels to arguments. Problems arise when cycles (loops) are introduced in the framework, since they introduce *undecided* labels and the same argument could be given different labels according to different semantics. In this paper we focus on the *grounded* semantics, since it always allows to compute one single labelling. Our approach can be extended to deal with different semantics, but semantics with multiple or no extensions must be handled with care, in particular when investigating about credulous approaches to multiple extensions semantics.

4 Creating conditional labels

The formal definition of conditional labels we gave is not constructive and therefore the issue about how to actually compute conditional labels has to be addressed. One of the key aspects of argumentation frameworks is the possibility for arguments to influence their own justification status through loops: this is a *global* property of the framework which is hard to instantiate on a single argument. A first approach could be considering the unfolding of the graph (that is, building a tree rooted in a node of the graph such that each path in the tree corresponds to a (possibly cyclic) path stating from the root node in the graph), but this can not be done for two main reasons: first of all, *breaking* the loops causes an irreparable loss of information (and therefore one could end up computing conditional labels for a completely different framework); secondly, the number of unfolding could be exponential over the number of arguments: in this case, the overall complexity is the same of the exhaustive approach (*try all combinations of attacks and see what is the result*), thus making the whole process pointless.

Our approach consists in assigning to each argument a triple of **local** labels (that is, labels created by only taking into account the attackers of the argument) and then using a substitution mechanism to generate the final labels.
The local labels correspond to:

$$a^+ = \bigwedge_{b \ s.t. \ (b,a)\in R} b^-$$

The meaning of this formula is: *in order to ensure a's acceptance, all of a's attackers must be out.*

$$a^- = a^\circ \vee \bigvee_{b \ s.t. \ (b,a)\in R} b^+$$

The meaning of this formula is: *in order to ensure a's rejection, either a is defeated or one of a's attackers is accepted.*

$$a^? = \left(\bigvee_{b \ s.t. \ (b,a)\in R} b^? \right) \wedge \left(\bigwedge_{b \ s.t. \ (b,a)\in R} b^- \vee b^? \right)$$

The meaning of this formula is: *in order to have an argument a undecided, at least one of a's attackers has to be undecided and all of a's attackers must be out or undecided.*
Note that this definition of grounded semantics mirrors Dung's original formulation.
The a° in the second formula means *a has to be defeated* and no substitution is required; b^+, b^- and $b^?$ refer to other formulae and have to be substituted to the actual formulae they refer to.

After this initial definition, the substitution process takes place. It consists in substituting the references to other labels to those labels' actual values.

Simplifications need to be specified:

- $\top \vee \alpha \rightsquigarrow \top$ (you either do nothing or do α: doing nothing is more convenient)
- $\bot \vee \alpha \rightsquigarrow \alpha$ (you can either fail or do α: in order to succeed you have to do α)
- $\top \wedge \alpha \rightsquigarrow \alpha$ (you have to both do nothing and α, therefore α)
- $\bot \wedge \alpha \rightsquigarrow \bot$ (you fail and you have to do α: you still fail)
- $\alpha \wedge \alpha \rightsquigarrow \alpha$
- $\alpha \vee \alpha \rightsquigarrow \alpha$
- $\alpha \vee (\alpha \wedge \beta) \rightsquigarrow \alpha$
- $\alpha \wedge (\alpha \vee \beta) \rightsquigarrow \alpha$

Consider again the framework in Figure 1.3 (reinstatement a-b-c). The initial conditional labels are:

- a^+: $\top$, a^-: a°, $a^?$: $\bot$
- b^+: a^-, b^-: $a^+ \vee b^\circ$, $b^?$: $a^? \wedge (a^? \vee a^-)$
- c^+: b^-, c^-: $b^+ \vee c^\circ$, $c^?$: $b^? \wedge (b^? \vee b^-)$

Substituting in b^* we get:

- b^+: a°, b^-: $\top \vee b^\circ$, $b^?$: $\bot \wedge (\bot \vee a^\circ)$

And after simplifying:

- b^+: a°, b^-: $\top$, $b^?$: $\bot$

Doing the same for c^* we get the conditional labels:

- a^+: $\top$, a^-: a°, $a^?$: $\bot$
- b^+: a°, b^-: $\top$, $b^?$: $\bot$
- c^+: $\top$, c^-: $a^\circ \vee c^\circ$, $c^?$: $\bot$

The conditional labels give us information about the 'static' Caminada labelling of the arguments and also provide us information about what minimal set of arguments we should defeat in order to assign a certain label to a certain argument.

In case of loops, new problems arise: the substitution mechanism can end up visiting the same node multiple times, so some termination techniques have to be addressed. Consider, for instance, the framework $AF = \langle \{a\}, \{(a, a)\} \rangle$. Computing the conditional labels without termination techniques we obtain:

- $a^+ : a^- = \underline{a^+ \vee a^\circ} = a^- \vee a^\circ = a^+ \vee a^\circ \vee a^\circ \rightsquigarrow \underline{a^+ \vee a^\circ} = ...$
- $a^? : \underline{a^? \wedge (a^? \vee a^-)} \rightsquigarrow a^? = \underline{a^? \wedge (a^? \vee a^-)} \rightsquigarrow ...$

Simplification rules keep the size of formulae under control, but both in a^+ and $a^?$ we end up cycling among the same set of labels without termination. The main consideration is that, according to the definition we have given so far, the substitution process goes on until it reaches unattacked arguments (the only ones which do not require further substitution). But if a framework's component is a loop with no ingoing arcs, this will never happen. Moreover, considering the a^+ label in the previous example, one could notice that both a^+ and a^- appear in the label: this is, intuitively, an unsatisfiable request. Therefore, termination rules have to be applied.

Let $i, j \in \{+, -, ?\}$. If α^i appears in the body of α^j:

- if $i = j = ?$, $\alpha^i \rightsquigarrow \top$
- else, $\alpha^i \rightsquigarrow \bot$

We express our termination conditions as simplification rules. The meaning is the following: if, substituting in the body of a conditional formula for an argument α, a conditional formula over the same argument is reached, the argument α belongs to a loop. So in this case the $a^?$ label is satisfied while a^+, a^- are not: if there is no way to give this argument an *in-out* label navigating the whole loop, it is pointless to go through the whole loop again.
Applying these rules to the previous example we get:

- $a^+ : a^- \rightsquigarrow \bot$
- $a^- : a^+ \vee a^\circ \rightsquigarrow \bot \vee a^\circ \rightsquigarrow a^\circ$
- $a^? : a^? \wedge (a^? \vee a^-) \rightsquigarrow a^? \rightsquigarrow \top$

which is exactly what we want to obtain: there is no way to make a *in* ($a^+ : \bot$), a can be directly defeated ($a^- : a^\circ$), a is already *undec* so there is no need to do anything in order to make it *undec* ($a^? : \top$).

We now present some examples of conditional labelling.
Consider the example visualized in Figure 4.1. The basic labels are:

- $a^+ : b^-$, $a^- : b^+ \vee a^\circ$, $a^? : b^?$
- $b^+ : a^-$, $b^- : a^+ \vee b^\circ$, $b^? : a^?$

Solving the labels, for a we get $a^+ : b^\circ$, $a^- : a^\circ$, $a^? : \top$, and this is exactly what we want to obtain.
Consider the example visualized in Figure 4.2. The basic labels are:

- $a^+ : \top$, $a^- : a^\circ$, $a^? : \bot$
- $b^+ : a^- \wedge c^-$, $b^- : a^+ \vee c^+ \vee b^\circ$, $b^? : (a^? \vee c^?) \wedge (a^- \vee a^?) \wedge (c^- \vee c^?)$
- $c^+ : b^-$, $c^- : b^+ \vee c^\circ$, $c^? : b^?$

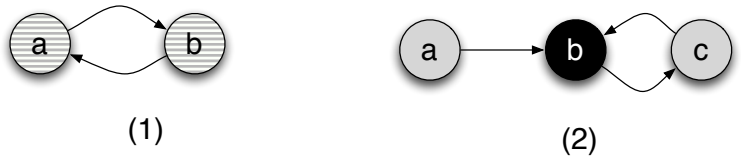


Fig. 4: Basic frameworks. As in the previous figures, plain grey nodes represent *in* arguments and black nodes represent *out* arguments. *undec* arguments are depicted as dashed grey nodes.

Consider argument b: it is *out*, but can be labelled *in* if we attack both a and c or *undec* if we attack a (thus activating the $b-c$ loop). We compute the conditional labels in the following way:

b^+ : $a^- \wedge c^-$
$= a \wedge (b^+ \vee c^\circ)$
$= a^\circ \wedge (\bot \vee c^\circ)$
$\rightsquigarrow a^\circ \wedge c^\circ$ (b can be labelled *in* by defeating a and c)

b^- : $a^+ \vee c^+ \vee b^\circ$
$= \top \vee b^- \vee b^\circ$
$= \top \vee \bot \vee b^\circ$
$\rightsquigarrow \top$ (no move is required in order to label b *out*)

$b^?$: $(a^? \vee c^?) \wedge (a^- \vee a^?) \wedge (c^- \vee c^?)$
$= (\bot \vee b^?) \wedge (a^\circ \vee \bot) \wedge ((b^+ \vee c^\circ) \vee b^?)$
$\rightsquigarrow (b^?) \wedge (a^\circ) \wedge ((b^+ \vee c^\circ) \vee b^?))$
$= (b^?) \wedge (a^\circ) \wedge ((\bot \vee c^\circ) \vee \top)$
$\rightsquigarrow (b^?) \wedge (a^\circ) \wedge (\top)$
$= (\top) \wedge (a^\circ) \wedge (\top)$
$\rightsquigarrow a^\circ$ (b can be labelled *undec* by defeating a)

Our approach can be decomposed in four phases:

1. associate each argument to three base labels,
2. compute conditional labels by substitution,
3. find target sets (for instance, by dnf-normalizing the formulae),
4. find a move such that it satisfies a target set of the goal formula.

The biggest challenge lies in step (2), because the substitution process for each formula has the size of the framework as upper bound and the same substitutions take place several times, especially in highly connected frameworks. A support for implementation can be a preprocessing phase of loop detection: loops are the main cause of complexity in label substitution, and knowing which loops an argument belongs to can help propagating activation-deactivation conditions. We call *active* a loop of arguments such that all arguments are labelled *undec* under grounded semantics, *not active* otherwise. Attacking some argument in order to make the arguments of the loop switch from *undec* to *in* or *out* is what

we call *deactivating* the loop; we call the opposite process *activating* the loop. For instance, in the framework in Figure 5.1, the *b*-*c*-*e*-*f* loop is active.

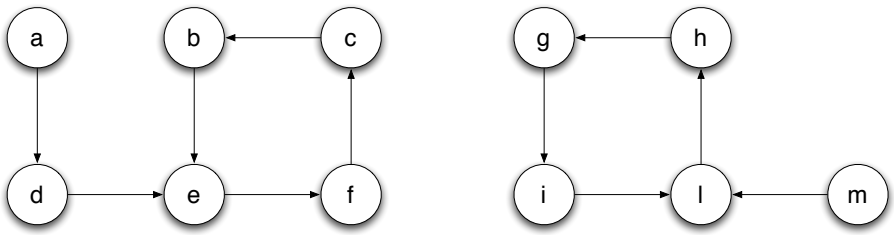


Fig. 5: Two argumentation frameworks with odd cycles

Some conditional labels are:

- $b^+ : c^\circ \vee e^\circ \vee a^\circ = f^+ = c^- = e^-$
- $b^- : b^\circ \vee f^\circ = f^- = c^+ = e^+$
- $b^? : \top = f^? = c^? = e^?$

According to the definition and the substitution algorithm, all those labels would be computed sequentially. But they just mirror the possible deactivations of the cycle, splitted in two sets according to the position (even/odd) of arguments along the cycle. So detecting the cycle one could compute the conditional labels of a single argument and then copy them (alternating from green to red formulae according to even/odd path) for each argument in the loop. This also holds for activation conditions for not active loops (like the one in Figure 5.2) and can be easily extended to odd-length cycles.
Therefore, loop detection can be a major improvement for our algorithm's performances.
We developed a methodology based on computing the powers of the adjacency matrix of the graph. Consider the framework visualized in Figure 6: there is a single loop (a) and two bigger ones (*b*-*c* and *d*-*e*-*f*), plus edges which do not belong to loops. The adjacency matrix of the framework is represented in Table 1 (let it be m^1).

In fact, m^1 gives us information about self-loops in the framework: the arguments attacking themselves correspond to the 1 on the main diagonal of m^1: in our example, a. We express this property as $a \in diagonal(m^1)$
But what happens if we compute $m^2 = m^1 * m^1$? On the n-th element of the main diagonal of m^2 we will have a 1 iff the n-th argument is able to reach itself in n steps: that is, if it belongs to a n-deep loop. This is easy to constructively prove by showing how a multiplication between matrices is made. So, for a framework $AF = \langle A, R \rangle$ with $\mid A \mid = n$ we can just compute m^2, m^3, .. m^n to detect all loops.

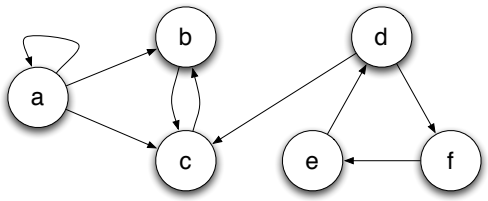


Fig. 6: Frameworks with loops

	a	b	c	d	e	f
a	1	1	1	0	0	0
b	0	0	1	0	0	0
c	0	1	0	0	0	0
d	0	0	1	0	0	1
e	0	0	0	1	0	0
f	0	0	0	0	1	0

Table 1: Adjacency matrix for the framework in Figure 6

Note that in m^1 we detect a, in m^2 b and c, in m^3 d,e,f and no new loop is detected in m^4. Notice that $\alpha \in diagonal(m^p) \Rightarrow \alpha \in diagonal(m^{k*p}), \forall k \in \mathbb{N}$. For instance, in the previous example, $a \in diagonal(m^p) \forall p > 0$ and $b, c \in diagonal(m^2), diagonal(m^4)$. This redundancy of information can be overcome by cross-checking or by removing the elements on the diagonal of the adjacency matrix before multiplying it again.
The number of arguments is the upper bound for loop depths but can be narrowed down in several ways, for instance by detecting connected components or pruning siphons and traps: in the first case, the deepest loop consists of the maximal values over the number of arguments of each connected component; in the second one, siphons and traps can not be part of loops, thus allowing the lowering of the upper bound.

m^2	a	b	c	d	e	f
a	1	1	1	0	0	0
b	0	1	0	0	0	0
c	0	0	1	0	0	0
d	0	1	0	0	1	0
e	0	0	1	0	0	1
f	0	0	0	1	0	0

m^3	a	b	c	d	e	f
a	1	1	1	0	0	0
b	0	0	1	0	0	0
c	0	1	0	0	0	0
d	0	0	1	1	0	0
e	0	1	0	0	1	0
f	0	0	1	0	0	1

m^4	a	b	c	d	e	f
a	1	1	1	0	0	0
b	0	1	0	0	0	0
c	0	0	1	0	0	0
d	0	1	1	0	0	1
e	0	0	1	1	0	0
f	0	1	0	0	1	0

Table 2: powers of adjacency matrix

5 Related Work

Conditional labelling is closely related to the dialogues games [5, 1]. In argumentation theory, such games regulate dialogues where two parties argue about the tenability of one or more claims or arguments, each trying to persuade the other participant to adopt their point of view. Such dialogues are often called persuasion dialogues. Among others, Prakken [5] presents a formal framework for a class of argumentation dialogues, where each dialogue move either attacks or surrenders to a preceding move of the other participant. For instance, each claim, why and since move is viewed as an attacking reply and each concede move is a surrendering reply.

Amgoud and Hameurlain [1] argue that a strategy is a two steps decision process: i) to select the type of act to utter at a given step of a dialogue, and ii) to select the content which will accompany the act. The first step consists of selecting among all the acts allowed by the protocol, the best option which according to some strategic beliefs of the agent will at least satisfy the most important strategic goals of the agent. The second step consists of selecting among different alternatives, the best one which, according to some basic beliefs of the agent, will satisfy the functional goals of the agent.

Roth et al. [6] start from two principles: i) the outcome of a dispute depends on the strategies actually adopted by parties, but ii) this does not mean that the outcome can never be predicted because by using game theoretical solution concepts, the actions themselves can often be found. They use defeasible logic in combination with standard probability calculus in order to prove that a defeasible proof holds, on the basis of the probabilities assigned to the premises. This probability of a claim was then interpreted in the game theoretical sense as the payoff for the proponent of the claim.

In comparison with this kind of frameworks, we share the idea that the first step consists in choosing the next move depending on the strategies of the agents. The differences are that we are not interested in providing a complete framework for argumentation dialogues games, we aim at providing a tool which can be used in those systems and which can be integrated with strategies. We do not restrict our framework to deal with two agents, and we extend the well-known argumentation labelling in order to provide a complete information about the argumentation framework on which it is applied.

6 Summary

In this paper, we present a new kind of argument labelling, called conditional labelling. Conditional labelling allows to associate to each argument the information concerning its possible justification statuses, depending on the changes in the framework. In particular, we express this information by means of propositional formulae which express which arguments should be attacked in order to get the desired argument accepted, not accepted, or undecided. While it is quite straightforward to assign those conditional labels in argumentation frameworks

without cycles and multiple attacks, it is rather complicated in the general case. When an argumentation framework with cycles is considered, it is possible to have in the conditional label α^* of an argument another α^* because the conditional labelling algorithm, using substitution, looks for all the attackers of the node until it finds the node itself. The conditional labelling allows the agents to avoid the exhaustive search of all the possible combinations in adding new arguments, and decreases the exponential complexity this search requires. Loop detection via powers of the adjacency matrix is proposed as a preprocessing mechanism to compute common labels among the arguments in a loop.
Future work addresses several issues: first of all, a deeper investigation on the complexity results related to the computation of the new labellings is necessary. From a purely argumentative perspective, it would be nice to find out how conditional labels can be useful *after* a move: that is, if the previous information can be used to compute new conditional labels after the framework has been modified. Associating a cost concept to moves, our labelling lets agents link action costs to goals' outcomes, and can therefore be used as an underlying mechanism to develop strategies in a game theoretical context.

References

1. L. Amgoud and N. Hameurlain. A formal model for designing dialogue strategies. In H. Nakashima, M. P. Wellman, G. Weiss, and P. Stone, editors, *Autonomous Agents and Multiagent Systems (AAMAS)*, pages 414–416. ACM, 2006.
2. M. Caminada. On the issue of reinstatement in argumentation. In M. Fisher, W. van der Hoek, B. Konev, and A. Lisitsa, editors, *Journées Européennes sur la Logique en Intelligence Artificielle (JELIA)*, volume 4160 of *Lecture Notes in Computer Science*, pages 111–123. Springer, 2006.
3. P. M. Dung. 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.
4. H. Jakobovits and D. Vermeir. Robust semantics for argumentation frameworks. *J. Log. Comput.*, 9(2):215–261, 1999.
5. H. Prakken. Coherence and flexibility in dialogue games for argumentation. *J. Log. Comput.*, 15(6):1009–1040, 2005.
6. B. Roth, R. Riveret, A. Rotolo, and G. Governatori. Strategic argumentation: a game theoretical investigation. In *International Conference on AI and Law (ICAIL)*, pages 81–90. ACM, 2007.

Multi-sorted Argumentation

Tjitze Rienstra[3] and Alan Perotti[2] and Serena Villata[4] and Dov Gabbay[1] and Leendert van der Torre[3] and Guido Boella[2]

[1] Dept. Computer Science, King's College London
dov.gabbay@kcl.ac.uk
[2] Dip. di Informatica, University of Turin
perotti@di.unito.it,guido@di.unito.it
[3] Computer Science and Communication, University of Luxembourg
tjitze.rienstra@uni.lu,leendert@vandertorre.com
[4] INRIA Sophia Antipolis
serena.villata@inria.fr

Abstract. In the theory of abstract argumentation, the acceptance status of arguments is normally determined for the complete set of arguments at once, under a single semantics. However, this is not always desired. For example, in multi-agent systems, the provenance of the arguments and the competence of the agents often suggest different evaluation criteria for different arguments. In this paper, we extend the notion of an argumentation framework to a multi-sorted argumentation framework. In a multi-sorted argumentation framework, the arguments are partitioned into a number of cells, where each cell is associated with a semantics under which its arguments are evaluated. We demonstrate our theory with a number of examples, and we relate our theory to the theory of modal fibring of argumentation networks.

1 Introduction

An abstract argumentation framework is composed of a set of arguments and an attack relation among them. When an argument attacks another argument, they cannot be accepted simultaneously. Therefore, a semantics is chosen to determine which arguments are finally acceptable. This semantics is applied to the entire framework. However, this choice leads to a number of limitations when we try to use this framework in multiagent systems, for instance. In reasoning about trust, each agent evaluates the arguments of the other agents by looking at the competence of the agents proposing the arguments. Depending on the competence of the agents, a different kind of evaluation, more skeptical or more credulous, is addressed.

In this paper, we answer the following research question:

- How to evaluate the arguments of a single abstract argumentation framework under different semantics?

The following example demonstrates the need for evaluating arguments of a single argumentation framework under different semantics.

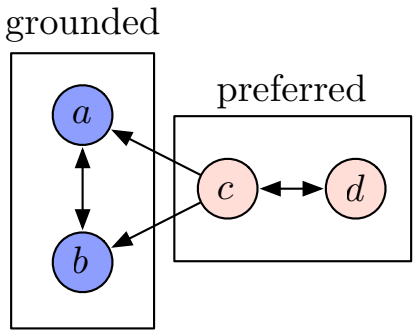


Fig. 1. An argumentation framework evaluated under different semantics.

Consider the framework visualized in figure 1. It could represent a situation where a mother and her son argue about how the son should spend his money. The son has €100 to spend. Arguments a and b are the son's, and they represent:

a I should take €75 to town tonight, to spend it on alcohol.
b I should buy a €100 pocket knife, to protect myself on school grounds.

Arguments c and d are the mother's, and they represent:

c You need a €50 school book, and you will pay for it yourself.
d I will pay for the €50 school book myself.

Both parties have a dilemma, and seek to make one of their arguments true (i.e. the evaluate them credulously). Moreover, their decisions affect each other; if the son has to pay for the book himself, both of the arguments he considers are defeated, and he cannot spend the money like he wants to. However, if the mother decides to pay for the book herself, it does not mean that the son can spend his money irresponsibly.

The framework shown in figure 1 can be seen as combining the arguments of two agents. The conventional theory of abstract argumentation allows a choice of semantics only for the complete set of arguments. Consider the evaluation from the mother's point of view. Clearly, the mother does not regard the son as competent, as far as this issue is concerned, and will choose to evaluate his arguments skeptically. But a grounded evaluation, while appropriate as far as arguments a and b are concerned, makes it impossible to choose between c and d. On the other hand, a preferred evaluation, while allowing a choice between c and d, does not exclude the undesirable outcome that a or b are accepted. A more appropriate evaluation requires a grounded evaluation of the son's arguments, while at the same time her own arguments are evaluated under a preferred semantics.

The research question breaks down into the following sub-questions:

– How to formally define the notion of multi-sorted argumentation?
– How to apply multi-sorted argumentation to reasoning about trust?

In the formalization that we propose, multi-sorted argumentation is based on two elements: a regular argumentation framework and a sorting. A sorting supplements the argumentation framework with information on how the framework is divided into cells, and which cell in the sorting is to be evaluated under which semantics. A sorted extension is a set of arguments that are acceptable with respect to the sorting. We prove a number of properties of sorted extensions, that we consider to be desirable. For example, sorted extensions should be conflict-free. Moreover, some properties of the semantics associated with each cell should be preserved, i.e. if the semantics of all cells are admissible (resp. complete), then the sorted extensions should also be admissible (resp. complete).

In reasoning about trust, the competence of an agent with respect to a particular topic influences the evaluation of the other agents. Moreover, the "assignment" of a particular semantics to one or more arguments is not fixed, but may change over time. For instance, an agent, who was an expert of neural networks 10 years ago and whose arguments were evaluated under the preferred semantics, but now works on economics, requires now a more skeptical evaluation of its arguments on neural networks. We will demonstrate our formalization by applying it to a similar scenario.

The paper is organized as follows: Section 2 provides the basic concepts of argumentation theory; in section 3 we introduce the notions of sortings and sorted extension, and in section 4 we study some properties of sorted extensions; section 5 discusses an application of multi-sorted argumentation to reasoning about trust; and in section 6 we relate our theory to the theory of modal fibring. Finally, sections 7 and 8 present related work, conclusions and future work.

2 Preliminaries

The following definitions set forth the basics of Dung's well-known theory of argumentation [4].

Definition 1 (Argumentation Framework) *We assume as given a set $\mathcal{U}$, called the* universe of arguments*. An* argumentation framework *$\mathcal{AF}$ is a pair $\langle A, R\rangle$ with finite $A \subset \mathcal{U}$ and a binary relation $R \subseteq A \times A$, called the* attack relation*.*

Definition 2 (Conflict Free) *Let $\mathcal{AF} = \langle A, R\rangle$ be an argumentation framework. A set $S \subseteq A$ is* conflict free *iff there are no arguments $a, b \in \mathcal{S}$ such that aRb. If S is conflict free, we write $cf(S)$.*

We follow Baroni & Giacomin's [2] generalized approach, where the acceptability of arguments is considered with respect to a designated subset of arguments. This set, which we call the set of qualified arguments, contains the arguments that an extension may consist of. Intuitively, it is used to filter out

arguments that do not qualify for acceptance. This is necessary when we evaluate only a subset of arguments, but at the same time know that some arguments in this subset cannot be accepted due to attacks from outside the subset.

Definition 3 (Defense) *Let $\mathcal{AF} = \langle A, R\rangle$ be an argumentation framework. A set $\mathcal{S} \subseteq A$* defends *an argument a iff $\forall b \in A$ such that bRa, $\exists c \in \mathcal{S}$ such that cRb. Let $D_Q(\mathcal{S}) = \{a \in Q \mid \mathcal{S}$ defends $a\}$.*

Definition 4 (Acceptance Function) *An* acceptance function

$$\mathcal{E} : 2^{\mathcal{U}} \times 2^{\mathcal{U}\times\mathcal{U}} \times 2^{\mathcal{U}} \rightarrow 2^{2^{\mathcal{U}}}$$

is a partial function that associates each argumentation framework $\langle A, R\rangle$ and each set of qualified arguments $Q \subseteq A$, with sets of subsets of A, called extensions*: $\mathcal{E}(\langle A, R\rangle, Q) \subseteq 2^A$.*

The following definitions are equivalent to those in Dung's original theory, if we set $Q = A$.

Definition 5 (Acceptability Semantics) *Let $\mathcal{AF} = \langle A, R\rangle$ be an argumentation framework and $Q \subseteq A$ a set of qualified arguments. Acceptance functions for conflict free ($\mathcal{E}_{cf}$), admissible ($\mathcal{E}_{ad}$), complete ($\mathcal{E}_{co}$), grounded ($\mathcal{E}_{gr}$) and preferred ($\mathcal{E}_{pr}$) extensions are defined as follows:*

- $\mathcal{S} \in \mathcal{E}_{cf}(\mathcal{AF}, Q)$ *iff* $\mathcal{S} \subseteq Q$ *and* $cf(\mathcal{S})$.
- $\mathcal{S} \in \mathcal{E}_{ad}(\mathcal{AF}, Q)$ *iff* $cf(\mathcal{S})$ *and* $\mathcal{S} \subseteq D_Q(\mathcal{S})$.
- $\mathcal{S} \in \mathcal{E}_{co}(\mathcal{AF}, Q)$ *iff* $cf(\mathcal{S})$ *and* $\mathcal{S} = D_Q(\mathcal{S})$.
- $\mathcal{S} \in \mathcal{E}_{gr}(\mathcal{AF}, Q)$ *iff* $\mathcal{S}$ *is minimal in* $\mathcal{E}_{co}(\mathcal{AF}, Q)$ *w.r.t. set inclusion.*
- $\mathcal{S} \in \mathcal{E}_{pr}(\mathcal{AF}, Q)$ *iff* $\mathcal{S}$ *is maximal in* $\mathcal{E}_{co}(\mathcal{AF}, Q)$ *w.r.t. set inclusion.*

The definitions above are reformulations of those in [2], where, in addition, generalized definitions for stable and CF2 semantics can be found.

3 Multi-sorted Argumentation

We now define the main ingredients of our extension: sortings and sorted extensions. A sorting supplements the argumentation framework with information on how the framework is divided into cells, and which cell in the sorting is to be evaluated under which semantics. In the following definitions, we assume a fixed argumentation framework $\mathcal{AF} = \langle A, R\rangle$.

Definition 6 (Sorting) *A* sorting $\mathbb{S}$ *is a pair $\langle P, T\rangle$, where P is a partition of A and $T : P \rightarrow \{cf, ad, co, gr, pr\}$ a function associating each cell in P to a semantics.*

The following example demonstrates this representation, for the framework shown in figure 1, that was discussed in the introduction.

Example 1. The situation shown in figure 1 is formally represented by a framework $AF = \langle\{a, b, c, d\}, R\rangle$, where aRb, bRa, cRa, cRb, cRd and dRc and sorting $\mathbb{S} = \langle\{C_1, C_2\}, T\rangle$, where $C_1 = \{a, b\}$, $C_2 = \{b, c\}$, $T(C_1) = gr$ and $T(C_2) = pr$.

The following definitions allow the evaluation of the acceptance status of arguments in a cell, given an extension S. The intuition behind these definitions is as follows. Given a cell C and extension S, we determine whether $S \cap C$ is an extension for C, by first restricting C to those arguments that are not defeated by arguments outside C. This set, denoted by C', makes up the arguments of what we call the subframework for C. Next, we further restrict the arguments of C' to those that are defended by S. This set, denoted by C'', contains the arguments in C that are qualified for acceptance.

Definition 7 (Subframework) *Let P be a partition of A, $C \in P$ a cell and $S \subseteq A$ an extension. The* subframework *for C, given S, is argumentation framework $\langle C', R \downarrow C'\rangle$ where $R \downarrow C'$ is the attack relation R restricted to the arguments in C', i.e. $R \downarrow C' = \{(a, b) \in R \mid a, b \in C'\}$ and where $C' = \{a \in C \mid \nexists b \in S \setminus C, bRa\}$*

Definition 8 (Qualified Arguments of a Subframework) *Let $\langle C', R \downarrow C'\rangle$ be a subframework for a cell C and extension S. The* qualified arguments of $\langle C', R \downarrow C'\rangle$*, denoted by C'', are defined as follows.*

$$C'' = \{a \in C' \mid \forall b \in A \setminus C, (bRa \rightarrow \exists c \in S, cRb)\}$$

Given an extension S, we can determine whether it is a sorted extension by checking that for each $C \in P$, we have that $C \cap S$ is an extension of the subframework for C, given the qualified arguments of the subframework for C. The semantics under which the subframework for C is evaluated, is the semantics associated with C.

Definition 9 (Sorted Extension) *A set $S \subseteq A$ is a* sorted extension *of $AF = \langle A, R\rangle$ and $\mathbb{S} = \langle P, T\rangle$ iff for all $C \in P$, we have*

$$C \cap S \in \mathcal{E}_{T(C)}(\langle C', R \downarrow C'\rangle, C'')$$

The sorted acceptance function *$\mathcal{E}_{srt}$ is defined as follows: $\mathcal{S} \in \mathcal{E}_{srt}(AF, \mathbb{S})$ iff S is a sorted extension of AF and $\mathbb{S}$.*

The following example demonstrates the computation of the sorted extensions for the framework used in our earlier example.

Example 1 (Continued). Consider the extensions $S_1 = \{c\}$, $S_2 = \{d\}$ and $S_3 = \{a, d\}$, for the framework shown in figure 1. We will determine whether $S_1, S_2, S_3 \in \mathcal{E}_{srt}(AF, \mathbb{S})$.

- Given S_1, we have $C'_1 = \emptyset$ and $C'_2 = \{c, d\}$ (a and b are defeated by c); and $C''_1 = \emptyset$ and $C''_2 = \{c, d\}$ (no argument is undefended). We have that $S_1 \cap C_1 = \emptyset$ and $\emptyset \in \mathcal{E}_{gr}(\langle C'_1, R \downarrow C'_1\rangle, C''_1)$ and $S_1 \cap C_2 = \{c\}$ and $\{c\} \in \mathcal{E}_{pr}(\langle C'_2, R \downarrow C'_2\rangle, C''_2)$. It follows that $S_1 \in \mathcal{E}_{srt}(AF, \mathbb{S})$.
- Given S_2, we have $C'_1 = \{a, b\}$ and $C'_2 = \{c, d\}$ (no argument is defeated by an argument in another cell); and $C''_1 = \{a, b\}$ and $C''_2 = \{c, d\}$ (no argument is undefended from an attack from another cell). We have that $S_2 \cap C_1 = \emptyset$ and $\emptyset \in \mathcal{E}_{gr}(\langle C'_1, R \downarrow C'_1\rangle, C''_1)$ and $S_2 \cap C_2 = \{d\}$ and $\{d\} \in \mathcal{E}_{pr}(\langle C'_2, R \downarrow C'_2\rangle, C''_2)$. It follows that $S_2 \in \mathcal{E}_{srt}(AF, \mathbb{S})$.
- Given S_3, we have $C'_1 = \{a, b\}$ and $C'_2 = \{c, d\}$; and $C''_1 = \{a, b\}$ and $C''_2 = \{c, d\}$. We have that $S_3 \cap C_1 = \{a\}$ but $\{a\} \notin \mathcal{E}_{gr}(\langle C'_1, R \downarrow C'_1\rangle, C''_1)$. It follows that $S_3 \notin \mathcal{E}_{srt}(AF, \mathbb{S})$.

In conclusion, S_1 and S_2 are sorted extensions, and S_3 is not. There are no other sorted extensions.

4 Properties

In this section we present some properties of sorted extensions. In the following, we say that a semantics x satisfies conflict-freeness if, given any framework $\langle A, R\rangle$, all extensions $\mathcal{E}_x(\langle A, R\rangle, A)$ are conflict-free, and similarly for admissibility and completeness. We then have that the all semantics considered here satisfy conflict-freeness; all semantics except conflict-free satisfy admissibility; and all semantics except conflict-free and admissible satisfy completeness.

We first prove the preservation of the conflict-free, admissible and completeness properties of sorted extensions, i.e. whenever the semantics associated with all cells of the partitioning satisfy these properties, then the sorted extensions satisfy them as well.

Proposition 1. *For any AF and $\mathbb{S} = \langle P, T\rangle$, if $\forall C \in P$, $T(C)$ is a conflict-free semantics, then $\forall S \in \mathcal{E}_{srt}(AF, \mathbb{S})$, S is conflict-free.*

Proof. Let $AF = \langle A, R\rangle$, $\mathbb{S} = \langle P, T\rangle$, and $S \in \mathcal{E}_{srt}(AF, \mathbb{S})$. We know that $\forall C \in P, T(C)$ is a conflict-free semantics. Note that it follows that $\forall C \in P$, $S \cap C$ is conflict free. Now suppose the contrary, i.e. there are $a, b \in S$ s.t. bRa. Let $C \in P$ be the cell s.t. $a \in C$. Because $S \cap C$ is conflict-free, we have $b \in S \setminus C$. Then by definition 7, $a \notin C'$. Because we have that $C'' \subseteq C' \subseteq C$ and $S \cap (C \setminus C'') = \emptyset$, it follows that $a \notin S$. Contradiction. □

Note that, since all the semantics that we consider satisfy conflict-freeness, we have that every sorted extension is conflict-free.

Proposition 2. *For any AF and $\mathbb{S} = \langle P, T\rangle$, if $\forall C \in P$, $T(C)$ is an admissible semantics, then $\forall S \in \mathcal{E}_{srt}(AF, \mathbb{S})$, S is admissible.*

Proof. Let $AF = \langle A, R\rangle$, $\mathbb{S} = \langle P, T\rangle$, and $S \in \mathcal{E}_{srt}(AF, \mathbb{S})$. By property 1 we have that S is conflict-free. We also know that $\forall C \in P, T(C)$ is an admissible semantics. Note that it follows that $\forall C \in P$, $S \cap C$ is admissible w.r.t. the framework $\langle C', R \downarrow C'\rangle$. Suppose now that S is not admissible, i.e. there are $a \in S$, $b \in A$, bRa and $\nexists c \in S$ s.t. cRb. Because S is conflict free, we know that $b \notin S$. Let $C \in P$ be the cell s.t. $a \in C$. Because $S \cap C$ is admissible w.r.t. the framework $\langle C', R \downarrow C'\rangle$, we have that $b \notin C'$. By definition 7 we have that $\forall b' \in (C \setminus C')$, $\exists c' \in S$ s.t. $c'Rb'$. Therefore, $b \notin (C \setminus C')$ and so $b \in A \setminus C$. By definition 8 it now follows that $a \notin C''$. But then, because we have that $C'' \subseteq C' \subseteq C$ and $S \cap (C \setminus C'') = \emptyset$, it follows that $a \notin S$. Contradiction. □

Proposition 3. *For any AF and $\mathbb{S} = \langle P, T\rangle$, if $\forall C \in P$, $T(C)$ satisfies completeness, then $\forall S \in \mathcal{E}_{srt}(AF, \mathbb{S})$, S is complete.*

Proof. Let $AF = \langle A, R\rangle$, $\mathbb{S} = \langle P, T\rangle$, and $S \in \mathcal{E}_{srt}(AF, \mathbb{S})$. Suppose S defends a. We will show that $a \in S$ (i.e. that S is complete). Let $C \in P$ be the cell s.t. $a \in C$. The fact that S defends a, gives us $\forall b \in A$ s.t. bRa, $\exists c \in S, cRb$. By property 1 we have that S is conflict-free. It follows that $\forall b \in A$ s.t. bRa, $b \notin S$. By definitions 7 and 8 it follows that $a \in C''$. Because $S \cap C$ is complete, it follows that $a \in S$. □

Consider the case where the sorting associates all cells with the same semantics. We call this the uniform case. A natural question to ask is whether the set of sorted extensions will then be equivalent to the set of extensions of the framework evaluated under this semantics in the conventional way. We formalize this property as follows.

Definition 10 (Uniform Case Extension Equivalence) *Let $AF = \langle A, R\rangle$ and $\mathbb{S} = \langle \{C_1, \ldots, C_n\}, T\rangle$. Uniform case equivalence holds if and only if*

$$T(C_1) = \ldots = T(C_n) = x \text{ implies } \mathcal{E}_{srt}(AF, \mathbb{S}) = \mathcal{E}_x(AF, A)$$

The following example shows that, in general, this property does not hold.

Example 2. Let $AF = \langle \{a, b\}, R\rangle$, where aRb, bRa; and $\mathbb{S} = \langle \{C_1, C_2\}, T\rangle$, where $C_1 = \{a\}$, $C_2 = \{b\}$ and $T(C_1) = T(C_2) = gr$. The grounded extension of AF is $\emptyset$. We now show that $\emptyset \notin \mathcal{E}_{srt}(AF, \mathbb{S})$: we have $C_1' = \{a\}$ and $C_2' = \{b\}$ (no argument is defeated); and $C_1'' = \emptyset$ and $C_2'' = \emptyset$ (both arguments are undefended). We have that $S \cap C_1 = \emptyset$ and $\emptyset \notin \mathcal{E}_{gr}(\langle C_1', R \downarrow C_1'\rangle, C_1'')$ (the grounded extension of $\langle C_1', R \downarrow C_1'\rangle$ is not a subset of C_1''). It follows that $\emptyset \notin \mathcal{E}_{srt}(AF, \mathbb{S})$.

The fact that uniform case extension equivalence does not hold may be considered to be undesirable. On the other hand, the reason why it does not hold is clear; every cell is evaluated separately, and separate evaluation of a cell C under a semantics x may lead to a result that is different from the result of evaluating the complete framework under semantics x. In any case, an obvious way

to overcome the problem is to replace cells associated with the same semantics with their union.

Finally, an important thing to realize is that, given a cell associated with a certain semantics, say grounded, a sorted extension may not represent a grounded evaluation of the arguments in this cell, when we consider this cell in isolation. This becomes obvious when we look at the following example.

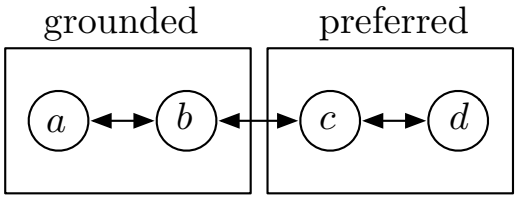


Fig. 2. A sorted argumentation framework.

Example 3. The framework shown in figure 2 is formally represented by $AF = \langle\{a, b, c, d\}, R\rangle$, where aRb, bRa, bRc, cRb, cRd and dRc; and a sorting $\langle\{C_1, C_2\}, T\rangle$, where $C_1 = \{a, b\}$, $C_2 = \{c, d\}$ and $T(C_1) = gr$ and $T(C_2) = pr$.

Consider the extension $S = \{a, c\}$. Note that a is accepted, while the cell $\{a, b\}$ is associated with the grounded semantics. Let us see if S satisfies the conditions for being a sorted extension.

- Given S, we have $C_1' = \{a\}$ and $C_2' = \{c, d\}$ (b is defeated by c); and $C_1'' = \{a\}$ and $C_2'' = \{c, d\}$ (no argument is undefended). We have that $S \cap C_1 = \{a\}$ and $\{a\} \in \mathcal{E}_{gr}(\langle C_1', R \downarrow C_1'\rangle, C_1'')$ and $S \cap C_2 = \{c\}$ and $\{c\} \in \mathcal{E}_{pr}(\langle C_2', R \downarrow C_2'\rangle, C_2'')$. It follows that $S \in \mathcal{E}_{srt}(AF, \mathbb{S})$

Again, this behavior may seem undesirable. It is, however, justifiable. In the example above, selecting c to be accepted accords with the preferred evaluation of the cell $\{c, d\}$. But given this selection, the only complete extension is $\{a, c\}$. Note also that the extension $\{a\}$ is in fact a grounded extension for the subframework $\langle C_1', R \downarrow C_1'\rangle$, where $C_1' = \{a\}$. A possible way to deal with this behavior is to apply a selection criteria for extensions based on a notion of preference. For example, another sorted extension of the framework described above is $\{d\}$. If actual groundedness for the cell C_1 is important, then this extension would be preferred over $\{a, c\}$.

5 An Application in Reasoning about Trust

In reasoning about trust [3], there are, among others, two aspects which are used to evaluate the trustworthiness of the agents: the sincerity, and the competence.

Roughly, the trust in the agent is not absolute, but it is relative to the competence domain, e.g., a doctor is trusted on arguments concerning health. The idea is that each agent is associated with a number of competence domains on which it is trusted or not. Sincerity, on the other hand, is based on the social qualities that the other agents assign to a particular agent. Let us consider the example visualized in figure 3. This example presents the same argumentation framework of example 1, but in a different context application.

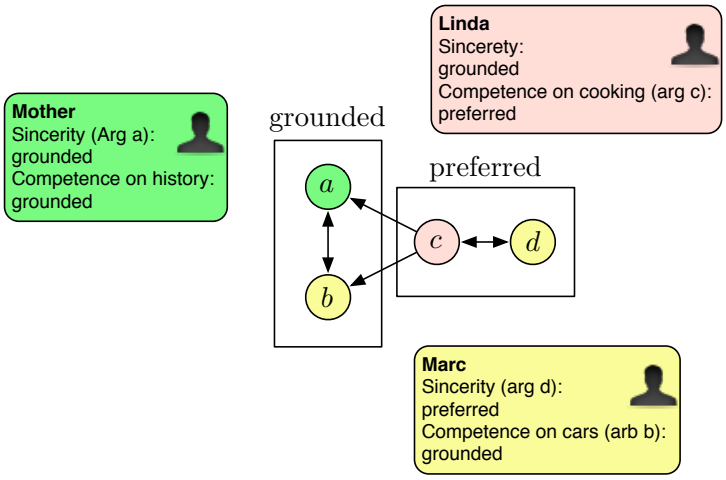


Fig. 3. A multi-sorted argumentation framework with the competence of the sources.

There are three agents, the mother, her daughter (Linda) and her son (Marc). In this simplified example, we assume that all agents agree on whether or not a particular agent is to be trusted. Whether or not an agent is trusted, is reflected in the way their arguments are evaluated. The general idea is that arguments of an untrusted agent are evaluated skeptically (i.e. using the grounded semantics), whereas arguments of agents that are trusted, are evaluated credulously (i.e. using the preferred semantics). Moreover, we distinguish sincerity from competence on a particular topic. In our example, the arguments of the mother are evaluated skeptically by Linda and Marc, both concerning sincerity and concerning her competence on the subject of history. The arguments of Linda are evaluated skeptically with regard to sincerity, and credulously with regard to her competence on the subject of cooking. Finally, the arguments of Marc are evaluated credulously in regard to sincerity, and skeptically in regard to his competence on the subject of cars. Thus, the trustworthiness of the agents influences the acceptability of the arguments they propose.

In the example of figure 3, the arguments a and d concern sincerity, and the arguments b and c concern competence on the subject of cars and cooking, respectively. Argument a is put forward by the mother, b and d by Marc, and c by Linda. Thus we get the following situation: $AF = \langle\{a, b, c, d\}, R\rangle$, with $aRb, bRa, cRa, cRb, cRd, dRc$, and $\mathbb{S} = \langle\{C_1, C_2\}, T\rangle$, where $C_1 = \{a, b\}$, $C_2 = \{c, d\}$, $T(C_1) = gr$ and $T(C_2) = pr$. As demonstrated in example 1, the sorted extensions of this framework are $S_1 = \{c\}$ and $S_2 = \{d\}$. In this example, these extensions can be justified as follows: On the one hand, the cooking competence of Linda makes her argument c accepted in S_1. On the other hand, the sincerity of Marc makes his argument d accepted in S_2. Argument a, of the mother, is evaluated skeptically, and is not accepted in any of the sorted extensions. Note that in this example, arguments c and d are evaluated in the same way, i.e., under the preferred semantics, but further degrees of acceptability may be expressed in order to express degrees of credibility, for instance using fuzzy logic. The exploration of this perspective is left as future work.

6 The Modal Fibring Approach

Multi-sorted argumentation can be expressed as a special case of fibring of modal argumentation frameworks. Following Gabbay and Modgil [5], we represent argumentation subframeworks as possible worlds in a Kripke structure; moreover, we compute sorted extensions as models of the Kripke structure. We can also apply some semantic-based criteria to select desired extensions over the set of possible ones.

Definition 11 (Modal Argumentation Framework) *Let $\mathcal{AF} = \langle A, R\rangle$ be an argumentation framework. A* modal argumentation framework *MAF is a tuple $\langle A, R, MA, MR, S\rangle$ where MA is a set of meta-arguments, $MR \subseteq MA \times MA \cup\ MA \times A$, and $S \in \{cf, ad, co, gr, pr\}$.*

We enrich each subframework with meta-arguments, which represent some properties of the original set of arguments (such as the property of *being attacked*). We will refer to the original set of arguments as *actual arguments*.

We adapt Kripke possible world semantics to the case where possible worlds are argumentation frameworks related by an accessibility relation: argumentation frameworks are possible worlds in a Kripke structure, and modalities are applied to arguments. Thus, $\Diamond\alpha$ in a framework/world w is interpreted as a *possible attack* in the sense that there is a framework/world w' accessible from w, in which α is a justified argument.

Definition 12 (Distributed Argumentation Framework) *A* distributed argumentation framework *DAF is a tuple $\langle W, AR\rangle$ where W is a set of modal argumentation frameworks and $AR \subseteq W \times W$.*

The core idea here is to use modal meta-arguments as pointers between arguments in different worlds, so that we do not to lose information about attacks

between arguments in different subframeworks; modal relations act as consistency constraints between modal meta-argument and actual arguments and can be used to ensure global consistency over the justification statuses of arguments in different worlds.

Definition 13 (Sorting-based DAF) *Let $\mathcal{AF} = \langle A, R\rangle$ and $\mathbb{S} = \langle P, T\rangle$, with $P = \{C_1, \ldots, C_n\}$. For each $i = 1 \ldots n$, let $M_i = \langle C_i, R \downarrow C_i, MA_i, MR_i, T(C_i)\rangle$. A DAF $\langle W, AR\rangle$ is a $\mathbb{S}$-based DAF if and only if the following conditions are satisfied:*

1. *$W = \{M_1, .., M_n\}$*
2. *For any C_i, C_j s.t. $i \neq j$, $\exists a \in C_i, b \in C_j$ s.t. aRb if and only if*
   (a) $\{\diamond a, x_a\} \subseteq MA_j$, and $\{(\diamond a, x_a), (x_a, \diamond a), (\diamond a, b)\} \subseteq MR_j$, and
   (b) $(b, a) \in AR$

In the following example, we demonstrate the computation of sorted extensions using a sorting-based DAF.

Example 4. Let $\mathcal{AF} = \langle A, R\rangle$ and $\mathbb{S} = \{P, T\}$ be the framework and sorting of example 3 (shown in figure 2). The corresponding sorting-based $DAF = \langle W, AR\rangle$, shown in figure 4. We have $W = \{M_1, M_2\}$ with $M_1 = \{A, R, MA_1, MR_1, gr\}$, $MA_1 = \{a, b, x_c, \diamond c\}$ and $RA_1 = \{(a, b), (b, a), (x_c, \diamond c), (\diamond c, x_c)\}$, and similarly for M_2. Note how, in figure 4, attacks across cells are not part of the DAF; the fact that b and c attack each other is conveyed by the modal meta-arguments $\diamond b$ (attacking c) and $\diamond c$ (attacking b).

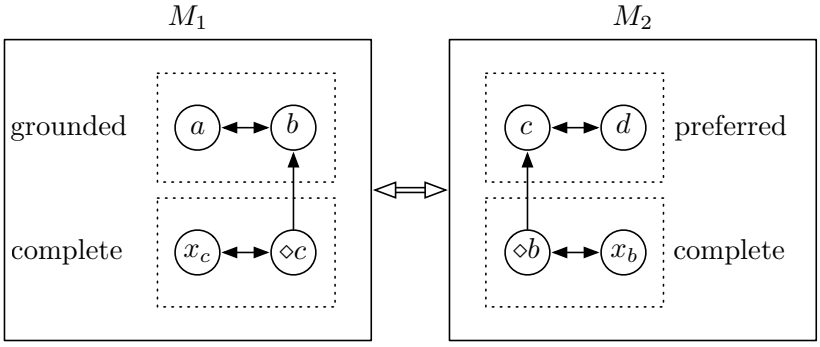


Fig. 4. The sorting-based DAF corresponding to the framework in figure 2.

We now demonstrate the computation of sorted extensions using the sorting-based DAF. Whenever there is a requirement about a subframework to have a specific semantics-based extension we convert in into a requirement of extensions

on the actual arguments in the model world; complete extensions are computed on the meta-arguments for the sake of generality.

Every complete multi-sorted extension on the whole framework is the union over the extensions of the subframeworks.

1. For each $M_i \in \{M_1, M_2\}$ (with $M_i = \langle C_i, R \downarrow C_i, MA_i, MR_i, T(C_i)\rangle$.
   (a) Compute the complete extensions of $\langle MA_i, MR_i\rangle$ (i.e., of the meta-arguments of the subframework/world)
   (b) For each admissible extension, compute the possible extensions of the actual arguments of the subframework/world according to the semantics specified by $T(C_i)$.
2. For each M_i, we thus obtain a set of extensions. Each union of these extensions on the condition that it satisfies the *consistency check*, is a sorted extension of the modal framework. Projecting these extensions over the actual arguments (i.e., removing the meta-arguments) gives an actual sorted extension.

The consistency check of an extension involves checking whether the information conveyed by the modal meta-arguments is consistent with the justification statuses of the actual arguments to which they correspond. For example, let e_1 and e_2 be extensions of M_1 and M_2, obtained as described above. We then have that the union $e_1 \cup e_2$ satisfies the consistency check if and only if,

$$\Diamond\alpha \in e_i \leftrightarrow \alpha \in e_j \wedge \diamond\beta \in e_j \leftrightarrow \beta \in e_i$$

For the framework in this example, we do the following:

- The possible complete extensions of the meta-arguments in the subframework/world M_1 are $\{\emptyset, \{x_c\}, \{\Diamond c\}\}$. Then, starting from each one of them, we compute the grounded extension over the actual arguments of M_1. These extensions are: $\{\emptyset\}$, $\{x_c\}$ and $\{\Diamond c, a\}$.
- Similarly, we compute the complete extensions over the meta-arguments in the subframework/world M_2 and then, for each one of them, the preferred extensions over the actual arguments of M_2: from $\{\emptyset\}$ we get $\{c\}$ and $\{d\}$, from $\{\Diamond b\}$ we get $\{\Diamond b, d\}$ and from $\{x_b\}$ we get $\{x_b, c\}$ and $\{x_b, d\}$.
- We compute the sorted extension by computing the cartesian product over the two sets of extensions (a set of possible extensions for each subframework/world) and then selecting the ones that satisfy the consistency check. So, for instance, the $\{\emptyset\}$ extension from the first world can not be merged with the $\{c\}$ extension from the second world since $\Diamond c$ does not belong to the first one. The resulting set of consistent extensions is consist of $\{d\}$, $\{x_b, d\}$, $\{x_c, d\}$, $\{x_c, x_b, d\}$, $\{\Diamond c, a, c\}$ and $\{\Diamond c, a, x_b, c\}$. Projecting over the actual arguments, we get $\{\{d\}, \{c, a\}\}$.

Another interesting property is that, in uniform cases (every world is associated with the same semantics) we can easily impose a preference criteria over the admissible multi-sorted extensions in order to get the same semantics we'd have

by evaluating the whole framework in that semantics. For instance, in Example 2, the possible multi-sorted extensions we compute by means of modal fibring are $\{\emptyset, \{a\}, \{b\}\}$. Note that we don't allow $\{a, b\}$ to be a multi-sorted extension: this is because we don't have both a and $\Diamond b$ in the possible extensions of the first world nor we have b and $\Diamond a$ in the possible extensions of the second world: basically, by means of modal pointers, we know that those two arguments attack each other and we prevent them from being part of the same multi-sorted extension. We can impose a semantics-based selection criteria: since all worlds are associated with grounded semantics, we may select the minimal extension over the set of multi-sorted extensions $\{\emptyset, \{a\}, \{b\}\}$: this is $\emptyset$, which is exactly the grounded extension of the whole framework. Selecting the maximal sets w.r.t. set inclusion we get $\{\{a\}, \{b\}\}$, which are the preferred extensions of the whole framework.

The modal fibring is just another way to deal with multi-sorted argumentation frameworks. Compared with the main approach discussed in this paper, the main difference is that it does not require an extension S as a parameter, but the framework has to undergo a precise transformation process: adding meta-arguments, removing cross-cell attacks, introducing accessibility relations. This approach also has the interesting property of uniform case extension equivalence. Going towards the field of multi-agent systems, the modal meta-argument could be intended as a call to a remote procedure or a generic communication process with another agent, the owner of the referenced argument whose justification status the caller wants to know.

7 Related Work

In the literature, some formalisms use different kinds of arguments for representing diverse kinds of information, or for distinguishing arguments that are used for distinct purposes. Prakken [6] proposes an argument-based semantics that combines grounded and preferred semantics. The motivation is that reasoning about beliefs should be skeptical, while reasoning about action should be credulous. A dialectical proof theory is defined which is sound and complete with respect to the semantics. We believe that our formalization can serve the same purpose, while being more general. Finally, an interesting aspect of Prakken's proposal is the definition of a dialectical proof theory. This is something we have not considered.

Other related work includes Amgoud & Prade [1], who introduce explanatory, rewards and threats arguments for negotiation dialogues. In practical reasoning, Rotstein et al. [7] propose different types of arguments to represent categorized domain information, like belief, goals or plans.

The idea of multi-sorted argumentation involves the characterization of a group of arguments that have some common features, and these features distinguish them from other groups of arguments. In the mentioned approaches,

argument types and their features are defined for a specific domain, e.g., dialogue systems, practical reasoning. Our approach to multi-sorted argumentation distinguishes these groups of arguments by applying a different semantics to evaluate them, and we provide a general characterization of the notion of multi-sorted argument.

8 Conclusion and Future Work

We have presented a theory of multi-sorted argumentation, that generalizes Dung's theory of abstract argumentation in that it allows different parts of a framework to be evaluated under different semantics. We have proven some basic properties, namely the preservation of conflict-freeness, admissibility and completeness.

We have applied our theory to examples where we reason about trust in a multi-agent context. Different agents are associated with different competence domains in which they are trusted, or considered sincere by other agents. The examples demonstrate that multi-sorted argumentation provides a viable framework to solve some of the central issues that arise in abstract argumentation applied to trust and multi-agent contexts.

The modal fibring approach, building on work by Gabbay and Modgil [5], adds another interesting angle to our theory. The fact that multi-sorted argumentation is expressible in modal argumentation frameworks demonstrates the generality of modal argumentation. We expect that modal argumentation will be a useful framework to investigate more sophisticated forms of multi-sorted argumentation.

There is much work still to be done, on all the aspects described above. First of all, a further generalization is possible if we make some of the assumptions that we made optional. For example, instead of a strict partitioning of the framework, we could allow overlapping subsets. This is natural, because the same argument may be put forward by different agents, each associated with a different semantics for the evaluation of their arguments.

Secondly, we have applied our theory only to some small examples. It will be interesting to apply it to real-world examples, and to compare it with other approaches to multi-agent argumentation and reasoning about trust. Another possible application of our theory, which we have not touched upon, is bounded reasoning in multi-agent systems: dividing a framework into different sets could facilitate a stepwise evaluation of smaller parts of a larger framework. In addition, arguments that are not the focus of a particular issue, could be evaluated using a computationally cheaper semantics. For example, a 'don't care' attitude towards a set of arguments could result in only requiring conflict-freeness for this set.

References

1. Leila Amgoud and Henri Prade. Handling threats, rewards, and explanatory arguments in a unified setting. *Int. J. Intell. Syst.*, 20(12):1195–1218, 2005.

2. P. Baroni, M. Giacomin, and G. Guida. SCC-recursiveness: a general schema for argumentation semantics. *Artificial Intelligence*, 168(1-2):162–210, 2005.
3. Cristiano Castelfranchi and Rino Falcone. *Trust Theory: A Socio-Cognitive and Computational Model*. Wiley, 2010.
4. Phan Minh Dung. 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.
5. Dov Gabbay and Sanjay Modgil. Modal argumentation (draft paper #322a, march 2009).
6. Henry Prakken. Combining sceptical epistemic reasoning with credulous practical reasoning. In Paul E. Dunne and Trevor J. M. Bench-Capon, editors, *COMMA*, volume 144 of *Frontiers in Artificial Intelligence and Applications*, pages 311–322. IOS Press, 2006.
7. Nicolás D. Rotstein, Alejandro Javier García, and Guillermo Ricardo Simari. Reasoning from desires to intentions: A dialectical framework. In *AAAI*, pages 136–141. AAAI Press, 2007.

An Implemented Dialogue System for Inquiry and Persuasion

Luke Riley[1], Katie Atkinson[1], Terry Payne[1] and Elizabeth Black[2]

[1] Department of Computer Science,
University of Liverpool,
L69 3BX, UK
{L.J.Riley,katie,trp@liverpool.ac.uk}
[2] Department of Information and Computer Sciences,
Utrecht University,
80.089, NL
{lizblack@cs.uu.nl}

Abstract. In this paper, we present an implemented system that enables autonomous agents to engage in dialogues that involve inquiries embedded within a process of practical reasoning. The implementation builds upon an existing formal model of value-based argumentation, which has itself been extended to permit a wider range of arguments to be expressed. We present extensions to the formal underlying theory used for the dialogue system, as well as the implementation itself, and the resulting argumentation graph for a given case study. We discuss a number of interesting issues that have arisen from the implementation and the experimental avenues that this test-bed will enable us to pursue.

1 Introduction

Communication through argumentation is one of the key strands of work on computational argumentation. Work on agent-based dialogue systems has been greatly influenced by the dialogue typology of Walton and Krabbe [12]. A number of proposals have been set out for dialogue systems that encompass the main dialogue categories, for example see: [4] for inquiry dialogues; [10] for negotiation; [9] for persuasion; [8] for deliberation. However, very little work has been done on specifying and implementing systems that combine two or more dialogue types. In [3] a formal framework was set out for multi-agent dialogues over actions in which inquiry dialogues over beliefs are combined with persuasion dialogues over actions. The dialogue system allows agents with heterogeneous knowledge to each give input into a decision about how to act to achieve a shared goal. The underlying representation of argument is in terms of a formal version of an argumentation scheme, and critical questions that agents can employ to challenge assertions made by their peers for practical reasoning. Although this dialogue system has been set out in a formal specification [3], it has not previously been validated through an implementation. In this paper we present the details of an implementation of this dialogue system for inquiry and persuasion over action. For a full implementation to be realised it was necessary to extend the formalism presented in [3] to enable a richer set of arguments to be put forward, as we describe. The implemented system we

present provides not only a proof-of-concept in terms of an application of a formal specification, but we also note a number of issues that have been brought to light through the exercise. Furthermore, we consider this implementation to be a starting point for further investigations into agent argumentation dialogues, in particular, with respect to coalition formation.

The paper is structured as follows. In Section 2 we recapitulate from [3] the background material about the dialogue system we have implemented. In Section 3 we present new material that extends the formalism of [3] by providing the full list of critical questions associated with the argumentation scheme that is used in the dialogue system. In Section 4 we describe the implementation and demonstrate it with an example. In Section 5 we discuss issues that have arisen from the implementation, future avenues this work will allow us to explore and we conclude the paper.

2 Background

Our dialogue system allows agents to inquire about beliefs (to find the state of the world) and perform practical reasoning over the next action to perform. To do this, agents in our system may contain epistemic knowledge (beliefs), represented by Garcia and Simari's Defeasible Logic Programming [6], as well as normative knowledge about the effects of actions.

The following definitions provide the formal framework for modeling beliefs.

Definition 1: *A* **defeasible rule** λ *is denoted* $\alpha_1 \wedge \ldots \wedge \alpha_n \rightarrow \alpha_0$ *where* α_i *is a literal for* $0 \leq i \leq n$*. A* **defeasible fact** *is denoted* α *where* α *is a literal. A* **belief** *is either a defeasible rule or a defeasible fact.* $\mathcal{B}$ *denotes the set of all beliefs. We define the following functions* $\mathsf{DefeasibleSection}(\lambda) = \{\alpha_1, \ldots, \alpha_n\}$*;* $\mathsf{DefeasibleProp}(\lambda) = \alpha_0$*.* Each agent has a unique id x taken from a set $\mathcal{I}$ of agent identifiers. Each agent's belief base could be inconsistent.

Definition 2: *A* **belief base** *of an agent* x *is a finite set of beliefs, denoted* Σ^x*.*

The definition of a defeasible derivation is adapted from [6] to work with our assumption that all beliefs are defeasible.

Definition 3: *Let* Ψ *be a set of beliefs and* α *a literal. A* **defeasible derivation** *of* α *from* Ψ*, denoted* $\Psi \mid\sim \alpha$*, is a finite sequence* $\alpha_1, \alpha_2, \ldots, \alpha_n$ *of literals s.t.:* α_n *is* α*; and each literal* α_m *(*$1 \leq m \leq n$*) is in the sequence because either* α_m *is a defeasible fact in* Ψ*, or there exists a defeasible rule* $\beta_1 \wedge \ldots \wedge \beta_j \rightarrow \alpha_m$ *in* Ψ *s.t. every literal* β_i *(*$1 \leq i \leq j$*) is an element* α_k *preceding* α_m *in the sequence (*$k < m$*).*

A b-argument is a minimally consistent set of beliefs from which a claim can be defeasibly derived.

Definition 4: *A* **b-argument** *is denoted* $B = \langle \Phi, \phi \rangle$*. It is constructed from a set of, possibly inconsistent, beliefs* Ψ *where* ϕ *is a defeasible fact and* Φ *is a set of beliefs s.t.: 1)* $\Phi \subseteq \Psi$*; 2)* $\Phi \mid\sim \phi$*; 3)* $\forall \phi, \phi'$ *s.t.* $\Phi \mid\sim \phi$ *and* $\Phi \mid\sim \phi'$*, it is not the case that* $\phi \cup \phi' \vdash \bot$ *(where* $\vdash$ *represents classical implication); and there is no subset of* Φ *satisfying (1-3).* Φ *is called the* **support** *of the b-argument and* ϕ *is called the* **claim***. We define the following functions* $\mathsf{Support}(B) = \Phi$*;* $\mathsf{Claim}(B) = \phi$*;*.

For handling reasoning about the effects of actions, the following argumentation scheme for practical reasoning is used, taken from [1]:

In the current circumstances R, we should perform action A, which will realise goal G, which will result in the new circumstances S, which will promote some value V.

This scheme uses 'values' to describe a social interest an agent has, which will be increased by realising goal G [2]. An agent may propose an action including its justification by instantiating this scheme. Other agents can then challenge instantiations by posing critical questions (CQ) associated with the scheme. 16 critical questions are associated with the above scheme [1] which raise potential issues with: the validity of the elements instantiated in the scheme; the connections between the elements of the scheme; the side effects of actions; and the possible alternatives.

In [3] this scheme is represented as a Value-based Transition System (VATS), a modified version of an Action-Based Transition System (AATS) [13],which has been extended to enable the representation of values.

Definition 5: *The VATS formalism is as follows: A* **VATS** *for an agent x, denoted S^x, is a 9-tuple $\langle Q^x, q_0^x, Ac^x, Av^x, \rho^x, \tau^x, \Phi^x, \pi^x, \delta^x \rangle$ s.t.:*

- Q^x *is a finite set of* states*;*
- $q_0^x \in Q^x$ *is the designated* initial state*;*
- Ac^x *is a finite set of* actions*;*
- Av^x *is a finite set of* values*;*
- $\rho^x : Ac^x \mapsto 2^{Q^x}$ *is an* action precondition function*, which for each action $a \in Ac^x$ defines the set of states $\rho(a)$ from which a may be executed;*
- $\tau^x : Q^x \times Ac^x \mapsto Q^x$ *is a partial* system transition function*, which defines the state $\tau^x(q, a)$ that would result by the performance of a from state q. As this function is partial, not all actions are possible in all states;*
- Φ^x *is a finite set of* atomic propositions*;*
- $\pi^x : Q^x \mapsto 2^{\Phi^x}$ *is an interpretation function, which gives the set of primitive propositions satisfied in each state: if $p \in \pi^x(q)$, then this means that the propositional variable p is satisfied (equivalently, true) in state q; and*
- $\delta^x : Q^x \times Q^x \times Av^x \mapsto \{+, -, =\}$ *is a* valuation function *which defines the* status *(promoted ($+$), demoted ($-$), or neutral ($=$)) of a value $v \in Av^x$ ascribed by the agent to the transition between two states: $\delta^x(q, q', v)$ labels the transition between q and q' with respect to the value $v \in Av^x$.*

Note, $Q^x = \emptyset \leftrightarrow Ac^x = \emptyset \leftrightarrow Av^x = \emptyset \leftrightarrow \Phi^x = \emptyset$.

With its VATS an agent can construct a-arguments for and against actions. The critical questions formalised in Section 3.2 that follow the a-argument formalism are classified as a-arguments also.

Definition 6: *An* **a-argument** *constructed by an agent x from its VATS S^x is a 6-tuple $A = \langle q_x, a, q_y, p, v, s \rangle$ s.t.: $q_x = q_0^x$; $a \in Ac^x$; $\tau^x(q_x, a) = q_y$; $p \in \pi^x(q_y)$; $v \in Av^x$; $\delta^x(q_x, q_y, v) = s$ where $s \in \{+, -, =\}$.*

We define the following functions: $\mathsf{Action}(A) = a$; $\mathsf{Goal}(A) = p$; $\mathsf{Value}(A) = v$; $\mathsf{EndState}(A) = q_y$; $\mathsf{Polarity}(A) = s$.

If $\mathsf{Polarity}(A) = +(-resp.)$*, then we say A is an a-argument for (against resp.) action a to achieve goal p. If* $\mathsf{Polarity}(A)$ *has the value "=", then we say A is an a-argument that is neutral with regards to action a.*

Our framework assumes a closed cooperative multi-agent system. Agents collaborate to find the best way to achieve the dialogue initiator's goal by entering a *persuasion over action* ($pAct$) dialogue where every agent asserts all the a-arguments and critical questions relative to the information currently in the dialogue. However, before these can be asserted, each agent must determine its individual initial state, so that the correct arguments can be found that allow the agents to find consensus over the truth assignment of the known propositions. To do this, each agent participating in the dialogue must first open an inquiry (inq) sub-dialogue (i.e. a dialogue that is embedded within a top-level dialogue) [3] with the other agents; the results of which are p or $\neg p$, $\forall p \in \Phi^x$.

Our dialogue is defined as a dialogue game [7, 11]. Dialogue games typically consist of a set of communicative acts (called moves), a set of rules stating which moves are legal for any point of the dialogue (the protocol), a set of rules defining the effect of making a move and a set of rules that determine when a dialogue terminates. Within our framework, a dialogue denoted $\mathcal{D}_r^t$, is a sequence of moves $m_r, \ldots, m_t$ where $r, \ldots, t \in \mathbb{N}$ represents the time-point at which each move was made with r being the starting point of the dialogue and t the end point. If $r = 1$, then this dialogue is considered a top level dialogue whose type is $pAct$, which is opened through the $pAct$ strategy by the dialogue initiator. If this original dialogue is closed then the dialogue game is over. If $\mathcal{D}_r^t, r \neq 1$ then this is a sub dialogue. The following functions operate over a dialogue[4]:

- $\mathsf{Current}(\mathcal{D}_1^t)$ returns the most recently opened dialogue that has not been closed.
- $\mathsf{Type}(\mathcal{D}_r^t)$ returns the type of dialogue $\mathcal{D}_r^t$ (i.e. $pAct$ or inq).
- $\mathsf{Initiator}(\mathcal{D}_r^t)$ returns the agent who opened dialogue D_r^t.
- $\mathsf{Participants}(\mathcal{D}_r^t)$ returns the set of agents in the dialogue D_r^t.
- $\mathsf{Topic}(\mathcal{D}_r^t)$ returns the goal the agents are trying to achieve iff $\mathsf{Type}(D_r^t) = pAct$.
- $\mathsf{Topic}(\mathcal{D}_r^t)$ returns the set of propositions which the agents are jointly trying to construct b-arguments for iff $\mathsf{Type}(D_r^t) = inq$.
- $\mathsf{Turn}(\mathcal{D}_r^t)$ returns the identifer of the agent whose turn it is.

The moves that the agents can perform are presented in Table 1. Agents take it in turns to perform one move at a time.

Move	**Format**
$open$	$\langle x, open, dialogue(\theta, \gamma, \Lambda)\rangle$
$assert$	$\langle x, assert, \Psi\rangle$
$close$	$\langle x, close, dialogue(\theta, \gamma, \Lambda)\rangle$

Table 1. The format for moves used in this dialogue game, where x represents the agent making the move and either $\theta = pAct$ and γ is a proposition (representing the dialogue goal), or $\theta = inq$ and γ is a set of propositions (that the agent is inquiring over); Λ is a list of agents ($\Lambda = [x_1, \ldots, x_n]$, $\{x_1, \ldots, n_n\} \subseteq \mathcal{I}$); Ψ is either a set of a-arguments or critical questions (if $\theta = pAct$) or Ψ is a set of b-arguments or beliefs (if $\theta = inq$); and x is an agent ($x \in \mathcal{I}$).

All agents' assertions are stored in a publicly readable commitment store (*CS*) that grows monotonically over time, as follows:

[3] Further details of the inquiry sub-dialogue are discussed in Section 3.4.

[4] Further dialogue details are given in [3].

Definition 7: **Commitment store update.**
For a pAct dialogue $(\mathcal{D}_1^t)$ *with participants* $\{x_1, \ldots, x_n\}$*, for all* $x \in \{x_1, \ldots, x_n\}$ *and a commitment store of agent* x *at time-point* t *denoted* CS_x^t*,*

$$CS_x^t = \begin{cases} \emptyset & \textit{iff } t = 0, \\ CS_x^{t-1} \cup \Psi & \textit{iff } m_t = \langle x, assert, \Psi \rangle, \\ CS_x^{t-1} & \textit{otherwise.} \end{cases}$$

Dialogues commence when an event triggers one agent to open a *pAct* dialogue through its *pAct* strategy (see Section 3.5) to identify the action necessary to achieve a given proposition p, where $p = \mathsf{Topic}(D_1^t)$. The other agents that have been included in the open dialogue move then initiate their individual *pAct* strategies, which are guaranteed to find all the relevant arguments related to the dialogue topic, via the use of the *pAct* protocol (Section 3.3) and the *inq* protocol, before terminating.

Once the *pAct* dialogue has terminated, the system evaluates the arguments. This is achieved within our framework using a *Value-Based Argumentation Framework* (VAF) [2], which is an extended version of Dung's abstract Argumentation Framework (AF) [5]. The output of this evaluation is a recommended action (or non-action) that should be performed. An action can only be recommended by the system if it is present in an a-argument or a CQ that also promotes a value. In the event that there is more than one acceptable action the choice is offered to the dialogue initiator.

3 Extending the Formalisation of Critical Questions

The dialogue system set out in [3] handled only three of the possible seventeen critical questions associated with the practical reasoning argumentation scheme. Here we extend the dialogue system by specifying all the necessary critical questions and show their use within the dialogue system. The CQs formalised in Section 3.2 that do not follow Definition 2 are not classified as a-arguments. All CQs can be asserted by any agent to challenge an assertion of any other agent (including itself) and if all agents run the $pAct$ protocol then CQs and a-arguments can only be asserted if there does not exist an assertion in the publicly readable commitment store that includes the same instantiated variables.

3.1 The State Comparison Definition

One particular issue that arose when implementing the dialogue system was the need for a mechanism to clarify how agents with possibly distinct, or partially distinct, sets of beliefs can accurately compare states (since agents' VATS reflect only an individual's representation of the world). As such, we define that two agents, m and n can compare their respective states $q_m \approx q_n$ iff $\pi(q_m) \cap \Phi^n = \pi(q_n) \cap \Phi^m$, else $q_m \not\approx q_n$. The intersection is used to eliminate propositions that reside in only one of the agents beliefs.

When the above approximation holds, the two states q_n and q_m cannot reasonably be said to be different, as both states will agree for each shared proposition. However, these two states may not be identical as the same conclusion can be reached whatever

the values of the distinct propositions. If the comparison does not hold then the states are different due to both agents holding inconsistent values for their shared propositions.

This comparison requires either; each agent to have an internal model of the other agent's beliefs, or an instantiated state in an assertion should make explicit all the propositions that the agent holds to be true or false. Both will allow an agent to access the beliefs of another. This paper chooses the latter option due to the ease of implementation of such a representation. There are no privacy issues by using this method in this framework, as the inquiry dialogue requires each agent to broadcast all their propositions, resulting in beliefs and b-arguments that reside in the publicly readable commitment store. The following shows how the state comparison definition works, when:

$$\Phi^m = \{p, q, r, t\}, \Phi^n = \{p, r, v\}, q_m = [p, \neg q, \neg r, t], q_n = [p, \neg r, v]$$

The state comparison definition:

$$\pi(q_m) \cap \Phi^n = \pi(q_n) \cap \Phi^m$$

The substitution:

$$\{p, t\} \cap \{p, r, v\} = \{p, v\} \cap \{p, q, r, t\}$$
$$\{p\} = \{p\}$$

$Conclusion$: No evidence to suggest the states are different.

3.2 The Additional Critical Questions

We now define arguments that instantiate the remaining critical questions given in [1] that are applicable to our framework (those that are not applicable are discussed subsequent to the presentation of the definitions). Within the formal definitions given below we also give the natural language representation of the questions. In these definitions CQs and a-arguments are collectively referred to as *arguments*. Matched to the definitions are critical question illustrations that visually show a situation where each CQ could be posed. All CQ illustrations usually assume (unless otherwise stated) that both agents have in the initial state $\neg p$ (via the interpretation function), the pAct dialogue topic is to achieve p and agent 1 takes the first turn.

Definition 8: **A cq2-argument** *Answers the question* 'Assuming the circumstances, does the action have the stated consequences?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a, q_y \rangle$ *s.t.* $q_x = q_0^x$; $a \in Ac^x$, $\tau^x(q_x, a) = q_y$. *It challenges an argument* $\langle q'_x, a', q'_y, p', v', s' \rangle$ *or an argument* $\langle q'_{..}, a', q'_{..}, v', s' \rangle$ *iff* $q_x \approx q'_{..}$, $a = a'$, $q_y \not\approx q'_y$.

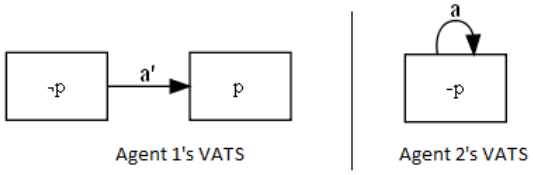


Fig. 1: Illustration of Cq2-argument. Agent 2 will assert a cq-argument when agent 1 asserts an argument to achieve p. These example VATS can also be used to illustrate a cq15-argument (Definition 20). The cq-argument that agent 2 actually chooses depends on which formal conditions are met.

Definition 9: **A cq3-argument** *Answers the question* 'Assuming the circumstances, and the action has the stated consequences, will the action bring about the desired goal?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a, q_y, \neg p\rangle$ *s.t.* $q_x = q_0^x$; $a \in Ac^x$; $\tau^x(q_x, a) = q_y$; $p \notin (q_y)$. *It challenges an argument* $\langle q'_x, a', q'_y, v', s'\rangle$ *iff* $q_x \approx q'_x$, $a = a'$, $q_y \approx q'_y$.

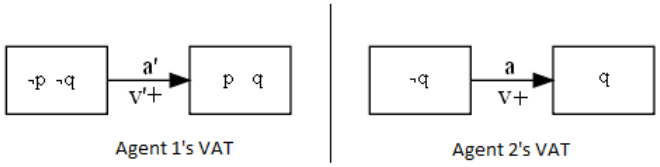


Fig. 2: Illustration of Cq3-argument. Agent 2 will assert a cq3-argument when agent 1 asserts an argument to achieve p. The initial state of Agent 1 is $[\neg p, \neg q]$ and the initial state of Agent 2 is $[\neg q]$

Definition 10: **A cq4-argument** *Answers the question* 'Does the goal realise the value stated?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a, q_y, p, v, \{=, -\}\rangle$ *s.t.* $q_x = q_0^x$; $a \in Ac^x$; $\tau^x(q_x, a) = q_y$; $p \in \pi(q_y)$; $v \in Av^x$; $\delta^x(q_x, q_y, v) \neq +$. *It challenges an argument* $\langle q'_x, a', q'_y, p', v', s'\rangle$ *iff* $q_x \approx q'_x$, $p = p'$, $v = v'$, $s' = +$.

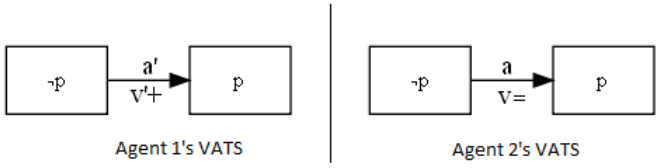


Fig. 3: Illustration of Cq4-argument. Agent 2 will assert a cq4-argument when agent 1 asserts an argument to achieve p

Definition 11: **A cq5-argument** *Answers the question* 'Are there alternative ways of realising the same consequences?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a, q_y\rangle$ *s.t.* $q_x = q_0^x$; $a \in Ac^x$; $\tau^x(q_x, a) = q_y$. *It challenges an argument* $\langle q'_x, a', q'_y, v', s'\rangle$ *or an argument* $\langle q'_x, a', q'_y, p', v', s'\rangle$ *iff* $q_x \approx q'_x$, $a \neq a'$, $q_y \approx q'_y$.

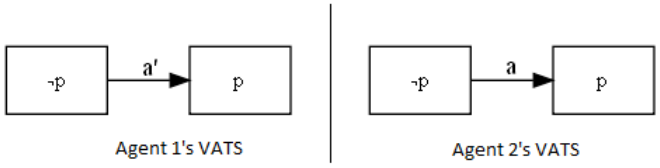


Fig. 4: Illustration of Cq5-argument. Agent 2 will assert a cq5-argument when agent 1 asserts an argument to achieve p

Definition 12: *A* **cq6-argument** *Answers the question* 'Are there alternative ways of realising the same goal?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a, q_y, p, v, +\rangle$ *s.t.* $q_x = q_0^x$; $a \in Ac^x$; $\tau^x(q_x, a) = q_y$; $p \in \pi(q_y)$; $v \in Av^x$; $\delta(q_x, q_y, v) = +$. *It challenges an argument* $\langle q'_x, a', q'_y, p', v', s'\rangle$ *iff* $q_x \approx q'_x$, $a \neq a'$, $p = p'$, $s' = +$.

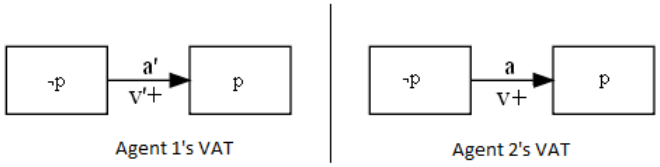


Fig. 5: Illustration of Cq6-argument. Agent 2 will assert a cq-argument when agent 1 asserts an argument to achieve p. These example VATS can also be used to illustrate a cq7-argument (Definition 13). The cq-argument that agent 2 actually chooses depends on which formal conditions are met. The main difference between CQ6 and CQ7 being if the agent knows the new action achieves the goal (CQ6) or not (CQ7).

Definition 13: **A cq7-argument** *Answers the question* 'Are there alternative ways of promoting the same value?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a, q_y, v, + \rangle$ *s.t.* $q_x = q_0^x$; $a \in Ac^x$; $\tau^x(q_x, a) = q_y$; $v \in Av^x$; $\delta(q_x, q_y, v) = +$. *It challenges an argument* $\langle q'_x, a', q'_y, p', v', s' \rangle$ *or an argument* $\langle q'_x, a', q'_y, v', s' \rangle$ *iff* $q_x \approx q'_x$, $a \neq a'$, $v = v'$, $s' = +$.

See Fig. 5 for an illustration of example VATS that could produce a cq7-argument.

Definition 14: **A cq8-argument** *Answers the question* 'Does doing the action have a side effect which demotes the value?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a, q_y, v, - \rangle$ *s.t.* $q_x = q_0^x$; $a \in Ac^x$; $\tau^x(q_x, a) = q_y$; $v \in Av^x$; $\delta^x(q_x, q_y, v) = -$. *It challenges an argument* $\langle q'_x, a', q'_y, p', v', s' \rangle$ *or an argument* $\langle q'_x, a', q'_y, v', s' \rangle$ *iff* $q_x \approx q'_x$, $a = a'$, $v = v'$, $s' = +$.

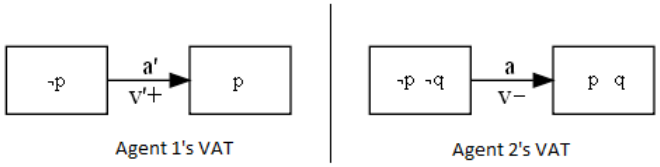


Fig. 6: Illustration of Cq8-argument. Agent 2 will assert a cq-argument when agent 1 asserts an argument to achieve p. The initial state of of agent 2 is $[\neg p, \neg q]$ and the side effect is q. These example VATS can also be used to illustrate a cq9-argument (Definition 15) The cq-argument that agent 2 actually chooses depends on which formal conditions are met.

Definition 15: *A* **cq9-argument** *Answers the question* 'Does doing the action have a side effect which demotes some other value?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a, q_y, v, - \rangle$ *s.t.:* $q_x = q_0^x$; $a \in Ac^x$; $\tau^x(q_x, a) = q_y$; $v \in Av^x$; $\delta^x(q_x, q_y, v) = -$. *It challenges an argument* $\langle q'_x, a', q'_y, p', v', + \rangle$ *or an argument* $\langle q'_x, a', q'_y, v', s' \rangle$ *iff* $q_x \approx q'_x$, $a = a'$, $v \neq v'$, $s' = +$.

See Fig. 6 for an illustration of example VATS that could produce a cq9-argument.

Definition 16: *A* **cq10-argument** *Answers the question* 'Does doing the action have a side effect which promotes some other value?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a, q_y, v, + \rangle$ *s.t.:* $q_x = q_0^x$*;* $a \in Ac^x$*;* $\tau^x(q_x, a) = q_y$*;* $v \in Av^x$*;* $\delta^x(q_x, q_y, v) = +$*. It challenges an argument* $\langle q'_x, a', q'_y, p', v', s' \rangle$ *or an argument* $\langle q'_x, a', q'_y, v', s' \rangle$ *iff* $q_x \approx q'_x$*,* $a = a'$*,* $v \neq v'$*,* $s' = +$*.*

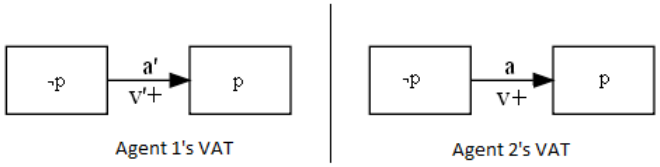


Fig. 7: Illustration of Cq10-argument. Agent 2 will assert a cq-argument when agent 1 asserts an argument to achieve p. These example VATS can also be used to illustrate a cq11-argument (Definition 17). The cq-argument that agent 2 actually chooses depends on which formal conditions are met.

Definition 17: **A cq11-argument** *Answers the question* 'Does doing the action preclude some other action which would promote some other value?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a, q_y, v, + \rangle$ *s.t.* $q_x = q_0^x$*;* $a \in Ac^x$*;* $\tau^x(q_x, a) = q_y$*;* $v \in Av^x$*;* $\delta(q_x, q_y, v) = +$ *. It challenges an argument* $\langle q'_x, a', q'_y, p', v', s' \rangle$ *or an argument* $\langle q'_x, a', q'_y, v', s' \rangle$ *iff* $q_x \approx q'_x$*,* $a \neq a'$*,* $v \neq v'$*,* $s' = +$*.*

See Fig. 7 for an illustration of example VATS that could produce a cq11-argument.

Definition 18: **A cq13-argument** *Answers the question* 'Is the action impossible?'. *It is constructed from a VATS* S^x *and denoted* $\langle a \rangle$ *s.t.* $a \notin Ac^x$*. It challenges any argument that includes* a' *in its definition, iff* $a = a'$*.*

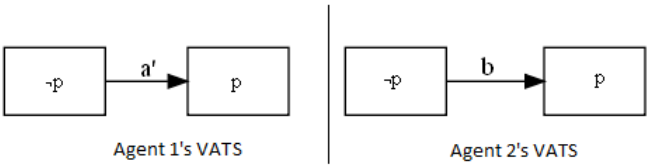


Fig. 8: Illustration of Cq13-argument. Agent 2 will assert a cq13-argument when agent 1 asserts an argument to achieve p

Definition 19: **A cq14-argument** *Answers the question* 'Are the consequences as described possible?'. *It is constructed from a VATS* S^x *and denoted* $\langle q_x, a \rangle$ *s.t.* $q_x = q_0^x$*;* $a \in Ac^x$*;* $\tau^x(q_x, a) \notin Q^x$*. It challenges any argument that includes* q'_x *and* a' *in its definition, iff* $q_x \approx q'_x$*,* $a = a'$*.*

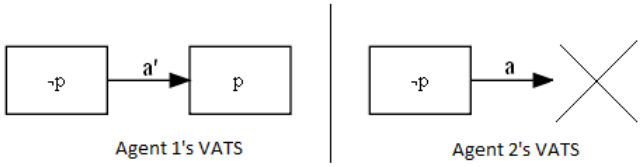


Fig. 9: Illustration of Cq14-argument. Agent 2 will assert a cq14-argument when agent 1 asserts an argument to achieve p

Definition 20: **A cq15-argument** *Answers the question* 'Can the desired goal be realised?'. *It is constructed from a VATS* S^x *and denoted* $\langle \neg p \rangle$ *s.t.* $p \in \Phi^x$. *It challenges an argument* $\langle q'_x, a', q'_y, p', v', s' \rangle$ *iff* $p = p'$, $(\forall q \in Q)(p \notin \pi(q))$.

See Fig. 1 for an illustration of example VATS that could produce a cq15-argument.

Definition 21: **A cq16-argument** *Answers the question* 'Is the value indeed a legitimate value?'. *It is constructed from a VATS* S^x *and denoted* $\langle v, - \rangle$ *s.t.* $v \notin Av^x$. *It challenges an argument* $\langle q'_x, a', q'_y, p', v', s' \rangle$ *or an argument* $\langle q'_x, a', q'_y, v', s' \rangle$ *iff* $v = v'$.

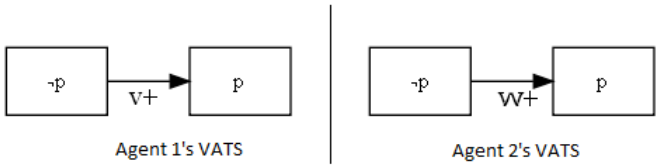


Fig. 10: Illustration of Cq16-Argument. Agent 2 will assert a cq16-argument when agent 1 asserts an argument to achieve p.

Missing from the above list are CQ1 (are the believed circumstances true?), CQ12 (are the circumstances as described possible?) and CQ17 (is the other agent guaranteed to execute its part of the desired joint action?), their omission is explained here.

We assume that cooperative agents all accept the outcome of the inquiry dialogues and hence the representation issues concerning conflicting views of the initial state (as raised by CQ1 and CQ12) should be resolved. Note that under this assumption the outcome of the inquiry will be accepted by all agents even though it may be possible for an agent to construct a relevant counter argument. Resolution of this issue is left for future work. CQ17 is omitted as the system is not currently designed for joint actions.

3.3 Extending the pAct Protocol

To implement the extra critical questions into the system, they need to be included into a protocol so that the agents can use these in a dialogue move. The protocol this implementation uses, named the $pAct$ protocol is an extended version of the one presented in [3]. It returns the set of possible moves that are legal for each agent in the dialogue when the current dialogue is of the type $pAct$. It does not allow an assertion of an a-argument or a CQ if there has already been another asserted that includes all the same instantiated variables.

It takes the top level dialogue from the set of all dialogues $\mathcal{D}$, the identifier of the agent from the set of all identifiers $\mathcal{I}$ and returns the set of legal moves which is an element of the set of all subsets of the set of all moves $\mathcal{M}$. Possible moves for an agent in the $pAct$ dialogue are: an assertion of an a-argument to achieve the dialogue goal; an assertion of a CQ; a move to close the $pAct$ dialogue or a move to open a nested inq dialogue.

3.4 Defining the Inquiry Protocol

An Inquiry Protocol needs to be formally defined as the details were left out of [3]. This protocol returns the set of possible moves that are legal for each agent in the dialogue when the current dialogue is of the type Inq. It takes the top level dialogue from the set of all dialogues $\mathcal{D}$, the identifier of the agent from the set of all identifiers $\mathcal{I}$ and returns the set of legal moves which is an element of the set of all subsets of the set of all moves $\mathcal{M}$.

This protocol will not allow any proposition to become a b-argument without supporting evidence. As this system is not integrated into another, supporting evidence takes the form of defeasible facts or a fully supported defeasible rule. A defeasible rule λ is fully supported when there is a defeasible derivation for the head of the rule that includes the rule and can be constructed from the union of all the commitment stores.

The protocol works by firstly allowing each agent to assert all its relevant beliefs that are not already present in the commitment store ($\Xi_{\mathsf{a}}(D_1^t, x)$). A belief can be either a defeasible rule or a defeasible fact. A defeasible fact is relevant if it is an element of the dialogue topic (Ξ_{a} part (2)(ii,a)) or an element of a defeasible rule in the combined commitment store of all the agents (Ξ_{a} part (2)(ii,b)). The combined commitment store is denoted CSs. A defeasible rule is relevant if its defeasibleProp function returns a belief that is an element of the topic (Ξ_{a} part (3)(ii,a)) or an element of another defeasible rule in the CSs (Ξ_{a} part (3)(ii,b)). Secondly in $\Xi_{\mathsf{b}}(D_1^t, x)$ the agent checks to see if any of their asserted beliefs are fully supported ($\Phi \subseteq CSs$). If they are these beliefs get asserted as b-arguments ($B = \langle \Phi, \phi \rangle$), if they have not been already ($B \notin CSs$). Each agent only asserts a b-argument of a ϕ if it asserted the belief that included ϕ ($\phi \in CS_x^t$). This is to eliminate multiple assertions of b-arguments. Lastly if the agents cannot assert anything new then the only move that will be returned is the close dialogue move.

Definition 22: *The* **Inquiry protocol** *is a function* $\Xi : \mathcal{D} \times \mathcal{I} \mapsto \wp(\mathcal{M})$. *If* D_1^t *is a top-level dialogue s.t.* $\mathsf{Current}(D_1^t) = D_r^t$, $\mathsf{Turn}(D_r^t) = x$, $\mathsf{Participants}(D_r^t) = \Lambda = [x_1, \ldots, x_n]$, $CSs = \bigcup_{\forall x_i \in \{x_1,\ldots,x_n\}} CS_{x_i}^t$, $\mathsf{Type}(D_r^t) = Inq$, $\mathsf{Topic}(D_r^t) = \Phi^{Initiator(D_r^t)}$ *and* $1 \leq t$, *then* $\Xi(D_1^t, x)$ *is*

$$\Xi_{\mathsf{a}}(D_1^t, x) \cup \Xi_{\mathsf{b}}(D_1^t, x) \cup \{\langle x, close, dialogue(Inq, \Phi^{Initiator(D_r^t)}, \Lambda)\rangle\}$$

where

$\Xi_{\mathsf{a}}(D_1^t, x) = \{\langle x, assert, \Phi\rangle |$
(1) $\Phi \neq \emptyset$ *where* Φ *is a set of beliefs, and*
(2) $\forall \phi \in \Phi$ *where* ϕ *is a defeasible fact:*
(i) $\phi \notin CSs, \phi \in \Sigma^x$ *, and*

either (ii,a) $\phi \in \mathsf{Topic}(D_r^t)$,
or (ii,b) $\exists \lambda \in CSs$ *s.t.* $\phi \in \mathsf{DefeasibleSection}(\lambda)$
(3) $\forall \lambda \in \Phi$ *where* λ *is a defeasible rule:*
(i) $\lambda \notin CSs, \lambda \in \Sigma^x$ *, and*
either (ii,a) $\mathsf{DefeasibleProp}(\lambda) \in \mathsf{Topic}(D_r^t)$,
or (ii,b) $\exists \lambda' \in CSs$ *s.t.* $\mathsf{DefeasibleProp}(\lambda) \in \mathsf{DefeasibleSection}(\lambda')$

$\Xi_{\mathsf{b}}(D_1^t, x) = \{\langle x, assert, \Psi\rangle |$
(1) $\Psi \neq \emptyset, \Psi$ *is a set of b arguments , and*
(2) $\forall B \in \Psi$*:* $B = \langle \Phi, \phi \rangle$ *is a b-argument,* $\Phi \subseteq CSs$*,* $\phi \in CS_x^t$ *and* $B \notin CSs$

3.5 pAct Strategy

Agents of this system use a $pAct$ strategy. This strategy either opens a $pAct$ dialogue if the agent is the dialogue initiator or selects one move out of the set of legal moves returned from the correct protocol (the $pAct$ protocol if $\mathsf{Type}(\mathsf{Current}(\mathcal{D}_1^t)) == pAct$, else the $inquiry$ protocol). The strategy is honest as agents do not assert anything they do not believe about their propositions or cannot construct from their VATS.

When using the $pAct$ protocol, agents prefer a move to open an inquiry dialogue to assert moves to close moves. So once they start asserting a-arguments and CQs, they already know the truth value of all their propositions so that the arguments presented are relevant to the actual state of the world. Also as the close move is least preferred then no agent will then attempt to close the dialogue until they have run out of other moves and so the dialogue is exhaustive.

When using the $inquiry$ protocol, agents prefer assertion moves to close moves. Again the dialogue is exhaustive as the close move is the least preferred.

Lastly, both dialogue types will not complete until all agents have made a close move one after another without a different move separating them.

4 Implementation

The implementation uses the *Java Agent DEvelopment Framework (JADE)*[5] to facilitate the storage, modelling and use of the agents' epistemic and normative knowledge at runtime. The user can inspect the initial Value-Based Argumentation Framework (VAF), the preferred extension and the final recommended action. Other elements that can be inspected include the VATS of the agents in the dialogue; further details on both the b-arguments, a-arguments and critical questions and lastly the ability to view the complete resulting dialogue. In addition, the user can modify the value order after dialogue termination, which may result in different preferred extensions and/or recommended actions being generated.

Agents are modelled within a closed environment, and communicate by broadcasting messages to other agents within the dialogue via a shared blackboard. The use of a blackboard to record communicative acts eliminates the need for agents to individually retain the dialogue history. Agents take turns to update the blackboard, to avoid the need for complex coordination strategies.

[5] http://jade.tilab.com/

All attacks in the VAF are finalised after the dialogue has terminated. This is to eliminate the possible effect of dialogue order on the resulting graph. Several issues were identified whilst developing a model to select a final recommended action. Since some critical questions concern *problem formulation* issues [1], not all arguments have an associated value or an associated action (e.g. those derived from CQ13, CQ15, CQ16); instead these arguments have been implemented to automatically defeat any other argument that they attack. In addition, there may be arguments that neither promote or demote a value (e.g. CQ4). As a consequence, these arguments are not of concern when determining the choice of action.

Finally, scenarios can arise whereby b-arguments may claim logical contradictions (e.g. p and $\neg p$). To resolve this, the current implementation assumes that the assertion holds (i.e. p) and the contradiction is ignored. This conflict strategy was picked because of time constraints. In the future a better policy will be considered.

4.1 Implementation example

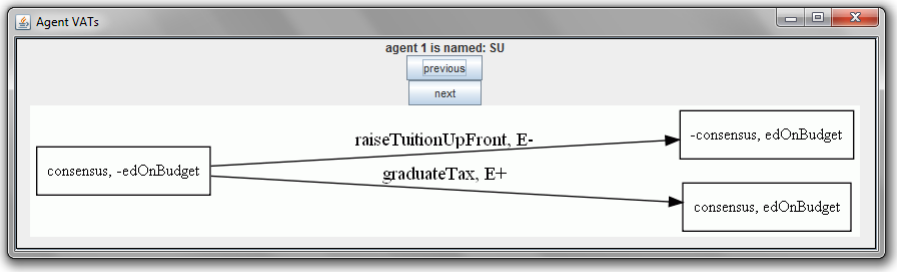


Fig. 11: VATS for SU, S^{SU}

We have thus far evaluated our system through the use of examples scenarios. We briefly consider here an example debate concerning proposals to raise university tuition fees. A VATS produced by this system can be seen in Figure 11. Arguments this agent SU can produce given the transitions in it VATS include, among others:

Where there is majority consensus and the education budget is exceeded, implement a graduate tax, which will retain consensus and stick to budget and promote equality for students.

Also, a CQ9 argument (identifying an alternative action that demotes the value) could be generated:

Where there is majority consensus and the education budget is exceeded, raise tuition fees upfront, which will violate consensus and stick to budget but demotes equality for students.

Of course, further arguments could be posed against these examples, depending on the VATS of the other agents in the dialogue.

An example of full dialogue output in terms of a VAF can be seen in Figure 12. Each node represents an argument. The edges show which argument(s) they are attacking and the edge caption shows which critical question attack these edges represent. Nodes that are not attacking another are just a-arguments. A VAF is evaluated to enable the system to determine the winning argument(s) once supplied with a value ordering. It uses this value ordering to compute the preferred extension which is then filtered to removed all the arguments that do not include an action with a promoted value. The final results are then offered to the user.

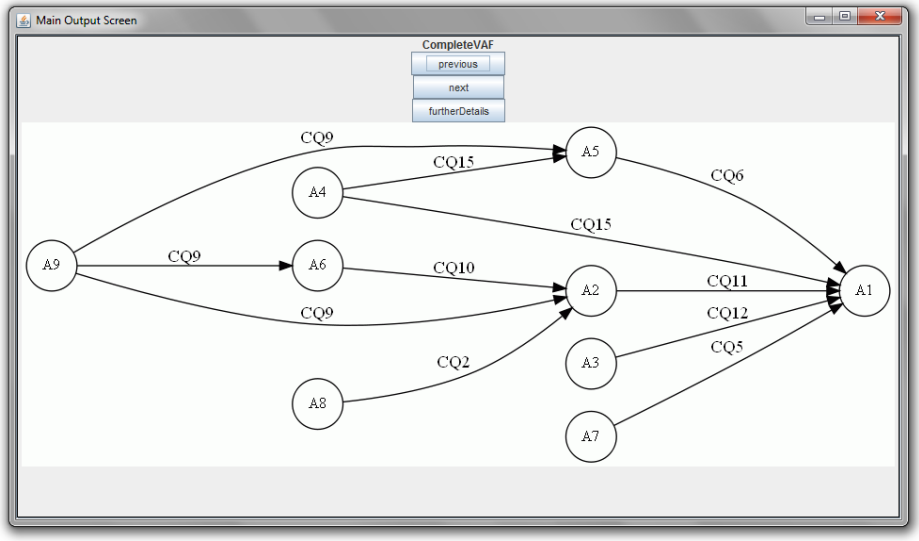


Fig. 12: VAF for the example dialogue

5 Discussion and Concluding Remarks

In this paper we have taken a formal theory of argumentation, extended it, and implemented it to produce a working agent dialogue system for conducting inquiry and persuasion over action dialogues. Our contribution is both in terms of the extended formalism and the implemented system itself, which provides grounds for future work.

The inclusion of additional critical questions enables agents to better identify ambiguities within their shared models, and thus construct additional arguments to find some consensus. However, the existence of these additional critical questions could also undermine the ability of agents forming some consensus. This is in part due to the fact that in some contexts, all existing arguments could be defeated by a carefully selected question which, when posed, may result in no recommended action. Thus, it may be possible that the efforts of other agents to arrive at consensus may be undermined by

an agent that possesses flawed normative and/or epistemic knowledge. Implementing the stages of practical reasoning from [1] may eliminate this problem, as the first stage (*problem formulation*) would resolve representation issues, the second stage (*epistemic reasoning*) would be represented by the inquiry dialogue and the third stage (*action selection*) would be represented by the persuasion over action dialogue.

An interesting future extension to this work would be to see how this system could be modified to allow for each agent to have its own preference order, instead of the implemented one global preference order model. This modification could lead to a potentially uncooperative multi-agent system, which could be further explored by introducing aspects of coalition formation.

Acknowledgements

Luke Riley is grateful for support from the EPSRC. Katie Atkinson was partially supported by the FP7-ICT-2009-4 Programme, IMPACT Project, Grant Agreement No. 247228. Elizabeth Black is funded by the European Union Seventh Framework Programme (FP7/2007-2011), grant agreement 253911. The views expressed are those of the authors and are not necessarily representative of their project or funding body. We also thank the anonymous referees for their constructive comments.

References

1. K. Atkinson and T. J. M. Bench-Capon. Practical reasoning as presumptive argumentation using action based alternating transition systems. *Artificial Intelligence*, 171(10–15):855–874, 2007.
2. T. J. M. Bench-Capon. Persuasion in practical argument using value based argumentation frameworks. *J. of Logic and Computation*, 13(3):429–48, 2003.
3. E. Black and K. Atkinson. Dialogues that account for different perspectives in collaborative argumentation. In *Proceedings of the Eighth International Conference on Autonomous Agents and Multi-Agent Systems*, pages 867–874, 2009.
4. E. Black and A. Hunter. An inquiry dialogue system. *JAAMAS*, 19(2):173–209, 2009.
5. P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77:321–357, 1995.
6. A. J. García and G. R. Simari. Defeasible logic programming an argumentative approach. *Theory and Practice of Logic Prog.*, 4(1–2):95–138, 2004.
7. P. McBurney and S. Parsons. Games that agents play: A formal framework for dialogues between autonomous agents. *J. of Logic, Language and Information*, 11(3):315–334, 2002.
8. D. H. P. McBurney and S. Parsons. The eightfold way of deliberation dialogue. *International Journal of Intelligent Systems*, 22(1):95–132, 2007.
9. H. Prakken. Formal systems for persuasion dialogue. *KER*, 21(2):163–188, 2006.
10. I. Rahwan, S. D. Ramchurn, N. R. Jennings, P. McBurney, S. Parsons, and L. Sonenberg. Argumentation-based negotiation. *KER*, 18(4):343–375, 2003.
11. C. Reed. Dialogue frames in agent communications. In *3rd Int. Conf. on Multi-Agent Systems*, pages 246–253, 1998.
12. D. Walton and E. Krabbe. *Commitment in dialogue: Basic concepts of interpersonal reasoning*. State University of New York Press, Albany NY, 1995.
13. M. Wooldridge and W. van der Hoek. On obligations and normative ability: Towards a logical analysis of the social contract. *J. of Applied Logic*, 3:396–420, 2005.

Resource Boundedness and Argumentation

Nicolás D. Rotstein, Nir Oren, and Timothy J. Norman

dot.rural Digital Economy Hub,
University of Aberdeen, United Kingdom
e-mail: {nico.rotstein, n.oren, t.j.norman}@abdn.ac.uk

Abstract. In this paper we extend the traditional Dung framework for argumentation with cardinality constraints over the set of warranted arguments. This results in a new definition for argumentation semantics wherein arguments within an extension are both in some sense consistent and compliant with the constraints imposed on the system. After discussing the theoretical aspects of such a *resource-bounded argumentation framework* we describe its utility via an application to a concrete application domain: the scheduling of demand responsive transport.

1 Introduction

A common subthread of recent work in argumentation has concentrated on applying abstract argumentation semantics to problems from other domains. Examples of such work include addressing normative conflict [12], trust [11], practical reasoning [3], amongst others. In this paper, we follow this tradition, and apply ideas from argumentation theory to the domain of demand responsive transport (DRT) [2]. This domain (described in more detail in Section 4) can be seen as an instantiation of a more general scheduling problem. Here, a set of passengers with certain requirements must be allocated to a set of vehicles, with each vehicle able to contain only a fixed number of passengers and traverse only a certain route during a single period of time. Now the semantics for abstract argument frameworks might allow us to identify some alternatives, but such semantics do not take the *resource bounds* (such as a vehicle being able to hold only a certain number of passengers) of the domain into account. This limitation suggests the possibility of enhancing Dung's argumentation framework [9] to cater for such resource bounds, and this paper investigates this process.

We modify Dung's seminal argument framework (AF) with the addition of constraints over different properties of arguments. For example, as we discuss in more depth in Section 4, we can associate a possible trip for a passenger with an argument. Now, if different passengers have different weights, and different vehicles have different maximum loads, we can constrain the legal combinations of trips based on the total weights the vehicles would have to carry. This constraint depends on a relatively complex attribute associated with an argument, which could be interpreted as consumption of resources. Taking this into account we then place some sort of limit (*e.g.*, less than, equal to or greater than) over the total number of arguments that can appear within an extension.

The remainder of this paper is structured as follows: we begin with Section 2 explaining *why* traditional AFs are ill-suited for representing resource bounds, following which in Section 3 we formalise the notion of a *resource-bounded argumentation framework* (RAF), and then describe two possible semantics of such argumentation frameworks. Section 4 then examines our application domain in more detail, and finally we discuss related work and future extensions to our current approach in Section 5, before concluding in Section 6.

2 Resource Boundedness and Traditional Conflicts

Constraints and resource bounds add a new source of conflict to arguments which is not adequately captured by standard Dung argumentation frameworks. In order to illustrate this assertion, let us assume the existence of two undefeated arguments a_1, a_2 within some argument framework. Since both are undefeated, most semantics would have them warranted. Now consider the situation where each of these arguments represent the use of some resource, and only a single instance of this resource is available. This additional constraint does not make a_1 or a_2 any less acceptable but simply states that both arguments cannot be warranted together. The simplest representation for this situation is a mutual defeat. However, not every RBness[1] situation can be modelled with such simplicity.

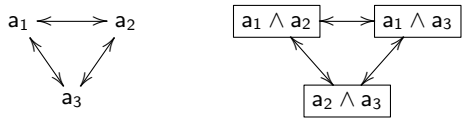
Now consider the case of three undefeated arguments a_1, a_2, a_3 requiring resource r in order to be applicable. Again, any semantics would have these three arguments warranted. Assume that there are 2 available instances (or *tokens*) of r, and that the RB on r requires maximising its consumption; hence, exactly two of these arguments should be warranted together. One possible approach to representing this scenario utilising the previous idea is to set a mutual defeat between every pair of arguments, as shown in Figure 1(a). In this case, the preferred semantics would yield the following set of extensions: $\{\{\mathsf{a}_1\}, \{\mathsf{a}_2\}, \{\mathsf{a}_3\}\}$. Intuitively, the desired set of extensions is $\{\{\mathsf{a}_1, \mathsf{a}_2\}, \{\mathsf{a}_2, \mathsf{a}_3\}, \{\mathsf{a}_1, \mathsf{a}_3\}\}$, *i.e.*, every combination of three arguments with no repetition, taken two at a time. This indicates that there are two problems that must be solved: devising a framework yielding the desired set of extensions, and dealing with the combinatorial explosion brought about by the construction of this set. This combinatorial explosion should be avoided both in the representation, as well as in the computation of warranted arguments.

A potential solution for the representation problem generates a combinatorial problem: the creation of *clusters of arguments* attacking one another, as depicted in Figure 1(b). For n arguments competing for m ($m < n$) tokens of the same resource, the amount of clusters is $\frac{n!}{m!(n-m)!}$, which grows factorially. This is applicable to every set of arguments in the framework that is involved in RBness.

3 The Resource-Bounded Argumentation Framework

There is a need to add a new element to the argumentation framework to best represent the kind of conflicts brought by RBs, *i.e.*, conflicts at the level of extensions. RBs

[1] "RBness" will be used as the short form for "resource boundedness".



(a) (b)

Fig. 1. *(a) Three arguments, two tokens (b) Clusters of arguments*

indicate that certain arguments, competing for a certain resource, cannot be warranted together. This *resource boundedness relation* will be a new element in the new framework we are proposing.

Definition 1 (Resource-Bounded Argumentation Framework (RAF))
A RAF is a tuple $\langle \mathsf{A}, \mathsf{D}, \mathsf{R} \rangle$*, where* A *is a set of arguments,* $\mathsf{D} \subseteq \mathsf{A} \times \mathsf{A}$ *is the defeat relation over arguments*[2]*, and* $\mathsf{R} \subseteq 2^{\mathsf{A}} \times \mathfrak{f}$ *is the resource boundedness relation over arguments, where* $\mathfrak{f} : 2^{\mathsf{A}} \rightarrow \{true, false\}$ *is a boolean function.*

We refer to elements of R*, of the form* $(\rho, \mathfrak{f})$ *as* resource bounds*, abbreviated as RB. Here,* ρ *is a set of arguments, and* $\mathfrak{f}$ *is the boolean function found in the resource boundedness relation.*

The previous example concerning three arguments competing for two tokens of the same resource can be now represented by the RAF $\langle \{\mathsf{a}_1, \mathsf{a}_2, \mathsf{a}_3\}, \{\}, \{(\{\mathsf{a}_1, \mathsf{a}_2, \mathsf{a}_3\}, \Sigma = 2)\}\rangle$, as illustrated in Figure 2, where the sum symbol is a shortcut for the constraint over the amount of tokens allowed. Hence, $\mathfrak{f}(\{\mathsf{a}_1, \mathsf{a}_2, \mathsf{a}_3\}) \equiv \sum(\mathsf{a}_1, \mathsf{a}_2, \mathsf{a}_3) = 2 \equiv \mathit{false}$ and the RB needs to be taken into account.

In order to simplify our presentation, in the remainder of the paper we will only consider one type of resource bound, which operates over the summation of the number of arguments appearing in the extension and referred to by the resource bound. While we refer only to this type of RB, our results are intended to be applicable to any type of resource bound of the form described in the definition above.

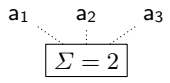


Fig. 2. *Three arguments and a resource bound*

Definition 2 (RB Compliance) *A set of arguments* E ***complies*** *(or is* ***compliant****) with an RB* $(\rho, \mathfrak{f})$ *iff* $\mathfrak{f}(E \cap \rho) = true$*. Given a set of RBs* R*, if* E *complies with every RB in* R*, we say that* E ***complies*** *(or is* ***compliant****) with* R*.*

[2] An argument a defeating an argument b will be written as $\mathsf{a} \rightarrow \mathsf{b}$.

The combinatorial explosion over the representation is now solved, as there is no need to explicitly state all the sets of arguments compliant with all RBs, as illustrated in Figure 1(b). However, the semantic problem of how to compute extensions compliant with all RBs still remains. The rest of this section is focused on this, providing two different approaches.

Now RBs may not apply to all arguments found in a RAF. Arguments unaffected by RBs are referred to as *unbounded arguments*.

Definition 3 (Unbounded Argument) *An argument* a *is* ***unbounded*** *wrt. a set of RBs* R *iff* $\mathsf{a} \notin \rho$*, for any* $(\rho, \cdot) \in \mathsf{R}$.

Before introducing the notion of a *resource-bounded extension* we need the definition for an *admissible set of arguments*. In this article, we restrict the study to the admissibility based semantics, and concentrate on the preferred semantics.

Definition 4 (Acceptable Argument & Admissible Set of Args. [9])
Given two arguments a, b *in a set of arguments* A *and a subset* $S \subseteq \mathsf{A}$*:*

1. a *is attacked by* S *iff there is an argument in* S *attacking* a*;*
2. a *is* ***acceptable*** *with respect to* S *iff for each* b*: if* b *attacks* a *then* b *is attacked by* S.
3. *A conflict-free set of arguments* S *is* ***admissible*** *iff each argument in* S *is acceptable with respect to* S.

Definition 5 (AF Extension) *Given a RAF* $F = \langle \mathsf{A}, \mathsf{D}, \mathsf{R} \rangle$ *and an argumentation semantics* S*, a set of arguments* $E \subseteq \mathsf{A}$ *is an* ***AF extension*** *of* F *following* S *iff* E *is an extension of the associated AF* $\langle \mathsf{A}, \mathsf{D} \rangle$ *following* S.

3.1 An Approach for RB-compliant Extensions: Partitioning

Once the set of extensions of a framework has been computed, a straightforward solution for making extensions compliant with RBs is to partition them. Next, we define how such partitioning should be performed, and what the set of extensions looks like.

Definition 6 (RB Extension by Partitioning) *Let* $F = \langle \mathsf{A}, \mathsf{D}, \mathsf{R} \rangle$ *be a RAF and* E_{af}*, an AF extension of* F *following argumentation semantics* S*. An* ***RB extension*** $E_{rb} \subseteq E_{af}$ *for* F *following* S *is a set of arguments such that all of the following hold:*

1. E_{rb} *complies with* R
2. E_{rb} *is admissible*
3. *if* $\mathsf{a} \in E_{af}$ *and* a *is unbounded wrt.* R*, then* $\mathsf{a} \in E_{rb}$

An RB extension denotes a set of arguments that is warranted in terms of the chosen semantics while being compliant with all RBs. Note that there is no maximality requirement for these extensions, as we only rely on compliance with RBs. Sometimes, a set of arguments qualifying as an RB extension could have several subsets that are also RB extensions, which differs with traditional argumentation semantics. However, as stated by condition (3), every unbounded argument in an AF extension E_{af} must belong to an

RB extension E_{rb}, and $E_{rb} \subseteq E_{af}$. This condition ensures some sort of partial maximality, *i.e.*, maximality only for those arguments that are not actually constrained by any resource limitation.

Consider the RAF depicted in Figure 2, changing the RB to $\Sigma \leq 2$. In this case, under the preferred semantics, every set of two arguments would be an RB extension, but also every singleton set, and even the empty set. All of these sets are admissible and compliant with the RB. Adding an unbounded argument a_4 to the example would make a_4 appear in all the RB extensions. Choosing one of these extensions is, as always, dependent on the application domain.

Returning to traditional AFs, an informal interpretation of their semantics is that an extension describes a maximal set of arguments that can be believed in despite having information in opposition. Hence, sometimes we will be interested in *maximal RB extensions*, with no regard to their subsets. Again, this is totally dependent on the application domain. Another question worth asking is whether there is a relation between the number of AF extensions and the number of RB extensions of a given RAF. The answer is that there is not. For instance, the RAF $\langle\{\mathsf{a}\}, \{\}, \{(\{\mathsf{a}\}, \Sigma \leq 1)\}\rangle$ has only one AF extension: $\{\mathsf{a}\}$, but two RB extensions: $\{\{\}, \mathsf{a}\}$. On the other hand, if we consider the RAF $\langle\{\mathsf{a}, \mathsf{b}\}, \{\mathsf{a} \rightarrow \mathsf{b}\}, \{(\{\mathsf{a}, \mathsf{b}\}, \Sigma = 2)\}\rangle$, it has one AF extension ($\{\mathsf{a}\}$) but no RB extensions. Furthermore, even when considering single-extension semantics (such as the grounded semantics) the partitioning approach could yield multiple extensions.

Now it should be clear that there are some situations where no RB extension is compliant with some resource bound. We refer to such a case as a resource bound truncated RAF.

Definition 7 (RB Truncated RAF) *Given a set of AF extensions X for a RAF $F = \langle \mathsf{A}, \mathsf{D}, \mathsf{R}\rangle$ following argumentation semantics S and an RB $\rho \in \mathsf{R}$, F is **RB truncated** iff there is no $E' \subseteq E, E \in X$ such that E' complies with ρ.*

In other words, a RAF is RB truncated iff it has an empty set of RB extensions and a non-empty set of AF extensions. A simple example of RB truncation is a RAF with an argument a that is a member of all extensions and is the only one associated with an RB wherein $\mathsf{f} = \Sigma \geq 2$ Such a RAF is truncated since the RB cannot be complied with. Truncated RAFs yield no extensions, and can thus not be solved. An external mechanism could be defined to discover the source of truncation, and even be used as a tool to understand what constraints are preventing the framework from yielding extensions. Dealing with truncated RAFs in this way is out of the scope of the current paper.

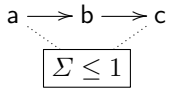
While RB truncated RAFs may occur, we can also guarantee the existence of RB extensions in some cases, as formalised by the following proposition:

Proposition 1. *Let $F = \langle \mathsf{A}, \mathsf{D}, \mathsf{R}\rangle$ be a RAF and S, an argumentation semantics, it holds that:*

1. *E_{af} is both an AF extension and an RB extension for F following S iff E_{af} complies with R;*
2. *the set of AF extensions and RB extensions for F following S coincide iff every AF extension for F following S complies with every R.*

Due to space constraints, we do not provide algorithms for computing RB extensions in this paper. However, the following examples give some indication of this process. Here, and in the rest of the paper, examples will utilise the preferred semantics, unless stated otherwise.

Example 1 *Consider the following* RAF $F_1 = \langle \mathsf{A}_1, \mathsf{D}_1, \mathsf{R}_1 \rangle$*, where* $\mathsf{A}_1 = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$, $\mathsf{D}_1 = \{\mathsf{a} \to \mathsf{b}, \mathsf{b} \to \mathsf{c}\}$*, and* $\mathsf{R}_1 = \{(\{\mathsf{a}, \mathsf{c}\}, \Sigma \leq 1)\}$.



The only AF extension of F *is* $\{\mathsf{a}, \mathsf{c}\}$*, which is not compliant with the RB* $\Sigma \leq 1$*. The subsets of this extension compliant with Definition 6 are the empty set and* $\{\mathsf{a}\}$*, since* $\{\mathsf{c}\}$ *is not being defended by its own set (*i.e.*, is inadmissible) and thus does not meet condition (2).*

Example 2 *Let* $\langle \mathsf{A}_2, \mathsf{D}_2, \mathsf{R}_2 \rangle$ *be a* RAF*, where* $\mathsf{A}_2 = \{\mathsf{a}, \mathsf{w}, \mathsf{x}, \mathsf{y}, \mathsf{z}\}$, $\mathsf{D}_2 = \{\mathsf{x} \to \mathsf{y}, \mathsf{y} \to \mathsf{x}, \mathsf{z} \to \mathsf{w}\}$*, and* $\mathsf{R}_2 = \{(\{\mathsf{a}, \mathsf{z}\}, \Sigma \leq 1), (\{\mathsf{a}, \mathsf{x}, \mathsf{z}\}, \Sigma \leq 2)\}$*, representing that* a *and* z *require a resource whose only token has to be consumed, and* a, x, z *require a resource with 2 tokens available. Figure 3 shows* F_2 *along with the set of preferred AF extensions* $\{\{\mathsf{a}, \mathsf{x}, \mathsf{z}\}, \{\mathsf{a}, \mathsf{y}, \mathsf{z}\}\}$.

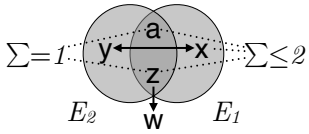


Fig. 3. *AF extensions and resource bounds in* F_2

Extension $E_1 = \{\mathsf{a}, \mathsf{x}, \mathsf{z}\}$ *cannot be taken as a whole due to RB* $\Sigma \leq 2$*. Therefore, the seven RB-compliant subsets are:* $E_{11} = \{\mathsf{a}, \mathsf{x}\}; E_{12} = \{\mathsf{a}, \mathsf{z}\}; E_{13} = \{\mathsf{x}, \mathsf{z}\}; E_{14} = \{\mathsf{a}\}, E_{15} = \{\mathsf{x}\}, E_{16} = \{\mathsf{z}\}, E_{17} = \emptyset$*. Note that some of these subsets are also tied to RB* $\Sigma = 1$*. Those that do not contain* a *and/or* z *will not be RB extensions, namely,* E_{12}*,* E_{15} *and* E_{17}*.*

Regarding extension $E_2 = \{\mathsf{a}, \mathsf{y}, \mathsf{z}\}$*, the subsets complying with RB* $\Sigma = 1$ *are* $E_{21} = \{\mathsf{a}, \mathsf{y}\}; E_{22} = \{\mathsf{z}, \mathsf{y}\}$*. Both comply with RB* $\Sigma \leq 2$*. Finally, the set of RB extensions is* $\{\{\mathsf{a}\}, \{\mathsf{z}\}, \{\mathsf{a}, \mathsf{x}\}, \{\mathsf{x}, \mathsf{z}\}, \{\mathsf{a}, \mathsf{y}\}, \{\mathsf{z}, \mathsf{y}\}\}$*. Maximal RB extensions are shown in Figure 4.*

The process of computing RB extensions can be seen to resemble a *meta-argumentation* process [5,7], up to one level. The difference between our approach and those based on meta-argumentation is the nature of these *meta-conflicts*. In the RAF, these are not traditional pairwise conflicts, but more complex structures relating sets of potentially warranted arguments to the resources they require and their availability.

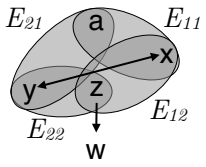


Fig. 4. *Maximal RB extensions considering resource bounds in* F_2

3.2 An Approach for RB-compliant Extensions: Modifying the RAF

There is an alternative for obtaining RB-compliant extensions. The previous approach partitions an extension whenever it encounters RBs that are not complied with by it. Once all RBs are complied with by every extension, the process stops, and an appropriate criterion should be used to choose which extension is taken. In this section we introduce a different method, which removes enough arguments in every extension in order to make them compliant with all RBs, and we will show that this method can result in different extensions than the approach described in Definition 6.

Now one question that arises is whether this method is intuitively correct. More specifically, is it appropriate to remove arguments from an extension that have been deemed as warranted by the argumentation semantics? In order to answer this question, consider a framework with only one extension; every argument in that extension would be sceptically warranted. Now assume a that a subset of these warranted arguments is not compliant with a certain RB, and that this situation could be resolved by removing some of the (previously) warranted arguments. The absence of those arguments could lead to other arguments becoming warranted. These newly warranted arguments could be seen to be "sub-optimally warranted". In some domains, accepting such arguments allows one to obtain a solution where no solution would have otherwise been computable, and we thus claim that there is a dependency on the application domain regarding whether taking these options instead of the (discarded) best ones makes sense.

Example 3 *Consider the RAF in Example 2, depicted in Figure 3. At least one argument has to be removed in order for the framework to comply with* R_2*. Note that deleting* z *allows both RBs to be complied with. Now, since* z *no longer exists, argument* w *becomes undefeated, thus it is included in both extensions. This is shown in Figure 5.*

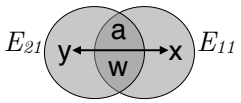


Fig. 5. *Framework* F_2 *after an RB-driven deletion*

If we compare extension E_{21} from Example 2 with extension E_{22} from Example 3, the difference is the inclusion of argument w in E_{22}. This occurred due to the modification of the framework caused by withdrawing x, the sole defeater of w.

In most domains, it makes little sense to withdraw arguments at random. Instead, some *selection criteria*, often based on some preferences, can be used to identify which arguments should be withdrawn.

Definition 8 (Selection Criterion) *Given a set of arguments A, a **selection criterion** $\succ \subset A \times A$ determines a total order over A. Argument a being **preferred** to argument b is denoted as $\mathsf{a} \succ \mathsf{b}$.*

Now given a resource bound that needs to be satisfied, these selection criteria can guide us as to which arguments must be deleted in order to comply with the RB. We encapsulate this concept within the definition of a RB deletion, as follows.

Definition 9 (RB Deletion) *Let $F = \langle \mathsf{A}, \mathsf{D}, \mathsf{R} \rangle$ be a RAF and $\succ$, a selection criterion over A. An **RB deletion** for a set of arguments γ wrt. an RB $(\rho, \mathfrak{f}) \in \mathsf{R}$ is a set of arguments $\delta \subseteq \rho$ such that both of the following hold:*

1. $\mathfrak{f}(\rho \cap \gamma \setminus \delta) = true$.
2. *for every $\mathsf{e} \in \delta$ and every $\mathsf{f} \in (\rho \setminus \delta)$ it holds that $\mathsf{f} \succ \mathsf{e}$.*

The first condition requires that the RB be complied with by the set of arguments when not considering the RB deletion. The second condition specifies that those arguments that are to be left out are the least preferred wrt. the selection criterion. Note that unbounded arguments are never included into an RB deletion, as they are subsets of the RB set.

Since the selection criterion has to make a decision for every pair of arguments, an RB deletion will leave out the least preferred arguments in the set in order to satisfy some RB. However, there could be many RB deletions associated with each RB, and therefore a choice has to be made. For instance, consider the RB deletion for set $\{\mathsf{a}, \mathsf{b}\}$ wrt. RB $(\{\mathsf{a}, \mathsf{b}\}, \Sigma \leq 1)$, where $\mathsf{a} \succ \mathsf{b}$. The two valid RB deletions are $\{\mathsf{b}\}$ and $\{\mathsf{a}, \mathsf{b}\}$, as both satisfy the RB and when choosing which argument to leave out, we drop the least preferred, *i.e.*, b.

We are now in a position to define how an RB extension can be computed through RB deletion. Informally, given an extension of the underlying argument framework with respect to some semantics, and a subset of R which this extension does not comply with, we identify those RB deletions that, when applied to the original argument framework, will allow all RBs to be complied with. We then compute the RB extension by creating a new argument framework which does not contain those arguments found in the RB deletions, and computing the extensions of this new argument framework. This is formalised by the following definition.

Definition 10 (RB Extension by Deletions) *Let $F = \langle \mathsf{A}, \mathsf{D}, \mathsf{R} \rangle$ be a RAF. An **RBD extension** E_{rbd} for F following argumentation semantics S is an extension of the AF $\langle \mathsf{A}', \mathsf{D}' \rangle$ following S, where:*

1. $\mathsf{A}' = \mathsf{A} \setminus \{\delta \mid$ *where* δ *is an RB deletion for an AF extension for* F *following* S *wrt. an RB in* $\mathsf{R}\}$*;*
2. $\mathsf{D}' = \{\mathsf{a} \rightarrow \mathsf{b} \in \mathsf{D} \mid \mathsf{a} \in \mathsf{A}', \mathsf{b} \in \mathsf{A}'\}$*;*
3. E_{rbd} *complies with* R*.*

The process of computing RBD extensions can be seen as operating in two steps. First, given a RB unaware solution (the AF extension), arguments that are non-compliant with the resource bounds are removed. Following this, the argumentation process is repeated in order to discover new solutions.

Now it is important to note that different combinations of RB deletions could yield different RBD extensions, and our definition permits any of these combinations to be applied. Thus a single RAF may yield multiple RBD extensions for each extension computable from the original AF according to some semantics.

Example 4 *Consider* RAF F_2 *in Example 2, as in Example 3 and the RB* $(\{\mathsf{a}, \mathsf{x}, \mathsf{z}\}, \Sigma \leq 2)$*, with the selection criterion defined as* $\mathsf{a} \succ \mathsf{x} \succ \mathsf{z}$*. The least preferred argument in the RB set is* z *and its deletion makes both RB functions true. Hence, the RB deletion is* $\{\mathsf{z}\}$*. Finally, the RBD extensions of* F_2 *are the extensions of* $\langle \mathsf{A}'_2, \mathsf{D}'_2 \rangle$*, where* $\mathsf{A}'_2 = \{\mathsf{a}, \mathsf{b}, \mathsf{w}, \mathsf{y}\}$ *and* $\mathsf{D}' = \{\mathsf{a} \rightarrow \mathsf{b}, \mathsf{b} \rightarrow \mathsf{a}, \mathsf{b} \rightarrow \mathsf{y}\}$*.*

The following example illustrates a situation where RB extensions and RBD extensions can overlap:

Example 5 *Consider the* RAF $\langle \mathsf{A}_5, \mathsf{D}_5, \mathsf{R}_5 \rangle$*, as depicted in Figure 6. Set of arguments* $\{\mathsf{c}, \mathsf{e}\}$ *is not compliant with the RB* $\Sigma = 1$*. In the partitioning approach, extension* E *should be split in two:* $E_1 = \{\mathsf{b}, \mathsf{e}, \mathsf{f}\}$ *and* $E_2 = \{\mathsf{b}, \mathsf{c}, \mathsf{f}\}$*, but* E_2 *does not comply with the RB* $\Sigma > 1$*, nor does any subset, since the RB specifies a lower bound. Hence the only RB extension is* E_1*.*

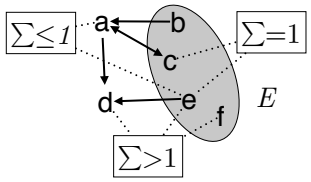


Fig. 6. RAF *for Example 5*

Regarding the approach by deletions, if we choose to remove c*, the resulting AF yields* E_1 *again. On the other hand, if we remove* e*, the resulting AF yields the extension* $\{\mathsf{b}, \mathsf{c}, \mathsf{d}, \mathsf{f}\}$*.*

Finally, to illustrate the utility of RBD extensions, consider the following example, which yields a RB truncated RAF, but which has RBD extensions.

Example 6 *Consider the RAF F_6 depicted in Figure 7. The only AF extension for F_6 is $\{\mathsf{a}, \mathsf{c}, \mathsf{d}\}$. The partitioning approach would attempt to shrink that extension until it gets RB-compliant subsets. Note that this is not possible, as there is no partition compliant with RB $\Sigma = 3$.*

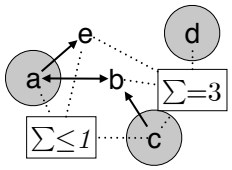


Fig. 7. *RAF for Example 6*

On the other hand, the approach by deletions would consider the RB deletions $\{\mathsf{c}\}$ and $\{\mathsf{a}, \mathsf{c}\}$, given that they yield the RBD extension $\{\mathsf{b}, \mathsf{d}, \mathsf{e}\}$, which is compliant with both RBs. In this case, we could choose to perform a minimal RB deletion, removing only c.

It is important to note that the application of one RB deletion, which deletes arguments from the AF, can cause other RBs in the RAF to no longer be complied with, requiring additional RB deletions over those previously applied in order to comply with an RB. This suggests that the order in which RB deletions are applied is important when computing extensions, and that an implemented system might have to utilise backtracking if properties such as the maximality of an RBD extension are required.

As with the partitioning approach, there is no general relation between the cardinality of the set of RBD extensions with respect to the set of AF extensions of a given RAF.

Proposition 2. *Let $F = \langle \mathsf{A}, \mathsf{D}, \mathsf{R} \rangle$ be a RAF and S, an argumentation semantics, it holds that:*

1. *E_{af} is both an AF extension and an RBD extension for F following S iff the empty set is an RBD deletion for E_{af} wrt. an RB in* R*;*
2. *the set of AF extensions and RBD extensions for F following S coincide iff the empty set is an RB deletion for every AF extension for F following S wrt. every RB in* R*.*

Now, given some RAF, it can be the case that there is no RB deletion for one or more members of its resource bound set R. We call such a RAF a RBD truncated RAF. Furthermore, while a RAF may not be truncated, it may have no RBD extensions as any sequence of RB deletions which can be applied can cause some other RB to be violated.

Having formalised various concepts around the idea of resource bounded argumentation, we now describe its use in the FITS project, wherein it is used as a simple scheduler.

4 The FITS Project: an Application for the RAF

The FITS project (standing for "flexible integrated transport services") falls within the Rural Digital Economy Hub (*dot.rural*[3]) agenda, funded by RCUK. FITS aims at providing a *virtual transport marketplace* to improve the existing connection between transport demand and supply in rural areas of the United Kingdom. The engine of this system is argumentation-based.

Flexible transportation systems (FTS) provide services to users based on demand, attempting to maximise ubiquitousness without compromising cost and quality. FTSs involve several elements, namely: acquisition of data (passengers, traffic, environment, fleet, *etc.*), evaluation of plausible options and journey planning. In this research the main focus is set on assisting the decision making process of passengers when selecting transport mode and route, while taking into account the operators' preferences, who provide resources. The objective of this research is the implementation of an argumentation-based expert system built upon a multi-agent system. Each passenger would have an associated agent, in charge of collecting relevant data, either permanently or triggered by trip requests. Similarly, agents will act on behalf of operators, imposing their restrictions and preferences regarding the usage of vehicles.

The aim of this effort is not to emulate (or replace) already existing FTS solutions for scheduling [2,8]. Instead, our approach looks to add value to the user's choice, taking the plausible options given as an output by some scheduler. We contend that the nature of argumentation for decision making allows for a clear presentation about how the process is carried out. Moreover, an interesting interface challenge lies in how to show an incrementally complex argumentation graph backing the suggestions made by the system.

Including RBs into the system permits a more concise representation, while accurately reflecting how the decision is being made. The following are the different elements that come into play in the FITS project domain:

- each possible journey for any passenger will constitute an **argument**;
- each sub-graph containing the alternatives for one passenger will be star-connected with **conflicts**;
- **defeat** is determined upon conflict, relying on passengers' preferences;
- the number of available seats in a given vehicle at a certain stop will determine a **resource bound**;
- the **selection criterion** for arguments is designed to ensure that passengers with fewer options get seats.

Additional RBs can also be taken into account, such as operator-side constraints controlling the minimum number of seats or minimum revenue. However, these have not yet been integrated into the system.

Our current focus involves how to determine the best journey for each passenger, while taking into account other passengers through resource bounds. This means that passenger's choices are not always weighted in isolation; a global balance must be

[3] `http://www.dotrural.ac.uk`

sought in order to achieve a certain degree of fairness. We have implemented a prototype system, which we are currently evaluating over randomly generated scenarios (maps, passengers and vehicles). The system computes possible journeys (*i.e.*, arguments), conflicts, RBs, and then makes a decision using the RBD approach on top of the preferred semantics. Within FITS, individual passengers' arguments associated with different journey options form separate subgraphs linked by RBs, and all RBs indicate a single upper bound (representing seat availability within a vehicle). When an RB is not met, we choose to maximise resource consumption, and therefore consider minimal RB deletions.

Figure 8 illustrates an argument graph generated by a sample scenario from our system. Rectangles representing individual journeys (and thus arguments) contain a passenger name, the journey's total cost and distance, the sequence of stops, the sequence of vehicles used and at which stops, and the passenger's preferences. For example, argument (6) states that one of Ruth's possible journeys has a cost of $52; a length of 86 kilometres; goes from $s4$ to $s5$ through $s1$ and $s3$, and is travelled by taking vehicles $v7$ at $s4$ and then $v5$ at $s1$; finally, she has a preference for shorter trips. If distances are equal, she looks for as few changes as possible, and if these are equal she looks to minimise the trip's cost. If multiple trips equally satisfy all these requirements, one of them is randomly selected. Solid arrows within the graph indicate defeat based on preferences, with two way arrows linking equivalently preferred options. Elliptical nodes indicate RBs due to seat restrictions; for instance, $v5/s3(1)$ means that vehicle $v5$ at stop $s3$ has only one seat available.

Treating Figure 8 as a standard AF, we see that trip (6) would appear in all preferred extensions. This indicates that this trip is the one most preferred by Ruth. However, when considering resource bounds, our system does not allow Ruth to undertake this trip. To see why, consider, for example, the resource bound $v5/s3(1)$, stating that vehicle 5 has only one free seat at location s3. Utilising RBD extensions, our seat allocation strategy (allocating seats to passengers with fewer options) means that $(7) \succ (6)$. Therefore, argument (7) will appear in the RBD extension, while argument (6) will not, meaning that Hien will obtain this vehicle's seat. This resource bound also means that argument (2) cannot appear in the RBD extension. Since arguments (1) and (3) are equally preferred, the system randomly allocates Rob to the trip represented by argument (1). Within Figure 8, the different status of these arguments is indicated by different colours and shadings.

5 Discussion and Future Work

As hinted at in Section 4, selection criteria can be used, in the process of computing RBD extensions, to provide us with certain desirable properties for those extensions. Such properties could, for example, include minimising the number of arguments removed from an extension, maximising the size of some (or the largest) extension, and so on. Such properties have an analogy with the concept of *minimal change* in the area of belief revision [1]. More specifically, since the RB deletion approach fundamentally modifies the knowledge base, it is desirable to ensure that this change is performed with the smallest possible representational impact.

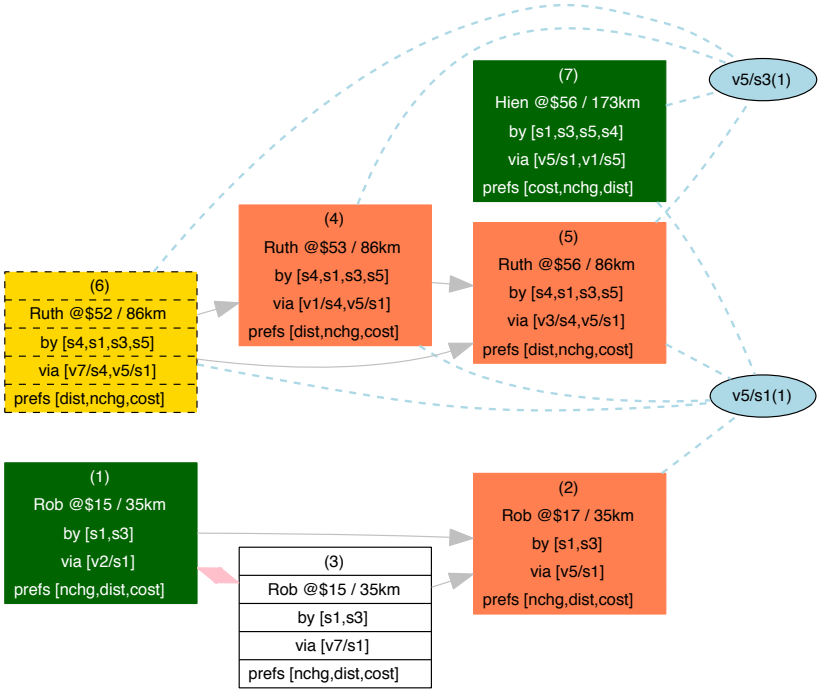


Fig. 8. Screenshot

One (computationally intensive) approach for minimising the representational impact of RB deletion would involve considering all possible RB deletions, examining the resultant extension(s), and selecting the one that meets our (domain specific) impact requirements. As an example, consider yet another possible metric of minimal change, namely the preservation of arguments found within an extension. Referring to Example 1, we see that the AF extension $\{\mathsf{a}, \mathsf{c}\}$ does not comply with RB $\Sigma \leq 1$. There are three possible RB deletions: $\{\mathsf{a}, \mathsf{c}\}, \{\mathsf{a}\}, \{\mathsf{c}\}$. The first and second options yield RBD extension $\{\mathsf{b}\}$, while the third one yields $\{\mathsf{a}\}$. The latter is the one that best preserves the AF extension. By having c as the least preferred extension within the selection criteria, we can ensure that the desired extension will be obtained. One interesting piece of future work involves determining the conditions required to guarantee that RBD extensions will meet some representational impact requirements without requiring the fine tuning of the selection criteria. One possible inspiration for this work might arise from the recent interest in change over argumentation frameworks [4,6,13].

The formal properties of RAFs provide fertile ground for additional future work. For example, it would be useful to identify the situations under which RB deletions exist, but

will not yield a RBD extension. In the short term, our focus lies in investigating whether a mapping exists between RAFs and traditional AFs. The propositions in this paper identify some situations where such a mapping exists, and a more complex conversion process, following the ideas of [**?**], has allowed us to map between RAFs and Dung AFs in many situations. However, this mapping yields incorrect results in some cases, and we intend to investigate when, and why the mapping process succeeds or fails.

Within the argumentation domain, recent work on weighted argument frameworks [10] bears some relation to our approach. Here, the authors define an abstract argument framework that assigns different weights to attacks. An inconsistency budget permits conflicts to exist within an extension only when the total weight of attacks is below this budget. This can be viewed as a single type of constraint on attacks. In our current work we constrain arguments, rather than attacks, and allow for arbitrary types of constraints. We believe that there is no mapping between weighted frameworks and RAFs, unless attacks are associated to resource consumption, but further theoretical investigation is needed to verify this intuition. More generally, RBs within an RAF can be seen as a type of constraint, and applications of results from work on both constraint satisfaction [] and constraint optimisation [] to both practical implementations of RAFs, and the theory behind them might yield interesting results.

6 Conclusions

In this article we have presented a novel approach to formal argumentation that takes into account limitation of resources. The introduction of resource bounds (RB) calls for a redefinition on how to compute the set of warranted arguments. To this end, we propose two different approaches: one of them considers partitions of pre-calculated extensions; the other one considers the removal of those arguments that make extensions not to be compliant with certain RBs.

Resource-bounded argumentation frameworks are based on Dung's standard argument framework, but augment it by labelling arguments with additional parameters, and placing constraints on the combinations of labels that are permitted to exist within an extension. We concentrated on arguments with a single label consisting of the number 1, with our constraints requiring that the sum of these for some set of arguments to be equal, less than, or greater than some value. Naturally, our formalism allows for the implementation of any other constraint.

Finally, we detailed how RAFs can be used to attack an important real world problem, namely that of scheduling dynamic transport provision. We believe that RAFs are applicable to a large variety of other domains where constraints exist, and can be used to bring the strong reasoning processes found in argumentation theory to tackle such problems, something which current work in argumentation has been unable to do.

Acknowledgments

The authors would like to thank Nagendra R. Velaga and John D. Nelson (*dot.rural Digital Economy Hub, University of Aberdeen*) for their contribution on the application of our model to the transportation domain.

References

1. Alchourrón, C., Gärdenfors, P., Makinson, D.: On the logic of theory change: Partial meet contraction and revision functions. The Journal of Symbolic Logic 50, 510–530 (1985)
2. Ambrosino, G., Nelson, J.D., Romanazzo, M.: Demand responsive transport services: Towards the flexible mobility agency. ENEA Publications (2004)
3. Atkinson, K., Bench-Capon, T.J.M.: Practical reasoning as presumptive argumentation using action based alternating transition systems. Artif. Intell. 171(10-15), 855–874 (2007)
4. Baumann, R., Brewka, G.: Expanding argumentation frameworks: Enforcing and monotonicity results. In: COMMA. pp. 75–86 (2010)
5. Boella, G., Gabbay, D., van der Torre, L., Villata, S.: Meta-Argumentation Modelling I: Methodology and Techniques. Studia Logica 93, 297–355 (2009)
6. Boella, G., Kaci, S., van der Torre, L.: Dynamics in argumentation with single extensions: Abstraction principles and the grounded extension. In: ECSQARU. pp. 107–118 (2009)
7. Boella, G., van der Torre, L., Villata, S.: On the acceptability of meta-arguments. In: IAT. pp. 259–262 (2009)
8. Brake, J., Mulley, C., Nelson, J.D., Wright, S.: Key lessons learned from recent experience with flexible transport services. Transport Policy 14(6), 458–466 (2007)
9. 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)
10. Dunne, P.E., Hunter, A., McBurney, P., Parsons, S., Wooldridge, M.: Weighted argument systems: Basic definitions, algorithms, and complexity results. Artif. Intell. 175(2), 457–486 (2011)
11. Matt, P.A., Morge, M., Toni, F.: Combining statistics and arguments to compute trust. In: AAMAS. pp. 209–216 (2010)
12. Oren, N., Panagiotidi, S., Vázquez-Salceda, J., Modgil, S., Luck, M., Miles, S.: Towards a formalisation of electronic contracting environments. In: COIN@AAMAS&AAAI. pp. 156–171 (2008)
13. Rotstein, N.D., Moguillansky, M., Falappa, M.A., García, A.J., Simari, G.R.: Argument Theory Change: Revision Upon Warrant. In: 2nd International Conference on Computational Models of Argument (COMMA 2008). pp. 336–347 (2008)

Argumentation Schemes for Policy-Driven Planning

Alice Toniolo[1], Timothy J. Norman[1], and Katia Sycara[2]

[1] Department of Computing Science, University of Aberdeen, Scotland, UK
{a.toniolo,t.j.norman}@abdn.ac.uk
[2] Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, US
katia@cs.cmu.edu

Abstract. Members of a heterogeneous team of autonomous agents may have conflicting opinions about what is the best course of action to adopt. For effective teamwork, mechanisms that enable agreements to be reached regarding a shared plan are essential. We propose a model of arguments based on argumentation schemes that can be used in team deliberative dialogues focussed on establishing such agreements. We explore conflict situations where some actions may be temporally incompatible with existing commitments or may conflict with norms. This model facilitates the exchange of information about plans for agents with different objectives and norms in order to achieve more favourable agreements.

Keywords: Argumentation schemes, Practical reasoning, Planning

1 Introduction

Members of a heterogeneous team of autonomous agents may have different resources and capabilities as well as differences in objectives and policies. In this context, teamwork is, often, hampered by conflicting opinions about what is the best course of action to adopt in a joint plan. Recent work on argumentation for deliberative dialogues in multi-agent system has shown that argumentation is a promising approach for generating consistent collaborative plans [1, 7]. We believe that the use of argumentation-based models to support dialogues during planning activity has the potential to facilitate agreements for a joint plan.

In this paper we present a model for generating arguments in multi-agent deliberative dialogues that allows agents to clarify the nature of the conflicts in a joint plan. In previous research, the focus of the discussion was mostly to decide which action was more adequate according to values or costs [1, 7]. In contrast, in our approach, agents are heterogenous and have individual objectives and plan-constraints to follow. We consider, in fact, a discussion within a team focussed upon what is the best course of action to adopt in a collaborative plan on the basis of individual plans. In this paper, we explore a wider set of conflict situations than other approaches since the adoption of new actions in the plan may be temporally incompatible with actions that have already been committed to. Furthermore, plan constraints are not sufficiently expressive for

describing the reasons for adopting a certain plan. Team members might have internal regulations such as norms which will forbid them from or oblige them to perform an action or to achieve a state in the plan. Norms and plan-constraints guide the choice of the plan and, therefore, they should be considered in the discussion about a course of action. Previous work has considered these issues for argumentation in separate contexts for solving conflicts in practical reasoning [1, 7] and for norm adoption [6], but in this paper we bring together norm and plan-constraints within a single coherent model. In this paper, we propose a model of arguments that can contribute in team deliberative dialogues focussed on establishing agreements on a joint plan, based on argumentation schemes, that defines different kinds of arguments and rules for arguing about actions, goals and norms.

This paper is organised as follows. Section 2 describes a model for representing plans. In Section 3 we introduce the deliberative dialogue. In Section 4 the general structure of the argumentation framework is introduced followed by the different kinds of arguments. In Section 5 we describe the links among arguments of different nature. In Section 6 we discuss a possible application for this model. Section 7 introduces related work and Section 8 discusses future directions.

2 A Model of Plans

In our model, we consider agents that prepare individual plans and, then, engage in deliberative dialogue regarding collaborative actions and other dependencies. Teams are heterogeneous and team members may have different objectives not necessarily known to others. In this section we present a language for describing multi-agent plans that underpins our model.

Language. The language is composed of sets of constants Ψ_C, variables Ψ_V and predicate symbols Ψ_P. The connectives are $\wedge, \vee, \neg$. We refer to the set of agents, as $Agt = \{x, y, z, \dots\} \subset \Psi_C$. Actions and features of the world are represented as predicates composed by symbols in Ψ_P with an associated vector of n parameters in Ψ_V bound with values in Ψ_C. Actions are represented in the form $a(V_1, \dots, V_n)$ and features of the world as $p(V_1, \dots, V_m)$ where $a, p \in \Psi_P$ and $V_i \in \Psi_V$. Actions and features are instantiated by binding their variables V_i to constants $c_i \in \Psi_C$. For convenience, we refer to instantiated actions as $\alpha_1, \alpha_2, etc.$ and to features as $\rho_1, \rho_2, etc.$ A goal g is as a label $g = \rho_1 \circ \dots \circ \rho_q$ where $\circ \in \{\wedge, \vee\}$. The set of all instantiated actions is $\mathcal{A} = \{\alpha_1, \dots, \alpha_n\}$, the set of instantiated features of the world is $\mathcal{R} = \{\rho_1, \dots, \rho_m\}$ and the set of goals is $\mathcal{G} = \{g_1, \dots, g_l\}$. In referring to plans, we distinguish between a partially ordered plan P that captures the causal dependencies among actions and a temporally grounded plan $\vec{P}$ obtained from P.

Partially Ordered Plan. Each agent has an individual plan representation based on classical planning problems. The planning domain for an agent x includes an initial state of the world $\mathcal{R}^x_{init} \subset \mathcal{R}$, a goal state $\mathcal{G}^x_P \subset \mathcal{G}$ and a set of actions $\mathcal{A}^x \subset \mathcal{A}$. An action $\alpha_k \in \mathcal{A}^x$ is defined in terms of its preconditions $\mathcal{R}_{Pr}(\alpha_k) \subset \mathcal{R}$ and effects $\mathcal{R}_{Ef}(\alpha_k) \subset \mathcal{R}$. This captures the idea that if the action

were to be executed in a world in which the preconditions are satisfied, the effects would be expected to be produced. We define an application rule for the action α_k as $R(\alpha_k) = \{\mathcal{R}_{Pr}(\alpha_k), \alpha_k \Rightarrow \mathcal{R}_{Ef}(\alpha_k)\}$. The set of application rules is called Π_R. In the knowledge of an agent a subset of planning rules is known as $\Pi_R^x \subset \Pi_R$. The planning problem for an agent x is defined by the initial state of the world $\mathcal{R}_{init}^x \subset \mathcal{R}$ (i.e. a set of features about the world that is considered to be true by agent x), the set of application rules Π_R^x and the set of goals $\mathcal{G}_P^x$. An individual partially ordered plan P^x for agent x is a solution to this problem such that starting from the initial state of the world and executing a set of action $\mathcal{A}_P^x$, the world would result in a state $\mathcal{R}_{fin}^x$ that contains features that satisfy the goals $\mathcal{G}_P^x$. The plan P^x consists in a set of actions $\mathcal{A}_P^x \subseteq \mathcal{A}^x$, a set of causal links $\mathcal{D}_P^x$ and a set of safety conditions $\mathcal{C}_P^x$ (following the approach of [4]). The partially ordered plan P^x also contains two *special* actions $START$ and $STOP$ with application rules $R(START) = \{\emptyset, START \Rightarrow \mathcal{R}_{init}^x\}$ and $R(STOP) = \{\mathcal{R}_{fin}^x, STOP \Rightarrow \emptyset\}$. Dependencies $\mathcal{D}_P^x$ and safety conditions $\mathcal{C}_P^x$ among actions in the partially ordered plan are defined as follows. Given an action α_k for an agent x with application rule $R(\alpha_k) = \{\mathcal{R}_{Pr}(\alpha_k), \alpha_k \Rightarrow \mathcal{R}_{Ef}(\alpha_k)\}$, if $\rho \in \mathcal{R}_{Pr}(\alpha_k)$ there is an action α_h in the plan that has an effect ρ in order for action α_k to be executed. Action α_h has an application rule $R(\alpha_h) = \{\mathcal{R}_{Pr}(\alpha_h), \alpha_h \Rightarrow \mathcal{R}_{Ef}(\alpha_h)\}$ where $\rho \in \mathcal{R}_{Ef}(\alpha_h)$. If ρ is true at the outset ($\rho \in \mathcal{R}_{init}^x$) then $\alpha_h = START$. The dependence is a causal link between α_h and α_k and a tuple $\langle \alpha_h, \rho, \alpha_k \rangle$ defines it. The set of tuples representing the causal links specified in the partially ordered plan is called $\mathcal{D}_P^x$. Since α_k depends on α_h, the execution of α_h should precede the execution of α_k in the plan. We represent this as the safety condition $\alpha_h \ll \alpha_k$. Furthermore, an action α_j is a threat to the causal link $\langle \alpha_h, \rho, \alpha_k \rangle$ if $R(\alpha_j) = \{\mathcal{R}_{Pr}(\alpha_j), \alpha_j \Rightarrow \mathcal{R}_{Ef}(\alpha_j)\}$ and $\neg\rho \in \mathcal{R}_{Ef}(\alpha_j)$. In order for the individual plan to be consistent α_j should be scheduled before or after the two actions, $\alpha_j \ll \alpha_h$ or $\alpha_k \ll \alpha_j$, so that there are no threats for the causal links in P^x. These safety conditions are gathered in a set $\mathcal{C}_P^x$. For discussion purposes we also define the set of features effectively brought about by x to achieve g_k as $\mathcal{R}_P^x(g_k)$; for example, if $g_k = \rho_1 \vee \rho_2$, $\rho_1 \in \mathcal{R}_P^x(g_k)$ or $\rho_2 \in \mathcal{R}_P^x(g_k)$ or both. Therefore, the final state of the world, when goals $\mathcal{G}_P^x$ are achieved is composed by $\mathcal{R}_{fin}^x = \mathcal{R}_P^x(g_1) \cup \cdots \cup \mathcal{R}_P^x(g_l)$. The sequence of actions that will lead an agent to achieve a goal g_k is represented as $\mathcal{A}_P^x(g_k) \subseteq \mathcal{A}_P^x$.

Temporally Grounded Plan. In order for the parties to compare their plans during the dialogue, participants need a common reference point for actions and states of the world. In our approach this reference is given by a common time line. Agents are provided with a definition of a set of integer time stamps $T \subset \Psi_C$ with $T = \{t_0, \ldots, t_n\}$. A partially ordered plan P^x can be scheduled along this time line obtaining a temporally grounded plan $\vec{P}^x$ which can be compared with the other parties. $\vec{P}^x$ maintains the order defined by the causal links and safety conditions in P^x but specifies when actions are scheduled to take place. A feature of the world ρ is considered true for an agent x if it holds at time t_k and the representation of this concept is expressed by $[t_k]\rho$. A set $[t_k]\mathcal{R}^x = \{[t_k]\rho_1, \ldots, [t_k]\rho_n\}$ represents a state of the world for an agent x which holds

at time t_k. Actions rely on the same time line and their duration is represented as $[t_{start}, t_{stop}]\alpha$ for an action α being performed in the interval between t_{start} and t_{stop}. We assume that plan $\vec{P}^x$ starts in t_{init} and finishes at time t_{end}. The plan $\vec{P}^x$ is a collection of actions $\mathcal{A}^x_{\vec{P}}$ where for each action $\alpha \in \mathcal{A}^x_P$ the corresponding action $[t_1, t_2]\alpha \in \mathcal{A}^x_{\vec{P}}$. Preconditions and effects of all the actions in the plan along with the initial and final state of the world gather into the set of temporally grounded features of the world $\mathcal{R}^x_{\vec{P}}$. We assume that unless modified the features of the world will persist in time. The set of goals for plan $\vec{P}^x$ coincides with the set of goals for the partially ordered plan P^x ($\mathcal{G}^x_P = \mathcal{G}^x_{\vec{P}}$). Henceforth, the plan elements will be expressed with their temporal reference.

Norms. An important aspect underlying our model is that of normative constraints. Norms represent rules that specify what an agent is obliged, permitted or forbidden to do in terms of actions and features of the world under certain conditions. We assume that everything is permitted when not explicitly prohibited. Norms are instantiated during the planning process. An individual plan P^x includes all the actions and features obliged and does not include the forbidden ones. The set of norms for an agent x is called Π^x_N. A norm $N_i \in \Pi^x_N$ is described by actions and features of the world that cause its activation. These premises are represented as a set of actions ($\mathcal{A}_{Prem}(N_i) \subseteq \mathcal{A}$) and set of features of the world ($\mathcal{R}_{Prem}(N_i) \subseteq \mathcal{R}$). The norm N_i regulates conclusions about an action $\alpha_k \in \mathcal{A}$ or a feature of the world $\rho_k \in \mathcal{R}$. A norm for an agent x also specifies the interval $[t_{n1}, t_{n2}]$, where t_{n1} is the time when the norm becomes active and t_{n2} indicates the time when the norm expires. Hence, an instantiated norm for an agent x is: $N_i = \{\mathcal{R}_{Prem}(N_i) \cup \mathcal{A}_{Prem}(N_i) \Rightarrow [t_{n1}, t_{n2}]O/F_x : \rho_k/\alpha_k\}$ where F=Forbid and O=Oblige. We assume that all norms for some agent x are logically consistent.

Goal generation. An agent is motivated by state of the world to adopt some goals. An agent x generates a set of goal candidates according to its view of the world; for example, if agent x believes that the world has features $\rho_a, \ldots, \rho_q$, x will generate goal g_k. A goal g_k among the generated goal is, then, committed to only if a norm-consistent plan for achieving g_k exists. In this case g_k will be part of the committed goals $\mathcal{G}^x_P$ and the actions of the plan for g_k will be included in $\mathcal{A}^x_P$. The rule representing the state of the world that motivate agent x to activate goal g_k is defined as $S(g_k) = \{\rho_a, \ldots, \rho_q \Rightarrow g_k\}$. Agent x is provided with a set of activation rules for goals called Π^x_S and $S(g_k) \in \Pi^x_S$.

Example 1. Agent x represents a travel agency located which wants to schedule a new trip to the seaside. This goal is motivated by the fact that it is summer and people may plan holidays soon. There are two possible destinations called C and D. The travel agency decides that C is the only destination possible because there is evidence that in D the water is polluted and an internal policy forbids, for the safety of the customers, to schedule trips to destination in these conditions. In order to reach C a customer cm would leave from a location A, he would reach a location B by a train tr (indicated with the instantiated predicate $take(cm, tr, A, B)$) and from this point he will take a bus bs to achieve the destination C ($take(cm, bs, B, C)$). In the description of this plan some predicates are used as follows: $in(?obj, ?loc)$ indicates that object $?obj$ is in a location

$?loc$; $dest(?obj, ?loc)$ indicates that the destination of object $?obj$ is location $?loc$; three other predicates give information about a location: $waterPollutedIn(?loc)$, $summerIn(?loc)$, $holidayIn(?loc)$. We refer to instantiated features in the plan as: $\rho_1 = in(cm, A)$, $\rho_2 = in(tr, A)$, $\rho_3 = in(tr, B)$, $\rho_4 = in(cm, B)$, $\rho_5 = in(cm, C)$, $\rho_6 = in(cm, D)$, $\rho_7 = in(bs, B)$, $\rho_8 = in(bs, C)$, $\rho_9 = dest(tr, B)$, $\rho_{10} = dest(bs, C)$, $\rho_{11} = waterPollutedIn(D)$, $\rho_{12} = summerIn(A)$, $\rho_{13} = holidayIn(A)$. The actions are $\alpha_1 = take(cm, tr, A, B)$, $\alpha_2 = take(cm, bs, B, C)$.

The rules for actions, norms and goals are represented in the following sets. The initial conditions and goals are: $\mathcal{R}^x_{init} = \{\rho_1, \rho_2, \rho_7, \rho_9, \rho_{10}, \rho_{11}, \rho_{12}, \rho_{13}\}$; $\Pi^x_S = \{S(g_1)\}$ where $S(g_1) = \{\rho_{12} \wedge \rho_{13} \Rightarrow g_1\}$; $\mathcal{G}^x_{\vec{P}} = \{g_1\}$ where $g_1 = \rho_5 \vee \rho_6$; $\mathcal{R}^x_P(g_1) = \{\rho_5\}$. The norm is represented as $N_1 = \{[t_0]\rho_{11} \Rightarrow [t_0, t_6]F_x : \rho_6\}$ and $\Pi^x_N = \{N_1\}$. The actions are $\mathcal{A}^x_{\vec{P}} = \{[t_1, t_3]\alpha_1, [t_3, t_6]\alpha_2\}$ with applications rules $\Pi^x_R = \{R(\alpha_1), R(\alpha_2)\}$ where $R(\alpha_1) = \{[t_1]\rho_1 \wedge [t_1]\rho_2 \wedge [t_1]\rho_9, [t_1, t_3]\alpha_1 \Rightarrow [t_3]\rho_3 \wedge [t_3]\rho_4 \wedge [t_3]\neg\rho_1 \wedge [t_3]\neg\rho_2\}$ and $R(\alpha_2) = \{[t_3]\rho_4 \wedge [t_3]\rho_7 \wedge [t_3]\rho_{10}, [t_3, t_6]\alpha_2 \Rightarrow [t_6]\rho_5 \wedge [t_6]\rho_8 \wedge [t_6]\neg\rho_4 \wedge [t_6]\neg\rho_7\}$. The causal link between α_1, α_2 is $\mathcal{D}^x_P = \{\langle\alpha_1, \rho_4, \alpha_2\rangle\}$ and $\mathcal{C}^x_P = \{\alpha_1 \ll \alpha_2\}$. Both actions contribute to achieve g_1, $\mathcal{A}^x_P(g_1) = \{\alpha_1, \alpha_2\}$.

3 Deliberative Dialogue

In this section we present argumentation schemes and dialogical structures for deliberative dialogue. These schemes capture patterns of argument that commonly occur in deliberative dialogue [9]. These schemes can be seen as heuristics for structuring the argumentation dialogue concerned with practical reasoning, and hence, refer to an agent's planned activities. We, therefore, require sentences in the language for agents to use, that refer to agents doing actions, achieving goals, *etc.* To this end, we define the following: a predicate $do(x, \alpha_k, t_{start}, t_{stop})$ indicates that an agent x performs an action $[t_{start}, t_{stop}]\alpha_k$ in the interval $[t_{start}, t_{stop}]$; a predicate $bringAb(x, \rho_k, t_k)$ indicates that an agent x brings about a feature $[t_k]\rho_k$ at time $[t_k]$; a similar predicate $bringAb(x, \mathcal{R}, t_k)$ indicates that an agent x brings about a state of the world $[t_k]\mathcal{R}^x$ at time $[t_k]$; a predicate $achieve(x, g_k)$ indicates that an agent x achieves a goal g_k.

The formal structure of the arguments will be represented with the structure $\langle Prem \Rightarrow Concl\rangle$, where $Prem$ refers to the premises and $Concl$ to the conclusions. An argumentation scheme includes a set of possible attacks (ATK) and supports (SP) where attacks are those that negate elements presented in the premises and conclusions of the argument, while supports promote these. For the purpose of this research we consider ATKs and SPs as arguments that can be equivalently used as arguments during the discussion, where support is intended as a counterattack to the opponent's attack. ATKs and SPs will be identified by critical questions CQs.

The dialogue among agents follows existing protocols presented in the literature (e.g. [5]). For example, the planning process for a team of two agents x, y begins with preparing separate individual plans $\vec{P}^x$ and $\vec{P}^y$. Assume that agent x has evidence that some actions of its individual plan need to be discussed with the other member y. Many reasons can motivate this evidence: an action of an

agent may depend on the effects of the actions of others; an agent should act on behalf of another for certain reasons; one agent may need to obtain a permission from others to perform an action; *etc.* When an action is shared for discussion, x and y engage in a debate to decide what is the best course of action to adopt as a team. The conversation between x and y is opened when agent x informs y that it intends to discuss an action $[t_{k1}, t_{k2}]\alpha_k \in \mathcal{A}^x_{\vec{P}}$. When agent x performs the action, the topic of the debate is $do(x, \alpha_k, t_{k1}, t_{k2})$, otherwise when α_k is to be performed by agent y the discussion is about $do(y, \alpha_k, t_{k1}, t_{k2})$. Agent y who receives an enquiry about the action can accept the proposal if it does not cause any conflicts with its commitments and norms in $\vec{P}^y$; otherwise, y will reject the proposal. In case of refusal, agent x may intend to discuss about the action by sending an argument in support of its decision. The two agents will, then, exchange a series of arguments to support their own opinions or to attack others' opinions. The discussion will end either when the two parties agree on which course of action to perform, or when there are no other new arguments to exchange. In case the agreement is not reached, agent x should withdraw the action. Agent x has to reconsider part of the individual plan discarding $[t_{k1}, t_{k2}]\alpha_k$ and replacing it with an alternative that would not conflict with the new information about y's plan gathered during the discussion. In case of agreement, agent y adds $[t_{k1}, t_{k2}]\alpha_k$ on the set of actions $\mathcal{A}^y_{\vec{P}}$, and update $\vec{P}^y$ with further information gathered during the discussion. Action $[t_{k1}, t_{k2}]\alpha_k$ becomes part of a set of agreed committed actions. For convenience we refer to this set as $\mathcal{A}^{x,y}_{\vec{P}}$.

4 Argumentation

We take as our starting point an adaptation of Atkinson's argumentation scheme for an individual action [1]. Proponent x uses this argument to initiate the debate about action $[t_{k1}, t_{k2}]\alpha_k$. The application rule $R(\alpha_k) \in \Pi^x_R$ has the structure $R(\alpha_k) = \{[t_{k1}]\mathcal{R}_{Pr}(\alpha_k), [t_{k1}, t_{k2}]\alpha_k \Rightarrow [t_{k2}]\mathcal{R}_{Ef}(\alpha_k)\}$. The argument is:

Arg_I: - Given preconditions $[t_{k1}]\mathcal{R}_{Pr}(\alpha_k)$
- $do(x, \alpha_k, t_{k1}, t_{k2})$
- will $bringAb(x, \mathcal{R}_{Ef}(\alpha_k), t_{k2})$
- that will contribute to $achieve(x, g_k)$ [where $\alpha_k \in \mathcal{A}^x_P(g_k)$]

$\Rightarrow$ therefore $do(x, \alpha_k, t_{k1}, t_{k2})$

The dialogue commences with this argument. The opponent y, receiving the argument, has a wider set of information about the action under discussion and y can select new arguments according to conflicts with its commitments and norms. An agent follows these critical questions to formulate such arguments:

- **CQ1:** Is the action possible according to the commitments of the plan?
- **CQ2:** Is there any norm which regulates actions or states of the world?
- **CQ3:** Is the achievement of the goal justified?

A description of these arguments is presented in the following sections.

4.1 Arguments for plan constraints

In this section we explore the first critical question *CQ1*, presenting the structure of the argument along with attacks and supports concerning with plan constraints. The argumentation scheme for plan-constraints Arg_p is an extension of the initial argument Arg_I. Atkinson [1] defines an argumentation scheme for practical reasoning regarding a single action α_k warranted by preconditions, where the effects allow the agent to achieve a goal which promotes or demotes certain values. In our model, we consider an extension of this structure such that the adoption of action α_k is considered in the context of related actions in the individual plan. This extension is fundamental in scenarios where agents are concerned with agreeing joint plans and not only about the best single action to perform. We focus, in fact, on the critical questions that challenge other arguments because of temporal conflicts between actions due to incompatible preconditions and effects. Since actions are applied over an interval of time, we require that for entire execution, there should be no direct interference on its preconditions and effects by other actions. We assume, for example, that a precondition is required to be true at the start of an action and throughout its execution. The following conditions describe when an action $[t_{k1}, t_{k2}]\alpha_k$ proposed by x can be safely included in y's plan. The application rule for α_k is $R(\alpha_k) = \{[t_{k1}]\mathcal{R}_{Pr}(\alpha_k), [t_{k1}, t_{k2}]\alpha_k \Rightarrow [t_{k2}]\mathcal{R}_{Ef}(\alpha_k)\}$.

1. if $\rho \in \mathcal{R}_{Ef}(\alpha_k)$ and there is a causal link $\langle \alpha_a, \neg\rho, \alpha_b \rangle \in \mathcal{D}_P^y$ where $[t_{a1}, t_{a2}]\alpha_a$, $[t_{b1}, t_{b2}]\alpha_b \in \mathcal{A}_{\vec{P}}^y$, then α_k is a threat to that link. In this case action α_k should respect the safety conditions $\alpha_k \gg \alpha_b$ or $\alpha_k \ll \alpha_a$. (Since nothing can be scheduled before $START$ and after $STOP$, for $\alpha_b = STOP$ the condition is $\alpha_k \ll \alpha_a$ and for $\alpha_a = START$ it is $\alpha_k \gg \alpha_b$). Therefore, the safety conditions indicate that intervals $[t_{a1}, t_{b2}]$ and $[t_{k1}, t_{k2}]$ must not overlap; i.e. $t_{k1} > t_{b2}$ or $t_{k2} < t_{a1}$. The interval $[t_{a1}, t_{b2}]$ is called the conflicting interval $\mathcal{T}_{\neg\rho}$ where $\neg\rho$ should hold and interferences of other actions are not allowed. We refer to actions α_a and α_b as actions that have induced the formation of $\mathcal{T}_{\neg\rho}$ and to the interval as $\mathcal{T}_{\neg\rho}^{(\alpha_a)}$ or $\mathcal{T}_{\neg\rho}^{(\alpha_b)}$ when highlighting the cause.
2. if $\rho \in \mathcal{R}_{Pr}(\alpha_k)$ and there is a causal link $\langle \alpha_a, \neg\rho, \alpha_b \rangle \in \mathcal{D}_P^y$ where $[t_{a1}, t_{a2}]\alpha_a$, $[t_{b1}, t_{b2}]\alpha_b \in \mathcal{A}_{\vec{P}}^y$, then the precondition of α_k would be negated by the execution of the sequence α_a and α_b. Action α_k should respect the safety conditions where $\alpha_k \gg \alpha_b$ or $\alpha_k \ll \alpha_a$. Hence, conclusions of point 1 apply.
3. if $[t_{k1}, t_{k2}]\alpha_k$ is not a threat for any causal links, it can be adopted if no features of the world achieved by α_k, $[t_{k2}]\rho \in [t_{k2}]\mathcal{R}_{Ef}(\alpha_k)$ negate side effects in $[t_{a2}]\mathcal{R}_{Ef}(\alpha_a)$ of an action $[t_{a1}, t_{a2}]\alpha_a \in \mathcal{A}_{\vec{P}}^y$. The interval $\mathcal{T}_{\neg\rho}^{(\alpha_a)} = [t_{a1}, t_{a2}]$ induced by α_a must not be overlapped by $[t_{k1}, t_{k2}]$.
4. if $[t_{k1}, t_{k2}]\alpha_k$ is not a threat for any causal links, it can be adopted if no preconditions for α_k, $[t_{k1}]\rho \in [t_{k1}]\mathcal{R}_{Pr}(\alpha_k)$, would be negated by an effect in $[t_{a2}]\mathcal{R}_{Ef}(\alpha_a)$ of an action $[t_{a1}, t_{a2}]\alpha_a \in \mathcal{A}_{\vec{P}}^y$. Conclusions of point 3 apply.

These conditions define conflicts that may arise when x proposes to adopt a new action α_k in $\mathcal{A}_{\vec{P}}^{x,y}$. These conflicts can be used by an agent y to dissuade agent x, the proponent of α_k. Argument Arg_p involves a new action

$[t_{k1}, t_{k2}]\alpha_k$ and an action $[t_{h1}, t_{h2}]\alpha_h$ already scheduled in the individual plan causing the conflict. We suppose that α_k is performed by agent x and α_h by y where $R(\alpha_k) \in \Pi_R^x$ and $R(\alpha_h) \in \Pi_R^y$. The application rule for α_h is $R(\alpha_h) = \{[t_{h1}]\mathcal{R}_{Pr}(\alpha_h), [t_{h1}, t_{h2}]\alpha_h \Rightarrow [t_{h2}]\mathcal{R}_{Ef}(\alpha_h)\}$. The previous conditions indicate that these two actions are conflicting because $\rho \in \mathcal{R}_{Pr}(\alpha_k) \cup \mathcal{R}_{Ef}(\alpha_k)$ and $\neg\rho \in \mathcal{R}_{Pr}(\alpha_h) \cup \mathcal{R}_{Ef}(\alpha_h)$. The interval $[t_{k1}, t_{k2}]$ overlaps the conflicting interval $\mathcal{T}_{\neg\rho}^{(\alpha_h)}$ induced by α_h; in the argument this is represented with predicate $overlaps([t_{k1}, t_{k2}], \mathcal{T}_{\neg\rho}^{(\alpha_h)})$. Argument Arg_p has the following structure:

Arg_p: - Given preconditions $[t_{k1}]\mathcal{R}_{Pr}(\alpha_k)$,
- $do(x, \alpha_k, t_{k1}, t_{k2})$,
- will $bringAb(x, \mathcal{R}_{Ef}(\alpha_k), t_{k2})$, [and $\rho \in \mathcal{R}_{Pr}(\alpha_k) \cup \mathcal{R}_{Ef}(\alpha_k)$],
- and given preconditions $[t_{h1}]\mathcal{R}_{Pr}(\alpha_h)$,
- $\neg do(y, \alpha_h, t_{h1}, t_{h2})$,
- will $\neg bringAb(y, \mathcal{R}_{Ef}(\alpha_h), t_{h2})$, [and $\neg\rho \in \mathcal{R}_{Pr}(\alpha_h) \cup \mathcal{R}_{Ef}(\alpha_h)$],
- that will contribute $\neg achieve(y, g_h)$ [where $\alpha_h \in \mathcal{A}_P^y(g_h)$],
- and $overlaps([t_{k1}, t_{k2}], \mathcal{T}_{\neg\rho}^{(\alpha_h)})$

$\Rightarrow$ therefore $\neg do(x, \alpha_k, t_{k1}, t_{k2}) \wedge do(y, \alpha_h, t_{h1}, t_{h2})$

Grouping the conditions in conflicts between two actions we obtain the questions:

- **ATK1.1:** If action α_k has an effect ρ that contradicts an effect $\neg\rho$ of an action α_h in the individual plan, does the interval of execution of α_k overlap the conflicting interval induced by α_h? Violation of conditions 1 or 3.
- **ATK1.2:** If action α_k has an effect ρ that negates a precondition $\neg\rho$ for an action α_h in the individual plan, does the interval of execution of α_k overlap the conflicting interval induced by α_h? Violation of condition 1.
- **ATK1.3:** If action α_k has a precondition ρ that would be negated by an effect $\neg\rho$ of an action α_h, does the interval of execution of α_k overlap the conflicting interval induced by α_h? Violation of conditions 2 or 4.
- **ATK1.4:** If action α_k has a precondition ρ that contradicts a precondition $\neg\rho$ for an action α_h, does the interval of execution of α_k overlap the conflicting interval induced by α_h? Violation of condition 2.

On the other hand, if x's plan contains a causal link from α_k to another action α_j such that $\langle \alpha_k, \rho, \alpha_j \rangle \in \mathcal{D}_P^x$, then α_k is necessary to perform α_j. Agent x can formulate a support answering the following question:

- **SP1.1** Does action α_k have effects that are fundamental preconditions for some other actions in the individual plan?

The formal representation of attacks and support for Arg_p is presented in Tab.1.

4.2 Arguments for norms

In this section we explore the critical question *CQ2* and define the argument for norms Arg_n. We consider norms as external regulations about what the agent is forbidden or obliged to do in terms of actions and states of the world. We assume that everything is permitted when not explicitly prohibited.

Table 1. Arguments Arg_p

CQ1-Is the action possible?
ATK1.1: $\langle [t_{k1}]\mathcal{R}_{Pr}(\alpha_k), do(x,\alpha_k,t_{k1},t_{k2}), bringAb(x,\mathcal{R}_{Ef}(\alpha_k),t_{k2}) \wedge \boldsymbol{\rho \in \mathcal{R}_{Ef}(\alpha_k)}$ $[t_{h1}]\mathcal{R}_{Pr}(\alpha_h), \neg do(y,\alpha_h,t_{h1},t_{h2}), \neg bringAb(y,\mathcal{R}_{Ef}(\alpha_h),t_{h2}) \wedge \boldsymbol{\neg\rho \in \mathcal{R}_{Ef}(\alpha_h)}$, $\neg achieve(y,g_h), overlaps([t_{k1},t_{k2}],\mathcal{T}^{(\alpha_h)}_{\boldsymbol{\neg\rho}}), \Rightarrow \neg do(x,\alpha_k,t_{k1},t_{k2}) \wedge do(y,\alpha_h,t_{h1},t_{h2})\rangle$
ATK1.2: $\langle [t_{k1}]\mathcal{R}_{Pr}(\alpha_k), do(x,\alpha_k,t_{k1},t_{k2}), bringAb(x,\mathcal{R}_{Ef}(\alpha_k),t_{k2}) \wedge \boldsymbol{\rho \in \mathcal{R}_{Ef}(\alpha_k)}$ $[t_{h1}]\mathcal{R}_{Pr}(\alpha_h) \wedge \boldsymbol{\neg\rho \in \mathcal{R}_{Pr}(\alpha_h)}, \neg do(y,\alpha_h,t_{h1},t_{h2}), \neg bringAb(y,\mathcal{R}_{Ef}(\alpha_h),t_{h2})$, $\neg achieve(y,g_h), overlaps([t_{k1},t_{k2}],\mathcal{T}^{(\alpha_h)}_{\boldsymbol{\neg\rho}}) \Rightarrow \neg do(x,\alpha_k,t_{k1},t_{k2}) \wedge do(y,\alpha_h,t_{h1},t_{h2})\rangle$
ATK1.3: $\langle [t_{k1}]\mathcal{R}_{Pr}(\alpha_k) \wedge \boldsymbol{\rho \in \mathcal{R}_{Pr}(\alpha_k)}, do(x,\alpha_k,t_{k1},t_{k2}), bringAb(x,\mathcal{R}_{Ef}(\alpha_k),t_{k2})$ $[t_{h1}]\mathcal{R}_{Pr}(\alpha_h), \neg do(y,\alpha_h,t_{h1},t_{h2}), \neg bringAb(y,\mathcal{R}_{Ef}(\alpha_h),t_{h2}) \wedge \boldsymbol{\neg\rho \in \mathcal{R}_{Ef}(\alpha_h)}$, $\neg achieve(y,g_h), overlaps([t_{k1},t_{k2}],\mathcal{T}^{(\alpha_h)}_{\boldsymbol{\neg\rho}}) \Rightarrow \neg do(x,\alpha_k,t_{k1},t_{k2}) \wedge do(y,\alpha_h,t_{h1},t_{h2})\rangle$
ATK1.4: $\langle [t_{k1}]\mathcal{R}_{Pr}(\alpha_k) \wedge \boldsymbol{\rho \in \mathcal{R}_{Pr}(\alpha_k)}, do(x,\alpha_k,t_{k1},t_{k2}), bringAb(x,\mathcal{R}_{Ef}(\alpha_k),t_{k2})$ $[t_{h1}]\mathcal{R}_{Pr}(\alpha_h) \wedge \boldsymbol{\neg\rho \in \mathcal{R}_{Pr}(\alpha_h)}, \neg do(y,\alpha_h,t_{h1},t_{h2}), \neg bringAb(y,\mathcal{R}_{Ef}(\alpha_h),t_{h2})$, $\neg achieve(y,g_h), overlaps([t_{k1},t_{k2}],\mathcal{T}^{(\alpha_h)}_{\boldsymbol{\neg\rho}}) \Rightarrow \neg do(x,\alpha_k,t_{k1},t_{k2}) \wedge do(y,\alpha_h,t_{h1},t_{h2})\rangle$
SP1.1: $\langle [t_{k1}]\mathcal{R}_{Pr}(\alpha_k), do(x,\alpha_k,t_{k1},t_{k2}), bringAb(x,\mathcal{R}_{Ef}(\alpha_k),t_{k2}) \wedge \boldsymbol{\rho \in \mathcal{R}_{Ef}(\alpha_k)}$ $[t_{j1}]\mathcal{R}_{Pr}(\alpha_j) \wedge \boldsymbol{\rho \in \mathcal{R}_{Pr}(\alpha_j)}, do(x,\alpha_j,t_{j1},t_{j2}), bringAb(x,\mathcal{R}_{Ef}(\alpha_j),t_{j2})$, $achieve(x,g_j) \Rightarrow do(x,\alpha_k,t_{k1},t_{k2}) \wedge do(x,\alpha_j,t_{j1},t_{j2})\rangle$

The activation of a norm is within an interval $[t_{n1},t_{n2}]$ that generally depends on the time of instantiation of the premises $\mathcal{R}_{Prem}(N_i) \cup \mathcal{A}_{Prem}(N_i)$. The norm expires when its premises no longer hold. This formalisation also includes the case where the interval is fixed; i.e. not related to the premises. The norm can be instantiated during the individual planning process and in this case the premises are part of the individual plan. However, norms can be activated by new knowledge about actions and features introduced during the discussion. The conflicts generated are identified by sets of currently obliged actions ($\mathcal{A}^x_O$) and states ($\mathcal{R}^x_O$), currently forbidden actions ($\mathcal{A}^x_F$) and states ($\mathcal{R}^x_F$) defined as:

1. An action $[t_{k1},t_{k2}]\alpha_k$ is in $\mathcal{A}^x_F$ if it conflicts with a norm $N_i \in \Pi^x_N$ that forbids x to perform α_k (or equivalently it obliges x not to perform α_k) within an interval $[t_{n1},t_{n2}]$ overlapping $[t_{k1},t_{k2}]$.
2. A feature $[t_k]\rho_k$ is in $\mathcal{R}^x_F$ if it conflicts with a norm $N_i \in \Pi^x_N$ that forbids x to bring about ρ_k (or equivalently it obliges x not to bring about ρ_k) within an interval $[t_{n1},t_{n2}]$ including t_k. In addition, given a feature $[t_k]\rho_k \in \mathcal{R}^x_F$ when a set of norms forbids the only actions possible to bring about $[t_k]\rho_k$.
3. An action $[t_{k1},t_{k2}]\alpha_k$ is in $\mathcal{A}^x_O$ if a norm $N_i \in \Pi^x_N$ obliges x to perform α_k (or equivalently it forbids x not to perform α_k) within an interval $[t_{n1},t_{n2}]$ overlapping $[t_{k1},t_{k2}]$.
4. A feature $[t_k]\rho_k$ is in $\mathcal{R}^x_O$ if a norm $N_i \in \Pi^x_N$ obliges x to bring about ρ_k (or it forbids x not to bring about ρ_k) within an interval $[t_{n1},t_{n2}]$ including t_k.

In our model we assume that plans of individual agents are norm-consistent. The execution of a plan $\vec{P}^x$ for agent x is norm-consistent when [3]:

1. no actions included in the instantiated plan are among the currently forbidden actions. If $[t_{k1},t_{k2}]\alpha_k \in \mathcal{A}^x_{\vec{P}}$ then $[t_{k1},t_{k2}]\alpha_k \notin \mathcal{A}^x_F$.

Table 2. Arguments Arg_n and Arg_g

CQ2-Is there any norm which forbid or oblige?
ATK2.1: $\langle \mathcal{R}_{Prem}(N_i) \cup \mathcal{A}_{Prem}(N_i), \boldsymbol{F} : do(x, \alpha_k, t_{k1}, t_{k2}) \Rightarrow \neg do(x, \alpha_k, t_{k1}, t_{k2}) \rangle$
ATK2.2: $\langle \mathcal{R}_{Prem}(N_i) \cup \mathcal{A}_{Prem}(N_i), \boldsymbol{F} : bringAb(x, \rho_k, t_k) \Rightarrow \neg bringAb(x, \rho_k, t_k) \rangle$
ATK2.3: $\langle \mathcal{R}_{Prem}(N_i) \cup \mathcal{A}_{Prem}(N_i), \boldsymbol{O} : do(x, \alpha_h, t_{h1}, t_{h2}) \Rightarrow do(x, \alpha_h, t_{h1}, t_{h2}) \rangle$
SP2.2: $\langle \mathcal{R}_{Prem}(N_i) \cup \mathcal{A}_{Prem}(N_i), \boldsymbol{O} : bringAb(x, \rho_h, t_h) \Rightarrow bringAb(x, \rho_h, t_h) \rangle$
CQ3-Is the goal justified?
ATK3.1:$\langle [t_a]\rho_a \wedge \cdots \wedge \neg[t_k]\rho_k \wedge \cdots \wedge [t_q]\rho_q, \neg achieve(x, g_k), \neg bringAb(x, \mathcal{R}_P^x(g_k), t_{end}) \Rightarrow \neg achieve(x, g_k) \rangle$
ATK3.2:$\langle [t_d]\rho_d \wedge \cdots \wedge [t_p]\rho_p, achieve(x, g_k) \wedge \neg achieve(y, g_h), bringAb(x, \mathcal{R}_P^x(g_k), t_{end}) \wedge \neg bringAb(y, \mathcal{R}_P^y(g_h), t_{end}) \Rightarrow \neg achieve(x, g_k) \wedge achieve(y, g_h) \rangle$
SP3.1:$\langle [t_a]\rho_a \wedge \cdots \wedge [t_q]\rho_q, achieve(x, g_k), bringAb(x, \mathcal{R}_P^x(g_k), t_{end}) \Rightarrow achieve(x, g_k) \rangle$

2. no features of the world included in the instantiated plan are among the currently forbidden features. If $[t_k]\rho_k \in \mathcal{R}_P^x$ then $[t_k]\rho_k \notin \mathcal{R}_F^x$.

Conflicts may arise when the proposed adoption of a new action or a new feature of the world in the already prepared individual plan generates inconsistencies. Following these two conditions, our approach considers a norm N_i as an attack for an action $[t_{k1}, t_{k2}]\alpha_k$ or a feature $[t_k]\rho_k$ when these are currently forbidden; i.e. part of $\mathcal{A}_F^x$ or $\mathcal{R}_F^x$. The norm is instead a support if actions or features are currently obliged; i.e. part of $\mathcal{A}_O^x$ or $\mathcal{R}_O^x$. The argument for norms is based on the normative reasoning scheme proposed by Oren et al. [6]. The structure involves the activated norm, its premises and conclusions. For example, given a norm $N_i = \{\mathcal{R}_{Prem}(N_i) \cup \mathcal{A}_{Prem}(N_i) \Rightarrow [t_{n1}, t_{n2}]\ F_x\colon \alpha_k/\rho_k\}$ where $N_i \in \Pi_N^y$, the argument has the following structure:

Arg_n: - Given premises $\mathcal{R}_{Prem}(N_i) \wedge \mathcal{A}_{Prem}(N_i)$,
- a norm $Forbids$ to
- $do(x, \alpha_k, t_{k1}, t_{k2})/bringAb(x, \rho_k, t_k)$
$\Rightarrow$ therefore $\neg do(x, \alpha_k, t_{k1}, t_{k2})/\neg bringAb(x, \rho_k, t_k)$

According to the instantiated norms and the conditions of norm-inconsistency an opponent y can argue against a proponent x following these critical questions:

- **ATK2.1:** Is performing an action currently forbidden?
- **ATK2.2:** Is achieving a feature of the world currently forbidden?

Supports are deduced in the same way following the conditions for maintaining the instantiated plan as norm-consistent. Therefore, the questions are:

- **SP2.1:** Is performing an action currently obliged?
- **SP2.2:** Is achieving a feature of the world currently obliged?

Table 2 presents a formal structure of attacks and supports for Arg_n.

4.3 Arguments for goals

In this section we consider conflicts among team members' goals regarding *CQ3*. Arguing about the adoption of the goal allows parties to understand what they can effectively pursue within a collaborative plan. A member's plan may be incompatible with another member's plan because the parties may have contradictory goals. The activation rule $S(g_k) = \{\rho_a, \ldots, \rho_q \Rightarrow g_k\}$ in Π_S^x indicates that if agent x believes that the world is in a particular state the goal g_k should be achieved. This is used as support for agent x to defend the achievement of goal g_k. A goal $g_k \in \mathcal{G}_P^x$ is achieved at time t_{end}. The set of features brought about by agent x to achieve g_k is $[t_{end}]\mathcal{R}_P^x(g_k)$, although the agent can bring about some features in $t_i < t_{end}$. Argument Arg_g involves the rule $S(g_k)$:

Arg_g: - Observations $[t_a]\rho_a \wedge \cdots \wedge [t_q]\rho_q \in \mathcal{R}_{\vec{P}}^x$
- lead to $achieve(x, g_k)$
- by $bringAb(x, \mathcal{R}_P^x(g_k), t_{end})$
$\Rightarrow$ therefore $achieve(x, g_k)$

The observations follows from the activation rule $S(g_k)$ but with their temporal specification, in order to share precise information about the world with other parties. This argument is formed to support a goal answering the question:

– **SP3.1** Is there any evidence that motivates the goal?

The adoption of the set of actions $\mathcal{A}_P^x(g_k)$ proposed by an agent to a partner is motivated by the intention to achieve a goal. The partner can oppose this decision by arguing that the proponent's view of the world is incorrect and consequently the achievement of the goal is not justified. For example, given the proponent's observations $[t_a]\rho_a \wedge \ldots [t_k]\rho_k \wedge \cdots \wedge [t_q]\rho_q \in \mathcal{R}_{\vec{P}}^x$ that justify the achievement of g_k, in the opponent's view of the world the feature $[t_k]\neg\rho_k \in \mathcal{R}_{\vec{P}}^y$ holds. This is an inconsistency in the state of the world that leads y to argue that x's goal g_k is not motivated and it should be dropped. Moreover, an agent y may argue against x's adoption of the goal g_k by asserting that in y's plan there is another goal g_h which conflicts with the goal g_k. Formally, a feature of the world brought about in order to achieve goal g_k by agent x, $[t_{end}]\rho_k \in [t_{end}]\mathcal{R}_P^x(g_k)$, contradicts the feature $[t_{end}]\neg\rho_k \in [t_{end}]\mathcal{R}_P^y(g_h)$ brought about to achieve g_h by agent y. This argument is based on y's activation rule for goal g_h, $S(g_h) \in \Pi_S^y$, $S(g_h) = \{\rho_d, \ldots, \rho_p \Rightarrow g_h\}$. These attacks can be formed by an opponent y, answering the following critical questions:

– **ATK3.1** Is there any conflicting evidence that does not motivate the goal?
– **ATK3.2** Does the goal conflict with another goal in the individual plan?

Table 2 presents formal attacks and supports for Arg_g.

5 Links Among Arguments

In the previous sections we have defined arguments that an agent can use during the deliberation dialogue. In this section we show how the formalism allows

1. $\boldsymbol{x} \mapsto \boldsymbol{y}$: Arg_I - $\langle [t_1]\rho_1 \wedge [t_1]\rho_2 \wedge [t_1]\rho_9, do(x, \alpha_1, t_1, t_3),$ $bringAb(x, \{\rho_3, \rho_4, \neg\rho_1, \neg\rho_2\}, t_3), achieve(x, g_1), \Rightarrow do(x, \alpha_1, t_1, t_3)\rangle$.
2. $\boldsymbol{y} \mapsto \boldsymbol{x}$: ATK1.1 - $\langle [t_1]\rho_1 \wedge [t_1]\rho_2 \wedge [t_1]\rho_9, do(x, \alpha_1, t_1, t_3), bringAb(x, \{\boldsymbol{\rho_3}, \rho_4, \neg\rho_1,$ $\neg\rho_2\}, t_3), [t_1]\rho_2, \neg do(y, \alpha_3, t_1, t_3), \neg bringAb(x, \{\boldsymbol{\neg\rho_3}, \rho_{14}\}, t_3), \neg achieve(y, g_2)$ $\Rightarrow \neg do(x, \alpha_1, t_1, t_3) \wedge do(y, \alpha_3, t_1, t_3)\rangle$.
3. $\boldsymbol{x} \mapsto \boldsymbol{y}$: SP1.1 - $\langle [t_1]\rho_1 \wedge [t_1]\rho_2 \wedge [t_1]\rho_9, do(x, \alpha_1, t_1, t_3), bringAb(x, \{\rho_3, \boldsymbol{\rho_4}, \neg\rho_1,$ $\neg\rho_2\}, t_3), [t_3]\boldsymbol{\rho_4} \wedge [t_3]\rho_7 \wedge [t_3]\rho_{10}, do(x, \alpha_2, t_3, t_6), bringAb(x, \{\rho_5, \rho_8, \neg\rho_4, \neg\rho_7\}, t_6),$ $achieve(x, g_1) \Rightarrow do(x, \alpha_1, t_1, t_3) \wedge do(x, \alpha_2, t_3, t_6)\rangle$
4. $\boldsymbol{y} \mapsto \boldsymbol{x}$: SP2.1 - $\langle do(y, \alpha_4, t_1, t_3), O : do(y, \alpha_3, t_1, t_3) \Rightarrow do(y, \alpha_3, t_1, t_3)\rangle$
5. $\boldsymbol{x} \mapsto \boldsymbol{y}$: SP3.1 - $\langle [t_0]\rho_{12} \wedge [t_0]\rho_{13}, achieve(x, g_1), bringAb(x, \rho_5, t_6)$ $\Rightarrow achieve(x, g_1)\rangle$
6. $\boldsymbol{y} \mapsto \boldsymbol{x}$: ATK2.2 -$\langle [t_0]\rho_{15}, F : bringAb(x, \rho_5, t_6) \Rightarrow \neg bringAb(x, \rho_5, t_6)\rangle$

Fig. 1. Example of Argumentation-Based Dialogue

agents to proceed in the discussion using various instantiations of these schemes. Since each argument shares new information about actions, states or goals that parties intend to achieve or to perform, an agent is able to counterattack the proposal with arguments about the newly gathered information. Arguments can deal with issues within the same area. However, some attacks and supports can also be formed within different areas according to the new shared information, shifting the discussion from one context to another. For example, the attacks for an argument Arg_p for plan constraints may be other arguments that negate an action with other actions. Moreover, arguments of type Arg_n exploring norms that forbid the execution of an action or the achievement of a state of the world are possible attacks. Arguments Arg_g that deny the possibility to achieve the goal may also be attacks for Arg_p. The same approach can be used for supporting arguments pertaining to different areas. In particular an argument Arg_p can be challenged by the questions CQ1, CQ2, CQ3. An argument Arg_n is challenged by CQ1, CQ2, and an argument Arg_g by CQ2, CQ3.

Example 2. (Based on Ex.1). The dialogue is presented in formal way in Fig.1. Agent x represents a travel agency attempting to prepare a plan for a customer to get to the seaside. x has an individual plan where a customer cm would take a train tr to reach location B from A. The agency x has to be sure that the service is active. In this case x should collaborate with the train company, represented in this example as agent y, communicating the intention to use the train tr for moving a customer cm from A to B at time $[t_1, t_3]$ (action α_1). The discussion commences with agent x informing y its intention to $do(x, \alpha_1, t_1, t_3)$. The first argument of x is presented in Fig.1[1]. Agent y does not agree with x about this choice for different reasons. The first attack proposed by agent y is based on the fact that train tr leaving from A is diverted towards another location E on that period and, therefore, it will not reach B at the scheduled time. The action of diverting is necessary for the train company to secure the part of railway under maintenance. The diversion is represented as follows. Given $\alpha_3 = divert(tr, A, B, E)$, $\rho_{14} = in(tr, E)$, $g_2 = secureRailway$, action $[t_1, t_3]\alpha_3 \in \mathcal{A}^y_{\vec{P}}$,

$R(\alpha_3) = \{[t_1]\rho_2, [t_1, t_3]\alpha_3 \Rightarrow [t_3]\neg\rho_3 \wedge [t_3]\rho_{14}\}$, $\alpha_3 \in \mathcal{A}^y_P(g_2)$. The attack made by agent y is in ATK1.1 form (Fig.1[2]). Agent x supports α_1 with α_2, following the causal link of ρ_4, explaining that the customer needs to reach B in order to catch the bus from B to C with SP1.1 in Fig.1[3]. Agent y can support its position with SP2.1 (Fig.1[4]). In y's knowledge base there is a norm that obliges y to divert the train when there is another action $\alpha_4 = work(A, B)$ representing the ongoing works on the railway between A and B. We represent the action as $[t_1, t_3]\alpha_4 \in \mathcal{A}^y_P$ and the norm $N_2 = \{[t_1, t_3]\alpha_4 \Rightarrow [t_1, t_3]O_y : \alpha_3\}$. Agent x attempts to justify the necessity to achieve its goal with SP3.1 (Fig.1[5]). This last argument allows y to clarify how agent x intends to achieve the goal, by bringing about $\rho_5 = in(cm, C)$. Agent y counterattack this choice with a new norm issued by the local authorities that enforces that travel agencies are not allowed to bring their customers to location C since it is in a protected zone. The attack made by agent y is in ATK2.2 form (Fig.1[6]) given the feature $\rho_{15} = protected(C)$ and the norm $N_3 = \{[t_0]\rho_{15} \Rightarrow [t_0, t_{end}]F_x : \rho_5\}$. At this point x cannot formulate any other argument and, therefore, it withdraws α_1. Agent x needs to re-plan avoiding the train from A to B.

6 Applications of the Argumentation Model

We propose, in this paper, a model or arguments for deliberative dialogue based on argumentation schemes that deals with actions, norms and goals. This model can be employed in a system where agents prepare an individual plan and, then, engage in discussions to identify the best course of action to adopt. This system can be applied in many domains where mechanisms for supporting collaborative decision-making among groups of individual users or organisations are required.

For effective teamwork, team partners need to agree on a common plan of activities to establish duties and responsibilities. For example, a company that collaborates with a team of service providers has to plan a workflow with the providers to obtain the services according to internal requirements and policies. Collaboration among members of a team increases the capabilities of individuals in the completion of a task. However, making effective decisions can be complex since partners may not be completely aware of each other's internal procedures. Incompatible objectives and interests in addition to internal policies increase the complexity of cooperation in decision-making.

Collaboration among partners can be more efficient and effective with the introduction of a system that transfers part of the workload for gathering relevant planning information and decision making to the agents. Agents, employing our model of argumentation, can mimic the human practical reasoning dialogue and work on behalf of their users to prepare a reasonable shared plan. This would allow users to better focus their efforts on more important issues of collaboration.

Moreover, partners could benefit from an intelligent system where agents work with their users to support collaborative decision making. Our model of dialogues can be embedded in systems used by team leaders for planning where agents would suggest effective arguments to their users during the discussion.

Earlier research has provided some evidence for the hypothesis that agent support for policy advice can help planners on the production of more cost/time efficient plans within a team planning context [8]. We believe that the use of our argumentation-based model to support deliberative dialogues would aid team members to reach agreements that maximise the utility of each partner.

A possible scenario where our system can be applied has been introduced in the examples of Sect. 2-5. A travel agency that targets a sustainable tourism may need to collaborate with transportation companies, local authorities and accommodation owners to cooperate in the creation of holiday packages. Geographical distribution can hamper the collaboration for these organisations. Our model can be embedded in a system for the partners to discuss about holiday opportunities. This system can aid in preparing holiday tours considering resources available, booking constraints and individual policies and, also, in planning the activities for finalising the package once the customers have booked. This technology would improve collaboration among the organisations and coordination for an effective delivery. The aim is to prepare an attractive holiday package, suitable for a large range of customers and our system has the potential to facilitate this process.

7 Related Work

Argumentation for deliberative dialogues has been the topic of a number of research efforts over the last few years [1, 7]. Reasoning about what to do in multi-agent planning is facilitated by argumentation because agents can share information about what it is more appropriate to perform before drawing up an agreed plan. In this way, an agent can clarify the reasons for conflicts and the team can better focus the effort when solving the shared problem. In this paper we have presented a first attempt to capture a rich framework for deliberative dialogue motivated by potential plan-constraints and other dependencies as norms and goals. Existing work that explores the integration of norms in practical reasoning focusses primarily on planning for norm-driven agents [3] and only few include norms in an argumentation frameworks [6]. In our model, we consider norms, as well as plan-constraints, in deliberative dialogue since they are both important reasons for a plan adoption.

In previous research, Rahwan and Amgoud [7] have proposed a logic-based approach of argument generation for practical reasoning in which they characterise the problem as the selection among a set of partial plans where justifications are provided for specific courses of action. Atkinson and Bench-Capon [1] proposed an approach based on argumentation schemes, accompanied by a wide set of critical questions that allow agents to present various alternatives to the action under discussion and evaluate the outcomes on the basis of the social values highlighted by the arguments. Belesiotis et al. [2] have explored the use of situation calculus as a language to present arguments about a plan in a multi-agent environment. In this work agents disagree on what to perform because of different initial views of the world.

In this existing research agents agree on a single best plan to perform where tasks are distributed among the team. However, it does not adequately address the requirements of applications where the discussion is focussed on agreeing joint plans where agents aim at fulfilling individual objectives and norms.

8 Conclusions

In this paper we have presented a model of arguments for deliberative dialogue, based on argumentation schemes for discussing about norms, actions and goals in a multi-agent system. Our aim is to construct a coherent model for referring to multi-agent plans, presenting the structure of different arguments based on this model and providing rules that allows agents to formulate arguments during the discussion. These rules allow the agents to shift the discussion from one area to another facilitating exchange of information about a joint plan. In future research we plan to strengthen the model with the use of a well formulated language for plans. Moreover, we will define in more details how the plan evolves during the argumentation-based discussion for evaluating the benefits of this approach.

References

1. Atkinson, K., Bench-Capon, T.: Practical reasoning as presumptive argumentation using action based alternating transition systems. Artificial Intelligence 171(10-15), 855–874 (July 2007)
2. Belesiotis, A., Rovatsos, M., Rahwan, I.: A generative dialogue system for arguing about plans in situation calculus. In: McBurney, P., Rahwan, I., Parsons, S., Maudet, N. (eds.) Argumentation in Multi-Agent Systems, Lecture Notes in Computer Science, vol. 6057, pp. 23–41. Springer Berlin / Heidelberg (2010)
3. Kollingbaum, M.J., Norman, T.J.: Norm adoption and consistency in the NoA agent architecture. In: Dastani, M., Dix, J., El Fallah-Seghrouchni, A. (eds.) Programming Multi-Agent Systems, Lecture Notes in Computer Science, vol. 3067, pp. 169–186. Springer Berlin / Heidelberg (2004)
4. McAllester, D., Rosenblitt, D.: Systematic nonlinear planning. In: Proceeding of the Ninth National Conference on Artificial Intelligence. pp. 634–639 (1991)
5. McBurney, P., Hitchcock, D., Parsons, S.: The eightfold way of deliberation dialogue. International Journal of Intelligent Systems 22(1), 95–132 (January 2007)
6. Oren, N., Luck, M., Miles, S., Norman, T.J.: An argumentation inspired heuristic for resolving normative conflict. In: Proceedings of the Fifth Workshop on Coordination, Organizations, Institutions and Norms in Agent Systems (2008)
7. Rahwan, I., Amgoud, L.: An argumentation-based approach for practical reasoning. In: Maudet, N., Parsons, S., Rahwan, I. (eds.) Argumentation in Multi-Agent Systems, Lecture Notes in Computer Science, vol. 4766, pp. 74–90. Springer Berlin / Heidelberg (2007)
8. Sycara, K., Norman, T.J., Giampapa, J.A., Kollingbaum, M.J., Burnett, C., Masato, D., McCallum, M., Strub, M.H.: Agent support for policy-driven collaborative mission planning. The Computer Journal 53(5), 528–540 (2010)
9. Walton, D.N.: Argumentation schemes for presumptive reasoning. Lawrence Erlbaum Associates (1996)

Bottom-Up Argumentation

Francesca Toni[1] and Paolo Torroni[2]

[1] Department of Computing - Imperial College London
London, UK
ft@imperial.ac.uk
[2] DEIS - University of Bologna
V.le Risorgimento, 2, 40136, Bologna - Italy
paolo.torroni@unibo.it

Abstract. Online social platforms, e-commerce sites and technical fora support the unfolding of informal exchanges, e.g. debates or discussions, that may be topic-driven or serendipitous. We outline a methodology for analysing these exchanges in computational argumentation terms, thus allowing a formal assessement of the dialectical validity of the positions debated in or emerging from the exchanges. Our methodology allows users to be engaged in this formal analysis and the assessement, within a dynamic process where comments, opinions, objections, as well as links connecting them, can all be contributed by users.

1 Introduction

Online social platforms, such as facebook[3], e-commerce sites, such as amazon[4], and technical fora, such as TechSupport Forum [5] support the unfolding of informal exchanges, in the form of debates or discussions, amongst several users. Some of these exchanges may be topic-driven (e.g. is a particular holiday destination worth visiting? Which book by Umberto Eco is best? How can a software bug be fixed?). Other may be serendipitous (e.g. while discussing the recent tsunami in Japan one may end up debating pros and cons of nuclear power stations).

While it is acknowledged (e.g. in [10]) that computational argumentation could benefit these online systems by supporting a formal analysis of the exchanges taking place therein, virtually all of the existing work considering online systems and argumentation focuses on extracting argumentation frameworks of one form or another manually or semi-automatically from these exchanges. For example, Heras et al [10] suggest the use of argument schemes as a way to understand the contributions in these exchanges, while Rahwan et al [11] suggest to map these contributions onto the AIF (Argument Interchange Format) format, again using argument schemes as well as semantic web technology for editing and querying arguments. These works implicitly assume that the extraction of argumentation frameworks is down to "argumentation engineers" external

[3] http://www.facebook.com/
[4] http://www.amazon.com/
[5] http://www.techsupportforum.com/forums/

to/passively engaged in the exchanges, and "fluent" in (one form or another of) computational argumentation.

On the other hand, work in computational argumentation predominantly focuses on determining the dialectical validity of a set of arguments, a single argument, or a claim, supported by arguments, with respect to a given, statically defined argumentation framework. Several notions of dialectical validity have been defined (e.g. see [4, 7, 1]) and several systems, for some or several of these notions, are available (e.g. see [9, 8]).

We propose a methodology linking these two lines of work. Rather than assuming the intervention of "argumentation engineers" observing the exchanges, we envisage that the active participants in the exchange are annotating the exchanges. In order for ordinary (rather than computational-argumentation fluent) users to be engaged in these annotations, we keep them very simple and graphical: annotations indicate that pieces of text in natural language are either *comments* or *opinions*, and *links* can be drawn to indicate source, support or objection. Opinions are expressed about comments, and comments and/or opinions can be linked to links too, very freely and in natural language as in online informal exchanges. We then propose an automated mapping from these annotations to an existing computational argumentation framework, Assumption-Based Argumentation (ABA) [6], paving the way to the automatic computation of the dialectical validity of comments, opinions, and links, and thus topics that these encompass. We envisage that users will add comments, opinions and links dynamically, in the same way as exchanges grow over time in existing online systems.

We term our methodology *bottom-up argumentation* because it takes a grass-root approach to the problem of deploying computational argumentation in online systems:

- the argumentation frameworks are obtained bottom-up starting from the users' comments, opinions and suggested links;
- no top-down intervention of or interpretation by "argumentation engineers" is required;
- our automated translation feeds building blocks of arguments and attacks up to an argumentation system for determining computational validity;
- topics emerge, bottom-up, during the underlying process, possibly serendipitously.

We choose ABA as the underlying computational argumentation framework since it is the simplest system we are aware of that i) is well suited to support practical argumentative reasoning [3], ii) can distinguish arguments, support as well as attack amongst them, iii) can support defeasibility of information as the system evolves over time, iv) is equipped with a variety of well-defined semantics and computational counterparts for assessing dialectical validity.

We will focus in this paper on social networks as these allow for the most free kinds of exchanges, and are thus the most general setting in which to show our methodology.

The paper is organised as follows.



Darwin's natural selection rules supreme.

If you have ever been in GB you must have experienced washing your hands with separate taps. You know what I mean.

The picture shows a tap specimen now inhabiting Imperial College restrooms. You can clearly see a minor, but significant, mutation in the DNA of its ancestors. In particular, with respect to the "hardcore separate taps" variety, which used to live there not so long ago, cold and hot water are still separate, but they seem to have developed a form of symbiosis.

Besides, the population of "hardcore separate taps" (the only tap variety accounted for, until recently) seems to be on its way to extinction. Even in my hotel I couldn't find any.

This is quite impressive, considering that we are only in 2011.

Fig. 1. Initial post on facebook

In section 2 we provide a concrete, motivating example for our methodology, of an exchange in a social network. We also discuss the main motivations for our proposed methodology. In section 3 we provide our basic system of annotations, in the context of the motivating example. In section 4 we give background on ABA. In section 5 we define the automated mapping between exchanges as given in section 3 and ABA, again illustrated for the motivating example. In section 6 we discuss some directions for future work and conclude.

2 Motivation

Let us consider a concrete case[6], where facebook user Paolo Rossi posts the picture and comment shown in Figure 1.

[6] This is a real discussion that took place in facebook. The comments have not been edited. We instead modified the users' names for reasons of privacy. As a disclaimer, this paper does not intend to take any position regarding the opinions present in this illustration.



Fig. 2. Separate Taps discussion: comments (a)

This post does not have a precisely identified subject or purpose. There is a picture showing two separate taps controlling the water flow of a single faucet, from which two separate streams of water flow. The *comment* is intended to be humourous, but it does not say whether separate taps are inconvenient, or antiquate, although that may be implied between the lines. Then, as more facebook



Fig. 3. Separate Taps discussion: comments (b)

users comment on this post (see Figures 2 and 3, where comments are labelled Ci starting from the initial post $C1$), some *opinions* start to emerge between the lines, grounded in the comments, and people start discussing them, to express their agreement and bring additional *support* to comments/opinions of other users, or else to show disagreement and bring up *objections*. For example, the

first three comments seem to agree, directly or indirectly, on the opinion (let us call it $O1$) that "separate taps are common in GB". $C1$ also seems to convey another opinion: "separate taps are antiquate" ($O3$). $O1$ and $O3$ together may support the further opinion that "GB is a backward country" ($O2$), although in a somehow implicit way. These relationships between comments and emerging opinions are of a positive nature, i.e., comments support certain opinions. However, there are also comments representing objections to opinions or to other comments. For example, $C3$ may support the opinion that "separate taps are inconvenient because they freeze/burn hands" ($O4$) whereas a different comment $C11$ supports a different, conflicting opinion, that "separate taps are not inconvenient as the basin solves temperature problems" ($O17$), hence we may read $O17$ as an objection to $O4$. A possible annotation of the facebook exchange in terms of opinions, objections and links is given at the end of the paper. This annotation may be contributed by the users engaged in the exchange or by other users, external to the exchange.

It has been often said that the Web 2.0 is a place for grassroots. Actually, this is exactly what happens here. New contributions and ideas are produced and shared in an exquisitely serendipitous, bottom-up approach. In general, debates in the social Web start with no clear purpose. If the one who posts the first comment has a purpose in mind, he or she does not usually state it. Different is the case of structured debates, or polls in which the objective is clear, for instance choosing one among three possible dates for a meeting. Here instead we are looking at chains of pseudo-random posts, like we find in facebook, in amazon or at the bottom of an online newspaper's article. Sometimes such chains of posts converge to some topic, then again they may totally diverge and focus on some other topic. They may happen to never find a focus.

Despite these features, we can still abstract away and recognize, within these exchanges, *arguments*. But, unlike arguments in the computational argumentation literature, these arguments are not structured or relevant to any predefined topic, opinion or goal. They emerge, *bottom-up*, from the grassroots. Form these arguments, a few mainstream opinions may emerge as the result of many comments, as if in a sort of "natural selection".

In this form of exchange we can identify a "struggle for existence" of arguments. The struggle determines what arguments will be most appreciated, upheld, agreed upon, and influential in the definition of the forthcoming generations of arguments, if we stick to the metaphor. But what are the forces that govern the struggle for existence of arguments in bottom-up argumentation? The rhetoric abilities of participants, their knowledge, their logical and social skills, all contribute greatly to the final result. But, since this is enabled by the presence of a social Web platform, the medium is also a player.

In this paper we outline a methodology for bringing computational argumentation (with its evalutative benefits) into these kinds of unstructured online exchanges while keeping the philosophy and style (simplicity, fun, freedom of expression) of the existing medium for social network. Indeed, we envisage that users can add further annotations, in the form of opinions grounded in/based on

comments, objections, as well as (directed) links connecting them. The opinions are in the same format (free text) as the comments. Links are just graphical.

3 Annotations

We will use the following terminology:

- *comment* stands for a "base-level" user comment, i.e. a comment posted in an online debate by a user; *comment*s will be denoted $C1, C2, \ldots$;
- *opinion* stands for a "meta-level" comment, containing information extracted or digested from part of one or more *comment*s or other *opinion*s, again by a user; *opinion*s will be denoted $O1, O2, \ldots$;
- *link*s are of two types:
  - straight lines directed from one or more starting points (the dotted end of the lines) to a target; these indicate either that the starting points are the source for the information held at the end point, if the starting point is a *comment*, or that the starting points provide support for the end point, if the starting point is not a *comment*; these lines can be seen as expressing a *basedOn* relation in the first case, and a *supportedBy* relation in the second;
  - dotted lines, again directed from one or more starting points (the dotted end of the lines) to a target; these indicate *objection*s from the starting points (typically opinions) to the end point; these lines can be seen as expressing a *objection* relation.

In our motivating example, the *basedOn* relation is used to model that the source of $O1$ is $C1$; the *supportedBy* relation is used to model that $O1, O3$, together, support $O2$; the *objection* relation is used to model that $O17$ disagrees with $O4$. In general, *basedOn*, *supportedBy* and *objection* relations can also hold between a *comment* or *opinion* and another *basedOn*, *supportedBy* or *objection* relation. Indeed, in our motivating example, the *objection* to $O4$ originating from $O17$ is *basedOn* another *comment*, $C11$.

We will see how to map *comment*s, *opinion*s, and *link*s onto a computational argumentation framework. The idea (and expected benefit) is to determine which *opinion*s are acceptable given the current state of the discussion, in relation with other *comment*s/*opinion*s. As the exchange proceeds, different views will emerge and become more or less acceptable.

If a user agrees on some *opinion* supported by some *comments*, we could say that the user "assumes" that such *comments* make sense, unless there are reasons not to do so. Likewise, if such *opinion* is subject to some *objections*, we could say that the user does not "assume" that such *objections* make sense, unless there are reasons to do so. These considerations (as well as the reasons put forward in the introduction) make us believe that Assumption-Based Argumentation [6] is a very natural candidate framework for modeling bottom-up argumentation.

4 Assumption-based argumentation

Assumption-Based Argumentation (ABA) is a general-purpose argumentation framework where arguments and attacks between them are built from *ABA frameworks*, which are tuples $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$ where

- $(\mathcal{L}, \mathcal{R})$ is a *deductive system*, with $\mathcal{L}$ a language and $\mathcal{R}$ a set of inference rules,
- $\mathcal{A} \subseteq \mathcal{L}$, referred to as the set of *assumptions*,
- $\overline{\ }$ is a (total) mapping from $\mathcal{A}$ into $\mathcal{L}$, where $\overline{x}$ is referred to as the *contrary* of x.

In this paper, we assume that inference rules have the syntax $s_0 \leftarrow s_1, \ldots, s_n$ (for $n \geq 0$) where $s_i \in \mathcal{L}$. We refer to $s_1, \ldots, s_n$ as the *premises* and to s_0 as the *head* of the rule. If $n = 0$, we represent a rule simply by its head and we call the rule a *fact*. As in [5], we will restrict attention to *flat* ABA frameworks, such that no assumption occurs in the head of a rule.

Rules may be domain-dependent or not, and some of the premises of rules may be assumptions. These can be used to render the rules defeasible. In this setting, contraries of assumptions can be seen as representing "defeaters".

An *(ABA) argument* in favour of a sentence $c \in \mathcal{L}$ supported by a set of assumptions $A \subseteq \mathcal{A}$ is a proof of c from A and (some of) the rules in $\mathcal{R}$. This proof can be understood as a tree (with root the claim and leaves the assumptions), as in [6], as a backward deduction, as in [5, 7], or as a forward deduction, as in [2], equivalently. For the purposes of this paper, we will use the notation $A \vdash_R c$ to stand for an argument for c supported by A by means of rules $R \subseteq \mathcal{R}$. When the rules can be ignored, we write an argument $A \vdash_R c$ simply as $A \vdash c$.

An argument $A \vdash c$ *attacks* an argument $A' \vdash c'$ if and only if $c = \overline{\alpha}$ for some $\alpha \in A'$.

Several "semantics" for ABA have been defined in terms of sets of assumptions fulfilling a number of conditions. These are expressed in terms of a notion of attack between sets of assumptions, where $A \subseteq \mathcal{A}$ attacks $A' \subseteq \mathcal{A}$ if and only if there is an argument $B \vdash c$, with $B \subseteq A$, attacking and argument $B' \vdash c'$, with $B' \subseteq A'$.

In this paper we will focus on the following notions:

- $A \subseteq \mathcal{A}$ is *conflict-free* if and only if A does not attack itself
- $A \subseteq \mathcal{A}$ is *admissible* if and only if A is conflict-free and attacks every $B \subseteq \mathcal{A}$ that attacks A
- $A \subseteq \mathcal{A}$ is *preferred* if and only if A is (subset) maximal admissible.

Note that these notions can be equivalently expressed in terms of arguments, rather than assumptions, as shown in [7].

Given an ABA framework $\mathcal{F}=\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ } \rangle$ and a (conflict-free or admissible) set of assumptions $A \subseteq \mathcal{A}$ in $\mathcal{F}$, the (conflict-free or admissible) *extension* (respectively) $\mathcal{E}_{\mathcal{F}}(A)$ is the set of all sentences supported by arguments with support a set of assumptions $B \subseteq A$:

$\mathcal{E}_{\mathcal{F}}(A) = \{s \in \mathcal{L} | \exists B \vdash s \text{ with } B \subseteq A\}$.

In the remainder of this section, we will use the following conventions. Uppercase letters denote variables that are implicitly universally quantified. Variables O, C, L are used to represent opinions, comments and links between them, respectively. Variables X, Y are used to represent items that can be either opinions or comments. The rules/assumptions/ contraries are to be intended as schemata, standing for all their ground instances over appropriate universes (for comments, objections and links). Assumptions are always of the form $asm(\_)$, where asm is either α (for assumptions about opinions), χ (for assumptions about comments), or λ/λ^a (for assumptions about straight/dotted links). The contrary of assumption $asm(a)$ is of the form $c\_asm(a)$, for any a, formally: $\overline{asm(a)} = c\_asm(a)$.

5 An ABA mapping for bottom-up argumentation

Domain-dependent Facts and Rules. For each *comment*, the ABA model contains a fact $comment(C)$, where C is the comment's label. In our illustration, we have 14 facts: $comment(c1)$, $comment(c2)$, ..., $comment(c14)$.

For each *opinion*, the ABA model contains a fact $opinion(O)$, where O is the opinion's label. In our illustration, we have 19 opinions: $opinion(o1)$, $opinion(o2)$, ..., $opinion(o19)$.

For each straight link, the ABA model contains a fact $link(L, O, X)$, where L is the link's label (chosen to determine it univocally), X the target item and Y the starting point item. For our example, these links are listed in Table 1.

Table 1. Straight links

link(l_1_1,o1,c1).	*link*(m_2_1_3,o2,o1).	*link*(m_2_1_3,o2,o3).
link(l_2_19,o2,o19).	*link*(l_3_1,o3,c1).	*link*(l_3_2,o3,c2).
link(l_3_3,o3,c3).	*link*(l_4_3,o4,c3).	*link*(l_5_4,o5,c4).
link(l_6_5,o6,c5).	*link*(m_7_6_7,o7,c6).	*link*(m_7_6_7,o7,c7).
link(l_8_8,o8,c8).	*link*(l_8_9,o8,c9).	*link*(l_9_9,o9,c9).
link(l_10_9,o10,c9).	*link*(l_11_9,o11,c9).	*link*(l_12_9,o12,c9).
link(l_13_11,o13,c11).	*link*(l_14_12,o14,c12).	*link*(l_14_18,o14,o18).
link(l_15_9,o15,c9).	*link*(l_15_10,o15,c10).	*link*(l_16_9,o16,c9).
link(l_17_11,o17,c11).	*link*(l_18_12,o18,c12).	*link*(l_19_13,o19,c13).
link(l_19_14,o19,c14).	*link*(l_l_4_17_11,l_4_17,c11).	
link(l_l_11_12_9,l_11_12,c9).	*link*(l_l_16_15_9,l_16_15,c9).	

All these links are from *comment*s to *opinion*s or from *opinion*s to *opinion*s, except for the last three that are from *comment*s to *link*s. For each objection link, the ABA model contains a triplet $alink(L, O, X)$, where L is the link's label, O the attacked *opinion*, and X the supporting *comment* or *opinion*.

Dotted links are listed in Table 2.

Table 2. Dotted links

alink(l_4_17,o4,o17).	*alink*(l_9_10,o9,o10).	*alink*(l_11_12,o11,o12).
alink(l_11_18,o11,o18).	*alink*(l_18_11,o18,o11).	*alink*(l_16_15,o16,o15).
alink(l_17_7,o17,o7).	*alink*(l_17_19,o17,o19).	

Opinions can be supported by comments, or by other opinions, or by both. For each *opinion*, the ABA framework contains:

- one or more rules $basedOn(O) \leftarrow C, \ldots, L, \ldots$ indicating the assumptions underlying *opinion* O in terms of *link*s to *comments*;
- one or more rules $supportedBy(O) \leftarrow O, \ldots, L, \ldots$ indicating the assumptions underlying *opinion* O in terms of links to *opinions*.

For our example, the *basedOn* relations are listed in Table 3 and *supportedBy* relations are listed in Table 4. Note that links from multiple starting points (such as that between $(c6, c7)$ and $o7$) are modeled by a single rule, whereas multiple independent links (such as that between $c1$ and $o3$, or between $c2$ and $o3$) are modeled by multiple rules. Absence of links is modelled by rules with an empty body (facts).

Table 3. *basedOn* relations

basedOn(o1)	← c1,l_1_1.	*basedOn*(o2)	← .
basedOn(o3)	← c1,l_3_1.	*basedOn*(o3)	← c2,l_3_2.
basedOn(o3)	← c3,l_3_3.	*basedOn*(o4)	← c3,l_4_3.
basedOn(o5)	← c4,l_5_4.	*basedOn*(o6)	← c5,l_6_5.
basedOn(o7)	← c6,c7,m_7_6_7.	*basedOn*(o8)	← c8,l_8_8.
basedOn(o8)	← c9,l_8_9.	*basedOn*(o9)	← c9,l_9_9.
basedOn(o10)	← c9,l_10_9.	*basedOn*(o11)	← c9,l_11_9.
basedOn(o12)	← c9,l_12_9.	*basedOn*(o13)	← c11,l_13_11.
basedOn(o14)	← c12,l_14_12.	*basedOn*(o15)	← c9,l_15_9.
basedOn(o15)	← c10,l_15_10.	*basedOn*(o16)	← c9,l_16_9.
basedOn(o17)	← c11,l_17_11.	*basedOn*(o18)	← c12,l_18_12.
basedOn(o19)	← c13,l_19_13.	*basedOn*(o19)	← c14,l_19_14.
basedOn(l_4_17)	← c11,l_l_4_17_11.	*basedOn*(l_11_12)	← c9,l_l_11_12_9.
basedOn(l_16_15)	← c9,l_l_16_15_9.		

Domain-independent Facts and Rules. Domain-independent facts and rules are used as follows.

- We rely on an *opinion* O if we can rely on other *comments* that support O (if any) and/or on *opinions* that support O (if any), and if it is legitimate

Table 4. *supportedBy* relations

supportedBy(o1) ← .	*supportedBy*(o2) ← o1,o3,m_2_1_3.
supportedBy(o2) ← o19,l_2_19.	*supportedBy*(o3) ← .
supportedBy(o4) ← .	*supportedBy*(o5) ← .
supportedBy(o6) ← .	*supportedBy*(o7) ← .
supportedBy(o8) ← .	*supportedBy*(o9) ← .
supportedBy(o10) ← .	*supportedBy*(o11) ← .
supportedBy(o12) ← .	*supportedBy*(o13) ← .
supportedBy(o14) ← o18,l_14_18.	*supportedBy*(o15) ← .
supportedBy(o16) ← .	*supportedBy*(o17) ← .
supportedBy(o18) ← .	*supportedBy*(o19) ← .

to assume O. Therefore the following ABA rule is used, for all *opinions* O:

$$O \leftarrow \mathit{basedOn}(O), \mathit{supportedBy}(O), \alpha(O), \mathit{opinion}(O).$$

The defeasibility of an *opinion* O is modeled by the assumption $\alpha(O)$.

– We rely on a *comment* C if it is legitimate to assume C. Therefore the following ABA rule is used, for all *comments* C:

$$C \leftarrow \chi(C), \mathit{comment}(C).$$

The defeasibility of a *comment* C is modeled by the assumption $\chi(C)$.

– We rely on a *link* L if it is legitimate to assume L. Therefore the following ABA rule is used, for all straight *links* L:

$$C \leftarrow \lambda(C), \mathit{link}(L, \_, \_).$$

The defeasibility of a *link* L is modeled by the assumption $\lambda(L)$.

– We rely on an dotted *link* L if it is legitimate to assume the *link*, L and the attacker. The following rule is then used:

$$c\_\alpha(X) \leftarrow Y, \lambda^a(L), \mathit{alink}(L, X, Y).$$

The defeasibility of a dotted *link* L is modeled by the assumption $\lambda^a(L)$.

– All *opinions*, *comments*, *links* are in principle legitimate. Therefore possible assumptions are $\alpha(O), \chi(C), \lambda(L), \lambda^a(L')$ for all O, C, L, L' in our universe of symbols such that *opinion*(O), *comment*(C), *link*(L), *alink*(L').

– Finally, the following contraries are given:

$$\overline{\alpha(O)} = c\_\alpha(O). \quad \overline{\chi(C)} = c\_\chi(C). \quad \overline{\lambda(L)} = c\_\lambda(L). \quad \overline{\lambda^a(L)} = c\_\lambda^a(L).$$

Note that this translation from an annotated exchange of views on the social Web and ABA can be performed automatically. The resulting ABA framework can then be fed into an ABA system, such as CaSAPI [9], to determine which items (opinions, links etc) can be accepted dialectically.

6 Conclusions

We have outlined a generic methodology to benefit exchanges of views in social networks (but also e-commenrce systems or technical fora) by deploying computational argumentation. We have taken the view to modify only minimally the existing style for social networks, and allow users to unearth opinions and links. We have supported our proposal by means of a concrete illustration on top of facebook.

There are several directions for future work. We mention just a few here.

We have ignored the possibility of feedback by users, e.g. using the Like button in facebook. These need to be incorporated within our methodology.

We have introduced a separation between "base-level" (the comments as in existing social net sites) and a "meta-level" (our opinions, links etc). We envisage that these will need to blend eventually, and that, for example, opinions may feed back into comments.

We envisage to use ABA as the underlying mechanism for computational argumentation. A novel bottom-up tool for computing extensions will be required for ABA.

We have glossed over the choice of argumentaiton semantics: experimental psychology may be able to provide us with hints as to which semantics is the most suited.

References

1. Besnard, P., Hunter, A.: Elements of Argumentation. MIT Press (2008)
2. Bondarenko, A., Dung, P., Kowalski, R., Toni, F.: An abstract, argumentation-theoretic approach to default reasoning. Artif. Intell. 93(1-2), 63–101 (1997)
3. Dung, P.M., Toni, F., Mancarella, P.: Some design guidelines for practical argumentation systems. In: Baroni, P., Cerutti, F., Giacomin, M., Simari, G. (eds.) Proceedings of the Third International Conference on Computational Models of Argument (COMMA'10). vol. 216, pp. 183–194. IOS Press (2010)
4. Dung, P.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell. 77, 321–357 (1995)
5. Dung, P., Kowalski, R., Toni, F.: Dialectic proof procedures for assumption-based, admissible argume ntation. Artif. Intell. 170, 114–159 (2006)
6. Dung, P., Kowalski, R., Toni, F.: Assumption-based argumentation. In: Rahwan, I., Simari, G. (eds.) Argumentation in AI, pp. 199–218. Springer (2009)
7. Dung, P., Mancarella, P., Toni, F.: Computing ideal sceptical argumentation. Artif. Intell. 171(10–15), 642–674 (2007)
8. Egly, U., Gaggl, S.A., Woltran, S.: ASPARTIX: Implementing argumentation frameworks using answer-set programming. In: de la Banda, M.G., Pontelli, E. (eds.) Logic Programming, 24th International Conference, ICLP 2008. Lecture Notes in Computer Science, vol. 5366, pp. 734–738. Springer (2008)
9. Gaertner, D., Toni, F.: On computing arguments and attacks in assumption-based argumentation. IEEE Intelligent Systems, Special Issue on Argumentation Technology 22(6), 24–33 (November/December 2007)

10. Heras, S., Atkinson, K., Botti, V., Grasso, F., Julian, V., McBurney, P.: How argumentation can enhance dialogues in social networks. In: Baroni, P., Cerutti, F., Giacomin, M., Simari, G. (eds.) Proceedings of the Third International Conference on Computational Models of Argument (COMMA'10). vol. 216, pp. 267–274. IOS Press (2010)
11. Rahwan, I., Zablith, F., Reed, C.: Laying the foundations for a world wide argument web. Artificial Intelligence 171, 897–921 (July 2007)

Darwin's natural selection rules supreme.

If you have ever been in GB you must have experienced washing your hands with separate taps. You know what I mean.

The picture shows a tap specimen now inhabiting Imperial College restrooms. You can clearly see a minor, but significant, mutation in the DNA of its ancestors. In particular, with respect to the "hardcore separate taps" variety, which used to live there not so long ago, cold and hot water are still separate, but they seem to have developed a form of symbiosis.

Besides, the population of "hardcore separate taps" (the only tap variety accounted for, until recently) seems to be on its way to extinction. Even in my hotel I couldn't find any.

This is quite impressive, considering that we are only in 2011.

Added 14 March · Like · Comment

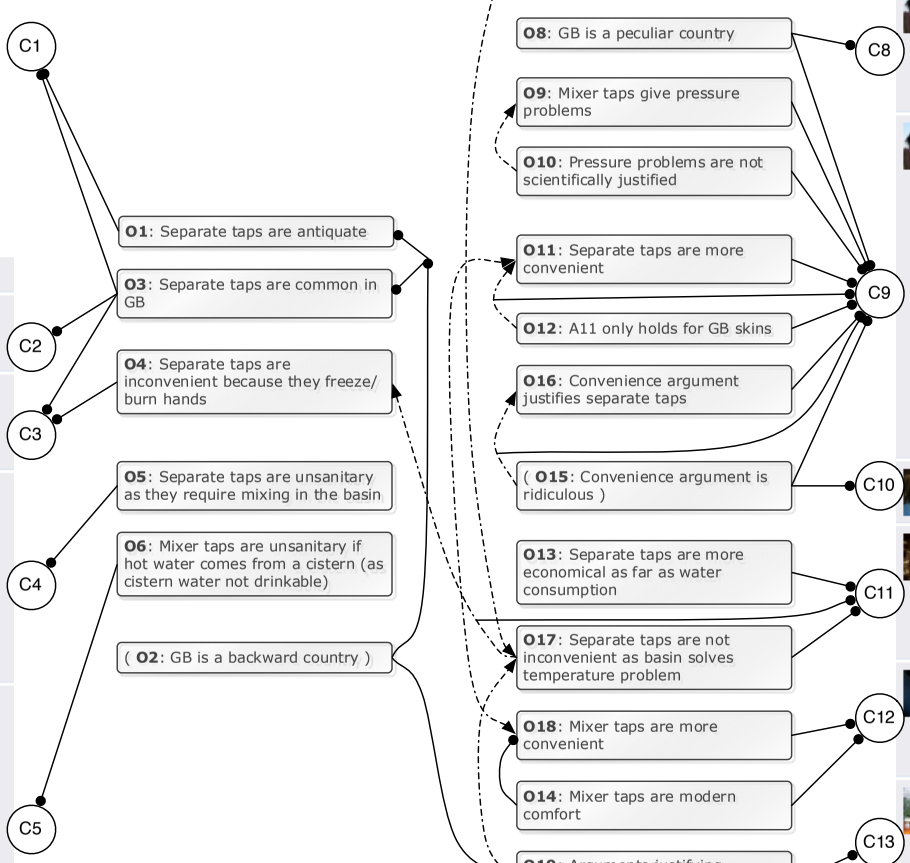
Leyla Gencer, Benjamin Franklin, Enrico Fermi and 3 others like this.

Isabella Rossellini this is what my kitchen tap used to be only up until 5 years ago :-)
Monday at 11:10 · Unlike · 1 person

Benjamin Franklin I remember not knowing whether I freeze or burn my hands under the tap. Took me a while to realise that there were 2 seperate streams coming out of the tap...
Monday at 11:54 · Unlike · 1 person

Robert Axelrod i was also fascinated by the "hardcore separate taps" design until someone explained to me that brits regarded a "mixer" (miscelatore) needed to produce a single stream "unsanitary" and a "health hazard". after all, we know that mixing the hot and cold sources in the tapped basin to produce a tolerable temperature is absolutely sanitary! just give them another 10-20 more generations (with lots of mutations) and the two distinct streams will merge into a single one.
Monday at 12:58 · Unlike · 2 people

Isabella Rossellini
I actually think the health problem was a real issue: the water for the hot tap used to come from a cistern, while the water for the cold tap comes directly from the mains, so the cold water is suitable for drinking, the water coming from the hot tap is not, that's why mixing was not advisable. Nowadays I think both taps get water from the mains, though for older homes this might not be the case (and therefore the two streams of water in the picture are a solution to the problem, if you want to drink tap water you can safely just turn the cold tap on, safe in the knowledge water won't be 'contaminated'
Monday at 13:22 · Unlike · 1 person



Audrey Hepburn The real challenge is to manage to take shower in a bath tube with only 2 separate taps :D
Monday at 13:32 · Unlike · 1 person

Maria Grazia Cucinotta .. and one in each end of the tube! that's my "worst experience" :)
Monday at 16:19 · Unlike · 1 person

Paolo Rossi It looks like separate taps, hardcore or not, are in the top-10 British peculiarities and subject o neverending debate :-) there is some history involving Churchill here :-) http://blogs.warwick.ac.uk/mhillebrandt/entry/britis h_peculiarities_i/
Monday at 16:29 · Like

Paolo Rossi
Some time ago I collected the possible arguments pro/con separate taps.
The "health hazard" argument is probably the most popular one.

There is also the "pressure problem" argument, by which, if you join together hot and cold water, you will cause water to flow in unpredictable directions in your plumbing system. It's a physical phenomenon that is only effective in the UK.

But my favorite argument is the "convenience" argument, by which hardcore separate taps are actually MORE convenient than rest-of-the-world taps.
That has to do with the left/right asymmetric sensitivity to temperature of British skin.
Monday at 17:01 · Like

Scott Adams absolutely :)
Monday at 17:34 · Unlike · 1 person

Tony Blair There is nothing wrong with having two taps. You simply put the stopper in the sink, mix the water to the correct temperature and wash your hands. It's far more economical than washing hands under a combined flow. There is, of course, the differences in source.
Monday at 17:48 · Like · 1 person

Jeremy Irons I thought a mixer tap in the bathroom was a weird gimmick until I lived in Germany, then I got addicted to it. Now I miss having a single lever to control flow and temperature. Space age stuff.
Yesterday at 11:13 · Unlike · 1 person

Diana Spencer My parents defend British separate taps like they were the last bastion of sanity in a crazy technological world. We have had many heated arguments about this... I'm getting angry just thinking about it!!
13 hours ago · Unlike · 1 person

Paolo Rossi :-)))
13 minutes ago · Like

An Argumentation Framework for Qualitative Multi-Criteria Preferences

Wietske Visser, Koen V. Hindriks and Catholijn M. Jonker

Man Machine Interaction Group, Delft University of Technology, The Netherlands
Wietske.Visser@tudelft.nl, K.V.Hindriks@tudelft.nl, C.M.Jonker@tudelft.nl

Abstract. Preferences between different alternatives (products, decisions, agreements etc.) are often based on multiple criteria. Qualitative Preference Systems (QPS) is a formal framework for the representation of qualitative multi-criteria preferences in which a criterion's preference is defined based on the values of attributes or by combining multiple subcriteria in a cardinality-based or lexicographic way. In this paper we present a language and reasoning mechanism to represent and reason about such qualitative multi-criteria preferences. We take an argumentation-based approach and show that the presented argumentation framework correctly models a QPS. Then we extend this argumentation framework in such a way that it can derive missing information from background knowledge, which makes it more flexible in case of incomplete specifications.

1 Introduction

In the context of practical reasoning, such as decision making and negotiation, preferences between the available alternatives play a key role. A system supporting a human user in such tasks should therefore have a representation of that user's preferences. In this paper we present an argumentation framework to represent and reason with *qualitative*, *multi-criteria* preferences. Preferences are modelled in a qualitative way because it is hard for humans to give exact numeric utilities. We use multiple criteria because it is a very natural thing to compare two alternatives on several criteria and base an overall preference on those comparisons. Criteria thus represent the *underlying interests*, or *reasons* for preferences. Moreover, the outcome space may be so large that it is infeasible to specify preference between outcomes directly.

We briefly present a framework for representing qualitative multi-criteria preferences, called Qualitative Preference Systems. In this framework, preferences between outcomes are determined by combining multiple criteria based on cardinality and lexicographic ordering. Ultimately, the criteria are based on preferences between the values of relevant variables. QPS is a framework that provides a formal definition of qualitative multi-criteria preferences. The aim of this paper is to provide a *language* and *reasoning mechanism* to reason about such qualitative preference systems. In addition, we provide the means of deriving information by default from background knowledge, which is useful when e.g. the outcomes are incompletely specified.

The approach we take is argumentation-based. Argumentation is a kind of defeasible reasoning, which allows for reasoning with incomplete information in a commonsense way, about things that are normally the case. Moreover, argumentation is a natural

way of reasoning for humans. As such, it is suitable for explaining the reasoning of a system to a human user. Finally, argumentation can be used in a persuasion dialogue, for example when multiple agents with different preferences have to agree on a common action.

Note that the argumentation framework presented here is *not* a preference-based argumentation framework (PAF) ins the sense of [1]. In a PAF, preference between arguments are used to determine the success of an attack between them. A similar approach, that considers preferences between rules in the logical language, has been taken in the specific context of decision making [2]. In contrast, the framework presented here aims to reason about preferences between objects outside of the argumentation framework ('outcomes') as opposed to preferences between arguments or logical rules.

The outline of the paper is as follows. In Section 2, we briefly recall qualitative preference systems. Section 3 presents the argumentation framework that provides the means to reason about a QPS. In Section 4 we extend the argumentation framework with background knowledge and the means to derive information by default. Finally, Section 5 concludes the paper.

2 Qualitative Preference Systems

In this section we briefly present qualitative preference systems. The main aim of a QPS is to determine preferences between *outcomes* (or *alternatives*). An outcome is represented as an assignment of values to a set of relevant variables. Every variable has its own domain of possible values. Constraints on the assignments of values to variables are expressed in a knowledge base. Outcomes are defined as variable assignments that respect the constraints in the knowledge base.

The preferences between outcomes are based on multiple *criteria*. Every criterion can be seen as a *reason* for preference, or as a preference from one particular *perspective*. A distinction is made between simple and compound criteria. Simple criteria are based on a single variable. Multiple (simple) criteria can be combined in order to determine an overall preference. In a QPS, this is done with compound criteria. There are two kinds of compound criteria: cardinality criteria and lexicographic criteria. The subcriteria of a cardinality criterion all have equal importance, and preference is determined by counting the number of subcriteria that support it. In a lexicographic criterion, the subcriteria are ordered by priority and preference is determined by the most important subcriteria.

Definition 1. (Qualitative preference system) A *qualitative preference system (QPS)* is a tuple $\langle Var, Dom, K, \Omega, \mathcal{C} \rangle$. Var is a finite set of *variables*. Every variable $X \in Var$ has a domain $Dom(X)$ of possible values. K is a set of constraints on the assignments of values to the variables in Var. Ω is the set of all outcomes. An *outcome* α is an assignment of a value $x \in Dom(X)$ to every variable $X \in Var$, such that no constraints in K are violated. α_X denotes the value of variable X in outcome α. $\mathcal{C} = \mathcal{C}_s \cup \mathcal{C}_c \cup \mathcal{C}_l$ is a set of criteria, where $\mathcal{C}_s$ contains simple criteria, $\mathcal{C}_c$ contains cardinality criteria and $\mathcal{C}_l$ contains lexicographic criteria. Weak preference between outcomes by a criterion c is denoted by the relation $\succeq_c$. $\succ_c$ denotes the strict subrelation, $\approx_c$ the indifference subrelation.

Definition 2. (Simple criterion) A *simple criterion* c is a tuple $\langle X_c, \geq_c \rangle$, where $X_c \in Var$ is a variable, and $\geq_c$, a preference relation on the possible values of X_c, is a preorder on $Dom(X_c)$. A simple criterion $c = \langle X_c, \geq_c \rangle$ *weakly prefers* an outcome α over an outcome β, denoted $\alpha \succeq_c \beta$, if $\alpha_{X_c} \geq_c \beta_{X_c}$.

Definition 3. (Cardinality criterion) A *cardinality criterion* c is a tuple $\langle C_c \rangle$ where C_c is a nonempty set of criteria (the *subcriteria* of c). A cardinality criterion $c = \langle C_c \rangle$ *weakly prefers* an outcome α over an outcome β, denoted $\alpha \succeq_c \beta$, iff $|\{s \in C_c \mid \alpha \succ_s \beta\}| \geq |\{s \in C_c \mid \alpha \not\succeq_s \beta\}|$.

Definition 4. (Lexicographic criterion) A *lexicographic criterion* c is a tuple $\langle C_c, \rhd_c \rangle$, where C_c is a nonempty set of criteria (the *subcriteria* of c) and $\rhd_c$, a *priority relation* among subcriteria, is a strict partial order (a transitive and asymmetric relation) on C_c. A lexicographic criterion $c = \langle C_c, \trianglerighteq_c \rangle$ *weakly prefers* an outcome α over an outcome β, denoted $\alpha \succeq_c \beta$, if $\forall s \in C_c(\alpha \succeq_s \beta \vee \exists s' \in C_c(\alpha \succ_{s'} \beta \wedge s' \rhd_c s))$.

3 Argumentation Framework

In this section we present an argumentation framework for reasoning about qualitative multi-criteria preferences as defined in qualitative preference systems. The AF provides the logical language to represent facts about outcomes, criteria and preferences, and the means to construct arguments that infer preferences from certain input.

3.1 Abstract Argumentation Framework

Our argumentation framework is a concrete instantiation of an abstract argumentation framework as defined by Dung [3]. To define which arguments are justified, we use Dung's preferred semantics.

Definition 5. (Abstract argumentation framework) An *abstract argumentation framework (AF)* is a pair $\langle \mathcal{A}, \rightarrow \rangle$ where $\mathcal{A}$ is a set of *arguments* and $\rightarrow$ is a *defeat* relation among those arguments.

Definition 6. (Preferred semantics) A *preferred extension* of an AF $\langle \mathcal{A}, \rightarrow \rangle$ is a maximal (w.r.t. $\subseteq$) set $S \subseteq \mathcal{A}$ such that: $\forall A, B \in S : A \not\rightarrow B$ and $\forall A \in S$: if $\exists B \in \mathcal{A} : B \rightarrow A$ then $\exists C \in S : C \rightarrow B$. An argument is credulously (resp. sceptically) *justified* w.r.t. preferred semantics if it is in some (resp. all) preferred extension(s). An argument is overruled if it is not in any extension. We also say that a formula is justified (resp. overruled) iff it is the conclusion of a justified (resp. overruled) argument.

An abstract AF can be instantiated by specifying the *structure of arguments* and the *nature of the defeat relation*. Prakken [4] presents such an instantiation that is itself still abstract: his *argumentation systems* define arguments as inference trees formed by applying inference rules and specify three kinds of defeat. We take the instantiation of an argumentation framework one step further and also define the *logical language* and the specific *inference schemes* that are used.

3.2 Arguments

Arguments are built from formulas of a logical language, that are chained together using inference steps. Every inference step consists of premises and a conclusion. Inferences can be chained by using the conclusion of one inference step as a premise in the following step. Thus a tree of chained inferences is created, which we use as the formal definition of an argument (cf. e.g. [5, 4]).

Definition 7. (Argument) An *argument* is a tree, where the nodes are inferences, and an inference can be connected to a parent node if its conclusion is a premise of that node. Leaf nodes only have a conclusion (a formula from the knowledge base), and no premises. A subtree of an argument is also called a *subargument*. `inf` returns the last inference of an argument (the root node), and `conc` returns the conclusion of an argument, which is the same as the conclusion of the last inference.

3.3 Defeat

We define two different kinds of defeat: rebuttal and undercut (note that, unlike e.g. [4], in the current framework there is no distinction between *attack* and *defeat*). An argument *rebuts* another argument if its conclusion contradicts a conclusion of the other argument. Which conclusions contradict each other is defined below after the language is introduced. Defeat by rebuttal is mutual. The term undercut is used in different ways in the literature; we use it for the same concept as e.g. [4]. An *undercutter* is an argument for the inapplicability of an inference step made in another argument. Hence, it is a kind of meta-reasoning (the conlusion of an undercutting argument is not part of the object language). Undercut works only one way. Defeat is defined recursively, which means that rebuttal can attack an argument on all its premises and (intermediate) conclusions, and undercut can attack it on all its inferences.

Definition 8. (Defeat) An argument A *defeats* an argument B ($A \to B$) if $\texttt{conc}(A)$ and $\texttt{conc}(B)$ are contradictory (*rebuttal*), or $\texttt{conc}(A) =$‘$\texttt{inf}(B)$ is inapplicable’ (*undercut*), or A defeats a subargument of B.

3.4 Language

The logical language provides the means to express statements about a the elements of a QPS. For a given QPS $S = \langle Var, Dom, K, \Omega, \mathcal{C} \rangle$, the *domain of discourse* is $D = Var \cup \bigcup_{X \in Var} Dom(X) \cup \Omega \cup \mathcal{C}$, i.e. variables and their possible values, outcomes and criteria.

We make a distinction between an *input* and *full* language. A knowledge base, which is the input for an argumentation framework, is specified in the input language. The input language allows us to express facts about the outcomes that are considered and details about the criteria that are used. With the full language we can also express preferences. Such statements can be *derived* from a knowledge base with the inference rules that will be introduced in the next section.

Basic expressions of the language (*atoms*) are built from predicates and terms. Let C be a set of *constants*. $i : C \mapsto D$ is an *interpretation function* that assigns an element

predicate	interpretation
`val(`o`,`x`,`y`)`	$i(o)_{i(x)} = i(y)$ where $i(o) \in \Omega$, $i(x) \in Var$, $i(y) \in Dom(i(x))$
	'the value of variable x in outcome o is y'
`sc(`c`,`x`)`	$i(c) \in \mathcal{C}_s, X_{i(c)} = i(x)$ where $i(x) \in Var$,
	'c is a simple criterion on variable x'
`valpref(`c`,`y_1`,`y_2`)`	$i(y_1) \geq_{i(c)} i(y_2)$ where $i(c) \in \mathcal{C}_s$, $i(y_1), i(y_2) \in Dom(X_{i(c)})$
	'simple criterion c weakly prefers value y_1 over value y_2'
`cc(`c`)`	$i(c) \in \mathcal{C}_c$
	'c is a cardinality criterion'
`lc(`c`)`	$i(c) \in \mathcal{C}_l$
	'c is a lexicographic criterion'
`sub(`c`,`c_1`)`	$i(c_1) \in C_{i(c)}$ where $i(c) \in \mathcal{C}_c \cup \mathcal{C}_l$, $i(c_1) \in \mathcal{C}$
	'c_1 is a subcriterion of criterion c'
`prior(`c`,`c_1`,`c_2`)`	$i(c_1) \rhd_{i(c)} i(c_2)$ where $i(c) \in \mathcal{C}_l$, $i(c_1), i(c_2) \in \mathcal{C}$
	'subcriterion c_1 has higher priority than subcriterion c_2 according to lexicographic criterion c'

Table 1. The predicates in $\mathcal{P}_{in}$ and their interpretation

from the domain of discourse to every constant in C. There are two sets of *predicates*. $\mathcal{P}_{in}$ contains predicates that can be used in the input language. $\mathcal{P}_{out}$ contains predicates that cannot be used in the input language and can only be derived. The predicates in $\mathcal{P}_{in}$ and $\mathcal{P}_{out}$ and their interpretation are in Table 1 and 2.

Formulas of the input language are just atoms of the input language. Formulas of the full language are atoms (A) or weakly negated atoms ($\sim A$). Weak negation is negation as failure: $\sim A$ is justified if A is not. Strong negation is not needed to model qualitative preference systems, but it will be added in the extended version of the AF presented in Section 4 in order to reason with background knowledge.

Definition 9. (Language) The input language is defined as follows.
$atom_{in}$ ::= $p(t_1, \ldots, t_n)$ where p is an n-ary predicate $\in \mathcal{P}_{in}$
$literal_{in}$ ::= $atom_{in}$
$formula_{in}$ ::= $literal_{in}$

The full language is defined as follows.
$atom_{out}$ ::= $p(t_1, \ldots, t_n)$ where p is an n-ary predicate $\in \mathcal{P}_{out}$
$literal$ ::= $literal_{in} \mid atom_{out}$
$formula$ ::= $literal \mid \sim literal$

Contradictory formulas Two arguments rebut each other if their conclusions are contradictory. There are two ways in which two formulas can be contradictory.

- The formulas specify different values for the same variable in the same outcome: `val(`o`,`x`,`y`)` and `val(`o`,`x`,`y'`)` contradict each other if $y \neq y'$.
- `prior(`c`,`c_1`,`c_2`)` and `prior(`c`,`c_2`,`c_1`)` contradict each other, since priority is asymmetric.

predicate	interpretation	
`pref(`c,o_1,o_2`)`	$i(o_1) \succeq_{i(c)} i(o_2)$	where $i(c) \in \mathcal{C}, i(o_1), i(o_2) \in \Omega$
	'criterion c weakly prefers outcome o_1 over outcome o_2'	
`spref(`c,o_1,o_2`)`	$i(o_1) \succ_{i(c)} i(o_2)$	where $i(c) \in \mathcal{C}, i(o_1), i(o_2) \in \Omega$
	'criterion c strictly prefers outcome o_1 over outcome o_2'	
`epref(`c,o_1,o_2`)`	$i(o_1) \approx_{i(c)} i(o_2)$	where $i(c) \in \mathcal{C}, i(o_1), i(o_2) \in \Omega$
	'criterion c equally prefers outcome o_1 and outcome o_2'	
`sp(`c,o_1,o_2,n`)`	$\lvert\{s \in C_{i(c)} \mid i(o_1) \succ_s i(o_2)\}\rvert = n$	where $i(c) \in \mathcal{C}_c, i(o_1), i(o_2) \in \Omega$
	'there are n subcriteria of cardinality criterion c	
	that strictly prefer outcome o_1 over outcome o_2'	
`nwp(`c,o_1,o_2,n`)`	$\lvert\{s \in C_{i(c)} \mid i(o_1) \not\succeq_s i(o_2)\}\rvert = n$	where $i(c) \in \mathcal{C}_c, i(o_1), i(o_2) \in \Omega$
	'there are n subcriteria of cardinality criterion c	
	that do not weakly prefer outcome o_1 over outcome o_2'	

Table 2. The predicates in $\mathcal{P}_{out}$ and their interpretation

Two other candidates for contradiction are not modelled as such because they are handled in a different way.

One might argue that φ and $\sim\varphi$ are contradictory, and hence arguments concluding them should rebut each other. However, the status of these conclusions is not equal. φ has to be derived and is grounded in facts in the knowledge base. $\sim\varphi$ on the other hand is an assumption that can be made in the absence of evidence to the contrary. φ is such evidence to the contrary, and that is why an argument concluding φ undercuts the inference of $\sim\varphi$ instead of rebutting the conclusion (see the inference schemes for weak negation and its undercutter below).

Incompatible preference statements, such as e.g. `spref(c,o1,o2)` and `epref(c,o1,o2)` will resolve because `epref(c,o1,o2)` can only be derived if `pref(c,o2,o1)`, in which case the `~pref(c,o2,o1)` premise needed to derive `spref(c,o1,o2)` will be undercut. Hence to have such arguments rebut each other would be superfluous.

Input knowledge base An input knowledge base is a set of formulas of the input language. A knowledge base *KB corresponds to* a QPS $S = \langle Var, Dom, K, \Omega, \mathcal{C}\rangle$ if the following condition holds: a formula φ is in *KB* iff its interpretation holds in *S*. Note that a knowledge base corresponding to a QPS is conflict-free, i.e. does not contain contradictory formulas.

Example 1. *We will use a running example throughout the paper to illustrate the details of the argumentation framework. Anne is planning to go on holiday with a friend. Anne's overall preference is based on three simple criteria:* `c1`*: that someone (she or the accompanying friend) speaks the language (*`sl`*),* `c2`*: that it is sunny (*`su`*) and* `c3`*: that she has not been there before (*`bb`*).* `c1` *and* `c2` *have equal priority, so they are aggregated in a cardinality criterion* `c4`*.* `c3` *and* `c4` *are combined in a lexicographic criterion* `c5` *where* `c3` *has higher priority than* `c4`*. This information can be represented in the following knowledge base.*

Facts about two of the possible outcomes:

```
val(o1,sl,true)     val(o1,su,true)     val(o1,bb,true)
val(o2,sl,false)    val(o2,su,true)     val(o2,bb,false)
```

Information about the preferences:

```
lc(c5)          cc(c4)          sc(c1,sl)     valpref(c1,true,false)
sub(c5,c3)      sub(c4,c1)      sc(c2,su)     valpref(c2,true,false)
sub(c5,c4)      sub(c4,c2)      sc(c3,bb)     valpref(c3,false,true)
prior(c5,c3,c4)
```

3.5 Inference rules

In this section we present the inference rules that are used in the argumentation framework to build arguments.

Weak negation The following two inference rules make sure that (i) a weakly negated formula can always be derived, but (ii) this inference will be undercut if the formula itself can be derived. So $\sim\varphi$ is sceptically justified iff φ is overruled.

$$\frac{}{\sim\varphi}\ asm(\sim\varphi) \qquad\qquad \frac{\varphi}{asm(\sim\varphi)\text{ is inapplicable}}\ asm(\sim\varphi)uc$$

Strict and equal preference The following inference schemes are used to derive strict and equal preference from weak preference according to the common definitions.

$$\frac{\texttt{pref}(c,o_1,o_2)\quad \sim\texttt{pref}(c,o_2,o_1)}{\texttt{spref}(c,o_1,o_2)} \qquad\qquad \frac{\texttt{pref}(c,o_1,o_2)\quad \texttt{pref}(c,o_2,o_1)}{\texttt{epref}(c,o_1,o_2)}$$

Preference by a simple criterion The following inference rule concludes that a simple criterion prefers one outcome over another if, for the variable that it is based on, it prefers the value of the first outcome over the value of the second. This is exactly the definition of preference by a simple criterion in a QPS.

$$\frac{\texttt{sc}(c,x)\quad \texttt{val}(o_1,x,y_1)\quad \texttt{val}(o_1,x,y_2)\quad \texttt{valpref}(c,y_1,y_2)}{\texttt{pref}(c,o_1,o_2)}$$

Example 2. *The following argument infers that simple criterion* `c1` *prefers* `o1` *over* `o2`. *Similar arguments can be constructed for* `c2` *and* `c3`.

$$\frac{\texttt{sc(c1,sl)}\quad \texttt{val(o1,sl,true)}\quad \texttt{val(o2,sl,false)}\quad \texttt{valpref(c1,true,false)}}{\texttt{pref(c1,o1,o2)}}$$

Preference by a cardinality criterion The next inference scheme derives preference by a cardinality criterion according to its definition in a QPS: an outcome o_1 is weakly preferred over an outcome o_2 if there are at least as many subcriteria that strictly prefer o_1 over o_2 as subcriteria that do not weakly prefer o_1 over o_2.

$$\frac{\texttt{cc}(c)\quad \texttt{sp}(c,o_1,o_2,l)\quad \texttt{nwp}(c,o_1,o_2,m)\quad l\geq m}{\texttt{pref}(c,o_1,o_2)}$$

Preference by a cardinality criterion is based on (i) the number of subcriteria that strictly prefer one outcome over the other, and (ii) the number of subcriteria that do not weakly prefer one outcome over the other. The following inference rules provide the required counting mechanism.

The next inference rules conclude that there are n subcriteria of c that strictly prefer o_1 over o_2, resp. that there are n subcriteria of c that do not weakly prefer o_1 over o_2.

$$\frac{\texttt{spref}(c_1,o_1,o_2) \quad \ldots \quad \texttt{spref}(c_n,o_1,o_2) \quad \texttt{sub}(c,c_1) \quad \ldots \quad \texttt{sub}(c,c_n)}{\texttt{sp}(c,o_1,o_2,n)}\ SP(c,o_1,o_2,n)$$

$$\frac{\sim\texttt{pref}(c_1,o_1,o_2) \quad \ldots \quad \sim\texttt{pref}(c_n,o_1,o_2) \quad \texttt{sub}(c,c_1) \quad \ldots \quad \texttt{sub}(c,c_n)}{\texttt{nwp}(c,o_1,o_2,n)}\ NWP(c,o_1,o_2,n)$$

If there are no subcriteria of c that strictly prefer o_1 over o_2, resp. that do not weakly prefer o_1 over o_2, no premises are needed to infer this.

$$\frac{}{\texttt{sp}(c,o_1,o_2,0)}\ SP(c,o_1,o_2,0) \qquad \frac{}{\texttt{nwp}(c,o_1,o_2,0)}\ NWP(c,o_1,o_2,0)$$

With these inference schemes, it is possible to derive a formula $\texttt{sp}(c,o_1,o_2,n)$ for any n between 0 and the actual number of subcriteria of c that strictly prefer o_1 over o_2. We want to make sure that only the formula that counts *all* subcriteria of c that strictly prefer o_1 over o_2 is justified. To this end, the following inference rules provide an undercutter for the previous schemes when they are non-maximal.

$$\frac{\texttt{spref}(c_1,o_1,o_2) \quad \ldots \quad \texttt{spref}(c_n,o_1,o_2) \quad \texttt{sub}(c,c_1) \quad \ldots \quad \texttt{sub}(c,c_n) \quad m<n}{SP(c,o_1,o_2,m) \text{ is inapplicable}}\ SP(c,o_1,o_2,m)uc$$

$$\frac{\sim\texttt{pref}(c_1,o_1,o_2) \quad \ldots \quad \sim\texttt{pref}(c_n,o_1,o_2) \quad \texttt{sub}(c,c_1) \quad \ldots \quad \texttt{sub}(c,c_n) \quad m<n}{NWP(c,o_1,o_2,m) \text{ is inapplicable}}\ NWP(c,o_1,o_2,m)uc$$

Example 3. *The following argument concludes that there is one subcriterion of* `c4` *that strictly prefers* `o1` *over* `o2`*.*

$$\dfrac{\dfrac{\dfrac{\vdots}{\texttt{pref(c1,o1,o2)}} \quad \dfrac{}{\sim\texttt{pref(c1,o2,o1)}}}{\texttt{spref(c1,o1,o2)}} \quad \texttt{sub(c4,c1)}}{\texttt{sp(c4,o1,o2,1)}}$$

It is also possible to construct an argument stating that there are two such criteria, but it will be undercut.

$$\dfrac{\dfrac{\dfrac{\vdots}{\texttt{pref(c1,o1,o2)}} \quad \dfrac{}{\sim\texttt{pref(c1,o2,o1)}}}{\texttt{spref(c1,o1,o2)}} \quad \dfrac{\dfrac{\vdots}{\texttt{pref(c2,o1,o2)}} \quad \dfrac{}{\sim\texttt{pref(c2,o2,o1)}}}{\texttt{spref(c2,o1,o2)}}\ *}{\texttt{sp(c4,o1,o2,2)}} \quad \texttt{sub(c4,c1)} \quad \texttt{sub(c4,c2)}$$

$$\dfrac{\dfrac{\vdots}{\texttt{pref(c2,o2,o1)}}}{* \text{ is inapplicable}}$$

The following argument concludes that c4 *prefers* o1 *over* o2.

$$\dfrac{\dfrac{\vdots}{\texttt{sp(c4,o1,o2,1)}} \quad \dfrac{}{\texttt{nwp(c4,o1,o2,0)}} \quad 1 \geq 0}{\texttt{pref(c4,o1,o2)}}$$

Preference by a lexicographic criterion The following inference rule concludes that a lexicographic criterion c prefers an outcome o_1 over an outcome o_2 if o_1 is preferred over o_2 by a subcriterion of c. This inference is undercut by the next inference rule if there is a subcriterion of c with higher priority that does not prefer o_1 over o_2.

$$\dfrac{\texttt{lc}(c) \quad \texttt{sub}(c,c_1) \quad \texttt{pref}(c_1,o_1,o_2)}{\texttt{pref}(c,o_1,o_2)}\ LC(c,c_1,o_1,o_2)$$

$$\dfrac{\texttt{lc}(c) \quad \texttt{sub}(c,c_2) \quad \sim\texttt{pref}(c_2,o_1,o_2) \quad \sim\texttt{prior}(c,c_1,c_2)}{LC(c,c_1,o_1,o_2) \text{ is inapplicable}}\ LC(c,c_1,o_1,o_2)uc$$

According to its definition in a QPS, a lexicographic criterion c prefers o_1 over o_2 if every subcriterion either (weakly) prefers o_1 over o_2 or there is a higher priority subcriterion that strictly prefers o_1 over o_2. So if c prefers o_1 to o_2, all undominated (w.r.t. priority) subcriteria prefer o_1 to o_2. $\texttt{pref}(c,o_1,o_2)$ can be derived based on any of those subcriteria, and there will be no justified undercutter. If c does not prefer o_1 to o_2, it may still be possible to construct an argument for $\texttt{pref}(c,o_1,o_2)$, but it will be undercut because there is another subcriterion that does not prefer o_1 to o_2 and does not have lower priority. So together this pair of inference schemes correctly models the definition of preference by a lexicographic criterion in a QPS.

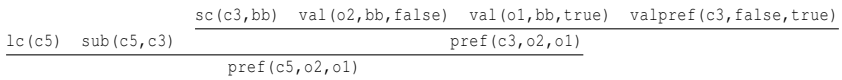
Example 4. *The following argument concludes that* c5 *prefers* o1 *to* o2 *based on its subcriterion* c4.

$$\dfrac{\texttt{lc(c5)} \quad \texttt{sub(c5,c4)} \quad \dfrac{\vdots}{\texttt{pref(c4,o1,o2)}}}{\texttt{pref(c5,o1,o2)}}\ *$$

However, this argument is undercut by the following one stating that there is another subcriterion, c3*, that does not prefer* o1 *to* o2 *and does not have lower priority than* c4.

$$\dfrac{\texttt{lc(c5)} \quad \texttt{sub(c5,c3)} \quad \dfrac{}{\sim\texttt{pref(c3,o1,o2)}} \quad \dfrac{}{\sim\texttt{sprior(c5,c4,c3)}}}{* \text{ is inapplicable}}$$

The only justified argument for preference between o1 *and* o2 *by* c5 *is the following one.*



3.6 Correspondence between QPS and AF

Theorem 1. Let $S = \langle Var, Dom, K, \Omega, \mathcal{C} \rangle$ be a QPS, KB a knowledge base that corresponds to S, and AF the argumentation framework built from KB. Then φ is a sceptically justified conclusion of AF iff its interpretation holds in S.

For every formula in KB, its interpretation holds in S (definition of correspondence). Every formula in the input language whose interpretation holds in S is in KB (definition of correspondence). All formulas in KB are justified since KB is conflict-free. For every inference rule, its conclusion is justified if and only if its premises are justified and all its undercutters (if any) are overruled. We have shown that every inference or pair of inference and its undercutter inference models the corresponding QPS definition: the interpretation of the conclusion holds in a QPS if and only if the interpretations of all premises hold and and the interpretations of the premises of all undercutters do not all hold.

4 Reasoning with Background Knowledge

The argumentation framework presented in the previous section models a QPS if the input is a knowledge base corresponding to that QPS. In order for a knowledge base to correspond to a QPS, it is necessary to specify the values of all variables for every outcome. This correpsonds to the formal (abstract) concept of an outcome as an assignment of a value to every variable in a given set of variables, as defined in the QPS framework.

In practice, an outcome is a concrete alternative (a decision, product, agreement etc.). The major difference is that not all attributes may be known. In a sense, such alternatives can be seen as partial outcomes (or sets of outcomes that share some attributes). Even though not all attributes may be specified beforehand, it is often possible to derive the values of some of the unspecified variables using background information. For example, if it is not specified whether someone speaks the language for a given holiday option, such information may be inferred if it is known that the destination is Barcelona which is in Spain, where the language is Spanish, Juan will accompany Anne, and he speaks Spanish.

In this section we introduce an extension of the argumentation framework in which it is possible to reason with such background knowledge. To this end, we extend the language and add one more inference scheme. This extension makes the system more flexible in case of incomplete specifications. If some attributes remain unknown even with reasoning with background knowledge, the argumentation framework still works correctly, it will just infer less preferences.

4.1 Language

Background knowledge is expressed using a set of predicates $\mathcal{P}_K$ which may differ per application domain. Atoms built with these predicates may also be negated (strong negation). Furthermore, a new construct is added to the input language: (defeasible) *rules* that consist of a set of (possibly weakly negated) antecedents and a consequent (the same kind of rules are used in [6]).

Definition 10. (Language) The input language is defined as follows.

$atom_{in}$ ::= $p(t_1, \ldots, t_n)$ where p is an n-ary predicate $\in \mathcal{P}_{in}$
$atom_K$ ::= $p(t_1, \ldots, t_n)$ where p is an n-ary predicate $\in \mathcal{P}_K$
$literal_{in}$ ::= $atom_{in} \mid atom_K \mid \neg atom_K$
$rule$ ::= $literal_{in}, \ldots, literal_{in}, \sim literal_{in}, \ldots, \sim literal_{in}$ => $literal_{in}$
$formula_{in}$ ::= $literal_{in} \mid rule$

The full language is defined as follows.

$atom_{out}$ ::= $p(t_1, \ldots, t_n)$ where p is an n-ary predicate $\in \mathcal{P}_{out}$
$literal$ ::= $literal_{in} \mid atom_{out}$
$formula$ ::= $literal \mid \sim literal \mid rule$

Contradictory formulas Adding strong negation to the language also adds an additional way in which two formulas can be contradictory.

– A and $\neg A$ contradict each other.

Example 5. *Anne's criteria for a holiday are the same as before, but the information that she has about her options is different. The values of the variables* `sl`, `su` *and* `bb` *on which her preferences are based are not specified. Instead, for every outcome she only knows who of her friends is going with her (*`fr`*): Juan (*`j`*) or Mario (*`m`*), and the destination (*`de`*): Barcelona (*`b`*) or Rome (*`r`*). Besides, she has some relevant background information. All of this is specified in the following knowledge base.*

Some facts from the background knowledge:

```
in(b,spain)                 in(r,italy)
mediterranean(spain)        mediterranean(italy)
language(spain,spanish)     language(italy,italian)
speaks(j,spanish)           speaks(m,italian)
beenTo(b)
```

Some rules from the background knowledge:

```
val(O,fr,X), val(O,de,C), in(C,Cn), language(Cn,L), speaks(X,L) => val(O,sl,true)
~val(O,sl,true) => val(O,sl,false)
val(O,de,C), in(C,Cn), mediterranean(Cn), ~val(O,su,false) => val(O,su,true)
val(O,de,C), beenTo(C) => val(O,bb,true)
~val(O,bb,true) => val(O,bb,false)
```

Facts about some of the possible outcomes:

```
val(o1,fr,j)     val(o2,fr,j)     val(o3,fr,m)     val(o4,fr,m)
val(o1,de,b)     val(o2,de,r)     val(o3,de,b)     val(o4,de,r)
```

Information about the preferences:

```
lc(c5)          cc(c4)          sc(c1,sp)     valpref(c1,true,false)
sub(c5,c3)      sub(c4,c1)      sc(c2,s)      valpref(c2,true,false)
sub(c5,c4)      sub(c4,c2)      sc(c3,n)      valpref(c3,false,true)
prior(c5,c3,c4)
```

4.2 Inferences

Defeasible modus ponens This inference rule applies a rule $L_1, \ldots, L_k, \sim L_l, \ldots, \sim L_m$ => L_n: when all its antecedents hold, the consequent is concluded.

$$\frac{L_1, \ldots, L_k, \sim L_l, \ldots, \sim L_m \texttt{ => } L_n \quad L_1 \quad \ldots \quad L_k \quad \sim L_l \quad \ldots \quad \sim L_m}{L_n} \; DMP$$

Note the difference between a rule in the language and an inference rule. Defeasible modus ponens is an inference rule that applies a rule from the language. We reserve inference rules for domain-independent inferences, and provide the possibility to specify domain-specific rules in the language. Instead of possible undercutters of an inference rule, it is possible to have weakly negated antecedents for the same purpose.

Example 6. *Below are some of the arguments that can be built with the knowledge base from Example 5. The values for the variables* `su` *and* `bb` *can be derived in a similar way.*

$$\frac{r \quad \texttt{val(o1,fr,j)} \quad \texttt{val(o1,de,b)} \quad \texttt{in(b,spain)} \quad \texttt{lang(spain,spanish)} \quad \texttt{speaks(j,spanish)}}{\texttt{val(o1,sl,true)}}$$

where r is `val(O,fr,X), val(O,de,C), in(C,Cn), lang(Cn,L), speaks(X,L) => val(O,sl,true).`

$$\frac{\texttt{~val(O,sl,true) => val(O,sl,false)} \quad \overline{\texttt{~val(o2,sl,true)}}}{\texttt{val(o2,sl,false)}}$$

The argument deriving a preference for `o1` *over* `o2` *by criterion* `c5` *is the same as in Example 4, except that* `val(o2,bb,false)` *and* `val(o1,bb,true)` *are derived instead of taken directly from the knowledge base (for reasons of space, the argument is cut in three).*

$$\frac{\texttt{lc(c5)} \quad \texttt{sub(c5,c3)} \qquad \dfrac{\texttt{sc(c3,bb)} \quad A \quad B \quad \texttt{valpref(c3,false,true)}}{\texttt{pref(o1,o2)}}}{\texttt{pref(c5,o2,o1)}}$$

$$A: \; \frac{\texttt{~val(O,bb,true) => val(O,bb,false)} \quad \texttt{~val(o2,bb,true)}}{\texttt{val(o2,bb,false)}}$$

$$B: \; \frac{\texttt{val(O,de,C), beenTo(C) => val(O,bb,true)} \quad \texttt{val(o1,de,b)} \quad \texttt{beenTo(b)}}{\texttt{val(o1,bb,true)}}$$

5 Conclusion

In this paper we presented an argumentation framework for representing and reasoning about qualitative multi-criteria preferences. We showed that this argumentation framework models the preferences as defined by qualitative preference systems. Qualitative

preference systems use both cardinality and lexicographic ordering to combine multiple criteria, which are ultimately based on the attributes of the outcomes. In an extension of the base argumentation framework we added the means to reason with background knowledge, which adds expressivity and flexibility in case of incomplete specifications.

Argumentation about preferences has been studied extensively in the context of *decision making* [7, 8]. The aim of decision making is to choose an action to perform. The quality of an action is determined by how well its consequences satisfy certain criteria. For example, [8] present an approach in which arguments of various strengths in favour of and against a decision are compared. However, it is a two-step process in which argumentation is used only for epistemic reasoning. Also in [9, 10], preferences are based on arguments, but not themselves derived using argumentation. In our approach, we combine reasoning about knowledge, criteria and preferences between outcomes in a single argumentation framework.

Within the context of argumentation, an approach that is related to criteria is *value-based argumentation* [11, 12]. Values are used in the sense of 'fundamental social or personal goods that are desirable in themselves' [12], and are used as the basis for persuasive argument in practical reasoning. A value can be seen as a binary criterion that is satisfied if the value is promoted. In value-based argumentation, arguments are associated with values that they promote. Values are ordered according to importance to a particular audience. An argument only defeats another argument if it attacks it and the value promoted by the attacked argument is not more important than the value promoted by the attacker. In this framework, every argument is associated with only one value, while in many cases there are multiple values or interests at stake. [13] define so-called *value-specification argumentation frameworks*, in which arguments can support multiple values, and preference statements about values can be given. However, the preference between arguments is not derived from the preference between the values promoted by the arguments. Besides, there is no guarantee that a value-specification argumentation framework is consistent, i.e., some sets of preference statements do not correspond to a preference ordering on arguments.

In value-based argumentation, we cannot argue *about* what values are promoted by the arguments or the ordering of values; this mapping and ordering are supposed to be given. But these might well be the conclusion of reasoning, and might be defeasible. Therefore, it would be natural to include this information at the object level. [14] describe some argument schemes regarding the influence of certain perspectives on values. However, for the aggregation of multiple values, they assume a given order on sets of values, whereas we want to derive such an order from an order on individual values.

In our future work we would like to look into the possibilities that the presented framework offers to not only derive missing information about the attributes of outcomes, but also information about e.g. the criteria that are used and their preferences between attribute values, or priority between subcriteria. This would be especially useful when modelling other agents' preferences, e.g. the opponent in negotiation or someone you have to make a joint decision with. Often, another person's preferences are not (completely) known, but some of them may be inferred by default.

Acknowledgements

This research is supported by the Dutch Technology Foundation STW, applied science division of NWO and the Technology Program of the Ministry of Economic Affairs. It is part of the Pocket Negotiator project with grant number VICI-project 08075.

References

1. Amgoud, L., Cayrol, C.: Inferring from inconsistency in preference-based argumentation frameworks. Journal of Automated Reasoning **29** (2002) 125–169
2. Kakas, A., Moraïtis, P.: Argumentation based decision making for autonomous agents. In: Autonomous Agents and Multiagent Systems (AAMAS 2003). (2003) 883–890
3. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. Artificial Intelligence **77** (1995) 321–357
4. Prakken, H.: An abstract framework for argumentation with structured arguments. Argument and Computation **1**(2) (2010) 93–124
5. Vreeswijk, G.A.W.: Abstract argumentation systems. Artificial Intelligence **90**(1-2) (1997) 225–279
6. Prakken, H., Sartor, G.: Argument-based extended logic programming with defeasible priorities. Journal of Applied Non-Classical Logics **7** (1997) 25–75
7. Ouerdane, W., Maudet, N., Tsoukiàs, A.: Argumentation theory and decision aiding. In Ehrgott, M., Figueira, J.R., Greco, S., eds.: New Trends in Multiple Criteria Decision Analysis. Springer Verlag (2010)
8. Amgoud, L., Prade, H.: Using arguments for making and explaining decisions. Artificial Intelligence **173**(3-4) (2009) 413–436
9. Bonnefon, J.F., Fargier, H.: Comparing sets of positive and negative arguments: Empirical assessment of seven qualitative rules. In: 17th European Conference on Artificial Intelligence (ECAI 2006), IOS Press (2006) 16–20
10. Dubois, D., Fargier, H., Bonnefon, J.F.: On the qualitative comparison of decisions having positive and negative features. Journal of Artificial Intelligence Research **32** (2008) 385–417
11. Bench-Capon, T.J.M.: Persuasion in practical argument using value based argumentation frameworks. Journal of Logic and Computation **13**(3) (2003) 429–448
12. Bench-Capon, T., Atkinson, K.: Abstract argumentation and values. In Rahwan, I., Simari, G.R., eds.: Argumentation in Artificial Intelligence. Springer (2009) 45–64
13. Kaci, S., van der Torre, L.: Preference-based argumentation: Arguments supporting multiple values. International Journal of Approximate Reasoning **48**(3) (2008) 730–751
14. van der Weide, T., Dignum, F., Meyer, J.J., Prakken, H., Vreeswijk, G.: Practical reasoning using values: Giving meaning to values. In: 6th International Workshop on Argumentation in Multi-Agent Systems (ArgMAS 2009). (2009)