A Framework for Multiagent Deliberation Based on Dialectical Argumentation

A. G. Stankevicius G. R. Simari

Grupo de Investigación en Inteligencia Artificial (GIIA)
Departamento de Ciencias de la Computación
Universidad Nacional del Sur
Bahía Blanca - Buenos Aires - ARGENTINA
e-mail: {ags,grs}@cs.uns.edu.ar

Abstract

Simply put, a multiagent system can be seen as a collection of autonomous agents that as a whole are able to accomplish goals beyond the reach of any of its members. Agent interaction is widely acknowledged as the feature that provides this added potential. Since many, if not all, of the attractive agent interactions can be recasted as deliberations, a formalization for this process is being actively seek.

Deliberations among agents resembles a dialectical process like the one present in many formalizations of defeasible argumentation. This paper exploits that resemblance by defining a framework for multiagent deliberation based on a particular dialectical process borrowed from a well-established system of defeasible argumentation.

Keywords: deliberation, defeasible argumentation, multiagent systems.

1 Introduction

Simply put, a multiagent system can be seen as a collection of autonomous agents that as a whole are able to accomplish goals beyond the reach of any of its members. Agent interaction is widely acknowledged as the feature that provides this added potential. This interaction comes in several flavors—coordination, cooperation, and collaboration among others—but there is one seemingly ubiquitous: deliberation. A group agents deliberate when they need to come to a mutually accepted position about some issue. Since many if not all of the attractive agent interactions can be recasted as deliberations, a formalization for this process is being actively seek.

Since deliberations and negotiations share a common structure, successful approaches to either one can generate similar results in the other. Considering this, the recent findings in the field of negotiation can be used as a guide to tackle multiagent deliberation. The traditional approach for modeling negotiation resort to game theory [12]. Even though several insightful issues have been explored under this conception, it depends on the strong assumption that each agent is aware of the complete pay-off matrix (*i.e.*, they know their preferences and also the preferences of their counterparts) *before* the negotiation begins. One might conclude that this assumption restrains the applicability of game-theoretic based negotiation.

A new approach that has recently gained a lot of attention considers negotiation from the point of view of defeasible argumentation [9, 8, 7]. We agree with this particular view; in fact, we have argued in a previous work [16] that negotiations among agents resembles a dialectical process like the one present in many formalizations of defeasible argumentation. This resemblance can also be exploited in order to formalize multiagent deliberation. Consequently, this paper defines a framework for multiagent deliberation based on a particular dialectical process the dialectical analysis borrowed from the well-established system of defeasible argumentation defined in [6].

The remainder is structured as follows. Section 2 describes the proposed framework. Section 3 analyzes the behavior of this framework with a toy example of deliberation. Finally, section 4 gathers the conclusions obtained and outlines the future work.

2 The framework

This section defines the proposed framework for multiagent deliberation based on dialectical argumentation. We begin by characterizing how agents represent their knowledge, and then we discuss the different types of agents inhabiting the framework. Finally, we describe the dispute protocol that underlies the actual deliberation.

2.1 Knowledge representation

Every agent must use the same coding for its knowledge. We would like to model the epistemic state of the agents with logic programs an alternative already explored in the literature with satisfying results [10, 1]. Yet, conventional logic programming cannot deal with partial and potentially contradictory information, a recurring situation when modeling real world agents. Following the solution suggested in [2], we adopt a representation for the agents' knowledge based on a defeasible logic program [4], a formalism that by combining traditional logic programming with defeasible argumentation avoids those shortcomings.[1]

Defeasible logic programming represents knowledge using strict and defeasible rules. Strict rules capture certain information (*e.g.*, Fred being a penguin allows us to conclude that Fred is a bird), and defeasible rules capture tentative information (*e.g.*, Tweety being a bird allows us to conclude that Tweety usually flies). In this system, a *literal* is either an atomic predicate p or its negation $\sim p$. Note that the symbol "$\sim$" denotes strong negation (also known as classical negation), which should not be confused with the traditional negation in logic programming (negation as failure).

Definition 2.1 [Defeasible logic program] A *defeasible logic program* is a finite set of *strict* and *defeasible rules*. A strict rule has the form "$l \leftarrow p_1, \ldots, p_n$.", $n \geq 0$, where l is a literal and each p_i is either a literal or the symbol "`not`" of negation as failure followed by a literal. If $n = 0$, we say that l is a *fact*, denoted "l.". A defeasible rule has the form "$l \prec p_1, \ldots, p_n$.", $n \geq 0$, with the same considerations for l and the p_i as before. If $n = 0$ we say that l is a *presumption*, denoted "$l \prec$ true.". ■

[1]for an in-deep discussion of this system we refer the interested reader to [14, 13, 6].

Since nonmonotonicity can be expressed using defeasible rules, we do not allow the use of negation as failure in the representation of knowledge within agents. Moreover, in our framework we assume that defeasible rules do not have an empty body. Even though presumptions are particularly useful in knowledge representation [4], the effect of allowing them in a multiagent scenario is unclear, and subject of further investigation.

Definition 2.2 [Knowledge Base] A *knowledge base* is a finite set KB of tuples $\langle rule, Ag \rangle$, where $rule$ is a rule either strict or defeasible, and Ag is the name of the agent believing it. When needed, the set KB can be divided in the disjoint sets Π of tuples containing strict rules and Δ of tuples containing defeasible rules. ■

We extend the notion of defeasible logic program with labels that allow the agent to model not only its own knowledge but also knowledge about other agents. Each agent uses its knowledge to build arguments. An argument represents a defeasible reason for an assertion.

Definition 2.3 [Argument] Let $KB = \Pi \cup \Delta$ be the knowledge base of an agent Ag, and let Π_{Ag} (resp. Δ_{Ag}) be set of rules contained in the tuples of Π (resp. Δ) labeled with Ag. An *argument* $\mathcal{A}$ for a literal h is a subset of Δ_{Ag}, such that:

- there exists a defeasible derivation for h from $\Pi_{Ag} \cup \mathcal{A}$,
- the set $\Pi_{Ag} \cup \mathcal{A}$ is non-contradictory, and
- $\mathcal{A}$ is minimal with respect to set inclusion (*i.e.*, there is no $\mathcal{A}' \subset \mathcal{A}$ such that $\mathcal{A}'$ satisfies the two previous conditions).

If $\mathcal{A}$ is an argument for h, $\langle \mathcal{A}, h \rangle$ is also called *argument structure*. We say that $\langle \mathcal{A}, h \rangle$ is a *sub-argument* of $\langle \mathcal{A}', h' \rangle$ if and only if $\mathcal{A} \subseteq \mathcal{A}'$. ■

The set of justified literals constitutes the epistemic state of the agent. A literal h is said to be justified only when it is supported by a justified (*i.e.*, non-defeated) argument $\mathcal{A}$. The formal definition of defeat follows.

Definition 2.4 [Counter-argument] Let $KB = \Pi \cup \Delta$ be the knowledge base of an agent Ag. We say that $\langle \mathcal{A}_1, h_1 \rangle$ *counter-argues* $\langle \mathcal{A}_2, h_2 \rangle$ *at the literal* h *with respect to* KB, if and only if there is a sub-argument $\langle \mathcal{A}, h \rangle$ of $\langle \mathcal{A}_2, h_2 \rangle$ such that the set $\Pi_{Ag} \cup \{h_1, h\}$ is contradictory. ■

Definition 2.5 [Defeat] Let KB be the knowledge base of an agent Ag. An argument $\langle \mathcal{A}_1, h_1 \rangle$ *defeats* $\langle \mathcal{A}_2, h_2 \rangle$ *at the literal* h *with respect to* KB, if and only if there is a sub-argument $\langle \mathcal{A}, h \rangle$ of $\langle \mathcal{A}_2, h_2 \rangle$ such that $\langle \mathcal{A}_1, h_1 \rangle$ counter-argues $\langle \mathcal{A}_2, h_2 \rangle$ at h with respect to KB, and either:

- $\langle \mathcal{A}_1, h_1 \rangle$ is *strictly more specific*[2] *with respect to* KB than $\langle \mathcal{A}, h \rangle$ (proper defeat), or
- $\langle \mathcal{A}_1, h_1 \rangle$ is *unrelated by specificity with respect to* KB to $\langle \mathcal{A}, h \rangle$ (blocking defeat)

■

In order to establish whether $\mathcal{A}$ is a non-defeated argument, counter-arguments that could be defeaters for $\mathcal{A}$ are looked for. Since defeaters are also arguments, there may exist defeaters for the defeaters, and so on, thus requiring a complete recursive analysis. This recursive analysis is structured as a *dialectical tree*, whose formal definition follows.

[2] a notion introduced by Poole in [11], later extended for defeasible logic programming in [5].

Definition 2.6 [Dialectical tree] Let *KB* be the knowledge base of an agent *Ag*. A *dialectical tree* for $\langle \mathcal{A}, h \rangle$, denoted $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$, is recursively defined as follows:

1. A single node labeled with an argument $\langle \mathcal{A}, h \rangle$ having no defeaters with respect to *KB* is by itself the dialectical tree for $\langle \mathcal{A}, h \rangle$.

2. Let $\langle \mathcal{A}_1, h_1 \rangle, \ldots, \langle \mathcal{A}_n, h_n \rangle$ be all the defeaters with respect to *KB* for $\langle \mathcal{A}, h \rangle$. We construct the dialectical tree for $\langle \mathcal{A}, h \rangle$, $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$, by labeling the root node with $\langle \mathcal{A}, h \rangle$ and by making this node the parent node of the roots of the dialectical trees for $\langle \mathcal{A}_1, h_1 \rangle, \ldots, \langle \mathcal{A}_n, h_n \rangle$.

■

As shown in [15], the dialectical analysis can effortlessly be recasted into a dispute between two opposing parties. We already mentioned that deliberations among agents resembles a dialectical process: the dialectical analysis is this process.

2.2 Types of agents

There are two types of agents involved in this framework: regular agents and arbiters. To begin with, we discuss conditions that regular agents must satisfy, and then we address the differences between arbiters and regular agents.

Although we are not assuming any particular architecture for the agents, a certain behavior is required in our framework to successfully engage other agents. Since knowledge representation has been fixed, every agent must have an information repository capable of holding the defeasible logic program that characterizes its epistemic state. Naturally, they must also have a suitable inference engine. The inference engine defined for defeasible logic programming [4] is appropriate making the minor adjustments needed to avoid mixing knowledge corresponding to different agents.

Every agent must be aware of the existence of its counterparts by maintaining all the information required to locate and access any of them. Moreover, it is assumed that they understand the following set of performatives:

`ask(A,B,C)`: Agent *A* ask *B* whether it believes in *C*. As a consequence, agent *B* uses the performative `tell` to inform agent *A* about the current status (according to its *KB*) of *C* .

`tell(A,B,C,D)`: Agent *A* tells *B* that its state regarding *C* is *D*. This performative usually comes as a response to an `ask`. There are three possible states `D` for the literal `C`:

- If the literal is believed, then `D = yes`.
- If the literal is not believed, then `D = no`.
- If the literal is neither believed nor disbelieved, then `D = ?`.

`why(A,B,C)`: Agent *A* asks *B* why *C* should be believed. If *A* still believes in *C*, agent *A* answers with a `because` providing one of the arguments justifying *C*. Otherwise, agent *A* answers with a `tell` letting agent *B* know its current opinion over *C*.

`because(A,B,C,D)`: Agent *A* hands over to *B* an argument *C* justifying *D*. Naturally, this performatives makes sense only over believed literals. It usually comes as a response to a `why`, but it is also used throughout deliberations.

`engage(A,B,C,D)`: Agent A lets B know that it wants to deliberate about C arbitrated by D. As a consequence, agent B decides whether it is willing to deliberate (answering with an `accept`) or not (answering with a `reject`).

`accept(A,B,C,D)`: Agent A agrees to deliberate with B about C arbitrated by D. At this stage, agents A and B begin to deliberate according to the dispute protocol defined later in section 2.3.

`reject(A,B,C,D)`: Agent A refuses to deliberate with B about C arbitrated by D. Agent B may retry another `engage` modifying the subject or the arbiter previously proposed.

Additionally, the agents may implement other performatives or even a full agent communication language such as KQML [3] or FIPA.

Finally, arbiters ensure that deliberations among regular agents obey the guidelines established by the framework. They have the structure of a regular agent with the addition of an *argument pool*. Since the argument pool is only used in deliberations, its role is described in the sequel along with the dispute protocol.

2.3 Deliberation protocol

Given the relevance of deliberation in multiagent systems, several protocols that characterize the deliberation process have been proposed [12, 8, 9]. Still, the protocol is only a part of this process: there are actions to be performed *before* and *after* the deliberation itself.

Therefore, we decompose a deliberation in the following steps:

1. An agent decides that it needs to deliberate about a certain matter with another agent.

2. The agent engages the chosen counterpart. It contacts the other agent through an `engage`. If the agent gets an `accept`, the deliberation is ready to begin. Otherwise, the agent can either change the subject, propose another arbiter, or give up the deliberation attempt.

3. The actual deliberation takes place. In this step, the deliberation is performed according to the dispute protocol defined below.

4. The outcome of the deliberation is accounted.

Notice how arbiters are summoned before the beginning of the actual deliberation. Even though we endorse conceiving coordination and cooperation as a by-products of deliberation among *many* agents, we restrict our analysis to disputes between *pairs* of agents. The extension to the general case is under development.

The term "deliberation" has been used with diverse meanings in the literature. In our framework, we understand it to be the process that allows an agent to persuade another agent about some matter: an agent prevailing in a deliberation can influence the epistemic state of its counterpart. Briefly stated, deliberations in this framework are strictly over claims (*i.e.*, literals), and can take place involving only two regular agents: a proponent backing the claim, and an opponent usually rejecting it.

Definition 2.7 [Deliberation] Let Ag_1 and Ag_2 be regular agents, and let h be a literal believed by Ag_1. Then, agent Ag_1 can deliberate with Ag_2 over h arbitrated by an arbiter Ar if and only if:

- agent Ag_2 accepts the terms of the deliberation, and
- the strict knowledge of both agents is consistent (*i.e.*, $\Pi_{Ag_1} \cup \Pi_{Ag_2} \not\vdash \bot$).

■

The consistency precondition in a deliberation averts those disputes that cannot be settled in any way (*i.e.*, the conflict can be traced back to the strict knowledge). Unfortunately, one may argue that this precondition is too restrictive since it prevents agents from deliberating about any issue once a conflict arises between the strict part of their knowledge. As a future work, we expect to refine this precondition into a less restrictive one.

Definition 2.8 [Dispute] Let Ag_1 and Ag_2 be two regular agents, and let Ar be an arbiter. Suppose that Ag_1 proposed Ag_2 to deliberate over a claim h arbitrated by Ar, and that Ag_2 accepted the proposal. Then, a *dispute* between agents Ag_1 and Ag_2 over h arbitrated by Ar follows this scheme:

1. The proponent (agent Ag_1) initiates the discussion providing the arbiter (agent Ar) with a justified argument (justified according to its *KB*) supporting h. The performative `because` is used to convey this initial argument to the arbiter. The turn goes to the opponent (agent Ag_2).
2. The opponent (agent Ag_2) either relinquish its turn or rebuts (according to its *KB*) an argument previously posed by its counterpart. In the former, the turn goes back to the proponent (agent Ag_1). In the latter, the rebutting argument is sent to the arbiter through a `because`, passing the turn to the proponent.
3. The proponent (agent Ag_1) must rebut (according to its *KB*) an argument previously posed by its counterpart. If it can, the performative `because` provides the arbiter with the rebutting argument, and the turn goes back to the opponent (agent Ag_2). In any other case, the dispute is over.

■

As usual, the proponent bears the burden of the proof. Before taking into account the possible outcomes of a deliberation, let us delve into the bookkeeping performed by the arbiter amidst the dispute. The arbiter begins by checking whether the deliberation may proceed. In order to guarantee the consistency precondition, both the proponent and the opponent declare to the arbiter their current strict knowledge, and the arbiter stores it in its *KB*. Once deliberating, the arbiter must verify the validity of every move made by the contenders. To this purpose, the conditions on arguments are checked (see definition 2.3), and the rebutting arguments are verified with respect to the corresponding *KB*.

The arbiter also keeps track of every argument structure introduced throughout the dispute using its argument pool. This argument pool is organized as follows.

Definition 2.9 [Argument pool] An *argument pool* is a set of sequences composed by pairs $(\langle \mathcal{A}, h \rangle, Ag)$, where each pair contains an argument structure and the name of the agent that introduced this argument structure. ■

As the discussion progress, the argument pool stores the (partial) argumentation lines being developed in the deliberation. Finally, in order to avoid the so-called fallacious argumentation [13, 6] the arbiter impose some extra restriction on the argument structures that are allowed to be introduced on a given stage of the dispute.

Definition 2.10 [Acceptable move] Let $\langle \mathcal{B}, h \rangle$ be an argument structure of the proponent (agent Ag_1), and let $\langle \mathcal{C}, h' \rangle$ be an argument structure of the opponent (agent Ag_2). In this setting, the proponent can move $\langle \mathcal{B}, h \rangle$ to rebut $\langle \mathcal{C}, h' \rangle$ if and only if the pool of arguments contains at least one sequence with $[(\langle \mathcal{A}_1, h_1 \rangle, Ag_1), (\langle \mathcal{A}_2, h_2 \rangle, Ag_2), \ldots, (\langle \mathcal{A}_n, h_n \rangle, Ag_1), (\langle \mathcal{C}, h' \rangle, Ag_2)]$ as its prefix, and also the following conditions are met:

- $\langle \mathcal{B}, h \rangle$ rebuts $\langle \mathcal{C}, h' \rangle$ according to the knowledge base of Ag_1.
- $[(\langle \mathcal{A}_1, h_1 \rangle, Ag_1), (\langle \mathcal{A}_2, h_2 \rangle, Ag_2), \ldots, (\langle \mathcal{A}_n, h_n \rangle, Ag_1), (\langle \mathcal{C}, h' \rangle, Ag_2), (\langle \mathcal{B}, h \rangle, Ag_1)]$ does not appear as prefix of any sequence already present in the argument pool,
- in $[(\langle \mathcal{A}_1, h_1 \rangle, Ag_1), (\langle \mathcal{A}_2, h_2 \rangle, Ag_2), \ldots, (\langle \mathcal{A}_n, h_n \rangle, Ag_1), (\langle \mathcal{C}, h' \rangle, Ag_2), (\langle \mathcal{B}, h \rangle, Ag_1)]$, all the arguments introduced by the same agent are non-contradictory, and
- the argument $\langle \mathcal{B}, h \rangle$ is not a sub-argument of the arguments posed by agent Ag_1 in the sequence $[(\langle \mathcal{A}_1, h_1 \rangle, Ag_1), (\langle \mathcal{A}_2, h_2 \rangle, Ag_2), \ldots, (\langle \mathcal{A}_n, h_n \rangle, Ag_1), (\langle \mathcal{C}, h' \rangle, Ag_2)]$.

If these conditions are met, the argument pool is updated by adding the pair $(\langle \mathcal{B}, h \rangle, Ag_1)$ to the sequence $[(\langle \mathcal{A}_1, h_1 \rangle, Ag_1), (\langle \mathcal{A}_2, h_2 \rangle, Ag_2), \ldots, (\langle \mathcal{A}_n, h_n \rangle, Ag_1), (\langle \mathcal{C}, h' \rangle, Ag_2)]$ denoting that $\langle \mathcal{C}, h' \rangle$ has been rebutted by $\langle \mathcal{B}, h \rangle$. The case where Ag_1 is the opponent and Ag_2 is the proponent is defined in a like manner. ■

Once the dispute is over, the proponent wins if every sequence in the argument pool has an odd length (*i.e.*, all the argumentation lines successfully sustained the attacks). In contrast, the opponent wins if there exists a sequence in the argument pool with an even length. Notice that a clever agent can gain some additional insights into the belief structure of its counterpart by keeping track of the moves made throughout the discussion. Finally, the outcome of deliberation depends upon which agent prevailed in it.

Definition 2.11 [Deliberation outcome] Let Ag_1 and Ag_2 be two regular agents that recently finished a deliberation over a certain claim h. Suppose that agent Ag_1 prevailed in the dispute. The possible outcomes of this deliberation are:

- If Ag_1 was the proponent, its KB can remain unchanged. In contrast, Ag_2 is now committed to believe h (it has been persuaded to), and must update its KB accordingly. In other words, if Ag_2 receives an `ask` about h after the deliberation, it is now compelled to answer positively.
- If Ag_1 was the opponent, neither Ag_1 nor Ag_2 need to update their knowledge bases.

■

We already stressed that a deliberation encompasses more tasks than the blindly compliance of some protocol. Taking account of the deliberation outcome can be particularly challenging. Suppose that an agent Ag_1—who believes in h—engages agent Ag_2 —who believes in $\sim h$—in a deliberation over h, and that Ag_1 manages to prevail in it. According to our definition, Ag_2

is now committed to believe in h, but it certainly cannot believe in h and $\sim h$ at the same time! Even though the actual mechanism implementing this behavior is independent from our framework, we believe that the outcome of a deliberation should be treated as a perception of the agent, pretending that the agent loosing the discussion was persuaded by its counterpart to "see" the truth of the claim deliberated over. Naturally, this makes sense only in the context of agents that already have some mechanism for perception.

3 A toy example

This section presents a toy example that explores two scenarios where different outcomes are attained starting from the same situation (recall that the outcome of a deliberation is asymmetric by definition). In this example, agents Ag_1 and Ag_2 are going to argue whether certain car is expensive or not.

Prior to the actual deliberation we need to establish what is believed by each agent. Suppose that both agents agree on the following defeasible rules,[3]

$$\sim\mathsf{expensive(X)} \mathrel{-\!\!\prec} \mathsf{beetle(X)}. \qquad\qquad \mathsf{expensive(X)} \mathrel{-\!\!\prec} \mathsf{new\text{-}beetle(X)}.$$

saying that beetles—a widely known Volkswagen model—are usually inexpensive, and that the recently introduced new-beetle is typically quite expensive (at least when compared against its elder brother). Besides, they agree that crashed cars are usually not expensive:

$$\sim\mathsf{expensive(X)} \mathrel{-\!\!\prec} \mathsf{crashed(X)}.$$

Since new-beetles share a lot of features with its predecessor (big doors, beetle-like shape, same manufacturer, etc.), both agents accept the following strict rule saying that new-beetles are a subclass of traditional beetles:

$$\mathsf{beetle(X)} \leftarrow \mathsf{new\text{-}beetle(X)}.$$

Finally, suppose that both agents know that some car they refer to as c is a new-beetle, a situation modeled by the following fact:

$$\mathsf{new\text{-}beetle(c)}.$$

To make it more interesting, suppose that only agent Ag_1 knows that this particular car is crashed.

$$\mathsf{crashed(c)}.$$

Summing up, the knowledge bases of agent Ag_1 and Ag_2 are composed of the following information:

Π_{Ag_1}	Δ_{Ag_1}
$\mathsf{beetle(X)} \leftarrow \mathsf{new\text{-}beetle(X)}.$	$\sim\mathsf{expensive(X)} \mathrel{-\!\!\prec} \mathsf{beetle(X)}.$
$\mathsf{new\text{-}beetle(c)}.$	$\mathsf{expensive(X)} \mathrel{-\!\!\prec} \mathsf{new\text{-}beetle(X)}.$
$\mathsf{crashed(c)}.$	$\sim\mathsf{expensive(X)} \mathrel{-\!\!\prec} \mathsf{crashed(X)}.$

Π_{Ag_2}	Δ_{Ag_2}
$\mathsf{beetle(X)} \leftarrow \mathsf{new\text{-}beetle(X)}.$	$\sim\mathsf{expensive(X)} \mathrel{-\!\!\prec} \mathsf{beetle(X)}.$
$\mathsf{new\text{-}beetle(c)}.$	$\mathsf{expensive(X)} \mathrel{-\!\!\prec} \mathsf{new\text{-}beetle(X)}.$
	$\sim\mathsf{expensive(X)} \mathrel{-\!\!\prec} \mathsf{crashed(X)}.$

[3]a rule containing variables stands for all its ground instances.

According to this information, the following arguments regarding whether c is expensive can be built:

- $\langle \mathcal{A}_1, \sim\mathsf{expensive(c)}\rangle$, where $\mathcal{A}_1 = \{\sim\mathsf{expensive(X)} \prec \mathsf{beetle(X)}.\}$
- $\langle \mathcal{A}_2, \mathsf{expensive(c)}\rangle$, where $\mathcal{A}_2 = \{\mathsf{expensive(X)} \prec \mathsf{new\text{-}beetle(X)}.\}$
- $\langle \mathcal{A}_3, \sim\mathsf{expensive(c)}\rangle$, where $\mathcal{A}_3 = \{\sim\mathsf{expensive(X)} \prec \mathsf{crashed(X)}.\}$

Note that agent Ag_1 can build all the three arguments, but agent Ag_2 can only build $\mathcal{A}_1$ and $\mathcal{A}_2$. Moreover, Ag_1 believes in ~expensive(c) since $\mathcal{A}_3$ is strictly more specific than $\mathcal{A}_2$, but Ag_2 believes the opposite since $\mathcal{A}_2$ is strictly more specific than $\mathcal{A}_1$.

There are two scenarios to consider. In the first place, let us assume that Ag_1 wants to engage Ag_2 in a deliberation about ~expensive(c) arbitrated by Ar, and that Ag_2 is willing to accept. The deliberation might proceed as follows (where exp stands for expensive):

	Ag_1	Ag_2
1.-	engage(Ag_1, Ag_2, ~exp(c), Ar)	
2.-		accept(Ag_2, Ag_1, ~exp(c), Ar)
3.-	because(Ag_1, Ar, $\mathcal{A}_1$, ~exp(c))	
4.-		because(Ag_2, Ar, $\mathcal{A}_2$, exp(c))
5.-	because(Ag_1, Ar, $\mathcal{A}_3$, ~exp(c))	

At this stage, the deliberation is over since the opponent cannot make new moves, and the proponent was able to successfully defend every line of argumentation. The argument pool kept by the arbiter traversed the following states:

$Pool_1 = Pool_2 = \{\}$
$Pool_3 = \{[\langle \mathcal{A}_1, \sim\mathsf{expensive(c)}\rangle]\}$
$Pool_4 = \{[\langle \mathcal{A}_1, \sim\mathsf{expensive(c)}\rangle, \langle \mathcal{A}_2, \mathsf{expensive(c)}\rangle]\}$
$Pool_5 = \{[\langle \mathcal{A}_1, \sim\mathsf{expensive(c)}\rangle, \langle \mathcal{A}_2, \mathsf{expensive(c)}\rangle, \langle \mathcal{A}_3, \sim\mathsf{expensive(c)}\rangle]\}$

Therefore, the proponent prevailed in the deliberation. According to definition 2.11, agent Ag_2 must update its KB in order to believe in ~expensive(c).

For the second scenario, let us assume the complementary situation where Ag_2 wants to engage Ag_1 in a deliberation about expensive(c) arbitrated by Ar, and that Ag_1 is willing to accept. In this case, the deliberation might proceed as follows:

	Ag_1	Ag_2
1.-		engage(Ag_2, Ag_1, exp(c), Ar)
2.-	accept(Ag_1, Ag_2, exp(c), Ar)	
3.-		because(Ag_2, Ar, $\mathcal{A}_2$, exp(c))
4.-	because(Ag_1, Ar, $\mathcal{A}_3$, ~exp(c))	

At this stage, the deliberation is over since the proponent cannot make any move. However, note that the argument pool still contains an open argumentation line defeating the claim being disputed.

$Pool_1 = Pool_2 = \{\}$
$Pool_3 = \{[\langle \mathcal{A}_2, \mathsf{expensive(c)}\rangle]\}$
$Pool_4 = \{[\langle \mathcal{A}_2, \mathsf{expensive(c)}\rangle, \langle \mathcal{A}_3, \sim\mathsf{exp(c)}\rangle]\}$

Therefore, the opponent prevailed in the deliberation. Notice that the proponent, albeit loosing the deliberation, does not need to change its KB to believe in ~expensive(c).

4 Conclusions

In this paper we have defined a framework that allows agents to deliberate, based on the resemblance between a dialectical analysis and the actual discussion underlying deliberations. From our viewpoint, agents deliberate when they need to persuade other agents about some matter. Although this framework is currently restricted to deliberations between pairs of agents, the extension to an arbitrary number is being pursed since we firmly believe that the advanced interactions among agents—coordination, cooperation, and collaboration among others—are in fact by-products of deliberation.

Recall that a deliberation in this setting can proceed only if certain preconditions are fulfilled. Among those requirement, the consistency precondition stating that the strict knowledge of agents willing to deliberate should be conflict free seems too restrictive. As a future work, we plan to explore new refinements of this precondition to make it less restrictive.

We have also suggested that the deliberation process encompasses more than just a protocol outlining the exchange of information. In consequence, four stages have been identified. To our surprise, formalizing the final stage where the outcome of the deliberation is taken into account seems more difficult than characterizing the actual dispute. We have sketched a tentative approach for this final stage: to consider the outcome of the deliberation as a new perception observed by the agent.

Finally, we have pointed out that an agent in this framework can gain insights into the belief structure of another agent engaged in a deliberation by examining the moves made throughout the dispute. This situation deserve further analysis since it models an interesting aspect present in the traditional deliberations among human beings.

References

[1] Baral, C., and Gelfond, M. Logic Programming and Knowledge Representation. *Journal of Logic Programming 12* (1993), 1–80.

[2] Capobianco, M., and Chesñevar, C. I. Using Logics Programs to Model an Agent's Epistemic State. In *Proceedings of the 7th Workshop on Aspectos Teóricos de la Inteligencia Artificial (ATIA), 2nd Workshop of Investigadores en Ciencias de la Computación (WICC)* (La Plata, Argentina, May 2000), Universidad Nacional de La Plata.

[3] Finin, T., Labrou, Y., and Mayfield, J. KQML as an Agent Communication Language. In *Software Agents*, J. Bradshaw, Ed. MIT Press, 1997.

[4] García, A. J. La Programación en Lógica Rebatible: su definición teórica y computacional. Master's thesis, Departamento de Ciencias de la Computación, Universidad Nacional del Sur, Bahía Blanca, Argentina, June 1997.

[5] García, A. J., and Simari, G. R. El criterio de especificidad en la programación en lógica rebatible. In *Proceedings of the III Workshop sobre Aspectos Teóricos de la Inteligencia Artificial* (Nov. 1996), Universidad Nacional de San Luis.

[6] García, A. J., Simari, G. R., and Chesñevar, C. I. An Argumentative Framework for Reasoning with Inconsistent and Incomplete Information. In *Proceedings of the Workshop on Practical Reasoning and Rationality* (Brighton, United Kingdom, Aug. 1998), 13th European Conference on Artificial Intelligence, pp. 13–19.

[7] JENNINGS, N. R., PARSONS, S., NORIEGA, P., AND SIERRA, C. On argumentation-based negotiation. In *Proceedings of the International Workshop on Multi-Agent Systems* (Boston, United States, 1998).

[8] KRAUS, S., SYCARA, K., AND EVENCHIK, A. Reaching Agreements through Argumentation: A Logical Model and Implementation. *Artificial Intelligence 104*, 1–2 (1998), 1–69.

[9] PARSONS, S., SIERRA, C., AND JENNINGS, N. Agents that Reason and Negotiate by Arguing. *Journal of Logic and Computation 8*, 3 (1998), 261–292.

[10] PEREIRA, L. M., APARÍCIO, J. N., AND ALFERES, J. J. Nonmonotonic Reasoning with Well Founded Semantic. In *Proceedings of the 8th International Conference on Logic Programming* (June 1991), K. Furokawa, Ed., MIT, pp. 475–489.

[11] POOLE, D. L. On the Comparison of Theories: Preferring the Most Specific Explanation. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence* (1985), pp. 144–147.

[12] ROSENSCHEIN, J., AND ZLOTKIN, G. *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers*. Artificial Intelligence Series. MIT Press, 1994.

[13] SIMARI, G. R., CHESÑEVAR, C. I., AND GARCÍA, A. J. The Role of Dialectics in Defeasible Argumentation. In *Proceedings of the XIV Conferencia Internacional de la Sociedad Chilena para Ciencias de la Computación* (Concepción, Chile, Nov. 1994), Universidad de Concepción, pp. 111–121.

[14] SIMARI, G. R., AND LOUI, R. P. A Mathematical Treatment of Defeasible Reasoning and its Implementation. *Artificial Intelligence 53*, 1–2 (1992), 125–157.

[15] STANKEVICIUS, A. G., AND GARCÍA, A. J. Modelling Negotiation Protocols in a Dialectical Framework. In *Proceedings of the 6th Workshop on Aspectos Teóricos de la Inteligencia Artificial (ATIA), 1st Workshop of Investigadores en Ciencias de la Computación (WICC)* (San Juan, Argentina, May 1999), Universidad Nacional de San Juan, pp. 69–76.

[16] STANKEVICIUS, A. G., GARCÍA, A. J., AND SIMARI, G. R. Could Negotiation Among Agents be Regarded as an Argumentative Process. In *Proceedings of the 7th Workshop on Aspectos Teóricos de la Inteligencia Artificial (ATIA), 2nd Workshop of Investigadores en Ciencias de la Computación (WICC)* (La Plata, Argentina, May 2000), Universidad Nacional de La Plata.