Argumentation Systems and Agent Programming Languages

Sebastian Gottifredi and **Alejandro J. Garcia** 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

In this work we will present an integration of a query-answering argumentation approach with an abstract agent programming language. Agents will argumentatively reason via queries, using information of their mental components. Special context-based queries will be used to model the interaction between mental components. Deliberation and execution semantics of the proposed integration are presented.

Introduction

In this work, we present a declarative agent programming language (APL) that uses argumentation as a reasoning formalism. Agent mental components of the proposed APL will be represented in Defeasible Logic Programing (DeLP) (Garcia and Simari 2004). Hence, an agent developer will be able to represent conflicting goals and beliefs, and the argumentation formalism will be used for deciding which beliefs and goals are regarded as warranted. Our approach follows the spirit of 3APL, where interactions between mental components are made through queries in the sense of logic programming; therefore we will use DeLP contextual queries (García et al. 2007) to connect the mental components. We will propose semantic rules to show agent execution dynamics and a set of properties.

Several articles in the literature recognize the relevance of using argumentation as the reasoning mechanism of agent systems (Bench-Capon and Dunne 2007). Since the beginnings of BDI (Bratman, Israel, and Pollack 1991), the use of defeasible reasoning to handle intentional reasoning was considered. Later, (Parsons, Sierra, and Jennings 1998) move forward in that direction and points the advantages of using argumentation in BDI agents. Nowadays, various papers (Amgoud, Devred, and Lagasquie 2008; Rotstein, Garcia, and Simari 2007; Rahwan and Amgoud 2006) present important contributions in this area, connecting argumentative approaches to BDI agent architectures. These articles show clearly how agent mental components can be benefited with the use of argumentation for reasoning with conflicting information. However, in the field of APL, although deliberation is an important topic, it is necessary to establish execution semantics (or agent state dynamics). That is, in APLs area is important to specify rules that show how actions will affect agent mental components, and thus agent future reasoning. The above mentioned papers propose BDI architectures instead of APLs, therefore, they are mainly focused in the argumentative support for agent deliberation and not in agent execution dynamics. Here, we will present two contributions in the area of argumentation and APLs: an integration of a concrete argumentation formalism to the deliberative model of an APL; and a set of APL semantic rules for agent execution dynamics, which is combined with the argumentation formalism.

Recently, logic based agent programming languages have received special attention (Bordini, Braubach, and et al. 2006), (Dastani, van Riemsdijk, and Meyer 2005), (Bordini, Wooldridge, and Hübner 2007). These languages provide interesting features like declarative agent specifications, clear semantics of agent reasoning mechanisms, and verifiable agent development. However, none of them propose an argumentative formalism for agent reasoning. Since the focus of this research is related to agent specification and implementation, our approach will be inspired on these languages.

DeLP Basis

In order to provide an argumentative reasoning service for multi-agent systems, an implementation of DeLP, called DeLP-server, has been developed (García et al. 2007). A DeLP-server is a stand-alone program that can interact with multiple clients. A public DeLP-program can be stored in a server, and clients may send queries to the server and receive the corresponding answer.

In DeLP (Garcia and Simari 2004), knowledge is represented using facts, strict rules or defeasible rules. *Facts* are ground literals representing atomic strict information or the negation of atomic information using the strong negation "$\sim$". *Defeasible Rules* (d-rules) are denoted $L_0 -\prec L_1, \ldots, L_n$ (where L_i are literals) and represent defeasible knowledge, *i. e.* tentative information. In this paper we will consider a restricted form of program that do not have strict rules.

Definition 1 (Restricted Defeasible Logic Program) *A restricted defeasible logic program (de.l.p. for short) $\mathcal{P}$ is a set of facts and d-rules. When required, $\mathcal{P}$ is denoted* (Ψ,Δ)

distinguishing the subset Ψ *of facts and the subset* Δ *of d-rules.*

Strong negation is allowed in the head of program rules, and hence, may be used to represent contradictory knowledge. From a program (Ψ,Δ) contradictory literals could be derived, however, the set Ψ must possess certain internal coherence (it has to be non-contradictory). Given a literal L the complement with respect to strong negation will be noted $\overline{L}$.

Definition 2 (DeLP-query) *A* DeLP-query *is a ground literal that DeLP will try to warrant.*

To deal with contradictory information, in DeLP, *arguments* for conflicting pieces of information are built and then compared to decide which one prevails. The prevailing argument provides a *warrant* for the information that it supports. In DeLP, a query L is *warranted* from a program (Ψ,Δ) if a *non-defeated* argument $\mathcal{A}$ supporting L exists. An *argument* $\mathcal{A}$ for a literal L, denoted $\langle \mathcal{A}, L\rangle$, is a minimal set $\mathcal{A}\subseteq\Delta$, such that $\mathcal{A} \cup \Psi$ is non-contradictory and there is a derivation for L from $\mathcal{A} \cup \Psi$.

To establish if $\langle \mathcal{A}, L\rangle$ is a non-defeated argument, *defeaters* for $\langle \mathcal{A}, L\rangle$ are considered, *i.e.*, counter-arguments that are preferred to $\langle \mathcal{A}, L\rangle$ by some criterion. In DeLP the argument comparison criterion is modular, thus, the most appropriate criterion for the domain that is being represented can be selected. In the examples in this paper we will use *generalized specificity*, a criterion that favors two aspects of an argument: it prefers (1) a more precise argument or (2) a more concise argument

A defeater $\mathcal{D}$ for an argument $\mathcal{A}$ can be *proper* ($\mathcal{D}$ is preferred to $\mathcal{A}$) or *blocking* (same strength). A defeater can attack the conclusion of another argument or an inner point of it. Since defeaters are arguments, there may exist defeaters for them, defeaters for these defeaters, and so on. Thus, leading to a exhaustive tree structure called *dialectical tree* (for more details see (Garcia and Simari 2004)). In a dialectical tree, every node (except the root) is a defeater of its parent, and leaves are non-defeated arguments. In a dialectical tree every node can be marked as *defeated* or *undefeated*: leaves are marked as undefeated nodes, and inner nodes are marked defeated when there is at least a child marked undefeated, or are marked undefeated when all its children are marked defeated.

Definition 3 (Warranting a DeLP-query) *A DeLP-query* Q *is warranted from a de.l.p.* $\mathcal{P}$ *(noted* $\mathcal{P}|\sim_w Q$*) if there exists an argument* $\mathcal{A}$ *supporting* Q *such that* $\mathcal{A}$ *is the root of a dialectical tree and it is marked as undefeated.*

Next, we will show that the set of warranted queries inferred by a *de.l.p.* are non-contradictory sets (a *de.l.p.* can not warrant a literal and its complement simultaneously).

Proposition 1 *Given a de.l.p.* $\mathcal{P} = (\Psi, \Delta)$ *where* Ψ *is consistent, the set* $W = \{Q_i \mid \mathcal{P}|\sim_w Q_i\}$ *is non-contradictory.*
Proof Sketch: Let W be the set of warranted queries from $\mathcal{P}$ and $h_1 \in W$. Thus, there exists an argument $\langle \mathcal{A}_1, h_1\rangle$ for h_1 which is undefeated in a dialectical process. Suppose $h_2 \in W$, and h_1 and h_2 are complementary ($h_1 = \overline{h_2}$). As h_2 is warranted there exists an argument $\langle \mathcal{A}_2, h_2\rangle$ for h_2 which is undefeated in the argumentation process. As h_1 and h_2 are contradictory $\langle \mathcal{A}_1, h_1\rangle$ is a counter-argument for $\langle \mathcal{A}_2, h_2\rangle$ and vice versa. Depending on the comparison criterion used, $\langle \mathcal{A}_2, h_2\rangle$ can be a proper defeater of $\langle \mathcal{A}_1, h_1\rangle$(or viceversa), or $\langle \mathcal{A}_2, h_2\rangle$ and $\langle \mathcal{A}_1, h_1\rangle$are blocking defeaters. Since $\langle \mathcal{A}_2, h_2\rangle$ is undefeated and is a defeater (blocking or proper) $\langle \mathcal{A}_1, h_1\rangle$, h_1 is not warranted from $\mathcal{P}$, which contradicts our hypothesis.□

To answer queries, a DeLP-server will use the public knowledge stored in it, together with individual knowledge that clients can send attached to a query, creating a particular *context* for that query (see (García et al. 2007) for details). This context is knowledge that the server will use for answering the query and will not affect other future queries. That is, a client agent cannot make permanent changes to the public *de.l.p.* stored in a server. The temporal scope of the context sent in a query [$Context$,Q] is limited and disappears once the query Q has been answered. Since contextual information can be in contradiction with the information stored in the server, different types of contextual queries were defined. In this work we will use one kind of contextual query, which will be used to model the interaction of the mental components of the agent programming language proposed in the next section. This query is specifically designed for this work and was not presented in (García et al. 2007).

Definition 4 (Contextual query) *Let* $\mathcal{P}$ = (Ψ,Δ) *be a de.l.p., a contextual query for* $\mathcal{P}$ *is a pair* $[(\Phi^+, \Theta^-), Q]$ *where* Q *is a DeLP-query,* Φ^+*and* Θ^-*are non-contradictory sets representing the context to be added and removed from* $\mathcal{P}$ *respectively, in order determine the status of* Q*.*

This special kind of contextual query will temporarily add and remove elements from $\mathcal{P}$. The literals of Φ^+ will be added as facts to Ψ in a way that Ψ consistency is not harmed. The literals of Θ^- will determine two actions: the literals to be removed form Ψ and the rules to be removed from Δ (whose head coincides with them).

Definition 5 (Contextual query warrant) *Let* $\mathcal{P}$= (Ψ,Δ) *be a de.l.p., and* $[(\Phi^+, \Theta^-), Q]$ *a contextual query.* $\mathcal{P}|\sim_w[(\Phi^+, \Theta^-), Q]$ *iff* $\mathcal{P}_{cq}|\sim_w Q$ *where* $\mathcal{P}_{cq} = (((\Psi \oplus \Phi^+) \setminus \Theta^-, (\Delta \ominus \Theta^-))$. *Let* $C(\Phi^+) = \{\overline{L}|L \in \Phi^+\}$*, then* $(\Psi \oplus \Phi^+) = (\Psi \setminus C(\Phi^+)) \cup \Phi^+$. *Let* $(\Delta \ominus \Theta^-) = \{r_i | r_i = L_h \prec L_0, \ldots, L_k \in \Delta \wedge L_h \notin \Theta^-\}$

Example 1 *Consider the de.l.p.* $\mathcal{PB}_1 = (\Psi_1^B, \Delta_1^B)$*, used by an agent to determine if it can spend money and if it can travel, where the elements of* $\Psi_1^B = \{d, s, gf, gs, \sim a\}$ *represent that it has debts (d), has savings (s), has girlfriend (gf), has a good salary (gs) and is not afraid to flight (*$\sim a$*).* Δ_1^B =$\{$ $(m \prec s)$ $(m \prec d, gs, s)$ $(\sim m \prec d)$ $(\sim m \prec \sim s)$ $(t \prec v)$ $(\sim t \prec \sim m)\}$*, where t, m, and v are abbreviations of can travel, can spend money, and has one week vacation, respectively. From* $\mathcal{PB}_1$*, the contextual queries* $[(\emptyset, \emptyset), m]$*,* $[(\emptyset, \{s\}), \sim t]$*,* $[(\{v\}, \emptyset), t]$ *and* $[(\{v\}, \{\sim t\}), t]$ *are warranted.*

Next, with the following propositions we will show that temporary modifications made by the contextual query are

sound, and a *de.l.p.* cannot warrant complementary literals for the same context.

Proposition 2 *Let* $[(\Phi^+, \Theta^-), Q]$ *be a contextual query for a de.l.p.* $\mathcal{P}$= (Ψ,Δ). *Then the program* $\mathcal{P}_{cq}$ *used to determine the status of* Q*, is a de.l.p.*
Proof: Let $\mathcal{P} = (\Psi,\Delta)$ a *de.l.p.* and $[(\Phi^+, \Theta^-), Q]$ a contextual query. By Def. 5 $\mathcal{P}_{cq} = (((\Psi \oplus \Phi^+) \setminus \Theta^-, (\Delta \ominus \Theta^-))$ will be used to determine the status of Q. Ψ is consistent because $\mathcal{P}$ is a *de.l.p.*, Φ^+ is consistent by Def. 4 and by Def. 5 $\oplus$ will remove from Ψ all the complements of the literals of Φ^+ (removes all possible conflicts), then it will add all the literals of Φ^+ to Ψ, thus $(\Psi \oplus \Phi^+)$ is consistent. Removing literals does no harm consistency, then $(\Psi \oplus \Phi^+) \setminus \Theta^-$ is consistent. Since the removal of d-rules do not interfere with the integrity of a *de.l.p.*, $\mathcal{P}_{cq}$ is a *de.l.p.*□

Proposition 3 *Given a de.l.p.* $\mathcal{P} = (\Psi, \Delta)$*, if* $\mathcal{P}|\sim_w [(\Phi^+, \Theta^-), Q]$ *then* $\mathcal{P} \not|\sim_w [(\Phi^+, \Theta^-), \overline{Q}]$.
Proof: Let $\mathcal{P}$ be a *de.l.p.* and $[(\Phi^+, \Theta^-), Q]$ a contextual query such that $\mathcal{P}|\sim_w [(\Phi^+, \Theta^-), Q]$. By Prop. 2, we know that the output $\mathcal{P}_{cq}$ of applying Φ^+, Θ^- to $\mathcal{P}$ is a *de.l.p.*By Prop. 1 the set of warranted queries W from a *de.l.p.* $\mathcal{P}_{cq}$ is non-contradictory, and by hypothesis $Q \in W$, then $\overline{Q} \notin W$. Finally, $\mathcal{P} \not|\sim_w [(\Phi^+, \Theta^-), \overline{Q}]$ □

APL Formal Concepts

In this section, we will show how the defeasible argumentation can be integrated into agent programming languages using DeLP-Servers. We will introduce a declarative BDI agent programming language, called DeLPAP. This language is partially inspired on 3APL (Dastani, van Riemsdijk, and Meyer 2005), where the agent is composed by a Belief Base, a Goal Base and a set of Reasoning Rules. Also in these kind of APLs, interactions between each mental component are made through queries in the sense of logic programing. Following that approach we will use contextual queries to model the interaction between agent internal components. Next, we will introduce the formal concepts of DeLPAP: syntax, semantics and framework properties.

Syntax

Agents in DeLPAP are characterized by three components: belief and goal knowledge bases, and a set of plan rules. Knowledge bases represent agent mental components, and will be specified by *de.l.p.*s. These knowledge bases are used to compute agents beliefs and goals in each deliberative cycle. The plan rules will be used to generate plans in order to achieve goals.

Beliefs The belief base, is used to represent the information that the agent has and infers about the world. In DeLPAP it will be represented by a *de.l.p.* and generally denoted $\mathcal{PB} = (\Psi^B, \Delta^B)$. Therefore an agent developer will be able to specify potentially contradictory information using the d-rules. For instance, the *de.l.p.* $\mathcal{PB}_1$ of Ex. 1 can be a belief base of an agent to determine, for example, if it can travel or if it can spend money.

Perceptual information will be also considered as part of the beliefs, and therefore it can appear in the body of belief base d-rules. For instance, in Ex. 1, the literal v used to denote that the agent has one week vacation, can be considered as a perception and it is used in one of the d-rules of that example. All perceptual elements will be identified by the set of literals E. However, we will explain how the $\mathcal{PB}$ interacts with E in the semantics subsection, since E will change dynamically as the world changes.

In order to refer to Beliefs from the other components, we will use special literals $B(L)$ (with L a literal) called *actual beliefs*. These literals will denote that the agent believes in L. In the semantics subsection we will explain how these special literals are calculated from the belief base.

Goals Goals are used to denote situations of the world that the agent want to realize. The goal base has knowledge about agent goals, and it is used to determine which of these goals are good options for the agent in each deliberative cycle. In agent programming, goals may be unconditional or conditional. Unconditional (or independent) goals are always adopted. Conditional goals are not always adopted, they depend on other goals and beliefs. Goals may also be conflicting, *i. e.*, some goals may not be adoptable at the same time, therefore, a mechanism to decide which goals the agent must adopt should be used. In DeLPAP goals will be represented by a *de.l.p.*, where: unconditional goals will be modeled using facts, and conditional goals using d-rules. Since strong negation can be in facts and d-rules, conflicting goals will be representable. The formalism will identify the goals that are in conflict, and the argumentation process will be used for deciding which one to adopt. *Actual beliefs* in the body of d-rules will be used to obtain belief information.

Definition 6 (Goal Base) *The agent goal base is a de.l.p.* $\mathcal{PG} = (\Psi^G, \Delta^G)$*, where* Δ^G *has rules of the form* $(L_h \prec B(L_0), \ldots, B(L_k), L_{k+1}, \ldots, L_n)$*, with* $k + n \geq 0$ *and* L_i *is a Literal*

Example 2 *Following the belief base of Ex.1, now consider the goal base* $\mathcal{PG}_1 = (\emptyset, \Delta_1^G)$*, used by the agent to determine if want to visit a place or not. Here,* Δ_1^G =*{* $(p \prec B(t), B(v), B(gf))$ $(h \prec B(t), B(v))$ $(\sim p \prec h, B(v))$ $(\sim h \prec p, B(v))$*}. Letters p and h are abbreviations to denote "want to visit Paris" and "want to visit Hong Kong", and actual belief literals B(t),B(v) and B(gf) denote agent beliefs of Ex.1. For example, the first rule expresses that if it believes that it can travel, it has one week vacation and it has a girlfriend there are good reasons to visit paris; the fourth rule denotes that if there are reasons to visit Paris and it believes that has one week vacation, it will no be possible to visit Hong Kong.*

Notice that not all rules need to denote reasons to adopt goals, a rule can be used to denote reasons against adopting a goal. For instance, the third and fourth d-rules of Ex.2 denotes reasons against visiting Paris and Hong Kong respectively.

In order to refer to Goals from the other components, we will use special literals $G(L)$ (with L a literal) called *actual goals*. These literals will denote that the agent wants to achieve L. In the semantics subsection we will explain how to obtain these literals from the goal base.

Plan Rules In order to decide how to act DeLPAP agents will use planing rules, which are based in the reasoning rules used in 3APL. These rules will establish a mapping between goals and plans. Basically, a plan rule will determine the plan to execute with the objective of achieving its associated goals, when some belief preconditions are met. We will capture these notions in the following definition:

Definition 7 (Plan Rule) *a plan rule has the form* $\rho = \kappa \leftarrow \beta \mid \pi$, *where* $\kappa = \{G(K_0), \ldots, G(K_n)\}$, $\beta = \{B(P_0), \ldots, B(P_m)\}$ *with* $K_{i,i>0}$, $P_{j,j>0}$ *literals and* π *a plan. The set of all planing rules available for the agent is called plan rule base and denoted* $\mathcal{R}$

In this work we will assume that plans will be simple sequences of actions of the form $[a_1, \ldots, a_m]$. Clearly, more complex plan structures like the ones presented in 3APL can be easily adapted to DeLPAP. However, to simplify the explanation of the basis of DeLPAP, we will leave the plan structure as elementary as possible. Actions can be used to interact with the environment or dynamically modify the mental components. The former will be represented by atoms and the later will be defined next:

Definition 8 (Mental Action) *A Mental Action (or Ma for short) is a triplet* $Ma = (Base, \beta, Pos)$, *where Base is a de.l.p. representing the mental component that will be affected by the action,* $\beta = \{B(L_0), \ldots, B(L_k)\}$ *with* $k \geq 0$, *and* $Pos = \{[X_1, \ldots, X_n, not\ Y_1, \ldots, not\ Y_m]\}$ *with* $n \geq 0$, $m \geq 0$, *where* X_i *is an element to be added to Base and* Y_j *is an element to be removed from Base.* X_i *and* Y_j *can be a literal or a d-rule.*

Example 3 *Continuing Ex. 1 and Ex. 2, consider that the agent has the following plan rules set* $\mathcal{R}_1$*:* $G(p) \leftarrow B(\sim a) \mid [flightParis, (\mathcal{PB}, \{B(s)\}, \{p, \sim s\})]$ *and* $G(h) \leftarrow B(\sim a) \mid [flightHK, (\mathcal{PB}, \{B(s)\}, \{h, \sim s\})]$

For instance, the first plan rule expresses that if the agent wants to visit Paris, and believes that it is no afraid to flight, then it should first execute an external action of flying to Paris and then execute a mental action to denote that he has spent its savings and that he has visited Paris.

As in most declarative agent programming languages, to program a DeLPAP agent means to specify the *de.l.p.*s of the mental components, and the plan rule base. For instance, a DeLPAP agent can be specified using the belief base of Ex.1, the goal base of of Ex.2, and the plan rule base of Ex. 3. However, the initial beliefs and goals of individual agents and their environment can change during the execution of the agent, while the plan rules will remain the same. In the following subsection we will give the operational semantics of a DeLPAP agent dynamics.

Semantics

In this section, we will define DeLPAP formal semantics. As in most cognitive agent programming languages, practical reasoning involves two fundamental processes: to decide which goals are going to be pursued, and to choose plans to achieve them. Semantics of DeLPAP will describe these processes. The decision process requires considering alternative goals, and the selection and commitment to some of them. Selected goals will have an influence on its actions, restrict future practical reasoning, and persist in time. The planning process consists on selecting a set of actions that will allow to satisfy the committed goal. Basically, in DeLPAP this behavior consists on determining the warranted goals and beliefs, use them to establish which plans are executable, and finally execute a plan.

Following the style from the semantics specification of other BDI agent programming languages such as AgentSpeak(L) and 3APL, the semantics will be given using a transition system. A transition system is a set of derivation rules for deriving transitions. A transition is a transformation of one system configuration C into another C' if a condition holds, and it corresponds to a single computation step. The configuration of a DeLPAP agent is a snapshot of its current mental components at a certain moment. Since the agent will be executing plans to achieve its goals, the dynamic model of a DeLPAP agent will also have an adopted plan or currently executing plan, denoted Π. Also, the dynamic model of a DeLPAP agent will be characterized by a set of literals E denoting the current perception.

Definition 9 (Configuration) *A configuration of a DeLPAP agent is a tuple* $\langle E, \mathcal{PB}, \mathcal{PG}, \Pi \rangle$ *where E is a set of literals denoting agent perceptions,* $\mathcal{PB}$ *is a belief base,* $\mathcal{PG}$ *is a goal base, and* Π *is the adopted plan.*

The set $\mathcal{R}$ of plan rules is not included in the agent configuration, because it will not change during agent execution. We will just assume that the set E changes over time. However we will not proceed any further with the dynamics and semantics of E, since it is mainly domain dependent. We will use ϵ (the empty plan) to denote that the agent has not an adopted plan. The initial configuration of an agent specifies the initial goal and belief bases and the initial plan base empty, that is $\langle E, \mathcal{PB}, \mathcal{PG}, \epsilon \rangle$.

From a given configuration it is possible to determine what the agent believes in (*actual beliefs*), which will be literals warranted from the belief base. In actual beliefs warranting process perceptions should be taken account. For that purpose we will use the contextual queries (Def. 4) to add (and revise if necessary) the perpetual information to the belief base, as we will show next:

Definition 10 (Actual belief) *Let* $C = \langle E, \mathcal{PB}, \mathcal{PG}, \Pi \rangle$ *be an agent configuration, a literal L will be an actual belief* $\langle E, \mathcal{PB}, \mathcal{PG}, \Pi \rangle \vdash_B B(L)$ *iff* $\mathcal{PB} |\!\sim_w [(E, \emptyset), L]$. *The set of all the actual beliefs will be denoted* ***B***

Example 4 *Suppose an agent configuration* $C_1 = \langle E, \mathcal{PB}_1, \mathcal{PG}_1, \epsilon \rangle$, *where* $\mathcal{PB}_1$ *and* $\mathcal{PG}_1$ *correspond to Ex.1 and Ex.2 respectively,* $E = \{v\}$. *Then in* C_1 *actual beliefs will be B(gs), B(s), B(d), B(gf), B(∼a), B(m), B(t) and B(v).*

The following propositions, show that perceptual items are considered as beliefs, and that the set of actual beliefs is non contradictory.

Proposition 4 *Let* $C = \langle E, \mathcal{PB}, \mathcal{PG}, \Pi \rangle$ *be an agent configuration, if* $L \in E$ *then* $C \vdash_B B(L)$.

Proof: Let $C = \langle E, \mathcal{PB}, \mathcal{PG}, \Pi \rangle$ be an agent configuration and $L \in E$. By Def.5, $L \in (\Psi^G \oplus E)$, therefore, there will

be an undefeated argument for L in $(((\Psi^G \oplus E) \setminus \emptyset, (\Delta^G \ominus \emptyset))$. Then, $\mathcal{PB}|\sim_w[(E, \emptyset), L]$, and $C \vdash_B B(L)$ □

Proposition 5 *Let $C = \langle E, \mathcal{PB}, \mathcal{PG}, \Pi\rangle$ be an agent configuration, if $C \vdash_B B(L)$ then $C \nvdash_B B(\overline{L})$.*
Proof: Straight forward from Prop.3 and Def. 10.

Goal semantics describe the goals that an agent want to achieve in a given configuration. These goals are called *actual goals*, and will be the warranted literals of the goal base. However, using the information of the goal base alone is not enough to determine these goals: goal arguments may involve beliefs and already achieved goals should not be considered as actual goals. Therefore, actual beliefs are needed. We will capture these notions using contextual queries as it will shown next:

Definition 11 (actual goal) *Let $\langle E, \mathcal{PB}, \mathcal{PG}, \Pi\rangle$ be an agent configuration, a literal L will be an actual goal $\langle E, \mathcal{PB}, \mathcal{PG}, \Pi\rangle \vdash_G G(L)$ iff $\mathcal{PG}|\sim_w[(\boldsymbol{B}, A), L]$, where $A = \{L_i | B(L_i) \in \boldsymbol{B}\}$. The set of actual goals will be denoted **G***

Example 5 *Suppose an agent conf. C_1 of Ex.4. The actual goal of C_1 will be G(p), that is the agent wants to visit Paris.*

As introduced above, the semantics of actual goals should only consider those unachieved goals. Remember that goals represent situations of the world that the agent wants to realise. Therefore, if an agent believes in a literal L, the goal for L should be considered as achieved and ignored in the actual goals warranting process. Next, we will show that our agent programming framework follows these semantics.

Proposition 6 *Let $C = \langle E, \mathcal{PB}, \mathcal{PG}, \Pi\rangle$ be an agent configuration, if $C \vdash_B B(L)$ then $C \nvdash_G G(L)$.*
Proof: Let $C = \langle E, \mathcal{PB}, \mathcal{PG}, \Pi\rangle$ be an agent configuration, with $\mathcal{PG} = (\Psi^G, \Delta^G)$ such that $C \vdash_B B(L)$. By Def.10 $B(L) \in \mathbf{B}$. Let $A = \{L_i | B(L_i) \in \mathbf{B}\}$, then $L \in A$. From the *de.l.p.* $\mathcal{PG}_{cq} = (((\Psi^G \oplus \mathbf{B}) \setminus A, (\Delta^G \ominus A))$ there is no argument for L, since by Def.5 there will be no fact L in $(\Psi^G \oplus \mathbf{B}) \setminus A$ or a d-rule with head L in $\Delta^G \ominus A$. Therefore $\mathcal{PG}_{cq} \not|\sim_w L$, $\mathcal{PG} \not|\sim_w [(\mathbf{B}, A), L]$, and $C \nvdash_G G(L)$ □

Example 6 *Suppose that the agent of Ex.4 is in a new situation, it has achieved the goal of visiting Paris. This situation is described by the conf. $C_2 = \langle E, \mathcal{PB}_2, \Delta_1^B, \mathcal{PG}_1, \epsilon\rangle$, where $\mathcal{PB}_2 = (\Psi_1^B \cup \{p\}, \Delta_1^B)$ and $E, \Delta_1^B, \mathcal{PG}_1, \Pi$ are the same as in Ex.4. Therefore from C_2 its actual goals will be {G(h)}, that is the agent wants to visit Hong Kong. Note that now B(p) is an actual belief of C_2, thus, p was not be considered in goal warranting process.*

Next, we will show that actual goals for a given configuration are non-contradictory

Proposition 7 *Let $C = \langle E, \mathcal{PB}, \mathcal{PG}, \Pi\rangle$ be an agent configuration, if $C \vdash_G G(L)$ then $C \nvdash_G G(\overline{L})$.*
Proof: Straight forward from Prop.3 and Def. 11

Using the actual beliefs and goals, the agent will be able to determine which of its plan rules are applicable, which will establish plans to execute. Basically, a plan rule will be applicable iff its associated goals are actual goals and its associated beliefs are actual beliefs.

Definition 12 (Plan Selection) *Let $C = \langle E, \mathcal{PB}, \mathcal{PG}, \epsilon\rangle$ be an agent configuration, $\mathcal{R}$ the agent plan rules set, and $\rho = \kappa \leftarrow \beta \mid \pi \in \mathcal{R}$*

$$\frac{C \vdash_B b_i \; \forall b_i \in \beta \wedge C \vdash_G g_i \; \forall g_i \in \kappa}{\langle E, \mathcal{PB}, \mathcal{PG}, \epsilon\rangle \rightarrow \langle E, \mathcal{PB}, \mathcal{PG}, \pi\rangle}$$

For example, suppose the conf. C_1 of Ex.4 and the set $\mathcal{R}_1$, Since $C_1 \vdash_B B(\sim a)$ and $C_1 \vdash_G G(p)$, then first rule of $\mathcal{R}_1$ will be applicable and the plan $[flightParis, (\mathcal{PB}, \{B(s)\}, \{p, \sim s\})]$ will be adopted.

This plan selection semantics completes the agent commitment model. When a plan from a plan rule ρ is adopted, the agent commits to the goals of ρ. However, it is important to notice that the execution of the plan does not ensure that ρ goals will be achieved. The achievement status of a goal is only determined by agent beliefs.

Clearly an agent might have more than one applicable plan rule. In this case it has to select one among those using some strategy, for instance, prefer the shorter plan or the one that tries to achieve its more important goals. However, this topic is left for future work.

The actions of the adopted plan will be executed in a sequential fashion. Mental actions introduce permanent changes in the mental components. Next we will present a function where their effects are described:

Definition 13 (Update Function) *Let ($\mathcal{P}$, Pre, Pos) be a Ma, where Pos= { $[X_1, \ldots, X_n$, not $Y_1, \ldots,$ not $Y_m]$,C), and $\mathcal{P}$ = ((Ψ, Δ), C) a de.l.p.Using the elements in Pos we define $AddS$ = $\{X_i \mid \forall X_i$ literal $\}$, $DelS$= $\{\overline{X_i} \mid \forall X_i$ literal $\} \cup \{Y_j \mid \forall Y_j$ literal $\}$, AddD= $\{X_i \mid \forall X_i$ d-rule $\}$, DelD = $\{Y_j \mid \forall Y_j$ d-rule $\}$. Then, the mental action update function is: $\mathcal{T}(Ma, \mathcal{P})$ = (((($\Psi \setminus DelS) \cup AddS$), (($\Delta \setminus DelD) \cup AddD$))*

During the plan execution, if the mental action preconditions are met, the update function will be applied. In contrast, if the preconditions are not met, we say the plan is blocked and it will be removed from execution. Thus, with this behavior, the agent will drop the adopted plan when it is not achievable. Next we will formally show these notions.

Definition 14 (Mental Action Execution) *Let $C = \langle \mathcal{PB}, \mathcal{PG}, [ma, a_0, \ldots, a_m]\rangle$ be an agent configuration, where $[ma, a_0, \ldots, a_m]$ is the adopted plan, and ma = $(\mathcal{PB}, \beta, Pos)$ a mental action. The execution of ma is defined in two cases:*

$$\frac{C \vdash_B b_i \; \forall b_i \in \beta \; \wedge \; (\mathcal{T}(ma, \mathcal{PB}) = \mathcal{PB}')}{\langle E, \mathcal{PB}, \mathcal{PG}, [ma, a_0, \ldots, a_m]\rangle \rightarrow \langle E, \mathcal{PB}', \mathcal{PG}, [a_0, \ldots, a_m]\rangle}$$

$$\frac{\exists b_i \in \beta \;\; C \nvdash_B b_i}{\langle E, \mathcal{PB}, \mathcal{PG}, [ma, a_0, \ldots, a_m]\rangle \rightarrow \langle E, \mathcal{PB}, \mathcal{PG}, \epsilon\rangle}$$

When a plan is finished, ϵ is set as the adopted plan

For instance, suppose that the configuration $C_3 = \langle E, \mathcal{PB}_1, \mathcal{PG}_1, [(\mathcal{PB}_1, \{B(s)\}, \{p, \sim s\})]\rangle$ denotes the situation were the agent is commited to visit to Paris and has already executed the first action of the plan. Since $C_3 \vdash_B B(s)$, then first action of the plan will executable and it will produce the belief base $\mathcal{PB}_3 = (\{\sim s, p, gs, d, gf\}, \Delta_1^B)$ leading to the configuration $\langle E, \mathcal{PB}_3, \mathcal{PG}_1, \epsilon\rangle$

Notice that we have only defined the transition of a mental action to modify the agents belief base. The transition rules to modify goal base can be analogously defined. Next with the following property we will show that in the DeLPAP framework the strict knowledge always remains non-contradictory.

Proposition 8 *Given a DeLPAP agent configuration* $\langle E, \mathcal{PB}, \mathcal{PG}, \Pi \rangle$ *where the sets of facts of* $\mathcal{PB}$ *and* $\mathcal{PG}$ *are non-contradictory. After any possible transitions, these sets will remain non-contradictory.*

Proof: Let $\langle E, (\Psi^B, \Delta^B), (\Psi^G, \Delta^G), \Pi \rangle$ be the agent configuration, where Ψ^B and Ψ^G are non-contradictory. Basically, the only transition capable of harming the consistency of Ψ^B or Ψ^G is the execution of a mental action α which modifies them. Let Ψ_X be the set affected by α, then: if α removes a literal L from Ψ_X, then clearly Ψ_X remains non-contradictory. if α adds a literal L to Ψ_X and $\overline{L} \notin \Psi_X$, then clearly Ψ_X remains non-contradictory. if α adds a literal L to Ψ_X and $\overline{L} \in \Psi_X$, then Ψ_X remains non-contradictory because by definition 13 $\overline{L}$ will be removed from Ψ_X □

The previous propositions show that although the knowledge representation language allows strong negation, if an agent starts with its Ψ sets consistent, regardless the action the agent executes or the changes in the environment, the inferences of the agent will be non-contradictory. Thus, with these properties we have shown that DeLPAP agents can use strong negation and be safe of inconsistencies.

Related Work and Conclusion

In this work we presented a 3APL style agent programming language where mental components are represented and use defeasible argumentation logic programming as the reasoning mechanism. For the proposed APL we have shown how deliberation is made using the argumentation mechanism, and how this deliberation is affected by actions. In order to do this we present syntax, semantics and some properties of this APL. In particular, we used delp-server contextual queries to handle mental component interaction. We have shown that contextual queries were really handy in that task because they modularise the interaction: they provide all the perceptual, belief and goal information that a component needs to answer the query, and the query itself, in one step. Also the special contextual query that we presented allowed to represent the correct semantics for goal queries. That is, an agent will only pursue unachieved goals. We have also shown that through its life span an agent will be safe of inconsistency.

The idea of using defeasible argumentation in the reasoning process of cognitive agents is not new, and there exists previous approaches that relate cognitive agents to argumentation frameworks (Rahwan and Amgoud 2006; Rotstein, Garcia, and Simari 2007; Amgoud, Devred, and Lagasquie 2008). However, none of these approaches propose an agent programming language, they present agent architectures. Their main objective is to determine what is the best thing to do in a given situation. Therefore, their mainly focus is in the impact of argumentation into deliberation, and they put little attention to the dynamics of the model. For example, these works do not specify how actions modify agent mental components, or what happens with the arguments for an achieved desire. In this work, we address these issues presenting semantics rules for actions and deactivating arguments for achieved goals. Another difference resides in the aim of this work, since we wanted to present an implementation friendly approach we used DeLP contextual queries concrete mechanism for mental components interaction. In the architecture presented (Amgoud, Devred, and Lagasquie 2008) intentions are computed in one step, which is a very interesting property of the deliberative model. In this work, since our focus was on presenting the complete dynamic model, some deliberative elements were simplified.

As future work, we plan to study different goal types (maintenance, achievement and procedural) and determine their impact in the dynamics of the arguments. We are also planning on providing the agent with argumentative tools to reason with its committed goals. Also, we want to connect an argumentative planning system to the APL.

References

Amgoud, L.; Devred, C.; and Lagasquie, M. 2008. A constrained argumentation system for practical reasoning. In *International Joint Conference on Autonomous Agents and Multi-Agents Systems (AAMAS)*, 429–436.

Bench-Capon, T. J. M., and Dunne, P. E. 2007. Argumentation in artificial intelligence. *Artif. Intell.* 171:619–641.

Bordini, R. H.; Braubach, L.; and et al. 2006. A survey of programming languages and platforms for multi-agent systems. In *Informatica*, 33–44.

Bordini, R. H.; Wooldridge, M.; and Hübner, J. F. 2007. *Programming Multi-Agent Systems in AgentSpeak using Jason*. John Wiley & Sons.

Bratman, M.; Israel, D.; and Pollack, M. 1991. Plans and resource-bounded practical reasoning. In Cummins, R., and Pollock, J. L., eds., *Philosophy and AI: Essays at the Interface*, 1–22.

Dastani, M.; van Riemsdijk, M. B.; and Meyer, J.-J. C. 2005. Programming multi-agent systems in 3apl. In *Multi-Agent Programming*. 39–67.

Garcia, A., and Simari, G. 2004. Defeasible logic programming: An argumentative approach. *Theory and Practice of Logic Programming* 4:95–138.

García, A.; Rotstein, N.; Tucat, M.; and Simari, G. 2007. An argumentative reasoning service for deliberative agents. In *International Conference on Knowledge Science, Engineering and Management (KSEM)*, 128–139.

Parsons, S.; Sierra, C.; and Jennings, N. R. 1998. Agents that reason and negotiate by arguing. *J. Log. Comput.* 8(3):261–292.

Rahwan, I., and Amgoud, L. 2006. An argumentation based approach for practical reasoning. In *International Joint Conference on Autonomous Agents and Multi Agent Systems*, 347–354.

Rotstein, N. D.; Garcia, A. J.; and Simari, G. R. 2007. Reasoning from desires to intentions: A dialectical framework. In *AAAI Conference on Artificial Intelligence*, 136–141.