Modelling Agent Reasoning in a Logic Programming Framework for Possibilistic Argumentation*

Carlos I. Chesñevar
Computer Science Department
University of Lleida – C/Jaume II, 69
25001 Lleida, Spain
cic@eps.udl.es

Guillermo R. Simari
Dept. of Computer Science & Engineering
Universidad Nacional del Sur – Alem 1253
8000 Bahía Blanca, Argentina
grs@cs.uns.edu.ar

Teresa Alsinet
Computer Science Department
University of Lleida – C/Jaume II, 69
25001 Lleida, Spain
tracy@eps.udl.es

Lluís Godo
AI Research Institute
IIIA – CSIC – Campus UAB
08193 Bellaterra, Spain
godo@iiia.csic.es

Abstract

A well-known problem in multiagent systems (MAS) involves finding an adequate formalization of an agent's knowledge to perform defeasible inferences in a computationally effective way. In the last years, argument-based approaches have proven to be an attractive setting to achieve this goal. Dealing with uncertainty and fuzziness associated with the available knowledge are also common requirements in MAS. Such features, however, are not embedded in most argument-based formalisms. Possibilistic Defeasible Logic Programming (P-DeLP) has recently appeared as an alternative to solve the above problem. P-DeLP is a logic programming language which combines features from argumentation theory and logic programming, incorporating as well the treatment of possibilistic uncertainty and fuzzy knowledge at object-language level. This paper describes how P-DeLP can be applied in the context of formalizing an agent's beliefs and perceptions, along with an argumentative inference procedure to determine which of the agent's beliefs are ultimately accepted (or warranted*).*

KEY WORDS: Agent reasoning, Argumentation, Logic Programming, Uncertainty

1. Introduction and motivations

A well-known problem in multiagent systems (MAS) involves finding an adequate formalization of an agent's knowledge to perform defeasible inferences in a computationally effective way. In the last years, argument-based approaches have proven to be a attractive setting to achieve this goal [28, 22, 24]. Dealing with uncertainty and fuzziness associated with the agent's knowledge are also common requirements when formalizing MAS. Such features, however, are usually not embedded at object-level in argument-based formalisms.

Possibilistic Defeasible Logic Programming (P-DeLP) [14] has recently appeared as an alternative to solve the above problem. P-DeLP is a logic programming language which combines features from argumentation theory and logic programming, incorporating as well the treatment of possibilistic uncertainty and fuzzy knowledge at object-language level. These knowledge representation features are formalized on the basis of PGL [1, 2], a possibilistic logic based on Gödel fuzzy logic. In PGL formulas are built over fuzzy propositional variables and the certainty degree of formulas is expressed with a necessity measure. In a logic programming setting, the proof method for PGL is based on a complete calculus for determining the maximum degree of possibilistic entailment of a fuzzy goal. The top-down proof procedure of P-DeLP is based on the one used in *defeasible logic programming* [17], which has already been integrated in a number of real-world applications such as intelligent web search [11, 13], cluster-

* A slightly different version of this paper (not considering how to model agent reasoning capabilities) was originally published in the Proceedings of the Intl. Conference in Uncertainty in Artificial Intelligence (UAI 2004), pages 76-84.

ing [18], and natural language processing [10], among others.

This paper describes how P-DeLP can be applied in the context of formalizing an agent's beliefs and perceptions, along with an effective argumentative inference procedure to determine which of the agent's beliefs are ultimately accepted (or *warranted*). In our approach, the agent's knowledge will be represented in terms of certainty-weighted formulas. We model the distinction between strict and defeasible knowledge by attaching different certainty weights (weight 1 corresponds to strict knowledge, and weights in the interval $[0, 1)$ correspond to defeasible knowledge). Arguments will be computed as proof trees supporting a given formula (goal) with a certainty weight. These weights will be used to solve conflicts among contradictory goals.

The rest of the paper is structured as follows. First in Section 2 we will discuss the knowledge representation features provided by P-DeLP, including its syntax and semantics at object-language level. Then in Section 3 we present the central notion of *argument* in P-DeLP as well as an associated procedural mechanism for obtaining them. In Section 4 we formalize the notions of attack among arguments and the top-down proof procedure for computing ultimately undefeated arguments (or *warrants*) in P-DeLP. In Section 5 we present a worked example, showing how P-DeLP can be used to model beliefs and reasoning capabilities of an intelligent agent. Finally, in Section 6 we discuss related work and the most important conclusions that have been obtained.

2. The P-DeLP programming language: fundamentals

The P-DeLP language $\mathcal{L}$ is defined from a set of ground fuzzy atoms (fuzzy propositional variables) $\{p, q, \ldots\}$ together with the connectives $\{\sim, \wedge, \leftarrow\}$. The symbol $\sim$ stands for *negation*. A *literal* $L \in \mathcal{L}$ is a ground (fuzzy) atom q or a negated ground (fuzzy) atom $\sim q$, where q is a ground (fuzzy) propositional variable. A *rule* in $\mathcal{L}$ is a formula of the form $Q \leftarrow L_1 \wedge \ldots \wedge L_n$, where $Q, L_1, \ldots, L_n$ are literals in $\mathcal{L}$. When $n = 0$, the formula $Q \leftarrow$ is called a *fact* and simply written as Q. In the following, capital and lower case letters will denote literals and atoms in $\mathcal{L}$, respectively.

Fuzzy propositions provide us with a suitable representation model in situations where our agent has vague or imprecise information about the real world. For instance, the fuzzy statement "the engine speed is low" can be nicely represented by the fuzzy proposition $engine\_speed(low)$, where low is a fuzzy set defined over the domain `revs_per_minute`, say an interval [0, 6000]. In the case low actually denotes a crisp interval of number of revolutions, the above proposition is to be interpreted as "$\exists x \in low$ such that the engine speed is x". In the case low denotes a fuzzy interval with a membership function $\mu_{low} : [0, 6000] \rightarrow [0, 1]$, the above proposition is interpreted in possibilistic terms as "for each $\alpha \in [0, 1]$, $\exists x \in [\mu_{low}]_\alpha$ such that the engine speed is x, is certain with a necessity of at least $1 - \alpha$", where $[\mu_{low}]_\alpha$ denotes the α-cut of μ_{low}, the set of values defined as $[\mu_{low}]_\alpha = \{u \in [0, 6000] \mid \mu_{low}(u) \geq \alpha\}$. So, fuzzy propositions can be seen as (flexible) restrictions on an existential quantifier [16]. It must be noted that negation in P-DeLP is used to contradict statements represented by fuzzy propositions. For instance, in the case low denotes a crisp interval of revolutions, $\sim engine\_speed(low)$ is interpreted as "$\neg[\exists x \in low$ such that the engine speed is x]", or equivalently "$\forall x \in low$, x does not correspond with the engine speed".

Definition 1 (P-DeLP formulas) *The set Wffs($\mathcal{L}$) of wffs in $\mathcal{L}$ are* facts *and* rules *built over the literals of $\mathcal{L}$. A* certainty-weighted *clause in $\mathcal{L}$, or simply* weighted clause*, is a pair of the form (φ, α), where φ is a wff in $\mathcal{L}$ and $\alpha \in [0, 1]$ expresses a lower bound for the certainty of φ in terms of a necessity measure.*

The P-DeLP language is based on Possibilistic Gödel Logic or PGL [1]. There are different reasons for choosing PGL as the underlying logic to model both uncertainty and fuzziness.[1] A key feature of PGL is that it can be extended with a partial matching mechanism between fuzzy propositional variables based on a necessity-like measure which preserves completeness for a particular class of formulas [2].

Since wffs in $\mathcal{L}$ involve fuzzy propositional variables, the underlying semantics of P-DeLP is many-valued[2] instead of Boolean and possibilistic models are defined as possibility distributions over all the possible set of many-valued interpretations. A detailed description about the many-valued and possibilistic semantics of P-DeLP can be found in [14].

The proof method for P-DeLP formulas, written $\vdash$, is defined by derivation based on the following triviality axiom and a particular instance of the generalized modus ponens rule:

Axiom: $(\varphi, 0)$

Generalized modus ponens (GMP):

$$\frac{(L_0 \leftarrow L_1 \wedge \cdots \wedge L_k, \gamma) \quad (L_1, \beta_1), \ldots, (L_k, \beta_k)}{(L_0, \min(\gamma, \beta_1, \ldots, \beta_k))}$$

1 See [14] for discussion.

2 Actually, it corresponds to the well known Gödel infinitely-valued calculi.

Due to the negation connective of P-DeLP the GMP rule allows us to define a complete calculus for determining the maximum degree of possibilistic entailment of a goal from a set of clauses if we restrict ourselves to sets of clauses satisfying the following *forward reasoning* constraint: The possibilistic entailment degree of a goal Q from a set of clauses Γ must be univocally determined by those clauses of Γ having Q in their head or leading to one of these clauses by resolving them with other clauses by applying the GMP rule.

A detailed description of soundness and completeness for P-DeLP clauses satisfying the forward reasoning constraint can be found in [14]. In the sequel we will restrict ourselves to weighed clauses in $\mathcal{L}$ satisfying the forward reasoning constraint.

3. Argumentation in P-DeLP

In P-DeLP we distinguish between *certain* and *uncertain* clauses. A clause (φ, α) will be referred as certain if $\alpha = 1$ and uncertain, otherwise. Moreover, a set of clauses Γ will be deemed as *contradictory*, denoted $\Gamma \vdash \bot$, if $\Gamma \vdash (q, \alpha)$ and $\Gamma \vdash (\sim q, \beta)$, with $\alpha > 0$ and $\beta > 0$, for some atom q in $\mathcal{L}$.

A P-DeLP program is a set of weighted clauses in $\mathcal{L}$ in which we distinguish certain from uncertain information. As additional requirement, certain knowledge is required to be non-contradictory. Formally:

Definition 2 (P-DeLP program) *A P-DeLP program $\mathcal{P}$ (or just program $\mathcal{P}$) is a pair (Π, Δ), where Π is a non-contradictory finite set of certain clauses, and Δ is a finite set of uncertain clauses.*

Definition 3 (Argument. Subargument) *Given a P-DeLP program $\mathcal{P} = (\Pi, \Delta)$, a set $\mathcal{A} \subseteq \Delta$ of uncertain clauses is an* argument *for a goal Q with necessity degree $\alpha > 0$, denoted $\langle \mathcal{A}, Q, \alpha \rangle$, iff:*

1. $\Pi \cup \mathcal{A} \vdash (Q, \alpha)$;
2. $\Pi \cup \mathcal{A}$ *is non contradictory; and*
3. *There is no $\mathcal{A}_1 \subset \mathcal{A}$ such that $\Pi \cup \mathcal{A}_1 \vdash (Q, \beta)$, $\beta > 0$.*

Let $\langle \mathcal{A}, Q, \alpha \rangle$ and $\langle \mathcal{S}, R, \beta \rangle$ be two arguments. We will say that $\langle \mathcal{S}, R, \beta \rangle$ is a subargument *of $\langle \mathcal{A}, Q, \alpha \rangle$ iff $\mathcal{S} \subseteq \mathcal{A}$. Notice that the goal R may be a subgoal associated with the goal Q in the argument $\mathcal{A}$.*

Note that from the definition of argument, it follows that on the basis of a P-DeLP program $\mathcal{P}$ there may exist different arguments $\langle \mathcal{A}_1, Q, \alpha_1 \rangle$, $\langle \mathcal{A}_2, Q, \alpha_2 \rangle$, ..., $\langle \mathcal{A}_k, Q, \alpha_k \rangle$ supporting a given goal Q, with (possibly) different necessity degrees $\alpha_1, \alpha_2, \ldots, \alpha_k$.

It must be remarked that the three conditions in the above definition are inherited from similar definitions in the argumentation literature [26, 7, 12]. Given a program $\mathcal{P} = (\Pi, \Delta)$, we define the following procedural rules (based on the P-DeLP calculus [14]) to construct arguments:

1) Building arguments from facts (INTF)

$$\frac{(Q, 1)}{\langle \emptyset, Q, 1 \rangle}$$

for any weighted fact $(Q, 1) \in \Pi$

$$\frac{(Q, \alpha),\ \Pi \cup \{(Q, \alpha)\} \nvdash \bot, \text{ with } \alpha < 1,}{\langle \{(Q, \alpha)\}, Q, \alpha \rangle}$$

for any weighted fact $(Q, \alpha) \in \Delta$

2) Building Arguments by GMP (MPA):

$$\frac{\begin{array}{c}\langle \mathcal{A}_1, L_1, \alpha_1 \rangle \quad \langle \mathcal{A}_2, L_2, \alpha_2 \rangle \quad \ldots \langle \mathcal{A}_k, L_k, \alpha_k \rangle \\ (L_0 \leftarrow L_1 \wedge L_2 \wedge \ldots \wedge L_k, \gamma) \text{ with } \gamma < 1 \\ \Pi \cup \{(L_0 \leftarrow L_1 \wedge L_2 \wedge \ldots \wedge L_k, \gamma)\} \cup \bigcup_{i=1}^{k} \mathcal{A}_i \nvdash \bot\end{array}}{\langle \bigcup_{i=1}^{k} \mathcal{A}_i \cup \{(L_0 \leftarrow L_1 \wedge L_2 \wedge \ldots \wedge L_k, \gamma)\}, L_0, \beta \rangle}$$

for any weighted rule $(L_0 \leftarrow L_1 \wedge L_2 \wedge \ldots \wedge L_k, \gamma) \in \Delta$, with the necessity degree $\beta = min(\alpha_1, \ldots, \alpha_k, \gamma)$.

3) Extending Arguments (EAR):

$$\frac{\langle \mathcal{A}, P, \alpha \rangle \quad \Pi \cup \{(P, \alpha)\} \vdash (Q, \alpha)}{\langle \mathcal{A}, Q, \alpha \rangle}$$

for any argument $\langle \mathcal{A}, P, \alpha \rangle$, whenever (Q, α) follows from $\Pi \cup \{(P, \alpha)\}$.

The basic idea with the argument construction procedure is to keep a trace of the set $\mathcal{A}$ of all uncertain information used to derive a given goal Q with necessity degree α. Appropriate preconditions ensure that the proof obtained always follows condition 2 in Def. 3.

4. Computing Warrant in P-DeLP

Given a program $\mathcal{P}$, it can be the case that there exist conflicting arguments $\langle \mathcal{A}_1, Q, \alpha_1 \rangle$ and $\langle \mathcal{A}_2, \sim Q, \alpha_2 \rangle$.[3] Such conflict among arguments will be formalized by the notions of counterargument and defeat presented next.

Definition 4 (Counterargument) *Let $\mathcal{P}$ be a program, and let $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ and $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$ be two arguments wrt $\mathcal{P}$. We will say that $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ counterargues $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$ iff there exists a subargument (called* disagreement subargument*) $\langle \mathcal{S}, Q, \beta \rangle$ of $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$ such that $\Pi \cup \{(Q_1, \alpha_1), (Q, \beta)\}$ is contradictory.*

Defeat among arguments involves a *preference criterion* on conflicting arguments, defined on the basis of necessity measures associated with arguments.

---

3 For a given goal Q, we write $\sim Q$ as an abbreviation to denote "$\sim q$" if $Q \equiv q$ and "q" if $Q \equiv \sim q$.

Definition 5 (Preference criterion $\succeq$) *Let $\mathcal{P}$ be a P-DeLP program, and and let $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ and $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$ be conflicting arguments in $\mathcal{P}$. We will say that $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ is* preferred *over $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$ (denoted $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle \succeq \langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$) iff $\alpha_1 \geq \alpha_2$. If it is the case that $\alpha_1 > \alpha_2$, then we will say that $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ is* strictly preferred *over $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$, denoted $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle \succ \langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$. Otherwise, if $\alpha_1 = \alpha_2$ we will say that both arguments are* equi-preferred, *denoted $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle \approx \langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$.*

Definition 6 (Defeat) *Let $\mathcal{P}$ be a P-DeLP program, and let $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ and $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$ be two arguments in $\mathcal{P}$. We will say that $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$* defeats *$\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$ (or equivalently $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ is a* defeater *for $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$) iff*

1. *$\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ counterargues $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$ with disagreement subargument $\langle \mathcal{A}, Q, \alpha \rangle$.*
2. *Either it holds that $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle \succ \langle \mathcal{A}, Q, \alpha \rangle$, in which case $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ will be called a* proper defeater *for $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$, or $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle \approx \langle \mathcal{A}, Q, \alpha \rangle$, in which case $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ will be called a* blocking defeater *for $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$.*

As in most argumentation systems [12, 23], P-DeLP relies on an exhaustive dialectical analysis which allows to determine if a given argument is *ultimately* undefeated (or *warranted*) wrt a program $\mathcal{P}$. An *argumentation line* starting in an argument $\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle$ is a sequence $[\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle, \langle \mathcal{A}_1, Q_1, \alpha_1 \rangle, \ldots, \langle \mathcal{A}_n, Q_n, \alpha_n \rangle, \ldots]$ that can be thought of as an exchange of arguments between two parties, a *proponent* (evenly-indexed arguments) and an *opponent* (oddly-indexed arguments). In order to avoid *fallacious* reasoning, argumentation theory imposes additional constraints on such an argument exchange to be considered rationally acceptable wrt a P-DeLP program $\mathcal{P}$, namely:

1. **Non-contradiction:** given an argumentation line λ, the set of arguments of the proponent (resp. opponent) should be *non-contradictory* wrt $\mathcal{P}$.
2. **No circular argumentation:** no argument $\langle \mathcal{A}_j, Q_j, \alpha_j \rangle$ in λ is a sub-argument of an argument $\langle \mathcal{A}_i, Q_i, \alpha_i \rangle$ in λ, $i < j$.
3. **Progressive argumentation:** every blocking defeater $\langle \mathcal{A}_i, Q_i, \alpha_i \rangle$ in λ is defeated by a proper defeater $\langle \mathcal{A}_{i+1}, Q_{i+1}, \alpha_{i+1} \rangle$ in λ.

An argumentation line satisfying the above restrictions is called *acceptable*, and can be proven to be finite. Given a program $\mathcal{P}$ and an argument $\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle$, the set of all acceptable argumentation lines starting in $\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle$ accounts for a whole dialectical analysis for $\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle$ (i.e. all possible dialogues rooted in $\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle$, formalized as a *dialectical tree*, denoted $\mathcal{T}_{\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle}$).

Nodes in a dialectical tree $\mathcal{T}_{\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle}$ can be marked as *undefeated* and *defeated* nodes (U-nodes and D-nodes, resp.). A dialectical tree will be marked as an AND-OR tree: all leaves in $\mathcal{T}_{\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle}$ will be marked U-nodes (as they have no defeaters), and every inner node is to be marked as *D-node* iff it has at least one U-node as a child, and as *U-node* otherwise. An argument $\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle$ is ultimately accepted as valid (or *warranted*) wrt a DeLP program $\mathcal{P}$ iff the root of $\mathcal{T}_{\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle}$ is labelled as *U-node*.

Definition 7 (Warrant) *Given a program $\mathcal{P}$, and a goal Q, we will say that Q is warranted wrt $\mathcal{P}$ with a necessity degree α iff there exists a warranted argument $\langle \mathcal{A}, Q, \alpha \rangle$.*

For a given program $\mathcal{P}$, a P-DeLP interpreter will find an answer for a goal Q by determining whether Q is supported by some warranted argument $\langle \mathcal{A}, Q, \alpha \rangle$. Different doxastic attitudes are distinguished when providing an answer for the goal Q according to the associated status of warrant.

1. Answer YES (with a necessity α) whenever Q is supported by a warranted argument $\langle \mathcal{A}, Q, \alpha \rangle$;
2. Answer NO (with a necessity α) whenever for $\sim Q$ is supported by a warranted argument $\langle \mathcal{A}, \sim Q, \alpha \rangle$;
3. Answer UNDECIDED whenever (1) and (2) do not hold.

5. Modelling an intelligent agent in P-DeLP

Next we will present an example which illustrates how P-DeLP can be used to model the beliefs and reasoning capabilities of an agent. Consider an intelligent agent controlling an engine with three switches $sw1$, $sw2$ and $sw3$. These switches regulate different features of the engine, such as pumping system, speed, etc. This agent may have the following certain and uncertain knowledge about how this engine works, e.g.:

- If the pump is clogged, then the engine gets no fuel.
- When $sw1$ is on, normally fuel is pumped properly.
- When fuel is pumped properly, fuel seems to work ok.
- When $sw2$ is on, usually oil is pumped.
- When oil is pumped, usually it works ok.
- When there is oil and fuel, usually the engine works ok.
- When there is heat, then the engine is usually not ok.
- When there is heat, normally there are oil problems.
- When fuel is pumped and speed is low, then there are reasons to believe that the pump is clogged.
- When $sw2$ is on, usually speed is low.
- When $sw2$ and $sw3$ are on, usually speed is not low.
- When $sw3$ is on, usually fuel is ok.

(1)	$(\sim fuel\_ok \leftarrow pump\_clog, 1)$
(2)	$(sw1, 1)$
(3)	$(sw2, 1)$
(4)	$(sw3, 1)$
(5)	$(heat, 1)$
(6)	$(pump\_fuel \leftarrow sw1, 0.6)$
(7)	$(fuel\_ok \leftarrow pump\_fuel, 0.3)$
(8)	$(pump\_oil \leftarrow sw2, 0.8)$
(9)	$(oil\_ok \leftarrow pump\_oil, 0.8)$
(10)	$(engine\_ok \leftarrow fuel\_ok \wedge oil\_ok, 0.3)$
(11)	$(\sim engine\_ok \leftarrow heat, 0.95)$
(12)	$(\sim oil\_ok \leftarrow heat, 0.9)$
(13)	$(pump\_clog \leftarrow pump\_fuel \wedge low\_speed, 0.7)$
(14)	$(low\_speed \leftarrow sw2, 0.8)$
(15)	$(\sim low\_speed \leftarrow sw2, sw3, 0.8)$
(16)	$(fuel\_ok \leftarrow sw3, 0.9)$

Figure 1. P-DeLP program $\mathcal{P}_{eng}$

Suppose also that the agent knows some particular facts: $sw1$, $sw2$ and $sw3$ are on, and there is heat. The knowledge of such an agent can be modelled by the program $\mathcal{P}_{eng}$ shown in Fig. 1, where the finite set of certain clauses (i.e. Π) is from clause (1) to (5), and the finite set of uncertain clauses (i.e. Δ) is from clause (6) to (16). Note that uncertainty is assessed in terms of different necessity measures. From the P-DeLP program in Fig. 1 different *arguments* can be derived using the procedural rules defined in Section 3. Thus, for example, the argument $\langle \mathcal{B}, fuel\_ok, 0.3\rangle$ can be derived from $\mathcal{P}_{eng}$ as follows:

i)	$\langle \emptyset, sw1, 1\rangle$	from (2) via INTF.
ii)	$\langle \mathcal{B}', pump\_fuel, 0.6\rangle$	from (6) and i) via MPA.
iii)	$\langle \mathcal{B}, fuel\_ok, 0.3\rangle$	from (7) and ii) via MPA.

where

$$\begin{array}{rcl} \mathcal{B}' & = & \{(pump\_fuel \leftarrow sw1, 0.6)\} \text{ and} \\ \mathcal{B} & = & \{\ (pump\_fuel \leftarrow sw1, 0.6)\ ; \\ & & (fuel\_ok \leftarrow pump\_fuel, 0.3)\}. \end{array}$$

Similarly, an argument $\langle \mathcal{C}, oil\_ok, 0.8\rangle$ can be derived from $\mathcal{P}_{eng}$ using the rules (3), (8) and (9) via INTC, MPA, and MPA, resp., with

$$\begin{array}{rcl} \mathcal{C} & = & \{(pump\_oil \leftarrow sw2, 0.8); \\ & & (oil\_ok \leftarrow pump\_oil, 0.8)\}.^4 \end{array}$$

Finally, an argument $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$ can be derived from $\mathcal{P}_{eng}$ as follows:

i)	$\langle \mathcal{B}, fuel\_ok, 0.3\rangle$	as shown above.
ii)	$\langle \mathcal{C}, oil\_ok, 0.8\rangle$	as shown above.
iii)	$\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$	from i), ii), 10) via MPA.

where $\mathcal{A}_1$=$\{(engine\_ok \leftarrow fuel\_ok \wedge oil\_ok, 0.3)\ \} \cup \mathcal{B} \cup \mathcal{C}$. Note that arguments $\langle \mathcal{C}, oil\_ok, 0.8\rangle$ and $\langle \mathcal{B}, fuel\_ok, 0.3\rangle$ are subarguments (see Def. 3) of $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$.

Let us assume that the agent is in charge of controlling the engine, answering queries from other agents (e.g. a supervisor agent) about the status of the engine. For instance, the query $? - (engine\_ok, 0.8)$ corresponds with proving whether the engine works ok with a certainty degree of at least 0.8. In order to answer this query, the agent will apply the procedure described in the previous sections: first will compute an argument supporting $engine\_ok$, and then will perform a recursive analysis of defeaters for these arguments, computing its associated dialectical tree.

In this particular example, the agent will find an argument supporting $engine\_ok$, namely $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$. A counteragument (see Def. 4) for $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$ can be found, namely $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6\rangle$, obtained from (2), (3), (6), (14), (13) and (1) by applying INTF, INTF, MPA, MPA, MPA, and EAR, resp., with

$$\begin{array}{rcl} \mathcal{A}_2 & = & \{\ (pump\_fuel \leftarrow sw1, 0.6), \\ & & (low\_speed \leftarrow sw2, 0.8), \\ & & (pump\_clog \leftarrow pump\_fuel \wedge low\_speed, 0.7)\}. \end{array}$$

The argument $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6\rangle$ is a counterargument for the argument $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$ as there exists a subargument $\langle \mathcal{B}, fuel\_ok, 0.3\rangle$ in $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$ such that the set $\Pi \cup \{(fuel\_ok, 0.3), (\sim fuel\_ok, 0.6)\}$ is contradictory. Note as well that $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6\rangle$ is a proper defeater (Def. 6) for $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$, as $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6\rangle$ counterargues the argument $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$ with disagreement subargument $\langle \mathcal{B}, fuel\_ok, 0.3\rangle$, and $0.6 > 0.3$.

As the defeater $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6\rangle$ is also an argument, a recursive analysis can be carried out by the agent, computing an *argumentation line* rooted in $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$. In fact, note that $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6\rangle$ has the subargument $\langle \mathcal{A}_2', low\_speed, 0.8\rangle$, with $\mathcal{A}_2' = \{(low\_speed \leftarrow sw2, 0.8)\}$. From the program $\mathcal{P}_{eng}$ (Fig. 1) a blocking defeater for the argument $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6\rangle$ can be derived, namely $\langle \mathcal{A}_3, \sim low\_speed, 0.8\rangle$, obtained from (3), (4) and (15) via INTF, INTF and MPA, respectively. In this case we have:

$$\mathcal{A}_3 \quad = \quad \{\ (\sim low\_speed \leftarrow sw2, sw3, 0.8)\ \}$$

Note that this third defeater can be thought of as an answer of the proponent to the opponent, reinstating the first argument $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$, as it defeats the opponent's defeater $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6\rangle$. The above situation can be expressed in the following argumentation line:[5]

$$[\ \langle \mathcal{A}_1, engine\_ok, 0.3\rangle, \langle \mathcal{A}_2, \sim fuel\_ok, 0.6\rangle, \langle \mathcal{A}_3, \sim low\_speed, 0.8\rangle\ ]$$

In order for the preceding analysis to be exhaustive, every possible argumentation line rooted in $\langle \mathcal{A}_1, engine\_ok, 0.3\rangle$ should be analysed.

5 Note that the proponent's last defeater in the above sequence could be on its turn defeated by a blocking defeater $\langle \mathcal{A}_2', low\_speed, 0.8\rangle$, resulting in
$[\langle \mathcal{A}_1, engine\_ok, 0.3\rangle, \langle \mathcal{A}_2, \sim fuel\_ok, 0.6\rangle, \langle \mathcal{A}_3, \sim low\_speed, 0.8\rangle, \langle \mathcal{A}_2', low\_speed, 0.8\rangle \ldots]$
However, such line is *not acceptable*, as it violates the condition of non-circular argumentation.

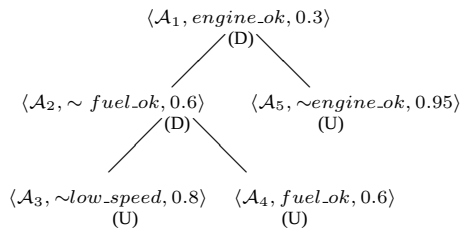

Figure 2. Dialectical tree for $\langle \mathcal{A}_1, engine\_ok, 0.3 \rangle$

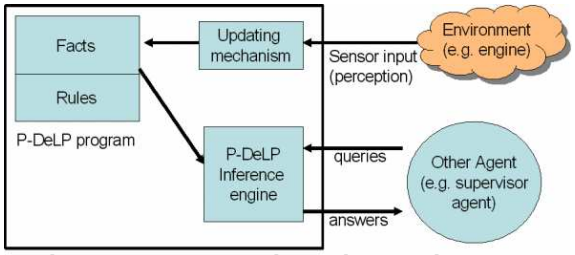

Figure 3. A P-DeLP-based agent in a MAS context

In this particular case, note that the argument $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6 \rangle$ has a second (blocking) defeater $\langle \mathcal{A}_4, fuel\_ok, 0.6 \rangle$, computed from (4), (16) via INTF and MPA, respectively. The argument $\langle \mathcal{A}_1, engine\_ok, 0.3 \rangle$ has also a second defeater $\langle \mathcal{A}_5, \sim engine\_ok, 0.95 \rangle$, computed from (5), (11) via INTF and MPA, respectively. In this case we have:

$$\mathcal{A}_4 = \{ (fuel\_ok \leftarrow sw3, 0.9) \}$$
$$\mathcal{A}_5 = \{ (\sim engine\_ok \leftarrow heat, 0.8) \}$$

There are no more arguments to consider. As a consequence, there are three acceptable argumentation lines rooted in $\langle \mathcal{A}_1, engine\_ok, 0.3 \rangle$, namely:

- [$\langle \mathcal{A}_1, engine\_ok, 0.3 \rangle$, $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6 \rangle$, $\langle \mathcal{A}_3, \sim low\_speed, 0.8 \rangle$]
- [$\langle \mathcal{A}_1, engine\_ok, 0.3 \rangle$, $\langle \mathcal{A}_2, \sim fuel\_ok, 0.6 \rangle$, $\langle \mathcal{A}_4, fuel\_ok, 0.9 \rangle$]
- [$\langle \mathcal{A}_1, engine\_ok, 0.3 \rangle$, $\langle \mathcal{A}_5, \sim engine\_ok, 0.95 \rangle$]

Fig. 2 shows the corresponding dialectical tree $\mathcal{T}_{\langle \mathcal{A}_1, engine\_ok, 0.3 \rangle}$ rooted in the argument $\langle \mathcal{A}_1, engine\_ok, 0.3 \rangle$. The marking procedure defined in Section 4 results in the nodes of the tree $\mathcal{T}_{\langle \mathcal{A}_1, engine\_ok, 0.3 \rangle}$ marked as U-nodes and D-nodes as shown in the figure.

The argument $\langle \mathcal{A}_1, engine\_ok, 0.3 \rangle$ is the only possible argument the agent can compute supporting the query $engine\_ok$. As this argument is ultimately defeated, it is not warranted (Def. 7). On the contrary, the agent can compute there exists an argument $\langle \mathcal{A}_5, \sim engine\_ok, 0.95 \rangle$ supporting $\sim engine\_ok$, and such argument has no defeaters, and therefore it is warranted. The answer to goal $engine\_ok$ will therefore be NO, with $\alpha = 0.95$.

In a MAS context, intelligent agents will encode their knowledge about the world using a P-DeLP program. Figure 3 outlines the different elements associated with a P-DeLP-based agent. Clearly, our agent will be usually performing its activities in a dynamic environment, so that it should also be able to reason, plan, and act according to new perceptions from the outside world. Such perceptions will be sensed by the agent, integrating them into its current beliefs. For the sake of simplicity, we will assume that such perceptions constitute new facts to be added to the agent's knowledge base. As already stated in the introduction, fuzzy propositions provide us with a suitable representation model as our agent will probably have vague or imprecise information about the real world, as its sensors are not perfect devices.

Defining a generic procedure for updating the agent's knowledge base is not easy, as completely new incoming information (e.g. facts with new predicate names) might result in the strict knowledge Π becoming contradictory (see Def. 2). In some particular cases the agent will only perceive changes in the necessity measure of the already known facts (e.g. the agent has a fact $(heat, 1)$ in the knowledge base, but the sensed temperature has changed, modelled by a new fact $(heat, 0.8)$). In such cases, a simple but effective strategy can be applied, similar as the one suggested in [8]. We will make the assumption that new perceptions always supersede old ones, so that a if a new perception $(p, value)$ is sensed at time t, and the agent has already a fact $(p, value')$ in its strict knowledge base Π, then the updated knowledge base will be computed as $\Pi \setminus \{(p, value')\} \cup \{(p, value)\}$.

6. Conclusions and related work

In this paper we have described how to model an agent's beliefs and perceptions using P-DeLP. A salient feature of P-DeLP is that it is based on two logical frameworks that have already been implemented (namely PGL [2] and DeLP [17]). Several features leading to efficient implementations of the argumentative proof procedure described in this paper have been also recently studied, particularly those related to comparing conflicting arguments by specificity [27] and defining transformation properties for DeLP programs [9].

In this context, P-DeLP keeps all the original features of DeLP while incorporating more expressivity and representation capabilities by means of possibilistic uncertainty and fuzzy knowledge. One particularly interesting feature of P-DeLP is the possibility of

defining aggregated preference criteria by combining the necessity measures associated with arguments with other syntax-based criteria (e.g. specificity [26, 27]).

In the last years the development of combined approaches based on qualitative reasoning and uncertainty has deserved much research work [21]. Preference-based approaches to argumentation have been developed, many of them oriented towards formalizing conflicting desires in multiagent systems [3, 4, 5]. In contrast with these preference-based approaches, the P-DeLP framework involves necessity measures explicitly attached to fuzzy formulas and the proof procedure of the underlying possibilistic fuzzy logic is used for computing the necessity measure for arguments.

There have also been generic approaches connecting defeasible reasoning and possibilistic logic (e.g.[6]), and recently a number of hybrid approaches connecting argumentation and uncertainty have been developed. Probabilistic Argumentation Systems [19, 20] use probabilities to compute degrees of support and plausibility of goals, related to Dempster-Shafer belief and plausibility functions. However this is not a dialectics-based framework as opposed to the one presented in this paper. In [25] a fuzzy argumentation system based on extended logic programming is proposed. In contrast with our framework, this approach relies only on fuzzy values applied to literals and there is no explicit treatment of possibilistic uncertainty.

Part of our current research work will be developed into different directions. First, we will extend the existing implementation of DeLP to incorporate the new features of P-DeLP. Second, we will apply the resulting implementation of P-DeLP to improve existing real-world applications of DeLP and to develop new ones. Finally, we will analyze extending P-DeLP to first order. It must be remarked that the Generalized Modus Ponens rule used in P-DeLP is syntactically the same as the one used in possibilistic logic [15]. As a consequence, to implement the machinery of P-DeLP the underlying possibilistic fuzzy logic PGL can be replaced by the possibilistic logic. The advantage of this approach is that the current logic programming system can be extended to first order, incorporating fuzzy unification between fuzzy constants [2].

Acknowledgements This research was partially supported by CYCYT Projects LOGFAC (TIC2001-1577-C03-01/03) and SOFTSAT (TIC2003-00950), by Ramón y Cajal Program (Ministerio de Ciencia y Tecnología, Spain), by CONICET (Argentina), by the Secretaría General de Ciencia y Tecnología de la Universidad Nacional del Sur and by Agencia Nacional de Promoción Científica y Tecnológica (PICT 2002 No. 13096).

References

[1] T. Alsinet and L. Godo. A complete calculus for possibilistic logic programming with fuzzy propositional variables. In *Proc. of the UAI-2000 Conference*, pages 1–10, 2000.

[2] T. Alsinet and L. Godo. A proof procedure for possibilistic logic programming with fuzzy constants. In *Proc. of the ECSQARU-2001 Conference*, pages 760–771, 2001.

[3] L. Amgoud. A formal framework for handling conflicting desires. In *Proc. of the ECSQARU-2003 Conference*, pages 552–563, 2003.

[4] L. Amgoud and C. Cayrol. Inferring from inconsistency in preference-based argumentation frameworks. *Journal of Automated Reasoning*, 29:125–169, 2002.

[5] L. Amgoud and H. Prade. Reaching agreement through argumentation: A possibilistic approach. In *Proc. of the KR 2004 Conference. Whistler, Canada*, pages 175–182, June 2004.

[6] S. Benferhat, D. Dubois, and H. Prade. The possibilistic handling of irrelevance in exception-tolerant reasoning. *Annals of Math. and AI*, 35:29–61, 2002.

[7] P. Besnard and A. Hunter. A logic-based theory of deductive arguments. *Artificial Intelligence*, 128:203–235, 2001.

[8] M. Capobianco, C. Chesñevar, and G. Simari. An argument-based framework to model an agent's beliefs in a dynamic environment. In *Proc. of the First International Workshop on Argumentation in Multiagent Systems. AAMAS 2004 Conference, New York, USA*, pages 163–178, July 2004.

[9] C. Chesñevar, J. Dix, F. Stolzenburg, and G. Simari. Relating Defeasible and Normal Logic Programming through Transformation Properties. *Theoretical Computer Science*, 290(1):499–529, 2003.

[10] C. Chesñevar and A. Maguitman. An Argumentative Approach to Assessing Natural Language Usage based on the Web Corpus. In *Proc. of the ECAI-2004 Conference. Valencia, Spain*, pages 581–585, Aug. 2004.

[11] C. Chesñevar and A. Maguitman. ARGUENET: An Argument-Based Recommender System for Solving Web Search Queries. In *Proc. of the 2nd IEEE Intl. IS-2004 Conference. Varna, Bulgaria*, pages 282–287, June 2004.

[12] C. Chesñevar, A. Maguitman, and R. Loui. Logical Models of Argument. *ACM Computing Surveys*, 32(4):337–383, Dec. 2000.

[13] C. Chesñevar, A. Maguitman, and G. Simari. A first approach to argument-based recommender systems based on defeasible logic programming. In *Proc. of the 10th Intl. Workshop on Non-Monotonic Reasoning (NMR-2004). Whistler, Canada*, pages 109–117, June 2004.

[14] C. I. Chesñevar, G. Simari, T. Alsinet, and L. Godo. A Logic Programming Framework for Possibilistic Argumentation with Vague Knowledge. In *Proc. of the Intl. Conference in Uncertainty in Artificial Intelligence (UAI 2004). Banff, Canada*, pages 76–84, July 2004.

[15] D. Dubois, J. Lang, and H. Prade. Possibilistic logic. In D. Gabbay, C. Hogger, and J. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 3, Nonmonotonic Reasoning and Uncertain Reasoning, pages 439–513. Claredon Press, 1994.

[16] D. Dubois, H. Prade, and S. Sandri. Possibilistic logic with fuzzy constants and fuzzily restricted quantifiers. In F. Arcelli and T. Martin, editors, *Logic Programming and Soft Computing*, chapter 4, pages 69–90. Research Studies Press, 1998.

[17] A. García and G. Simari. Defeasible Logic Programming: An Argumentative Approach. *Theory and Practice of Logic Programming*, 4(1):95–138, 2004.

[18] S. Gómez and C. Chesñevar. A Hybrid Approach to Pattern Classification Using Neural Networks and Defeasible Argumentation. In *Proc. of 17th Intl. FLAIRS Conference. Miami, Florida, USA*, pages 393–398. American Association for Artificial Intelligence, May 2004.

[19] R. Haenni, J. Kohlas, and N. Lehmann. Probabilistic argumentation systems. Handbook of Defeasible Reasoning and Uncertainty Management Systems, 2000.

[20] R. Haenni and N. Lehmann. Probabilistic Argumentation Systems: a New Perspective on Dempster-Shafer Theory. *Int. J. of Intelligent Systems*, 1(18):93–106, 2003.

[21] S. Parsons. *Qualitative Methods for Reasoning under Uncertainty*. MIT Press, 2001.

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

[23] H. Prakken and G. Vreeswijk. Logical Systems for Defeasible Argumentation. In D. Gabbay and F.Guenther, editors, *Handbook of Phil. Logic*, pages 219–318. Kluwer, 2002.

[24] I. Rahwan, S. Ramchurn, N. Jennings, P. McBurney, S. Parsons, and L. Sonenberg. Argumentation-based negotiation. *Knowledge Engineering Review*, 18(4), 2003.

[25] R. Schweimeier and M. Schroeder. Fuzzy argumentation and extended logic programming. In *Proceedings of ECSQARU Workshop Adventures in Argumentation (Toulouse, France)*, Sept. 2001.

[26] G. Simari and R. Loui. A Mathematical Treatment of Defeasible Reasoning and its Implementation. *Art. Intelligence*, 53:125–157, 1992.

[27] F. Stolzenburg, A. García, C. Chesñevar, and G. Simari. Computing Generalized Specificity. *J. of Non-Classical Logics*, 13(1):87–113, 2003.

[28] M. J. Wooldridge. *Introduction to Multiagent Systems*. John Wiley and Sons, 2002.