An argumentation framework with uncertainty management designed for dynamic environments

Marcela Capobianco and Guillermo R. Simari

Artificial Intelligence Research and Development Laboratory
Department of Computer Science and Engineering
Universidad Nacional del Sur – Av. Alem 1253, (8000) Bahía Blanca Argentina
Email: {*mc,grs*}*@cs.uns.edu.ar*

Abstract. Nowadays, data intensive applications are in constant demand and there is need of computing environments with better intelligent capabilities than those present in today's Database Management Systems (DBMS). To build such systems we need formalisms that can perform complicate inferences, obtain the appropriate conclusions, and explain the results. Research in argumentation could provide results in this direction, providing means to build interactive systems able to reason with large databases and/or different data sources.
In this paper we propose an argumentation system able to deal with explicit uncertainty, a vital capability in modern applications. We have also provided the system with the ability to seamlessly incorporate uncertain and/or contradictory information into its knowledge base, using a modular upgrading and revision procedure

1 Introduction and motivations

Nowadays, data intensive applications are in constant demand and there is need of computing environments with better intelligent capabilities than those present in today's Database Management Systems (DBMS). Recently, there has been progress in developing efficient techniques to store and retrieve data, and many satisfactory solutions have been found for the associated problems. However, the problem of how to *understand* and *interpret* a large amount of information remains open, particularly when this information is uncertain, imprecise, and/or inconsistent. To do this we need formalisms that can perform complicate inferences, obtain the appropriate conclusions, and explain the results.

Research in argumentation could provide results in this direction, providing means to build interactive systems able to reason with large databases and/or different data sources, given that argumentation has been successfully used to develop tools for common sense reasoning [8, 4, 14].

Nevertheless, there exists important issues that need to be addressed to use argumentation in these kind of practical applications. A fundamental one concerns the quality of the information expected by argumentation systems: most of them are unable to deal with explicit uncertainty which is a vital capability in

modern applications. Here, we propose an argumentation-based system that addresses this problem, incorporating possibilistic uncertainty into the framework following the approach in [9]. We have also provided the system with the ability to seamlessly incorporate uncertain and/or contradictory information into its knowledge base, using a modular upgrading and revision procedure.

This paper is organized as follows. First, we present the formal definition of our argumentation framework showing its fundamental properties. Next, we propose an architectural software pattern useful for applications adopting our reasoning system. Finally, we state the conclusions of our work.

2 The OP-DeLP programming language: fundamentals

Possibilistic Defeasible Logic Programming (P-DeLP) [1, 2] is an important extension of DeLP in which the elements of the language have the form (φ, α), where φ is a DeLP clause or fact. Below, we will introduce the elements of the language necessary in this presentation. Observation based P-DeLP (OP-DeLP) is an optimization of P-DeLP that allows the computation of warranted arguments in a more efficient way, by means of a pre-compiled knowledge component. It also permits a seamless incorporation of new perceived facts into the program codifying the knowledge base of the system. Therefore the resulting system can be used to implement practical applications with performance requirements. The idea of extending the applicability of DeLP in a dynamic setting, incorporating perception and pre-compiled knowledge, was originally conceived in [5]. Thus the OP-DeLP system incorporates elements from two different variants of the DeLP system, O-DeLP [5] and P-DeLP [9]. In what follows we present the formal definition of the resulting system.

The concepts of signature, functions and predicates are defined in the usual way. The *alphabet* of OP-DeLP programs generated from a given signature Σ is composed by the members of Σ, the symbol "$\sim$" denoting strong negation [11] and the symbols "(", ")", "." and ",". Terms, Atoms and Literals are defined in the usual way. A *certainty weighted literal*, or simply a weighted literal, is a pair (L, α) where L is a literal and $\alpha \in [0, 1]$ expresses a lower bound for the certainty of φ in terms of a necessity measure.

OP-DeLP programs are composed by a set of *observations* and a set of *defeasible rules*. Observations are weighted literals and thus have an associated certainty degree. In real world applications, observations model perceived facts. Defeasible rules provide a way of performing tentative reasoning as in other argumentation formalisms.

Definition 1. *A* defeasible rule *has the form* $(L_0 \prec L_1, L_2, \ldots, L_k, \alpha)$ *where* L_0 *is a literal,* $L_1, L_2, \ldots, L_k$ *is a non-empty finite set of literals, and* $\alpha \in [0, 1]$ *expresses a lower bound for the certainty of the rule in terms of a necessity measure.*

Intuitively a defeasible rule $L_0 \prec L_1, L_2, \ldots, L_k$ can be read as *"*$L_1, L_2, \ldots, L_k$ *provide tentative reasons to believe in* L_0*"* [15]. In OP-DeLP these rules also have a certainty degree, that quantifies how strong is the connection between the

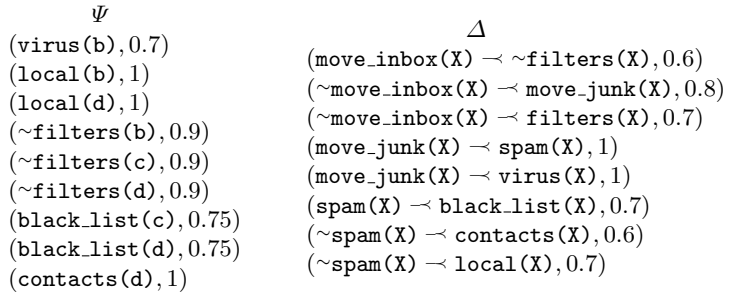

Fig. 1. An OP-DeLP program for email filtering

premises and the conclusion. A defeasible rule with a certainty degree 1 models a strong rule.

A set of weighted literals Γ will be deemed as contradictory, denoted as $\Gamma \mid\sim \bot$, iff $\Gamma \mid\sim (l, \alpha)$ and $\Gamma \mid\sim (\neg l, \beta)$ with α and $\beta > 0$. In a given OP-DeLP program we can distinguish certain from uncertain information. A clause (γ, α) will be deemed as *certain* if $\alpha = 1$, otherwise it will be *uncertain*.

Definition 2 (OP-DeLP Program). *An* OP-DeLP program $\mathcal{P}$ *is a pair* $\langle \Psi, \Delta \rangle$, *where* Ψ *is a* non contradictory *finite set of observations and* Δ *is a finite set of defeasible rules.*

Example 1. Fig.1 shows a program for basic email filtering. Observations describe different characteristics of email messages. Thus, `virus(X)` stands for "message `X` has a virus"; `local(X)` indicates that "message `X` is from the local host"; `filters(X)` specifies that "message `X` should be filtered" redirecting it to a particular folder; `black_list(X)` indicates that "message `X` is considered dangerous" because of the server it is coming from; and `contacts(X)` indicates that "the sender of message `X` is in the contact list of the user".

The first rule expresses that if the email does not match with any user-defined filter then it usually should be moved to the "inbox" folder. The second rule indicates that unfiltered messages in the "junk" folder usually should not be moved to the inbox. According to the third rule, messages to be filtered should not be moved to the inbox. The following two rules establish that a message should be moved to the "junk" folder if it is marked as spam or it contains viruses. Finally there are three rules for spam classification: a message is usually labeled as spam if it comes from a server that is in the blacklist. Nevertheless, even if an email comes from a server in the blacklist it is not labeled as spam when the sender is in the contact list of the user. Besides, a message from the local host is usually not classified as spam.

In OP-DeLP the proof method, written $\mid\sim$, is defined by derivation based on the following instance of the generalized modus ponens rule (GMP): $(L_0 \prec L_1 \wedge \cdots \wedge L_k, \gamma), (L_1, \beta_1), \ldots, (L_k, \beta_k) \vdash (L_0, \min(\gamma, \beta_1, \ldots, \beta_k))$, which is a particular instance of the well-known possibilistic resolution rule. Literals in the set of

observations Ψ are the basis case of the derivation sequence, for every literal Q in Ψ with a certainty degree α it holds that (Q, α) can be derived from $\mathcal{P} = (\Psi, \Delta)$.

Given an OP-DeLP program $\mathcal{P}$, a query posed to $\mathcal{P}$ corresponds to a ground literal Q which must be supported by an *argument* [15, 10].

Definition 3. [Argument]–[Subargument] *Let $\mathcal{P} = \langle \Psi, \Delta \rangle$ be a program, $\mathcal{A} \subseteq \Delta$ is an* argument *for a goal Q with necessity degree $\alpha > 0$, denoted as $\langle \mathcal{A}, Q, \alpha \rangle$, iff:* (1) *$\Psi \cup \mathcal{A} \vdash (Q, \alpha)$,* (2) *$\Psi \cup \mathcal{A}$ is non contradictory, and* (3) *there is no $\mathcal{A}_1 \subset \mathcal{A}$ such that $\Psi \cup \mathcal{A}_1 \vdash (Q, \beta)$, $\beta > 0$. An argument $\langle \mathcal{A}, Q, \alpha \rangle$ is a* subargument *of $\langle \mathcal{B}, R, \beta \rangle$ iff $\mathcal{A} \subseteq B$.*

As in most argumentation frameworks, arguments in O-DeLP can attack each other. An argument $\langle \mathcal{A}_1, Q_1, \alpha \rangle$ *counter-argues* an argument $\langle \mathcal{A}_2, Q_2, \beta \rangle$ at a literal Q if and only if there is a sub-argument $\langle \mathcal{A}, Q, \gamma \rangle$ of $\langle \mathcal{A}_2, Q_2, \beta \rangle$, (called *disagreement subargument*), such that Q_1 and Q are complementary literals. Defeat among arguments is defined combining the counterargument relation and a preference criterion "$\preceq$". This criterion is defined on the basis of the necessity measures associated with arguments.

Definition 4. [Preference criterion $\succeq$][9] *Let $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ be a counterargument for $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$. 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 5. [Defeat][9] *Let $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ and $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$ be two arguments built from a program $\mathcal{P}$. Then $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$* 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) Argument $\langle \mathcal{A}_1, Q_1, \alpha_1 \rangle$ counter-argues argument $\langle \mathcal{A}_2, Q_2, \alpha_2 \rangle$ with disagreement subargument $\langle \mathcal{A}, Q, \alpha \rangle$; and (2) Either it is true 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 [7, 13], OP-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 an OP-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 $\mathcal{P}$. Non-contradiction for a set of arguments is defined as follows: a set $S = \bigcup_{i=1}^{n} \{\langle \mathcal{A}_i, Q_i, \alpha_i \rangle\}$ is *contradictory* wrt $\mathcal{P}$ iff $\Psi \cup \bigcup_{i=1}^{n} \mathcal{A}_i$ is contradictory.
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 proved 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 dialogs 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*) iff the root of $\mathcal{T}_{\langle \mathcal{A}_0, Q_0, \alpha_0 \rangle}$ is labeled as *U-node*.

Definition 6. [Warrant][9] *Given a program $\mathcal{P}$, and a literal Q, Q is warranted wrt $\mathcal{P}$ iff there exists a warranted argument $\langle \mathcal{A}, Q, \alpha \rangle$ than can be built from $\mathcal{P}$.*

To answer a query for a given literal we should see if there exists a warranted argument supporting this literal. Nevertheless, in OP-DeLP there may be different arguments with different certainty degrees supporting a given query. This fact was not considered in [9], but we are clearly interested in finding the warranted argument with the highest certainty degree.

Definition 7. [Strongest Warrant] *Given a program $\mathcal{P}$, and a literal Q, we will say that α is the* strongest warrant degree *of Q iff (1) there exists a warranted argument $\langle \mathcal{A}, Q, \alpha \rangle$ than can be built from $\mathcal{P}$ and (2) no warranted argument $\langle \mathcal{B}, Q, \beta \rangle$ such that $\beta > \alpha$ can built from $\mathcal{P}$.*

Note that to find out the strongest warrant degree for a given literal Q we need to find the strongest warranted argument supporting it, that is, the warranted argument supporting Q with the higher certainty degree. Then, to find the strongest warrant degree for a literal Q we must first build the argument $\mathcal{A}$ that supports the query Q with the highest possible certainty degree and see if $\mathcal{A}$ is a warrant for Q. Otherwise we must find another argument $\mathcal{B}$ for Q with the highest certainty degree among the remaining ones, see if it is a warrant for Q, and so on, until a warranted argument is found or there are no more arguments supporting Q.

Example 2. Consider the program shown in Example 1 and let `move_inbox(d)` be a query wrt this program. The search for a warrant for `move_inbox(d)` will result in an argument $\langle \mathcal{A}, \texttt{move\_inbox(d)}, 0.6 \rangle$, with

$$\mathcal{A} = \{(\texttt{move\_inbox(d)} \prec \sim\texttt{filters(d)}, 0.6)\}$$

allowing to conclude that message d should be moved to the folder Inbox, as it has no associated filter with a certainty degree of 0.6. However, there exists a defeater for $\langle \mathcal{A}, \texttt{move\_inbox(d)}, 0.6 \rangle$, namely $\langle \mathcal{B}, \sim\texttt{move\_inbox(d)}, 0.7 \rangle$, as there are reasons to believe that message d is spam:

$$\begin{aligned}\mathcal{B} = \{&(\sim\texttt{move\_inbox(d)} \prec \texttt{move\_junk(d)}, 0.8)\\ &(\texttt{move\_junk(d)} \prec \texttt{spam(d)}, 1), (\texttt{spam(d)} \prec \texttt{black\_list(d)}, 0.7)\}\end{aligned}$$

Using the preference criterion, $\langle \mathcal{B}, \sim\texttt{move\_inbox(d)}, 0.7\rangle$ is a proper defeater for $\langle \mathcal{A}, \texttt{move\_inbox(d)}, 0.6\rangle$. However, two counterarguments can be found for $\langle \mathcal{B}, \sim\texttt{move\_inbox(d)}, 0.7\rangle$, since message d comes from the local host, and the sender is in the user's contacts list:

- $\langle \mathcal{C}, \sim\texttt{spam(d)}, 0.6\rangle$, where $\mathcal{C} = \{(\sim\texttt{spam(d)} \prec \texttt{contacts(d)}, 0.6)\}$.
- $\langle \mathcal{D}, \sim\texttt{spam(d)}, 0.9\rangle$, where $\mathcal{D} = \{(\sim\texttt{spam(d)} \prec \texttt{local(d)}, 0.9)\}$.

$\mathcal{B}$ defeats $\mathcal{C}$ but is defeated by $\mathcal{D}$. There are no more arguments to consider, and the resulting dialectical tree has only one argumentation line: $\mathcal{A}$ is defeated by $\mathcal{B}$ who is in turn defeated by $\mathcal{D}$. Hence, the marking procedure determines that the root node $\langle \mathcal{A}, \texttt{move\_inbox(d)}, 0.6\rangle$ is a `U-node` and the original query is warranted.

3 Dialectical graphs and pre-compiled knowledge

To obtain faster query processing in the OP-DeLP system we integrate pre-compiled knowledge to avoid the construction of arguments which were already computed before. The approach follows the proposal presented in [5] where the pre-compiled knowledge component is required to: (1) minimize the number of stored arguments in the pre-compiled base of arguments (for instance, using one structure to represent the set of arguments that use the same defeasible rules); and (2) maintain independence from the observations that may change with new perceptions, to avoid modifying also the pre-compiled knowledge when new observations are incorporated.

Considering these requirements, we define a database structure called *dialectical graph*, which will keep a record of all possible *arguments* in an OP-DeLP program $\mathcal{P}$ (by means of a special structure named potential argument) as well as the counterargument relation among them. Potential arguments, originally defined in [5] contain non-grounded defeasible rules, depending thus only on the set of rules Δ in $\mathcal{P}$ and are independant from the set of observations Ψ.

Potential arguments have been devised to sum-up arguments that are obtained using *different* instances of the *same* defeasible rules. Recording every generated argument could result in storing many arguments which are structurally identical, only differing on the constants being used to build the corresponding derivations. Thus, a potential argument stands for several arguments which use the same defeasible rules. Attack relations among potential arguments can be also captured, and in some cases even defeat can be pre-compiled. In what follows we introduce the formal definitions, adapted from [5] to fit the OP-DeLP system.

Definition 8. [Weighted Potential argument] *Let Δ be a set of defeasible rules. A subset A of Δ is a* potential argument *for a literal Q with an upper bound γ for its certainty degree, noted as $\langle\langle \mathsf{A}, Q, \gamma\rangle\rangle$ if there exists a non-contradictory set of literals Φ and an instance $\mathcal{A}$ that is obtained finding an instance for every rule in A, such that $\langle \mathcal{A}, Q, \alpha\rangle$ is an argument wrt $\langle \Phi, \Delta\rangle (\alpha \leq \gamma)$ and there is no instance $\langle \mathcal{B}, Q, \beta\rangle$ of A such that $\beta > \gamma$*

The nodes of the dialectical graph are the potential arguments. The arcs of our graph are obtained calculating the counterargument relation among the nodes previously obtained. To do this, we extend the concept of counterargument for potential arguments. A potential argument $\langle\langle \mathsf{A}_1, Q_1, \alpha\rangle\rangle$ *counter-argues* $\langle\langle \mathsf{A}_2, Q_2, \beta\rangle\rangle$ at a literal Q if and only if there is a non-empty potential subargument $\langle\langle \mathsf{A}, Q, \gamma\rangle\rangle$ of $\langle\langle \mathsf{A}_2, Q_2, \beta\rangle\rangle$ such that Q_1 and Q are contradictory literals.[1] Note that potential counter-arguments may or may not result in a real conflict between the instances (arguments) associated with the corresponding potential arguments. In some cases instances of these arguments cannot co-exist in any scenario (*e.g.*, consider two potential arguments based on contradictory observations). Now we can finally define the concept of dialectical graph:

Definition 9. [Dialectical Graph] *Let $\mathcal{P} = \langle \Psi, \Delta\rangle$ be an OP-DeLP program. The* dialectical graph *of Δ, denoted as $\mathcal{G}_\Delta$, is a pair* $(\text{PotArg}(\Delta), C)$ *such that: (1)* $\text{PotArg}(\Delta)$ *is the set* $\{\langle\langle \mathsf{A}_1, Q_1, \alpha_1\rangle\rangle, \ldots, \langle\langle \mathsf{A}_k, Q_k, \alpha_k\rangle\rangle\}$ *of all the potential arguments that can be built from Δ; (2) $\mathcal{C}$ is the counterargument relation over the elements of* $\text{PotArg}(\Delta)$.

We have devised a set of algorithms to use the dialectical graph for improving the inference process. For space reasons these algorithms are not detailed in this work, the interested reader may consult [6] for a more detailed treatment of this subject. We have also compared the obtained algorithms theoretically with standard argument-based inference techniques (such as those used in P-DeLP). At the inference process, we have found out that complexity is lowered from $O\left(2^{|\Delta'|^{3.(2^{|\Delta'|})/4}}\right)$ to $O(2^{|\Delta'|}.|\Delta'|)$.

4 A proposed architecture for OP-DeLP applications

Applications that use the OP-DeLP system will be be engineered for contexts where: (1) information is uncertain and heterogeneous, (2) handling of great volume of data flows is needed, and (3) data may be incomplete, vague or contradictory. In this section we present an architectural pattern that can be used in these applications.

Previous to proposing a pattern we started analyzing the characteristics of OP-DeLP applications. First, we found that data will generally be obtained from multiple sources. Nowadays the availability of information trough the Internet has shifted the issue of information from quantitative stakes to qualitative ones [3]. For this reason, new information systems also need to provide assistance for judging and examining the quality of the information they receive.

For our pattern, we have chosen to use a multi-source perspective into the characterization of data quality[3]. In this case the quality of data can be evaluated by comparison with the quality of other homologous data (i.e. data from different information sources which represent the same reality but may have

[1] Note that $P(X)$ and $\sim P(X)$ are contradictory literals although they are non-grounded. The same idea is applied to identify contradiction in potential arguments.

contradictory values). The approaches usually adopted to reconcile heterogeneity between values of data are : (1) to prefer the values of the most reliable sources, (2) to mention the source ID for each value, or (3) to store quality meta-data with the data. We have chosen to use the second approach. In multi-source databases, each attribute of a multiple source element has multiple values with the ID of their source and their associated quality expertise. Quality expertise is represented as meta-data associated with each value. We have simplified this model for an easy and practical integration with the OP-DeLP system. In our case, data sources are assigned a unique certainty degree. For simplicity sake, we assume that different sources have different values. All data from a given source will have the same certainty degree. This degree may be obtained weighting the plausibility of the data value, its accuracy, the credibility of its source and the freshness of the data.

OP-DeLP programs basically have a set of observations Ψ and a set of rules Δ. The set of rules is chosen by the knowledge engineer and remains fixed. The observation set may change according with new perceptions received from the multiple data sources. Nevertheless, inside the observation set we will distinguish a special kind of perceptions, those with certainty degree 1. Those perceptions are also codified by the knowledge engineer and cannot be modified in the future by the perception mechanism. To assure this, we assume that every data source has a certainty value γ such that $0 < \gamma < 1$.

Example 3. Consider the program in Example 2. In this case data establishing a given message is from the local host comes from the same data source and can be given a certainty degree of 1. The same applies for `contacts(X)`. The algorithm that decides whether to filter a given message is another data source with a degree of 0.9, the filter that classifies a message as a virus is another data source with a degree of 0.7, and the algorithm that checks if the message came from some server in the blacklist is a different source that has a degree of 0.75. Note that we could have different virus filters with different associated certainty degrees if we wanted to build higher trust on this filter mechanism.

The scenario just described requires an updating criterion different to the one presented in [5], given that the situation regarding perceptions in OP-DeLP is much more complex. To solve this, we have devised Algorithm 1, that summarizes different situations in two conditions. The first one acts when the complement of the literal Q is already present in the set Ψ. Three different cases can be analyzed in this setting: (1) If both certainty degrees are equal it means that both Q and its complement proceed from the same data source. Then the only reason for the conflict is a change in the state of affairs, thus an update is needed and the new literal is added. (2) If $\alpha > \beta$ it means that the data sources are different, Thus we choose to add (Q, α) since it has the higher certainty degree. (3) If $\alpha < \beta$ we keep $(\overline{Q}, \beta)$. Note that (1) is an update operation [12] while (2) and (3) are revisions over Ψ. The difference between updating and revision is fundamental. Updating consists in bringing the knowledge base up to date when the world changes. Revision allows us to obtain new information about a static scenario [12].

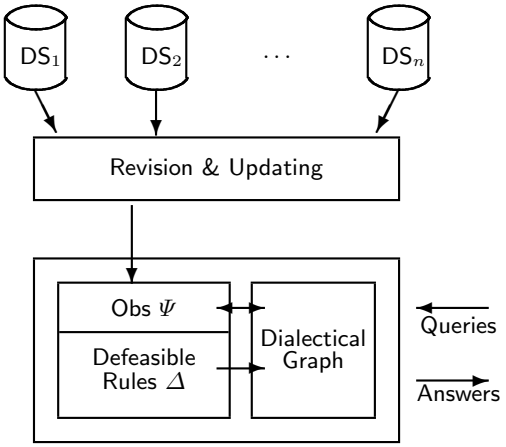

Fig. 2. Architecture for applications using OP-DeLP as underlying framework

The second condition in Algorithm 1 considers the case when Q was in Ψ with a different certainty degree. Then it chooses the weighted literal with the highest degree possible. Note that the observations initially codified that have a certainty degree of 1 cannot be deleted or modified by algorithm 1.

Algorithm 1 `UpdateObservationSet`

`Input:` $\mathcal{P} = \langle \Psi, \Delta \rangle$, (Q, α)
`Output:` $\mathcal{P} = \langle \Psi, \Delta \rangle$ {*With* Ψ *updated*}

`If there exists a weighted literal` $(\overline{Q}, \beta) \in \Psi$ `such that` $\beta \leq \alpha$ `Then`
  `delete`$((\overline{Q}, \beta)$
  `add`$((Q, \alpha))$
`If there exists a weighted literal` $(Q, \beta) \in \Psi$ `such that` $\alpha \leq \beta$ `Then`
  `delete`$((Q, \beta)$
  `add`$((Q, \alpha))$

Finally, fig. 2 summarizes the main elements of the O-DeLP-based architecture. Knowledge is represented by an OP-DeLP program $\mathcal{P}$. Perceptions from multiple sources may result in changes in the set of observations in $\mathcal{P}$, handled by the updating mechanism defined in algorithm 1. To solve queries the OP-DeLP inference engine is used. This engine is assisted by the dialectical graph (Def. 9) to speed-up the argumentation process. The final answer to a given query Q will be *yes*, with the particular certainty degree of the warranted argument supporting Q, or *no* if the system could not find a warrant for Q from $\mathcal{P}$.

5 Conclusions

In this work we have defined an argumentation-based formalism that integrates uncertainty management. This system was also provided with an optimization mechanism based on pre-compiled knowledge. Using this, the argumentation system can comply with real time requirements needed to administer data and model reasoning over this data in dynamic environments. Another contribution is the architectural model to integrate OP-DeLP in practical applications to administer and reason with data from multiple sources.

As future work, we are developing a prototype based on the proposed architecture to extend the theoretical complexity analysis with empirical results and to test the integration of the OP-DeLP reasoning system in real world applications. We are also working on the integration of OP-DeLP and database management systems by means of a strongly-coupled approach.

References

1. T. Alsinet, C. I. Chesñevar, L. Godo, S. Sandri, and G. R. Simari. Formalizing argumentative reasoning in a possibilistic logic programming setting with fuzzy unification. *International Journal of Approximate Reasoning*, 48(3):711–729, 2008.
2. T. Alsinet, C. I. Chesñevar, L. Godo, and G. R Simari. A logic programming framework for possibilistic argumentation: Formalization and logical properties. *Fuzzy Sets and Systems*, 159(10):208–228, 2008.
3. L. Berti. Quality and recommendation of multi-source data for assisting technological intelligence applications. In *Proc. of 10th International Conference on Database and Expert Systems Applications*, pages 282–291, Italy, 1999. AAAI.
4. Bryant and Krause. An implementation of a lightweight argumentation engine for agent applications. *Lecture Notes in Computer Science*, 4160(1):469–472, 2006.
5. M. Capobianco, C. I. Chesñevar, and G. R. Simari. Argumentation and the dynamics of warranted beliefs in changing environments. *Journal of Autonomous Agents and Multiagent Systems*, 11:127–151, 2005.
6. M. Capobianco and G. Simari. A proposal for making argumentation computationally capable of handling large repositories of uncertain data. In *Proceedings of the third international conference on scalable uncertainty management*, 2009. to appear.
7. C. I. Chesñevar, A. G. Maguitman, and R. P. Loui. Logical Models of Argument. *ACM Computing Surveys*, 32(4):337–383, 2000.
8. C. I. Chesñevar, A. G. Maguitman, and G. R. Simari. Argument-based critics and recommenders: A qualitative perspective on user support systems. *Data & Knowledge Engineering*, 59(2):293–319, 2006.
9. C. I. Chesñevar, G. R. Simari, T. Alsinet, and L. Godo. A logic programming framework for possibilistic argumentation with vague knowledge. In *Proc. of Uncertainty in Artificial Intelligence Conference (UAI 2004), Banff, Canada (to appear)*, 2004.
10. A. García and G. Simari. Defeasible Logic Programming: An Argumentative Approach. *Theory and Practice of Logic Programming*, 4(1):95–138, 2004.
11. M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, pages 365–385, 1991.
12. H. Katsuno and A. Mendelzon. On the difference between updating a knowledge base and revising it. In P.Gardenfors, editor, *Belief Revision*, pages 183–203. Cambridge University Press, 1992.
13. H. Prakken and G. Vreeswijk. Logical systems for defeasible argumentation. In *Handbook of Philosophical Logic*, volume 4, pages 219–318. 2002.
14. I. Rahwan, S. D. Ramchurn, N. R. Jennings, P. McBurney, S. Parsons, and L. Sonenberg. Argumentation-based negotiation. *The Knowledge Engineering Review*, 18(4):343–375, 2003.
15. G. R. Simari and R. P. Loui. A Mathematical Treatment of Defeasible Reasoning and its Implementation. *Artificial Intelligence*, 53(1–2):125–157, 1992.