Modelling Argumentation in a Logic-Programming Setting: Formalization and Logical Properties

Guillermo R. Simari – Carlos I. Chesñevar – Alejandro J. García

Artificial Intelligence Research & Development Laboratory (LIDIA)
Department of Computer Science – Universidad Nacional del Sur – Bahía Blanca, Argentina
Email: {grs,cic,ajg}@cs.uns.edu.ar – Tel/Fax: +54-291-455-5135/5136

Keywords: defeasible argumentation, logic programming, labelled deduction

1 Introduction and motivations

Defeasible reasoning based on different views of argumentation [19, 5, 15] has proven to be a successful approach to formalizing the act of reasoning with incomplete and potentially inconsistent information. Recent research (notably [1]) has shown that defeasible argumentation constitutes a confluence point for characterizing traditional approaches to non-monotonic reasoning (such as Gelfond's extended logic programming and Reiter's default logic). In this context, logic programming (LP) has provided a useful setting for exploring both theoretical and practical issues in defeasible argumentation. Dung's work on argumentative semantics for logic programs [6] paved the way for other formalisms, most of them based on different versions of extended logic programming (such as Prakken's [14], among others).

This paper reports some of the main results from two major research lines that have been explored in our laboratory since 1994, continuing the research work started in 1987 [18]. First, we will describe the most relevant features of *defeasible logic programming* (DeLP) [8], a LP-based formalism for defeasible argumentation. Second, we will detail some aspects of LDS_{ar} [3], a formalization for defeasible argumentation based on labelled deduction (LD) [7] which allows the exploration of several logical properties of DeLP, as well as other related formalisms. Finally, we will present the main conclusions that have been obtained, as well as a brief sketch of our ongoing research.

2 How we perceive argumentation

2.1 Knowledge representation and inference: argument structures and defeat

A knowledge base (*DeLP* program) can be thought of as a pair (Π, Δ), where Π and Δ represent sets of non-defeasible and defeasible knowledge, respectively.

The set Π is assumed to be *non-contradictory.*[1] This distinction is quite common in many existing argumentative frameworks (such as Prakken & Sartor's [15] and Vreeswijk's [20]), and it can be traced back to the original Simari-Loui formalism [19] and the first ideas of nonmonotonic reasoning [16]. In *DeLP* two different negations are allowed: strong negation "$\sim$" for representing contradictory literals and default negation "*not*" for representing incomplete information. The underlying representational language of *DeLP* consists of ground facts and two kinds of ground rules: *defeasible* rules ($l \prec q_1, \dots q_k$) and *strict* (non-defeasible) rules ($l \leftarrow q_1, \dots, q_k$), where l is a ground literal, and $q_1, \dots, q_k$, represents a conjunction of literals. Default negation may be used only in the body of defeasible rules. The symbols " $\prec$ " and " $\leftarrow$ " denote meta-relations between sets of literals. *DeLP* rules are thus to be thought of as *inference rules* rather than rules in the object language.

Our notion of *argument* is a variant of the original definition proposed in [19], properly suited to a logic programming setting.

Definition 1 (Argument Structure). *Let $\mathcal{P} = (\Pi, \Delta)$ be a DeLP program. A set $\mathcal{A} \subseteq \Delta$, is an* argument structure *for a literal h, denoted $\langle \mathcal{A}, h \rangle$, if and only if:*
1. there exists a defeasible derivation for h from $\Pi \cup \mathcal{A}$.
2. $\Pi \cup \mathcal{A}$ is non-contradictory.
3. $\mathcal{A}$ is minimal (there is no proper subset $\mathcal{A}'$ of $\mathcal{A}$ such that $\mathcal{A}'$ satisfies conditions (1) and (2)).
An argument $\langle \mathcal{A}_1, h_1 \rangle$ is a sub-argument *of another argument $\langle \mathcal{A}_2, h_2 \rangle$ if $\mathcal{A}_1 \subseteq \mathcal{A}_2$.*

In the case of *DeLP*, the notion *defeasible derivation* is the usual derivation used in logic programming, considering both strict and defeasible rules. Minimality imposes a kind of 'Occam's razor principle' [19] on argument construction, as any superset of an argument $\mathcal{A}$ can be proven to be 'weaker' than $\mathcal{A}$ itself as far as possible attacking arguments are concerned. The non-contradiction requirement forbids the use of defeasible rules in an argument $\mathcal{A}$ whenever $\Pi \cup \mathcal{A}$ entails two complementary literals p and $\overline{p}$.

We assume the existence of a partial order "$\leq$" on argument structures. Hence, if $\langle \mathcal{A}_1, h_1 \rangle \leq \langle \mathcal{A}_2, h_2 \rangle$ we will say that $\langle \mathcal{A}_2, h_2 \rangle$ is preferred over $\langle \mathcal{A}_1, h_1 \rangle$. For example, specificity [19] is typically used as a syntax-based preference criterion among arguments, although any other partial order would also be valid (see [8]).

Now we are ready to introduce the notion of *defeater.* Informally, $\langle \mathcal{A}_1, h_1 \rangle$ is a defeater for $\langle \mathcal{A}_2, h_2 \rangle$ if both argument structures are in contradiction and $\langle \mathcal{A}_2, h_2 \rangle$ is not preferred over $\langle \mathcal{A}_1, h_1 \rangle$. Formally,

[1] We use the term *non-contradictory* instead of *inconsistent* to emphasize independence of the underlying logical language. Contradiction arises when deriving two complementary literals wrt some form of negation.

Definition 2 (Defeater).
Let $\langle \mathcal{A}_1, h_1 \rangle$ and $\langle \mathcal{A}_2, h_2 \rangle$ be two argument structures. $\langle \mathcal{A}_1, h_1 \rangle$ is a defeater *for $\langle \mathcal{A}_2, h_2 \rangle$ at literal h iff:*

(a) there exists a sub-argument $\langle \mathcal{A}, h \rangle$ of $\langle \mathcal{A}_2, h_2 \rangle$ such that $\Pi \cup \{h_1, h\}$ is a contradictory set, and $\langle \mathcal{A}_1, h_1 \rangle$ is preferred over $\langle \mathcal{A}, h \rangle$ (proper defeater), or
(b) there exists a sub-argument $\langle \mathcal{A}, h \rangle$ of $\langle \mathcal{A}_2, h_2 \rangle$ such that $\Pi \cup \{h_1, h\}$ is a contradictory set, and $\langle \mathcal{A}_1, h_1 \rangle$ is not related to $\langle \mathcal{A}, h \rangle$ (blocking defeater), or
(c) the extended literal "not h_1" is in the body of a defeasible rule in $\langle \mathcal{A}_2, h_2 \rangle$.

2.2 Computing warrant through dialectical analysis

The dialectical analysis is carried out by using a tree-like structure called *dialectical tree* (first introduced in [17]) which facilitates the analysis and implementation of defeat and reinstatement among arguments, as well as the definition of "fallacy-checks" (such as detecting circular and/or contradictory argumentation during the dialectical process). A dialectical tree rooted in an argument $\langle \mathcal{A}_0, q_0 \rangle$ is defined by 'bundling' together all possible *acceptable argumentation lines* starting in $\langle \mathcal{A}_0, q_0 \rangle$, i. e. sequences of arguments and defeaters which are fallacy-free. A subsequent marking procedure allows the determination of whether the original argument $\langle \mathcal{A}_0, q_0 \rangle$ is ultimately accepted or *warranted.*

An *argumentation line* $\lambda = [\ \langle \mathcal{A}_0, q_0 \rangle, \langle \mathcal{A}_1, q_1 \rangle, \langle \mathcal{A}_2, q_2 \rangle, \ldots, \langle \mathcal{A}_n, q_n \rangle\ \ldots\ ]$ can be thought of as an exchange of arguments between two parties, a proponent and an opponent [17], such that each $\langle \mathcal{A}_i, q_i \rangle$ defeats the previous argument $\langle \mathcal{A}_{i-1}, q_{i-1} \rangle$ in the sequence. Dialectics imposes additional requirements on such an argument exchange for it to be considered rationally acceptable. In such a setting, *fallacious* reasoning (such as circular argumentation and falling into self-contradiction) is to be avoided. This can be done by requiring that all argumentation lines be *acceptable* [17]. An acceptable argumentation line starting with an argument $\langle \mathcal{A}_0, q_0 \rangle$ constitutes an exchange of arguments which can be pursued until no more arguments can be introduced because of the aforementioned dialectical constraints.

Definition 3 (Acceptable argumentation line). *Let $\mathcal{P}$ be a dlp, and let $\lambda = [\langle \mathcal{A}_0, q_0 \rangle, \langle \mathcal{A}_1, q_1 \rangle, \ldots, \langle \mathcal{A}_n, q_n \rangle, \ldots\ ]$ be an argumentation line in $\mathcal{P}$. Let $\lambda' = [\langle \mathcal{A}_0, q_0 \rangle, \langle \mathcal{A}_1, q_1 \rangle, \ldots, \langle \mathcal{A}_k, q_k \rangle\ ]$, be an initial segment of λ starting in $\langle \mathcal{A}_0, q_0 \rangle$. The sequence λ' is an* acceptable argumentation line *in $\mathcal{P}$ iff it is the longest subsequence in λ satisfying the following conditions:*

1. *The sets ${\lambda'}_S$ and λ'_I are each non-contradictory sets of arguments wrt $\mathcal{P}$.*[2]
2. *No argument $\langle \mathcal{A}_j, q_j \rangle$ in λ' is a sub-argument of an earlier argument $\langle \mathcal{A}_i, q_i \rangle$ of λ' ($i < j$).*

[2] Non-contradiction for a set of arguments is defined as follows: a set $S = \bigcup_{i=1}^{n} \{\langle \mathcal{A}_i, q_i \rangle\}$ is *contradictory* wrt a *DeLP* program $\mathcal{P}$ iff $\Pi \cup \bigcup_{i=1}^{n} \mathcal{A}_i$ is contradictory.

3. *There is no subsequence of arguments [$\langle \mathcal{A}_{i-1}, q_{i-1} \rangle$, $\langle \mathcal{A}_i, q_i \rangle$, $\langle \mathcal{A}_{i+1}, q_{i+1} \rangle$] in λ', such that $\langle \mathcal{A}_i, q_i \rangle$, is a blocking defeater for $\langle \mathcal{A}_{i-1}, q_{i-1} \rangle$ and $\langle \mathcal{A}_{i+1}, q_{i+1} \rangle$ is a blocking defeater for $\langle \mathcal{A}_i, q_i \rangle$.*

The rationales for the conditions in Definition 3 are to be understood in a dialectical setting [17]. Condition 1 disallows the use of contradictory information on either side (proponent or opponent). Condition 2 eliminates the *"circulus in demonstrando"* fallacy (circular reasoning). Finally, condition 3 enforces the use of a stronger argument to defeat an argument which acts as a blocking defeater.

Definition 4 (Dialectical Tree). *Let $\langle \mathcal{A}_0, h_0 \rangle$ be an argument structure from a program $\mathcal{P}$. A dialectical tree for $\langle \mathcal{A}_0, h_0 \rangle$, denoted $\mathcal{T}_{\langle \mathcal{A}_0, h_0 \rangle}$, is obtained as follows:*

1. *The root of the tree is labeled with $\langle \mathcal{A}_0, h_0 \rangle$.*
2. *Let N be a node of the tree labeled $\langle \mathcal{A}_n, h_n \rangle$, and $[\langle \mathcal{A}_0, h_0 \rangle, \langle \mathcal{A}_1, h_1 \rangle, \ldots, \langle \mathcal{A}_n, h_n \rangle]$ the sequence of labels of the path from the root to N. Let $\langle \mathcal{B}_1, q_1 \rangle$, $\langle \mathcal{B}_2, q_2 \rangle$, $\ldots$, $\langle \mathcal{B}_k, q_k \rangle$ be all the defeaters for $\langle \mathcal{A}_n, h_n \rangle$.*
   For each defeater $\langle \mathcal{B}_i, q_i \rangle$ $(1 \leq i \leq k)$, such that the argumentation line $[\langle \mathcal{A}_0, h_0 \rangle, \ldots, \langle \mathcal{A}_n, h_n \rangle, \langle \mathcal{B}_i, q_i \rangle]$ is acceptable, then the node N has a child N_i labeled $\langle \mathcal{B}_i, q_i \rangle$.
   If there is no defeater $\langle \mathcal{B}_i, q_i \rangle$ that satisfies that condition, then N is a leaf.

Note that a dialectical tree is an AND-OR tree. Leaves can be marked as *undefeated nodes* (*U-nodes*), as they have no defeaters. Every inner node is to be marked as *D-node* iff it has at least one active defeater (*U-node*) as a child, and as *U-node* otherwise.

Definition 5 (Warranted literals). *Let $\mathcal{A}$ be an argument structure for a literal h, and $\mathcal{T}^*_{\langle \mathcal{A}, h \rangle}$ its associated marked dialectical tree. The literal h is warranted iff the root of $\mathcal{T}^*_{\langle \mathcal{A}, h \rangle}$ is a U-node.*

Based on the notion of warrant, we will define a modal operator of belief "B", where "Bh" stands for "h is warranted". The possible answers of an interpreter of *DeLP* can be defined in terms of this operator. There are four possible answer for a query h: (1) YES, if Bh; (2) NO, if $B \sim h$; (3) UNDECIDED, if $\neg Bh$ and $\neg B \sim h$; and (4) UNKNOWN, if h is not in the program's signature.

3 Formalizing argumentation using labelled deduction

3.1 Motivations and fundamentals

The study of logical properties of defeasible argumentation, particularly those related to the *DeLP* framework, motivated the development of LDS_{ar} [3], an argumentation formalism based on the *labelled deduction* methodology [7].[3] In

[3] It must be remarked that the use of labelled deduction as a basis for formalizing an argumentation framework was first explored in [11].

labelled deduction, the usual notion of formula is replaced by the notion of *labelled formula*, expressed as *Label*:*f*, where *Label* represents a label associated with the wff *f*. A labelling language $\mathcal{L}_{Label}$ and knowledge-representation language $\mathcal{L}_{kr}$ can be combined to provide a new, labelled language, in which labels convey additional information also encoded at object-language level. Formulas are labelled according to a family of *deduction rules*, and with agreed ways of propagating labels via the application of these rules.

In LDS_{ar}, the language $\mathcal{L}_{kr}$ is the one of extended logic programming. Labels extend this language by distinguishing defeasible and non-defeasible information. A consequence relation $\vdash_{Arg}$ propagates labels, implementing the SLD resolution procedure along with a consistency check every time new defeasible information is introduced in a proof. This information is collected into a *support set*, containing all defeasible information needed to conclude a given formula. Thus, arguments are modelled as labelled formulas $\mathcal{A}$:h, where $\mathcal{A}$ stands for a set of (ground) clauses, and h for an extended literal.

Given a knowledge base Γ the consequence relation $\vdash_{Arg}$ allows the inference of labelled formulas of the form *argument*:*literal*. Since arguments may be in conflict, a new, extended consequence relationship $\vdash_{\mathcal{T}}$ will be defined. Those wffs derivable from Γ via $\vdash_{\mathcal{T}}$ will correspond to dialectical trees. These new labelled wffs will therefore have the form *dialectical tree*:*conclusion*.

3.2 Some logical properties. Equivalence results

LDS_{ar} provides a useful formal framework for studying logical properties of defeasible argumentation in general, and of *DeLP* in particular. Equivalence results with other argumentative frameworks were also studied, particularly those relating *DeLP* with other LP-based formalisms.

Cummulativity was proven to hold for argumentative formulae. This let us think of a *DeLP* program as a knowledge base containing ‘atomic’ arguments (facts and rules), which can be later on extended by incorporating new, more complex arguments. This feature makes it easier to formalize *dialectical databases*, a TMS-based approach to defeasible argumentation which is currently being explored [2]. Cummulativity is proven *not* to hold for warranted conclusions, following the intuitions suggested by Prakken & Vreeswijk [15].

Superclassicality was shown to hold for both argument construction and warrant wrt SLD resolution. In other words, if $Th_{sld}(\Gamma)$ denotes the set of conclusions that can be obtained from Γ via SLD, then it holds that $C_{arg}(\Gamma) \subseteq Th_{sld}(\Gamma)$ and $C_{war}(\Gamma) \subseteq Th_{sld}(\Gamma)$, where C_{arg} and C_{war} stand for the consequence operator for argument construction and warrant, respectively. This implies, among other things, that the analysis of attack between arguments can be focused on literals in defeasible rules. Analogously, right weakening is proven to hold for both C_{arg} and C_{war}. This implies that (warranted) arguments with a conclusion x are also (warranted) arguments for y whenever $y \leftarrow x$ is present as a strict rule.

Another interesting issue concerns the definition of *variants* for LDS_{ar}. Since LDS_{ar} is a logical framework, its knowledge-encoding capabilities are determined by the underlying logical language, whereas the inference power is characterized by its natural deduction rules. Adopting a different KR language or modifying the existing inference rules will lead to different variants of LDS_{ar}. Thus, for instance, adopting a full first-order language will lead to a logical system with a behavior similar to the SL framework [19]. On the other hand, restricting the KR language to Horn clauses will result in a formulation closer to normal logic programming (NLP) under well-founded semantics.[4] Figure 1 summarizes some of these variants, and shows how they can be related to some existing argumentation frameworks, such as Simari-Loui's [19], MTDR [17], *DeLP* [3] and NLP (normal logic programming), conceptualized in an argumentative setting as suggested in [1]. Two distinguished variants of *DeLP* deserved particular attention, namely $DeLP_{not}$ and $DeLP_{neg}$ (*DeLP* restricted to default and strict negation, resp.). In [4], the relation between these variants of *DeLP* and normal logic programming was explored. Different criteria under which both strict and defeasible rules could be rewritten into a simpler but semantically equivalent form were defined.

The notion of dialectical tree and acceptable argumentation lines proved to be very useful for capturing different aspects of the process of argumentation. It should be remarked that similar approaches have been recently tried in other formalisms (see for example [13]). A formal analysis proved that dialectical trees can be pruned (following the procedure introduced in [17]) without affecting the marking procedure. An equivalence theorem between top-down and bottom-up computation of dialectical trees was also shown to hold.

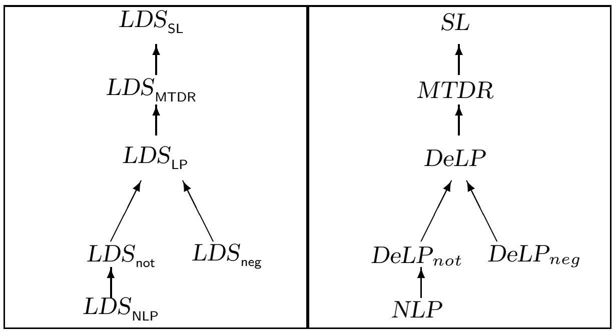

Fig. 1. A taxonomy relating the expressive power of LDS_{ar} and different argumentation systems

[4] A full discussion of different argumentative frameworks encompassed by LDS_{ar} can be found in [3].

4 Conclusions. Ongoing work

As we have shown in this paper, *DeLP* provides an LP-based setting for defeasible argumentation that combines the well-known advantages of the logic programing paradigm (such as declarativity and implementability) together with the dialectical considerations required to model argumentative processes. The *DeLP* language incorporates the natural benefits of allowing both strong and default negation, distinguishing at the same time between defeasible and non-defeasible information by introducing defeasible and strict rules. Specificity was adopted as the argument-comparison criterion, but it can be replaced by any other partial order among arguments without changing the rest of framework. Implementation issues deserved special consideration. In order to gain efficiency, the language was implemented using an abstract machine defined and implemented as an extension of the Warren Abstract Machine. Recent work showed how to extend *DeLP* capabilities into a multiagent environment [9].

During the last decade, a 'clash of intuitions' has appeared within the argumentation community [5, 15], where different, alternative approaches have been explored. As we have briefly sketched in the second part of this paper, having a logical formalism such as LDS_{ar} makes it easier to analyze, compare and relate different features associated with existing argumentative frameworks, providing at the same time a test-bed for studying other related issues (such as argumentation protocols, resource-bounded reasoning, etc.). These aspects are directly related to formalizing multiagent environments, in which argumentation plays a major role when modelling the communicative and reasoning abilities of the agents involved [12]. Research in this direction is currently being pursued.

References

1. Bondarenko, A. G., Dung, P. M., Kowalski, R. A., and Toni, F. An Abstract, Argumentation-Theoretic Approach to Default Reasoning. *Artificial Intelligence 93*, 1–2 (1997), 63–101.
2. Capobianco, M., Chesñevar, C. I., and Simari, G. R. A Formalization of Dialectical Bases for Defeasible Logic Programming. In *Proceedings of the 6th Internacional Congress of Informatics Engineering* (Capital Federal, Apr. 2000), Universidad Buenos Aires.
3. Chesñevar, C. I. *Formalizating Processes of Defeasible Argumentation as Labelled Deductive Systems.* PhD thesis, Departamento de Ciencias de la Computación, Universidad Nacional del Sur, Bahía Blanca, Argentina, Jan. 2001. `http://cs.uns.edu.ar/~cic`.
4. Chesñevar, C. I., Dix, J., Stolzenburg, F., and Simari, G. R. Relating Defeasible and Normal Logic Programming through Transformation Properties. *Theoretical Computer Science (submitted)* (2000).
5. Chesñevar, C. I., Maguitman, A., and Loui, R. Logical Models of Argument. *ACM Computing Surveys 32*, 4 (Dec. 2000), 337–383.
6. Dung, P. M. On the Acceptability of Arguments and its Fundamental Role in Nomonotonic Reasoning and Logic Programming. In *Proc. of the 13th. International Joint Conference in Artificial Intelligence (IJCAI), Chambéry, Francia* (1993), pp. 321–357.

7. Gabbay, D. *Labelling Deductive Systems (vol.1).* Oxford University Press (Volume 33 of Oxford Logic Guides), 1996.
8. García, A. J. *Defeasible Logic Programming: Language, Operational Semantics and Parallelism.* PhD thesis, Departamento de Ciencias de la Computación, Universidad Nacional del Sur, Bahía Blanca, Argentina, Dec. 2000. http://cs.uns.edu.ar/∼ajg.
9. García, A. J., Gollapally, D., Tarau, P., and Simari, G. Deliberative stock market agents using jinni and defeasible logic programming. In *Proceedings of ESAW'00 Engineering Societies in the Agents' World, Workshop of ECAI 2000* (Aug. 2000).
10. García, A. J., Simari, G. R., and Chesñevar, C. I. An Argumentative Framework for Reasoning with Inconsistent and Incomplete Information. In *Proceedings of the 13th European Conference on Artificial Intelligence, Workshop on Practical Reasoning and Rationality* (Aug. 1998), pp. 13–19. `http://cs.uns.edu.ar/giia.html`.
11. Hunter, A. Defeasible reasoning with structured information. In *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning (KR'94)* (1994), M. Kaufmann, Ed., pp. 281–292.
12. Parsons, S., Sierrra, C., and Jennings, N. Agents that Reason and Negotiate by Arguing. *Journal of Logic and Computation 8* (1998), 261–292.
13. Prakken, H. Relating protocols for dynamic dispute with logics for defeasible argumentation. *Synthese (to appear)* (2000).
14. Prakken, H., and Sartor, G. Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-classical Logics 7* (1997), 25–75.
15. Prakken, H., and Vreeswijk, G. Logical systems for defeasible argumentation. In *Handbook of Philosophical Logic*, D. Gabbay, Ed. Kluwer Academic Publisher, 1999.
16. Reiter, R. A Logic for Default Reasoning. *Artificial Intelligence 13*, 1,2 (Apr. 1980), 81–132.
17. Simari, G. R., Chesñevar, C. I., and García, A. J. The Role of Dialectics in Defeasible Argumentation. In *Proceedings of the XIV Conferencia Internacional de la Sociedad Chilena para Ciencias de la Computación* (Nov. 1994), pp. 111–121. `http://cs.uns.edu.ar/giia.html`.
18. Simari, G. R., and Loui, R. P. Confluence of Argument Systems: Poole's Rules Revisited. In *Proc. of the 3rd. International Workshop on Nonmonotonic Reasoning* (South Lake Tahoe, California, June 1990), pp. 223–232.
19. Simari, G. R., and Loui, R. P. A Mathematical Treatment of Defeasible Reasoning and its Implementation. *Artificial Intelligence 53*, 1–2 (1992), 125–157.
20. Vreeswijk, G. A. *Studies in Defeasible Argumentation.* PhD thesis, Vrije University, Amsterdam (Holanda), 1993.