Relating DeLP and Default Logic

Telma Delladio, Guillermo R. Simari

Laboratorio de Investigación y Desarrollo en IA
Dpto de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Bahía Blanca - B8000CPB - ARGENTINA*
{td,grs}@cs.uns.edu.ar

Abstract

In this paper, we establish a relation between an argumentation based system: Defeasible Logic Programming (DELP), and a nonmonotonic system: Reiter's Default Logic. This relation is achieved by introducing a variant of DELP and a transformation that maps default theories to defeasible logic programs. The transformation allows to associate the answers of a DELP Interpreter with the consequences, credulous and skeptical, of the default theory. Thus, this work establishes a link between a well understood nonmonotonic system and a argumentation based system.

Keywords: Knowledge Representation, Nonmonotonic Reasoning, Argumentative Reasoning, Default Logic, Defeasible Logic Programming.

1 Introduction and Motivation

In general, it is interesting and important to compare, analyze and assess the alternative tools that could be used to confront a specific problem. In particular, in the area of Artificial Intelligence there are several research lines dedicated to the development of formalisms and tools regarding Knowledge Representation. These formalisms are so diverse that many times it is difficult to recognize their advantages, disadvantages and differences in order to make a plausible use of them. For this reason, it is interesting to analyze the relation among knowledge representation formalisms to evaluate their differences and similarities. Several works relating diverse approaches of defeasible and non-monotonic reasoning have been developed [8, 7, 5, 6, 2, 3].

In this paper, we analyze the relation between an argumentation based system like Defeasible Logic Programming (DELP), and a nonmonotonic system like Reiter's Default Logic. In order to establish this relation we introduce (Section 3) a variant of DELP, called DELP$^{\emptyset}$, and a number of properties it verifies. Then, we define a transformation (Seccion 5) that allows to map default theories to defeasible logic programs. The transformation allows to associate the answers of a DELP$^{\emptyset}$ interpreter with the consequences, credulous and skeptical, of the original default theory. Finally, we relate the results of this work with the Dung's argumentative framework for Default Logic defined in [9], and we briefly discuss how the relation established between Default Logic an DELP can be used to relate DELP to other meaningful nonmonotonic formalisms.

*This work was partially supported by CONICET (PIP 5050), the *Agencia Nacional de Promoción Cient´ ca y Tecnológica* (PICT 13096, 15043, and PAV 076) and the *Secretar´a de Ciencia y Tecnolog´a of Universidad Nacional del Sur* (24/N016). Telma Delladio is partially supported by CONICET

2 DeLP

Defeasible Logic Programming (DeLP) is a formalism that combines Logic Programming and Defeasible Argumentation. In DeLP, knowledge is represented using facts, strict rules or defeasible rules:

- *Facts* are ground literals representing atomic information or the negation of atomic information using the strong negation "$\neg$ " (*e.g.* $\neg$ *rain*).
- *Strict Rules* are denoted $L_0 \leftarrow L_1, \ldots, L_n$, where the *head* L_0 is a ground literal and the *body* $\{L_i\}_{i>0}$ is a set of ground literals (*e.g.* $\neg\, day \leftarrow night$).
- *Defeasible Rules* are denoted $L_0 \prec L_1, \ldots, L_n$, where the *head* L_0 is a ground literal and the *body* $\{L_i\}_{i>0}$ is a set of ground literals. (*e.g.* $cold \prec winter$).

Syntactically, the symbol "$\prec$" is all that distinguishes a defeasible rule from a strict one. Pragmatically, a defeasible rule is used to represent defeasible knowledge, *i.e.* tentative information that may be used if nothing could be posed against it. A defeasible rule "Head $\prec$ Body." is understood as expressing that "*reasons to believe in the antecedent* Body *provide reasons to believe in the consequent* Head" [14].

A Defeasible Logic Program (de.l.p.) $\mathcal{P}$ is a set of facts, strict rules and defeasible rules. When required, $\mathcal{P}$ is denoted (Π, Δ) distinguishing the subset Π õf facts and strict rules, and the subset $\Delta$of defeasible rules. Observe that strict and defeasible rules are ground.

Strong negation is allowed in the head of program rules, and hence may be used to represent contradictory knowledge. From a program (Π, Δ) contradictory literals could be derived, however, the set Π (which is used to represent non-defeasible information) must possess certain internal coherence. Therefore, Π has to be non-contradictory, *i.e.* no pair of contradictory literals can be derived from Π. Given a literal L the complement with respect to strong negation will be denoted $\overline{L}$ (*i.e.* $\overline{a}$ = $\neg\, a$ and $\overline{\neg\, a}$ = a).

DeLP incorporates an argumentation formalism for the treatment of the contradictory knowledge that can be derived from (Π, Δ) This formalism allows the identification of the pieces of knowledge that are in contradiction. A dialectical process is used for deciding which information prevails. In particular, the argumentation-based definition of the inference relation makes it possible to incorporate a treatment of preferences in an elegant way.

In DeLP a literal L is *warranted* from (Π, Δ) if there exists a non-defeated argument $\mathcal{A}$ supporting L. In short, an *argument* for a literal L, denoted $\langle \mathcal{A}, L\rangle$, is a minimal set of defeasible rules $\mathcal{A}\subseteq\Delta$ such that $\mathcal{A}\cup\Pi$ is non-contradictory and there is a derivation for L from $\mathcal{A}\cup\Pi$. In order to establish if $\langle \mathcal{A}, L\rangle$ is a non-defeated argument, *argument rebuttals* or *counter-arguments* that could be *defeaters* for $\langle \mathcal{A}, L\rangle$ are considered, *i.e.*, counter-arguments that by some criterion are preferred to $\langle \mathcal{A}, L\rangle$. An argument $\langle \mathcal{A}_1, L_1\rangle$ counter-argues or *attacks* $\langle \mathcal{A}_2, L_2\rangle$ at some literal h, if and only if there exists a sub argument $\langle \mathcal{A}, h\rangle$ of $\langle \mathcal{A}_2, L_2\rangle$ (*i.e.* $\mathcal{A} \subseteq \mathcal{A}_2$) such that h and L_2 disagree; that is, $\Pi \cup \{h, L_2\}$ is contradictory.

Since counter-arguments are arguments, there may exist defeaters for them, and defeaters for these defeaters, and so on. Thus, a sequence of arguments called *argumentation line* is constructed, where each argument defeats its predecessor in the line. Some restrictions are imposed over these lines to be considered *acceptable argumentation lines*.

- Non circularity: circular argumentation lines are not permitted.
- Concordance: the set of supporting arguments must be non contradictory and the same is required for interfering arguments.
- Blocking-Blocking situations: if a blocking defeater A_i occurs in the line $[A_1, \ldots, A_k]$, A_{i+1} cannot be a blocking defeater for A_i.

Usually, each argument has more than one defeater and more than one argumentation line exists. Therefore, a tree of arguments called *dialectical tree* is constructed, where the root is $\langle \mathcal{A}, h\rangle$ and each path from the root to a leaf is an argumentation line. A *dialectical analysis* of this tree is used for deciding whether a literal is warranted. This dialectical analysis is carried out labeling the arguments conforming the dialectical tree. The arguments in the leaves of the tree are considered undefeated. Every inner node with at least a child marked as undefeated, is considered and marked as a defeated argument. In the other case, it is undefeated. Following this analysis, a literal h is said warranted if there is a dialectical tree where the root is an argument for h that has been marked as undefeated (for a detailed explanation of this dialectical process see [10]).

In DeLP, given a query Q there are four possible answers: YES, if Q is warranted; NO, if the complement of Q is warranted; UNDECIDED, if neither Q nor its complement is warranted; and UNKNOWN, if Q is not in the language of the program.

3 DeLP$^{\emptyset}$ variant

In DeLP, several elements can be adjusted thus defining a number of variants of DeLP; for instance, the notions of attack and defeat, as well as the conditions required for acceptable argumentation lines. We will consider a DeLP variant, that we call DeLP$^{\emptyset}$, observing the following condition:

- The relation defining the comparison criterion is the empty set.

In general, given two conflicting arguments A and B, they can be compared using some criterion. In that case, if A is better than B, A is a *proper defeater* for B. But, if neither of the two is better than the other, A is a *blocking defeater* for B, and vice versa. Note that, in DeLP$^{\emptyset}$, since the comparison criterion is empty, every attack is a blocking defeat and since there are no proper defeaters, this criterion turns attack into defeat.

Remark 3.1
Every argumentation line in DeLP$^{\emptyset}$ contains two arguments at most.

Suppose there is an acceptable argumentation line $\Gamma = [A_1, \ldots, A_n]$, $n > 2$. In such case, there is a subsequence of arguments $[A_i, A_{i+1}, A_{i+2}]$ in Γ. Since every defeater in DeLP$^{\emptyset}$ is a blocking defeater, A_{i+2} is a blocking defeater for A_{i+1} and, A_{i+1} is a blocking defeater for A_i. But, in this case, Γ would not be an acceptable argumentation line, because there cannot be two consecutive blocking attacks (see the third condition of an acceptable argumentation line).

Remark 3.2
Every dialectical tree in DeLP$^{\emptyset}$ has, at most, two levels.

Since every path of a dialectical tree is an acceptable argumentation line, and in DeLP$^{\emptyset}$, argumentation lines are composed by one or two arguments, every path contains, at most, two arguments. Thus, every dialectical tree has, at most, two levels.

Remark 3.3
Every dialectical tree whose root is marked as *undefeated* is a tree with just one node.

If the argument of the root has a child, this means that the root has a defeater and the root is then defeated (since, from remark 3.1: there are no defeaters for the defeaters)

Remark 3.4
In DeLP$^{\emptyset}$, a literal l is *warranted* iff any argument for l is not attacked.

If a literal l is *warranted* there is a dialectical tree, for an argument A supporting l, whose root is marked as *undefeated* (from the definition of warranted literal in DeLP). This dialectical tree has a single node (from remark 3.3) and this means that there is no argument attacking it. If there is some argument that attacks the root A, it has to be in the tree and then the root would be marked as *defeated*.

Remark 3.5
In DeLP$^{\emptyset}$ an argument A is warranted iff every literal present in A is warranted.

This condition establishes that all literals contained in the defeasible derivation that constitutes a warranted argument are also warranted. Suppose this is not true, then exists a warranted argument A such that a literal L_i present in A is not warranted. In this case, every argument for L_i is defeated and L_i is an attack point in the argument A. Therefore, A is attacked and defeated (from remark 3.4), which cannot happen, since we assumed that A is a warranted argument.

As mentioned, in DeLP, two literals p and q disagree if $\Pi \cup \{p, q\}$ is a contradictory set (Π is the set of strict rules). Then, if Π is empty, p and q must be complementary literals.

Remark 3.6
Let A be an argument in DeLP$^{\emptyset}$, if a literal p is present in A and there is an argument B for q such that p and q disagree then A is not warranted.

In this case, B attacks A in p, for this reason A is defeated.

Remark 3.7 (Valid for general de.l.p.)
If there are no strict rules, p and q disagree iff $p \equiv \overline{q}$.

4 Default Logic

A Default Theory $T = \langle W, D\rangle$ consists of a set of facts W of ground sentences. Each default rule in D has the form $\frac{a : b_1, \ldots, b_n}{\ldots}$ (sometimes written $a :$

$b_1, \ldots, b_n/c$), where a is called the prerequisite, b_i are the justifications and c is the consequent of the default. When the justification and the consequent of a default rule are the same, $\frac{a:c}{c}$, the default rule is called a *normal default rule*. In general $just(\delta)$ denotes the set of justifications present in the rule δ, and given a set of default rules R, $just(R)$ is used to denote all the justifications present in the default rules of R.

The intuitive meaning of a default is: if a can be derived and it is possible to consistently assume each b_i, then conclude c. Given a default theory $T = \langle W, D\rangle$ an extension E (or a Reiter extension) is a theory E satisfying that

$$E = \bigcup\{W_i \mid \textit{ i is a natural number }\}$$

$$W_0 = W$$

$$W_{i+1} = Th(W_i) \cup \{\gamma \mid (\exists \tfrac{\alpha:\beta_1,\ldots,\beta_n}{\gamma} \in D) \wedge (\{\beta_i\} \cup E \nvdash \bot, \forall i, 1 \leq i \leq n) \wedge (\alpha \in W_i)\}$$

Another way to characterize extensions in Default Logic is through an operational semantics [1]. In this characterization each extension is defined by a set $In(\Pi)$, where Π is a closed and successful process. Given a sequence of default rules $S = \langle \delta_0, \ldots, \delta_n\rangle$ the set $In(S)$ collects the information obtained by the application of the defaults in S; that is, $In(S) = Th(W \cup \{\gamma \mid \frac{\alpha:\beta}{\gamma} \textit{ occurs in } S)\})$ Then, a process is a special kind of sequence of default rules $\langle \delta_0, \ldots, \delta_n\rangle$ where each default δ_k is applicable to $In(\langle \delta_0, \ldots, \delta_{k-1}\rangle)$. A process Π is closed if there is no applicable default rule δ in D such that δ does not occur in Π, and a process Π is successful if $In(\Pi) \nvdash \overline{\beta}$ for all β that is a justification of some default rule in Π. Given a default theory $T = \langle W, D\rangle$, a literal l is a skeptical consequence of T if l belongs to every extension of T, and l is a credulous consequence of T if l is present in some extension but not in each extension. A default theory that has at least one extension is called *coherent*.

In this work, we will consider finite propositional default theories with the following restrictions:

1. The theory $T = \langle W, D\rangle$ is coherent.
2. The set of facts W is empty.
3. For every default $\alpha : \beta/\gamma$, formulas β and γ are single literals.

Some of these restrictions could be dropped and we will analyse this situation later. We are interested, at this stage, in default theories that verify the condition: $W = \emptyset$, since working with these theories enable us to establish an indirect relation between DELP an another nonmonotonic formalisms. In particular, it is well known the works that study the relation between Normal Logic Programming and Default Logic. This connection is achieved through a link between stable models for normal logic programs [11] and skeptical consequences of default theories [4]. Normal logic programs are translated into a default theory composed by an empty set of facts, and a set of default rules obtained as follows. Each rule of the form:

$$c \leftarrow a_1 \ldots, a_n, \textit{not } b_1, \ldots, \textit{not } b_m$$

is translated into a default rule of the form:

$$a_1, \ldots, a_n : \neg\, b_1, \ldots, \neg\, b_m/c$$

In this way, a relation between DELP and this type of default theories (with an empty set of facts) establishes a indirect link between DELP and Normal Logic Programming.

5 Translating Default Theories into DELP$^{\emptyset}$ programs

In this section, we present a transformation that allows to map default theories to defeasible logic programs. The transformation is defined for default theories that follow the restrictions given in section 4.

Given a default theory $T = \langle \emptyset, D\rangle$, we transform T into a de.l.p. $P = (\emptyset, \Delta)$ as follows:

1. For each default $\delta_i = \alpha : \beta/\gamma \in D$, the set Δ in the de.l.p. P includes the following *defeasible rules*:

   (i) $\gamma \prec \alpha, p_i$
   (ii) $\neg\, p_i \prec \overline{\beta}$
   (iii) $p_i \prec$

   where p_i is a new literal and rules (ii) and (iii) are called *guard rules*, and $\overline{\beta}$ is the complement of β.

   When necessary, we denote the de.l.p. P as $\mathcal{T}^{DL}_{delp}(T)$

The first defeasible rule (i), indicates that if the prerequisite α is given, then the consequent γ could be derived. However, this is only allowed if it is possible to consistently assume the justification, β, and this restriction is verified when the second rule does not apply (ii). If the complement of the justification, (*i.e.* $\overline{\beta}$), is derived, there exists a derivation (and an argument) for $\neg\, p_i$ and this constitutes an attack to the argument for α. The last rule introduced by the translation (iii) is simply used to enable, by default, the new literal p_i.

The translation for a default rule $\delta = \alpha : \beta/\gamma$ introduces a new literal p_i to block the derivation of γ. That is, in case that the complement of the justification is derived, literal p_i turns into an attack point, and the argument for the consequence will be attacked. Therefore, this argument is defeated classifying the literal γ as UNDECIDED.

Note that default rules with an empty prerequisite are written $true : \beta/\gamma$. These rules are translated, in the same way, to the following defeasible rules:

(i) $\gamma \prec p_i$ (ii) $\neg\, p_i \prec \overline{\beta}$ (iii) $p_i \prec$

Then, we will show that normal default rules can be translated in a simpler, reduced form.

Example 1
Consider the default theory $T_1 = \langle \emptyset, D_1 \rangle$, where

$$D_1 = \{(true : a/a), (a : \neg\, x/y), (a : \neg\, y/x), (a : d/d)\}$$

The corresponding de.l.p. will be $P_{T_1} = (\emptyset, \Delta_1)$, where Δ_1 has the following defeasible rules.

D_1	(i)	(ii)	(iii)
$true : a/a$	$a \prec p_1$	$\neg\, p_1 \prec \neg\, a$	$p_1 \prec$
$a : \neg\, x/y$	$y \prec a, p_2$	$\neg\, p_2 \prec x$	$p_2 \prec$
$a : \neg\, y/x$	$x \prec a, p_3$	$\neg\, p_3 \prec y$	$p_3 \prec$
$a : d/d$	$d \prec a, p_4$	$\neg\, p_4 \prec \neg\, d$	$p_4 \prec$

Each default rule is translated into a defeasible rule, using an extra literal (p_i) acting as a guard. A derivation for $\neg\, p_i$ implies that the justification (from the original default rule) cannot be assumed consistently. In this way, the transformation captures, through these three defeasible rules, the behavior of the original default rule.

Example 2
Consider a de.l.p. $P_{T_2} = (\emptyset, \Delta_2)$, obtained from a default theory $T_2 = \langle \emptyset, D_2 \rangle$, where Δ_2 has the rules:

D_2	(i)	(ii)	(iii)
$x : a/b$	$b \prec x, p_1$	$\neg\, p_1 \prec \neg\, a$	$p_1 \prec$
$y : \neg\, b/c$	$c \prec y, p_2$	$\neg\, p_2 \prec b$	$p_2 \prec$
$true : x/x$	$x \prec p_3$	$\neg\, p_3 \prec \neg\, x$	$p_3 \prec$

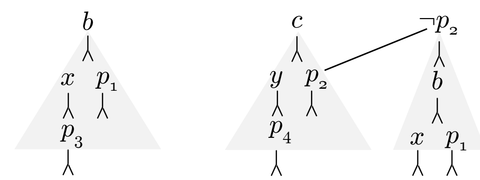

Figure 1: Arguments in P_{T_2}

This example shows the use given to the new literals p_i introduced in the translation. Literal p_2 determines an attack point in the argument for the literal c and this argument is defeated (see remark 3.5 and figure 1). For this reason, literal c is not warranted in DELP$^{\emptyset}$, it is an UNDECIDED literal. This attack reflects the incompatibility between the original default rules $y : \neg\, b/c$ and $x : a/b$. In the original default theory, literal c is a credulous consequence, since no successful process includes both default rules. There is only a successful process including $x : a/b$.

It is possible to identify in the de.l.p. P obtained by the translation, two kind of attacks. Note that, every attack in P reflects the existence of two applicable default rules that are incompatible. This incompatibility arises for one of the following reasons:

- the consequences of both default rules are contradictory, or
- the consequence of one of them is contradictory with the justification of the other

Figure 2 depicts an attack that arise from contradiction between the consequences of two default rules, and figure 3, an attack over a justification. In this case the (artificial) attack point is the new literal introduced by the translation.

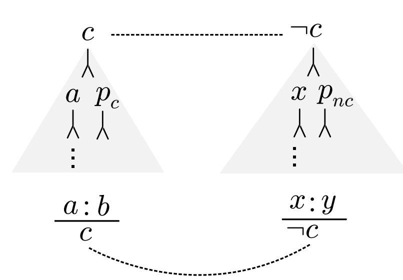

Figure 2: Consequence attack

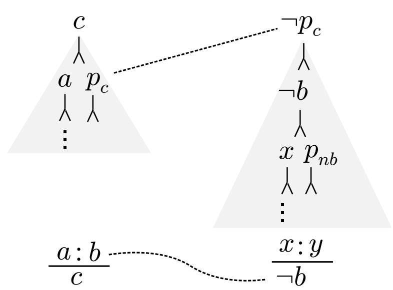

Figure 3: Justification attack

It is interesting to note, that normal default rules could be translated in a more concise manner. The translation of a general default rule $\alpha : \beta/\gamma$ has to model the two main characteristics captured by a default rule:

- the antecedent α is needed to derive the consequent γ (i).
- there is no knowledge against the justification β (ii).

This translation is required to model, in DeLP$^{\emptyset}$, the interaction between conflicting information in the same way it is done in Default Logic. Direct conflicts between default rules arise when their consequents are contradictory or the consequent of one default is contradictory with the justification of the other. For normal default theories justifications and consequents are the same, therefore, a direct conflict between default rules arises when it is possible to derive information against the consequent of a default rule. For this reason it is possible to give a reduced translation for normal default rules.

Reduced transformation
If the variant considered is DeLP$^{\emptyset}$, the translation of a *normal* default rule $\alpha : \gamma/\gamma$ can be reduced to a single defeasible rule: $\gamma \prec \alpha$.

That is, in the reduced transformation any argument for $\neg\gamma$ attacks the argument for γ (see figure 4). This attack establishes a defeat, because in DeLP$^{\emptyset}$, attack determines defeat. In the initial transformation, any argument for $\neg\gamma$ allows the formation of an argument for the literal $\neg p_i$ that attacks (block), in the same way, the argument for γ. In this way, both transformations reflect, in the obtained DeLP$^{\emptyset}$ program, the same pretended behaviour of the original default logic.

Hence, in what follows, normal default rules will be translated using the reduced form.

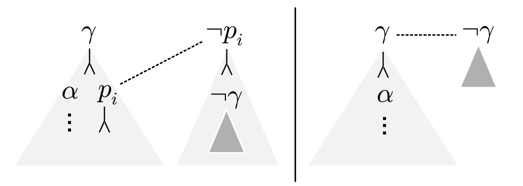

Figure 4: Reduced transformation for normal defaults

In this way, a DeLP$^{\emptyset}$ program obtained by translating the normal default rules into the general form or into the reduced form, models in the same way the original default theory. The dialectical analysis that could be carried out in any of these translations is equivalent, and this is because of the comparison criterion. The only kind of defeaters present in DeLP$^{\emptyset}$ are blocking defeaters. For this reason, if an argument has two defeaters both are blocking defeaters. The elimination or addition of defeaters does not change the scenario: the main argument remains defeated (see figure 5). These characteristics are proper of DeLP$^{\emptyset}$, since using a different comparison criterion proper defeats can arise and, in these situations, the elimination of one defeater could provoke others defeaters to change their status (see figure 6).

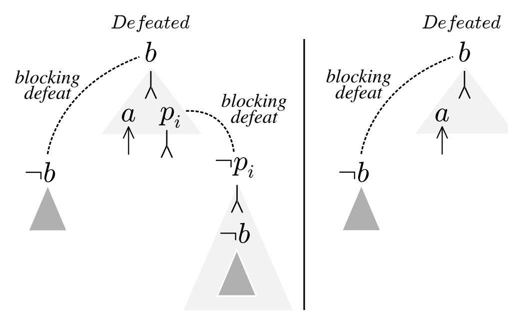

Figure 5: Defeats in DeLP$^{\emptyset}$

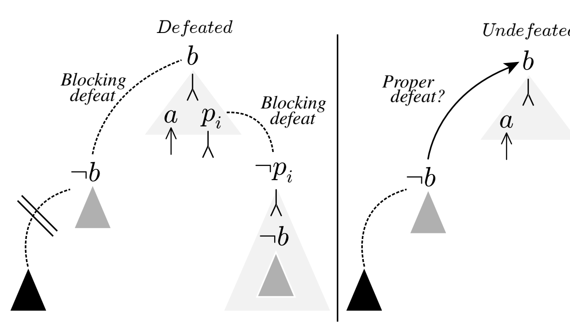

Figure 6: Defeats in general DeLP

Relation between DL and DELP$^{\emptyset}$

Let $T = (W, D)$ be a default theory, such that $W = \emptyset$, P the de.l.p. obtained by the translation proposed, and l a literal.

- Literal l is a skeptical consequence of T iff l is warranted from P.

- Literal l is a credulous consequence of T iff l is undecided in P

In order to understand this relation, we can analyze the relation between the processes determining the extensions of the default theory, and the arguments that can be constructed using the defeasible rules obtained by the translation. Note that we are considering just the original literals in the theory, not the p_i literals added by the translation.

Given a default rule $\delta_i = \alpha : \beta/\gamma$ we define $t(\delta_i)$ as the set of defeasible rules obtained by the translation of a default rule δ_i. Thus, in the general case $t(\delta_i) = \{(\gamma \prec \alpha, p_i), (\neg\, p_i \prec \overline{\beta}), (p_i \prec\ )\}$. In the same way, we define the set of defeasible rules obtained by the entire set of default rules D as $t(D) = \bigcup t(\delta_i), \forall \delta_i \in D$. Finally, given a de.l.p. $P = (\Pi, \Delta)$ and a set $R \subseteq \Delta$ we denote $args_P(R) = \{A : \langle A, h\rangle$ *is an argument structure in* P *and* $A \subseteq R\}$.

Now, let Γ be a closed and successful process of a default theory $T = \langle \emptyset, D\rangle$, $E = In(\Gamma)$ be the corresponding extension, Γ_s be the set of defaults rules in Γ (*i.e.* $\Gamma_s = \{\delta \mid \delta$ *occurs in* $\Gamma\}$), and $P = \mathcal{T}^{DL}_{delp}(T)$ be the de.l.p. obtained by the translation proposed. Note that arguments in $args_P(t(\Gamma_s))$ are conflict free; that is, for all argument A in $args_P(t(\Gamma_s))$ there is no other argument B attacking A. Otherwise, conflicting default rules would belong to Γ, and this is not possible since Γ is a successful process. Moreover, $args_P(t(\Gamma_s))$ is a maximal set of non-conflicting arguments since every argument B that does not attack an argument in $args_P(t(\Gamma_s))$ comes from default rules that are not in conflict with the rules in Γ. If such argument B exists, Γ would not be closed.

In this way, if a literal w is warranted from P there exists a non attacked argument $\langle A_w, w\rangle$ that is in every maximal conflict-free set of arguments. Then, literal w will be in every extension of T and it is a skeptical consequence of T. On the other hand, given a literal u if every supporting argument $\langle A_u, u\rangle$ is attacked by other argument $\langle B_{u'}, u'\rangle$ both arguments have to be in different conflict-free sets of arguments. Therefore, two or more extensions exists and literal u cannot be present in all of them. For this reason, u will be a credulous consequence of T. Note that, u is undecided in $\mathcal{T}^{DL}_{delp}(T)$.

5.1 Dung's framework for DL

The relation established between Default Logic and DELP is, in some aspects, similar to the one established in Dung's work [9] which considers a default theory as an argumentation framework. There, an argumentation framework $AF(T) = \langle AR_T, attacks_T\rangle$ is defined for a default theory $T = \langle W, D\rangle$, where:

- $AR_T = \{(K, k) \mid K \subseteq just(D) : K$ *is a support for* $k\}$

- $(K, k)\ attacks_T\ (K', k')$ iff $\overline{k} \in K'$

A set K is said to be a support for k with respect to T if there exists a *default derivation* $k_1, k_2 \ldots, k_m$ with $k_m = k$ such that for each k_i , either $k_i \in W$, or k_i is consequence of the preceding elements in the sequence or $k_i = \gamma$ for a default rule $\frac{\alpha : \beta_1,\ldots,\beta_n}{\gamma}$ such that α is a previous element in the sequence and every β_i is in K.

Defining this framework, Reiter's extensions of a default theory $T = \langle W, D\rangle$ can be associated to the stable extensions of AR_T. Remember that in Dung's framework a set of arguments S is a stable extension iff $S = \{A \mid A$ *is not attacked by any argument in* $S\}$. On the one hand, given a set of arguments A in AR_T the set of consequences it supports are defined: $flat(A) = \{k \mid \exists(K, k) \in A\}$. On the other hand, given a set of consequences E, the set of arguments that are consistent with it, is also defined: $ARGS(E) = \{(K, k) \in AR_T \mid \forall j \in K, \{j\} \cup E \not\vdash \bot\}$. Therefore, E is a Reiter's extension of $T = \langle W, D\rangle$ iff $E = flat(ARGS(E))$.

Considering the transformation proposed, we can see that arguments from $\mathcal{T}^{DL}_{delp}(T)$ can be used to identify the arguments of this argumentative framework. Suppose that $\langle A, t\rangle$ is an argument in the de.l.p.$\mathcal{T}^{DL}_{delp}(T)$, we can define the following set of literals:

$$K_A = \{l : (\neg\, p_i \prec \neg\, l) \in \mathcal{T}^{DL}_{delp}(T) \textit{ and } (p_i \prec\ ) \in A\}$$

This set K_A constitutes a support for literal t with respect to T. That is, $(K_A, t) \in AR_T$ in Dung's framework. Observe that every argument $\langle A, t\rangle$ in $\mathcal{T}^{DL}_{delp}(T)$ is constructed using the defeasible rules obtained by the translation. However, the existence of a supporting argument for t is because of the presence of a set of default rules (in the original default theory) that allows a *default derivation S* for t. The default rules that could be used in $T = \langle W, D\rangle$ for constructing *S* are identified by the literals p_i mentioned in A. If a defeasible rule $p_i \prec$ is present in A, the default rule δ_i is used for the construction of *S*, and this means that its justifications, $just(\delta_i)$, can be consistently assumed.

5.2 Dropping some restrictions

In section 4 we establish some restrictions for the default theories considered. On the one hand, we are considering default rules such that their justification are single literals. This restriction can be dropped, since given a default rule of the form $\alpha : \beta/\gamma$ where $\beta = \beta_1, \ldots, \beta_n$ with each β_i is a single literal, the translation is given by the rules

(i) $\gamma \prec \alpha, p_{i_1}, \ldots, p_{i_n}$
(ii) $\neg\, p_{i_k} \prec \overline{\beta_k}$ for all $1 \leq k \leq n$
(iii) $p_{i_k} \prec$ for all $1 \leq k \leq n$

On the other hand, we are considering only default theories with an empty set of facts. This condition could be dropped translating all clauses in W as strict rules in the de.l.p.. In this case, W has to be consistent and each formula is translated as a set of contrapositive rules as strict rules in the de.l.p.. For each clause $C = (c_1 \vee \ldots \vee c_n)$ in W we include, for all i $(1 \leq i \leq n)$, the *strict rules* in the de.l.p.:

$$c_i \leftarrow \neg\, c_1, \ldots, \neg\, c_{i-1}, \neg\, c_{i+1}, \ldots, \neg\, c_n$$

Example 3
Given the theory $T_3 = \langle W_3, D_3 \rangle$ where $W_3 = \{(x), (w), (t \rightarrow \neg\, b)\}$ and $D_3 = \{(x : b/b), (w : t, r/q)\}$. The associated de.l.p. P_3 has the rules:

$x \leftarrow$	$b \prec x$	$\neg\, p_t \prec \neg\, t$	$p_t \prec$
$w \leftarrow$	$q \prec w, p_t, p_r$	$\neg\, p_r \prec r$	$p_r \prec$
$\neg\, b \leftarrow t$			
$\neg\, t \leftarrow b$			

Finally, note that we are considering *coherent* default theories (*i.e.* the existence of extensions are guaranteed). As mentioned in [9] having default theories with default rules of the form $\alpha : \beta/\neg\, \beta$ prevents to conclude any literal, since this kind of defaults collapse the theory, and none extension can be obtained. Under the preferred semantics (instead of the stable ones), this non intuitive behavior is avoided because this paradoxical default does not interfere with the others. In case of DeLP$^{\emptyset}$ and the translation proposed, the behavior will be similar in the case of non coherent default theories. The defeasible rules obtained by the translation of this conflicting kind of defaults will not interfere with the arguments supported by meaningful defaults.

5.3 Answers and Extensions

It is interesting to note that the relation established in this work associates types of consequences (skeptical or credulous) from a Default theory with the answers (YES, NO, UNDECIDED) given by a DeLP$^{\emptyset}$ interpreter. However, the concept of extension, present in Default Logic, is not clearly recognizable in DeLP$^{\emptyset}$ and for this reason it is not possible, without an extra analysis, to identify the notion of extension in the defeasible program obtained by the translation. For instance, if two literals are UNDECIDED in a $\mathcal{T}^{DL}_{delp}(T)$, they could belong to the same extension of T, or they could belong to different extensions. Hence, the match between literals and extensions cannot be recognized by DeLP$^{\emptyset}$ by considering just the answer given by the interpreter. An external mechanism should be provided.

6 Conclusions

The transformation presented in this work allows to map Default Theories to a special variant of DeLP (the simplest variant). In this way, default theories can be modeled by simple de.l.p.'s and this result allow us to extend this work to other formalisms, mainly over those whose correspondence with Default Logic (of some of its variants) have been already defined.

There are several works, in the field of Knowledge Representation dedicated to relate different formalisms and semantics of nonmonotonic reasoning formalisms [3, 5, 9, 7, 12, 13, 2, 8]. We think that it is significant to carry out this work since, as mentioned before, it is important to asses the different alternatives present in the area. There are very different approaches for nonmonotonic reasoning and is useful to clarify the relationship among them. However, this task is not easy mainly because several dissimilar approaches have been developed. This work presents a first analysis on the relation between a well understood nonmonotonic system as Reiter Default Logic, and a argumentation based system like DeLP. Many works have been developed relating Default Logic, or some of its variants, to other nonmonotonic formalisms [9, 7, 12].

Acknowledgements

This work was partially supported by the *Consejo Nacional de Investigaciones Científicas y Técnicas* -CONICET- (PIP 5050), the *Agencia Nacional de Promoción Científica y Tecnológica* (PICT 13096, 15043, and PAV 076) and the *Secretaría de Ciencia y Tecnología of Universidad Nacional del Sur* (24/N016). Telma Delladio is partially supported by CONICET. Thanks to Nicolás Rotstein for his useful comments and remarks.

References

[1] Grigoris Antoniou. *Nonmonotonic Reasoning*. The MIT Press, Cambridge, Massachusetts, 1997.

[2] Grigoris Antoniou, Michael Maher, David Billington, and Guido Governatori. Comparison of sceptical naf-free logic programming approaches. *Logic Programming and Nonmonotonic Reasoning*, 1730:347–356, 1999.

[3] Grigoris Antoniou, Michael J. Maher, and David Billington. Defeasible logic versus logic programming without negation as failure. *Journal of Logic Programming*, 42(1):47–57, 2000.

[4] Chitta R. Baral and V. S. Subrahmanian. Dualities between alternative semantics for logic programming and nonmonotonic reasoning, September 22 1998.

[5] G. Brewka. On the relation between defeasible logic and well-founded semantics. *LPNMR*, pages 121–132, 2001.

[6] Gerhard Brewka. Well-founded semantics for extended logic programs with dynamic preferences. *Journal of Artificial Intelligence Research*, 4:19–36, 1996.

[7] Gerhard Brewka and Georg Gottlob. Well-founded semantics for default logic. *Fundamenta Informaticae*, 31(3/4):221–236, 1997.

[8] 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, 2002.

[9] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reason, logic programming, and N-person games. *Artificial Intelligence*, 77:321–357, 1995.

[10] Alejandro García and Guillermo Simari. Defeasible logic programming, an argumentative approach. *Theory and Practice of Logic Programming*, 4(1):95–138, 2004.

[11] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In Robert A. Kowalski and Kenneth Bowen, editors, *Proceedings of the Fifth International Conference on Logic Programming*, pages 1070–1080, Cambridge, Massachusetts, 1988. The MIT Press.

[12] Guido Governatori, Michael Maher, Grigoris Antoniou, and David Billington. Argumentation semantics for defeasible logic. *Journal of Logic and Computation*, 14:675–702, 2004.

[13] Frederick Maier and Donald Nute. Relating defeasible logic to the well-founded semantics for normal logic programs. *Proceedings of the 11th Worshop on Nonmonotonic Reasoning*, pages 293–301, 2006.

[14] Guillermo R. Simari and Ronald P. Loui. A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence*, 53(2–3):125–157, 1992.