Sceptical logic programming based default reasoning – Defeasible logic rehabilitated

G. Antoniou, D. Billington and M.J. Maher
CIT, Griffith University
Nathan, QLD 4111, Australia
db,ga,mjm@cit.gu.edu.au

Abstract

There is a nonmonotonic logic which: (i) is suitable for default reasoning; (ii) came right from the beginning with an implementation; and (iii) is more general than other approaches which have been proposed recently. In fact this logic has been known for over 10 years now: it is Nute's defeasible logic. Despite these obvious advantages the logic is not in the mainstream of nonmonotonic reasoning, mainly because it has not been studied sufficiently as a formal system, and because its relationship to other formalisms has not been studied up to now.

This paper seeks to change the perception of defeasible logic. In particular, it presents the logic in a formal setting, derives several properties, including representational results, and gives an operational semantics. Finally the paper discusses some relationships of defeasible logic to other default reasoning approaches.

1 Introduction

Defeasible Logic [10, 11] is an approach to sceptical nonmonotonic reasoning [9, 1] that has a very distinctive feature: It was designed to be easily implementable right from the beginning, unlike most other approaches. In fact it has an implementation as a straightforward extension of Prolog [5]. Also, the logic has been found to be appropriate for the representation of many problems known in the default reasoning community.

Despite these obvious advantages, for a long time research in this particular approach has been mostly neglected. Reasons for this include: the fact that the logic has not been studied sufficiently as a formal system; that up to now only a proof theory but no semantics has been provided; that the proof theory has been difficult to use; and that relationships to other default reasoning approaches have not been established.

This paper seeks to move defeasible logic into the mainstream of nonmonotonic reasoning by alleviating these problems. In particular, it establishes several formal properties of the logic; it provides a semantic characterisation of the formalism; it develops normal forms, and studies possible semantics preserving simplifications of the proof theory; and it establishes relationships to some other nonmonotonic reasoning formalisms.

Section 2 gives a formal definition of defeasible logic. There are five kinds of features in Defeasible Logic: facts, strict rules, defeasible rules, defeaters, and a superiority relation among rules. Essentially the superiority relation provides information about the relative strength of rules, that is, it provides information about which rules can overrule which other rules.

Section 3 looks at defeasible logic as a formal system, and describes some of its properties. One of the results is surprising: the superiority relation does not add to the expressive power of the formalism. To be more precise, for every defeasible theory T with a superiority relation we can construct a defeasible theory T' with an empty superiority relation such that T and T' support the same conclusions in the logical language of T.

Section 4 discusses some aspects of defeasible logic as a knowledge representation language, and illustrates some of them using well-known examples. Section 5 gives a semantic characterization of a subset of defeasible logic. We use the semantic framework presented in [14, 15] which has been applied to a variety of logics for nonmonotonic and uncertainty reasoning [14]. We show how we need to modify the framework to suit our purposes.

Section 6 compares defeasible logic to some other known default reasoning approaches. Firstly we restate an older result [2] that nonmonotonic inheritance networks [8] are a special case of defeasible logic. Also, we show that Grosof's courteous logic programs [7] are a special case of defeasible logic. Finally we compare defeasible logic with LPwNF, an approach to logic programming without negation as failure, presented by Dimopoulos and Kakas [6]. We discuss how the two approaches differ. The main difference is that LPwNF does not take into account *teams of rules* supporting a conclusion (in Grosof's terminology), but rather views rules separately. By doing so, LPwNF fails to draw desirable conclusions that defeasible logic can, as we show in this paper.

2 Basics of Defeasible Logic

2.1 Informal Presentation

We begin by presenting the basic ingredients of Defeasible Logic. A defeasible theory[1] consists of five different kinds of knowledge: facts, strict rules, defeasible rules, defeaters, and a superiority relation.

Strict rules are rules in the classical sense: whenever the premises of a rule are given, we are allowed to apply the rule and get a conclusion. When the premises

[1]a knowledge base in Defeasible Logic

are indisputable (e.g. facts) then so is the conclusion. An example of a strict rule is "Emus are birds". Written formally:

$$emu(X) \rightarrow bird(X).$$

Defeasible rules are rules that can be defeated by contrary evidence. An example of such a rule is "Birds typically fly"; written formally:

$$bird(X) \Rightarrow flies(X).$$

The idea is that if we know that something is a bird, then we may conclude that it flies, *unless there is other evidence suggesting that it may not fly.*

Defeaters are rules that cannot be used to draw any conclusions. Their only use is to prevent some conclusions. In other words, they are used to defeat some defeasible rules by producing evidence to the contrary. An example is "If an animal is heavy then it may not be able to fly". Formally:

$$heavy(X) \leadsto \neg flies(X)$$

The main point is that the information that an animal is heavy is not sufficient evidence to conclude that it doesn't fly. It is only evidence that the animal *may* not be able to fly. In other words, we don't wish to conclude $\neg flies(X)$ in this case.

The *superiority relation* among rules is used to define priorities among rules, that is, where one rule may override the conclusion of another rule. For example, given the defeasible rules

$$\begin{array}{lrl} r: & bird(X) & \Rightarrow flies(X) \\ r': & brokenWing(X) & \Rightarrow \neg flies(X) \end{array}$$

which contradict one another, no conclusive decision can be made about whether a bird with broken wings can fly. But if we introduce a superiority relation $>$ with $r' > r$, then we can indeed conclude that it can't fly.

It turns out that we only need to define the superiority relation over rules with contradictory conclusions. Also notice that a cycle in the superiority relation is coutnerintuitive from the knowledge representation perspective. In the above example, it makes no sense to have both $r > r'$ and $r' > r$. Consequently, the defeasible logic we discuss requires an acyclic superiority relation.

2.2 Technical Details

In this paper we restrict attention to propositional defeasible logic, and assume that the reader is familiar with the notation and basic notions of propositional logic. If q is a literal, $\sim q$ denotes the complementary literal (if q is a positive literal p then $\sim q$ is $\neg p$; and if q is $\neg p$, then $\sim q$ is p).

A *rule* r consists of its *antecedent* $A(r)$ (written on the left; $A(r)$ may be omitted if it is the empty set) which is a finite set of literals, an arrow, and its *consequent* $C(r)$ which is a literal. There are three kinds of rules: *Strict rules* are denoted by $A \rightarrow p$, *defeasible rules* by $A \Rightarrow p$, and defeaters by $A \leadsto p$. Given a set R of rules, we denote the set of all strict rules in R by R_s, the set of strict and defeasible rules in R by R_{sd}, the set of defeasible rules in R by R_d, and the set of defeaters in R by R_{dft}. $R[q]$ denotes the set of rules in R with consequent q.

A *superiority relation on* R is an acyclic relation $>$ on R (that is, the transitive closure of $>$ is irreflexive). When $r_1 > r_2$, then r_1 is called *superior* to r_2, and r_2 *inferior* to r_1. A rule is called *inferior* iff it is inferior to another rule.

A *defeasible theory* T is a triple $(F, R, >)$ where F is a finite set of literals (called *facts*), R a finite set of rules, and $>$ a superiority relation on R. A *conclusion* of T is a tagged literal and can have one of the following four forms:

- $+\Delta q$, which is intended to mean that q is strictly provable in T.
- $-\Delta q$, which is intended to mean that we have proved that q is not strictly provable in T.
- $+\partial q$, which is intended to mean that q is defeasibly provable in T.
- $-\partial q$ which is intended to mean that we have proved that q is not defeasibly provable in T.

Provability is defined below. It is based on the concept of a *derivation* (or proof) in $T = (F, R, >)$. A derivation is a finite sequence $P = (P(1), \ldots P(n))$ of tagged literals satisfying the following conditions ($P(1..i)$ denotes the initial part of the sequence P of length i):

$+\Delta$: If $P(i+1) = +\Delta q$ then either
  $q \in F$ or
  $\exists r \in R_s[q]\ \forall a \in A(r) : +\Delta a \in P(1..i)$

$-\Delta$: If $P(i+1) = -\Delta q$ then
  $q \notin F$ and
  $\forall r \in R_s[q]\ \exists a \in A(r) : -\Delta a \in P(1..i)$

$+\partial$: If $P(i+1) = +\partial q$ then either
  (1) $+\Delta q \in P(1..i)$ or
  (2) (2.1) $\exists r \in R_{sd}[q] \forall a \in A(r) : +\partial a \in P(1..i)$ and
    (2.2) $-\Delta \sim q \in P(1..i)$ and
    (2.3) $\forall s \in R[\sim q]$ either
      (2.3.1) $\exists a \in A(s) : -\partial a \in P(1..i)$ or
      (2.3.2) $\exists t \in R_{sd}[q]$ such that

$\forall a \in A(t) : +\partial a \in P(1..i)$ and
$t > s$

$-\partial$: If $P(i+1) = -\partial q$ then
(1) $-\Delta q \in P(1..i)$ and
(2) (2.1) $\forall r \in R_{sd}[q]\ \exists a \in A(r) : -\partial a \in P(1..i)$ or
(2.2) $+\Delta \sim q \in P(1..i)$ or
(2.3) $\exists s \in R[\sim q]$ such that
(2.3.1) $\forall a \in A(s) : +\partial a \in P(1..i)$ and
(2.3.2) $\forall t \in R_{sd}[q]$ either
$\exists a \in A(t) : -\partial a \in P(1..i)$ or
not $t > s$

The elements of a derivation are called *lines* of the derivation. We say that a tagged literal L is *provable* in $T = (F, R, >)$, denoted $T \vdash L$, iff there is a derivation in T such that L is a line of P.

3 Properties of Defeasible Logic

3.1 A Normal Form for Defeasible Logic

We propose a normal form for defeasible theories. The main purpose of this normal form is to provide a separation of concerns, within a defeasible theory, between definite and defeasible conclusions. In defeasible logic, a strict rule may participate in the superiority relation. This participation has no effect on the definite conclusions of the theory, but can affect the defeasible conclusions. We consider theories where this occurs to be somewhat misleading, and propose a normal form in which definite and defeasible reasoning are separated as much as is practicable.

Definition 3.1 *We call a defeasible theory $T = (F, R, >)$* normalized *iff the following three conditions are satisfied:*

(a) Every literal is defined either solely by strict rules, or by a single strict rule and other non-strict rules.

(b) No strict rule participates in the superiority relation $>$.

(c) $F = \emptyset$

Every defeasible theory can be transformed into an equivalent normalised form. We now define this transformation explicitly.

Definition 3.2 *Consider a defeasible theory $T = (F, R, >)$. We define $normal(T) = (\emptyset, R', >)$, where R' is defined below.*

Let $'$ be a function which maps propositions to new (previously unused) propositions, and rule names to new rule names. We extend this, in the obvious way, to literals and conjunctions of literals.

$R' = R_d \cup R_{dft} \cup$
$\{\rightarrow f' \mid f \in F\} \cup$
$\{r' : A' \rightarrow C' \mid r : A \rightarrow C \textit{ is a strict rule in } R\} \cup$
$\{r : A \Rightarrow C \mid r : A \rightarrow C \textit{ is a strict rule in } R\} \cup$
$\{p' \rightarrow p \mid p \textit{ is a literal in the signature } \Sigma \textit{ of } T\}$

The rules derived from F and rules $p' \rightarrow p$ are given distinct new names.

It is clear from the transformation described above that $normal(T)$ is normalized (i.e. satisfies conditions (a)–(c)). Notice that strict rules have been altered to become defeasible rules, although their names are unchanged. Thus although $>$ is unchanged, it now no longer concerns any strict rule.

Theorem 3.1 *For every defeasible theory T in the signature Σ we can effectively construct a normalized defeasible theory T', such that T and T' have the same conclusions in Σ.*

3.2 Simulating the Superiority Relation

In this section we show that the superiority relation does not contribute anything to the expressive power of defeasible logic. Of course it does allow one to represent information in a more natural way.

We define below a transformation $trans$ that eliminates all uses of the superiority relation. For every rule r, it introduces a positive literal denoted by $inf(r)$. Intuitively, $inf(r)$ expresses that r is overruled by a superior rule.

Definition 3.3 *Let $T = (F, R, >)$ be a defeasible theory. Define $trans(T) = (F, R', \emptyset)$, where*

$$\begin{aligned}
&R' = \\
&\{A(r_1) \cup \{\neg inf(r_1)\} \Rightarrow inf(r_2) \mid r_1 > r_2\} \cup \\
&\{A(r) \rightarrow \neg inf(r) \mid r \in R\} \cup \\
&\{\neg inf(r) \rightarrow C(r) \mid r \textit{ is a strict rule }\} \cup \\
&\{\neg inf(r) \Rightarrow C(r) \mid r \textit{ is a defeasible rule }\} \cup \\
&\{\neg inf(r) \rightsquigarrow C(r) \mid r \textit{ is a defeater }\}.
\end{aligned}$$

The following result shows that if we restrict attention to conclusions in the old signature Σ (that is, if we disregard the new inf-symbols), then T and $trans(T)$ are equivalent.

Theorem 3.2 *Let $T = (F, R, >)$ be a defeasible theory in the signature Σ. Suppose no strict rule is inferior. Then T and $trans(T)$ allow the same conclusions in the signature Σ.*

3.3 A Minimal Set of Ingredients

As a result of the discussion so far, a defeasible theory can be viewed as a finite set of rules. There are three kinds of rules: strict, defeasible, and defeaters. In this section we show that it is impossible to simulate any of these types by the other two in a modular way. First we show this for strict rules.

Proposition 3.1 *Let $A \rightarrow p$ be a strict rule. Then, in general, there is no defeasible theory R' without strict rules, such that for all defeasible theories R, $R \cup \{A \rightarrow p\}$ and $R \cup R'$ allow the same conclusions in the signature Σ (where Σ is the signature of $R \cup \{A \rightarrow p\}$).*

Proposition 3.2 *Let $A \Rightarrow p$ be a defeasible rule. Then, in general, there is no defeasible theory R' without defeasible rules, such that for all defeasible theories R, $R \cup \{A \Rightarrow p\}$ and $R \cup R'$ allow the same conclusions in the signature Σ (where Σ is the signature of $R \cup \{A \Rightarrow p\}$).*

The above two results are quite trivial. The following theorem is more difficult to prove, and states that we cannot simulate defeaters in a modular way using strict and defeasible rules. The proof can be found in [4].

Theorem 3.3 *Let $A \leadsto p$ be a defeater. Then, in general, there is no defeasible theory R' without defeaters, such that for all defeasible theories R, $R \cup \{A \leadsto p\}$ and $R \cup R'$ allow the same conclusions in the signature Σ (where Σ is the signature of $R \cup \{A \leadsto p\}$).*

3.4 Properties of Defeasible Inference

Theorem 3.4 (Consistency) *Let T be a defeasible theory. If there is no q such that $T \vdash +\Delta q$ and $T \vdash +\Delta \neg q$, then there is no p such that $T \vdash +\partial p$ and $T \vdash +\partial \neg p$.*

Thus, if the strict part of T is consistent, then the defeasible part of T is consistent, too. Obviously this is the best one can hope for, since strict conclusions are always defeasible conclusions.

Theorem 3.5 *Suppose that P_1 and P_2 are derivations in a defeasible theory T. Then $P_1 \circ P_2$ (where $\circ$ denotes concatenation) is also a derivation in T.*

Theorem 3.6 (Cumulativity) *If $T \vdash +\partial p$ then, for all q, $T \vdash +\partial q$ iff $T \cup \{\Rightarrow p\} \vdash +\partial q$.*

Theorem 3.7 (Well-behaviour) *Let T be a defeasible theory. Then there is no p such that $T \vdash +\partial p$ and $T \vdash -\partial p$.*

The proofs of the above theorems are in [3]. There are cases where we can derive neither $+\partial p$ nor $-\partial p$. For example, if T consists only of the rules $p \Rightarrow q$ and $q \Rightarrow p$. But if we rule out this case, then defeasible logic is decisive in the following sense:

Theorem 3.8 (Decisiveness) *Let T be a defeasible theory such that its atom dependency graph is acyclic. Then, for every p, either $T \vdash +\partial p$ or $T \vdash -\partial p$.*

4 Defeasible Logic as a Knowledge Representation Language

4.1 Informal Discussion

Defeasible logic is a simple knowledge representation formalism. It avoids many features which make other default reasoning approaches computationally and conceptually complex.

- It is *sceptical*, thus avoiding the computation of alternative extensions.
- It is *logic programming based*, thus disregarding contrapositives and having a more efficient deductive basis.
- On the other hand, *negation as failure is not used*, further reducing the computational and conceptual load.
- Instead defeasible logic uses the concept of a superiority relation, which is an attractive tool for coping with conflicting information. Priorities occur naturally in many application domains, and can be based on specificity, authority, freshness, reliability etc.
- *Conflicts are local*, in that they arise only between rules with conflicting heads. This is in contrast to other default reasoning approaches, for example default logic, where a conflict may involve several defaults (which is computationally expensive).

On the other hand, as we will show in the next section, defeasible logic can represent and treat adequately several well-known examples, despite its conceptual simplicity. More examples can be found in [12, 5].

4.2 Examples

Example 4.1 (The Nixon diamond)

$quaker(nixon)$

$republican(nixon)$

$r_1: \ quaker(X) \Rightarrow \neg militarist(X)$[2]

$r_2: \ republican(X) \Rightarrow militarist(X)$

If the superiority relation is empty, then we can prove both $-\partial militarist(nixon)$ and $-\partial\neg militarist(nixon)$. But if we add $r_2 > r_1$, then we can conclude $+\partial militarist(nixon)$.

Example 4.2 (The birds example)

$bird(tweety)$

$penguin(opus)$

$r_1: \ penguin(X) \rightarrow bird(X)$

$r_2: \ bird(X) \Rightarrow flies(X)$

$r_3: \ penguin(X) \Rightarrow \neg flies(X)$

$r_3 > r_2$

Here we can prove $+\partial flies(tweety)$, $+\partial\neg flies(opus)$ and $-\partial flies(opus)$.

Example 4.3 (Hierarchies with exceptions)

$nautilus(nancy)$

$r_1: \ nautilus(X) \rightarrow cephalopod(X)$

$r_2: \ cephalopod(X) \rightarrow mollusc(X)$

$r_3: \ nautilus(X) \Rightarrow shell(X)$

$r_4: \ cephalopod(X) \Rightarrow \neg shell(X)$

$r_5: \ mollusc(X) \Rightarrow shell(X)$

$r_3 > r_4$

$r_4 > r_5$

Here we can derive $+\partial shell(nancy)$ and $-\partial\neg shell(nancy)$.

Example 4.4 (Yale Shooting Problem)

[2] Rules with free variables are interpreted as rule schemas, that is, as the set of all ground instances.

$alive(start)$

$\neg loaded(start)$

$loaded(load(X))$

$r_1: \ loaded(X) \Rightarrow \neg alive(shoot(X))$

$r_2: \ alive(X) \Rightarrow alive(action(X))$[3]

$r_3: \ loaded(X) \Rightarrow loaded(action(X))$

$r_1 >$ any instance of r_2

As desired, we can prove $+\partial \neg alive(shoot(wait(load(start))))$.

Example 4.5 (No propagation of ambiguity) We borrow this example from [7].

$quaker(nixon)$

$republican(nixon)$

$r_1: \ quaker(X) \Rightarrow pacifist(X)$

$r_2: \ republican(X) \Rightarrow \neg pacifist(X)$

$r_3: \ republican(X) \Rightarrow footballfan(X)$

$r_4: \ pacifist(X) \Rightarrow antimilitary(X)$

$r_4: \ footballfan(X) \Rightarrow \neg antimilitary(X)$

The superiority relation is empty.

Here we can prove $+\partial \neg antimilitary(nixon)$. This occurs by proving $+\partial footballfan(nixon)$ and $-\partial pacifist(nixon)$. The latter makes sure that R_4 cannot be applied, thus there is no conflict with rule R_5, which therefore fires and gives the desired result.

This behaviour should be contrasted with approaches based on extensions, e.g. default logic. There we would get three extensions (if we represent each defeasible rule by a corresponding normal default):

One including $pacifist(nixon), antimilitary(nixon)$

One including $pacifist(nixon), footballfan(nixon), \neg antimilitary(nixon)$

One including $\neg pacifist(nixon), \neg antimilitary(nixon)$

Note that $\neg antimilitary(nixon)$ is *not* included in the intersection of these extensions. The difference between these two kinds of sceptical reasoning has already been pointed out in [16].

[3] where *action* is any action in the language; in this case it can be *load*, *wait* or *shoot*. Both r_2 and r_3 are rule schemas expressing the frame axioms.

5 Operational Semantics

5.1 Informal Discussion

In this section we will give a semantic characterization of defeasible logic. Essentially we will adopt the semantic framework of [15] which has been used to characterize several nonmonotonic and uncertainty reasoning methods. Before discussing how that framework needs to be adapted to be suitable for defeasible logic, let us first outline the basic idea using a simple example from default logic [13]. Consider the default theory T consisting of the fact p and the defaults $\delta_1 = \frac{p:q}{q}$ and $\delta_2 = \frac{q:r}{r}$. Obviously T has the single extension $Th(\{p, q, r\})$. It is obtained from the process (δ_1, δ_2) (see [1]).

Let W be the set of all possible (total) worlds over the propositional language $\{p, q, r\}$. The semantic counterpart of the process above is the following so-called *default partition sequence*:

$S =< W_0, W_1, W_2, W_3 >$, where
$W_0 = \{w \mid w \models \neg p\}$
$W_1 = \{w \mid w \models (p \wedge \neg q)\}$
$W_2 = \{w \mid w \models (p \wedge q \wedge \neg r)\}$
$W_3 = \{w \mid w \models (p \wedge q \wedge r)\}$.

First note that S forms a partition of W. W_0 includes all worlds in which the fact p is false. W_1 includes those world not in W_0 in which the consequent of the first default is false. Equally W_2 contains those worlds not in $W_0 \cup W_1$ in which the consequent of the second default is not true. Finally W_3 consists of the remaining worlds, in which the fact and the consequents of the defaults applied are true. The set of all formulae true in W_3 is an extension of T, indeed its only extension.

Please note also that the prerequisite p of the first default applied is true in all worlds in $W_1 \cup W_2 \cup W_3$, and that there is at least one world in W_3 in which the justification of δ_1 is true. This is an instance of the general definition of a default partition sequence, which can be found in [15].

Essentially $< W_i, \ldots, W_l >$ defines a "frame of reference", that is, the body of knowledge which builds the current context before applying the i-th rule. Each reasoning step is performed with respect to the current context. Once an inference is made, the frame of reference is updated by pruning out additional worlds, those in which the new conclusion is false.

There are several fundamental problems with trying to apply this semantic framework to defeasible logic. Firstly the framework considers rules stand alone. But in defeasible logic, only the consideration of all rules for p and $\neg p$ can lead to the decision to conclude p or not (see proof theory in section 2.2). This "team consideration" must be reflected in the semantic characterization. Secondly, in defeasible logic we may make both positive and negative conclusions: $+\partial$ and $-\partial$.

Particularly with respect to negative conclusions, it is unclear which rule should be used, as required in the semantic framework outlined above.

Our solution to this problem is to switch attention from a sequence of rules (e.g. a process in default logic) to *a sequence of conclusions* P which, as explained in section 2.2, are tagged literals. This still leaves us with a problem. If we follow the idea of the semantics for default logic, if the i-th line of P is

$$+\partial p$$

then we should collect in W_i all worlds w not in $W_0 \cup \ldots \cup W_{i-1}$ in which p is false. But what if the i-th line of P is instead

$$-\partial \neg p?$$

Then we would be tempted to use the same W_i and it would be natural. But in defeasible logic $+\partial p$ and $-\partial\neg p$ are different: The former states that we can prove p, while the second says that we cannot prove $\neg p$. Obviously we need to distinguish between these two cases. We solve this problem by considering *partial worlds*, where it is possible that neither p nor $\neg p$ is true.

Finally we note that defeasible logic allows conclusions of different rigour: strict conclusions ($+\Delta$ and $-\Delta$) and defeasible conclusions ($+\partial$ and $-\partial$). To model this we would need two different kinds of propositions in our worlds. This can be easily done, but for the sake of technical simplicity, in the next section we consider only defeasible theories where the strict part is empty – in that case we just disregard all strict conclusions $-\Delta p$ for all literals p.

5.2 Formal Definition

Let Σ be a propositional signature, that means, a set of propositional atoms. We call w a *partial world* (in Σ) iff $w \subseteq \Sigma \cup \neg\Sigma$, and there is no $p \in \Sigma$ such that $\{p, \neg p\} \subseteq w$. For literals q, define $w \models q$ (q is true in w) iff $q \in w$. Let W be the set of all partial worlds (in Σ).

A *partition sequence* of W is a tuple $< W_1, \ldots, W_l >$ ($l \geq 1$) such that the non-empty elements W_i form a partition of W.

Definition 5.1 *Let $T = (R, >)$ be a defeasible theory without facts and strict rules. A* DL partition sequence *for T is a partition sequence $S =< W_1, \ldots, W_l >$ such that there is a sequence of tagged literals $P =< L_1, \ldots, L_{l-1} >$ satisfying the following conditions:*

(a) For all $i = 1, \ldots, l-1$, if $P(i) = +\partial q$ then $W_i = \{w \mid w \notin W_1 \cup \ldots \cup W_{i-1}$ and $w \not\models q\}$; and if $P(i) = -\partial q$, then $W_i = \{w \mid w \notin W_1 \cup \ldots \cup W_{i-1}$ and $w \models q\}$.

(b) For all $i = 1, \ldots, l-1$:

Case 1: *If* $P(i) = +\partial q$ *then*

(i) $\exists r \in R_d[q]$ *such that* $\forall a \in A(r) \forall w \in W_i \cup \ldots \cup W_l : w \models a$ *and*

(ii) $\forall s \in R[\sim q]$ *either*

(1) $\exists a \in A(s) \forall w \in W_i \cup \ldots \cup W_l : w \not\models a$ *or*

(2) $\exists t \in R_d[q]$ *such that*

$\forall a \in A(t) \forall w \in W_i \cup \ldots \cup W_l : w \models a$ *and*

$t > s$

Case 2: *If* $P(i) = -\partial q$ *then either*

(i) $\forall r \in R_d[q]$ $\exists a \in A(r) \forall w \in W_i \cup \ldots \cup W_l : w \not\models a$ *or*

(ii) $\exists s \in R[\sim q]$ *such that*

(1) $\forall a \in A(s) \forall w \in W_i \cup \ldots \cup W_l : w \models a$ *and*

(2) $\forall t \in R_d[q]$ *either*

$\exists a \in A(t) \forall w \in W_i \cup \ldots \cup W_l : w \not\models a$ *or*

not $t > s$

Theorem 5.1 *Let* T *be a defeasible theory without facts and strict rules. Then*

(a) $T \vdash +\partial L$ *iff there exists a DL partition sequence* $< W_1, \ldots, W_l >$ *for* T *such that for all* $w \in W_l$, $w \models L$.

(b) $T \vdash -\partial L$ *iff there exists a DL partition sequence* $< W_1, \ldots, W_l >$ *for* T *such that for all* $w \in W_l$, $w \not\models L$.

The proof is fairly simple: In one direction one shows that the sequence of tagged literals corresponding to S defines a derivation in T (one needs just add appropriate tagged literals $-\Delta q$ at the beginning). In the opposite direction, given a derivation P in T, and after removing any $-\Delta q$ tagged literals, the corresponding partition sequence defined according to condition (b) in the definition 5.1 satisfies also condition (a), that is, it is a DL partition sequence. Both proofs are based on easy induction on the length of the sequences involved.

6 Relationship to other approaches

6.1 Inheritance Networks

Nonmonotonic inheritance networks [8] were an early nonmonotonic reasoning approach which had powerful implementations, even though they lacked declarativity. In [2] it was shown that they can be viewed as a special case of defeasible logic.

A nonmonotonic inheritance network consists of a set of objects, a set of properties, and a set of arcs. Below is a list of the possible kinds of arcs (a is an object, p and q are properties):

$a \Rightarrow p$, meaning that a has the property p.

$a \not\Rightarrow p$, meaning that a doesn't have property p.

$p \Rightarrow q$, meaning that an object with property p typically has property q.

$p \not\Rightarrow q$, meaning that an object with property p typically doesn't have property q.

A nonmonotonic inheritance network N is naturally translated into a defeasible theory $T(N)$:

For every arc $a \Rightarrow p$ in N include the fact $p(a)$ in $T(N)$.

For every $a \not\Rightarrow p$ in N include the fact $\neg p(a)$ in $T(N)$.

For every path $a \Rightarrow \ldots \Rightarrow p \Rightarrow q$ in N include the rule $p(a) \Rightarrow q(a)$ in $T(N)$.

For every path $a \Rightarrow \ldots \Rightarrow p \not\Rightarrow q$ in N include the rule $p(a) \Rightarrow \neg q(a)$ in $T(N)$.

Missing is the definition of the superiority relation which behaves the same as specificity in inheritance networks. The complicated definition is found in [2]. That paper also proposes a way of compiling specificity into the definition of a derivation, which can be used to make the translation of a nonmonotonic inheritance network into a defeasible theory modular.

Theorem 6.1 *Let N be a nonmonotonic inheritance network. Then we may construct a defeasible theory $T(N)$, such that for every literal q, q is supported by N iff $T(N) \vdash +\partial q$.*

6.2 Courteous Logic Programs

Courteous logic programs were introduced rather recently in [7] as a simple and efficient form of default reasoning, to be used as the reasoning basis for intelligent agents. The approach shares some basic ideas of defeasible logic. In particular, it is logic programming based, implements sceptical reasoning, and is based on competing teams of rules, and a priority relation.

So, for a propositional atom p, there is a team consisting of rules for p which can be applied, and a competing team consisting of rules for $\neg p$ which can be applied. The basic rule is that a team wins iff every member of the opposing team is refuted. Refutation is based on prioritization, and is carried out as in defeasible logic. It is possible that the teams defeat one another, in which case neither p nor $\neg p$ is deduced. All conclusions are collected in an *answer set*. Every courteous logic program has exactly one answer set, since it is a sceptical reasoning approach.

Compared to defeasible logic, courteous logic programs are more special in the following respects:

- The atom dependency graph of a courteous logic program must be acyclic (thus recursion is not possible). This condition is central in the courteous logic programs framework, but is not necessary in defeasible logic.
- Defeasible logic distinguishes between strict and defeasible conclusions, courteous logic programs do not. Thus defeasible logic is more fine-grained.
- Defeasible logic has the concept of a defeater, courteous logic programs do not. Thus defeasible logic offers a greater flexibility in the expression of information.

On the other hand, there seems to be a major difference between the two approaches, in that courteous logic programs may use negation as failure, while the concept is not used in defeasible logic. But it can be easily shown that a courteous logic program with negation as failure can be modularly translated into one without negation as failure. Every rule

$$L \leftarrow L1 \wedge \ldots \wedge L_n \wedge \ fail M_1 \wedge \ldots \wedge \ fail M_k$$

can be replaced by the following rules:

$L \leftarrow L_1 \wedge \ldots \wedge L_n \wedge p_l$

$p_l \leftarrow$

$\neg p_l \leftarrow M_1$

$\ldots$

$\neg p_l \leftarrow M_k$

where p_l is a new propositional atom. It can be easily shown that the same literals in the original language are included in the answer set of the original courteous logic program C and its transformation $trans(C)$ which doesn't use negation as failure.

Let C be a courteous logic program without negation as failure. Define the defeasible theory $df(C)$ as follows: For every labeled rule

$$< l > \ L_0 \leftarrow L_1 \wedge \ldots \wedge L_n$$

in C include in $df(C)$ the defeasible rule

$$df(l): \ L_1, \ldots, L_n \Rightarrow L_0.$$

And for every fact $R(i,j)$ in the prioritization sub-program of C (see [7] for details) include in the extension of the superiority relation of $df(C)$ the fact $df(i) > df(j)$.

Theorem 6.2 *Let C be a courteous logic program. A literal q is in the answer set of C iff $df(C) \vdash +\partial q$.*

Thus courteous logic programs are a special case of defeasible logic. But on the other hand, defeasible logic is not more computationally expensive for the cases covered by courteous logic programs.

6.3 Logic Programming without Negation as Failure (LPwNF)

LPwNF was introduced in [6] as an alternative way of default reasoning, one in which negation as failure is simulated by the use of a priority relation $>$. The paper introduced a proof theory and a corresponding argumentation framework.

The main idea of LPwNF is the following: In order to prove a literal q, a *type A derivation* must be found which proves q. One part of this derivation is a top-level proof of q in the sense of logic programming (SLD-resolution). But additionally every attack on this argument must be counterattacked. Attacks are generated in *type B derivations*.

In general, a rule r in a type B derivation can attack a rule r' in a type A derivation if they have complementary heads, and r is not weaker than r', that is, not $r' > r$. On the other hand, a rule r in a type A derivation can attack a rule r' in a type B derivation if they have complementary heads, and $r > r'$. This reflects the notion of scepticality: it should be easier to attack a positive argument than to counterattack (i.e. attack the attacker).

The whole intuition strongly resembles the idea of defeasible logic. In both we need to counter each rule s against q by a rule t for q such that $t > s$ (and t can be applied, of course). So superficially one would expect both approaches to be closely related. But a careful analysis reveals a major difference between the logics: *LPwNF argues on the basis of single rules, while defeasible logic argues on the basis of teams with the same head.* The difference can be illustrated by the following simple example. Consider the defeasible theory

$$layEggs(platypus)$$
$$hasFur(platypus)$$
$$monotreme(platypus)$$
$$hasBill(platypus)$$
$$r_1: \ monotreme(X) \Rightarrow mammal(X)$$
$$r_2: \ hasFur(X) \Rightarrow mammal(X)$$
$$r_3: \ laysEggs(X) \Rightarrow \neg mammal(X)$$
$$r_4: \ hasBill(X) \Rightarrow \neg mammal(X)$$
$$r_1 > r_3$$
$$r_2 > r_4$$

Intuitively we conclude that $platypus$ is $mammal$. The reason is that for every reason against this conclusion (r_3 and r_4) there is a stronger reason for $mammal(platypus)$ (r_1 and r_2 respectively). It is easy to see that $+\partial mammal(platypus)$ is indeed provable in defeasible logic: there is a rule in support of $mammal(platypus)$, and every rule for $\neg mammal(platypus)$ is overridden by a rule for $mammal(platypus)$.

But the corresponding logic program without negation as failure is unable to prove $mammal(platypus)$: If we start with r_1 trying to build an A derivation, then we must counter the attack r_4 (which is not inferior to r_1) used in a B derivation. But LPwNF does not allow counterattacks on r_4 by another rule with head $mammal(platypus)$, but only by an attack on the body of r_4. The latter is impossible in our case (there is no rule with head $\neg hasBill(platypus)$). Thus the attack via r_4 succeeds and the proof of $mammal(platypus)$ via r_1 fails. Equally the proof of $mammal(platypus)$ via r_2 fails, due to an attack via rule r_3. Thus $mammal(platypus)$ cannot be proven.

So LPwNF fails to derive conclusions that defeasible logic can, and which we regard as intuitive. But the opposite is true, as the next results states.

Theorem 6.3 *Let q be a literal which can be sceptically proven in the logic program without negation as failure P. Let $T(P)$ be the defeasible theory containing the same rules as P written as defeasible rules, and the same superiority relation. Then $T(P) \vdash +\partial q$.*

It should be clear, though, that LPwNF supports both sceptical and credulous reasoning, while defeasible logic supports only sceptical reasoning.

7 Conclusion

Defeasible logic is a sceptical logic-programming based default reasoning formalism which doesn't use negation as failure. Its basic idea is that teams for a potential conclusion and its negation compete. A team for a conclusion wins if it can be shown that every objection to the conclusion can be overruled. In this paper we studied defeasible logic as a formal system, and gave an operational semantics. Furthermore we indicated its suitability as a knowledge representation formalism by expressing several known examples in defeasible logic, and by comparing it to several other default reasoning formalisms.

Of course there remains a lot to be done. Firstly we intend to give other semantic characterisations of the logic. Our top priority is to give an argumentation theoretic semantics of defeasible logic. We are currently studying the argumentation framework of [6], but it is not suitable as it stands. Also we plan to establish formal results relating defeasible logic to LPwNF and to Priority Logic [17].

Another avenue of research is to provide a better, more efficient implementation of defeasible logic. We will be using some techniques presented in section 3. The main tools that we used there are modular transformations, that is, transformations that apply to each unit of information, independent of its context. Such transformations are valuable, since they can be the basis of the compilation of a defeasible theory into a more efficiently executable form. It is here that the modular nature of the transformations is important, since modifications to the original theory do not require the recompilation of the entire theory – only an incremental change need be made to the compiled theory.

References

[1] G. Antoniou. *Nonmonotonic Reasoning.* MIT Press 1997.

[2] D. Billington, K. de Coster and D. Nute. A modular translation from defeasible nets to defeasible logic. *Journal of Experimental and Theoretical Artificial Intelligence* 2 (1990): 151-177.

[3] D. Billington. Defeasible Logic is Stable. *Journal of Logic and Computation* 3 (1993): 370–400.

[4] D. Billington, G. Antoniou and M.J. Maher. *Representation results in defeasible logic.* Technical Report, CIT, Griffith University 1997.

[5] M.A. Covington, D. Nute and A. Vellino. *Prolog Programming in Depth.* Prentice Hall 1997.

[6] Y. Dimopoulos and A. Kakas. Logic Programming without Negation as Failure. In *Proc. ICLP-95*, MIT Press 1995.

[7] B.N. Grosof. *Practical Prioritized Defaults via Logic Programs.* IBM Research Report RC 20464, T.J. Watson Research Center 1996.

[8] J.F. Horty, R.H. Thomason and D. Touretzky. A skeptical theory of inheritance in nonmonotonic semantic networks. In *Proc. AAAI-87*, 358-363.

[9] V. Marek and M. Truszczynski. *Nonmonotonic Logic*, Springer 1993.

[10] D. Nute. Defeasible Reasoning. In *Proc. 20th Hawaii International Conference on Systems Science*, IEEE Press 1987, 470–477.

[11] D. Nute. Defeasible Logic. In D.M. Gabbay, C.J. Hogger and J.A. Robinson (eds.): *Handbook of Logic in Artificial Intelligence and Logic Programming Vol. 3*, Oxford University Press 1994, 353-395.

[12] D. Nute. A decidable quantified defeasible logic. In D. Prawitz, B. Skyrms and D. Westerstahl (Eds.): *Logic, Methodology and Philosophy of Science IX*, Elsevier 1994, 263 – 284.

[13] R. Reiter. A logic for default reasoning. *Artificial Intelligence* 13(1980): 81–132.

[14] C.M. Teng. Possible world partition sequences: A unifying framework for uncertain reasoning. In *Proc. 12th Conference on Uncertainty in Artificial Intelligence*, 1996, 517-524.

[15] C.M. Teng. *Many default logics: Instantiating generic non-monotonic partition sequences.* Technical Report 656, Department of Computer Science, University of Rochester 1997.

[16] D.D. Touretzky, J.F. Horty and R.H. Thomason. A clash of intuitions: The current state of nonmonotonic multiple inheritance systems. In *Proc. IJCAI-87*, 476-482, Morgan Kaufmann 1987.

[17] X. Wang, J. You and L. Ynan. *Logic programming without default negation revisited.* TR96-25, Department of Computing Science, University of Alberta 1996.