Defeasible Logic with Dynamic Priorities

Grigoris Antoniou
Univ. of Bremen, Germany
ga@tzi.de

Abstract.

Defeasible logic is a nonmonotonic reasoning approach based on rules and priorities. Its design supports efficient implementation, and it shows promise to be successfully deployed in applications.

So far only static priorities have been used, provided by an external superiority relation. In this paper we show how dynamic priorities can be integrated, where priority information is obtained from the deductive process itself. Dynamic priorities have been studied for other related reasoning systems, such as default logic and argumentation. We define a proof theory, study its formal properties, and provide an argumentation semantics.

1 Introduction

Defeasible reasoning is a nonmonotonic reasoning approach in which the gaps due to incomplete information are closed through the use of defeasible rules that are usually appropriate. Defeasible logics were introduced and developed by Nute over several years [15]. These logics perform defeasible reasoning, where a conclusion supported by a rule might be overturned by the effect of another rule. Roughly, a proposition p can be defeasibly proved ($+\partial p$) only when a rule supports it, and it has been demonstrated that no applicable rule supports $\neg p$; this demonstration makes use of statements $-\partial q$ which mean intuitively that an attempt to prove q defeasibly has failed finitely. These logics also have a monotonic reasoning component, and a priority on rules.

This family of approaches has recently attracted considerable interest. Its use in various application domains has been advocated, including the modelling of regulations and business rules [11], modelling of contracts [18], legal reasoning [16] and agent negotiations [9]. Also, defeasible reasoning (in the form of courteous logic programs [10]) provides the foundation for IBM's Business Rules Markup Language and for planned W3C activities. Therefore defeasible reasoning is arguably one of the most successful subareas in nonmonotonic reasoning as far as applications and integration to mainstream IT is concerned.

Recent theoretical work on defeasible logics has: (i) established some relationships to other logic programming approaches without negation as failure [1]; (ii) analysed the formal properties of these logics [3, 14], and (iii) has delivered efficient implementations [13].

So far defeasible logics had *fixed priorities*, given in the form of an external superiority relation. But *dynamic priorities*, where priorities are derived within the logic, have been considered in related areas, such as for extended logic programs [5, 6] and argumentation [17]. Reasons for considering dynamic priorities include the following:

- *Case-based priorities*: Sometimes preference of a rule over another is conditional on a property holding. In such cases it is natural to write $a \rightarrow r_1 > r_2$ and apply the priority information whenever a has been derived.
- *Defeasible priorities*: Sometimes case-based priorities do not apply always, but will be subject to "attacks" from other arguments. This happens, for example, in the legal domain, when different prioritisation principles can be applied (specificity, hierarchy, recency). In such cases the principles should be stated conditionally using defeasible rules, and may be defeated by stronger principles.
- We expect dynamic priorities to be particularly useful for intelligent agents working on the Web. Such agents will frequently need to make decisions about the reliability of information, especially to resolve conflicts among different sources.

In this paper we show how dynamic priorities can be incorporated in the framework of defeasible logics. We consider two variations: one in which cyclic priorities may emerge, and one in which such cycles are avoided. We study both approaches, and derive various theoretical results concerning their proof theory and semantics.

2 The Approach

2.1 A Language for Defeasible Reasoning

A defeasible theory (a knowledge base in defeasible logic) consists of three different kinds of knowledge: strict rules, defeasible rules, and a superiority relation. (Fuller versions of defeasible logic also have facts and defeaters, but [3] shows that they can be simulated by the other ingredients).

Strict rules are rules in the classical sense: whenever the premises 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, not inferior, evidence suggesting that it may not fly.*

In the defeasible logics defined so far in the literature, an external, acyclic *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 $r' > r$, with the intended meaning that r' is strictly stronger than r, then we can indeed conclude that the bird cannot fly.

2.2 Formal Definition

In this paper we restrict attention to essentially propositional defeasible logic. Rules with free variables are interpreted as rule schemas, that is, as the set of all ground instances; in such cases we assume that the Herbrand universe is finite. We 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).

Rules are defined over a *language* (or *signature*) Σ, the set of propositions (atoms) and labels that may be used in the rule.

A *rule* $r : A(r) \hookrightarrow C(r)$ consists of its unique *label* r, its *antecedent* $A(r)$ ($A(r)$ may be omitted if it is the empty set) which is a finite set of literals, an arrow $\hookrightarrow$ (which is a placeholder for concrete arrows to be introduced in a moment), and its *head* (or *consequent* or *conclusion*) $C(r)$ which is a literal. In writing rules often we omit set notation for antecedents. There are two kinds of rules, each represented by a different arrow. Strict rules use $\rightarrow$ and defeasible rules use $\Rightarrow$.

Given a set R of rules, we denote the set of all strict rules in R by R_s. $R[q]$ denotes the set of rules in R with consequent q.

A *defeasible theory* D is a finite set of rules R.

2.3 Defeasible Logic with Dynamic Priorities

The approach we take is as follows:

- For every rule r we associate a unique label $label(r)$.
- In the language language we introduce a binary predicate $\succ$ which applies to rule labels. The intuitive meaning of $label(r_1) \succ label(r_2)$ is that rule r_1 is stronger than rule r_2.
- Literals including $\succ$ may appear anywhere in a rule: as antecedents or as consequents, negated or not negated. For example, we can write:

  $a \Rightarrow label(r_1) \succ label(r_2)$
  $\neg label(r_1) \succ label(r_2) \Rightarrow c$

- A defeasible theory is just a set of rules. We don't use external priority information, but only the priorities derived from the theory.

We consider two variants of defeasible logic with dynamic priorities. The first one is simpler and allows cyclic priority information to be derived (thus resulting in the possibility of inconsistencies). The second approach checks for potential cycles in the priorities, and rules them out.

According to the first approach, it is the user's responsibility to ensure that cycles do not occur (just as it is the programmer's responsibility to avoid dividing by zero). The second approach prevents cycles from occurring, in anticipation of its use in distributed environments, where cycles can easily occur when putting together knowledge from various sources.

3 DDL1: Dynamic Defeasible Logic without Cycle Checking

3.1 Proof Theory

A *conclusion* of a defeasible theory D consists of a sign, a tag, and a literal. The $+$ sign denotes provability, and the $-$ sign denotes finite failure to prove. The tags we use are those introduced for the family of defeasible logics in [2]: Δ denotes strict provability, and ∂ denotes defeasible provability. More tags will be introduced soon. So, a conclusion $+\partial q$ is intended to mean that q is defeasibly provable in D. And $-\Delta$ means that q is provably not strictly provable.

Provability is defined below. It is based on the concept of a *derivation* (or *proof*) in $D = R$. A derivation is a finite sequence $P = P(1), \ldots, P(n)$ of tagged literals satisfying the following conditions. The conditions are essentially inference rules phrased as conditions on proofs.$P(1..i)$ denotes the initial part of the sequence P of length i.

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

That means, to prove $+\Delta q$ we need to establish a proof for q using strict rules only. This is a deduction in the classical sense – no proofs for the negation of q need to be considered (in contrast to defeasible provability below, where opposing chains of reasoning must be taken into account, too).

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

The definition of $-\Delta$ is the so-called *strong negation* of $+\Delta$: normal negation rules like De-Morgan rules are applied to the definition, $+$ is replaced by $-$, and vice versa. Therefore in the following we may omit giving inference conditions of both $+$ and $-$.

$+\partial$: If $P(i+1) = +\partial q$ then either
(1) $+\Delta q \in P(1..i)$ or
(2) (2.1) $\exists r \in R[q]\ \forall a \in A(r) : +\partial a \in P(1..i)$ 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) $\exists t \in R[q]$ such that
$\forall a \in A(t) + \partial a \in P(1..i)$ and
$+\partial(label(t) \succ label(s)) \in P(1..i)$

Intuitively: to show that q is provable defeasibly we have two choices: (1) We show that q is already definitely provable; or (2) we need to argue using the defeasible part of D as well. In particular, we require that there must be a strict or defeasible rule with head q which can be applied (2.1). But now we need to consider possible "counterattacks", that is, reasoning chains in support of $\sim q$. To be more specific: to prove q defeasibly we must show that $\sim q$ is not definitely provable (2.2). Also (2.3) we must consider the set of all rules which are not known to be inapplicable and which have head $\sim q$. Essentially each such rule s attacks the conclusion q. For q to be provable, each such rule s must have been established as non-applicable (2.3.1). Alternatively, there is an applicable rule with head q which is stronger than the rule s, according to the priority information derived so far.

We note that the only difference to the corresponding inference condition in the "classical" defeasible logic [3] is the last line: we use the priority derived so far, instead of an external superiority relation.

The inference condition $-\partial$ is defined in a similar way, as the strong negation of the $+\partial$ inference condition.

Also we note that the conclusion q may be a literal $label(r_1) \succ label(r_2)$.

Example 3.1 Consider the theory

$r_1 :\Rightarrow p$ $\quad$ $r_3 :\Rightarrow q$
$r_2 : p \Rightarrow r_3 \succ r_4$ $\quad$ $r_4 :\Rightarrow \neg q$

(in examples the notation $r : A \Rightarrow C$ implies that r is the unique label of the rule). We can derive $+\partial p, +\partial(r_3 \succ r_4)$, and $+\partial q$.

□

3.2 Ambiguity Propagation

In [2] we discussed the property of ambiguity propagation, noting that the original defeasible logic was ambiguity blocking. A preference for ambiguity blocking or ambiguity propagating behaviour is one of the properties of non-monotonic inheritance nets over which intuitions can clash [19]. Ambiguity propagation results in fewer conclusions being drawn, which might make it preferable when the cost of an incorrect conclusion is high. For these reasons an ambiguity propagating version of DL is of interest.

The solution to achieve ambiguity propagation behaviour is to separate the invalidation of a counterargument from the derivation of $-\partial$ tagged literals. We do so by introducing a third level of provability (besides definite and defeasible provability), called *support* and denoted by $\int$. Intuitively, a literal p is supported if there is a chain of reasoning that would lead us to conclude p in the absence of conflicts.

Thus in our modification of the $+\partial$ condition, a counterargument may be disregarded only if a literal in its body is known not to be supported ($-\int$), which is stronger than the previous condition that made it sufficient for a literal in the body not be defeasibly provable. First we modify the $+\partial$ condition (the $-\partial$ condition is modified accordingly following the Principle of Strong Negation).

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

Next we define the inference conditions for support. If we ignore the superiority relation we could define it simply as a simple chain. However, in situations where two conflicting rules can be applied and one rule is inferior to another, the inferior rule should not be counted as supporting its conclusion. Thus we refine the inference rule as follows ($-\int$ is defined accordingly):

$+\int$: If $P(i+1) = +\int q$ then either
$+\Delta q \in P(1..i)$ or
$\exists r \in R[q]$ such that
$\forall a \in A(r) : +\int a \in P(1..i)$, and
$\forall s \in R[\sim q]$ either
$\exists a \in A(s) : -\partial_{am} a \in P(1..i)$ or
$+\partial(label(r) \succ label(s)) \in P(1..i)$

3.3 Properties

First we make the following observation: if $\succ$ does not occur in a defeasible theory, then DDL1 behaves exactly as the classical defeasible logic.

Theorem 1 (Coherence) *It is impossible to derive both $+tag\ q$ and $-tag\ q$ from a defeasible theory D, for tag $\in \{\Delta, \partial, \partial_{ap}, \int\}$.*

Theorem 2 (Consistency) *Let the set $\{label(r_1) \succ label(r_2) \mid D \vdash +\partial(label(r_1) \succ label(r_2))\}$ not contain a cycle. Then $+\partial p$ and $+\partial \neg p$, or $+\partial_{ap} p$ and $+\partial_{ap} \neg p$, only if $+\Delta p$ and $+\Delta \neg p$.*

In other words, an inconsistency in the defeasible part can only occur if the same inconsistency already occurs in the strict part. This property does not hold if we do not assume the extra condition, as the following example shows.

Example 3.2 Consider the theory

$r_1 :\Rightarrow r_3 \succ r_4$ $\quad$ $r_3 :\Rightarrow p$
$r_2 :\Rightarrow r_4 \succ r_3$ $\quad$ $r_4 :\Rightarrow \neg p$

We have $-\Delta p$ and $-\Delta \neg p$, but $+\partial p$ and $+\partial \neg p$. It is instructive to see the difference between $r_4 \succ r_3$ and $\neg r_3 \succ r_4$. If we modify rule r_2 to $\Rightarrow \neg r_3 \succ r_4$, then we can derive $-\partial p$ and $-\partial \neg p$.

□

The following theorem says that there exists a chain of increasing expressive power among several tags, similar to the situation in defeasible logic without dynamic priorities.

Theorem 3 $+\Delta \subset +\partial_{ap} \subset +\partial \subset +\int$.

For each inclusion there are defeasible theories in which the inclusion is strict.

4 DDL2: Dynamic Defeasible Logic with Cycle Checking

4.1 Proof Theory

Define $Chains(label(r_1) \succ label(r_2))$ to be the set of all sequences of strict or defeasible rules, without multiple occurrences $C = (r^{(1)}, \ldots, r^{(n)})$ such that

- $C(r^{(i)}) = label(r'_i) \succ label(r'_{i+1})$, for all $i \in \{1, \ldots, n-1\}$.
- $r'_1 = r_1$, $r'_n = r_2$.

Intuitively, *Chains* collects all chains of priority statements, starting with $label(r_1)$ and ending at $label(r_2)$, that can be derived if all rules in the chain are applied. For a rule with head $label(r_2) \succ label(r_1)$ to be applied, we must first find in all chains for $label(r_1) \succ label(r_2)$ an inapplicable rule.

Formally, we expand the inference condition for $+\partial$ by a case (2.4), in case $P(i+1) = +\partial(label(r_1) \succ label(r_2))$.

$$(2.4)\ \forall C \in Chains(label(r_2) \succ label(r_1))$$
$$\exists r \in C\ \exists a \in A(r) : -\partial a \in P(1..i)$$

The inference conditions for $-\partial$, $+\partial_{ap}$ and $-\partial_{ap}$ are modified in the same way. The inference conditions for strict provability and support remain unaffected.

Example 4.1 Consider

$r_1 : \Rightarrow r_3 \succ r_4$ $\quad$ $r_3 : \Rightarrow p$
$r_2 : \Rightarrow r_4 \succ r_3$ $\quad$ $r_4 : \Rightarrow \neg p$

We cannot derive $+\partial(r_3 \succ r_4)$ because of the chain (r_2, r_1), in which neither rule can be proven to be inapplicable. Therefore $-\partial p$ and $-\partial \neg p$.

□

4.2 Properties

First we make the following observation: if $\succ$ does not occur in a defeasible theory, then DDL2 behaves exactly as the classical defeasible logic.

Theorem 4 (Coherence) *It is impossible to derive both $+tag\ q$ and $-tag\ q$ from a defeasible theory D, for tag $\in \{\Delta, \partial, \partial_{ap}, \int\}$.*

Theorem 5 (Consistency) *$+\partial p$ and $+\partial \neg p$, or $+\partial_{ap} p$ and $+\partial_{ap} \neg p$, only if $+\Delta p$ and $+\Delta \neg p$.*

In other words, an inconsistency in the defeasible part can only occur if the same inconsistency already occurs in the strict part.

As in DDL1, there exists a chain of increasing expressive power among several tags, similar to the situation in defeasible logic without dynamic priorities.

Theorem 6 $+\Delta \subset\subset +\partial_{ap} \subset +\partial \subset +\int$.

For each inclusion there are defeasible theories in which the inclusion is strict.

5 Argumentation Semantics

For space limitations we will only discuss the ambiguity blocking variant of DDL2. The argumentation semantics will be an expansion of [8] which considered the classic defeasible logic with static priorities.

5.1 Arguments

Usually arguments are defined to be proof trees (or monotonic derivations). However, defeasible logic requires a more general notion of argument due to the notion of team defeat (see [8] for details).

An *proof tree* for a literal p based on a set of rules R is a (possibly infinite) tree with nodes labelled by literals such that the root is labelled by p and for every node with label h:

> If $b_1, \ldots, b_n$ label the children of h then there is a ground instance of a rule in R with body $b_1, \ldots, b_n$ and head h.

Any literal labelling a node of proof tree P is called a *conclusion* of P. However, when we refer to *the* conclusion of a proof tree, we refer to the literal labelling the root. If all rules in a proof tree are strict, then the tree is called *strict*, else *defeasible*.

An *argument* A for p is a set of proof trees for p. A is *finite*, or *strict*, if all proof trees in A are finite or strict, respectively. A (*proper*) *subargument* of A is a (proper) subtree of a tree in A. Given a defeasible theory D, the set of arguments that can be generated from D is denoted by $Args_D$. Finally we define $\succ_{Args}$ to be the set of all literals $label(r_1) \succ label(r_2)$ which are conclusions of a rule that appears in an argument in $Args$.

At this stage we can characterize the definite conclusions of defeasible logic in argumentation-theoretic terms.

Proposition 5.1 *Let D be a defeasible theory and p be a literal.*

1. *$D \vdash +\Delta p$ iff there is a finite strict argument for p in $Args_D$*
2. *$D \vdash -\Delta p$ iff there is no (finite or infinite) strict argument for p in $Args_D$*

5.2 Interaction Between Arguments

An argument A *attacks* as argument B if a conclusion in A is the complement of a conclusion in B; or if $label(r_1) \succ label(r_2)$ is a conclusion in A, and all rules in a member of $Chains(label(r_2) \succ label(r_1))$ occur in a proof tree in B.

An argument A *defeats* a defeasible argument B at literal q w.r.t a set of arguments $Args$ if (1) there exist r_A in A and r_B in B with conclusions $\sim q$ and q, respectively, such that $label(r_A) \not\succ_{Args} label(r_B)$; or (2) $q = label(r_1) \succ label(r_2)$, r_B in B has conclusion q, and all rules in a member of $Cycles(label(r_2) \succ label(r_1))$ occur in a proof tree of A.

A set of arguments S defeats a defeasible argument B w.r.t. $Args$ if there is an $A \in S$ that defeats B w.r.t. $Args$. Note that $Args$ is used as a context which provides the superiority relation.

Example 5.1 Consider

$r_1 :\ a \Rightarrow p$ $\quad$ $\rightarrow a$
$r_2 :\ p \Rightarrow q$ $\quad$ $\rightarrow b$
$r_3 :\ b \Rightarrow \neg p$ $\quad$ $r_5 : \Rightarrow r_2 \succ r_4$
$r_4 :\ \neg p \Rightarrow \neg q$

Consider the arguments $A :\ a \Rightarrow p \Rightarrow q$ and $B :\ b \Rightarrow \neg p \Rightarrow \neg q$, and the set $Args = \{\Rightarrow r_2 \succ r_4\}$. The only priority information is $r_2 \succ_{Args} r_4$. Therefore A defeats B at $\neg p$ and $\neg q$ w.r.t. $Args$, and B defeats A at p w.r.t. $Args$.

□

Similar to [8] we define team defeat and undercut as follows: An argument *team defeats* a defeasible argument B at q w.r.t. $Args$ if for every rule r_B with conclusion q there is a rule r_A in A with conclusion $\sim q$, such that $label(r_A) \succ_{Args} label(r_B)$.

Let $strong(S)$ be the set of arguments, all of whose subarguments are supported by S. Obviously $S \subseteq strong(S)$. Also, note that if $A_1, A_2 \in strong(S)$ are arguments for the literal q, then $A_1 \cup A_2$ is also an argument for q. Thus there exists a maximal argument for q in $strong(S)$, denoted by $max(q, S)$.

A defeasible argument A is *undercut* by a set of arguments S if there is a literal q such that $strong(S)$ defeats a proper subargument of A at q w.r.t. S, and A does not team defeat $max(\sim q, S)$ at $\sim q$ w.r.t. S.

5.3 The Status of Arguments

An argument A is *supported* by a set of arguments S if every conclusion in A is also the conclusion of a finite argument in S. An argument A for p is *acceptable w.r.t. a set of arguments* S if (1) A is strict; or (2):

(2a) every proper subargument of A is supported by S, and
(2b) every argument attacking A is either undercut by S or team defeated by A.

Let D be a defeasible theory. We define J_i^D as follows:

- $J_0^D = \emptyset$
- $J_{i+1}^D = \{a \in Args_D \mid a \text{ is acceptable w.r.t. } J_i^D\}$

The set of *justified arguments* in D is $JArgs^D = \cup_{i \geq 1} J_i^D$. A literal is *justified* if it is the conclusion of a finite argument in $JArgs^D$.

Theorem 7 *$D \vdash +\partial q$ iff q is justified.*

An argument A is *rejected by sets of arguments S and T* if (1) A is defeasible, and either

(2a) a proper subargument of A is in S, or
(2b) there exists an argument B attacking A, such that B is supported by T, and A does not team defeat B w.r.t. T.

We define R_i^D as follows:

- $R_0^D = \emptyset$
- $R_{i+1}^D = \{a \in Args_D \mid a \text{ is rejected by } R_i^D \text{ and } JArgs^D\}$

The set of *rejected arguments* in D is $RArgs^D = \cup_{i \geq 1} R_i^D$. A literal is *rejected* if there is no argument for p in $Args^D - RArgs_D$.

Theorem 8 *$D \vdash -\partial q$ iff q is rejected.*

6 Conclusion

In this paper we studied dynamic priorities in the defeasible logics. We studied two variants, depending on whether cyclic priority information is allowed to occur or not. For each variant we considered levels of proof: strict, defeasible with ambiguity propagation, defeasible with ambiguity blocking, and support. We defined the proof theory, derived coherence and consistency, and established a chain of inclusion for the various levels of proof. Finally we defined a more abstract characterisation in the form of an argumentation semantics.

The next step in this research line is to implement the logics and use them in applications.

REFERENCES

[1] G. Antoniou, M.J. Maher and D. Billington. Defeasible Logic versus Logic Programming without Negation as Failure, Journal of Logic Programming, 42 (2000): 47–57.

[2] G. Antoniou, D. Billington, G. Governatori and M.J. Maher. A Family of defeasible reasoning logics and its implementation. In *Proc. 14th European Conference on Artificial Intelligence (ECAI'2000)*, 459–463.

[3] G. Antoniou, D. Billington, G. Governatori and M.J. Maher. Representation results for defeasible logic. *ACM Transactions on Computational Logic* 2 (2001): 255–287

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

[5] G. Brewka. Well-Founded Semantics for Extended Logic Programs with Dynamic Priorities. *Journal of Artificial Intelligence Research* 4 (1996): 19-36.

[6] G. Brewka and T. Eiter. Preferred Answer Sets for Extended Logic Programs. *Artificial Intelligence* 109 (1999): 297-356.

[7] J.P. Delgrande, T Schaub and H. Tompits. Logic Programs with Compiled Preferences. In *Proc. ECAI'2000*, 464–468.

[8] G. Governatori and M.J. Maher. An Argumentation-Theoretic Characterization of Defeasible Logic. In *Proc. ECAI'2000.*

[9] G. Governatori, A. ter Hofstede and P. Oaks. Defeasible Logic for Automated Negotiation. In *Proc. Fifth CollECTeR Conference on Electronic Commerce*, Brisbane 2000.

[10] B.N. Grosof. Prioritized conflict handling for logic programs. In *Proc. International Logic Programming Symposium*, MIT Press 1997, 197–211.

[11] B.N. Grosof, Y. Labrou and H.Y. Chan. A Declarative Approach to Business Rules in Contracts: Courteous Logic Programs in XML. In *Proc. 1st ACM Conference on Electronic Commerce (EC-99)*, ACM Press 1999.

[12] H.A. Kautz and B. Selman. Hard problems for simple default theories. *Artificial Intelligence* 28 (1991): 243–279.

[13] M.J. Maher, A. Rock, G. Antoniou, D. Billington and T. Miller. Efficient Defeasible Reasoning Systems. In *Proc. 12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2000)*, IEEE 2000, 384–392.

[14] M.J. Maher. Propositional Defeasible Logic has Linear Complexity. *Theory and Practice of Logic Programming*, to appear.

[15] 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.

[16] H. Prakken. *Logical Tools for Modelling Legal Argument: A Study of Defeasible Reasoning in Law.* Kluwer Academic Publishers 1997.

[17] H. Prakken, G. Sartor. Argument-based logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics* 7 (1997): 25-75.

[18] D.M. Reeves, B.N. Grosof, M.P. Wellman, and H.Y. Chan. Towards a Declarative Language for Negotiating Executable Contracts, *Proceedings of the AAAI-99 Workshop on Artificial Intelligence in Electronic Commerce (AIEC-99)*, AAAI Press / MIT Press, 1999.

[19] 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*, Morgan Kaufmann 1987, 476–482.