A Defeasible Logic Programming System for the Web

Grigoris Antoniou and Antonis Bikakis
Computer Science Department, University of Crete, Greece
Institute of Computer Science, FORTH, Greece
{ga,bikakis}@csd.uoc.gr

Gerd Wagner
Eindhoven University of Technology
Department of Technology Management, Information Systems
G.Wagner@tm.tue.nl

Abstract

Defeasible reasoning is a rule-based approach for efficient reasoning with incomplete and inconsistent information. Such reasoning is, among others, useful for ontology integration, where conflicting information arises naturally; and for the modeling of business rules and policies, where rules with exceptions are often used. This paper describes these scenarios in more detail, and reports on the implementation of a system for defeasible reasoning on the Web. The system (a) is syntactically compatible with RuleML; (b) features strict and defeasible rules and priorities; (c) is based on a translation to logic programming with declarative semantics; and (d) is flexible and adaptable to different intuitions within defeasible reasoning.

1. Introduction

The development of the Semantic Web [1] proceeds in layers, each layer being on top of other layers. At present, the highest layer that has reached sufficient maturity is the ontology layer in the form of the description logic based languages of DAML+OIL and OWL.

The next step in the development of the Semantic Web will be the logic and proof layers. Rule systems can play a twofold role in the Semantic Web initiative: (a) they can serve as extensions of, or alternatives to, description logic based ontology languages; and (b) they can be used to develop declarative systems on top (using) ontologies.

Defeasible reasoning is a simple rule-based approach to reasoning with incomplete and inconsistent information. It can represent facts, rules, and priorities among rules. Its main advantage is the combination of two desirable features: enhanced representational capabilities allowing one to reason with incomplete and contradictory information, coupled with low computational complexity compared to mainstream nonmonotonic reasoning.

In this paper we report on the implementation of a defeasible reasoning system for reasoning on the Web. Its main characteristics are the following:

- Its user interface is compatible with RuleML [2], the main standardization effort for rules on the SW.
- The core of the system consists of a translation of defeasible knowledge into Prolog. However, the implementation is declarative because it interprets the not operator using Well-Founded Semantics [3].
- The main focus was flexibility. Strict and defeasible rules and priorities are part of the implementation. Also, a number of variants were implemented (ambiguity blocking, ambiguity propagating, conflicting literals).

2. Defeasible Logics

2.1 Basic definitions

A *defeasible theory D* is a couple (*R*,>) where *R* a finite set of rules, and > a superiority relation on R. In expressing the proof theory we consider only propositional rules. Rules containing free variables are interpreted as the set of their variable-free instances.

There are two kinds of rules (fuller versions of defeasible logics include also defeaters): *Strict rules* are denoted by $A \rightarrow p$, and are interpreted in the classical sense: whenever the premises are indisputable then so is the conclusion. *Defeasible rules* are denoted by $A \Rightarrow p$, and can be defeated by contrary evidence. A *superiority relation* on R is an acyclic relation > on R. When $r_1 > r_2$, then r_1 is called *superior* to r_2, and r_2 *inferior* to r_1. This expresses that r_1 may override r_2. A formal definition of the proof theory is found in [4].

2.2 Variants of defeasible logic

A literal is *ambiguous* if there is a chain of reasoning that supports a conclusion that p is true, another that supports that $\neg p$ is true, and the superiority relation does not resolve this conflict. In the ambiguity blocking variant

Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004)
1082-3409/04 $20.00 © 2004 **IEEE**

of defeasible logic, a rule containing an ambiguous literal in its premises cannot be used to "block" a conflicting rule (a rule containing a conflicting inference).

In the ambiguity propagation variant, even when the premises of a rule are ambiguous, this rule can be used to oppose a conflicting rule. 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 variant of DL is of interest. In our work, we implemented both variants of defeasible logic.

2.3 Conflicting literals

So far only conflicts among rules with complementary heads were detected and used. We considered all rules with head L as *supportive* of L, and all rules with head $\neg L$ as *conflicting*. However, in applications often literals are considered to be conflicting, and at most one of a certain set should be derived. For example, the risk an investor is willing to take may be classified in one of the categories low, medium, and high. The way to solve this problem is to use a constrain rule of the form

conflict :: low, medium, high

Now the rules that oppose the conclusion *high* are not just those with head $\neg high$, but also those with head *low* and *medium*. Similarly, the rules that support $\neg high$, include those with head *low* or *medium*.

In general, given a *conflict :: L, M*, we augment the defeasible theory by:

r_i: $q_1,q_2,\ldots,q_n \rightarrow \neg L$ *for all rules* r_i: $q_1,\ldots,q_n \rightarrow M$
r_i: $q_1,q_2,\ldots,q_n \rightarrow \neg M$ *for all rules* r_i: $q_1,\ldots,q_n \rightarrow L$
r_i: $q_1,q_2,\ldots,q_n \Rightarrow \neg L$ *for all rules* r_i: $q_1,\ldots,q_n \Rightarrow M$
r_i: $q_1,q_2,\ldots,q_n \Rightarrow \neg L$ *for all rules* r_i: $q_1,\ldots,q_n \Rightarrow M$

3. Translation into Logic Programs

The translation of a defeasible theory D into a logic program $P(D)$ has a certain goal: to show that

p is defeasibly provable in $D \Leftrightarrow$
p is included in all stable models of $P(D)$

We based our work on the translation which makes use of control literals, presented in [5]. We have made some extensions to support superiority relations, and the two different variants of defeasible logic.

In the ambiguity blocking variant: Given a fact p we translate it into the program clause

$a(p)$: *definitely*(p).

A strict rule r: $q_1,q_2,\ldots,q_n \rightarrow p$ is translated into

$b(r)$: *definitely*(p):- *definitely*$(q_1),\ldots,$*definitely*(q_n).

Additionally, for every literal p we introduce the clause

$c(p)$: *defeasibly*(p):- *definitely*(p).

A defeasible rule r: $q_1,q_2,\ldots,q_n \Rightarrow p$ is translated into

$d_1(r)$: *defeasibly*(p):- *defeasibly*(q_1), $\ldots,$*defeasibly*(q_n), *not definitely*$(\neg p)$,*ok*(r,p).
$d_2(r)$: *ok*(r,x):- *ok'*$(r,s_1),\ldots,$*ok'*(r,s_m).
where $\{s_1,\ldots,s_m\}$ = { *defeasible rules with head:* $\neg p$}
$d_3(r,s_i)$: *ok'*(r,s_i):- *blocked*(s_i). *for all* $s_i \in \{s_1,\ldots,s_m\}$
$d_4(r,s_i)$: *ok'*(r,s_i):- *defeated*(s_i). *for all* $s_i \in \{s_1,\ldots,s_m\}$
$d_5(r,q_i)$: *blocked*(r):- *not defeasibly*(q_i). *for all* $i \in \{1,2,\ldots,n\}$
$d_6(r,s_i)$: *defeated*(r):- *not blocked*(s_i), *sup*(s_i,r). *for all* $s_i \in \{s_1,\ldots,s_m\}$

A superiority relation $r > s$ is translated it into

$e(r,s)$: *sup*(r,s).

In the ambiguity propagation variant, for every literal p we add the new predicate

$s(p)$: *supported*(p):- *definitely*(p).

we change d_3 to

$d_3'(r,s_i)$: *ok'*(r,s_i):- *obstructed*(s_i). *for all* $s_i \in \{s_1,\ldots,s_m\}$

and add the program clauses

$d_7(r,q_i)$: *obstructed*(r):- *not supported*(q_i). *for all* $i \in \{1,2,\ldots,n\}$,
$d_8(r)$: *supported*(p):- *supported*$(q_1),\ldots,$*supported*(q_n), *not defeated*(r).

4. Translation into XML files

Another part of our work was the creation of a DTD which allows to translate defeasible theories into XML files. This is in fact an extension of the RuleML DTDs [5], which covers both strict and defeasible rules, as well as the superiority relations. The elements of the RuleML DTD that we added / modified are:

- The "rulebase" root element which uses "imp" and "def" rules, "fact" assertions and "superiority" relations.
- The "imp" element, which consists of a "_head" and a "_body" element, accepts a "name" attribute, and refers to the strict rules of a theory.
- The "def" element which is similar to the "imp" element, and refers to the defeasible rules of a theory.
- The "superiority" empty element, which accepts the name of two rules as its attributes ("sup" & "inf"), and refers to the superity relation betweeen these two rules.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284, 5 (2001): 34-43

[2] RuleML. *The Rule Markup Language Initiative*. www.ruleml.org

[3] A. van Gelder, K. Ross and J. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM* 38 (1991): 620—650

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

[5] G. Antoniou, M.J. Maher. Embedding Defeasible Logic into Logic Programs. In *Proc. ICLP 2002,* 393-404