Extending Defeasible Logic and Defeasible Prolog

Ahti-Veikko Pietarinen

1 Department of Philosophy, University of Helsinki, Finland, email: pietarin@cc.helsinki.fi. Supported by the Academy of Finland (1103130).

Abstract. Defeasible logic (DL) promotes enthymemtic, argumentative reasoning on incomplete set of premisses retracted on the presence of contrary information. Defeasible Prolog (d-Prolog) is a Prolog metainterpreter to implement DL. We give proof conditions for the even-if clauses of DL with the pre-emption of defeaters to prevent rules from rebutting more specific rules, implemented in d-Prolog.

1 INTRODUCTION

The goal of defeasible logic (DL) [4, 5, 6] is to formalise nonmonotonic inferences (‘typically, ϕ’s are ψ’, ‘reasonable grounds for holding ϕ warrant reasonable grounds for holding ψ’). Such inferences hold only if a defeasible theory contains no rules representing contrary information. Defeasible Prolog (d-Prolog) is a Prolog metainterpreter to implement DL. We give proof conditions for implement DL. We give proof conditions for the even-if clauses of DL with the pre-emption of defeaters to prevent rules from rebutting more specific rules, implemented in d-Prolog.

2 RULES FOR THE ‘EVEN-IF’ CONDITION

Defeasible even-if rule is ψ := Σ | Φ′ (‘if Φ, then (typically) ψ’ even if Σ holds’), permitting superiority by more specific antecedent:

Example 2.1 A beneficiary is suspect:
suspect(X) := beneficiary(X).

With an alibi a person usually is not suspect even if he is a beneficiary:
neg suspect(X) := beneficiary(X) | alibi(X).

Tom has an alibi: alibi(tom). He must not be a suspect. In both cases it can be concluded tentatively that neg suspect(tom) is a derivation from T. Let us then add the defeater, according to which a person with an alibi who has not provided reliable documents may well be suspect:

(suspect(X) := (alibi(X), neg rel_doc(X)).

If Tom has not provided reliable documents (neg rel_doc(tom)), it is difficult to conclude, with this amount of information, whether Tom is suspect or not. In extended d-Prolog (App.), both neg suspect(tom) and suspect(tom) are demonstrably not derivable from the T, since the rule

(suspect(X) := (alibi(X), neg rel_doc(X)) is not an acceptable rule to pre-empt an inferior rule

neg suspect(X) := beneficiary(X) | alibi(X),

because pre-emption is possible only if a rule is defeated with a superior rule.

Let n be arbitrary in t, and let θψ denote when ψ is a tentative conclusion derived from T. Conditions P⁺ and P⁻ hold for all n, the former for literals defeasibly derivable and the latter for literals demonstrably not derivable:

P⁺: Node n is labelled (K, R, ψψ) if either n has child (K, R, neg ψψ) or there exists ψ := Ξ | Φ ∈ R such that 1, 2 and 3 hold:

1. for every ϕ ∈ Φ, n has child (K, R, ψψ) (every literal in main body is derivable)
2. for every neg ψ := R ∈ H there exists η ∈ H and child of n (K, R, η) (every contrary rule must have nonderivable literal in the body)
3. for every neg ψ := Δ | Θ ∈ R or neg ψ := Θ ∈ R either a or b:
   a) there exists θ ∈ Θ and child of n (K, R, θθ) (every contrary rule must have nonderivable literal in main body; subsumes clause 2)
   b) i, ii and iii hold:
      i) there exists ϕ ∈ Φ ∪ Ξ and child of n (Φ, R, ϕϕ) (specificity: some literal in antecedents not derivable from antecedents of contraries)
      ii) for every ϕ ∈ Φ there exists child of n (Φ ∪ Ξ, R, ϕϕ) (specificity: any literal in main condition of contrary is derivable from antecedents of main rule)
      iii) for every ϕ ∈ Φ ∪ Ξ there exists child of n (K, R, ϕϕ) (pre-emption: every literal in body of main rule is defeasibly derivable).
Derivation of incompatible clauses and prevention of cycles depends on programmer. To prevent cyclic dependencies, we may stratify predicates to see which rules to be used to derive conclusion. Prior to referring to a negation of a fact, the fact itself needs to be defined: stratified program partitions its clauses into hierarchical sets in which negative goals are defined in lower-level predicates. If some programs cannot be stratified, in well-founded semantics some facts are undefined and in bivalent stable model semantics non-stratified programs may have several models or none. It is also possible to consider different grades of stratification in the sense that only certain kinds of rules, such as defeasible rules, strict rules or defeaters, are subject to stratification, amounting to semi-stratified programs.3

Another development intergrates defeasible rules with argumentative structures and defeasibility among arguments [3] in legal, economic and decision-making systems [7]. Answer-set programming performs nonmonotonic reasoning [1, 2], including defeasible rules.

4 CONCLUSIONS

We have given proof conditions for the even-if clauses of DL with the pre-emption of defeaters to prevent rules from rebuting more specific rules. The extension has been implemented in d-Prolog.

REFERENCES


A EXTENSION OF D-PROLOG

We add a two-place relation \( i \) (in App.) to distinguish primary condition \( \Phi \) in \( \psi :\subseteq \Phi \) and even-if condition \( \Xi \) in \( P^+ \) and \( P^- \). Three additional to defeat and undercut of \( [6] \) are needed.

Preemption (preempted/2) is similar to defeating (App.). Two new definitions for the defeasible derivability (def\_der(2)), three for defining superiority relation with defeasible specificity (sup\_rule/2) and some others for syntax and occurrences of the even-if rules also exist (App.).

Enabling preemption of defeaters increases size of proof trees considerably. These are the sizes without/with the defeater of Ex.2.1:

<table>
<thead>
<tr>
<th>pre-emption</th>
<th># suspect (tom)</th>
<th># neg suspect (tom)</th>
</tr>
</thead>
<tbody>
<tr>
<td>–</td>
<td>25 / 23</td>
<td>90 / 115</td>
</tr>
<tr>
<td>–</td>
<td>53 / 53</td>
<td>98 / 270</td>
</tr>
</tbody>
</table>

To limit the complexity of defeasible reasoning one may i) restrict the depth of \( t \) by generating paths only up to a certain limit, and after reaching the limit, proofs backtrack to earlier \( n \); ii) restrict branching factor of \( t \) by keeping the size of \( R \) and maximum number of conjunctions small. The former has a side effect: as argumentative inference uses depth-first search, constraining it may cause the horizon effect. Limiting maximal depth amounts to cheap loop checking. The latter reflects defeasible argumentation as only the most pertinent arguments should be used.2

2 In conversations it is often necessary to lay size restrictions to the set of assertions. Usually one or two defeaters is enough to render opposite views unwarranted.

3 As far as semantics is concerned, models of DL are ceteris paribus situations insensitive to vagueness. We believe that prototype theory concerning new information in pre-existing structures of knowledge by relaxing strict category membership is a good candidate for a defeasible model theory.