Extending a Defeasible Reasoner with Modal and Deontic Logic Operators

Efstratios Kontopoulos[1], Nick Bassiliades[1], Guido Governatori[2] and Grigoris Antoniou[3]

[1]*Dept. of Informatics, Aristotle Univ. of Thessaloniki, GR-54124 Thessaloniki, Greece,* *{skontopo, nbassili}@csd.auth.gr*

[2]*NICTA, Australia,* *guido.governatori@nicta.com.au*

[3]*Institute of Computer Science, FO.R.T.H., P.O. Box 1385, GR-71110, Heraklion, Greece* *antoniou@ics.forth.gr*

Abstract

Defeasible logic is a non-monotonic formalism that deals with incomplete and conflicting information. Modal logic deals with necessity and possibility, exhibiting defeasibility; thus, it is possible to combine defeasible logic with modal operators. This paper reports on the extension of the DR-DEVICE defeasible reasoner with modal and deontic logic operators. The aim is a practical defeasible reasoner that will take advantage of the expressiveness of modal logics and the flexibility to define diverse agent types and behaviors.

1. Introduction

Defeasible Logic (DL) [1] is a non-monotonic formalism that deals with incomplete and conflicting information, offering enhanced representational capabilities and low computational complexity. *Modal Logic* [2] is a system of formal logic that deals with *modalities*, i.e. expressions associated with *possibility* and *necessity*. Modal logics attach intentional operators to classical logic, which, however, requires complete and consistent information, rarely encountered in real-life scenarios that are often defeasible. Also, reasoning about motivational notions, like intentions or obligations, displays significant defeasibility. Already, DL is applied in a number of related fields and its extension with modal and deontic operators significantly improves its expressiveness, as recent work suggests [3]. Additionally, defeasible and modal logics provide a means for modeling *multi-agent systems* (MAS), where each agent is characterized by its own cognitive profile and normative system, as well as *policies* that define privacy requirements, access permissions etc.

In this paper we report on extending the DR-DEVICE DL reasoner [4] with reasoning capabilities on modal DL rule bases. The goal is to develop a practical DL reasoner that will take advantage of the expressive power of modal logics, accompanied by the flexibility to define various agent types and behaviors.

2. Defeasible, Modal and Deontic Logics

A DL rule base consists of a set of facts (F), a set of rules (R) and a superiority relationship ($>$). There are three rule types: a) *Strict rules* ($A \rightarrow p$) are classical logic rules: when the premises A are indisputable, then so is the conclusion p, b) *Defeasible rules* ($A \Rightarrow p$) can be defeated by contrary evidence, and c) *Defeaters* ($A \rightsquigarrow p$) can only defeat conflicting conclusions, by producing contrary evidence. The *superiority relationship* is an acyclic relation on R that resolves rule conflicts.

Modal logic deals with *possibility* and *necessity*, but often covers a wider range of concepts, paving the way for MAS – modal operators can express the cognitive states and interactions of agents. *Deontic logic* is a modal logic variation, concerned with *permission* and *obligation*, used in expressing *policies* that define access permissions, privacy requirements, etc. Especially for the Semantic Web, policies and automated trust negotiation are very appealing applications ([5], [6]).

3. Modal Defeasible Logic

The proposed system has been extended to deal with the following modal (deontic) operators:

- *Belief* (K): represents the agent's theory of the world
- *Intention* (I): corresponds to the agent's policies
- *Obligation* (O): represents agent's normative system
- *Agency* (Z): captures the agent's intentional actions
- *Permission* (P): corresponds to what the agent is permitted to do, according to its normative system

The basic directions for extending DL with modal operators are outlined in [3]: a *modalized literal* is a

978-0-7695-3496-1/08 $25.00 © 2008 IEEE
DOI 10.1109/WIIAT.2008.164

literal accompanied by a modal prefix. E.g. the fact "*I intend to go to Rome*" is represented as `I(goToRome)`. Also, new types of rules are introduced. E.g. the rule "*I intend to go to Rome for the summer*" is represented as: `summer` $\Rightarrow_I$ `goToRome`. Since rules are meant to introduce modalities, modalized literals appear only in the rule body and not in the head. A literal with prefix *K* is equivalent to the same literal without any prefix, i.e. it belongs to the agent's knowledge base. Hence, rules with mode *K* produce un-prefixed conclusions.

With the exception of the belief operator *K*, the rest of the modal operators treated by the system are *non-reflexive*, i.e. if *X* is a modal operator, *a* does not follow from *X*(*a*). Also, *iterated modalities*, like *I*(*I*(*a*)) or *O*(*I*(*a*)) or *modal reduction axioms*, like *Z*(*O*(*a*)) → *I*(*a*) and *I*(*O*(*a*)) → *I*(*a*), are not yet considered.

A conclusion in modal DL is represented with a tagged literal, similarly to the proof theory of DL. Thus, such a conclusion can have one of four forms:

- $\pm\Delta_X q$: *q* is (not) definitely provable in modality *X*.
- $\pm\partial_X q$: *q* is (not) defeasibly provable in modality *X*.

An example is the well-known "prisoner's dilemma" from game theory that can be formalized as:

p_1: committedCrime ∧ arrested $\Rightarrow_Z$ confess
p_2: committedCrime ∧ arrested $\Rightarrow_O$ ¬confess

This is a typical case of conflict among an agent's intentions and its normative system. According to the agent type, various conclusions can be drawn (see section 3.1 for an insight). More details regarding the language and inference of modal DL are found in [3].

3.1. Conflict Resolution

In modal DL a rule is attacked by another rule, when the two conclusions are complementary and the two rule modes are different. Depending on the rule modes, a set of criteria is designated, determining which rule prevails. There exist *basic attacks* that always apply, but there are also attacks that depend on the agent type. In general, beliefs override all other modes, except from agency, since they represent the agent's knowledge of the world and are considered superior to its policies and normative system. Actions, however, naturally override beliefs, since they can have a contradicting effect on the agent's knowledge base. Intention and agency are mutually attacked, since the latter are intentional by definition. Mutual attacks also exist among obligation and permission. Agents that encompass the above conflict resolution scheme are called *realistic*.

Nevertheless, the conflicts among intentions, obligations and actions are resolved via the *agent type*. Thus, agent types establish conflict resolution schemes, by determining the way that rules of various modes interact with each other. Our implementation currently includes only *social* (obligations override all intentions and actions to the contrary) and *deviant* agents (obligations are overridden by opposing intentions and intentional actions), accompanied by the *realistic* type, but the variety can easily be extended, as explained later.

3.2. Rule Conversion

There are cases when the rule mode is not inherited to the conclusion, but a different modality is adopted. This feature, called *rule conversion*, allows converting the mode of a rule into another mode, depending on the modalities, in which the corresponding rule premises have been proved. Generally speaking, conversions are a means to derive some rational side effects.

Conflicts and conversions are not directly interrelated, but both help portray the cognitive profile of an agent. However, similarly to conflicts, there exist rule conversions that apply to all agent types, but there are also agent type-specific conversions.

4. Implementation

DR-DEVICE [4] is a DL reasoner that employs an object-oriented RDF data model, treating properties as encapsulated attributes of resources. It employs a RuleML-like syntax for DL rules, which extends the official RuleML specifications (v. 0.91) with rule types, superiority relations among rules, conflicting literals, and constraints on predicate arguments and functions.

The RuleML syntax was further extended to embrace the required elements of modal logic, namely, rule modes and modalized literals. Two attributes are introduced: a) `ruleMode`, attached to the `Implies` element, and b) `modality`, attached to `Atom`. Both attributes are restricted to the values: `bel` (*belief*), `int` (*intention*), `obl` (*obligation*), `age` (*agency*) and `per` (*permission*). These attributes are optional; thus, when they are absent, their default value is `bel`. Finally, the notion of agent type is represented by an `agentType` attribute, attached to the rule base element (`RuleML`). More agent types can be easily added to the schema.

4.1. Theory Transformation

DR-DEVICE takes as input a modal DL rule base and applies a transformation that moves modalities from rules to conclusions, also taking care of *modal interactions* (i.e. conflicts, conversions). The transformation is based on the following two binary anti-

symmetric relations that define how pairs of modalities interact in rule conflicts ($\prec$) and rule conversions ($\mapsto$).

DEFINITION 1 (rule conflicts):

$\prec \subseteq M \times M$, $X \in M$, $Y \in M$, $Y \prec X$ if-f

$\forall r, r', q: r \in R^X[q],\ r' \in R^Y[\sim q],\ X \neq Y \rightarrow r > r'$

M is the set of all modality types and $R^Y[q] \subseteq R$ denotes the set of rules with mode Y and literal q as their consequent (R is the set of all rules). The above definition implies that for every pair $Y \prec X$, if there is an attack $\Rightarrow_X q$ / $\Rightarrow_Y \sim q$ for any literal q in the rule base, the conclusion of the rule with mode X will be derived ($+\partial_X q$), while the other conclusion will not be provable ($-\partial_Y \sim q$), i.e. mode X prevails over Y.

DEFINITION 2 (rule conversions):

$\mapsto \subseteq M \times M$, $U \in M$, $Y \in M$, $U \mapsto Y$ if-f

$\forall r: r \in {}_Y\left\|R^U[q]\right\| \rightarrow \exists r': r' \in {}_Y\left\|R^Y[q]\right\|$

where ${}_Y\left\|R^U[q]\right\|$ denotes the set of all instantiations of rule R with mode U, q as the consequent and Y as the modality of all antecedents of the rule instantiation. Notice that $\mapsto$ is reflexive, i.e. $\forall X$, $X \mapsto X$ is true.

The transformation is affected by the OO nature of DR-DEVICE. First, the modality of every rule body atom is transformed from an XML attribute into an argument of the `atom` element. Since atoms are treated as objects, the modality becomes a slot with the same name. Then, rule head atoms assume a modality that depends on the rule mode and the modalities of the body atoms. Rule conversions are considered as well and result in additional rules being added to the rule base: for each rule $Y(a_1) \wedge Y(a_2) \wedge \ldots \wedge Y(a_n) \Rightarrow_X q$ of the original DL theory (all antecedents share the same modality Y), a new rule $M(a_1) \wedge M(a_2) \wedge \ldots \wedge M(a_n) \wedge X \mapsto M \Rightarrow M(q)$ is appended to the theory. The additional rules assess whether a rule qualifies for conversion and are transparent to the user. Finally, the modality of each rule head atom also becomes an object slot.

After all atoms have been assigned a modality, rule attacks are tackled. In DR-DEVICE, two modalized literals, such as $I(p)$ and $O(\sim p)$, are not normally conflicting, since they are treated as different objects. In order to be considered as conflicting, extra rules are transparently added to the rule base, realizing *modality inclusion* (i.e. associations among modalities). The results of attacks are determined by the schemes for basic and agent-type attacks, namely by the rule conflicts relation. Depending on the result for each attack $\Rightarrow_X q$ / $\Rightarrow_Y \sim q$ among modalities X, Y and for every conclusion q in the rule base, the following apply:

- If $Y \prec X$, i.e. $+\partial_X q$ / $-\partial_Y \sim q$, then the modality inclusions to be incorporated are:
  $X(q) \rightsquigarrow Y(q)$ $\quad$ $X(\neg q) \rightsquigarrow Y(\neg q)$
- If $Y \prec X \wedge X \prec Y$, i.e. $-\partial_X q$ / $-\partial_Y \sim q$, then the modality inclusions are:
  $X(q) \rightsquigarrow Y(q)$ $\quad$ $Y(q) \rightsquigarrow X(q)$
  $X(\neg q) \rightsquigarrow Y(\neg q)$ $\quad$ $Y(\neg q) \rightsquigarrow X(\neg q)$

The above rules are formulated as defeaters, since they do not aim at deriving new knowledge, but are only used for defeating rules with contrary conclusions. As for the remaining two cases of attacks ($X \prec Y$ and $Y \not\prec X \wedge X \not\prec Y$), the former is the inverse of the first case and is treated similarly, while the latter does not result in the addition of extra rules to the rule base, since both conclusions ($+\partial_X q$, $+\partial_Y \sim q$) are derived.

4.2. Implementing Modal Interactions

In order to represent modal interactions in a simple yet expressible way that could facilitate re-use and evolution, two schemes have been deployed, for rule conversions and rule conflict resolution, which are defined in two external configuration files.

The *conflicts schema* consists of tuples $<A, X, Y>$:

- A is the agent type ($A \in$ {`realistic`, `deviant`, `social`}) – the `realistic` agent type is a subtype of the other two, thus all its conflicts are also included in the conflict set of each of the other types.
- X is the superior rule mode, namely, the mode of the rule that prevails ($X \in M$, $M \equiv \{K, I, O, Z, P\}$).
- Y is the inferior rule mode, namely, the mode of the rule that is defeated ($Y \in M$, $M \equiv \{K, I, O, Z, P\}$).

The *rule conversions schema* consists of tuples $<A, U, Y>$:

- A is the agent type ($A \in$ {`realistic`, `deviant`, `social`}) – the `realistic` agent type is a subtype of the other two, thus all its conversions are included in the rule conversion set of each of the other types.
- U is the rule mode ($U \in M$, $M \equiv \{K, I, O, Z, P\}$).
- Y is the modality of *all* premises ($Y \in M$, $M \equiv \{K, I, O, Z, P\}$, so that $Y(a_1) \wedge Y(a_2) \wedge \ldots \wedge Y(a_n) \Rightarrow_U q$).

The rule mode is converted from U to Y: $Y(a_1) \wedge Y(a_2) \wedge \ldots \wedge Y(a_n) \Rightarrow_Y q$.

4.3. Example: Prisoner's Dilemma

To illustrate the above, we use the prisoner's dilemma example from section 3, with a rule base of two rules and two additional facts:

f_1: committedCrime f_2: arrested

Suppose we deal with a *social* agent. The rule base is transformed by turning predicate modalities into atom slots and assigning modalities to rule heads, depending on the corresponding rule modes. Predicates with no modality assume modality *K*. Concerning rule attacks, the rule base contains only a single conclusion (confess) and a single attack ($\Rightarrow_Z$ *confess* / $\Rightarrow_O$ *~confess*); therefore, two defeaters are added for the specific agent type (any obligation is also the agent's intentional action). The transformed rule base is:

$f_{1\text{-CONV}}$: committedCrime(K)

$f_{2\text{-CONV}}$: arrested(K)

$p_{1\text{-CONV}}$: committedCrime(K) $\wedge$ arrested(K) $\Rightarrow$ confess(Z)

$p_{1\text{-CONV-VAR}}$: committedCrime(M) $\wedge$ arrested(M) $\wedge$ Z $\mapsto$ M $\Rightarrow$ confess(M)

$p_{2\text{-CONV}}$: committedCrime(K) $\wedge$ arrested(K) $\Rightarrow$ $\neg$confess(O)

$p_{2\text{-CONV-VAR}}$: committedCrime(M) $\wedge$ arrested(M) $\wedge$ O $\mapsto$ M $\Rightarrow$ $\neg$confess(M)

$p_{O\text{-Z-POS}}$: confess(O) $\rightsquigarrow$ confess(Z)

$p_{O\text{-Z-NEG}}$: $\neg$confess(O) $\rightsquigarrow$ $\neg$confess(Z)

Rule $p_{1\text{-CONV}}$ is defeated by $p_{O\text{-Z-NEG}}$ and it is concluded that $+\partial_O$*~confess* and $-\partial_Z$*confess*. A slight variation of the example that demonstrates rule conversion would include facts f_1':Z(committedCrime) and f_2':Z(arrested) instead of f_1, f_2. In this case, rule $p_{2\text{-CONV-VAR}}$ tries to conclude $\neg$confess(Z), since the agent is *social* and O$\mapsto$Z, while rule $p_{1\text{-CONV-VAR}}$ tries to conclude confess(Z) (since Z$\mapsto$Z). Thus, no conclusion can be derived, even though the agent is social and this would incline to its cooperativeness.

5. Related Work

The work in this paper relies on [3], where a thorough account of modal DL is given. The main differentiation among the two lines of research is the fact that [3] is based on the meta-program formalization presented in [7], while our approach adopts a DL theory transformation for turning a modal DL theory into a non-modal one. Other differences involve the permission operator and the definition of the agent type.

The system presented in [8] has a similar functionality to ours, having the capability of dealing with rule conflicts, rule conversions and describing various agent types. The main difference lies in the higher degree of centralization and modularization offered by our system: the agent type can be declared centrally in each rule base, while modal interactions are dealt with parametrically. Also, the RuleML-like language for describing a modal DL rule base is easy to grasp, since the extensions to the language are limited.

6. Conclusions and Future Work

This paper discusses the extension of DL with modal logic elements and reports on the implementation of a modal DL reasoner, based on DR-DEVICE, a DL reasoner over RDF metadata. DR-DEVICE was extended to represent and handle modal and deontic logic operators. The system also deals with conflict resolution and rule conversion. The submitted DL theory undergoes a transformation, imposed by the object-oriented nature of DR-DEVICE, so that modal DL reasoning can be successfully performed.

There is still room for improvement, since the system currently handles only three types of agents, but can easily be extended. New agent types will consequently allow more rule conversions and a more expressive rule language. The ultimate goal involves the implementation of a MAS for deploying argumentation scenarios among agents, each one with its own agenda and normative system.

7. References

[1]. Nute, D., "Defeasible Logic", In D. Gabbay (ed.) *Handbook of Logic and Artificial Intelligence*, Oxford University Press, 1994, 3:353-395.

[2]. Hughes, G., Cresswell, M., *A New Introduction to Modal Logic*, Routledge, London, 1996.

[3]. Governatori, G., Rotolo, A., "BIO Logical Agents: Norms, Beliefs, Intentions in Defeasible Logic", *J. of Autonomous Agents and MAS*, forthcoming, 2008.

[4]. Bassiliades, N., Antoniou, G., Vlahavas, I., "A Defeasible Logic Reasoner for the Semantic Web", *Int. J. on Semantic Web and Inf. Systems*, 2006, 2(1):1-41.

[5]. Bonatti, P.A., Duma, C., Fuchs, N., Nejdl, W., Olmedilla, D., Peer, J, Shahmehri, N., "Semantic web policies - a discussion of requirements and research issues", *3rd European Semantic Web Conf.*, LNCS vol. 4011, Budva, Montenegro, Springer, 2006.

[6]. Kagal, L., Berners-Lee, T., Connolly, D, Weitzner, D., "Using Semantic Web Technologies for Open Policy Management on the Web", *21st National Conf. on Artificial Intelligence (AAAI 2006)*, July 16 – 20, 2006.

[7]. Maher, M. J., Governatori, G., "A semantic decomposition of defeasible logics", *Proc. 16th Nat. Conf. on Artificial intelligence*, Orlando, AAAI, 1999, 299-305.

[8]. Antoniou, G., Dimaresis, N., Governatori, G., "A System for Modal and Deontic Defeasible Reasoning", *20th Australian Joint Conf. on Artificial Intelligence*, Gold Coast, Queensland, 2-6 December, 2007.