

Description Logics with Transitive Roles

Ian Horrocks and Graham Gough

Department of Computer Science

University of Manchester

email: {horrocks—graham}@cs.man.ac.uk

Abstract

This paper describes the logic \mathcal{ALCH}_{R^+} , which extends \mathcal{ALC}_{R^+} with a primitive role hierarchy, and presents an appropriate extension to the \mathcal{ALC}_{R^+} satisfiability testing algorithm. \mathcal{ALCH}_{R^+} is of interest because it provides useful additional expressive power and, although its satisfiability problem is EXPTIME-complete, the algorithm is relatively simple and is amenable to optimisation.

1 Introduction

The importance of transitively closed roles in Description Logics (DLs) has long been recognised [PL94], particularly in domains which are concerned with physically composed objects, for example in medicine [HRG96] and engineering [Sat95]. The logic \mathcal{ALC}_+ [Baa90] supports complete reasoning about roles and their transitive closures by extending \mathcal{ALC} [SSS91] with union, composition and transitive closure role forming operators but, unfortunately, has a satisfiability problem which is EXPTIME-complete.

The \mathcal{ALC}_{R^+} and \mathcal{ALC}_\oplus DLs were investigated in the hope that a more restricted form of transitive role might lead to a satisfiability problem in a lower complexity class [Sat96]. \mathcal{ALC}_{R^+} extends \mathcal{ALC} by allowing the use of transitive roles in concept expressions. In [Sat96] an algorithm for deciding the satisfiability of \mathcal{ALC}_{R^+} concept expressions is provided along with a proof of its soundness and completeness. It is also demonstrated that the complexity of the problem is PSPACE-complete, the same as for \mathcal{ALC} [DLNN95]. \mathcal{ALC}_\oplus extends \mathcal{ALC}_{R^+} to provide more expressive power by associating each non-transitive role R with a transitive super-role R^\oplus s.t. $(R^\oplus)^I \supseteq (R^+)^I$. The extension to the \mathcal{ALC}_{R^+} algorithm required for \mathcal{ALC}_\oplus is relatively minor but, although [Sat96] does not present a soundness and completeness proof for the extension, it is shown that the problem is EXPTIME-complete, the same as for \mathcal{ALC}_+ .

This paper describes \mathcal{ALCH}_{R^+} , a logic which generalises \mathcal{ALC}_\oplus by allowing the definition of a role hierarchy, and presents an appropriate extension to the \mathcal{ALC}_{R^+} algorithm. The \mathcal{ALC}_{R^+} soundness and completeness proof has also been extended [Hor97b] but is not presented here due to space restrictions. For the same reason, a familiarity with the usual Tarski style model theoretic semantics for \mathcal{ALC} is assumed [BHH⁺91].

As \mathcal{ALCH}_{R^+} is more general than \mathcal{ALC}_\oplus , but still less expressive than \mathcal{ALC}_+ , the complexity of its satisfiability problem is clearly also EXPTIME-complete. However it seems worthwhile to study this logic as it provides useful expressive power, allowing for example general inclusion axioms to be internalised in concept expressions, while having a satisfiability testing algorithm which is much simpler than that for \mathcal{ALC}_+ , and thus more amenable to optimisation [Hor97a].

2 The \mathcal{ALCH}_{R^+} Description Logic

The relationship between roles and their transitive orbits in \mathcal{ALC}_\oplus is equivalent to introducing a limited form of role hierarchy—given a set of role names \mathbf{R} and a set of transitive roles $\mathbf{R}_+ \subseteq \mathbf{R}$, the relationship between a role and its transitive orbit can be described by a role inclusion axiom of the form $R \sqsubseteq R^\oplus$ where $R \in \mathbf{R}$ and $R^\oplus \in \mathbf{R}_+$. \mathcal{ALCH}_{R^+} generalises \mathcal{ALC}_\oplus by allowing acyclic, but otherwise unrestricted, role inclusion axioms of the form $R \sqsubseteq S$, where $\{R, S\} \subseteq \mathbf{R}$. The semantics of the acyclic \sqsubseteq relation mean that it is reflexive (for all $R \in \mathbf{R}$, $R \sqsubseteq R$), antisymmetric (for any two roles R and S , $R \sqsubseteq S \wedge S \sqsubseteq R \Rightarrow R = S$) and transitive ($R \sqsubseteq P \wedge P \sqsubseteq S \Rightarrow R \sqsubseteq S$). The \sqsubseteq relation therefore defines a partial ordering in \mathbf{R} which, like the concept subsumption relation, can be stored as a hierarchy, a directed acyclic graph in which each role is linked to its most specific super-roles and sub-roles.

If \mathbf{R} is the set of all role names, $\mathbf{R}_+ \subseteq \mathbf{R}$ is the set of transitive roles names and \sqsubseteq is the inclusion relation which defines a partial ordering in \mathbf{R} , then as well as being correct for \mathcal{ALC} concept expressions, an \mathcal{ALCH}_{R^+} interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ must satisfy the additional

conditions:

1. if $\langle d, e \rangle \in R^{\mathcal{I}}$ and $\langle e, f \rangle \in R^{\mathcal{I}}$ and $R \in \mathbf{R}_+$, then $\langle d, f \rangle \in R^{\mathcal{I}}$
2. if $R \sqsubseteq S$, then $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$

3 The Expressiveness of \mathcal{ALCH}_{R+}

\mathcal{ALCH}_{R+} allows complex role hierarchies to be established. Consider Figure 1, for example, which shows a fraction of the role hierarchy from a medical terminology model developed as part of the GALEN project [RH97], with the notation R_+ being used to denote that R is a transitive role. \mathcal{ALCH}_{R+} is able to capture the knowledge that part-whole relations in the GALEN model are subdivided into structural roles and process roles, and that the structural roles are further subdivided into *HasDivision*, *isMadeOf* and *hasLayer*.

This is still strictly less expressive than \mathcal{ALC}_+ . \mathcal{ALC}_+ can simulate a primitive role hierarchy by using role disjunction: the role *StructuralPartitiveAttribute* can be represented in \mathcal{ALC}_+ by the role expression $(\text{StructuralPartitiveAttribute} \sqcup \text{HasDivision} \sqcup \text{isMadeOf} \sqcup \text{hasLayer})^+$, capturing the knowledge that *StructuralPartitiveAttribute* is a transitive super-role of *HasDivision*, *isMadeOf* and *hasLayer*. Unlike \mathcal{ALCH}_{R+} , however, \mathcal{ALC}_+ can also simulate a non-primitive role hierarchy. It can, for example, represent a role such as *ancestor* with the expression parent^+ , capturing the knowledge that *ancestor* is exactly equal to the transitive closure of *parent*.

Unlike \mathcal{ALC}_{\oplus} , \mathcal{ALCH}_{R+} 's role hierarchy also enables internalisation [Baa90] to be used to test satisfiability w.r.t. a terminology \mathcal{T} which contains a set of general concept inclusion axioms (GCIs). If \mathcal{T} contains the axioms $C_1 \sqsubseteq D_1, \dots, C_n \sqsubseteq D_n$, where $C_1, \dots, C_n, D_1, \dots, D_n$ are arbitrary concept expressions, the satisfiability of a concept expression A w.r.t. \mathcal{T} can be tested by:

1. Forming the GCIs into a single concept expression $G \doteq (D_1 \sqcup \neg C_1) \sqcap \dots \sqcap (D_n \sqcup \neg C_n)$;
2. Defining a role $T \in \mathbf{R}_+$ s.t. $\forall R \in \mathbf{R}. R \sqsubseteq T$;
3. Testing the satisfiability of $A \sqcap G \sqcap \forall T.G$.

Although the satisfiability problems for both logics are EXPTIME-complete, the additional expressive power of \mathcal{ALC}_+ is manifested in a more complex satisfiability testing algorithm which requires both reasoning about role expressions¹ and a more sophisticated cycle detection (blocking) mechanism in order to differentiate between cycles which lead to a valid model and those which do not. The algorithm for \mathcal{ALCH}_{R+} on the other hand does not need to consider role expressions, and cycle detection is straightforward as all cycles lead to valid

¹A relatively efficient mechanism for dealing with this problem using finite state automata is suggested in [Baa90].

models. This simplicity makes the algorithm amenable to a range of optimisation techniques which greatly enhance its performance in realistic applications: the optimised algorithm has been used in the FaCT system [FaC] and its effectiveness has been demonstrated by classifying a large medical terminology knowledge base which includes both a complex role hierarchy and numerous transitive roles [Hor97a].

4 A Tableau Algorithm for \mathcal{ALCH}_{R+}

Like other tableau algorithms, the \mathcal{ALCH}_{R+} algorithm tries to prove the satisfiability of a concept expression D by demonstrating a model of D —an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ s.t. $D^{\mathcal{I}} \neq \emptyset$. The model is represented by a tree whose nodes correspond to individuals, each node being labelled with a set of \mathcal{ALCH}_{R+} -concepts. When testing the satisfiability of an \mathcal{ALCH}_{R+} -concept D these sets are restricted to subsets of $\text{sub}(D)$, where $\text{sub}(D)$ is the closure of the subconcepts of D . The soundness and completeness of the algorithm can be proved by showing that the tree it creates corresponds to a tableau for D and that D is satisfiable if and only if there exists a tableau for D but, due to space restrictions, the proof is not presented here. As usual, it is assumed that D is in negation normal form, i.e., that negations are applied only to primitive concepts. This can easily be achieved using a combination of DeMorgan's laws and the identities $\neg \exists R.C = \forall R.\neg C$ and $\neg \forall R.C = \exists R.\neg C$.

The algorithm builds a tree where each node x of the tree is labelled with a set $\mathcal{L}(x) \subseteq \text{sub}(D)$ and may, in addition, be marked *satisfiable*. The tree is initialised with a single node x_0 , where $\mathcal{L}(x_0) = \{D\}$, and expanded either by extending $\mathcal{L}(x)$ for some leaf node x or by adding new leaf nodes. For a node x , $\mathcal{L}(x)$ is said to contain a *clash* if, for some concept C , $\{C, \neg C\} \subseteq \mathcal{L}(x)$ or $\perp \subseteq \mathcal{L}(x)$. $\mathcal{L}(x)$ is called a *pre-tableau* if it is clash-free and contains no unexpanded conjunction or disjunction concepts. Note that \emptyset is a pre-tableau.

Edges of the tree are either unlabelled or labelled R for some role name R occurring in $\text{sub}(D)$. Unlabelled edges are added when expanding $A \sqcup B$ concepts in $\mathcal{L}(x)$ and are the mechanism whereby the algorithm explores the alternative expansions offered by disjunctions. Labelled edges are added when expanding $\exists R.A$ terms in $\mathcal{L}(x)$ and correspond to relationships between pairs of individuals.

A node y is called an *R-successor* of a node x if there is an edge from x to y labelled R ; y is called a \sqcup -*successor* of x if there is a path, consisting of unlabelled edges, from x to y . A node x is an *ancestor* of a node y if there is a path from x to y regardless of the labelling of the edges. Note that both the \sqcup -successor and ancestor relations are reflexive: nodes are connected to themselves by the empty path.

The algorithm initialises a tree \mathbf{T} to contain a single

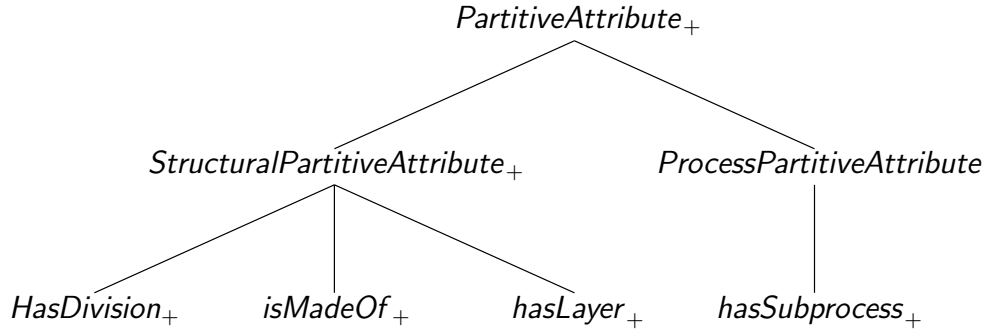


Figure 1: A fraction of the GALEN role hierarchy

node x_0 , called the *root* node, with $\mathcal{L}(x_0) = \{D\}$. \mathbf{T} is then expanded by repeatedly applying the rules from Table 1 until either the root node is marked *satisfiable* or none of the rules is applicable. If the root node is marked *satisfiable* then the algorithm returns *satisfiable*; otherwise it returns *unsatisfiable*.

5 Conclusion

\mathcal{ALCH}_{R+} usefully extends the expressive power of \mathcal{ALC}_{R+} and \mathcal{ALC}_{\oplus} by supporting both a primitive role hierarchy and the internalisation of GCIs.

The complexity of subsumption reasoning in \mathcal{ALCH}_{R+} is EXPTIME-complete, the same as for \mathcal{ALC}_{+} , but the algorithm is much simpler and is amenable to a range of optimisation techniques. Although the underlying complexity means that intractable problems may still arise, in practice the algorithm has demonstrated acceptable performance in realistic applications.

The complexity class of a problem is a relatively coarse grained measure and, while not underestimating the importance of theoretical complexity results, experience with \mathcal{ALCH}_{R+} suggests that consideration may also need to be given to the ‘practical’ complexity of subsumption testing algorithms.

References

- [Baa90] F. Baader. Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles. Research Report RR-90-13, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), 1990.
- [BHH⁺91] F. Baader, H.-J. Heinsohn, B. Hollunder, J. Muller, B. Nebel, W. Nutt, and H.-J. Profitlich. Terminological knowledge representation: A proposal for a terminological logic. Technical Memo TM-90-04, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), 1991.
- [DLNN95] F. M. Donini, M. Lenzerini, D. Nardi, and W. Nutt. The complexity of concept languages. Research Report RR-95-07, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), 1995.
- [FaC] The FaCT system. Available from www.cs.man.ac.uk/~horrocks/FaCT.
- [Hor97a] I. Horrocks. Optimisation techniques for expressive description logics. Technical Report UMCS-97-2-1, University of Manchester, Department of Computer Science, February 1997.
- [Hor97b] I. Horrocks. *Optimising Tableau Decision Procedures for Description Logics*. PhD thesis, University of Manchester, 1997. To appear.
- [HRG96] I. Horrocks, A. Rector, and C. Goble. A description logic based schema for the classification of medical data. In F. Baader, M. Buchheit, M.A. Jeusfeld, and W. Nutt, editors, *Reasoning about structured objects: knowledge representation meets databases. Proceedings of the 3rd Workshop KRDB’96*, pages 24–28, 1996.
- [PL94] L. Padgham and P. Lambrix. A framework for part-of hierarchies in terminological logics. In J. Doyle, E. Sandewall, and P. Torasso, editors, *Principals of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR’94)*, pages 485–496. Morgan-Kaufmann, 1994.
- [RH97] A. Rector and I. Horrocks. Experience building a large, re-usable medical ontology using a description logic with transitivity and concept inclusions. In *Proceedings of the Workshop on Ontological Engineering, AAAI*

\sqcap -rule:	If x is a leaf of \mathbf{T} , $\mathcal{L}(x)$ is clash-free, $A \sqcap B \in \mathcal{L}(x)$ and $\{A, B\} \not\subseteq \mathcal{L}(x)$ then $\mathcal{L}(x) \longrightarrow \mathcal{L}(x) \cup \{A, B\}$
\sqcup -rule:	If x is a leaf of \mathbf{T} , $\mathcal{L}(x)$ is clash-free, $A \sqcup B \in \mathcal{L}(x)$, $A \notin \mathcal{L}(x)$ and $B \notin \mathcal{L}(x)$ then create two \sqcup -successors y, z of x with: $\mathcal{L}(y) = \mathcal{L}(x) \cup \{A\}$ $\mathcal{L}(z) = \mathcal{L}(x) \cup \{B\}$
\exists -rule:	If x is a leaf of \mathbf{T} and $\mathcal{L}(x)$ is a pre-tableau then for each $\exists R.A \in \mathcal{L}(x)$ do: 1. $\ell_{Rx} := \{A\} \cup \{C \mid \forall S.C \in \mathcal{L}(x) \text{ and } R \sqsubseteq S\}$ $\quad \cup \{\forall S.C \mid \forall S.C \in \mathcal{L}(x), S \in \mathbf{R}_+ \text{ and } R \sqsubseteq S\}$ 2. If for some ancestor w of x , $\ell_{Rx} \subseteq \mathcal{L}(w)$ then create an R -successor y of x with $\mathcal{L}(y) = \emptyset$ 3. Otherwise create an R -successor y of x with $\mathcal{L}(y) = \ell_{Rx}$
SAT-rule:	If a node x is not marked <i>satisfiable</i> , and one of the following is true of x : 1. $\mathcal{L}(x)$ is a pre-tableau containing no concepts of the form $\exists R.A$ 2. $\mathcal{L}(x)$ is a pre-tableau and all R -successors of x are marked <i>satisfiable</i> 3. $\mathcal{L}(x)$ is not a pre-tableau and some \sqcup -successor of x is marked <i>satisfiable</i> then mark x <i>satisfiable</i>

Table 1: Tableau expansion rules for \mathcal{ALCH}_{R^+}

Spring Symposium (AAAI'97). AAAI Press, Menlo Park, California, 1997. To appear.

- [Sat95] U. Sattler. A concept language for engineering applications with part-whole relations. In *Proceedings of the International Conference on Description Logics—DL'95*, pages 119–123, Roma, Italy, 1995.
- [Sat96] U. Sattler. A concept language extended with different kinds of transitive roles. In G. Görz and S. Hölldobler, editors, *20. Deutsche Jahrestagung für Künstliche Intelligenz*, number 1137 in Lecture Notes in Artificial Intelligence, pages 333–345. Springer Verlag, 1996.
- [SSS91] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48:1–26, 1991.