Optimising Propositional Modal Satisfiability for Description Logic Subsumption

Ian Horrocks\(^1\) and Peter F. Patel-Schneider\(^2\)

\(^1\) University of Manchester, Manchester, UK (horrocks@cs.man.ac.uk)
\(^2\) Bell Labs Research, Murray Hill, NJ, U.S.A. (pfps@research.bell-labs.com)

**Abstract.** Effective optimisation techniques can make a dramatic difference in the performance of knowledge representation systems based on expressive description logics. Because of the correspondence between description logics and propositional modal logic many of these techniques carry over into propositional modal logic satisfiability checking. Currently-implemented representation systems that employ these techniques, such as FaCT and DLP, make effective satisfiable checkers for various propositional modal logics.

1 Introduction

Description logics are a logical formalism for the representation of knowledge about individuals and descriptions of individuals. Description logics represent and reason with descriptions similar to "all people whose friends are both doctors and lawyers" or "all people whose children are doctors or lawyers or who have a child who has a spouse". The computations performed by systems that implement description logics are based around determining whether one description is more general than (subsumes) another. There have been various schemes for computing this subsumption relationship, depending on the expressive power of the description logic and the degree of completeness of the system. As description logic systems perform numerous subsumption checks in the course of their operations, they need to have a highly-optimised subsumption checker.

Recent work [16] has shown that determining subsumption in expressive description logics is equivalent to determining satisfiability of formulae in propositional modal or dynamic logics. Thus one part of a system that implements a description logic is equivalent to a satisfiability checker for a propositional modal or dynamic logic. Several description logic systems have been built for such description logics, and thus include what is essentially a satisfiability checker, including KRIS [2] and CRACK [5]. These two systems have incorporated a number of optimisations to achieve better performance of their subsumption checkers.

Description logic systems are also optimised in other ways. In particular, their operations are optimised to avoid the potentially-costly subsumption checks whenever possible. There are also other optimisations to subsumption possible in description logic systems, having to do with the nature of the representation of knowledge in a description logic, but these have little or nothing to do with optimising propositional modal satisfiability.

© Springer-Verlag Berlin Heidelberg 1998
We have built two systems that explore the optimisations required to build an expressive description logic system, namely FaCT [11], a full description logic system, and DLP [14], an experimental system providing only a limited description logic interface. FaCT is available at http://www.cs.man.ac.uk/~horrocks; DLP is available at http://www-db.research.bell-labs.com/user/pfps.

We have incorporated a range of known, adapted and novel optimisation techniques into the subsumption checkers for these two systems. The optimisation techniques include: lexical normalisation, semantic branching search, boolean constraint propagation, dependency directed backtracking, heuristic guided search and caching.

These optimisations techniques make a drastic difference to the performance of the overall system. As evidence, KRIS is not able to load a modified version of the GALEN knowledge base because it gets stuck trying to determine one of the thousands of subsumptions required to load the knowledge base. FaCT and DLP, which have higher levels of optimisation, are able to easily load this knowledge base, classifying over two thousand definitions in about two hundred seconds.

We have also performed experiments with both FaCT and DLP on several test suites of propositional modal formulae. The optimisations built into the two systems qualitatively change their behaviour on the test suites, indicating that the optimisations have considerable utility simply taken as optimisations for reasoning in propositional modal logics.

2 Background

FaCT and DLP are designed to build and maintain taxonomies of named concepts. Given a collection of definitions of named concepts and statements about these concepts, they determine the subsumption partial order for the named concepts. To do this they have to determine subsumption relationships between descriptions in a description logic.

The description logic that DLP implements is called \( \mathcal{ALC}_{R^+} \). FaCT implements a considerably more-expressive logic, but most of the satisfiability optimisations in FaCT are demonstrable in \( \mathcal{ALC}_{R^+} \). \( \mathcal{ALC}_{R^+} \) is built up from atomic concepts and two kinds of atomic roles, non-transitive roles and transitive roles. Concepts in \( \mathcal{ALC}_{R^+} \) are formed using the grammar

\[
A \mid \top \mid \bot \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \exists R.C \mid \forall R.C \mid \exists T.C \mid \forall T.C, \]

where \( A \) is an atomic concept, \( C \) and \( D \) are concept expressions, \( R \) is a non-transitive role, and \( T \) is a transitive role.

The semantics of \( \mathcal{ALC}_{R^+} \) is a standard extensional semantics, using an interpretation \( \mathcal{I} \) that is a pair \((\Delta^\mathcal{I}, \mathcal{I})\) consisting of a domain and a mapping from concepts to subsets of the domain and from roles to binary relations on the domain (transitive relations for transitive roles, of course). The semantics for concept expressions are given in Table 1. One concept then subsumes another if

\[1\] Throughout the paper, we will be using the syntax of description logics. To translate into the syntax of modal propositional logics, replace \( \forall R \) with \( \square_R \) and \( \exists R \) with \( \diamond_R \) and perform several other obvious replacements.