Combining Description Logics with Stratified Logic Programs in Knowledge Representation

Jianhua Chen
Computer Science Department
Louisiana State University
Baton Rouge, LA 70803-4020
jianhua@bit.csc.lsu.edu

Abstract. Hybrid knowledge representations that combine description logics with logic programs are considered. Previous works combine description logics with Horn logic programs. In this paper, the expressive power of such hybrid systems is extended by allowing the combination of function-free, non-recursive stratified logic programs with description logics. Two model-theoretic definitions for the semantics of the hybrid knowledge representation are presented. It is shown that the inference problem based on the second semantics is decidable. When the logic program is Horn, the two semantics defined in this paper coincide with the semantics in [4] with regard to the inference problem.

1. Introduction

The use of hybrid representations is an important research problem in knowledge representation and reasoning. Several recent papers [2,4,5,6] addressed various aspects of hybrid knowledge representations. In [6], a framework based on unification of constraint logic programming, annotated logic programming, and stable model semantics, was developed to handle the multiple modes of reasoning in hybrid knowledge bases. The work in [2] investigated the connections between description logics and predicate logics, and compared their relative expressiveness. Combinations of description logics with Horn logic programs were investigated [3,4]. In particular, it was shown in [4] that the inference problem is decidable in a hybrid system combining the description logic $ALCNR$ with a non-recursive Horn logic program.

A natural extension in similar direction is to consider the integration of description logics with stratified logic programs for knowledge representation. Stratified logic programs are extensions of Horn logic programs which allow certain restricted form of negations in the antecedent of program rules, and thus are more expressive. The ability to combine stratified programs and description logics within one hybrid system significantly enhances the system’s expressive power.

In this paper, we present such an extension. The proposed hybrid system combines the description logic $ALCNR$ with a stratified logic program. We present two model-theoretic semantics for the hybrid knowledge representation system: The preferred model semantics, and the preferred-canonical model semantics. We show that the inference problem under the preferred-canonical model semantics is decidable based on a straightforward application of the decidability result in [4].

Springer-Verlag Berlin Heidelberg 2000
2. Preliminaries

In the new hybrid representation, a knowledge base \( D = T, p \) consists of two components: a terminology \( T \) in the description logic \( ALCNR \), and a logic program which is function-free, non-recursive, and stratified. The concepts and roles from the terminology \( T \) can occur (positively) in the antecedents of rules in \( p \).

2.1. The Terminological Component

A description logic language contains a set of unary relations called concepts that represent sets of objects in the domain of discourse, and binary relations called roles that represent relationships between these objects. Composite formulas in a description logic are built from primitive concepts and roles by using a set of constructors in the logic. Here as in [4], the description logic component in a hybrid knowledge base can be any subset of the \( ALCNR \) language. Descriptions in \( ALCNR \) are built in the following way: Each primitive concept \( A \) is a concept description; the special concepts \( \top \) (truth) and \( \bot \) (falsity) are concept descriptions; let \( C \) and \( E \) be concept descriptions and let \( R \) be a role description, then \( C \land E \), \( C \lor E \), \( \neg C \), \( R \), \( C \cdot R \), \( (\exists n R) \) and \( (\forall n R) \) are concept descriptions; each role description \( R \) is of the form \( R_1 \land \ldots \land R_m \) where each \( R_j \) is a primitive role.

The general terminological part of a terminology \( T \) is a set of sentences in \( ALCNR \), where a sentence is either a concept definition, a concept inclusion, or a role definition. A concept definition is of the form \( C := E \) where \( C \) is a concept name and \( E \) is a concept description. A concept inclusion is of the form \( C \sqsubseteq E \) where both \( C \) and \( E \) are concept descriptions. A role definition is of the form \( P := R \) where \( P \) is a role name and \( R \) is a role description. We do allow recursive concept definitions. The assertional part of \( T \) is a set of ground atoms of the form \( C(a) \), \( R(a, b) \).

The meaning of a terminology \( T \) is determined by a model-theoretic semantics. We define an interpretation \( I \) to be a non-empty domain \( O \), a mapping from the set of constants in \( T \) to \( O \) such that \( a = b \) if \( aI = bI \), a mapping from each concept name \( C \) to a unary relation \( CI \) in \( O \), and a mapping from each role name \( R \) to a binary relation \( RI \) on \( O \). The mappings of \( I \) can be naturally extended to the composite descriptions in a straightforward way.

An interpretation \( I \) satisfies a concept instance \( C(a) \) if \( aI \in CI \). \( I \) satisfies a role instance \( R(a, b) \) if \( (aI, bI) \in RI \). \( I \) satisfies a concept definition \( C := E \) if \( CI = EI \), it satisfies a concept inclusion \( C \sqsubseteq E \) if \( CI \subseteq EI \), and it satisfies a role definition \( P := R \) if \( PI = RI \). \( I \) is a model of a terminology \( T \) if \( I \) satisfies each sentence in \( T \).

2.2. Stratified Logic Programs

A function-free (normal) logic program is a set of program rules of the form

\[
\begin{align*}
B_1(X_1) \quad & \ldots \quad B_m(X_m) \quad \neg C_1(Y_1) \quad \ldots \quad \neg C_n(Y_n) \quad A(X),
\end{align*}
\]

Here each \( B_i \), \( C_j \) and \( A \) is an atom, \( m \) and \( n \) \( \geq 0 \). The conjunction on the left hand side