Specifying Over-Constrained Problems in Default Logic

Abdul Sattar[1], Aditya K. Ghose[2], Randy Goebel[3]

[1] School of Comp. and Info. Technology, Griffith University, Brisbane, Australia, 4111. sattar@cit.gu.edu.au

[2] Knowledge Systems Group, Basser Department of Computer Science, University of Sydney, Syndey NSW 2006, Australia. aditya@cs.su.oz.au

[3] Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada, T6G 2H1. goebel@cs.ualberta.ca

Abstract. In the previous studies, it has been shown that the classical constraint satisfaction problem (CSP) is deductive in nature, and can be formulated as a classical theorem proving problem [1, 10]. Constraint satisfaction problems for which an assignment of values to all variables which satisfy all available constraints is not possible are referred to as *over-constrained problems.* This paper shows how computing partial solutions to over-constrained problems can be viewed as a default reasoning problem. We propose two methods for translating over-constrained problem specifications with finite domains to two different variants of default logic. We argue that default logic provides the appropriate level of abstraction for representing and analyzing over-constrained problem even if other methods are used for actually computing solutions.

1 Introduction

A *constraint satisfaction problem* (CSP) involves a set of *variables*, a *domain* of possible values for each variable, and a set of *constraints*, representing *acceptable* and *non-acceptable* relations over subsets of variables. A *solution* is an assignment of values to variables that satisfy all constraints. Constraint satisfaction problems for which an assignment of values to each variable that satisfies all constraints is not possible are called *Over-Constrained Problems (OCP).* If the domains of variables in an OCP (resp. CSP) are restricted to be *finite*, we obtain the class of Finite Over-Constrained Problems (FOCP) (resp. Finite Constraint Satisfaction Problems (FCSP))[4].

Over-constrained problems are ubiquitous in AI and arise due to conflicting constraints in a domain of reasoning such as design problems in which conflicting goals have to be achieved, diagnosis in which competing hypotheses explain the set of symptoms, schedule conflicts etc. This problem has been variously termed as *partial constraint satisfaction* [5], reasoning with *constraint hierarchies* [14], reasoning with hard and soft constraints [12], or *constraint relaxation* and preferences over relaxations in resolving conflicting schedules [4].

[4] Mackworth [10] specifies a finite constraint satisfaction problem in an identical manner

Logically, the notion of constraint satisfaction has been viewed as a *deductive* reasoning problem [1, 10]. It has been previously shown that classsical CSP can be formulated as a hypothetical reasoning problem using the THEORIST system [13]. We show that formulating constraint satisfaction as a nonmonotonic reasoning problem is general enough to cover OCP's in addition to classical CSP's. Intuitively, conflicting constraints could be treated as conflicting defaults in a default reasoning framework. Thus, finding a solution to a partial constraint satisfaction problem corresponds to computing an extension of a default theory.

A variety of default reasoning systems exist in the literature. In this paper we shall consider a restricted version of Reiter's default logic [11] as well as a recent variant called prerequisite-free constrained default logic (PfConDL) presented by Delgrande, Schaub and Jackson in [3]. We shall present translations from finite over-constrained problem specifications to default theories in each of these formalisms. We have earlier shown how default extension computation can be viewed as solving over-constrained problems [7]. The research presented in this paper thus completes the picture by presenting the reverse translation as well.

We maintain that default logic provides the appropriate level of abstraction for representing and analyzing over-constrained problems even if other techniques are used for actually computing solutions. The ability to specify such problems in a formal language with well-defined semantics has several practical advantages. These include semantically well-founded criteria for defining preference relations on constraints and solutions as well as methods for revising OCP specifications in a principles manner. Further, we believe that complexity results from the default reasoning area can suggest tractable classes of OCP's.

2 Over-constrained problems

Formally, a constraint satisfaction problem (CSP) specification consists of a finite set of variables $Var = \{X_1, \ldots, X_n\}$, each associated with a domain of discrete values, $d_1, \ldots, d_n$, and a set of constraints $Con = \{C_1, \ldots, C_m\}$. Each constraint is a relation defined on some subset of the set of variables. A constraint C_i consists of the *constraint-subset* $S_i = \{X_{i_1}, \ldots, X_{i_{j(i)}}\}$, where $S_i \subseteq X$, denoting the subset of the variables on which C_i is defined and the *relation* rel_i defined on S_i such that $rel_i \subseteq d_{i_1} \times \ldots \times d_{i_{j(i)}}$.

Formally, an *over-constrained problem* is a constraint satisfaction problem for which there is no assignment of values to all variables such that all the constraints are satisfied. The following example from [5] illustrates the idea.

Example 1. Consider a robot seeking to select matching shoes, shirts and slacks while getting dressed. It has two kinds of shoes (cordovans, sneakers), two kinds of shirts (green, white) and three kinds of slacks (denims, dress blue, dress grey). The only allowable combinations are: white shirts and cordovan shoes, cordovan shoes and gray dress slacks, sneakers and denim slacks, green shirts and dress gray slacks, white shirts and denim slacks and white shirts and dress blue slacks.

We shall formulate the problem as a CSP. We consider three variables: Shoes (S_1), Shirts (S_2), and Slacks (S_3). The corresponding domains are: