

Research Statement

Deborah East

The constraint satisfaction problem (CSP) is defined as a triple $(X, D_x : x \in X, C)$, where X is a finite set of variables. The domain D_x is a finite set of values which can be assigned to x . A finite set of constraints C restrict the values from D_x which can be assigned to x . A solution to a CSP is an assignment of values which does not violate any of the constraints.

Constraint satisfaction problems arise in almost all areas of computer science. Artificial intelligence, combinatorial algorithms, neural networks, operations research, planning and VLSI design is a partial list of disciplines where constraint satisfactions problems occur. A specific example of constraint programming is the layout of wires in VLSI design. Because constraint programming covers a vast area, general methods are needed.

I am interested in developing languages and designing solvers for constraint satisfaction problems. I developed a language which can be used as a programming front-end for satisfiability solvers. The logic is a modification of the logic of propositional schemata; we explicitly separate theories into data and program and use a version of closed world assumption to define the semantics.

I extended the logic by adding constraints on the cardinality of sets. Constraints on the cardinality of sets facilitate the modeling of constraint satisfaction problems by exactly and concisely representing constraints. I also extended the logic by adding Horn clauses to the constraints. Problems which include transitive closure are modeled concisely using Horn clauses.

I have designed and implemented DATALOG with Constraints (DC). The DC grounder maintains the structure of the cardinality constraints as well as differentiating between constraints and Horn clauses in the grounded theories. The DC solver is designed to take advantage of the structure of cardinality constraints and Horn clauses. I have compared the performance of DC with satisfiability solvers and stable logic programming on combinatorial problems, graph problems and planning problems. The DC system is more efficient on many problems than the other systems tested. For all problems DC is competitive with the other systems. Papers detailing some of the results as well as the source code for implementations of the grounders and solvers are online and can be reached from my home page: "www.cs.uky.edu/~deast".

In the future, I intend to work on solvers for DC that use randomization techniques. Because of the difficulty of many constraint satisfaction problems, it is not always practical to use complete algorithms for finding solutions. Versions of DC which use restarts and some random choice may be more efficient for some constraint satisfaction problems than either complete algorithms or randomized satisfiability algorithms. I also intend to design and implement parallel versions of DC. In addition to further development of DC, I am interested in studying and developing simulation software.