Abstract

Constraint Logic Programming (CLP) has been traditionally applied to the modelling of complex problems, especially combinatorial problems, and to model knowledge bases. In this presentation, we focus on using CLP for program analysis and verification. First we consider the representation of program behavior: the rules and constraints of CLP provides for a natural specification of programs as a symbolic guarded transition system. The CLP execution model can then capture the symbolic traces of the underlying program, and these traces, in turn, divulge the properties that we seek. Secondly, we use the CLP formalism for the formal specification of complex properties of data structures. Here the CLP execution model can be used as a theorem-prover to dispense with the proof obligations arising from the program and its specifications. The traditional CLP execution model, however, is not automatically practical for these purposes. We shall present two refinements to CLP: one for reducing the number of symbolic traces that are needed to prove a property, and one to efficiently deal with data structure properties.