Programming Language Semantics: Theory, Practice and Applications

Qian Wang, Michael Nichols, and Gopal Gupta

In this project we study the formal semantics of programming languages and their applications. Specifically, we are interested in Horn logical semantics---denotational semantics expressed in Horn logic---of programming languages. There are many advantages that accrue from this change in notation:

1) Both the syntax and semantics specification, specified as logic programs, become executable, allowing for interpreters and compilers to be automatically obtained.

2) Given the logic programming basis, verification of program properties becomes possible. The Horn logical denotation of a program can be thought as an axiomatization of the solution of the problem the program solves.

3) A formal framework for resource conscious interpretation and compilation is obtained, non-deterministic semantics that model resources can be easily expressed.

4) A formal framework for language interoperability is also obtained; the problem of formal language translation can be seen as simply writing the executable semantics of one notation in terms of the syntax trees of the other.

The framework has been used to develop a number of applications ranging from verification of real-time systems, to developing filters for bioinformatics software, as well as translators for formal languages.

Both direct semantics and continuation semantics of imperative languages have been developed in the Horn logic notation, and used to automatically obtain interpreters and compilers (via partial evaluation). Currently, this Horn logical framework is being used to develop implementation infrastructure for domain specific languages, to obtain implementations automatically from architectural descriptions, and to develop a formal basis for specifying non-functional requirements in architecture description languages. Recently this framework has been applied to obtain target code for control systems specified using the SCR (software cost reduction) method. The code generation process is provably correct since given an SCR program, its target code is automatically obtained from the semantic specification of the SCR language via partial evaluation.

Reference: Qian Wang and G. Gupta, Large Provably Correct Code Generation for High Assurance Systems via Partial Evaluation. Proceedings of the International Symposium on Logic Programming Synthesis and Transformation (LOPSTR). 2003, Uppsala, Sweden.

G. Gupta. Horn Logic Denotations and Their Applications. In Logic Programming: The Next 25 Years. Springer Verlag. 1998. pp. 127-160.