Mechanized Operational Semantics

J Strother Moore
University of Texas at Austin, USA

I will show how we can formalize the semantics of a von Neumann programming language in a functional programming language and how we can reason about the functional language to prove theorems about programs in the von Neumann one. We will see that this approach allows one to execute programs in the von Neumann language. That is, the operational semantics has a dual use: as a simulation engine and as an axiomatic basis for code proofs. We will explore both partial and total correctness proofs. We will also see how various code proof styles (e.g., Floyd-Hoare inductive assertions) can be employed without the construction of any extra-logical machinery (e.g., a Hoare semantics or a verification condition generator).

The entire project will be carried out in a mechanized logic, namely, with the ACL2 theorem proving system. "ACL2" stands for "A Computational Logic for Applicative Common Lisp". The programming language supported by ACL2 is a very large, functional (or "applicative") subset of ANSI Standard Common Lisp. The semantics of that programming language is formalized via axioms and definitions within a first-order mathematical logic. A mechanized theorem proving environment supports the discovery of proofs in the theory. We use the name ACL2 for all three of these aspects: the programming language, the logical theory, and the theorem prover.

The von Neumann language used to illustrate this will be a very simplified version of the Sun Java Virtual Machine. An elaborate and accurate model of the JVM will be exhibited, but the detailed examples presented will be for a toy JVM that supports arithmetic, control, and method invocation only.

I anticipate the following sequence of lectures:

(1) Overview of ACL2 and its use to model the JVM. The lecture will conclude with some examples of code proofs of JVM bytecode programs.

(2) An operational semantics for a toy JVM, the development of a compiler from a simple programming language to the bytecode of that toy machine, executing programs on the machine.

(3) Proving programs correct directly from the operational semantics.

(4) The inductive assertion approach [2]

(5) The equivalence of the various methods [3,5]

If time permits we will explore elaborations of the operational semantics of the toy JVM towards the actual JVM by, for example, modeling the heap and multiple threads [1,4]

Reading all of [1], my Marktoberdorf 2002 lecture notes, will give the students an excellent background. However, I intend in these lectures to focus on a much simpler JVM model so we can explore code proof styles. But [1] shows how we use ACL2 to model practical von Neumann languages.

References

1. J. Strother Moore. *Proving Theorems about Java and the JVM with ACL2*. In Models, Algebras and Logic of Engineering Software, Summer School Marktoberdorf 2002; M. Broy and M. Pizka (eds); IOS Press; pp 227–290; 2003.
   `http://www.cs.utexas.edu/users/moore/publications/marktoberdorf-02/main.pdf`
2. J. Strother Moore. *Inductive Assertions and Operational Semantics*, In Proc of the CHARME 2003, D. Geist (ed); LNCS 2860; pp 289–303; Springer; 2003.
   `http://www.cs.utexas.edu/users/moore/publications/trecia/long.pdf`
3. J. Strother Moore and S. Ray. *Proof Styles in Operational Semantics*. In Proc of the Formal Methods in Computer-Aided Design (FMCAD 2004); A. J. Hu and A. K. Martin (eds); LNCS 3312; pp 67–81; Springer; 2004.
   `http://www.cs.utexas.edu/users/sandip/publications/proof-styles/proof-styles.pdf`
4. J. Strother Moore and H. Liu. *Java Program Verification via a JVM Deep Embedding in ACL2*. In Proc of the TPHOLS 2004, K. Slind, A. Bunker, and G. Gopalakrishnan (eds); LNCS 3223; pp 184–200; Springer; 2004.
   `http://www.cs.utexas.edu/users/moore/publications/tphol.5-14-06.pdf`
5. J. Strother Moore, S. Ray, W. Hunt, and J. Matthews. *A Mechanical Analysis of Program Verification Strategies*. (submitted for publication, 2007)
   `http://www.cs.utexas.edu/users/moore/publications/mech-analysis-of-verif-strategies.pdf`