Assignment Calculus:
A Pure Imperative Reasoning Language

A dissertation by Marc Bender

Abstract

What is a pure imperative language? A careful attempt to answer this question leads to many interesting questions about programming languages. This dissertation presents and pursues one possible definition: an imperative language is one whose operators are fundamentally *referentially opaque*; in simple terms, they make *substitution* problematic. In particular, we develop a new language, Assignment Calculus (***AC***), which we claim is a *core language* for imperative reasoning.

We begin the dissertation with a primarily philosophical, but example-driven, discussion of the above definition of pure imperative language. We also give a definition of a *reasoning language*, which we identify by several desirable properties that such a language should have. The *principle of substitutivity* (following Leibniz), *referential opacity* (Quine), and *intensionality* (Carnap) are introduced and discussed. Starting with some natural-language examples, we show the usefulness of Richard Montague's *intension* operator for handling certain types of problems, and then demonstrate that these problems are inherent to imperative programming languages. In fact we go much further and posit that intension — along with its dual, extension — are *fundamental parts of imperative reasoning.*

The main subject, our pure imperative reasoning language ***AC***, is introduced next. The formal syntax and operational semantics of ***AC*** is given. We define and derive some of its important properties. Next, the compositional denotational semantics of ***AC*** terms is given. Of note here is the interpretation of the domain of *possible worlds* or *states* as a *reflexive domain*; this allows for the storage of *procedures* in the state. A term rewriting system for ***AC*** is then presented.

The central result of the dissertation follows: we bind these three interpretations together by proving their equivalence.

Finally, taking ***AC*** as a starting point, we explore various extensions and variants. The core of ***AC***, when slightly enriched, is shown to be a self-contained Turing complete language. This bolsters our claim that ***AC*** is a core language for pure imperative reasoning.