

A Rewriting Semantics for Type Inference

George Kuan
University of Chicago
January 5, 2007

Abstract

When students first learn programming, they often rely on a simple operational model of a program's behavior to explain how particular features work. Because such models build on their earlier training in algebra, students find them intuitive, even obvious. Students learning type systems, however, have to confront an entirely different notation with a different semantics that many find difficult to understand.

In this work, I begin to build the theoretical underpinnings for treating type checking in a manner like the operational semantics of execution. Intuitively, each term is incrementally rewritten to its type. For example, each basic constant rewrites directly to its type and each lambda expression rewrites to an arrow type whose domain is the type of the lambda's formal parameter and whose range is the body of the lambda expression which, in turn, rewrites to the range type.

This is a placeholder document for the final version of George Kuan's Master's Thesis.