Theorems for free!

September 10, 2017

Philip Wadler 1989. Published at ICFP.

1 Introduction

Main idea of the paper is that every polymorphic function satisfies a theorem that we can derive from it. Every function of that type satisfies the theorem too.

The result that allows this is the *parametricity* result - it depends on paramteric polymorphism and Reynolds' abstraction theorem (terms evaluated in related environments yield related values). So types can be read as relations.

1.1 Hindley Milner vs System F

The paper uses **System F** as its type system, in which we must specify the types of bound variables explicitly (unlike in Hindley /Milner, where types can be inferred). System F is basically simply typed lambda calculus with universal quantification over types. Hindley/Milner is System F restricted to use type inference, so System F can be described as "more powerful" than Hindley/Milner. Also, if a function has a polymorphic type then type applications must be explicitly indicated, via subscripting. For example the instance of $r : \forall X.X^* \to X^*$ at type A is written $r_A : A^* \to A^*$.

Every program in Hindley Milner can be translated to one in System F by modifying the type inference algorithm to decorate programs with the appropriate type information. The opposite direction is not possible.

Both systems satisfy the strong normalisation property - that every term has a normal form and every reduction sequence leads to this normal form. Therefore there is no fixpoint operator in either system (It can be added as a a primitive, but this weakens the parametricity theorem).

Every recursive function that can be proved total in second order Peano arithmetic (such as Ackerman's function) can be written as a term in System F. This excludes interpreters for most languages.

(frame models?)