A Realizability Model for Impredicative Hoare Logic*

Gregory Malecha

March 31, 2008

This paper aims to address a common short-coming of much of the dependent typing work which we have seen so far; namely the difficulty of dealing with side-effects. In the language presented, an already rich dependently typed language is extended with indexed monadic types in the style of Hoare inference rules in order to express additional pre- and post-conditions of code. By allowing explicit references to the heap in the logic/types, side-effects of statements can be expressed and checked in a similar way to more standard type systems.

One major difficulty which arises from the addition of the heap is the need to address local heap computation in a modular way. This allows the programmer to reason about self-contained abstractions and compose their types in useful ways. To this end, separation logic[1] is employed to reason about disjunctive portions of the heap in a modular fashion. This is known as the *small footprint* approach and, in addition to making the logic more convenient, allows the logic to reason about pointer aliasing.

The mathematics behind much of the theory relies heavily on category and domain theory, so the presentation will focus primarily on type semantics in relation to separation logic and Hoare logic. I motivate the work by several applications presented throughout the paper and in previous work by Reynolds on separation logic in general[1].

References

[1] John Reynolds. Separation logic: A logic for shared mutable data structures. In *IEEE Symposium on Logic in Computer Science*, July 2002.

*Rasmus Lerchedahl Petersen, Lars Birkedal, Aleksandar Nanevski, Greg Morrisett