Implementing Intensional Computation

Jose Vergara[1]*

Centre for Quantum Computing and Intelligent Systems, School of Software,
University of Technology, Sydney `Jose.A.VergaraMedina@student.uts.edu.au`

∗ Joint work with Barry Jay

Abstract machines mediate between formal calculi and actual implementations. There are many abstract machines for lambda calculi, such as Landin's SECD machine, Curien's Categorical Abstract machine (CAM) and Krivine's machine, all built around the manipulation of closures.

However, these machines are not well suited to support intensional computation, in the style of pattern calculus, factorisation calculus or boa calculus, since there is no obvious way in which to factorise a closure. One alternative is to focus on combinators, in the style of Turner's Miranda, but then the closures are gone.

Our solution is to implement both closures and combinators in a new abstract machine, the At-Abstract machine or ATAM, whose structure is inspired by boa calculus.

The machine is developed in three stages to support: lambda calculus; querying of combinators (including data structures); and querying of closures. The machine for lambda calculus is a variant of the SECD machine. Querying of combinators requires support for intensional operators for factorisation and equality, and that partially applied operators be among the values. Querying of closures requires machinery for converting lambda abstractions to combinators, including reduction under the lambda and support for at-terms.

We have built an interpreter for the ATAM which has been used to check the examples to be presented in the talk.