Automated reasoning under the theory $\mathcal{H}^*$
(thesis abstract)

Fritz Obermeyer

February 23, 2009

In this thesis I study four extensions of untyped λ-calculi all under the maximally coarse semantics of the theory $\mathcal{H}^*$ (observable equality), and implement a system for reasoning about and storing abstract knowledge expressible in languages with these extensions. The extensions are:

(1) a semilattice operation $\mathbf{J}$, the join w.r.t the Scott ordering;
(2) a random mixture $\mathbf{R}$ for stochastic λ-calculus;
(3) a computational comonad $\langle \mathsf{code}, \mathsf{apply}, \mathsf{eval}, \mathsf{quote}, \{-\}\rangle$ for Gödel codes modulo provable equality; and
(4) a Π^1_1-complete oracle $\mathbf{O}$.

I develop three languages from combinations of these extensions. The syntax of these languages is always simple: each is a finitely generated combinatory algebra. The semantics of these languages are various fragments of Dana Scott's Dinfty models. Although the languages use ideas from the theory of computer programming languages, they have no operational semantics and do not describe programs.

The first language, SKJ , extends combinatory algebra with a join operation, with respect to the information ordering on terms. I show that an interpretation of types-as-closures reflects from D_∞ down to this definable fragment. The main theorem for SKJ is that simple types are definable as closures. The resulting type theory is very rich (ω-order polymorphism, dependent types, a type-of-types, power-types, quotient types) but logically unsound (every type is inhabited). However I demonstrate that this type system provides an expressive interpretation of type-as-symmetry and join-as-ambiguity.

The second language, SKRJ , extends combinatory algebra with a random bit, then with join. I show that SKRJ provides semantics for a monadic λ-calculus for convex sets of probability distributions (CSPDs). This follows from our choice of join distributing over randomness –the opposite of what one would expect from the usual interpretation of join-as-concurrency. I conjecture that a weakened definable-types theorem also holds in SKRJ , and provide some evidence to this effect. I also show that, in case simple types are definable, the monadic CSPD types are polymorphic in the simple definable types.

The third language, SKJO , extends combinatory algebra with join, a comonadic type of codes, and a carefully defined Π^1_1-complete oracle, so that exactly the Δ^1_1-predicates about SKJ -terms are realized as total terms of type $\mathsf{code} \to \mathsf{bool}$. This language is sufficient to represent all of predicative mathematics, and can thus serve to represent virtually any kind of abstract knowledge. As an example, I formulate the statement and proof of termination in Gödel's $\top$ in the language SKJO (and am working towards getting the proof to formally verify).

The final component of this thesis is a system, Johann, for automated reasoning about equality and order in the above languages. Johann was used to formally verify many of the theorems in this thesis (and even conjecture some simple theorems).

The general design focus is on efficient knowledge representation, rather than proof search strategies. Johann maintains a database of all facts about a set of (say 10k) objects, or terms-modulo-equivalence. The database evolves in time by randomly adding or removing objects. Each time an object is added, the database is saturated with facts using a forward chaining algorithm.

A specific design goal is to be able to run Johann for long periods of time (weeks) and accumulate useful knowledge, subject to limited memory. This requires statistical analysis of a corpus of interest (e.g. the set of problems to be verified in this thesis), statistical search for missing equations (from our Σ^0_1 approximation to a Π^0_2-complete theory), and careful choice of sampling distributions from which to draw objects to add-to and remove-from the database. The (add,remove) pair of distributions is chosen to achieve a *detailed balance* theorem, so that, at steady state, Johann (provably) remembers simple facts relevant to the corpus.