Correctness of Effect-based Program Transformations

Martin Hofmann
Universität München, Germany

We consider a type system capable of tracking reading, writing and allocation in a higher-order language with dynamically allocated references.

We give a denotational semantics to this type system which allows us to validate a number of effect-dependent program equivalences in the sense of observational equivalence. An example is the following:

$$x \;=\; M;\; y \;=\; M;\; N(x,y) \qquad \text{is equivalent to} \qquad x \;=\; M;\; N(x,x)$$

provided that M does not read from memory regions that it writes to and moreover does not allocate memory that is encapsulated in the values of x and y.

Here x can be a higher-order function or a reference or a combination of both.

The two sides of the above equivalence turn out to be related in the denotational semantics which implies that they are observationally equivalent, ie can be replaced by one another in any (well-typed) program.

On the way we learn popular techniques such as parametrised logical relations, regions, admissible relations, etc which belong to the toolbox of researchers in principles of programming languages.

The lectures are based on joint work with Nick Benton, Andrew Kennedy, Lennart Beringer.