State and Effect Logics for Deterministic, Non-deterministic, Probabilistic and Quantum Computation

Robin Adams and Bart Jacobs

Institute for Computing and Information Sciences, Radboud University Nijmegen, The Netherlands
robina@cs.ru.nl, bart@cs.ru.nl

We present a type theory that is suitable for reasoning about computations in four very different settings: deterministic, non-deterministic, probabilistic, and quantum computation.

This is possible because, in all cases, a computation can be modelled as a state transformer that transforms the state before the computation into the state after. It can also be modelled as a predicate transformer that transforms a predicate on the output into the weakest precondition on the input. There is a remarkable correspondence between this picture and the two pictures of quantum theory favored by Schrödinger (who described physical processes in terms of states) and Heisenberg (who used effects) [1].

This correspondence seems to be more than just a coincidence. All four forms of computation can be modelled using three categories in the following arrangement, which we call a state-and-effect triangle:

\[
\begin{array}{ccc}
\text{EMod}[-,M] & \rightarrow & \text{Conv}[-,M] \\
\downarrow & & \downarrow \\
\text{B} & \leftarrow & \text{S}
\end{array}
\]

In this structure:

• \(\text{B}\) is a category whose objects represent types and whose arrows represent computations;
• \(\text{EMod}\) is the category of effect modules over an effect monoid \(M\), whose elements represent the scalars (probabilities or truth values);
• \(\text{Conv}\) is the category of convex sets over the same effect monoid ([3], see also [2]);
• \(P\) and \(S\) (for ‘predicate’ and ‘state’) are functors which preserve the relevant structure of \(\text{B}\) (see below). \(PA\) is the effect module of predicates over the type \(A\), and \(SA\) is the convex set of all states of type \(A\).
• There is an adjunction between \(\text{EMod}^{\text{op}}\) and \(\text{Conv}\) formed by ‘homming’ into \(M\) ([2]).
• a natural transformation \(\models : S \rightarrow \text{EMod}[-,M]\). Given a state \(\omega \in SA\) and a predicate \(\phi \in PA\), the value \(\omega \models \phi\) is an element of \(M\) that represents the probability or truth value that \(\omega\) satisfies \(\phi\). In the case of deterministic computation, this will be a Boolean; in our other three examples, it will be a probability in \([0,1]\).

The four forms of computation are handled by the following four choices of the category \(\text{B}\):

The category \(\text{Set}\) for deterministic computation, where we model a computation as the function which, given an input
The category $Kl(P_*)$, the Kleisli category of the non-empty powerset monad $P_*$, for non-deterministic computation, where we model a computation as the function which maps an input to the set of possible outputs.

The category $Kl(D)$, the Kleisli category of the distribution monad $D$, for probabilistic computation, where we model a computation as the function which maps an input to the probability distribution of its outputs.

The category $CStar_{CPU}^{op}$, the opposite of the category of $C^*$-algebras and completely positive unital maps, for quantum computation.

Our aim for the future is to give a series of type theories of increasing strength, suitable for the four forms of computation. The final type theory will have these forms of judgement:

- $\Gamma \vdash M : A$, interpreted by the arrows of $B$;
- $\Gamma \vdash M = N : A$, interpreted by equality of arrows in $B$;
- $\Gamma \vdash \phi \text{ eff}$, which states that $\phi$ is an effect (predicate) in the context $\Gamma$. This will be interpreted by the elements of $P[\Gamma]$;
- $\Gamma \vdash \phi \leq \psi$, interpreted by the ordering in $P[\Gamma]$.

In this talk, we describe the first two of these theories, which capture the forms of reasoning common to all four models.

**Affine Type Theory** We first present an affine type theory: a type theory in which Contraction does not hold, but Weakening does. Thus, information may be destroyed, but may not be duplicated. (The ‘no-cloning theorem’ in quantum theory states that a state of a quantum system cannot be duplicated.)

The types of the affine type theory are given by: $\text{Type } A ::= X \mid I \mid A \otimes A \mid A + A$.

Affine type theory can be interpreted in all and only the affine categories $B$; i.e. symmetric monoidal categories in which the tensor unit is final, and which have coproducts which the tensor distributes over.

**Coproduct Logic** We extend the system to coproduct logic. Coproduct logic is a type theory that captures structures $(B, \text{Pred})$ where $B$ is an affine category, and $\text{Pred} : B \to \text Poset}^{op}$ is a functor which assigns, to each object of $B$, a poset of predicates or effects.

In $\text{Set}$ and $Kl(P_*)$, the predicates on a set $A$ are the subsets of $A$. In $Kl(D)$, the predicates on $A$ are the ‘fuzzy predicates’ in $[0, 1]^A$. In $CStar_{CPU}^{op}$, the predicates on $A$ are the effects on $A$, i.e. the elements $a \in A$ such that $0 \leq a \leq 1$.

**Future Work** We shall also discuss how comprehension and quotients may be added to these logics, and how this may provide a way of handling measurement with side-effects, as is so important for quantum computation.

**References**

