Effect-driven QuickChecking of compilers

How does one test a language implementation with QuickCheck (aka, property-based testing)? One approach is to generate programs following the grammar of the language. But in a statically-typed language such as OCaml too many of these candidate programs will be rejected as ill-typed by the type checker. As a refinement Pałka et al. propose to generate programs in a goal-directed, bottom-up reading up of the typing relation. We have written such a generator. However many of the generated programs has output that depend on the evaluation order, which is commonly under-specified in languages such as OCaml, Scheme, C, C++, etc. In this paper we develop a type and effect system for conservatively detecting evaluation-order dependence and propose its goal-directed reading as a generator of programs that are independent of evaluation order. We illustrate the approach by generating programs to test OCaml’s two compiler backends against each other and report on a number of bugs we have found doing so.