Dynamic languages such as JavaScript are more difficult to com pile than statically typed ones. Since no concrete type information is available, traditional compilers need to emit generic code that can handle all possible type combinations at runtime. We present an al ternative compilation technique for dynamically-typed languages that identifies frequently executed loop traces at run-time and then generates machine code on the fly that is specialized for the ac tual dynamic types occurring on each path through the loop. Our method provides cheap inter-procedural type specialization, and an elegant and efficient way of incrementally compiling lazily discov ered alternative paths through nested loops. We have implemented a dynamic compiler for JavaScript based on our technique and we have measured speedups of 10x and more for certain benchmark programs.

Unlike method-based dynamic compilers, our dynamic com piler operates at the granularity of individual loops. This design choice is based on the expectation that programs spend most of their time in hot loops. Even in dynamically typed languages, we expect hot loops to be mostly type-stable, meaning that the types of values are invariant. (12) For example, we would expect loop coun ters that start as integers to remain integers for all iterations. When both of these expectations hold, a trace-based compiler can cover the program execution with a