Partial evaluation has the ability to totally remove some overheads, such as a runtime security check. A security check typically walks the stack to determine what called a restricted method and then checks to see whether that method had permissions to call the restricted method. As the method on the stack at the point of the call can be determined within the compiler, if it is on the stack as a result of inlining, the precise method is provided by simplification. By evaluating the code of the security check or, if the check is pure, performing a reflective call, the result of the security check can be ascertained and removed if it will always pass.

#### On-stack replacement

*On-Stack Replacement* (OSR) is the process of swapping executing code while it is executing on the stack. For example, if a long running loop were being executed by code created by the JIT baseline compiler, it would be beneficial to swap that code for optimized code when the optimizing compiler produced it. Another example of its use is in invalidating an executing method if it had unsafely assumed properties of the class hierarchy, such as a class having no subclasses, which can allow improvements in inlining.

OSR works by introducing new bytecodes (in the case of the JIT baseline compiler) or new instructions (in the case of the optimizing compiler) that save the live state of execution of a method. Once the execution state is saved, the code can be swapped for newly compiled code, and the execution can continue by loading in the saved state.

#### Summary

This section has given brief descriptions of many of the advanced components of Jikes RVM's optimizing compiler. Writing these components in Java has provided a number of inherent benefits to these systems. Threading is available, allowing all of the components of the system to be threaded. The systems are designed to be thread-safe, allowing, for example, multiple compiler threads to run concurrently. Another advantage has come from Java's generic collections, which provide easy-to-understand libraries that have simplified development and allow the components of the system to concentrate on their role, rather than underlying data structure management.

## Exception Model

Are exceptions things that are exceptional? Although many programmers program with the belief that they are, virtual machines have had to optimize for a use-case that is common in benchmarks. The use-case in question is to read from an input stream and then throw an exception when a sought pattern occurs. This pattern occurs in both the SPECjvm98[†] jack benchmark and the DaCapo 2006[‡] lusearch benchmark.

† *http://www.spec.org/jvm98/*

‡ *http://dacapobench.org/*