CS614: Advanced Compilers

JIT Compilation (Cont.)

Manas Thakur
CSE, IIT Bombay

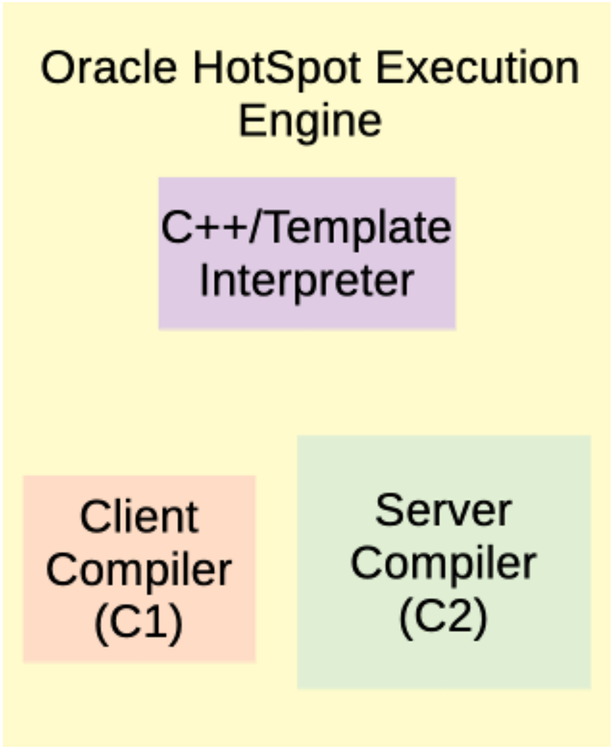
Spring 2025

JIT Compilation in the HotSpot VM

- Starts off with interpretation of bytecodes
- Hot spots get identified by profiling:
  - Method invocation counts
  - Backedge counts
- Identified code regions are inserted into a compilation queue
- Compiler threads compile methods incrementally in the background, while bytecode interpretation continues
- Entry points of methods changed dynamically
- Hot loops replaced on-the-stack
- Additional optimizations performed with possibility of deoptimization (again *incremental*)

JIT Compilation in the HotSpot VM

- C++/Template interpreter
- Two JIT compilers:
  - C1 (aka *client*)
  - C2 (aka *server*)
- Why should we have multiple JIT compilers?
  - JIT compilation has overhead; we can *tier* the compilation into multiple levels.



Tiered JIT in HotSpot

- 0: Interpreter with some profiling
- 1: Pure C1
- 2: C1 with invocation and backedge counting
- 3: C1 with full profiling
- 4: C2

Compiled method states:

- in use
- made not entrant
- zombie

The Interpreter

- Simple switch case

```
switch (bytecode) {
    case nop         : break;
    case aconst_null: push(null); break;
    case iconst_1    : push(1); break;
    ...
}
```

- Disadvantage: Slow
  - Too many comparisons
  - No optimizations

There also is a “Template Interpreter” that generates more optimized assembly snippets.

The C1 Compiler

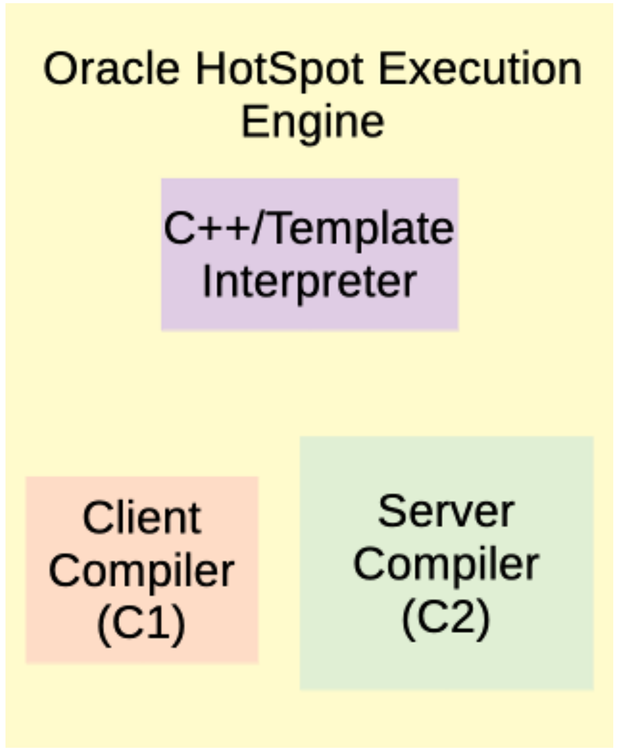
- Targets fast compilation
- Still performs several optimizations:
  - Method inlining
  - Dead code elimination
  - Heuristics for optimizing call sites
  - Constant folding
  - Peephole optimizations
  - Linear-scan register allocation, etc.

- Threshold: 1000 to 2000

The C2 Compiler

- Targets more-and-more optimization
- Performs expensive optimizations (apart from the ones performed by C1):
  - Escape analysis
  - Null-check elimination
  - Loop unrolling
  - Branch prediction
  - Graph-coloring based register allocation, etc.
- Threshold: 10000 to 15000

Tiered JIT in HotSpot



The OpenJ9 VM

+