Using Hardware Counters To Improve Dynamic Compilation

By: David Tam & John Wu

tamda@eecg.toronto.edu
johncwu@ca.ibm.com

February 6, 2004
Introduction

- Dynamic profiling using hardware counters
- Context: JVM
  - Dynamic compilation/optimization
- Intel Open Runtime Platform (ORP) JVM
  - method-at-a-time JIT
  - 2 stage JIT
Motivation

- ORP JVM may not accurately trigger dynamic compilation
- **GOAL**: improve trigger accuracy
  - Cycle counts from hardware counters may be more accurate (per method)
  - Hardware counters offer micro-architectural statistics
Tools and Platform

- Intel ORP JVM
- Performance Counter Library (PCL) by Berrendorf, Ziegler, Mohr
- Linux kernel 2.4.x
- Dual Athlon 1.5GHz, 512MB RAM
Details - Method Profiling

- Must account for:
  1. Start & end of methods
     - Multiple end points possible
  2. For method invocations
     - Stop counter at call sites
     - Resume counter at next line
- Insert assembly code that jumps to start/stop functions (C/C++)
Details - Illustration

```java
methodA() {
    ...
    rc = methodB();
    ...
    return rc;
}

methodB() {
    ...
    ..
    }
```
Baselines

SPECjvm98

Application

Execution Time (sec)

Jess  Compress  Db  Javac  Mpe-gaudio  Jack

Sun J2 SDK1.4.0
Original ORP
O1-only
O3-only
Overheads

SPECjvm98

Application

Execution Time (sec)

Jess  Compress  Db  Javac  Mpegaudio  Jack

64%  64%  12%  62%  49%  78%

226%
Results

SPECjvm98

Applications

Execution Time (sec)

- Original ORP
- 10B-cycles
- 1B-cycles
- 100M-cycles
- 10M-cycles
- 1M-cycles
- 10k-loop, 1M-cycles
- 100k-cycles
- 10k-cycles
- 1k-cycles
Results - magnified

SPECjvm98

Application

Execution Time (sec)

Jess 71%
Db 17%
javac 53%
Mpeg audio 42%
Jack 78%

Legend:
- Original ORP
- 10M-cycles
- 1M-cycles
- 10k-loop, 1M-cycles
- 100k-cycles
- 10k-cycles
- 1k-cycles
Conclusions

- Explored hardware counters
- Added generic infrastructure
  - Dynamic profiling
- Explored only 1 metric
- Results were bad
Future Work

- Reducing overhead
- Other metrics
- Other workloads
- Multi-threaded support
  - Exceptions
  - GC interference / support
- IBM Jikes RVM
- Feedback-directed optimization