# Cache Performance in Java Virtual Machines: A Study of Constituent Phases

Anand S. Rajan
ARM Inc.

Shiwen Hu and Juan Rubio
Laboratory for Computer Architecture
The University of Texas at Austin

# Motivation

- The execution of a Java program consists of distinct JVM phases
  - Class loading
  - Garbage collection
  - Execution

- Efficient execution of Java programs necessitates a comparative study of requirements and characteristics of JVM phases

# Outline

- Experimental methodology
- Instruction cache performance
- Data cache performance
- Impact on cache performance
  - Varying cache sizes
  - Varying application data sets
- Conclusion

# Methodology

- L1 cache behavior of three JVM phases
  - Class loading, garbage collection, and execution
  - Two execution modes: interpreted and JIT
- Experimental workloads: SPECjvm 98 benchmarks
  - Both s1 and s100 data sets are used
- LaTTe JVM:
  - An open-source, state-of-the-art JVM
  - Highly optimized JIT compiler
  - Fast mark-and-sweep garbage collector

# Methodology (Cont.)

- Cache simulator:
  - Based on Cachesim5 from Sun's Shade V6 tool suite
  - A JVM phase aware cache simulator

| Configuration | L1 instruction cache | L1 data cache |
|---|---|---|
| 1 | 16KB, 32 byte blocks, 2-way set associative | 16KB, 32 byte blocks, 4-way set associative, write through with write-no-allocate |
| 2 | 64KB, 32 byte blocks, 2-way set associative | 64KB, 32 byte blocks, 4-way set associative, write through with write-no-allocate |
| 3 | 256KB, 32 byte blocks, 2-way set associative | 256KB, 32 byte blocks, 4-way set associative, write through with write-no-allocate |

# Observations for Both Caches

- Class loading is negligible
  - In terms of cache misses
  - Holds for both execution modes
- Garbage collection is relatively more active in the JIT mode than the interpreted mode
  - Larger working set
  - Reduced total instruction counts and memory references

# Instruction Cache Performance

- Better instruction cache locality in garbage collection than execution phase
- Better instruction miss rate under JIT mode than under interpreted mode
  - Execution phase: high method reuse
  - Garbage collection: low miss rate

# Data Cache Performance

- Higher overall data cache miss rate under JIT mode
  - Better data locality under interpreted mode
- Data accesses reduce drastically under JIT mode
  - Bytecodes are read only once for compilation
  - Stack accesses are optimized into register-register operations

# Data Cache Reads

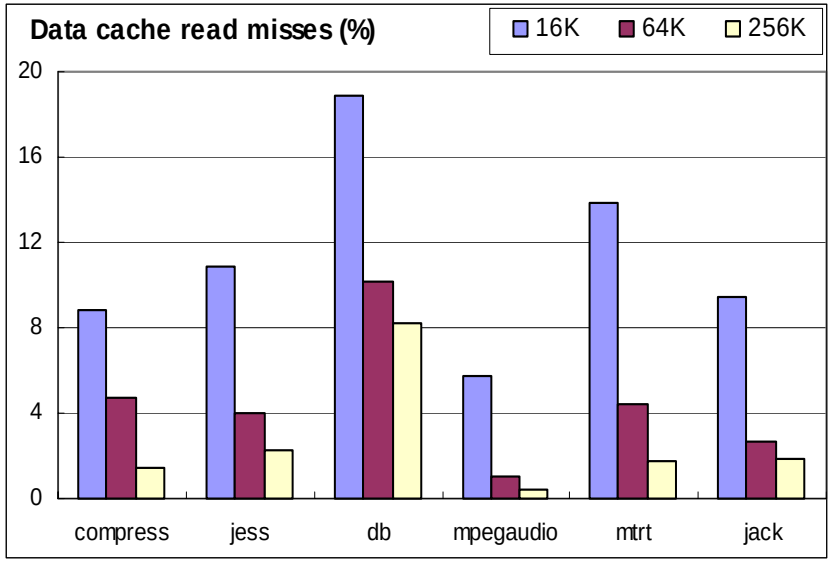
- Higher read miss rate under JIT mode
  - Interpreted mode: 0.94% (mpegaudio) to 5.1% (jess)
  - JIT mode: 5.75% (mpegaudio) to 19% (db)
- High read miss rate during garbage collection under both execution modes
  - Up to 19.4% (interpreted) and 18.8% (JIT)
  - Due to large working set and pointer chasings
- Read misses dominated by execution phase under both execution modes
  - Execution phase contributes > 90% of read misses

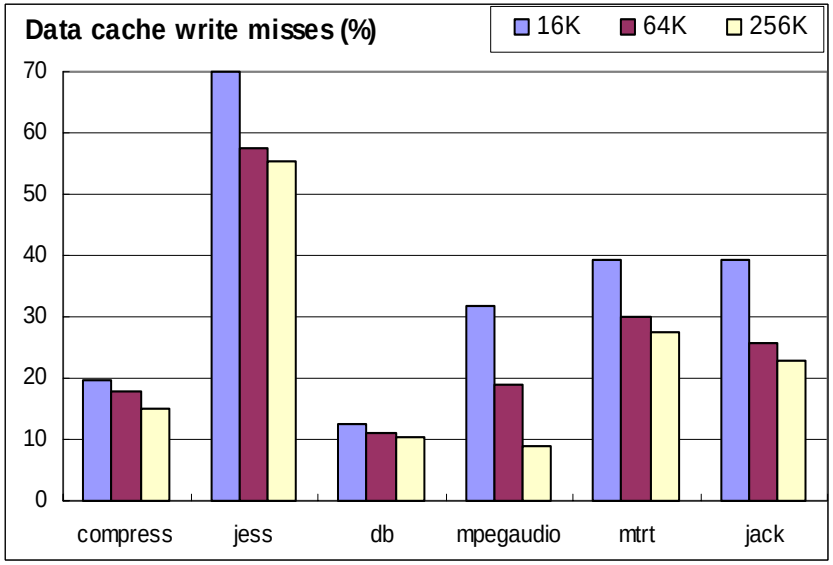
# Data Cache Writes

- High write miss rates in garbage collection
  - For both execution modes
  - 50% to 74%
- Garbage collection contributes 44%-78% of write misses
  - For both execution modes
  - Exceptions: compress and mpegaudio

# Data Cache Performance with Increased Sizes (Execution Phase)

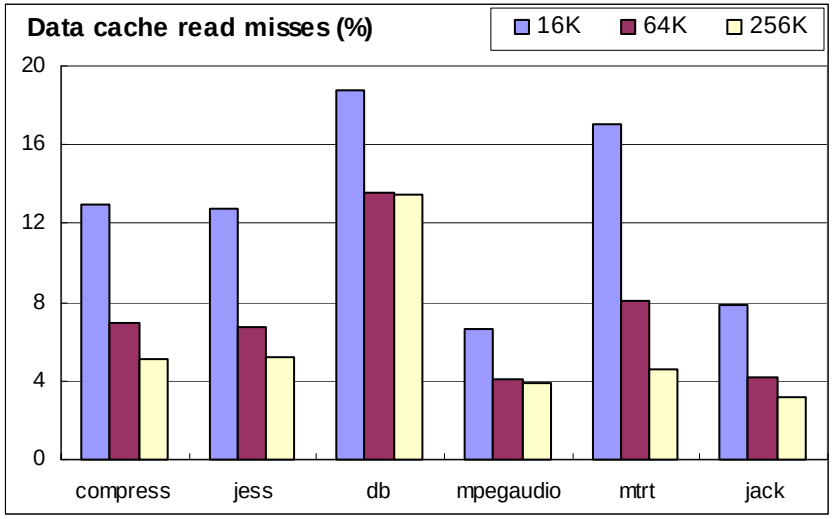
- Write misses are harder to be removed by larger caches
  - Most write misses are compulsory misses
  - Holds for both phases

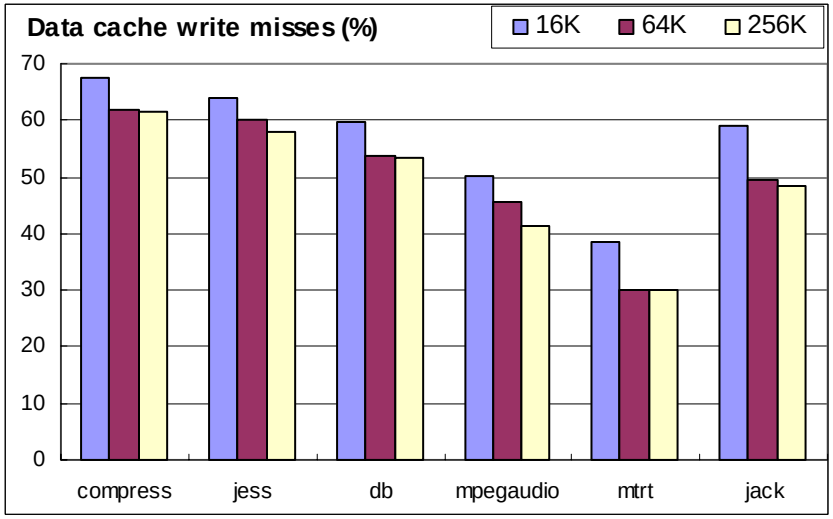


# Data Cache Performance with Increased Sizes (Garbage Collection Phase)

- Larger caches more effective for the execution phase than the garbage collector
  - Working set of the garbage collector is much larger than 256KB
- Diminishing reduction on data cache misses



# Impact of Larger Data Sets

- JVM phases perform differently as data set increases
  - Little change: class loading, JIT compilation
  - Big change: interpretation, garbage collection
- Garbage collection under both execution modes
  - Performance deteriorates for both data cache reads and writes
  - Performances of instruction cache accesses varies little

# Impact of Larger Data Sets (Cont.)

- Execution phase under interpreted mode:
  - Cache performance varies little
- Execution phase under JIT mode:
  - Performance improves for both instruction cache accesses and data cache writes
  - Performance of data cache reads deteriorates

# Conclusion

- L1 cache performance of Java programs under
  - Three phases
  - Two execution engines
  - Three cache configurations
  - Two application data sets
- L1 instruction cache performance determined by the execution phase
- Garbage collection is more significant in the JIT mode than in the interpreted mode

# Conclusion (Cont.)

- Higher data cache miss rates of the garbage collector than the execution engine
- Higher data cache miss rates in the JIT mode than in the interpreted mode
- A larger cache is more effective on eliminating
  - Read misses than write misses
  - misses from execution phase than from garbage collector
- Impact of changing data set varies depending on the JVM phase and cache access type

# THANK YOU

# Appendix

# Instruction Cache Performance

| Benchmark | Execution Phase | | Garbage Collection Phase | | Overall |
|---|---|---|---|---|---|
| | % Abs. miss | %Total miss | % Abs. miss | %Total miss | % Abs. miss |
| Compress (int) | 1.30 | 99.46 | 0.15 | 0.002 | 1.30 |
| (jit) | 0.07 | 98.85 | 0.16 | 1.03 | 0.07 |
| Jess (int) | 1.35 | 96.48 | 1.03 | 3.51 | 1.33 |
| (jit) | 1.48 | 85.68 | 0.68 | 14.28 | 1.26 |
| Db (int) | 0.16 | 99.35 | 0.10 | 0.62 | 0.16 |
| (jit) | 0.12 | 97.70 | 0.03 | 2.21 | 0.11 |
| Mpegaudio (int) | 0.60 | 99.99 | 0.45 | 0.003 | 0.60 |
| (jit) | 0.18 | 99.51 | 0.31 | 0.48 | 0.18 |
| Mtrt (int) | 0.47 | 68.51 | 0.42 | 31.40 | 0.46 |
| (jit) | 1.21 | 54.45 | 0.51 | 45.52 | 0.75 |
| Jack (int) | 0.72 | 97.14 | 0.89 | 2.86 | 0.72 |
| (jit) | 1.31 | 95.98 | 0.27 | 4.00 | 1.14 |

# Decomposition of Data Cache Misses

| Benchmark | Class Loading % | Execution % | Garbage Collection % | Overall D-Cache Miss % |
|---|---|---|---|---|
| Compress(int) | 0.001 | 99.84 | 0.15 | 2.98 |
| (jit) | 0.003 | 98.99 | 0.98 | 3.60 |
| Jess (int) | 0.003 | 81.04 | 18.94 | 6.11 |
| (jit) | 0.004 | 58.77 | 41.20 | 24.07 |
| Db (int) | 0.002 | 94.48 | 5.50 | 4.20 |
| (jit) | 0.004 | 86.33 | 13.65 | 19.52 |
| Mpegaudio (int) | 0.004 | 99.94 | 0.05 | 1.08 |
| (jit) | 0.005 | 99.63 | 0.34 | 11.06 |
| Mtrt (int) | 0.003 | 31.14 | 68.86 | 4.09 |
| (jit) | 0.005 | 28.16 | 71.81 | 21.47 |
| Jack (int) | 0.004 | 84.26 | 15.72 | 3.09 |
| (jit) | 0.007 | 60.82 | 39.14 | 18.87 |

# Performance of Data Cache Reads

| Benchmark | Execution Phase | | Garbage Collection Phase | | Overall |
|---|---|---|---|---|---|
| | % Abs. miss | %Total miss | % Abs. miss | %Total miss | % Abs. miss |
| Compress (int) | 2.03 | 99.94 | 16.14 | 0.05 | 2.03 |
| (jit) | 8.78 | 99.62 | 12.99 | 0.36 | 8.79 |
| Jess (int) | 4.92 | 95.26 | 14.74 | 4.73 | 5.08 |
| (jit) | 10.92 | 82.30 | 12.74 | 17.68 | 11.19 |
| Db (int) | 3.67 | 98.06 | 19.39 | 0.02 | 3.72 |
| (jit) | 18.83 | 95.94 | 18.76 | 4.04 | 18.82 |
| Mpegaudio (int) | 0.94 | 99.98 | 3.88 | 0.01 | 0.94 |
| (jit) | 5.75 | 99.75 | 6.66 | 0.21 | 5.75 |
| Mtrt (int) | 4.09 | 33.55 | 4.31 | 66.44 | 4.23 |
| (jit) | 13.83 | 33.18 | 17.04 | 66.79 | 15.84 |
| Jack (int) | 2.68 | 97.89 | 8.24 | 2.09 | 2.72 |
| (jit) | 9.48 | 90.03 | 7.86 | 9.91 | 9.29 |

# Performance of Data Cache Writes

| Benchmark | Execution Phase | | Garbage Collection Phase | | Overall |
|---|---|---|---|---|---|
| | % Abs. miss | %Total miss | % Abs. miss | %Total miss | % Abs. miss |
| Compress (int) | 6.18 | 99.73 | 74.56 | 0.26 | 6.19 |
| (jit) | 19.61 | 97.77 | 67.37 | 2.20 | 19.92 |
| Jess (int) | 5.68 | 55.81 | 65.32 | 44.17 | 9.53 |
| (jit) | 69.86 | 45.72 | 63.88 | 54.26 | 66.48 |
| Db (int) | 5.18 | 86.36 | 60.59 | 13.63 | 5.92 |
| (jit) | 12.54 | 40.34 | 59.59 | 59.61 | 59.61 |
| Mpegaudio (int) | 1.51 | 99.85 | 49.89 | 0.14 | 1.51 |
| (jit) | 31.91 | 99.54 | 50.04 | 0.44 | 31.95 |
| Mtrt (int) | 2.44 | 22.51 | 4.27 | 77.47 | 3.66 |
| (jit) | 39.31 | 21.85 | 38.69 | 78.12 | 38.82 |
| Jack (int) | 2.34 | 54.90 | 57.70 | 44.96 | 4.12 |
| (jit) | 39.21 | 43.56 | 58.99 | 56.42 | 48.35 |

# Cache Performance with Different Data Sets (JIT Execution)

| Miss Rates (%) | I-cache | | D-cache Read | | D-cache Write | |
|---|---|---|---|---|---|---|
| | **s1** | **s100** | **s1** | **s100** | **s1** | **s100** |
| Compress (int) | 0.01 | 1.3 | 0.24 | 2.03 | 0.75 | 6.18 |
| (jit) | 0.58 | 0.07 | 8.03 | 8.78 | 25.89 | 19.61 |
| Jess (int) | 1.43 | 1.35 | 7.75 | 4.92 | 8.18 | 5.68 |
| (jit) | 1.95 | 1.48 | 9.46 | 10.92 | 61.2 | 69.86 |
| Db (int) | 0.45 | 0.16 | 2.05 | 3.67 | 5.17 | 5.18 |
| (jit) | 1.79 | 0.12 | 8.87 | 18.83 | 54.77 | 12.54 |
| Mpegaudio (int) | 1.02 | 0.6 | 2.17 | 0.94 | 1.66 | 1.51 |
| (jit) | 1.7 | 0.18 | 9.01 | 5.75 | 47.48 | 31.91 |
| Mtrt (int) | 0.59 | 0.42 | 3.18 | 4.09 | 3.77 | 2.44 |
| (jit) | 1.78 | 1.21 | 9.04 | 13.83 | 55.41 | 39.31 |
| Jack (int) | 0.7 | 0.72 | 2.64 | 2.68 | 2.41 | 2.34 |
| (jit) | 1.78 | 1.31 | 10.01 | 9.48 | 54.59 | 39.21 |

# Cache Performance with Different Data Sets (Garbage Collection)

| Miss Rates (%) | I-cache | | D-cache Read | | D-cache Write | |
|---|---|---|---|---|---|---|
| | **s1** | **s100** | **s1** | **s100** | **s1** | **s100** |
| Compress (int) | 0.04 | 0.15 | 3.3 | 16.14 | 52.17 | 74.56 |
| (jit) | 0.25 | 0.16 | 6.73 | 12.99 | 51.49 | 67.37 |
| Jess (int) | 0.73 | 1.03 | 4.04 | 14.74 | 59.8 | 65.32 |
| (jit) | 0.38 | 0.68 | 4.92 | 12.74 | 50.99 | 63.88 |
| Db (int) | 0.05 | 0.1 | 0.96 | 19.39 | 47.15 | 60.59 |
| (jit) | 0.28 | 0.03 | 5.49 | 18.76 | 47.99 | 59.59 |
| Mpegaudio (int) | 0.51 | 0.45 | 4.17 | 3.88 | 49.89 | 49.89 |
| (jit) | 0.31 | 0.31 | 6.64 | 6.66 | 50.07 | 50.04 |
| Mtrt (int) | 0.29 | 0.47 | 13.29 | 4.31 | 65.68 | 4.27 |
| (jit) | 0.17 | 0.51 | 10.52 | 17.04 | 62.62 | 38.69 |
| Jack (int) | 0.69 | 0.89 | 6.37 | 8.24 | 60.35 | 57.7 |
| (jit) | 0.29 | 0.27 | 6.18 | 7.86 | 57.21 | 58.99 |