Performance Testing for JEP 450: Compact Object Headers

SPECjvm

This is SPECjvm2008 modified to run with JDK24. Scores are ops/min, more is better. All benchmarks have been run on a AWS c5.9xlarge instance.

JVM flags: `-Xms8g -Xmx8g -XX:+AlwaysPreTouch`

Notes: The CryptoRsa benchmark shows a legit regression of ~2%. I confirmed this by running 50 times (most benchmarks are run 3 times). The regression disappears when improving oopDesc::klass().

Benchmark	Baseline	-UCOH	%	+UCOH	%
Compiler.compiler	1127.077	1127.628	0.04%	1150.953	2.11%
Compiler.sunflow	3121.071	3190.964	2.23%	3202.471	2.60%
Compress	3100.924	3114.255	0.42%	3037.949	-2.04%
CryptoAes	214.615	215.491	0.40%	212.788	-0.86%
CryptoRsa	10284.167	10094.019	-1.85%	10110.913	-1.69%
CryptoSignVerify	48591.761	48541.003	-0.11%	48107.972	-1.00%
Derby	3646.723	3620.906	-0.71%	3598.845	-1.32%
MpegAudio	1012.491	1011.096	-0.14%	1005.387	-0.71%
ScimarkFFT.large	278.913	278.937	0.00%	278.613	-0.11%
ScimarkFFT.small	3348.048	3374.245	0.78%	3276.573	-2.14%
ScimarkLU.large	21.737	21.771	0.15%	21.640	-0.45%
ScimarkLU.small	5876.355	5870.420	-0.11%	5867.935	-0.15%
ScimarkMonteCarlo	17635.373	17767.207	0.74%	17845.029	1.18%
ScimarkSOR.large	324.000	322.590	-0.44%	322.137	-0.58%
ScimarkSOR.small	1647.129	1647.330	0.01%	1648.043	0.05%
ScimarkSparse.large	196.737	196.639	-0.05%	196.934	0.10%
ScimarkSparse.small	1305.586	1294.917	-0.82%	1289.868	-1.21%
Serial	48075.242	48503.947	0.89%	48583.323	1.05%
Sunflow	667.590	673.791	0.92%	701.846	5.13%
XmlTransform	2005.798	2016.622	0.53%	1965.465	-2.02%
XmlValidation	4503.275	4486.646	-0.37%	4485.866	-0.39%

Renaissance

This is renaissance-jmh-0.15.0, excluding Als, ChiSquare, DecTree, GaussMix, LogRegression, MovieLens, NaiveBayes, PageRank and DbShootout, which are not compatible with JDK24 due to SecurityManager restriction. Scores are ms/op, less is better. All benchmarks have been run on a AWS c5.9xlarge instance.

JVM flags: `-Xms8g -Xmx8g -XX:+AlwaysPreTouch`

Benchmark	Baseline	-UCOH	%	+UCOH	%
AkkaUct	799.526	796.987	-0.32%	792.829	-0.84%
Reactors	15192.454	15070.100	-0.81%	14533.405	-4.34%
FjKmeans	1048.697	1049.511	0.07%	953.387	-9.09%
FutureGenetic	2011.925	1991.429	-1.02%	2072.172	2.99%
Mnemonics	2760.609	2739.520	-0.77%	2824.556	2.31%
ParMnemonics	2270.166	2286.626	0.72%	2304.885	1.52%
Scrabble	53.150	53.525	0.70%	55.052	3.57%
Neo4jAnalytics	1482.061	1479.183	-0.20%	1511.736	2.00%

RxScrabble	104.344	103.626	-0.69%	105.156	0.77%
Dotty	697.111	700.798	0.52%	700.191	0.44%
ScalaKmeans	175.545	177.320	1.01%	188.345	7.29%
Philosophers	5598.596	5645.268	0.83%	6379.457	13.94%
ScalaStmBench7	1010.489	1018.077	0.75%	999.638	-1.08%
FinagleChirper	3643.971	3629.694	-0.40%	3681.006	1.01%
FinagleHttp	3392.307	3338.891	-1.58%	3269.203	-3.63%

SPECjbb2015

SPECjbb2015 in composite mode, run on a AWS c5.9xlarge instance, 10 times in a row, scores averaged over all runs. More is better.

JVM flags: `-Xms20g -Xmx20g -XX:+AlwaysPreTouch`

Score	Baseline	-UCOH	%	+UCOH	%
max-jops	49818	49700	-0.24%	51652	3.68%
critial-jops	39139	38726	-1.06%	41219	5.31%

GC Pause times

Young GC time, measured over 10 runs of SPECjbb2015 each, the total number of cycles is given, too.

```
Young GC time, baseline: avg: 71.4059 ms, #cycles: 22909
Young GC time, -UCOH:    avg: 73.2457 ms, #cycles: 23032
Young GC time, +UCOH:    avg: 69.8878 ms, #cycles: 20619
```

Concurrent marking time, measured over a run of SPECjvm compiler.compiler benchmark.

```
Concurrent marking time, baseline: avg: 1757.43 ms, #cycles: 69
Concurrent marking time, -UCOH:    avg: 1724.96 ms, #cycles: 78
Concurrent marking time, +UCOH:    avg: 1664.67 ms, #cycles: 62
```

SPECjbb2015 (run at Red Hat)

The following tests were run at Red Hat on a bare metal RHEL9 x64 i7-4770 8core machine, with the benchmark process isolated on 6 cores, 10 consecutive runs, all scores averaged. Perf statistics were done for the full benchmark process run, therefore include warmup and report generation.

JVM flags: `-Xshare:off -Xlog:gc* -XX:+UseG1GC -Xms4g -Xmx4g`

	-COH	+COH	%
maxjops	10136.6	10823.6	+6.78%
critjops	3742	3995.6	+6.78%
Number of GCs	4971.8	3469.8	-30.21%
GC Real Times, Sum, seconds	350.34	273	-22.08%
GC User Times, Sum, seconds	2031.83	1588.63	-21.81%
GC Sys Times, Sum, seconds	3.07	2.2	-28.34%
L1 Misses	1294363538563.4	1109038645193.3	-14.32%
L1 Loads	25511385965443.6	21610503370719.1	-15.29%
LLC Misses	273833668865.4	212658994859.6	-22.34%

LLC Loads	655211047461.7	555606059480	-15.20%
TLB Misses	37718612782.9	32378833894.3	-14.16%
TLB Loads	25308734883537.9	21437105901958.6	-15.30%
Instructions	83966573365463.8	72477839817765.8	-13.68%
Branches	15749934748078.2	13181107354563.9	-16.31%

To measure the reduction in liveset size, the same benchmark was repeated with a periodic Full GC being scheduled every 5 minutes, and the liveset size was measured after every full GC:

	-COH	+COH	Percent	Gains
Usage pre GC, MB	3894.9	3216.6	82.58	- 17.42
Usage post GC, MB	1071.7	885.9	82.66	- 17.34