Performance Compare of JSR-94 Engines

Part 2

(contact@witmate.com)

Abstract

As a continuous case of Performance Compare of JSR-94 Engines Part1 (http://witmate.com/materials/jsr94_performance_compare_part1.pdf), this article introduces a performance compare of press test case between Witmate and Drools which is introduced as the best performance rule engine by some information sources of Internet.

ATTENTION: This compare is done by Witmate, and only for our clients' system performance evaluation.

Test Case

Performance Compare of JSR-94 Engines Part1 introduced a general performance test case and results. Another important part of performance compares is press test case. The press test case is the test to know the performance when engines get a heavy load. The heavy load of this test case is a huge rule set with 8000 rules. By Witmate SL, it's like:

```
If v=1 then return "R1"
If v=2 then return "R2"
If v=3 then return "R3"
…
If v=8000 then return "R8000"
```

Test code generates a random integer number from 1 to 8000, and input into engines as the value of variable v.

Engines load this rule set, and execute it 5 times with Stateless and Stateful method, and log the average execution time and maximum memory usage. A batch command file boots up standalone java VM each time for each engine. And we executed this batch 10 times on test bed, then use the average value to generate compare result.

Test bed

The test bed we used is:

Hardware: P4 3GHz with HT CPU, 2G RAM

Software: Windows XP, Java 1.5.0_06

Java VM parameter: -Xnoclassgc –Xms1000m –Xmx1000m

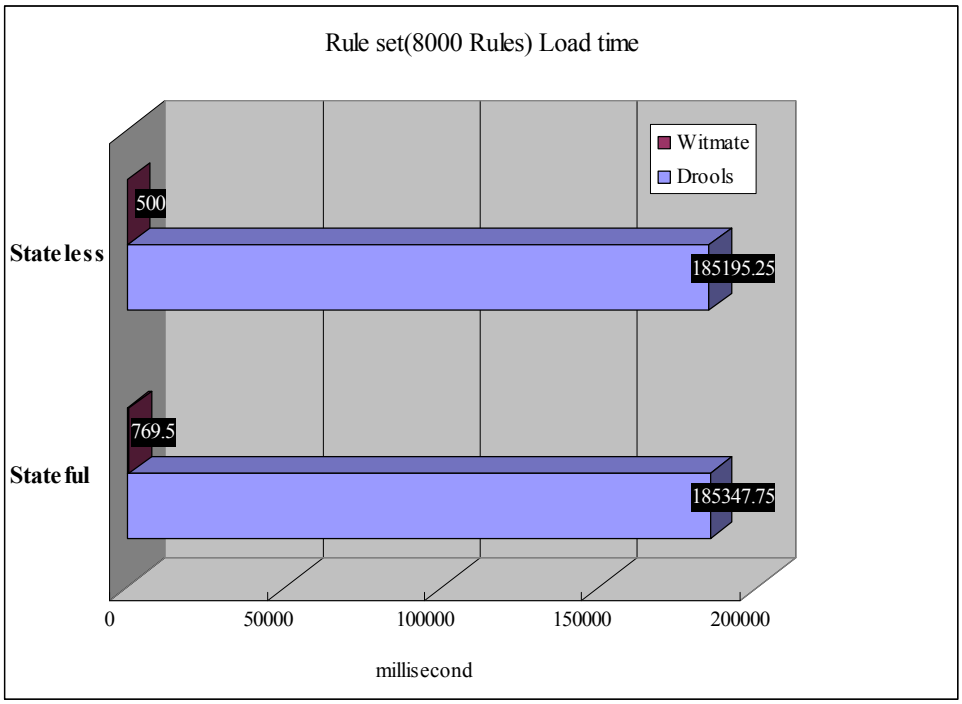
To get correct run time and memory usage, GC (garbage collection) is closed.

Rule set data: WitStream for Witmate and XML for Drools.

Rule Set Load time

Because the rule set is huge, so load time became to a concern cost.

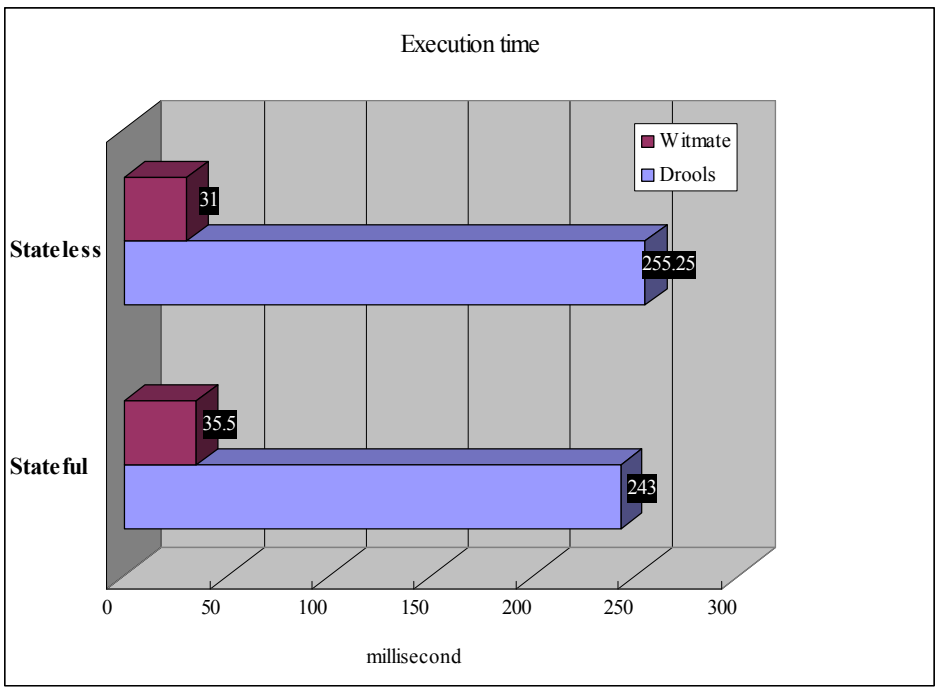
WitStream performance is impressible that is over 200 times faster than Drools loading.



Execution time

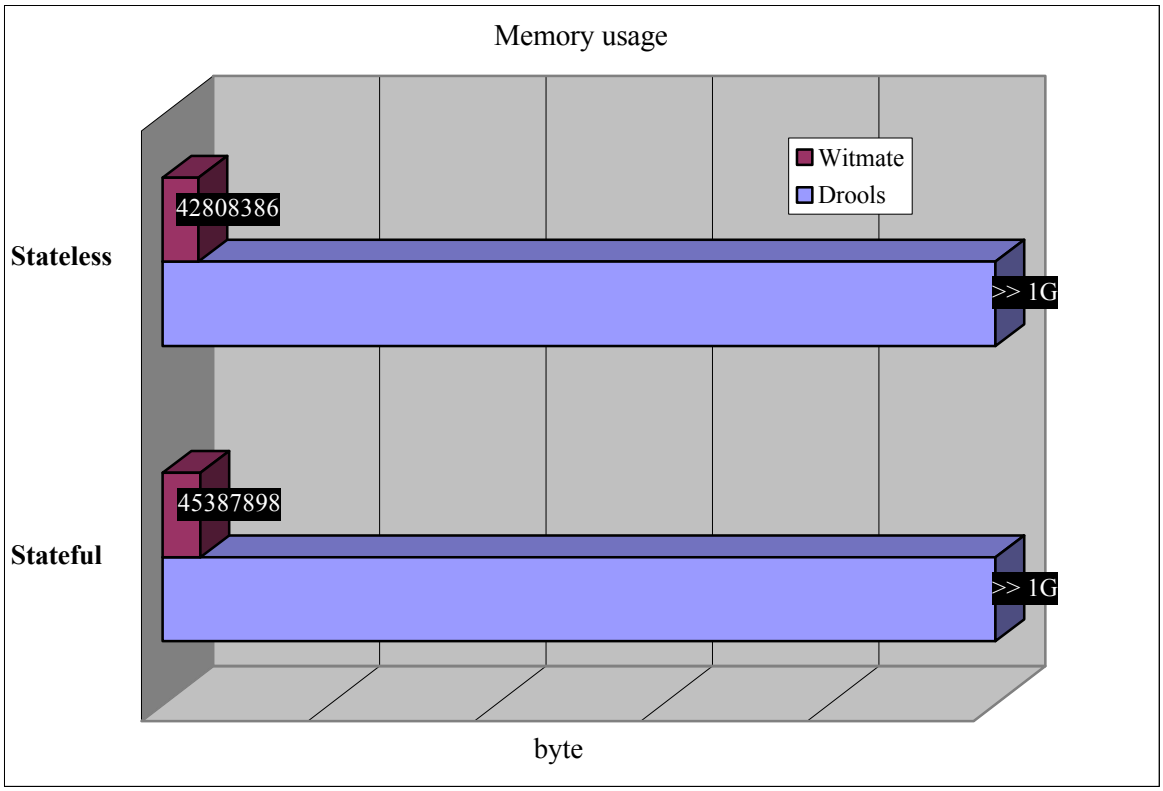
This is the time to fire a correct rule in 8000 rules.

Witmate keeps 8 times faster than Drools.



Memory usage

Memory used for firing a correct rule in huge rule set. Drools used too much memory, so JVM forced GC execution, 1688 times includes 10 times of full GC, to collect memory to keep Drools running. So the memory usage of Drools is much larger than 1 G. There is a content sample of GC log in appendix.



Conclusion

Witmate has impressible leading performance for heavy load. The advantage of memory usage is shown much clearer than general test case, and at same time, Witmate keeps the 8 times faster of execution than competitors.

WitStream is a high effective logic describing format for huge rule set loading.

From the results of Part 1 and 2, it is proved that Witmate is not only a mobile ready engine, but also a powerful engine for heavy business rule matches.

Appendix of GC log of Drools execution

This is the part of content of GC log file that logged, by a separated execution with performance test:

```
0.000: [GC 62879K->387K(1016128K), 0.0046744 secs]
0.062: [GC 63113K->669K(1016128K), 0.0020274 secs]
0.124: [GC 63203K->768K(1016128K), 0.0018277 secs]
0.186: [GC 63243K->852K(1016128K), 0.0017530 secs]
0.249: [GC 63783K->579K(1016128K), 0.0014242 secs]
0.312: [GC 63249K->613K(1016128K), 0.0014303 secs]
0.374: [GC 62788K->1054K(1016128K), 0.0020214 secs]
0.438: [GC 64013K->1111K(1016128K), 0.0020616 secs]
0.502: [GC 63624K->1629K(1016128K), 0.0027594 secs]
0.566: [GC 63911K->1215K(1016128K), 0.0020189 secs]
0.632: [GC 63659K->1263K(1016128K), 0.0019721 secs]
0.699: [GC 63995K->771K(1016128K), 0.0015349 secs]
0.768: [GC 62880K->1353K(1016128K), 0.0020493 secs]
0.839: [GC 64184K->1976K(1016128K), 0.0032375 secs]
...
59.770: [Full GC 948660K->13236K(1016128K), 0.0216227 secs]
59.863: [GC 73948K->26247K(1016128K), 0.0111111 secs]
...
72.154: [GC 938384K->886302K(1016128K), 0.0101920 secs]
72.230: [GC 943151K->891043K(1016128K), 0.0121582 secs]
72.307: [Full GC 947920K->9714K(1016128K), 0.0181976 secs]
72.393: [GC 66619K->28682K(1016128K), 0.0170609 secs]
...
98.976: [Full GC 950388K->11239K(1016128K), 0.0222344 secs]
...
128.358: [GC 309793K->260105K(1016128K), 0.0165271 secs]
128.449: [GC 322240K->266321K(1016128K), 0.0105969 secs]
128.529: [GC 328474K->278754K(1016128K), 0.0162726 secs]
128.620: [GC 340928K->284975K(1016128K), 0.0104983 secs]
128.700: [GC 347167K->297416K(1016128K), 0.0163885 secs]
128.791: [GC 359628K->303640K(1016128K), 0.0107066 secs]
128.871: [GC 365871K->316088K(1016128K), 0.0185420 secs]
128.966: [GC 378344K->322317K(1016128K), 0.0112872 secs]
129.047: [GC 384587K->334773K(1016128K), 0.0162173 secs]
129.138: [GC 397064K->341005K(1016128K), 0.0111040 secs]
129.220: [GC 403314K->353469K(1016128K), 0.0163794 secs]
129.312: [GC 415799K->359705K(1016128K), 0.0110510 secs]
129.393: [GC 422053K->372177K(1016128K), 0.0169579 secs]
129.485: [GC 434547K->378417K(1016128K), 0.0111727 secs]
129.566: [GC 440804K->390897K(1016128K), 0.0167875 secs]
129.657: [GC 453307K->397141K(1016128K), 0.0113035 secs]
129.738: [GC 459567K->409629K(1016128K), 0.0169164 secs]
131.117: [GC 472605K->411052K(1016128K), 0.0242323 secs]
133.634: [GC 474028K->413246K(1016128K), 0.0307076 secs]
135.404: [GC 476222K->415443K(1016128K), 0.0319372 secs]
136.943: [GC 478419K->417662K(1016128K), 0.0429128 secs]
138.633: [GC 480638K->419879K(1016128K), 0.0454071 secs]
140.274: [GC 482855K->422066K(1016128K), 0.0466141 secs]
141.972: [GC 485042K->424292K(1016128K), 0.0483690 secs]
143.521: [Full GC 479191K->23003K(1016128K), 0.2516208 secs]
145.558: [GC 85979K->25237K(1016128K), 0.0271813 secs]
147.498: [GC 88213K->27465K(1016128K), 0.0379882 secs]
149.548: [GC 90441K->29676K(1016128K), 0.0501943 secs]
151.613: [GC 92652K->31865K(1016128K), 0.0544699 secs]
153.018: [Full GC 73600K->33318K(1016128K), 0.2955122 secs]
155.506: [GC 96294K->35604K(1016128K), 0.0328349 secs]
157.850: [GC 98580K->37815K(1016128K), 0.0468522 secs]
160.264: [GC 100791K->40069K(1016128K), 0.0586432 secs]
162.791: [GC 103045K->42258K(1016128K), 0.0619571 secs]
164.517: [Full GC 83048K->43696K(1016128K), 0.3941816 secs]
167.481: [GC 106672K->45907K(1016128K), 0.0394120 secs]
170.159: [GC 108883K->48150K(1016128K), 0.0522190 secs]
```