WaveScript Benchmarks Performance Report

August 6, 2008

Machine information:
Linux chastity 2.6.22-14-generic #1 SMP Tue Feb 12 07:42:25 UTC 2008 i686 GNU/Linux

WaveScript SVN:
Revision: 3432

WaveScope Engine SVN:
(omitted for now)

1 Microbenchmarks

This section reports various microbenchmarks that stress the implementation of particular language constructs or data types.
Per-stream-element overheads

One thing that you can see, is that currently (2007.10) the C++/XStream engine has a high per-tuple (that is, per-element) on the communication channels relative to the ML backend. The **just_timer** test stresses this, doing nothing but passing a large number of unit tuples.

Focusing on scheduling overheads a bit more, we turn to the following data passing microbenchmarks. These do nothing but generate a stream of numbers, and then add up windows of those numbers. We vary the window size in the following graphs. The numbers are passed either one at a time (“raw”), or in bulk using arrays or lists.

Notes:
• FFT results for Scheme above depend on whether or not it is configured to use FFTW, or a native
Scheme fourier transform.

2 Language Shootout Benchmarks

This is where I will accumulate some of the small benchmarks from the language shootout. Here are some
per-benchmark comments:

• fannkuch - “pancake flipping”. This is a translation of the gcc version of the benchmark. Tests
indexed access to a small array.

3 Application Benchmarks

This section includes performance results on larger programs, namely, our current applications. Presently
(2007.10) the largest of these by far is the marmot application.

3.1 Marmot Application

We start off by looking at the original, hand-optimized marmot application that we deployed.
4 Data Representation Profiling

This is stale data for now... having sneaky problems with the datarep Makefile that are hosing regression tests. [2007.11.07]

This section includes an analysis of the efficiency of different data representations under different backends. This should theoretically be run on different hardware platforms as well (such as the ARM-based ensboxes).

4.1 Arrays of Arrays

Arrays of arrays are notable because they cannot generally be flattened (the inner arrays will always be pointers). In the future we may look at tentative flattening based on profiling data. But first, here are the times for repeatedly allocating an array of arrays, and for repeatedly folding the values in an array of arrays.

Next we look at allocating arrays of tuples and vice versa. We look at both square sizes and at highly skewed dimensions. This is limited by not being able to make tuples very large.
Then we do examine folding over arrays of tuples and tuples of arrays.

A  Appendix: Raw numbers for above graphs

Microbenchmarks

## User time for each benchmark/backend
Benchmark mltonO3 c2boehm c2boehmseglist c2 c2seglist c2def c2defseglist
just_timer 124.000 128.000 128.000 124.000 124.000 252.000 252.000
readfile_bigwins 552.000 64.000 72.000 0 8.000 4.000 8.000
printing_lists 1324.000 416.000 456.000 416.000 432.000 396.000 416.000
conv_SigsegArr 536.000 112.000 1804.000 200.000 1392.000 8.000 1640.000
fft 76.000 468.000 452.000 448.000 484.000 452.000 488.000

Language Shootout:

## User time for each language-shootout benchmark/backend
Benchmark c2
fannkuch2 4528.000

Application Benchmarks:

Benchmark mltonO3 c2boehm c2boehmseglist c2 c2seglist c2def c2defseglist
## Running orig marmot phase 1
run_first_phase 1496.000 2464.000 880.000 1564.000 1164.000 1656.000 748.000
## Running marmot2
test_marmot2 2220.000 5276.000 5232.000 4676.000 4692.000 4544.000 4624.000
## Running marmot3
test_heatmap 7916.000 3244.000 3208.000 2564.000 2556.000 3208.000 3208.000
## Running marmot multinode offline
run_3phases 9209.000 6088.000 4976.000 5684.000 5264.000 5740.000 4868.000

B Appendix: Additional system information

Top results before running benchmarks:

```
top - 06:15:28 up 15 days, 14:49, 5 users, load average: 2.03, 2.08, 1.74
Tasks: 165 total, 2 running, 163 sleeping, 0 stopped, 0 zombie
Cpu(s): 37.5%us, 5.1%sy, 0.0%ni, 56.4%id, 0.1%wa, 0.4%hi, 0.5%si, 0.0%st
Mem: 2073956k total, 1545516k used, 528440k free, 215852k buffers
Swap: 14996668k total, 34744k used, 14961924k free, 898076k cached
```

<table>
<thead>
<tr>
<th>PID</th>
<th>USER</th>
<th>PR</th>
<th>NI</th>
<th>VIRT</th>
<th>RES</th>
<th>SHR</th>
<th>S</th>
<th>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>24183</td>
<td>newton</td>
<td>25</td>
<td>0</td>
<td>34212</td>
<td>31m</td>
<td>492</td>
<td>R</td>
<td>99</td>
<td>1.6</td>
<td>1421:11 ikarus</td>
<td></td>
</tr>
<tr>
<td>11358</td>
<td>newton</td>
<td>15</td>
<td>0</td>
<td>34940</td>
<td>26m</td>
<td>4048</td>
<td>S</td>
<td>16</td>
<td>1.3</td>
<td>267:36.37 unison-2.18</td>
<td></td>
</tr>
<tr>
<td>6353</td>
<td>root</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>2</td>
<td>0.0</td>
<td>140:41.97 afs_rxlistener</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 root</td>
<td>18</td>
<td>0</td>
<td>2948</td>
<td>1856</td>
<td>532</td>
<td>S</td>
<td>0</td>
<td>0.1</td>
<td>0:03:57 init</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 root</td>
<td>11</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:00 kthread</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 root</td>
<td>19</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:13 migration/0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 root</td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:30 ksoftirqd/0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 root</td>
<td>RT</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:00 watchdog/0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6 root</td>
<td>RT</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:08 migration/1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7 root</td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:70 ksoftirqd/1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8 root</td>
<td>RT</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:00 watchdog/1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>9 root</td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:01 events/0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 root</td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03 events/1</td>
<td></td>
</tr>
</tbody>
</table>

Top results after running benchmarks:

```
top - 06:33:10 up 15 days, 15:07, 5 users, load average: 2.10, 2.13, 2.01
Tasks: 164 total, 3 running, 161 sleeping, 0 stopped, 0 zombie
Cpu(s): 37.5%us, 5.1%sy, 0.0%ni, 56.3%id, 0.1%wa, 0.4%hi, 0.5%si, 0.0%st
Mem: 2073956k total, 1399744k used, 679712k free, 208768k buffers
Swap: 14996668k total, 190768k used, 14976896k free, 764232k cached
```

<table>
<thead>
<tr>
<th>PID</th>
<th>USER</th>
<th>PR</th>
<th>NI</th>
<th>VIRT</th>
<th>RES</th>
<th>SHR</th>
<th>S</th>
<th>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>24183</td>
<td>newton</td>
<td>25</td>
<td>0</td>
<td>34212</td>
<td>31m</td>
<td>492</td>
<td>R</td>
<td>99</td>
<td>1.6</td>
<td>1437:52 ikarus</td>
<td></td>
</tr>
<tr>
<td>11358</td>
<td>newton</td>
<td>15</td>
<td>0</td>
<td>34940</td>
<td>26m</td>
<td>4048</td>
<td>S</td>
<td>16</td>
<td>1.3</td>
<td>269:19.22 unison-2.18</td>
<td></td>
</tr>
<tr>
<td>6353</td>
<td>root</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>2</td>
<td>0.0</td>
<td>140:52:05 afs_rxlistener</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 root</td>
<td>18</td>
<td>0</td>
<td>2948</td>
<td>1856</td>
<td>532</td>
<td>S</td>
<td>0</td>
<td>0.1</td>
<td>0:03:57 init</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 root</td>
<td>11</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:00 kthread</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 root</td>
<td>19</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:13 migration/0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 root</td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:30 ksoftirqd/0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 root</td>
<td>RT</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:00 watchdog/0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6 root</td>
<td>RT</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:08 migration/1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7 root</td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:70 ksoftirqd/1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8 root</td>
<td>RT</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:00 watchdog/1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>9 root</td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:01 events/0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 root</td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03 events/1</td>
<td></td>
</tr>
</tbody>
</table>