WaveScript Benchmarks Performance Report

September 8, 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: 3560

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

```sh
## Real or User time for each benchmark/backend
## LD_PRELOAD:
## NOSUDO:
## NICE:
Benchmark mlton03 c2boehm c2boehmsingliste c2 c2seglist c2def c2defseglist
just_timer 2500.000 2516.000 2520.000 2516.000 2516.000 5020.000 5024.000
readfile_bigwins 3800.000 488.000 1064.000 764.000 3980.000 276.000 996.000
printing_lists 2468.000 904.000 904.000 852.000 876.000 824.000 820.000
conv_SigsegArr 2316.000 412.000 7508.000 788.000 5640.000 32.000 6620.000
fft 132.000 912.000 964.000 804.000 992.000 868.000 892.000
```

Language Shootout:

```sh
## Real or User time for each benchmark/backend
## LD_PRELOAD:
## NOSUDO:
## NICE:
Benchmark c2
fannkuch2 4452.000
```
Application Benchmarks:

## Real or User time for each benchmark/backend
## LD_PRELOAD:
## NOSUDO:
## NICE:
Benchmark mltonO3 c2boehm c2boehmseglist c2 c2seglist c2def c2defseglist
## Running orig marmot phase 1
run_first_phase 7436.000 12353.000 4316.000 7716.000 5712.000 8313.000 3860.000
## Running marmot2
test_marmot2 2328.000 5244.000 5248.000 4712.000 4652.000 4564.000 4556.000
## Running marmot3
test_heatmap 7788.000 3204.000 3220.000 2516.000 2512.000 3192.000 3204.000
## Running marmot multinode offline
run_3phases 9533.000 6128.000 4932.000 5696.000 5324.000 5760.000 4828.000

B Appendix: Additional system information

Top results before running benchmarks:

top - 06:23:48 up 48 days, 14:57, 3 users, load average: 2.04, 2.01, 1.78
Tasks: 167 total, 3 running, 164 sleeping, 0 stopped, 0 zombie
Cpu(s): 27.8%us, 4.3%sy, 0.8%ni, 66.1%id, 0.1%wa, 0.4%hi, 0.4%si, 0.0%st
Mem: 2073956k total, 1561120k used, 512836k free, 115340k buffers
Swap: 14996668k total, 34740k used, 14961928k free, 1028332k 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>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>13870</td>
<td>newton</td>
<td>25</td>
<td>0</td>
<td>34236</td>
<td>31m</td>
<td>492 R</td>
<td>99</td>
<td>1.6</td>
<td>2860:01</td>
<td>ikarus</td>
</tr>
<tr>
<td>1 root</td>
<td></td>
<td>18</td>
<td>0</td>
<td>2948</td>
<td>1856</td>
<td>532 S</td>
<td>0</td>
<td>0.1</td>
<td>0:05:50</td>
<td>init</td>
</tr>
<tr>
<td>2 root</td>
<td></td>
<td>11</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:00</td>
<td>kthreadd</td>
</tr>
<tr>
<td>3 root</td>
<td></td>
<td>11</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:04</td>
<td>migration/0</td>
</tr>
<tr>
<td>4 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>migration/0</td>
</tr>
<tr>
<td>5 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:01</td>
<td>migration/0</td>
</tr>
<tr>
<td>6 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:01</td>
<td>migration/0</td>
</tr>
<tr>
<td>7 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:01</td>
<td>migration/0</td>
</tr>
<tr>
<td>8 root</td>
<td></td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>events/0</td>
</tr>
<tr>
<td>9 root</td>
<td></td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>events/0</td>
</tr>
<tr>
<td>10 root</td>
<td></td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>events/0</td>
</tr>
<tr>
<td>11 root</td>
<td></td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>events/0</td>
</tr>
<tr>
<td>31 root</td>
<td></td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>events/0</td>
</tr>
</tbody>
</table>

Top results after running benchmarks:

top - 06:42:50 up 48 days, 15:16, 3 users, load average: 2.01, 2.04, 1.95
Tasks: 167 total, 2 running, 165 sleeping, 0 stopped, 0 zombie
Cpu(s): 27.8%us, 4.3%sy, 0.8%ni, 66.1%id, 0.1%wa, 0.4%hi, 0.4%si, 0.0%st
Mem: 2073956k total, 1043004k used, 96728k buffers
Swap: 14996668k total, 34740k used, 14961928k free, 544536k 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>%CPU</th>
<th>%MEM</th>
<th>TIME+</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>13870</td>
<td>newton</td>
<td>25</td>
<td>0</td>
<td>34212</td>
<td>31m</td>
<td>492 R</td>
<td>101</td>
<td>1.6</td>
<td>2878:56</td>
<td>ikarus</td>
</tr>
<tr>
<td>1 root</td>
<td></td>
<td>18</td>
<td>0</td>
<td>2948</td>
<td>1856</td>
<td>532 S</td>
<td>0</td>
<td>0.1</td>
<td>0:05:51</td>
<td>init</td>
</tr>
<tr>
<td>2 root</td>
<td></td>
<td>11</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:00</td>
<td>kthreadd</td>
</tr>
<tr>
<td>3 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>migration/0</td>
</tr>
<tr>
<td>4 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>migration/0</td>
</tr>
<tr>
<td>5 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>migration/0</td>
</tr>
<tr>
<td>6 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>migration/0</td>
</tr>
<tr>
<td>7 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>migration/0</td>
</tr>
<tr>
<td>8 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>migration/0</td>
</tr>
<tr>
<td>9 root</td>
<td></td>
<td>34</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>migration/0</td>
</tr>
<tr>
<td>10 root</td>
<td></td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>events/0</td>
</tr>
<tr>
<td>11 root</td>
<td></td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>events/0</td>
</tr>
<tr>
<td>12 root</td>
<td></td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>events/0</td>
</tr>
<tr>
<td>31 root</td>
<td></td>
<td>10</td>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>0 S</td>
<td>0</td>
<td>0.0</td>
<td>0:00:03</td>
<td>events/0</td>
</tr>
</tbody>
</table>