Variability: A Performance Nightmare

Allan Porterfield
RENCI, UNC-Chapel Hill
My Interests

Compiler Optimization – improve application performance
Adaptive Runtime Scheduling – reduce application energy demands

Variability make both of these HARD (impossible?).

Chips are different (center of the wafer vs edge)
Temperature changes both time and energy required
BIOS settings change time and energy required
Random latencies from other system load (network, file …)
Butterfly effect on adaptive algorithms
Autotuning Problems

<table>
<thead>
<tr>
<th>Run #</th>
<th>nofuse</th>
<th>nofuseX16</th>
<th>nofuseX32</th>
<th>nofuseX64</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.333394</td>
<td>5.147411</td>
<td>5.086881</td>
<td>5.304340</td>
</tr>
</tbody>
</table>

Autotuning results from Jacobi-2D from PolyBench
Dell M620 with 2 Intel E5-2680 @ 2.7GHz
best 4 results from over a thousand compiler configurations

Looks like tiling improves cache utilization until a cache size is exceeded at which point it falls almost 6%
### Autotuning Problems

<table>
<thead>
<tr>
<th>Run #</th>
<th>nofuse</th>
<th>nofuseX16</th>
<th>nofuseX32</th>
<th>nofuseX64</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.333394</td>
<td>5.147411</td>
<td><strong>5.086881</strong></td>
<td>5.304340</td>
</tr>
<tr>
<td>2</td>
<td>5.144302</td>
<td><strong>5.077856</strong></td>
<td>5.125735</td>
<td>5.138848</td>
</tr>
<tr>
<td>3</td>
<td>5.369150</td>
<td>5.345376</td>
<td><strong>5.203415</strong></td>
<td>5.314528</td>
</tr>
<tr>
<td>4</td>
<td><strong>5.037441</strong></td>
<td>5.115155</td>
<td>5.179577</td>
<td>5.340713</td>
</tr>
<tr>
<td>5</td>
<td>5.408004</td>
<td>5.333346</td>
<td>5.156341</td>
<td><strong>5.083266</strong></td>
</tr>
<tr>
<td>6</td>
<td>5.242719</td>
<td>5.217077</td>
<td>5.373121</td>
<td><strong>5.022142</strong></td>
</tr>
<tr>
<td>7</td>
<td><strong>5.055487</strong></td>
<td>5.156706</td>
<td>5.199737</td>
<td>5.084126</td>
</tr>
<tr>
<td>Average</td>
<td>5.22721</td>
<td>5.19899</td>
<td>5.18926</td>
<td><strong>5.18399</strong></td>
</tr>
</tbody>
</table>

But on multiple runs a different story appears

– **Original answer sub-optimal**
Better Autotuning?

Make one pass of over all of the options (may be thousands)

Identify the fastest options (within 10% (?) of the absolute fastest)
(hopefully single digits versions)

Rerun subset 10+ times to find best average
Scheduling problem

ADCIRC – storm surge simulation
- synchronous SPMD with global synchronization each time step
Test on 6 Nodes Intel E5-2450 @ 2.1 GHz
10 runs on each region – same input used for all tests
Test ran approximately 1 hour – about 20% of production run

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
<th>Range</th>
<th>Slowdown</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-5</td>
<td>3419</td>
<td>3898</td>
<td>3523 (3481)</td>
<td>14(3.8)%</td>
<td>n/a</td>
</tr>
<tr>
<td>6-11</td>
<td>3495</td>
<td>3594</td>
<td>3553</td>
<td>2.8%</td>
<td>2.1%</td>
</tr>
<tr>
<td>12-17</td>
<td>3614</td>
<td>3743</td>
<td>3677</td>
<td>3.5%</td>
<td>5.6%</td>
</tr>
<tr>
<td>18-23</td>
<td>3434</td>
<td>3597</td>
<td>3500</td>
<td>4.7%</td>
<td>0.5%</td>
</tr>
<tr>
<td>24-29</td>
<td>3489</td>
<td>3573</td>
<td>3529</td>
<td>2.4%</td>
<td>1.3%</td>
</tr>
</tbody>
</table>

Average and Slowdown ignore one very slow test on region 0-5
Better SPMD Scheduling?

Rank nodes within a cluster according to their “slowest” processor for Intel processors – probably the amount of time a node spends in TurboBoost

Try to schedule jobs on nodes on groups of nodes. ignores network demand issues which can outweigh SPMD synchronization delays
Energy Usage Problem

ADCIRC – storm surge simulation
16 Nodes of Intel E5-2450 @ 2.1 GHz
10 identical executions – sorted by time
>10% between chips
Better Energy Usage?

Even socket energy demands – by reducing use by ‘expensive’ sockets

Schedule less work
  idle one or more cores
  use DVFS/DCM to reduce clock rate of cores
Problem with Understanding Results

100 executions (in order) of HPCCG
2 socket Intel E5-2650 @2.3GHz

Which run used?
System design evident
-- Socket 0 blue-brown lines
-- Socket 1 yellow-green lines
Air blows over Socket 1 then Socket 0, which runs measurable hotter
100 runs of test Castro AMR execution on 2 different days
input- inputs.2d.cyl_in_cartcoords
4 Nodes Intel E5-2450 @ 2.1 GHz
Same nodes, executable and inputs used both days

<table>
<thead>
<tr>
<th>Date</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/18/2016</td>
<td>18.56</td>
<td>34.51</td>
<td>25.11</td>
</tr>
<tr>
<td>4/19/2016</td>
<td>16.51</td>
<td>19.13</td>
<td>17.71</td>
</tr>
</tbody>
</table>

Differences
- Slurm Queue 4/18 ~4000 run as root
  basically every core in the other nodes of
  the bladecenter being used executing a
  Genetics workflow
- Slurm Queue 4/19 ~680 run as user
  only half of nodes in bladecenter busy
  executing ADCIRC

During debugging could repeat once but vanished during efforts to understand the cause
Easier to trust results?

Papers need to describe execution environment
  Temperature
  Other workload on system
  ...

Graphs that show the variance
  Candlestick, Whisker …