Parallel Performance Visualization on the Cray XT4

Luiz DeRose
Programming Environment Director
Cray Inc.
ldr@cray.com
Cray Apprentice²

- Call graph profile
- Communication statistics
- Time-line view
  - Communication
  - I/O
- Activity view
- Pair-wise communication statistics
- Text reports
- Source code mapping

Cray Apprentice² is target to help and correct:
- Load imbalance
- Excessive communication
- Network contention
- Excessive serialization
- I/O Problems
Statistics Overview

Switch Overview display
Load Balance View (Aggregated)

Min, Avg, and Max Values

-1, +1 Std Dev marks
Call Graph View
Call Graph View

Zoom
Call Graph View - Zoom

Width ⇔ inclusive time
Height ⇔ exclusive time

Load balance overview:
Heigh ⇔ Max time
Left bar ⇔ Average time
Right bar ⇔ Min time
Call Graph View - Zoom
Call Graph View – Function List

Mouse right click:
- hide node
- hide children

Function List off
Call Graph Hide Children

hidden children
Call Graph Unhide One Level
Call Graph Unhide One Level (2)
Call Graph Unhide One Level (3)
Call Graph Unhide All Children
Load Balance View (from Call Graph)

Min, Avg, and Max Values

-1, +1 Std Dev marks
In order to get a screendump, right click on the “File” menu and select “Screendump”.

List of functions and regions sorted by load imbalance (potential savings). In order to get this list in the call graph, click on the “>>” button and right click the call graph tab and select “potential savings”. This call graph is filtered to show only the functions that take a reasonable amount of time.

Load balance display. The height of the box represents the maximum time. The left bar is proportional to the average time. The right bar is proportional to the minimum time. Functions with load imbalance have a lot of yellow on the left side.

Main functions showing load imbalance. Normally load balance in MPI functions is caused by some load imbalance in computation. In this case, on “Calc3”, which has an artificial imbalance.
Source Mapping from Call Graph

```c
165  c  angle pipelining loop (batches of nmi angles)
166  c
167  DO  mo = 1, nmo
168  nmo = (nmo-1)*nmi
170  c  K-inflows (k=0 boundary)
172  c
173  IF (k2<1.0 .or. kbc.eq.0) THEN
174     m1 = 1, nmi
175     DO  j = 1, jk
176     DO  i = 1, ik
177     phi(kb(i,j,m1) = 0.0d=0
178     END DO
179     END DO
180     END DO
181     ELSE
182     IF (co_dsa) THEN
183         leak = 0.0
184         k = k0 - k2
185         DO  m1 = 1, nmi
186         m = m1 + nmo
187         DO  j = 1, jk
188         DO  i = 1, ik
189         phi(kb(i,j,m1) = phi(kbc(i,j,m)
190         leak = leak + w(min) * phi(kb(1,j,m1)) * di(i) * di(j)
191         & face[i,j,k+k3,3] = face[i,j,k+k3,3]
192         & + w(min) * phi(kb(1,j,m1))
193         END DO
194         END DO
195         END DO
196         leakage(5) = leakage(5) + leak
197     END IF
198     END IF
```

September 18-20, 2007  Luiz DeRose (ldr@cray.com) © Cray Inc.  Slide 72
### Function Profile

<table>
<thead>
<tr>
<th>Time</th>
<th>Period</th>
<th>Hits</th>
<th>Calls</th>
<th>Time Balance</th>
<th>Function</th>
<th>Line</th>
<th>Fnr</th>
</tr>
</thead>
<tbody>
<tr>
<td>124.17</td>
<td>1852</td>
<td>757</td>
<td>1</td>
<td>5.63</td>
<td>ssh</td>
<td>116</td>
<td>0.15</td>
</tr>
<tr>
<td>40.02</td>
<td>17776</td>
<td>116800</td>
<td>1</td>
<td>23.40</td>
<td>print</td>
<td>0</td>
<td>0.25</td>
</tr>
<tr>
<td>18.57</td>
<td>15527</td>
<td>45</td>
<td>1</td>
<td>48.26</td>
<td>exit</td>
<td>35</td>
<td>0.50</td>
</tr>
<tr>
<td>0.17</td>
<td>23899</td>
<td>1506</td>
<td>3</td>
<td>50.00</td>
<td>sqlite</td>
<td>0</td>
<td>0.12</td>
</tr>
<tr>
<td>2.87</td>
<td>25039</td>
<td>575</td>
<td>1</td>
<td>2.62</td>
<td>source</td>
<td>18</td>
<td>0.00</td>
</tr>
<tr>
<td>1.94</td>
<td>4630</td>
<td>144</td>
<td>1</td>
<td>2.59</td>
<td>mpi_vsnd</td>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>0.86</td>
<td>9589</td>
<td>192</td>
<td>2</td>
<td>2.47</td>
<td>mpi_bcast</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>0.41</td>
<td>5281</td>
<td>575</td>
<td>1</td>
<td>2.60</td>
<td>flt_earr</td>
<td>17</td>
<td>0.00</td>
</tr>
<tr>
<td>0.38</td>
<td>118000</td>
<td>19</td>
<td>1</td>
<td>10.98</td>
<td>std_real</td>
<td>135</td>
<td>0.00</td>
</tr>
<tr>
<td>0.23</td>
<td>772</td>
<td>308</td>
<td>1</td>
<td>95.76</td>
<td>fwrite</td>
<td>36</td>
<td>0.07</td>
</tr>
<tr>
<td>0.10</td>
<td>665</td>
<td>11090</td>
<td>2</td>
<td>17.30</td>
<td>rcv_real</td>
<td>164</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>7087</td>
<td>4</td>
<td>1</td>
<td>4.56</td>
<td>intinit_r</td>
<td>42</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>5405</td>
<td>4</td>
<td>1</td>
<td>4.59</td>
<td>init_r</td>
<td>77</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>1347</td>
<td>4</td>
<td>1</td>
<td>23.84</td>
<td>inner</td>
<td>72</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>9498</td>
<td>1</td>
<td>1</td>
<td>0.20</td>
<td>faplct</td>
<td>35</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>602</td>
<td>65</td>
<td>1</td>
<td>0.20</td>
<td>gto</td>
<td>36</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>4992</td>
<td>2</td>
<td>1</td>
<td>25.14</td>
<td>octane</td>
<td>17</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>4547</td>
<td>576</td>
<td>1</td>
<td>18.79</td>
<td>global_realm_main</td>
<td>221</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>3959</td>
<td>40</td>
<td>1</td>
<td>68.55</td>
<td>main</td>
<td>72</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>15509</td>
<td>40</td>
<td>1</td>
<td>98.10</td>
<td>global_realm_main</td>
<td>373</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>13353</td>
<td>40</td>
<td>1</td>
<td>10.23</td>
<td>internal</td>
<td>63</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>3362</td>
<td>48</td>
<td>1</td>
<td>37.74</td>
<td>task_init</td>
<td>26</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>3389</td>
<td>364</td>
<td>2</td>
<td>27.97</td>
<td>global_realm_main</td>
<td>347</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>3652</td>
<td>2</td>
<td>1</td>
<td>0.00</td>
<td>fopen</td>
<td>105</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>4549</td>
<td>48</td>
<td>1</td>
<td>7.54</td>
<td>initmain</td>
<td>175</td>
<td>0.00</td>
</tr>
</tbody>
</table>
Distribution by PE, by Call, & by Time
Environment & Execution Details

<table>
<thead>
<tr>
<th>PE</th>
<th>Total Used (MB)</th>
<th>Total Free (MB)</th>
<th>Largest Free (MB)</th>
<th>Fragments</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>97.26159</td>
<td>4085.525472</td>
<td>4065.59253</td>
<td>1534</td>
</tr>
<tr>
<td>1</td>
<td>95.57198</td>
<td>4067.257146</td>
<td>4067.244414</td>
<td>start</td>
</tr>
<tr>
<td>2</td>
<td>95.59117</td>
<td>4067.256815</td>
<td>4067.251653</td>
<td>end-start</td>
</tr>
<tr>
<td>3</td>
<td>95.57198</td>
<td>4067.257146</td>
<td>4067.244414</td>
<td>start</td>
</tr>
<tr>
<td>4</td>
<td>95.59117</td>
<td>4067.256815</td>
<td>4067.251653</td>
<td>end-start</td>
</tr>
<tr>
<td>5</td>
<td>95.57198</td>
<td>4067.257146</td>
<td>4067.244414</td>
<td>start</td>
</tr>
<tr>
<td>6</td>
<td>95.59117</td>
<td>4067.256815</td>
<td>4067.251653</td>
<td>end-start</td>
</tr>
<tr>
<td>7</td>
<td>95.57198</td>
<td>4067.257146</td>
<td>4067.244414</td>
<td>start</td>
</tr>
<tr>
<td>8</td>
<td>95.59117</td>
<td>4067.256815</td>
<td>4067.251653</td>
<td>end-start</td>
</tr>
<tr>
<td>9</td>
<td>95.57198</td>
<td>4067.257146</td>
<td>4067.244414</td>
<td>start</td>
</tr>
<tr>
<td>10</td>
<td>95.59117</td>
<td>4067.256815</td>
<td>4067.251653</td>
<td>end-start</td>
</tr>
<tr>
<td>11</td>
<td>95.57198</td>
<td>4067.257146</td>
<td>4067.244414</td>
<td>start</td>
</tr>
</tbody>
</table>
Time Line View (Sweep3D)
Time Line View (Zoom)

User Functions, MPI & SHMEM Line

I/O Line
Time Line View (Fine Grain Zoom)
Activity View
Pair-wise Communication
I/O Overview
I/O Traffic Report
I/O Rates

<table>
<thead>
<tr>
<th>FD</th>
<th>Write Calls</th>
<th>Write Tot (MB)</th>
<th>Write Min (MB/s)</th>
<th>Write Avg (MB/s)</th>
<th>Write Max (MB/s)</th>
<th>Read Calls</th>
<th>Read Tot (MB)</th>
<th>Read Min (MB/s)</th>
<th>Read Avg (MB/s)</th>
<th>Read Max (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>37</td>
<td>0.002</td>
<td>0.001</td>
<td>0.039</td>
<td>0.869</td>
<td>0</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>2</td>
<td>48</td>
<td>0.001</td>
<td>0.001</td>
<td>0.003</td>
<td>0.136</td>
<td>0</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>2</td>
<td>0.000</td>
<td>0.039</td>
<td>0.045</td>
<td>0.051</td>
</tr>
</tbody>
</table>
Hardware Counters Time Line
Controlling Trace File Size

- Several environment variables are available to limit trace files to a reasonable size:
  - PAT_RT_CALLSTACK
    - Limit the depth to trace the call stack
  - PAT_RT_HWPC
    - Avoid collecting hardware counters (unset)
  - PAT_RT_RECORD_PE
    - Collect trace for a subset of the PEs
  - PAT_RT_TRACE_FUNCTION_ARGS
    - Limit the number of function arguments to be traced
  - PAT_RT_TRACE_FUNCTION_LIMITS
    - Avoid tracing indicated functions
  - PAT_RT_TRACE_FUNCTION_MAX
    - Limit the maximum number of traces generated for all functions for a single process

- Use CrayPat API to toggle trace state (on / off) or to select functions to trace
  - int PAT_tracing_state (int state)
  - int PAT_trace_function (const void *addr, int state)

- Use the limit built-in command for ksh(1) or csh(1) to control how much disk space the trace file can consume
Additional API Functions

- `int PAT_profiling_state (int state)`
- `int PAT_record (int state)`
- `int PAT_sampling_state (int state)`
- `int PAT_tracing_state (int state)`
- `int PAT_trace_function (const void *addr, int state)`
- State can have one of the following:
  - PAT_STATE_ON
  - PAT_STATE_OFF
  - PAT_STATE_QUERY
- `int PAT_flush_buffer (void)`
Parallel Performance Visualization on the Cray XT4

Questions / Comments
Thank You!