Memory Throttling on BG/Q: A Case Study with Explicit Hydrodynamics

HotPower
October 5, 2014

Bo Li, Virginia Tech
Edgar A. Leon, Lawrence Livermore National Laboratory

LLNL-PRES-662081
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC
Future systems constrained by power

- NNSA’s Sequoia: 16+ petaflops, 7.9 MW
- Exascale target: 20 MW
- Power – large part of ownership cost
- Unable to use all compute resources at once
- Implications for app. and system developers
  - Optimizing for power/energy and runtime
- Computing under a power bound
  - Selectively allocating power on a fixed budget
Selective throttling to maximize performance under a power budget

- Shift power to resources on the critical path
  - Characterize code phases/regions/physics
  - Leverage IBM Blue Gene/Q’s memory throttling
  - Minimize impact on time-to-solution

- Contributions
  - Significant power shifting opportunities in explicit hydrodynamics
  - First to employ real throttling on a supercomputer
  - Linear regression model to guide throttling
  - Apply throttling on a code region basis
LULESH
Livermore Unstructured Lagrange Explicit Shock Hydrodynamics

- Shock-hydro mini-app
  - Lagrange hydrodynamics
  - Solves Sedov problem
  - Unstructured hex mesh
  - Single material
  - Ideal gas EOS
IBM Blue Gene/Q (BG/Q)

- Low power architecture
  - PowerPC A2 1.6 GHz cores
  - DDR3 1.33 GHz memory

- Memory throttling
  - Built-in on the DDR controller
  - Adds *idle cycles* between each read/write to DDR
  - Idle cycles from 0 to 128
  - Node granularity

- Power and energy
  - EMON2 high-resolution
  - Measure current and voltage up to 2KHz
  - 7 domains: *core* logic, *memory*, network...
  - Board granularity, 32 nodes
75% of overall power is static

- Component-specific static proportion
  - 73% for core
  - 53% for memory

- Throttling affects dynamic draw
Reduced memory bandwidth with increased throttling

STREAM benchmark

![Graph showing reduced memory bandwidth with increased throttling](chart.png)
Impact of throttling is region-dependent

<table>
<thead>
<tr>
<th>Region</th>
<th>IPC</th>
<th>Runtime</th>
<th>MemBW</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>0.541</td>
<td>11.85 s</td>
<td>18.38 GB/s</td>
</tr>
<tr>
<td>R2</td>
<td>0.554</td>
<td>32.02 s</td>
<td>15.56 GB/s</td>
</tr>
<tr>
<td>R3</td>
<td>0.216</td>
<td>1.10 s</td>
<td>20.88 GB/s</td>
</tr>
<tr>
<td>R4</td>
<td>0.654</td>
<td>13.80 s</td>
<td>9.12 GB/s</td>
</tr>
<tr>
<td>R5</td>
<td>0.321</td>
<td>16.21 s</td>
<td>13.72 GB/s</td>
</tr>
</tbody>
</table>

Effect of throttling on LULESH performance
Optimal memory speed is a function of region, size, and concurrency

- **Optimal**: minimum speed with no effect on performance
- Increased concurrency, higher demand for memory BW
- R1-R3 more sensitive to memory BW

![Graph showing optimal DDR idle cycles for different problem sizes and number of threads.](image)

![Graph showing optimal DDR idle cycles for Region 4 with different sizes and number of threads.](image)
Predicting optimal memory speed

- Linear regression model based on HW counters
  - Num. instructions, CPU cycles, L1 and L2 misses, main memory loads/stores, etc.

\[ f_{\text{min}} = \sum_{i=1}^{N} w_i \times \frac{\text{counter}_i}{\text{CPUcycles}} \]

Regression Model (R Squre = 0.67)

- Optimal DDR Idle Cycles
- Training Data Samples
- Predicted vs Observed
Our model predictions decrease performance by 3% in most cases.
Impact of memory throttling on power

Region 1

Region 4
Up to 20% less dynamic power with a 3% performance loss
Summary and conclusions

- Demonstrated significant opportunities for power shifting in explicit hydrodynamics
- Leveraged BG/Q’s memory throttling
  - Up to 20% dynamic power savings
- Employed a regression model to throttle memory
  - Low performance degradation for most configurations
  - Some inaccuracies with low concurrency
  - Not predictive of non-linear interactions
- Future work
  - Analyze a representative suite of HPC applications
  - Investigate machine learning techniques such as ANNs