Big Data in HPC Applications and Programming Abstractions

Saba Sehrish
Oct 3, 2012
## Big Data in Computational Science - Size

Data requirements for select 2012 INCITE applications at ALCF (BG/P)

<table>
<thead>
<tr>
<th>Project</th>
<th>On-line Data (TBytes)</th>
<th>Off-line Data (TBytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supernovae Astrophysics</td>
<td>100</td>
<td>400</td>
</tr>
<tr>
<td>Combustion in Reactive Gases</td>
<td>1</td>
<td>17</td>
</tr>
<tr>
<td>CO2 Absorption</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>Seismic Hazard Analysis</td>
<td>600</td>
<td>100</td>
</tr>
<tr>
<td>Climate Science</td>
<td>200</td>
<td>750</td>
</tr>
<tr>
<td>Energy Storage Materials</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Stress Corrosion Cracking</td>
<td>12</td>
<td>72</td>
</tr>
<tr>
<td>Nuclear Structure and Reactions</td>
<td>6</td>
<td>30</td>
</tr>
<tr>
<td>Reactor Thermal Hydraulic Modeling</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Laser-Plasma Interactions</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>Vaporizing Droplets in a Turbulent Flow</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Top 10 data producer/consumers instrumented with Darshan over the month of July, 2011. Surprisingly, three of the top producer/consumers almost exclusively read existing data.
Big Data in Computational Science - Complexity

Complexity is an artifact of science problems and codes:

Complexity in data models - multidimensional, hierarchical, tree-based, graph-based, mesh-oriented, multi-component data sets

Coupled multi-scale simulations generate multi-component datasets consisting of materials, fluid flows, and particle distributions.
Challenges we face in the I/O World

• We are looking at capacity but smart ways to manage the capacity to deal with not only size but complexity

• How are these data sets generated, which we need to store – scientific simulations, observations/experiments/sensors

• How to store and retrieve data – the I/O libraries

• What to store - useful data

• What data formats – self describing data

• What data layouts – optimized way of data retrieval
What I/O Programming Abstraction Options to use?

- Three Options
  - Use existing programming abstractions and I/O frameworks
  - Extend/Leverage these models
  - Develop New models

- Existing I/O programming abstractions for I/O in science — MPI-IO, PnetCDF, HDF5, ADIOS
- Abstractions in general for Big data: MapReduce (Hadoop)

- Extend/Leverage: RFSA, MRAP

- New: DAMSEL (incorporates data model of application into file formats and data layouts for exascale science)
Our Contributions

- Leverage Hadoop framework to understand scientific data formats and optimizations to improve performance
- Provide optimizations, etc for HPC applications with big data through RFSA
- Develop a new data model based I/O library
MRAP – MapReduce with Access Patterns

- MapReduce and the distributed file systems’ applicability to HPC
- Successfully used with web applications at Yahoo!, Google, Facebook, etc
- Can it meet the requirements of I/O intensive HPC applications?
  - Yes - because of a resilient framework that allows large scale data processing.
  - No - because access patterns in traditional HPC applications do not match directly with MapReduce splits.
- In MRAP - we add these HPC data semantics to the MapReduce framework
MRAP Design

1. APIs and templates to specify the access patterns e.g. non-contiguous access patterns, matching patterns
MRAP Optimizations

2. MRAP Data restructuring to organize data before hand to avoid/minimize data movement and remote data access
MRAP Optimizations

3. MRAP Scheduling to improve data locality using a weighted set cover-based approach and virtual splits

Independent Data chunks:
1) Map task (MT) 1 and MT 2 are scheduled on Nodes C and D.
2) MT 1 needs chunks 1 and 2, and MT 2 needs chunks 3 and 4.
3) Node C is assigned MT1 and Node D is assigned MT2.
4) Node D has chunks 1 and 3 and Node C has chunks 2 and 4. Both Nodes are missing a required chunk for their map task.
5) Virtual splits will be created for both map tasks because chunks are independent. Virtual splits improve performance by maximizing the number of local I/O requests.

Dependent Data chunks:
1) Chunks 1, 2, 3 are required by a map task.
2) Node D has two chunks, whereas Nodes A, B and C have the remaining chunk needed (chunk 2).
3) Our scheduler will determine the node with minimal latency using weighted set cover approach. E.g. Node D becomes the node that will process the map task. If Node C has the lowest latency, then chunk 2 will be transferred to Node D from Node C.
Performance Evaluations

MRAP API

MRAP Data restructuring

MRAP Scheduling
RFSA – A reduced function set abstraction for MPI-IO

• Ways to improve MPI-IO functions
  • Programmer productivity
    • Reducing number of I/O calls e.g. by automatically choosing which read/write function to choose
  • Performance
    • Optimizing locking mechanism by proposing a conflict detection algorithm
    • Optimizing collective I/O by a pipelining mechanism to overlap communication and I/O
Performance Evaluation

Optimizing locking mechanism

Automating selection of I/O calls

Improving collective I/O performance
DAMSEL

- Provide a set of API functions to support sophisticated data models e.g. block structured AMR, geodesic grid, etc
- Enable exascale computational science applications to interact conveniently and efficiently with the storage through data model API
- Develop a data model based storage library and provide efficient storage layouts
DAMSEL Example

- The FLASH is a modular, parallel multi-physics simulation, developed at University of Chicago
- Uses a structured adaptive-mesh refinement grid
  ✶ The problem domain is hierarchically partitioned into blocks of equal sizes (in array elements)
Summary

- Too much described in very less time
- I/O Abstractions for Big data HPC applications
  - MRAP
    - Based on MapReduce
  - RFSA
    - Based on MPI-IO
  - DAMSEL
    - Based on data models of computational applications
Acknowledgements

• University of Central Florida
  • Advisor: Jun Wang
  • Grant Mackey

• Northwestern University
  • Alok Choudhary
  • Wei-keng Liao

• Argonne National Laboratory
  • Rajeev Thakur
  • Rob Ross
  • Rob Latham

• Los Alamos National Laboratory/EMC
  • John Bent
Questions

Saba Sehrish
ssehrish@eecs.northwestern.edu