Synergistic Challenges in Data-Intensive Science and Extreme Scale Computing

Vivek Sarkar
Department of Computer Science
Rice University
vsarkar@rice.edu

NSF workshop on Big Data and Extreme-scale Computing (BDEC)
April 30 - May 1, 2013
Outline

1. DOE ASCAC Subcommittee report on Synergistic Challenges in Data-Intensive Science and Exascale Computing

2. Selected Research Topics in Big Data and Extreme Scale
ASCAC Subcommittee Report

Available via “Relevant Background Material” link in BDEC workshop web site

Synergistic Challenges in Data-Intensive Science and Exascale Computing

DOE ASCAC Data Subcommittee Report
March 2013
## ASCAC Subcommittee Members

<table>
<thead>
<tr>
<th>Last name</th>
<th>First name</th>
<th>Affiliation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chen (*)</td>
<td>Jacqueline</td>
<td>Sandia</td>
</tr>
<tr>
<td>Choudhary</td>
<td>Alok</td>
<td>Northwestern U.</td>
</tr>
<tr>
<td>Feldman</td>
<td>Stuart</td>
<td>Google</td>
</tr>
<tr>
<td>Hendrickson</td>
<td>Bruce</td>
<td>Sandia</td>
</tr>
<tr>
<td>Johnson</td>
<td>Chris</td>
<td>U. Utah</td>
</tr>
<tr>
<td>Mount</td>
<td>Richard</td>
<td>SLAC</td>
</tr>
<tr>
<td>Sarkar (**)</td>
<td>Vivek</td>
<td>Rice U.</td>
</tr>
<tr>
<td>White (*)</td>
<td>Victoria</td>
<td>FermiLab</td>
</tr>
<tr>
<td>Williams (*)</td>
<td>Dean</td>
<td>LLNL</td>
</tr>
</tbody>
</table>

(*) ASCAC member,  (**) Subcommittee chair
Our Charge

By this letter, I am charging the ASCAC to examine the potential synergies between the challenges of data-intensive science and exascale. The subcommittee should take into account the Department’s mission needs, which define the Office of Science’s unique role in data-intensive science vis-à-vis other agencies.
Data Challenges in Science

Overall trend: most science domains will become data-intensive in the exascale timeframe (and many well before then)

Source: notional figure, courtesy of Kathy Yelick

Source: Bill Harrod, SC12 plenary presentation
Data Challenges in High Energy Physics: Large Hadron Collider exemplar

• ATLAS and CMS detectors generate analog data at rates equivalent to 1PB/second
• Output rate after *data reduction* is 1GB/second ~ 10PB/year
• Storage of cumulative derived data, simulated data, replicated data is currently ~ 100PB, and is rapidly increasing
• Workflow: homogeneous community of physicists access read-only shared data using the Worldwide LHC Computing Grid (WLCG)
Data Challenges in Climate Science

Federated data enterprise system with significant challenges

- Velocity: distributing live data streams and large volume data movement quickly and efficiently
- Volume: analyzing large-volume data in-place for big data analytics
- Heterogeneous workflows: on-demand data products for heterogeneous communities (scientists, policy makers, farmers, insurance industry, …)

Earth System Grid Federation (ESGF) manages several petabytes of data
Data Challenges in Large-Scale Simulations: S3D Combustion code exemplar

- Goal: simulate turbulence-chemistry interaction at conditions that are representative of realistic systems
  - High pressure
  - Turbulence intensity
  - Turbulent length scales
  - Sufficient chemical fidelity to differentiate effects of fuels
- Exascale simulation will require 3PB of memory, and will generate 400PB of raw data (1PB every 30 minutes)
- Workflow challenges include co-design for simulation and in-situ analyses
Data Challenges in Biology and Genomics: KBase exemplar

- Data-intensive challenges include
  - Biophysical simulations of cellular environments
  - Cracking the ‘signaling code’ of the genome across the tree of life (reconstruction of cellular networks across species)
  - Reverse engineering the human brain

- KBase center currently manages about 2 petabytes of data (plant genomes, process data) for genomics research; workflow based on a service-based infrastructure

- Significant differences between data characteristics in Kbase and other domains (lots of integer data, random access, large intermediate data size during computations, poor locality in cross-correlation, …)
Data Challenges in Light Sources: APS and LCLS exemplars

Advanced Photon Source (APS)

- Includes about 65 beam lines, with ~ 1TB of data generated per day
  - Future light sources are expected to generate data at the rate of 1TB per second
- GridFTP and GlobusOnline services help some APS users with their workflow, but many others bring their own storage devices and perform manual analysis of their data

Linac Coherent Light Source (LCLS)

- Provides users access to ~ 2.5PB storage facility via LCLS portal, where data is stored for 2 years, and an on-line cache of ~ 50TB, where data is stored for 5 days.
- These volumes are expected to increase dramatically in the future
Data Analysis and Visualization: From Big Data to “Big Information”

“Hence a wealth of information creates a poverty of attention and a need to allocate that attention efficiently among the overabundance of information sources that might consume it.”

--- Herbert Simon, Designing Organizations for an Information-Rich World

• Widening gap between I/O and computational rates will make in-situ analysis & visualization a necessity for exascale
Topological Analysis & Volume Visualization of Combustion simulation
Data Streaming and Near-Sensor Computing

Data Streaming exemplar
- ORNL Spallation Neutron Source (SNS)
- Challenge: reduction and visualization of some of the large SNS data sets take hours after data has been collected
- ADARA streaming data system provides in-situ reduction of data as it is generated from the instrument
  - Challenges in in-situ reduction is synergistic with data movement challenges in exascale computing

Near-sensor computing exemplars
- HEP, radio telescopes, light sources, …
- Triggers detect events of interest to be recorded
- Filters reduce data as close to the instrument as possible
- After data has been reduced by triggers and filters, it is curated and archived for re-processing and re-analysis
Intertwined requirements for Big Data and Extreme-scale Computing

- Big Data generated by the data-driven paradigm will need to be analyzed by Extreme-scale Computing
  - “Extreme-scale systems” refer to all classes of systems built in ~ 2020 timeframe or later

- Extreme-scale Computing will generate Big Data
  - Data-intensive simulations on large Extreme-scale Computers will generate volumes of Big Data comparable to data generated by the largest science experiments

- Data-driven and data-intensive approaches have evolved somewhat independently of each other
  - Important for each to learn lessons from the other because their fates are intertwined
Recommendations

Recommendation 1: The DOE Office of Science should give high priority to investments that can benefit both data-intensive science and exascale computing so as to leverage their synergies.

- For science domains that need exascale simulations, commensurate investments in exascale computing capabilities and data infrastructure are necessary.
- In other domains, extreme-scale components of exascale systems are necessary for near-sensor computing and other tiers of data analysis.
- Research in algorithms to address fundamental challenges in concurrency, data movement, and resilience will benefit data analysis and computational techniques for both data-intensive science and exascale computing.
Recommendations (contd)

Recommendation 2: DOE ASCR should give high priority to research and other investments that simplify the science workflow and improve the productivity of scientists involved in exascale and data-intensive computing.

• Recommend paying greater attention to simplifying human-in-the-loop workflows for data-intensive science.
  • Virtual Data Facility (VDF) should provide a simpler portal for data services than current systems.
• Recommend development of libraries of scalable data analytics and data mining algorithms and software components for use in workflows.
• Recommend creation of new classes of proxy applications to capture the combined characteristics of simulation and analytics to feed into future design/co-design activities.
Recommendation 3: DOE ASCR should adjust investments in programs such as fellowships, career awards, and funding grants, to increase the pool of computer and computational scientists trained in both exascale and data-intensive computing.

- There is a significant gap between the number of current computational and computer scientists trained in both exascale and data-intensive computing and the future needs for this combined expertise in support of DOE’s science missions.
- ASCR investments such as fellowships, career awards, and funding grants should look to increase the pool of computer and computational scientists trained in both exascale and data-intensive computing.
Outline

1. DOE ASCAC Subcommittee report on Synergistic Challenges in Data-Intensive Science and Exascale Computing

2. Selected Research Topics in Big Data and Extreme Scale
Rice Habanero Multicore Software Project: Enabling Technologies for Extreme Scale

Two-level programming model
Declarative Coordination Language for Domain Experts, CnC (Intel Concurrent Collections)
+
Task-Parallel Languages for Parallelism-aware Developers: Habanero-C, Habanero-Java, Habanero-Scala

Portable execution model
1) Lightweight asynchronous tasks and data transfers
   - Creation: async tasks, future tasks, data-driven tasks
   - Termination: finish, future get, await
   - Data Transfers: asyncPut, asyncGet, asyncISend, asyncIRecv
2) Locality control for task and data distribution
   - Task Distributions: hierarchical places
   - Data Distributions: hierarchical places, distributed arrays
3) Inter-task synchronization operations
   - Mutual exclusion: isolated, actors
   - Collective and point-to-point synchronization: phasers, accumulators

Extreme Scale Platforms

http://habanero.rice.edu
Examples of BD-EC Synergies in Habanero Project

1. Data Layout Selection for Portable Performance
2. Asynchronous Collectives via Finish/Phaser Accumulators
3. Latency Tolerance with Event-Driven Tasks
4. Fresh Breeze Storage System
5. Big Data Array Programming Platform (APP)
1. Data Layout Selection for Portable Performance
(joint work with Kamal Sharma, Ian Karlin, Jeff Keasler, Jim McGraw)

Performance improvement for IRSMk relative to default layout and max # threads

<table>
<thead>
<tr>
<th>Platform</th>
<th>Best Manual Layout for platform</th>
<th>Automated Layout for platform</th>
<th>Performance improvement for IRSMk relative to default layout and max # threads</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM POWER 7 (32 threads)</td>
<td>5.02x</td>
<td>4.93x</td>
<td>5.02x</td>
</tr>
<tr>
<td>AMD APU (4 threads)</td>
<td>1.46x</td>
<td>1.43x</td>
<td>1.46x</td>
</tr>
<tr>
<td>IBM BG/Q (64 threads)</td>
<td>2.20x</td>
<td>2.08x</td>
<td>2.20x</td>
</tr>
</tbody>
</table>
2a. Asynchronous Collectives via Finish Accumulators
(joint work with Jun Shirako)

1. accumulator count = accumulator.factory.accumulator(SUM, int.class);
2. finish(count) nqueens_kernel(new int[0], 0);
3. System.out.println(“No. of solutions = “ + count.get().intValue());
4. ... // count.get() receives final value after finish
5. void nqueens_kernel(int [] a, int depth) {
6.   if (size == depth) count.put(1); // Send value asynchronously to count
7.   else
8.     /* try each possible position for queen at depth */
9.     for (int i = 0; i < size; i++) async {
10.        /* allocate a temporary array and copy array a into it */
11.           int [] b = new int [depth+1];
12.           System.arraycopy(a, 0, b, 0, depth);
13.           b[depth] = i;
14.           if (ok(depth+1,b)) nqueens_kernel(b, depth+1);
15.     } // for-async
16. } // nqueens_kernel()
2b. Asynchronous Collectives via Phaser Accumulators  
(joint work with Jun Shirako, David Peixotto, Bill Scherer)

```java
phaser ph = new phaser();
accumulator a = new accumulator(ph, accumulator.SUM, int.class);
accumulator b = new accumulator(ph, accumulator.MIN, double.class);

// foreach creates one task per iteration
foreach (point [i] : [0:n-1]) phased (ph) {
    int iv = 2*i + j;
    double dv = -1.5*i + j;
    a.put(iv);
    b.put(dv);
    // Do other work before next
    next;
}

int sum = a.get().intValue();
double min = b.get().doubleValue();
...  
```

**Allocation:** Specify operator and type

**put:** Send a value to accumulator

**next:** Barrier operation; advance the phase

**get:** Get the result from previous phase (no race condition)
3. Latency Tolerance with Event-Driven Tasks

(joint work with Open Community Runtime team, https://01.org/projects/open-community-runtime)

Hardware multithreading is limited to 2x-8x threads per core

Software multitasking with event-driven tasks (EDTs) can support 1000+ suspended tasks per core

Figure source: “POWER5: IBM’s Next Generation POWER Microprocessor”, Ron Kalla (IBM)
4. Fresh Breeze Storage System
(joint work with Kumud Bhandari, Jack Dennis, Guang Gao)

- Designed for archival/persistent storage
- Data is stored in 128-byte readonly blocks
  - No consistency issues
  - Can be easily accessed in parallel
- Each block has a unique handle (guid)
- Block may contain data or handles
- Data layouts can be tuned for optimization
5. Big Data Array Programming Platform (APP)
(joint work with Chris Jermaine, Zoran Budimlic, Michael Burke)

- Source program = DSL with big data array primitives
- Compiler generates control flow and data dependence graphs assuming dense representations
- Runtime (phase 1) generates query plan after “sparsification” and optimization
  - Leverage metadata and characteristics of actual data
- Runtime (phase 2) executes query plan on big data platform
Outline

1. DOE ASCAC Subcommittee report on Synergistic Challenges in Data-Intensive Science and Exascale Computing

2. Selected Research Topics in Big Data and Extreme Scale