Resource Management for Extreme-Scale Data-Intensive Computing

Ioan Raicu
Computer Science Department, Illinois Institute of Technology
Math and Computer Science Division, Argonne National Laboratory

September 13th, 2013
• Research Focus
  – Emphasize designing, implementing, and evaluating systems, protocols, and middleware with the goal of supporting data-intensive applications on extreme scale distributed systems, from many-core systems, clusters, grids, clouds, and supercomputers

• People
  – Dr. Ioan Raicu (Director)
  – 6 PhD Students
  – 2 MS Students
  – 4 UG Students

• Contact
  – http://datasys.cs.iit.edu/
  – iraicu@cs.iit.edu

Resource Management for Extreme-Scale Data-Intensive Computing
• Today (2013): Multicore Computing
  – $O(10)$ cores commodity architectures
  – $O(100)$ cores proprietary architectures
  – $O(1000)$ GPU hardware threads

• Near future (~2019): Manycore Computing
  ~1000 cores/threads commodity architectures
• **Today (2013):** Petascale Computing
  – $O(100K)$ nodes
  – $O(1M)$ cores

• **Near future (~2018):** Exascale Computing
  – ~1M nodes *(10X)*
  – ~1B processor-cores/threads *(1000X)*
Some Challenges to Overcome at Exascale Computing

• Programming paradigms
  – HPC is dominated by MPI today
  – Will MPI scale another 3 orders of magnitude?
  – Other paradigms (including loosely coupled ones) might emerge to be more flexible, resilient, and scalable

• Storage systems will need to become more distributed to scale ➔ Critical for resilience of HPC

• Network topology must be used in job management, data management, compilers, etc
Critical Technologies Needed to achieve Extreme Scales

- Fundamental Building Blocks (with a variety of resilience and consistency models)
  - Distributed hash tables (DHT)
    - Also known as NoSQL data stores or key/value stores
    - Examples: Chord, Tapestry, memcached, Dynamo, MongoDB, Kademlia, CAN, Tapestry, Memcached, Cycloid, Ketama, RIAK, Maidsafe-dht, Cassandra and C-MPI, BigTable, HBase
  - Distributed message queues (DMQ)
    - Example: SQS, RabbitMQ, Couch RQS, ActiveMQ, KAFKA, Hedwig

Resource Management for Extreme-Scale Data-Intensive Computing
• Global File Systems and Storage
• Job Management Systems
• Workflow Systems
• Monitoring Systems
• Provenance Systems
• Data Indexing
• Relational Databases
Many-Task Computing (MTC)

MTC emphasizes:
- bridging HPC/HTC
- many resources
  - short period of time
- many computational tasks
- dependent/independent tasks
- tasks organized as DAGs
- primary metrics are seconds

Advantages:
- Improve fault tolerant
- Maintain efficiency
- Programmability & Portability
- support embarrassingly parallel and parallel applications
• Swift/T
  o Active research project (CI UChicago & ANL)
  o Parallel Programming Framework
  o Throughput ~25k tasks/sec per process
  o Shown to scale to 128k cores

• Application Domains Supported
  o Astronomy, Biochemistry, Bioinformatics, Economics, Climate

Swift lets you write parallel scripts that run many copies of ordinary programs concurrently, using statements like this:

```python
foreach protein in proteinList {
  runBLAST(protein);
}
```

Images from Swift Case Studies -
http://www.ci.uchicago.edu/swift/case_studies/
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Characteristics</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Astronomy</td>
<td>Creation of montages from many digital images</td>
<td>Many 1-core tasks, much communication, complex dependencies</td>
<td>E</td>
</tr>
<tr>
<td>Astronomy</td>
<td>Stacking of cutouts from digital sky surveys</td>
<td>Many 1-core tasks, much communication</td>
<td>E (Falkon)</td>
</tr>
<tr>
<td>Biochemistry</td>
<td>Analysis of mass-spec data for post-translational protein modifications</td>
<td>10,000 – 100,000 K jobs for proteomic searches using custom serial codes</td>
<td>D</td>
</tr>
<tr>
<td>Biochemistry</td>
<td>Protein folding using iterative fixing algorithm, also exploring other biomolecule interactions</td>
<td>100s to 1000s of 1-1000 core simulations &amp; data analysis</td>
<td>O</td>
</tr>
<tr>
<td>Biochemistry</td>
<td>Identification of drug targets via computational screening</td>
<td>Up to 1M x 1 core</td>
<td>O (Falkon)</td>
</tr>
<tr>
<td>Bioinformatics</td>
<td>Metagenome modeling</td>
<td>1000’s of 1-core integer programming problems</td>
<td>D</td>
</tr>
<tr>
<td>Business economics</td>
<td>Mining of large text corpora to study media bias</td>
<td>Analysis and comparison of 70M+ text files of news articles</td>
<td>D</td>
</tr>
<tr>
<td>Climate</td>
<td>Ensemble climate model runs and analysis of output data</td>
<td>10s to 100s of 100-1000 core simulations</td>
<td>E</td>
</tr>
<tr>
<td>Economics</td>
<td>Generation of response surfaces for various economic models</td>
<td>1K to 1M 1-core runs (10K typical), then data analysis</td>
<td>O</td>
</tr>
<tr>
<td>Neuroscience</td>
<td>Analysis of functional MRI datasets</td>
<td>Comparison of images; connectivity analysis with SEM, many tasks (100K+)</td>
<td>O</td>
</tr>
<tr>
<td>Radiology</td>
<td>Training of computer aided diagnosis algorithms</td>
<td>Comparison of images; many tasks, much communication</td>
<td>D</td>
</tr>
<tr>
<td>Radiology</td>
<td>Image processing and brain mapping for neurosurgical planning research</td>
<td>1000’s of MPI application executions</td>
<td>O</td>
</tr>
</tbody>
</table>
ZHT: A distributed Key-Value store

- Light-weighted
- High performance
- Scalable
- Dynamic
- Fault tolerant
- Strong Consistency
- Persistent
- Versatile: works from clusters, to clouds, to supercomputers
Resource Management for Extreme-Scale Data-Intensive Computing
ZHT: Zero-Hope Distributed Hash Table

Latency in microsec

DynamoDB read
DynamoDB write
ZHT 4 ~ 64 nodes

0.99
0.9
0.8
0.6
0.4
0.2
0.0

0
5000
10000
15000
20000

Resource Management for Extreme-Scale Data-Intensive Computing
ZHT: Zero-Hope Distributed Hash Table

- TCP: no connection caching
- ZHT: TCP connection caching
- UDP non-blocking
- Memcached

Resource Management for Extreme-Scale Data-Intensive Computing
FusionFS Distributed File System

- A distributed file system co-locating storage and computations, while supporting POSIX
- Everything is decentralized and distributed
- Aims for millions of servers and clients scales
- Aims at orders of magnitude higher performance than current state of the art parallel file systems
This gap will grow even larger as parallel filesystems saturate external network – expected gap will be \(^\sim\)4 orders of magnitude faster performance.
**FusionFS Distributed File System**

- **16K-node scales**
  - FusionFS 2500GB/s (measured) vs. GPFS 64GB/s (theoretical)
  - 39X higher sustained throughput

- **Full system 40K-node scales**
  - Expected Performance: 100X higher I/O throughput
  - Expected Performance: 4000X higher metadata ops/sec
• Many hot topics related to distributed storage
  – Provenance (FusionProv) – uses ZHT
  – Information Dispersal Algorithms (IStore) – uses GPUs
  – SSD+HHD hybrid caching (HyCache)
  – Data Compression
• Improvements on the horizon
  – Non-POSIX interfaces (e.g. Amazon S3)
  – Explore viability of supporting HPC checkpointing
  – Deep indexing and search
MATRIX MTC execution Framework at Extreme Scales

• MATRIX - distributed MTC execution framework for distributed load balancing using Work Stealing algorithm
  • Distributed scheduling is an efficient way to achieve load balancing, leading to high job throughput and system utilization
  • Dynamic job scheduling system at the granularity of node/core levels for extreme scale applications
GeMTC: GPU-Enabled Many-Task Computing

GPU
- Streaming Multiprocessors (15 SMXs on Kepler K20)
- 192 warps * 32 threads

Coprocessors
- Intel Xeon Phi
- 60 cores * 4 threads per core = 240 hardware threads

GeMTC
- Efficient support for MTC on accelerators
GeMTC: GPU-Enabled Many-Task Computing

![Graph showing efficiency over sleep time for different GPU and CPU configurations.]

- **NVIDIA GPU (1 Worker)**
- **NVIDIA GPU (84 Workers)**
- **Xeon Phi (60 Workers)**
- **AMD Opteron (48 Workers)**

The graph illustrates the efficiency in relation to the sleep time (in microseconds) for various hardware configurations, highlighting the efficiency gains as sleep time increases.
• **Decentralization is critical**
  – Computational resource management
  – Storage systems

• **Preserving locality is critical!**
  – POSIX I/O on shared/parallel file systems ignore locality
  – Data-aware scheduling coupled with distributed file systems that expose locality is the key to scalability over the next decade

• **Co-locating storage and compute is GOOD**
  – Leverage the abundance of processing power, bisection bandwidth, and local I/O

---

Resource Management for Extreme-Scale Data-Intensive Computing
Active Funding ($)

- **NSF CAREER 2011 – 2015: $486K**
  - “Avoiding Achilles’ Heel in Exascale Computing with Distributed File Systems”, NSF CAREER

- **DOE Fermi 2011 – 2013: $84K**
  - “Networking and Distributed Systems in High-Energy Physics”, DOE FNAL

- **DOE LANL 2013: $75K**
  - “Investigation of Distributed Systems for HPC System Services”, DOE LANL

- **IIT STARR 2013: $15K**
  - “Towards the Support for Many-Task Computing on Many-Core Computing Platforms”, IIT STARR Fellowship

- **Amazon 2011 - 2013: $18K**
  - “Distributed Systems Research on the Amazon Cloud Infrastructure”, Amazon

- **NVIDIA 2013 – 2014: $12K**
  - “CUDA Teaching Center”, NVIDIA
• DOE 2011 – 2013: 450K hours
  – “FusionFS: Distributed File Systems for Exascale Computing”, DOE
    ANL ALCF; 450,000 hours on the IBM BlueGene/P

• XSEDE 2013: 200K hours
  – “Many-Task Computing with Many-Core Accelerators on XSEDE”,
    NSF XSEDE; 200K hours on XSEDE

• GLCPC 2013: 6M hours
  – “Implicitly-parallel functional dataflow for productive hybrid
    programming on Blue Waters”, Great Lakes Consortium for Petascale
    Computation (GLCPC); 6M hours on the Blue Waters Supercomputer

• NICS 2013: 320K hours
  – “Many-Task Computing with Many-Core Accelerators on Beacon”,
    National Institute for Computational Sciences (NICS); 320K hours on
    the Beacon system
More Information

• More information:
  – http://www.cs.iit.edu/~iraicu/
  – http://datasys.cs.iit.edu/

• Contact:
  – iraicu@cs.iit.edu

• Questions?