Big Data and Extreme Computing: a Storage-Based Pathway to Convergence

Gabriel Antoniu, Inria

Based on joint work with Pierre Matri, Alexandru Costan, María Pérez
BDEC: From the Previous Episodes...

Big Data and Extreme Computing workshops series (BDEC)

http://www.exascale.org/bdec/

Overarching goal:
1. Create an international collaborative process focused on the co-design of software infrastructure for extreme scale science, addressing the challenges of both extreme scale computing and big data, and supporting a broad spectrum of major research domains,
2. Describe funding structures and strategies of public bodies with Exascale R&D goals worldwide
3. Establishing and maintaining a global network of expertise and funding bodies in the area of Exascale computing

BDEC Workshop, Charleston, SC, USA, April 29-May 1, 2013
BDEC Workshop, Fukuoka, Japan, February 26-28, 2014
BDEC Workshop, Barcelona, Spain, January 28-30, 2015
BDEC Workshop, Frankfurt, June 16-17, 2016

Credits: Jack Dongarra
Big Data and Extreme Computing

High-end data analytics and HPC are both essential elements of an integrated computing research-and-development agenda:

- Big compute generates and is needed to analyze big data
- Networking and memory performance are critical to both

Programming models and tools are perhaps the biggest point of divergence between the scientific-computing and big-data ecosystems.
BDEC: From the Previous Episodes…

Divergent ecosystems

Application Level
- Mahout, R and Applications
- Applications and Community Codes

Middleware & Management
- Hive, Pig, Sqoop, Flume
- Map-Reduce, Storm
- Hbase BigTable (key-value store)
- FORTRAN, C, C++ and IDEs
- Domain-specific Libraries
- MPI-OpenMP CUDA/OpenCL
- Perf & Debug (e.g., PAPI)
- NA Libs

System Software
- VMS, Containers and Cloud Services
- HDFS (Hadoop File System)
- PFS (e.g., Lustre)
- Batch Scheduler
- System Monitoring

Cluster Hardware
- Ethernet Switches
- Local Node Storage
- Commodity X86 Racks
- Linux OS variant
- IB+Enet Switches
- SAN+Local Storage
- x86+GPUs or Accelerators

Credits: Dan Reed

G. Antoniu - BDEC Frankfurt, 15-17 June 2016
## EC vs. BD in the Past: Not the Same Application Requirements

<table>
<thead>
<tr>
<th>Extreme Computing</th>
<th>Big Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static/predictable requirements for resources</td>
<td>Volatile/unpredictable requirements for resources</td>
</tr>
<tr>
<td>Non-interactive</td>
<td>On-demand/predictable/controlled response time, often interactive</td>
</tr>
<tr>
<td>Focus on performance</td>
<td>Focus on “productivity”</td>
</tr>
<tr>
<td>Data is private</td>
<td>Data is shared and managed for sharing (e.g., provenance), used collaboratively</td>
</tr>
<tr>
<td>Focus on domain-dependent methods</td>
<td>Include a wider range of methods including domain-independent methods e.g., statistical methods</td>
</tr>
</tbody>
</table>

Credits: Kate Keahey
Stepping Stones: Towards EC/BD Convergence

Sharing the same resources
• Resource management methods need to evolve so that BD and EC can share resources

Convergence towards « stepping stones »
• Challenges and demonstrations
  • Software representing an entire system that can be used for BDEC

• HPC features available in the cloud (HPC)
• Cloud” features available on HPC platforms (availability, predictable response time, etc.)

Credits: Kate Keahey
A Catalyst for Convergence: Data Science
Data Science

Skills needed:
- Storage hardware and software architectures
- Large-scale distributed systems
- Data management services
- Data analysis
- Machine learning
- Decision making
- With a special flavour in advanced data storage solutions unifying cloud and HPC storage facilities

Source: New York Times, April 2013
An Approach: The BigStorage H2020 Project

Project overview

Data Science
- Modelling Big Data processing
- Energy-efficient analysis
- Data-driven decision making for Big Data applications

HPC-Cloud Convergence
- Applications
- Middleware, operating in the cloud and HPC environments
- Infrastructure for Storage and Computing

Storage Devices
- Storage acceleration
- Storage convergence
- Storage isolation

Energy
- Compression or de-duplication for storage footprint reduction
- Hints from application to storage system, enabling energy consumption reduction

www.bigstorage-project.eu
The BigStorage Consortium
HPC and Cloud Storage-Based Convergence: a Few Questions

• Multiple angles
  - How can applications exploit data?
  - What middleware, operating in the cloud and HPC environments?
  - What infrastructure for storage, appropriate for efficient computation and analysis?

• HPC and cloud infrastructures as an enabler of convergence
# HPC vs. BDA: Back to Application Requirements

<table>
<thead>
<tr>
<th>HPC</th>
<th>BDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static, predictable resource requirements</td>
<td>Volatile, unpredictable resource requirements</td>
</tr>
<tr>
<td>Many independent executions</td>
<td>Continuous execution</td>
</tr>
<tr>
<td>Domain-specific methods</td>
<td>Generic methods</td>
</tr>
<tr>
<td>Performance is crucial</td>
<td>Data is vital</td>
</tr>
</tbody>
</table>
Storage Systems: State of Practice

HPC

• Storage is unstructured
• Domain-specific data structures…
• …handled by Applications

• Usually one general-purpose storage system is provided on the platform, typically a POSIX-compliant FS (Lustre, GPFS, …)

BDA

• Storage is structured
• Generic data structures
  • tables, lists, maps…
• …provided by the storage layer

• Multiple purpose-specific storage systems are available (Key-Value Stores, SQL Databases, Time Series Databases, …)
HPC View: How to Converge?
POSIX Must Die! *(faster)*

More specifically: POSIX-IO

Many Top-500 supercomputers provide POSIX: ~80% (Lustre, GPFS)

Such systems already scale well
- Large capacities: 55 PB at LLNL
- High bandwidth: 1.4 TB/s at ORNL

Can we make them scale better?
- Who really needs fine-grained permissions?
- Who really needs file hierarchy?
- Neither of those are supported by MPI-IO

Handling such features has a cost!
BDA View: How to Converge? 
Is Purpose-Specific Storage Really Relevant?

BDA relies on **generic operators** for productivity
- Indexes
- Data aggregates
- Sometimes rich queries (SQL, …)

Such operators are provided at the storage level
- Enabled by data structures
- Easy to use by application developers

BUT what if we need a new operator?
- Not always easy to implement on top of structured storage
- Usually requires new, specific storage system components
- ... not an easy task!

Purpose-specific is fine as long as what you have is sufficient

Yet, BDA application requirements are volatile... **How about the future?**
BLOBs: a Way to Reconcile?
Could BLOBs be a solution for HPC?

Probably yes!

BLOB: Binary Large Object
• Can be a large collection of unstructured binary data stored as a single entity

A BLOB can provide the same access methods as a file
• Open
• Read at offset
• Write at offset
• Close

BUT with a flat namespace, without hierarchy semantics
• Reduced complexity
• Just keep what people actually need

Hierarchy can still be implemented atop BLOBs if necessary
BLOBs: a Way to Reconcile?
Could BLOBs be a solution for BDA?

It works already!

Most state-of-the-art BDA storage systems can be implemented on top of BLOBs

- Key-Value stores can be mapped directly
- Document stores: just serialize any structured data to a BLOB
- SQL Databases can be built on Key-Value Stores (FoundationDB)
- So, data tables can, too!
- Time Series databases, such as OpenTSDB, rely on tables

BLOBs are already used in BDA to store immutable objects
- Images, videos, soundtracks, …
The Case of RADOS

RADOS
A reliable, autonomous, distributed object store

LIBRADOS
RADOS-based BLOB management library

Application

RADOSGW
S3-like storage

Application

RBD
Distributed block device

CEPH FS
(A)lmost POSIX-compliant I/O interface

Application

G. Antoniu - BDEC Frankfurt, 15-17 June 2016
A Challenge: How to Manage Concurrency?

For HPC:
• Usually handled at some above the data storage layer (MPI-IO)
• Or not handled at all if the application does not need it

For BDA:
• Usually handled alongside consistency at the storage layer (ACID transactions)
  • Transactional databases
  • Transactional Key-Value Stores (Hyperdex, Espresso, …)

Can transactions be managed at middleware level for BDA?
•Hardly
•Concurrency middleware typically only guarantee isolation
•What about data consistency in case of failures?
How about Providing Transactions at the Storage Level?

Intuitively, a terrible idea for HPC. But is it?

Some transaction protocols are efficient: better than consensus algorithms such as Paxos

Transaction Chains – introduced in Lynx [1] – are one example

The idea: delay conflict resolution as late as possible

No conflict → No resolution → (Almost) no overhead!

[1] Zhang et al. – Transaction chains: achieving serializability with low latency in geo-distributed storage systems – SOSP’13
A Solution for Converged Storage

HPC  |  BDA

- Transactional BLOB store
- Application
- File System
- Transactional K/V Store
- Custom middleware
- Transactional SQL DB
- Application

G. Antoniu - BDEC Frankfurt, 15-17 June 2016
A Solution for Converged Storage

HPC + BDA

Application

Application

Transactional SQL DB

Application

Application

Transactional K/V Store

Custom middleware

Application

Application

File System

Transactional BLOB store
A Proof of Concept: Týr

A BLOB storage system

Support for high-throughput under high-concurrency
• Decentralized design, no dedicated metadata nodes
• Lock-free concurrency control (MVCC)

Built-in, lightweight ACID transactions
• Native consistency and access concurrency management
• Enables cool features: in-place, atomic binary updates
  • Increment, decrement, multiply, divide, shift bytes, …
  • Useful for counters, data aggregation, …

Independent of storage specifics (HDD, SSD, In-memory)

Experimental middleware implementations
• Transactional Key / Value Store
• Transactional POSIX file system
• Next on the list: ADIO / MPI-IO interface
Preliminary Evaluations

Use-case: a scientific monitoring service: **MonALISA**

- Monitoring application of the CERN LHC experiment
- Ingests data at a rate of up to 13 GB/s
- Computes more than 35,000 aggregates in real time
- Produces more than $10^9$ data files per year

- Current implementation based on legacy SQL does not scale

MonALISA server implemented on top of Týr and compared with a few state-of-the-art storage systems

- RADOS
- BlobSeer
- Azure Storage

Experiments were run on up to a 512-node Microsoft Azure cluster
How Does it Perform?

**Write performance**

- Aggregated throughput (million ops / s) vs Concurrent writers

**Read performance**

- Aggregated throughput (million ops / s) vs Concurrent readers

### Graphs

- **Týr (Atomic)**
- **Týr**
- **RADOS**
- **BlobSeer**
- **Azure**

G. Antoniu - BDEC Frankfurt, 15-17 June 2016
How Does it Perform?

Horizontal scalability
(65% read / 35% write workload)
Conclusion: Is This a Pathway to Convergence?

BLOBS can be a basis for storage for converged HPC-BDA systems.

Built-in transactions at storage level could be an enabling factor.

Thank you!

Contact: gabriel.antoniu@inria.fr
Spare slides
### HPC vs. BDA: What Does This Imply for Storage?

<table>
<thead>
<tr>
<th></th>
<th>HPC</th>
<th>BDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource allocation</td>
<td>Static provisioning</td>
<td>Elastic provisioning</td>
</tr>
<tr>
<td>Execution model</td>
<td>Unstructured storage</td>
<td>Structured storage</td>
</tr>
<tr>
<td>Operators on data</td>
<td>Recompute!</td>
<td>Replicate!</td>
</tr>
<tr>
<td>Data availability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>