Execution Environments for Big Data

Challenges for Storage Architectures and Software

Wolfgang E. Nagel
Phone: +49 351 - 463 - 35450
URL: http://tu-dresden.de/zh
E-mail: wolfgang.nagel@tu-dresden.de
Many ways to produce and use large amounts of data
- Experiments
- Simulations
- Sensors
- Digital copies

Come in different flavors
- (Semi-) structured vs. unstructured
- Distributed vs. centralized
- ??
How workflows will change

Today

- Collect or compute
- Move around and store, analyze, visualize
- Use storage devices as information hub

Future scenarios

- Analysis close to the data, in-situ processing
- Data driven workflows: automated analysis triggered by the arrival of new data
- Workflow management
  - Integrates data and compute task
  - Scalable and resilient
How workflows will change

Future scenarios

- Integrated information life cycle management
- Automatic metadata extraction
- Distributed but connected metadata and data
- Knowledge mining
- Data accessible through common (web) interfaces
How HPC architecture will change

From a data analytics point of view

- Extremely large memory, deeper hierarchies
- Many different storage technologies/options
  - Unique performance characteristics (NVRAM, PCM, SSD, hard disk, tapes)
  - Distributed as well as global resources
- Intelligent middleware for guidance of I/O layer

- Data intensive / Data driven applications have to guide design decisions for HPC systems
How HPC architecture will change

- Moving data to computing units and large memory
- Access to external data
  - High Performance Data Transfer Capabilities required
    - External data movers
    - Quality of service of the infrastructure
How: monitor everything (today, work in progress)

- node
- I/O network
- MDS
- OSS
- RAID
- SAN
- OSS
- OSS
- RAID
- RAID
- RAID
- node local events
- application events
- dynamic storage adaptation
- utilization
- errors
- backend, cache
- throughput, IOPS

TECHNISCHE UNIVERSITÄT DRESDEN

ZIH
Center for Information Services & High Performance Computing

Wolfgang E. Nagel
Future: Adapt the storage through intelligent middleware

- Applications have to change
- I/O semantics have to guide the **whole** storage subsystem through middleware
Questions

- What BigData/ExtremScale workflows do you envision?
- What are your requirements for a storage middleware?
- How close an I/O infrastructure has to be to ExtremScale Computing, how will we manage data movement and workflows?
Knowledge Environments

Reproducible Data-Driven Research
Policy-based Data Management
Interoperability Mechanisms

Reagan Moore
Applications of Policy-based Data Management

- Astronomy
  - Large Synoptic Survey Telescope, CyberSKA, NOAO
- Climate
  - NOAA National Climatic Data Center, NASA NCCS
- Cognitive science
  - Temporal Dynamics of Learning Center
- Engineering
  - CIBER-U
- Genomics
  - Wellcome Trust Sanger Institute, Broad Institute
- High-energy Physics
  - BaBar
- Neuroscience
  - International Neuroinformatics Coordinating Facility
- Oceanography
  - Ocean Observatories Initiative
- Plant biology
  - iPlant collaborative
- Seismology
  - Southern California EarthQuake Center
- Social Science
  - Odum
- Archives
  - Carolina Digital Repository, NCDC
- Collaboration Service
  - Australian Research Collaboration Service
- Data grids
  - UK e-Science data grid
- Libraries
  - French National Library, Chronopolis, Texas Dig Lib
Policy-based Data Management
Types of Knowledge Encapsulation

• Knowledge needed to interact with a community resource
  – Encapsulate the protocol needed for interaction

• Knowledge needed for a research analysis
  – Encapsulate processing steps within a workflow
  – Automate storage of workflow provenance and workflow results
  – Share workflows and support re-execution of workflows

• Knowledge needed to manage research results
  – Encapsulate management policies as computer actionable rules
<table>
<thead>
<tr>
<th>Components</th>
<th>Interoperability</th>
<th>Technologies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clients</td>
<td>Java, C, C++, OpenSocial</td>
<td>iDrop, iDrop-Web, MediaWiki, VIVO, FUSE, Fedora, Dspace, I/O libraries, Cheshire, Portals, DataBook, Facebook</td>
</tr>
<tr>
<td>Policies</td>
<td>Rules</td>
<td>Integrity, Replication, Description, Arrangement, ...</td>
</tr>
<tr>
<td>Policy Points</td>
<td>Rule base</td>
<td>iRODS Data Management Rules</td>
</tr>
<tr>
<td>Security</td>
<td>GSSAPI, PAM, Rules</td>
<td>Kerberos, GSI, CI-Login, InCommon, Unix, SHA-1, MD5, PAM</td>
</tr>
<tr>
<td>Scheduler</td>
<td>Rules, Micro-services</td>
<td>iRODS Rules, Micro-services</td>
</tr>
<tr>
<td>Metadata</td>
<td>Catalog Driver, Micro-services</td>
<td>iRODS iCat, mySQL, PostgreSQL, Oracle, HIVE</td>
</tr>
<tr>
<td>Storage Access</td>
<td>Storage driver, Micro-services</td>
<td>File system, archive, web, cloud, ERDDAP, PyDAP, NetCDF, HDF5</td>
</tr>
<tr>
<td>Rule Engine / workflows</td>
<td>Micro-services</td>
<td>Kepler, NCSA Cyberintegrator, Taverna, MakeFlow, Polyglot, {Hydro, Marine, Engg}-specific rules and micro-services</td>
</tr>
<tr>
<td>Messaging</td>
<td>Micro-services</td>
<td>iRODS Xmsg, AMQP</td>
</tr>
<tr>
<td>Networks</td>
<td>Network Driver, Micro-services</td>
<td>TCP/IP, Parallel TCP/IP, RBUDP, HTTP</td>
</tr>
</tbody>
</table>

**Blue** – DFC developed interfaces to the technology:  **Red** – DFC developed the technology
Extensibility Mechanisms

• Drivers:
  – Apply operations on data at the remote storage location

• Micro-services:
  – Encapsulate operations into a basic function that can be chained into a workflow

• Policies:
  – Encapsulate management policies as computer actionable rules
Knowledge Encapsulation

• Reproducible data driven research on massive data collections
  – Move processing to the data
  – Automate retrieval of data from
  – Capture processing steps in workflows that can be shared and re-executed
  – Automate capture of workflow provenance
Enables

• Processing within storage controllers (DDN)
  – Feature-based indexing of data
• Reproducible data driven research
  – Workflow provenance and workflow re-execution
• Creation of collaboration environments
  – Policies for shared collections & shared workflows
• Creation of reference collections
  – Management policies for assessment criteria
THE CHALLENGE OF THE NEXT DECADE IN NUMERICAL COSMOLOGY.
CRITICAL POINTS IN BIG DATA AND EXTREME-SCALE COMPUTING

JEAN-MICHEL ALIMI
LUTH, OBSERVATOIRE DE PARIS, FRANCE
DEUS CONSORTIUM (WWW.DEUS-CONSORTIUM.ORG)

Outline:
• Evolution of N-Body Cosmological simulations on the 1st Rank of top500
• White Paper: The challenges of the next decade in numerical cosmology.
• Comments on the survey by BDEC organizers
Evolution of N-Body Cosmological simulations on the 1st Rank of top500

Preliminary Remarks

Today we are able to perform Cosmological N-Body Simulations with $8192^3$ particle evolving in the entire volume of the observable universe where statistical errors are reduced to cosmic variance (ie minimal sample variance) and where we are guaranteed to detect rare supermassive halos.

With such resolutions we can follow the formation of all DM Halos with $M \geq 10^{14}$ solar masses.

2.5 trillions computing points (double precision)
Coarse Grid $8192^3$ (21 $h^{-1}$ Gpc)
formal resolution $524288^3$ (40 $h^{-1}$ kpc)

a ”Moore”-like law (dashed line) with an increasing factor of 2 every 18 months underestimates the acceleration of state-of-the-art cosmological N-body simulations.

At the top DEUS Simulation X.

JEAN-MICHEL ALIMI
Evolution of N-Body Cosmological simulations on the 1st Rank of top500

Preliminary Remarks

Today we are able to perform Cosmological N-Body Simulations with $8192^3$ particle evolving in the entire volume of the observable universe where statistical errors are reduced to cosmic variance (i.e., minimal sample variance) and where we are guaranteed to detect rare supermassive halos.

With such resolutions we can follow the formation of all DM Halos with $M \geq 10^{14}$ solar masses.

Tomorrow understanding the nature of the Dark Universe (DM and DE) is probably and firstly a big physical challenge. However, in term of numerical simulation, we can consider that the next challenge is to follow the formation of all DM halos with $M\geq10^{11} – 10^{12}$ solar masses (galaxy) (Theory, Observation)

Jean-Michel ALIMI
Evolution of N-Body Cosmological simulations on the 1st Rank of top500

Preliminary Remarks

Why is Extreme-Scale computing necessary? And Consequently, Why does numerical cosmology lead to Big (Huge) data problem?

Naive and simple analysis

Using the evolution of available memory capacity, the computing power and the size of the storage disks on the most powerful (in top500) supercomputers over the last past 20 years, we estimate the largest (in terms of number particles) N-body cosmological simulations which could be tomorrow performed.

Hypothesis:
• Memory limit: 200 B RAM per particle (Simple precision for a DEUS FUR simulation)
• Disk size limit: 320 B per particle and 10 snapshots are saved
• Cpu-time limit: All the supercomputer during 1 month with a performance, 10% Rmax linpack.
• Number of operations (Ishivama et al 2012): 50 operations per interaction, 500 time-steps, 180*npart*alog(npart) interactions...
The most powerful (in top500) supercomputers over the last past 20 years

<table>
<thead>
<tr>
<th>#year</th>
<th>#year</th>
<th>#GB</th>
<th>#Tflops</th>
<th>#Nb core</th>
<th>Disk(TB)</th>
<th>Type</th>
<th>#Name</th>
<th>#cie</th>
<th>#country</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012.9</td>
<td>2012.9</td>
<td>700000.</td>
<td>17600</td>
<td>552960</td>
<td>10000</td>
<td>3</td>
<td>Titan</td>
<td>Oak</td>
<td>Ridge USA</td>
</tr>
<tr>
<td>2012.5</td>
<td>2012.5</td>
<td>1600000</td>
<td>16320</td>
<td>1572864</td>
<td>55000</td>
<td>2</td>
<td>Sequoia</td>
<td>LLNL</td>
<td>USA</td>
</tr>
<tr>
<td>2011.5</td>
<td>2011.9</td>
<td>1300000</td>
<td>10500</td>
<td>705024</td>
<td>11000</td>
<td>1</td>
<td>K</td>
<td>RIKEN</td>
<td>JAPAN</td>
</tr>
<tr>
<td>2010.9</td>
<td>2010.9</td>
<td>262000.</td>
<td>2570</td>
<td>21504</td>
<td>2000</td>
<td>3</td>
<td>Tianhe-IA</td>
<td>NSC</td>
<td>CHINA</td>
</tr>
<tr>
<td>2009.9</td>
<td>2010.5</td>
<td>360000.</td>
<td>1759</td>
<td>224256</td>
<td>10000</td>
<td>1</td>
<td>Jaguar</td>
<td>Oak</td>
<td>Ridge USA</td>
</tr>
<tr>
<td>2008.5</td>
<td>2009.5</td>
<td>103600.</td>
<td>1105</td>
<td>116640</td>
<td>1000</td>
<td>4</td>
<td>Roadrunner</td>
<td>LLNL</td>
<td>USA</td>
</tr>
<tr>
<td>2005.9</td>
<td>2007.5</td>
<td>32768</td>
<td>281</td>
<td>131072</td>
<td>0</td>
<td>2</td>
<td>BluegeneL</td>
<td>LLNL</td>
<td>USA</td>
</tr>
<tr>
<td>2002.5</td>
<td>2004.5</td>
<td>10000</td>
<td>36</td>
<td>5120</td>
<td>700</td>
<td>5</td>
<td>Earth Sim</td>
<td>ESC</td>
<td>JAPAN</td>
</tr>
<tr>
<td>2000.9</td>
<td>2001.9</td>
<td>6000</td>
<td>7.2</td>
<td>8192</td>
<td>160</td>
<td>1</td>
<td>ASCI White</td>
<td>LLNL</td>
<td>USA</td>
</tr>
<tr>
<td>1997.5</td>
<td>2000.5</td>
<td>1212</td>
<td>1.1</td>
<td>9632</td>
<td>12.5</td>
<td>1</td>
<td>ASCI Red</td>
<td>Sandia</td>
<td>USA</td>
</tr>
<tr>
<td>1996.9</td>
<td>1996.9</td>
<td>128</td>
<td>0.368</td>
<td>2048</td>
<td>0.528</td>
<td>1</td>
<td>CP-PACS</td>
<td>Tsukuba</td>
<td>JAPAN</td>
</tr>
<tr>
<td>1996.5</td>
<td>1996.5</td>
<td>64</td>
<td>0.232</td>
<td>1024</td>
<td>0</td>
<td>1</td>
<td>SR2201</td>
<td>Tokyo</td>
<td>JAPAN</td>
</tr>
<tr>
<td>1994.9</td>
<td>1995.9</td>
<td>42</td>
<td>0.170</td>
<td>167</td>
<td>0</td>
<td>5</td>
<td>Wind Tunnel</td>
<td>NAL</td>
<td>JAPAN</td>
</tr>
<tr>
<td>1994.6</td>
<td>1994.6</td>
<td>60</td>
<td>0.143</td>
<td>3580</td>
<td>0</td>
<td>1</td>
<td>XP/S 140</td>
<td>Sandia</td>
<td>USA</td>
</tr>
<tr>
<td>1993.9</td>
<td>1993.9</td>
<td>35</td>
<td>0.124</td>
<td>140</td>
<td>0</td>
<td>5</td>
<td>Wind Tunnel</td>
<td>NAL</td>
<td>JAPAN</td>
</tr>
<tr>
<td>1993.6</td>
<td>1993.6</td>
<td>32</td>
<td>0.060</td>
<td>1024</td>
<td>0</td>
<td>1</td>
<td>C-M 5</td>
<td>LANL</td>
<td>USA</td>
</tr>
</tbody>
</table>

Jean-Michel Alimi
Evolution of N-Body Cosmological simulations on the 1st Rank of top500

Linear Interpolation.

So we are obviously in the era where it is the memory that limits us from these three quantities.

Today: $8192^3$

(*8) $16384^3$ probably could be performed between 2013 and 2018

(*64) $32768^3$ probably could be performed between 2016 and 2022

(> * 100) Beyond: deeply in the era of the extreme-scale computing and (Big) Huge data

JEAN-MICHEL ALIMI
The challenges of the next decade in numerical cosmology.  
Big Data and Extreme-Scale Computing

White Paper: Critical points to perform the next challenge in numerical cosmology

Recently we performed a numerical « challenge » in cosmology. We were able to perform a N-body gravitational simulation with 0.5 trillion particle evolving in the entire volume of the Observable Universe with 2.5 trillion computing point. Such computation was repeated for three different cosmological models.

All facets of HPC were solicited: computation time, memory usage, communication schemes, I/O management have to be strongly optimized in the same time.

The next challenge in numerical cosmology of the next decade is (at least) to win a factor of 100 in the number of particles, both from theoretical point of view and from observational point of view.

The role of numerical simulations to support next large observational projects (Euclid satellite (ESA/NASA 2020), BigBOSS ...)

JEAN-MICHEL ALIMI
The challenges of the next decade in numerical cosmology.

Big Data and Extreme-Scale Computing

Link between HPC and large scale instruments becomes more and more important

Observational motivations of extreme scale computing in cosmology

Density of Halos on the sky

Density of Halos in depth

Deep Full Sky

(10^{10} h^{-1} M_{\odot}, z=7-8)

halo density halo mass

Jean-Michel Alimi
The challenges of the next decade in numerical cosmology.  
Big Data and Extreme-Scale Computing

White Paper: Critical points to perform the next challenge in numerical cosmology

From our previous numerical experiments, we estimate the evolution and the difficulties (of all facets of HPC) to perform this challenge:

**Computing Time (factor of 100 in the number of particles): factor 5**

Efficiency of N-body/Poisson solver as a function of the number of MPI tasks in a weak-scaling configuration. The reference corresponds to 74 MPI tasks. The efficiency is shown at the beginning of the run (yellow), at 1/4th (red), half (purple), 3/4th (blue) and at the end of the run (green). The efficiency is first of the order of 60%, it falls to about 55% during a short time when the first refinements are triggered and finally it increases to 75%. Multigrid acceleration allows us to reach higher efficiencies comparatively to the efficiency of an ideal PM-FFT code in black.

From 3 days on 80 000 cores to 15 days
On 8 million cores

A minimum of 8 GB per process seems necessary and big effort in communication scheme
The challenges of the next decade in numerical cosmology.
Big Data and Extreme-Scale Computing

White Paper: Critical points to perform the next challenge in numerical cosmology

I/O (factor of 100 in the number of particles): factor 100 (scientific reasons)

Finally we get a new distribution between the computing time and the I/O time.
90% for the I/O.

More generally, because, the volume of scientific data is growing exponentially, as this volume can exceeded the capacity of storage and data management services that can be considered tomorrow available to one user of large data centers, we suggest a new way of doing supercomputing.

Two options:

Large scale Simulation « on demand »
Large statistics of simulations of smaller sizes.

Fault-tolerance (Data Integrity)

JEAN-MICHEL ALIMI
The survey by BDEC organizers

- **Architecture:**
  - What architectural changes are needed for extreme computing storage systems to make them better suited for BD?
  - What operational changes are needed to support new storage architectures?
  - Looking at future technologies, what future architectures are possible?

- **Workflows:**
  - For extreme computing and big data, describe a forwarding-looking workflow, from simulation to analysis.
  - What software is missing to support your workflow?
  - A plan for achieving interoperability among various systems that one might want to use.

- **Taxonomy:**
  - There are several forms of data-centric computing linked to extreme computing. One outcome of this workshop is to help describe these modes. Please outline how you use your data and how you answer questions about your science using your data.
  - Do you have a data-driven mini-application that demonstrates a new usage model?
  - What are cross-cutting concerns for BD (for example: data integrity)

- **Software:**
  - What software are you currently using to manage and explore your data?
  - What algorithms and software libraries/tools need development and improvement to address your big data needs
  - As you look to the future, what are the holes/gaps that have no planned solution?

- **Interoperability challenges:**
  - How to handle Data provenance (location, observed/simulated, type of system concerned) from a data representation and IT architectural point of view? How to annotate existing data sets and develop records for data citation and tracking?
  - What Information systems are used for providing semantic capacity to provide effective translation between data and conceptual models used by different communities?
  - What IT systems are used for providing information about the actual use of both observational data and simulated data?

---

Some comments following discussions with Stephane Requena (CTO GENCI)

**Jean-Michel Alimi**
CURIE : the French PRACE Tier0 supercomputer

- CURIE, France’s commitment to PRACE, is overseen by GENCI
- Located in TGCC and operated by CEA DAM teams
- A modular and balanced architecture by Bull
  - Cluster of SMP nodes with fat, thin and hybrid nodes
  - Complementary to other PRACE Tier0 systems
  - Fully available since March 8, 2012

In honour of Marie Curie

Global peak performance of 2 PFlop/s
- > 92 000 Intel cores,
- 360 TB memory,
- 10 PB Lustre @ 250 GB/s,
- 120 racks, < 200 m² - 2.5 MW
- 50 kms of cables

JEAN-MICHEL ALIMI
CURIE the French PRACE system

- 3 different x86 compute partitions
  - 360 BULL S6010 fat nodes
    - Intel NH EX 2.26 GHz, 11520 cores
    - 32 -> 128 cores/node
    - 128 -> 512 GB/node
    - 105 Tflops peak
  - 5040 BULL B510 thin nodes
    - Intel SNB 2.7 GHz, 80 640 cores
    - 16 cores and 64 GB/node
    - One local SSD per node
    - 1740 Tflops peak
  - 288 BULL B505 hybrid nodes
    - Intel WM EP 2.67 GHz & nVIDIA M2090 GPUs
    - One local SSD per node
    - 192 + 12 Tflops peak

- A full data centric approach using Lustre
  - 1st level: Tier0 private
  - 2nd level: shared between HPC systems

Jean-Michel Alimi
What architectural changes are needed for extreme computing storage systems to make them better suited for BD?

- Deploy data centric HPC approaches
- Very important to consider to deliver not only PFlops but also Pbytes
  - Balanced systems: cpu, mem capacity, network and disk bandwidth/capacity
- Multi level storage hierarchy: local SSD -> private Lustre -> shared Lustre -> archive

What operational changes are needed to support new storage architectures?

- Need to invest into fine tuning of the parallel file system (Lustre) -> one of the most important component of the system
- Development of Lustre monitoring tools
- Not always easy to debug Lustre problems on so big machines

Looking at future technologies, what future architectures are possible?

- Integration of new and capacitive memory technologies at the node level (HMC, PC Mem, ...)
- Evaluation of new BD methodologies: eg Map/Reduce, NoSQL, ...
  - Adapted on specific scientific domains
  - Used on top of Lustre or GPFS, no wish to have a new file system to support like HDFS
  - Interesting research done by KerData team@Inria -> use of MapReduce on top of GPFS and use of large-scale distributed storage service (BlobSeer)
For extreme computing and big data, describe a forwarding-looking workflow, from simulation to analysis.

- **Data in numerical cosmology consist usually of:**
  - *snapshots* corresponding to the x, v and identifiers of all particles for several (~30) redshifts during the simulation.
  - **backup of a fraction of the simulation box ("sample") at all computational coarse time-steps.** We store not only the particles and their properties, but also AMR cells describing the gravity field (Tree-merger).
  - **light-cones built during the dynamical computation stored at all time-steps** containing the particles and the AMR grid in spherical shells around observers at different space-time points; (ray tracing analysis, redshift space observations...).

At each coarse time step a light cone shell is extracted for a given observer. The cone is reconstructed by adding the shells at the end of the simulation.
For extreme computing and big data, describe a forwarding-looking workflow, from simulation to analysis.

- **From Simulation to analysis:**
  - « On the Fly »:
    » MPI based power spectrum computation code.
    » MPI-based parallel halo finder code (percolation technics)
    » Basic statistics on Halos distribution (number vs mass, number vs time....)
    » Halo properties (size, mass, velocity, angular momentum....)
  - « Post Processing »
    » Higher order statistics on matter field
    » Higher order statistics on Halos
    » Topological analysis (minkowski functional, voronoi tessellation....)
    » Dynamical analysis (weak lensing, velocity fields...)
    » ...
    » Visualization

- **From Analysis to Observation:**
  » Mock Catalog. Generation of virtual galactic catalog
    Statistical method
    Semi Analytical Method with phenomenological prescriptions
Workflow

- For extreme computing and big data, describe a forwarding-looking workflow, from simulation to analysis.
  - Next step for us: Using local SSD on Curie
    - Receive a small local higher bandwidth (400MB/s per node)
    - SSD can be used for writing temporary data and reliable and high-speed checkpoint restart (work in progress with the FTI lib by F. Cappello in Genci).

- What software is missing to support your workflow?
  - Monitoring tools for jobs and recovery work when error?
  - Remote viewing of integrated workflow results
    - Large volumes of data can not leave the center so easily
    - Easier to get out of compressed pixels
There are several forms of data-centric computing linked to extreme computing. One outcome of this workshop is to help describe these modes. Please outline how you use your data and how you answer questions about your science using your data.

- See before
Do you have a data-driven mini-application that demonstrates a new usage model?
- We developed during the test phase several tools to evaluate for example performance of I/O and performance of post-processing workflow.
- From these tools we could develop a data-driven mini-application limited to I/O with dynamical token system and limited to validation of computations (power spectrum computation and energy conservation) and limited to the post-processing workflow (detection of DM halos). Such mini-application could be used on test « snapshot » (set of positions, velocities and identifiers for a large number of particle).

What are cross-cutting concerns for BD (for example: data integrity)
- Data integrity, data sustainability (operating life of data around 10 years),
- We developed DEUVO database through Virtual Observatory interfaces. A worldwide user can access, process and analysis the data (Halo catalog, Halo particles, particles in sub-volume). Such a tool is limited to $4096^3$ simulations, it is inadequate for $8192^3$ simulations
- Resilience of next gen cosmology applications running on millions of cores
- Which programming model? Hierarchical MPI/OpenMP -> Reduce MPI memory footprint, optimise collectives and increase multi threading of the code, support hardware threads, expand transactional memory support
Software

Q What software are you currently using to manage and explore your data?

AMA-DEUS: A Multiple purpose Application for Dark Energy Universe Simulation

Numerous analysis program:
- Spatial Correlation
- Halo Statistics (halo mass function, Extreme value statistics...)
- Halo structures (profil, environment, sub-structures...)
- Topological analysis (minkowski functional, voronoi tesselation....)
- Dynamical analysis (weak lensing, velocity fields...)
...

Visualization:
A global application which integrates all aspects of the physical computational problem has been developed to perform numerical simulations of several scientific cases.

**AMA-DEUS: A Multiple purpose Application for Dark Energy Universe Simulation**

All facets of HPC were solicited: computation time, memory usage, communication schemes, I/O management must be strongly optimized in the same time.

**• Definition of Initial conditions.**
(Monte Carlo analysis of Observational Data and Homogeneous cosmological solutions)

**• Gravity Solver with PM7AMR algorithm.**
(A backup policy data adapted for managing Big Data Problem needs an optimized workflow on numerical data « on the fly »).

**• Storage of Numerical Data**
What algorithms and software libraries/tools need development and improvement to address your big data needs?

- The grand challenge DEUS allowed to stress all the components of the CURIE system and tuned some operational parameters just before being in full production. All facets of HPC were solicited: computation time, memory usage, communication schemes, I/O management must be strongly optimized in the same time.

- All improvements are at « low level »:
  - **Optimisation of communication scheme:** Balanced asynchronous MPI operations (Isend/Irecv) as originally implemented in Gravity solver and synchronous communications (Send/Recv, Bcast) for certain levels of refinements (particularly the coarser ones).
  - **Such optimisations have proven to be highly dependent on the MPI library as well as the IB topology.**
  - **This has required the implementation of specific tuning of the system with BULL HPC experts during the preparatory phase.**
  - **Optimisation of I/O management:** Scientific Goals imposes an effective policy regarding I/O and a quasi “on-the-fly” data post-processing.
Software

What algorithms and software libraries/tools need development and improvement to address your big data needs?

- All improvements are at « low level »:
  - **Optimisation of I/O management:** By using a dynamic system of I/O delegation based on tokens has been implemented in all the parts of our application. This token system, using MPI blocking instructions and parallel I/O allowed to saturate the bandwidth allocated for our simulations: finally up to 594 simultaneous writings were allowed in the case of snapshots, whereas in the case of the samples all tasks could write at the same time. The large variation in the size of shell outputs required the use of an adaptive token system. This has been set up to the extent that at each time-step the ratio of the volume of the overall box to the shell volume defines the number of concomitant writings. A part of the first level private LUSTRE parallel file system has been dedicated to DEUS experiment: 1.7 PB with a ~60-GB/s bandwidth, it was used at almost full speed: more than 40 GB/s writing during numerous periods of about half an hour and the same reading speed.

As you look to the future, what are the holes/gaps that have no planned solution?

- **Human resources for the post processing of the data**
Interoperability challenges

- How to handle Data provenance (location, observed/simulated, type of system concerned) from a data representation and IT architectural point of view? How to annotate existing data sets and develop records for data citation and tracking?
  - Link between HPC and large scale instruments will be more and more important
  - HPC mandatory for analysing massive volume of data generated by next generation instruments (Euclid, LSST, SKA, ...)

Jean-Michel ALIMI
Thank you for your attention
File system and runtime system for Big Data

Osamu Tatebe
University of Tsukuba
I/O performance requirement by exascale applications

- **Computational Science (Climate, CFD, ...)**
  - Read initial data (100TB~PB)
  - Write snapshot data (100TB~PB) periodically

- **Data Intensive Science (Particle Physics, Astrophysics, Life Science, ...)**
  - Data analysis of 10PB~EB experiment data
Scalable performance requirement for Parallel File System

<table>
<thead>
<tr>
<th>Year</th>
<th>FLOPS</th>
<th>#cores</th>
<th>IO BW</th>
<th>IOPS</th>
<th>Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>2008</td>
<td>1P</td>
<td>100K</td>
<td>100GB/s</td>
<td>O(1K)</td>
<td>Jaguar, BG/P</td>
</tr>
<tr>
<td>2011</td>
<td>10P</td>
<td>1M</td>
<td>1TB/s</td>
<td>O(10K)</td>
<td>K, BG/Q</td>
</tr>
<tr>
<td>2015</td>
<td>100P</td>
<td>10M</td>
<td>10TB/s</td>
<td>O(100K)</td>
<td></td>
</tr>
<tr>
<td>2018~2020</td>
<td>1E</td>
<td>100M</td>
<td>100TB/s</td>
<td>O(1M)</td>
<td></td>
</tr>
</tbody>
</table>

IO BW and IOPS are expected to be scaled-out in terms of # cores or # nodes
Technology trend

• HDD performance not increase so much
  – 300 MB/s, 5 W in 2020
  – 100 TB/s means $O(10^6)W$ 😞

• Flash, storage class memory
  – 1 GB/s, 0.1 W in 2020 😊
  – Cost, limited number of updates 😞

• Interconnects
  – 62 GB/s (Infiniband 4xHDR)
Current parallel file system

- Central storage array
- Separate installation of compute nodes and storage
- Network BW between compute nodes and storage needs to be scaled-up to scale out the I/O performance
Remember memory architecture

Shared memory  Distributed memory
Scaled-out parallel file system

• Distributed storage in compute nodes
• I/O performance would be scaled out by accessing near storage unless metadata performance is bottleneck
  – Access to near storage mitigates network BW requirement
  – The performance may be non uniform
Example of Scale-out Storage Architecture

- 3 years later snapshot
- Non-uniform but scale-out storage
- R&D of system software stacks is required to achieve maximum I/O performance for data-intensive science

CPU (2 sockets x2.0GHzx16 cores x32FPU)

- 12 Gbps SAS x 16
- 19.2 GB/s, 16 TB

Infiniband HDR

- 62 GB/s
- x 10

Metadata server

1TB local storage

- 12 Gbps SAS x 16
- 9.6 TB/s, 8 PB

9.6 TB/s, 8 PB x 500

96 TB/s, 80 PB
Challenge

• File system
  – Central storage cluster to distributed storage cluster
  – Scaled out parallel file system up to O(1M) clients
    • Scaled out MDS performance

• Runtime system
  – Optimization for non uniform storage access “NUSA”
Scaled out parallel file system

• Federate local storage in compute nodes
  – Special purpose
    • Google file system [SOSP’03]
    • Hadoop file system (HDFS)
  – POSIX(-like)
    • Gfarm file system [CCGrid’02, NGC’10]
Scaled-out MDS

- **GIGA+ [Swapnil Patil et al. FAST’11]**
  - Incremental directory partitioning
  - Independent locking in each partition
- **skyFS [Jing Xing et al. SC’09]**
  - Performance improvement during directory partitioning in GIGA+
- **Lustre**
  - MT scalability in 2.X
  - Proposed clustered MDS
- **PPMDS [Our JST CREST R&D]**
  - Shared-nothing KV stores
  - Nonblocking software transactional memory (*No lock*)

<table>
<thead>
<tr>
<th></th>
<th>IOPS (file creates per sec)</th>
<th>#MDS (#core)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GIGA+</td>
<td>98K</td>
<td>32 (256)</td>
</tr>
<tr>
<td>skyFS</td>
<td>100K</td>
<td>32 (512)</td>
</tr>
<tr>
<td>Lustre 2.4</td>
<td>80K</td>
<td>1 (16)</td>
</tr>
<tr>
<td><strong>PPMDS</strong></td>
<td><strong>157K</strong></td>
<td><strong>15 (240)</strong></td>
</tr>
</tbody>
</table>
Locality aware process scheduling

- Multiconstraint graph partitioning (MCGP) for workflow DAG [Tanaka et al, CCGrid 2012]
  - Minimize data transfer between nodes
  - Maximize parallelism
Summary

• App IO requirement
  – Computational Science
    • Scaled-out IO performance up to O(1M) nodes (100TB to 1PB per hour)
  – Data Intensive Science
    • Data processing for 10PB to 1EB data (>100TB/sec)

• File system and runtime R&D for scale out storage architecture
  – Central storage cluster to distributed storage cluster
    • Network wide RAID
    • Scaled out MDS
  – Runtime system for non uniform storage access “NUSA”
    • Locality aware process scheduling
Big Data Processing in Large-Scale Graph Analytics and Billion-Scale Social Simulation

Toyo Suzumura
IBM Research – Tokyo
Tokyo Institute of Technology
Outline

- Large-Scale Graph Analytics
- Towards Continuous Billion-Scale Social Simulation with Streaming Sensor data
Large-Scale Graph Analytics

Toyotaro Suzumura, Miyuru Dayarathna, Koji Ueno, Masaru Watanabe and ScaleGraph Team

Suzumura Laboratory,
Department of Computer Science
Graduate School of Information Science and Engineering
Tokyo Institute of Technology, Japan
Large-Scale Graph Mining is Everywhere

- Cybersecurity
- Medical Informatics
- Data Enrichment
- Social Networks
- Symbolic Networks

Internet Map

Social Networks

Symbolic Networks: Human Brain

Protein Interactions

Cyber Security (15 billion log entries / day for large enterprise)
Large-Scale Graph Processing System (2011-2016)

Sensors
- Smart Meters
- Smart Grid
- GPS
- SNS (Twitter)

Data Source

Large-Scale Graph Visualization

Real-Time Graph Stream Processing

Large-Scale Graph Library

Centrality

Shortest Path

Quickest Flow Problem

PageRank / RWR

Clustering

Semi-Definite Programming

Mix Integer Programming

Real-Time Stream Processing System

X10 Language

100 Peta Flops Heterogeneous Supercomputer

Large-Scale Graph Store
Understanding time-series nature of large-scale social networks (e.g. separation of degree, diameter, clustering, ..)

Crawled the entire Twitter follower/followee network of 826.10 million vertices and 29.23 billion edges. How could we analyze this gigantic graph?

Supercomputers
Degree of Separation and Diameter for Time-Evolving Twitter Network
Workflow for Temporal Analysis (1/3)

- Convert Twitter user profile and network files to input format for WebGraph API

Hadoop Serialized

91GB

User Profile (xml)

get ID

1

4nodes 20min.

.gz 10GB

ID list sorted by creation time

assign all IDs to new serial IDs

2

.gz 231GB

Follower Friend Network (adjacency list)

replace destination ID to new serial ID

3

8nodes 1hour.

Intermediate Graph data (edge list)

replace source ID to new serial ID

4

8nodes 1hour.

.gz 113GB

Numbering Graph data (edge list)

: with Python Script

: with Hadoop

Not use HDFS. Use only GPFS. Hadoop can read directly gzip files

91GB

91GB

91GB
Workflow for Temporal Analysis (2/3)

1. Numbering Graph data (edge list) compressed
   - .gz 113GB

2. Remove nodes and edges for timestamp graph (every 3 months)
   - 8 nodes 1 hour

3. Input format divided by hadoop reducer
   - 12/2006 .gz 300KB
   - 9/2012 .gz 85GB

Sequential processing on each timestamp graph

4. Merge
   - 6 nodes

5. Merge

6. Merge

Parallel processing every month in one go

- Input format for WebGraph API
  - 12/2006 raw text .gz 892KB
  - 9/2012 raw text 263GB

Total data size: 500GB (every 3 months)

Total data size: 1.5TB (every 3 months)

- Decompressed data size 1.5 TB
Workflow for Temporal Analysis (3/3)

- **Parallel processing every month in one go**
- **Sequential processing on each timestamp graph**

**Input format for WebGraph API**
- 12/2006 raw text 892KB
- ... 9/2012 raw text 263GB

**BVGraph for WebGraph API**
- Compression: 1 node 7 hours
- 2006年 Object 240KB
- 2012年 Object 73GB

**Total data size:**
- 1.5TB (every 3 months)
- 470 GB (every 3 months)

**Compute Degree of Separation and Diameter with HyperANF**
Workflow: Degree of Separation

- Use HyperANF in WebGraph on TSUBAME 2.0 Fat Node
  - take 16 hours with 1 node (64 cores, 512 GB RAM)

Total data size: 470 GB (every 3 months)
Programming models that offer performance and programmer productivity are very important for conducting big data analytics in Exascale Systems.

HPCS languages are an example for such initiatives.

It is very important for having complex network analysis software APIs in such languages.

Crawled the entire Twitter follower/followee network of 826.10 million vertices and 28.84 billion edges. How could we analyze this gigantic graph?
**ScaleGraph**: Large-Scale Graph Analytics Library

- **Aim** - Create an X10-based Large Scale Graph Analytics Library (beyond the scale of billions of vertices and edges).

- **Objectives**
  - To define concrete abstractions for Massive Graph Processing
  - To investigate use of X10 (i.e., PGAS languages) for massive graph processing
  - To support significant amount of graph algorithms (E.g., structural properties, clustering, community detection, etc.)
  - To create well defined interfaces to Graph Stores
  - To evaluate performance of each measurement algorithms and applicability of ScaleGraph using real/synthetic graphs in HPC environments.

**URL**: https://sites.google.com/site/scalegraph/
Programming Language X10

X10 is a new parallel distributed programming language being developed by IBM Research.

- X10 aims at improving the productivity of highly parallel and distributed applications.
  - Enables scalable programming for parallel distributed environment, where many multicore SMP chips and GPGPUs are interconnected.

- X10 adopts APGAS (Asynchronous Partitioned Global Address Space) programming model.
  - Can manage multiple machines as a global memory space partitioned into “Places”.
  - Can create lightweight asynchronous “Activities”.
  - Supports creation and reference of activities and objects in remote places.

- X10 supports various execution environments.
  - Can run both on Java execution environments and native environments.
  - Provides development tools integrated into Eclipse.

- X10 is being developed as an open source project.
  - See http://x10-lang.org/ for more information.

```java
public class MyDistCalc {
    public static def main(Array[String]) {
        val R = 1..1000; val D = Dist.makeBlock(R);
        val arr = DistArray.make[Int](D, ([i]:Point=>i));

        val places = arr.dist.places();
        val tmp = new Array[Int](places.size);
        finish for (i in 0..places.size-1) async {
            tmp(i) = at (places(i)) {
                val a = arr | here;
                var s:int = 0; for (pt in a) s += a(pt)*a(pt);
                s // return value of at
            }
        }

        var result:Int; for (pt in tmp) result += tmp(pt);
        Console.OUT.println(result); // -> 333833500

        // We can actually use DistArray.map and reduce
        val r = arr.map((i:Int)=>i).reduce(Int, +, 0);
        Console.OUT.println(r); // -> 333833500
    }
}
```

Distributed programming by X10
Graph500 is a new benchmark that ranks supercomputers by executing a large-scale graph search problem.

The benchmark is ranked by so-called **TEPS (Traversed Edges Per Second)** that measures the number of edges to be traversed per second by searching all the reachable vertices from one arbitrary vertex with each team’s optimized BFS (Breadth-First Search) algorithm.
Highly Scalable Graph Search Method for the Graph500 Benchmark

- We propose an optimized method based on 2D based partitioning and other various optimization methods such as communication compression and vertex sorting.
- We developed CPU implementation and GPU implementation.
- Our optimized GPU implementation can solve BFS (Breadth First Search) of large-scale graph with $2^{35}$ (34.4 billion) vertices and $2^{39}$ (550 billion) edges for 1.275 seconds with 1366 nodes and 4096 GPUs on TSUBAME 2.0.
- This record corresponds to 431 GTEPS.

![Performance Comparison with CPU and GPU Implementations](chart1)

![Scalable 2D partitioning based CPU Implementation with Scale 26 per 1 node](chart2)
Towards Continuous Billion-Scale Social Simulation with Real-Time Streaming Data

Toyotaro Suzumura
IBM Research – Tokyo
Tokyo Institute of Technology
Background: Large-scale Simulation is Everywhere

- We have entered into the era where proactive response is needed.
- Highly performance large-scale based simulation is required for timely decision.

**Overview and Goal:** A scalable large-scale agent simulation platform based on X10 that runs on various computing environment from a small cluster to Supercomputers with ten thousands of CPU cores and high speed network.

**Speed-up against 8 nodes**

Simulation Data: Japan Whole Country

- # of trips: 10 million
- # of simulation steps: 600

Environment: Intel Xeon Cluster (1 node : 12 cores)
XAXIS : Highly Scalable Agent-based Simulation Platform

XAXIS is a highly scalable general-purpose and agent-based simulation platform that runs on top of a wide range of systems from a single core to thousands of cores.
XAXIS: X10-based Agents eXecutive Infrastructure for Simulation

- **X10-based Distributed Agent Simulation Platform**
  - X10 is the state-of-the-art PGAS (Partitioned Global Address Space) language that brings high productivity when implementing highly parallel and distributed applications on post-peta or exascale machines
    - X10 provides the functionality that can seamlessly integrate with legacy applications written in Java or C++.

- **Programming Model**
  - The agent programming model of XAXIS is derived from our ZASE [Yamamoto, AAMAS2007] simulation platform
  - XAXIS provides compatible API interface of ZASE to developers.

The following diagram illustrates the software stack of XAXIS and its applications.

- Social Simulation (Java) (e.g. Traffic, CO2 Emission, Auction, Marketing)
- XAXIS API
- Java Bridge
- XAXIS : X10-Based Simulation Runtime
- X10 (Java, C++)
- Social Simulation (X10)
- X10 Bridge
We have achieved descent scalability by increasing the number of nodes up to 256 nodes, by performing the whole country-wide simulation with 600 simulation steps and 10 million vehicles with both Managed X10 and Native X10.

As shown in the figures below, real-time simulation is achieved in that especially it only takes 70 seconds to simulation 600 simulation steps with Native X10.

Experimental Environment: X10 2.3.1 Linux 2.6, Infiniband QDR Network, Intel Xeon5670 (12 Cores), 5GB RAM, OpenMPI.
Towards Continuous Billion-Scale Social Simulation with Real-Time Streaming Data

- **Applications**
  - Target Area: *Planet* (Open Street Map)
  - 7 billion people

- **Input Data**
  - Road Network (Open Street Map) for Planet: **300 GB** (XML)
  - Trip data for 7 billion people
    - 10 KB (1 trip) x 7 billion = 70 TB
  - Real-Time Streaming Data (e.g. Social sensor, physical data)

- **Simulated Output for 1 Iteration**
  - 700 TB
Summary

- **Software:**
  - What software are you currently using to manage and explore your data?
    - X10, Hadoop/HDFS, GPFS, MPI, ..
  - What algorithms and software libraries/tools need development and improvement to address your big data needs?
    - Scalable distributed algorithm for various large-scale graph analytics (e.g. ScaleGraph)
    - Easy-to-use interface for in-situ analysis for the above analytics
    - More advanced integrated development environment for PGAS languages (e.g. X10)

- **Architecture (Operational Aspect)**
  - From the operational aspects of supercomputers, it would be required to accept real-time streaming sensor data from outside

- **Taxonomy:** Graph data format (e.g. GML)

- **Workflows:**
  - A workflow that would support something like large-scale network analysis containing real-time streaming data processing, Hadoop-typed batched jobs, and large-scale graph analysis