Benchmarking Big Data in the Data Center: A TU Delft Perspective

Alexandru Iosup
Delft University of Technology
The Netherlands

Team: Undergrad Tim Hegeman, ... Grad Yong Guo, Mihai Capota, Bogdan Ghit
Researchers Marcin Biczak, Otto Visser  Staff Henk Sips, Dick Epema
Collaborators* Ana Lucia Varbanescu (UvA, Ams), Claudio Martella (VU, Giraph), KIT, Intel Research Labs, IBM TJ Watson, SAP, Google Inc. MV, Salesforce SF, ...

* Not their fault for any mistakes in this presentation. Or so they wish.
(TU) Delft – the Netherlands – Europe

Pop. 100,000
Founded 13th century
Pop. 16.5 M

Pop. 13,000
Founded 1842

Potsdam (We are here)
The Parallel and Distributed Systems Group at TU Delft

Alexandru Iosup
Grids/Clouds
P2P systems
Big Data
Online gaming

Dick Epema
Grids/Clouds
P2P systems
Video-on-demand e-Science

Ana Lucia Varbanescu (now UvA)
HPC systems
Multi-cores
Big Data e-Science

Henk Sips
HPC systems
Multi-cores
P2P systems

Johan Poutwelse
P2P systems
File-sharing
Video-on-demand

Home page
• www.pds.ewi.tudelft.nl

Publications
• see PDS publication database at publications.st.ewi.tudelft.nl

Winners IEEE TCSC Scale Challenge 2014
What is Cloud Computing? A Descendant* of the Grid Idea

* Subset.

“A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities [+] nontrivial QoS.” I. Foster, 1998 + 1999

Lessons From Grids

From Parallel to Many-Task Computing (users, not designers, find what works)

Show the numbers, please!

- Average job size is 1 (that is, there are **no tightly-coupled, only conveniently parallel jobs**)


Lessons From Grids

- 99.9(...9%) reliable
- 5x decrease in failure rate after first year [Schroeder and Gibson, DSN'06]

Production

- >10% jobs fail [Iosup et al., CCGrid'06]

DAS

- 20-45% failures [Khalili et al., Grid'06]

TeraGrid

- 27% failures, 5-10 retries [Dumitrescu et al., GCC'05]

Grid3

CERN LCG jobs

- 74.71% successful
- 25.29% unsuccessful

Source: dboard-gr.cern.ch, May'07.

Large-scale = unreliable infrastructure

Resource management is key

- Server
- Cluster
- Production

Resource management is key

Lessons from Grids, via a Detour

The Overwhelming Growth of Knowledge

“When 12 men founded the Royal Society in 1660, it was possible for an educated person to encompass all of scientific knowledge. […] In the last 50 years, such has been the pace of scientific advance that even the best scientists cannot keep up with discoveries at frontiers outside their own field.”

Tony Blair, PM Speech, May 2002

Professionals already know they don’t know [it all]

<table>
<thead>
<tr>
<th>Number of Publications</th>
<th>1993</th>
<th>1997</th>
<th>2001</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>12,428,733</td>
<td>1,265,808</td>
<td></td>
</tr>
<tr>
<td></td>
<td>713,730</td>
<td>1,347,985</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1,323,933</td>
<td>342,535</td>
<td></td>
</tr>
<tr>
<td></td>
<td>206,993</td>
<td>318,286</td>
<td></td>
</tr>
<tr>
<td></td>
<td>84,851</td>
<td>336,858</td>
<td></td>
</tr>
<tr>
<td>France</td>
<td>203,814</td>
<td>232,058</td>
<td></td>
</tr>
<tr>
<td>Canada</td>
<td>168,331</td>
<td>166,216</td>
<td></td>
</tr>
<tr>
<td>Italy</td>
<td>122,398</td>
<td>147,023</td>
<td></td>
</tr>
<tr>
<td>Switzerland</td>
<td>57,664</td>
<td>66,761</td>
<td></td>
</tr>
<tr>
<td>Netherlands</td>
<td><strong>83,600</strong></td>
<td><strong>92,526</strong></td>
<td></td>
</tr>
</tbody>
</table>

Data: King, The scientific impact of nations, Nature’04.
Lessons from Grids

From Hypothesis to Data

The Fourth Paradigm is suitable for professionals who already know they don’t know [enough to formulate good hypotheses], yet need to deliver quickly.

• Last few decades: a computational branch simulating complex phenomena

• Today (the Fourth Paradigm): data exploration
  unify theory, experiment, and simulation
  • Data captured by instruments or generated by simulator
  • Processed by software
  • Information/Knowledge stored in computer
  • Scientist analyzes results using data management and statistics

The Vision: Everyone Is a Scientist!  
(the Fourth Paradigm)

• Data as individual right, enabling high-quality lifestyle of individuals and modern societal services
• Data as workhorse in creating commercial services by SMEs (~60% gross value added, for many years)

EU reasons to address Big Data challenges
>500 million people
>85 million employees
>3 trillion euros / year gross value added

Data at the Core of Our Society: The LinkedIn Example

The State of LinkedIn

A very good resource for matchmaking workforce and prospective employers

Vital for your company’s life, as your Head of HR would tell you

Vital for the prospective employees

Sources: Vincenzo Cosenza, The State of LinkedIn, http://vincos.it/the-state-of-linkedin/
Data at the Core of Our Society: The LinkedIn Example

The State of LinkedIn

3-4 new users every second

Great, if you can process this graph: opinion mining, hub detection, etc.

150,000,000 registered members

Feb 2012

100M Mar 2011, 69M May 2010

Sources: Vincenzo Cosenza, The State of LinkedIn, http://vincos.it/the-state-of-linkedin/
Data at the Core of Our Society: The LinkedIn Example

The State of LinkedIn

3-4 new users every second

Great, if you can process this graph: opinion mining, hub detection, etc.

139/277 million questions of customer retention, so time-based analytics

Sources: Vincenzo Cosenza, The State of LinkedIn, http://vincos.it/the-state-of-linkedin/
LinkedIn Is Part of the “Data Deluge”

Data Deluge = data generated by humans and devices (IoT)
- Interacting
- Understanding
- Deciding
- Creating

Sources: IDC, EMC.
The Data Deluge Is A Challenge for Tech But Good for Us[ers]

- All human knowledge
  - Until 2005: 150 Exa-Bytes
  - 2010: 1,200 Exa-Bytes

- Online gaming (Consumer)
  - 2002: 20TB/year/game
  - 2008: 1.4PB/year/game (only stats)

- Public archives (Science)
  - 2006: GBs/archive
  - 2011: TBs/year/archive

Sources: Vincenzo Cosenza, The State of LinkedIn, http://vincos.it/the-state-of-linkedin/
The Challenge: The Three “V”s of Big Data*

When You Can, Keep *and* Process Everything

* New Vs later: ours is “vicissitude”

- **Volume**
  - More data vs. better models
  - Exponential growth + iterative models
  - Scalable storage and distributed queries

- **Velocity**
  - Speed of the feedback loop
  - Gain competitive advantage: fast recommendations
  - Analysis in near-real time to extract value

- **Variety**
  - The data can become messy: text, video, audio, etc.
  - Difficult to integrate into applications

Too big, too fast, does not comply with traditional DB

The “Data Deluge”

Data Deluge = data generated by humans and devices (IoT)

- Interacting
- Understanding
- Deciding
- Creating

Sources: IDC, EMC.
Can We Afford This Vision? The Current Tech
Big Data = Systems of Systems

Adapted from: Dagstuhl Seminar on Information Management in the Cloud, http://www.dagstuhl.de/program/calendar/partlist/?semnr=11321&SUOG
Can We Afford This Vision? The Current Tech
Monolithic Systems

• Monolithic
  • Integrated stack
    (can still learn from decades of sw.eng.)
  • Fixed set of homogeneous resources
    (we forgot 2 decades of distrib.sys.)
  • Execution engines do not coexist
    (we’re running now MPI inside Hadoop Maps,
     Hadoop jobs inside MPI processes, etc.)
  • Little performance information is exposed
    (we forgot 4 decades of par.sys., MPPs)
  • ...

Stuck in stacks!

A. L. Varbanescu and A. Iosup, On Many-Task Big Data Processing: from
GPUs to Clouds. Proc. of SC\textsuperscript{12} (MTAGS).
The “Big Data cake” in the Data Center

Online Social Networks = Hadoop / MapReduce framework

Financial Analysts

Universe Explorers

Multiple frameworks = Isolation, especially performance

Big Data Enthusiast
The Challenge: Can We Afford This Vision? Not with the Current Resources (An Anecdote)

Time magazine reported that it takes 0.0002 kWh to stream 1 minute of video from the YouTube data centre...

Based on Jay Walker’s recent TED talk, 0.01 kWh of energy is consumed on average in downloading 1 MB over the Internet.

The average Internet device energy consumption is around 0.001 kWh for 1 minute of video streaming.

For 1.6 B downloads of this 17 MB file and streaming for 4 minutes gives the overall energy for this one pop video in one year...

>300 GWh = more than some countries in a year, >35 MW of 24/7/365 diesel, >100 M liters of oil, 80,000 cars running for a year, ...

Note: Psy has now >2.75 billion views, so roughly 450 GWh (Jun 2014).
Can We Afford This Vision?  
Not with the Current Resources

- Energy resources

Global power consumption

Breakdown of EU power consumption

Data Source: Powering the Datacenter, DatacenterDynamics, 2013
One-third of global data center energy use is in U.S., but growth rates are fastest in emerging economies.

Sources: DatacenterDynamics and Jon Summers, UoL, UK.
Everyone is a Scientist!
Can We Afford This Vision?

I don’t know.
But we need to become very efficient.
For this, we need to combine sw.eng., distr.sys., parallel sys., DB.
Then, we need to show numbers!
Why Big Data Benchmarking?

- Establish and share best-practices in giving quantitative answers to important questions about Big Data

- Use in procurement
- Use in system design
- Use in system tuning and operation
- Use in performance management

- Use in training
Big Data in the Data Center: 10 Main Challenges in 4 Categories*

- **Methodological**
  1. Experiment compression, both design and actual evaluation
  2. Beyond black-box testing through testing short-term dynamics and long-term evolution
  3. Impact of middleware

- **System-Related**
  1. Reliability, availability, and system-related properties
  2. Massive-scale, multi-site benchmarking
  3. Performance isolation, multi-tenancy models

- **Workload-related**
  1. Workload = Dataset + Activity
  2. Statistical workload models + analysis of coverage
  3. Benchmarking performance isolation under various multi-tenancy workloads

- **Metric-Related**
  1. Beyond traditional performance: variability, elasticity, cost, etc.
  2. Uniform reporting

---

Iosup et al., IaaS Cloud Benchmarking: Approaches, Challenges, and Experience, MTAGS 2012.

SPEC Research Group (RG)

The Research Group of the Standard Performance Evaluation Corporation

Mission Statement

- Provide a platform for collaborative research efforts in the areas of computer benchmarking and quantitative system analysis
- Provide metrics, tools and benchmarks for evaluating early prototypes and research results as well as full-blown implementations
- Foster interactions and collaborations btw. industry and academia

Find more information on: http://research.spec.org
A Call to Arms

• Defining workloads
• Understanding the metrics, datasets, and algorithms used in practice: fill in our survey [http://goo.gl/TJwkTg](http://goo.gl/TJwkTg)
• Evaluating and reporting on various platforms

Join us within the SPEC RG Cloud Working Group
[http://research.spec.org/working-groups/rg-cloud-working-group.html](http://research.spec.org/working-groups/rg-cloud-working-group.html)
Agenda

1. Everyone is a scientist!
2. Benchmarking: let’s show the numbers
3. Datacenter Workloads
4. Cloud Performance & Perf. Variability
5. Performance of Graph-Processing Platforms (Giraph, GraphLab, …)
6. BitTorrent World: A MapReduce Workflow
7. Elastic MapReduce Performance
8. Conclusion
Agenda

1. Everyone is a scientist!
2. Benchmarking: let’s show the numbers
3. Conclusion
Data Center Workloads: Our Team

Alexandru Iosup  
TU Delft

BoTs  
Workflows  
Grids  
Big Data  
Statistical modeling

Dick Epema  
TU Delft

BoTs

Mathieu Jan  
TU Delft/INRIA

BoTs  
Statistical modeling

Ozan Sonmez  
TU Delft

BoTs

Thomas de Ruiter  
TU Delft

MapReduce  
Big Data  
Statistical modeling

Radu Prodan  
U.Isbk.

Thomas Fahringer  
U.Isbk.

Simon Ostermann  
U.Isbk.

Workflows  
Workflows  
Workflows
Statistical MapReduce Models From Long-Term Usage Traces

- Started 2010, excellent studies now exist
- Real traces
  - Yahoo
  - Google
  - 2 x Social Network Provider
  - (currently looking at 2 SME traces)

<table>
<thead>
<tr>
<th>Model</th>
<th>Tasks</th>
<th>Correlation</th>
<th>Map/Reduce Modeled</th>
<th>Sign. Level</th>
<th>Indirect Distr. Sel.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complex Model</td>
<td>Indirect</td>
<td>Run time – Disk</td>
<td>Separately</td>
<td>0.05</td>
<td>Best fits</td>
</tr>
<tr>
<td>Relaxed Complex Model</td>
<td>Indirect</td>
<td>Run time – Disk</td>
<td>Separately</td>
<td>0.02</td>
<td>All fits</td>
</tr>
<tr>
<td>Safe Complex Model</td>
<td>Direct</td>
<td>Run time – Disk</td>
<td>Separately</td>
<td>0.05</td>
<td>–</td>
</tr>
<tr>
<td>Simple Model</td>
<td>Direct</td>
<td>–</td>
<td>Together</td>
<td>0.05</td>
<td>–</td>
</tr>
</tbody>
</table>

August 6, 2014
Graph Processing Workloads

- No representative workloads, perhaps even algorithm coverage is difficult to analyze
- See work on graph processing
What is a Bag of Tasks (BoT)? A System View

BoT = set of jobs sent by a user...

\[ W_u = \{ J_i | user(J_i) = u \} \]

...that is submitted at most \( \Delta s \) after the first job

\[ ST(J') \leq ST(J) + \Delta \]

- Why Bag of Tasks? From the perspective of the user, jobs in set are just tasks of a larger job
- A single useful result from the complete BoT
- Result can be combination of all tasks, or a selection of the results of most or even a single task

Applications of the BoT Programming Model

• Parameter sweeps
  • Comprehensive, possibly exhaustive investigation of a model
  • Very useful in engineering and simulation-based science

• Monte Carlo simulations
  • Simulation with random elements: fixed time yet limited inaccuracy
  • Very useful in engineering and simulation-based science

• Many other types of batch processing
  • Periodic computation, Cycle scavenging
  • Very useful to automate operations and reduce waste
BoTs Are the Dominant Programming Model for Grid Computing (Many Tasks)

Mostly conveniently parallel jobs: 1 CPU
Perhaps multi-threaded apps.

Job runtime: several hours average.
Systems with half-hour average exist.

Memory requirements: modest, except High Energy Physics jobs.

Iosup et al., The Grid Workloads Archive, FGCS, 2008.
### BoTs by numbers: I/O, Files, Remote Sys

<table>
<thead>
<tr>
<th>T-12 part</th>
<th>I/O [KOps]</th>
<th>I/O Traffic [MB]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>Rd</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **I/O**: modest, except HEP
- **Rd:Wr** varies widely
- **I/O,HEP**: 65MBps/experiment
- **Upper bound for typical sci.apps.**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total</td>
<td>In</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Remote Sys.**: small Xfers, latency important
- **Netw**: 2-10GB, input mostly

Statistical BoT Workload Model

- Single arrival process for both BoTs and parallel jobs
- Validated with 7 grid workloads

What is a Wokflow?

WF = set of jobs with precedence (think Direct Acyclic Graph)
Applications of the Workflow Programming Model

• Complex applications
  • Complex filtering of data
  • Complex analysis of instrument measurements

• Applications created by non-CS scientists*
  • Workflows have a natural correspondence in the real-world, as descriptions of a scientific procedure
  • Visual model of a graph sometimes easier to program

• Precursor of the MapReduce Programming Model (next slides)

**Workflows Exist in Grids, but Did No Evidence of a Dominant Programming Model**

### Traces

<table>
<thead>
<tr>
<th>Trace</th>
<th>Source</th>
<th>Duration</th>
<th>Number of WFs</th>
<th>Number of Tasks</th>
<th>CPUdays</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>DEE</td>
<td>09/06-10/07</td>
<td>4,113</td>
<td>122k</td>
<td>152</td>
</tr>
<tr>
<td>T2</td>
<td>EE2</td>
<td>05/07-11/07</td>
<td>1,030</td>
<td>46k</td>
<td>41</td>
</tr>
</tbody>
</table>

### Selected Findings

Loose coupling
- Graph with 3-4 levels
- Average WF ~10s of jobs
- 75% WFs are <=40 jobs
  - 95% are <=200 jobs
- 85% WFs take <10 mins

---

1. Everyone is a scientist!
2. Benchmarking: let's show the numbers
3. Conclusion
Cloud Performance and Performance Variability: Our Team

Alexandru Iosup
TU Delft

Dick Epema
TU Delft

Nezih Yigitbasi
TU Delft

Athanasios Antoniou
TU Delft

Performance Variability
Isolation
Multi-tenancy
Benchmarking

Performance IaaS clouds

Performance Variability

Performance Isolation

Radu Prodan
U.Isbk.

Thomas Fahringer
U.Isbk.

Simon Ostermann
U.Isbk.

Benchmarking

Benchmarking

Benchmarking
Some Previous Work
(>50 important references across our studies)

Virtualization Overhead
- Loss below 5% for computation [Barham03] [Clark04]
- Loss below 15% for networking [Barham03] [Menon05]
- Loss below 30% for parallel I/O [Vetter08]
- Negligible for compute-intensive HPC kernels [You06] [Panda06]

Cloud Performance Evaluation
- Performance and cost of executing a sci. workflows [Dee08]
- Study of Amazon S3 [Palankar08]
- Amazon EC2 for the NPB benchmark suite [Walker08] or selected HPC benchmarks [Hill08]
- CloudCmp [Li10]
- Kosmann et al.
Production IaaS Cloud Services

- **Production IaaS cloud:** lease resources (infrastructure) to users, operate on the market and have active customers

<table>
<thead>
<tr>
<th>Name</th>
<th>Cores (ECUs)</th>
<th>RAM [GB]</th>
<th>Archi. [bit]</th>
<th>Disk [GB]</th>
<th>Cost [$/h]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Amazon EC2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>m1.small</td>
<td>1 (1)</td>
<td>1.7</td>
<td>32</td>
<td>160</td>
<td>0.1</td>
</tr>
<tr>
<td>m1.large</td>
<td>2 (4)</td>
<td>7.5</td>
<td>64</td>
<td>850</td>
<td>0.4</td>
</tr>
<tr>
<td>m1.xlarge</td>
<td>4 (8)</td>
<td>15.0</td>
<td>64</td>
<td>1,690</td>
<td>0.8</td>
</tr>
<tr>
<td>c1.medium</td>
<td>2 (5)</td>
<td>1.7</td>
<td>32</td>
<td>350</td>
<td>0.2</td>
</tr>
<tr>
<td>c1.xlarge</td>
<td>8 (20)</td>
<td>7.0</td>
<td>64</td>
<td>1,690</td>
<td>0.8</td>
</tr>
<tr>
<td><strong>GoGrid (GG)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GG.small</td>
<td>1</td>
<td>1.0</td>
<td>32</td>
<td>60</td>
<td>0.19</td>
</tr>
<tr>
<td>GG.large</td>
<td>1</td>
<td>1.0</td>
<td>64</td>
<td>60</td>
<td>0.19</td>
</tr>
<tr>
<td>GG.xlarge</td>
<td>3</td>
<td>4.0</td>
<td>64</td>
<td>240</td>
<td>0.76</td>
</tr>
<tr>
<td><strong>Elastic Hosts (EH)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EH.small</td>
<td>1</td>
<td>1.0</td>
<td>32</td>
<td>30</td>
<td>£0.042</td>
</tr>
<tr>
<td>EH.large</td>
<td>1</td>
<td>4.0</td>
<td>64</td>
<td>30</td>
<td>£0.09</td>
</tr>
<tr>
<td><strong>Mosso</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mosso.small</td>
<td>4</td>
<td>1.0</td>
<td>64</td>
<td>40</td>
<td>0.06</td>
</tr>
<tr>
<td>Mosso.large</td>
<td>4</td>
<td>4.0</td>
<td>64</td>
<td>160</td>
<td>0.24</td>
</tr>
</tbody>
</table>

Our Method

• Based on general performance technique: model performance of individual components; system performance is performance of workload + model [Saavedra and Smith, ACM TOCS’96]

• Adapt to clouds:
  1. Cloud-specific elements: resource provisioning and allocation
  2. Benchmarks for single- and multi-machine jobs
  3. Benchmark CPU, memory, I/O, etc.: 

<table>
<thead>
<tr>
<th>Type</th>
<th>Suite/Benchmark</th>
<th>Resource</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>SI</td>
<td>LMBench/all [24]</td>
<td>Many</td>
<td>Many</td>
</tr>
<tr>
<td>SI</td>
<td>Bonnie/all [25], [26]</td>
<td>Disk</td>
<td>MBps</td>
</tr>
<tr>
<td>SI</td>
<td>CacheBench/all [27]</td>
<td>Memory</td>
<td>MBps</td>
</tr>
<tr>
<td>MI</td>
<td>HPCC/HPL [28], [29]</td>
<td>CPU</td>
<td>GFLOPS</td>
</tr>
<tr>
<td>MI</td>
<td>HPCC/DGEMM [30]</td>
<td>CPU</td>
<td>GFLOPS</td>
</tr>
<tr>
<td>MI</td>
<td>HPCC/STREAM [30]</td>
<td>Memory</td>
<td>GBps</td>
</tr>
<tr>
<td>MI</td>
<td>HPCC/RandomAccess [31]</td>
<td>Network</td>
<td>MUPS</td>
</tr>
<tr>
<td>MI</td>
<td>HPCC/$b_{eff}$(lat,bw.) [32]</td>
<td>Comm.</td>
<td>μs, GBps</td>
</tr>
</tbody>
</table>

Single Resource Provisioning/Release

- Time depends on instance type
- Boot time non-negligible

**Multi-Resource Provisioning/Release**

- Time for *multi*-resource increases with number of resources

---

CPU Performance of Single Resource

- ECU definition: “a 1.1 GHz 2007 Opteron” ~ 4 flops per cycle at full pipeline, which means at peak performance one ECU equals 4.4 gigaflops per second (GFLOPS)
- Real performance 0.6..0.1 GFLOPS = ~1/4..1/7 theoretical peak

HPLinpack Performance (Parallel)

- Low efficiency for parallel compute-intensive applications
- Low performance vs cluster computing and supercomputing

Production Cloud Services

- **Production cloud:** operate on the market and have active customers

- **IaaS/PaaS:** Amazon Web Services (AWS)
  - EC2 (Elastic Compute Cloud)
  - S3 (Simple Storage Service)
  - SQS (Simple Queueing Service)
  - SDB (Simple Database)
  - FPS (Flexible Payment Service)

- **PaaS:** Google App Engine (GAE)
  - Run (Python/Java runtime)
  - Datastore (Database) ~ SDB
  - Memcache (Caching)
  - URL Fetch (Web crawling)

---

Our Method
Performance Traces

- CloudStatus*
  - Real-time values and weekly averages for most of the AWS and GAE services

- Periodic performance probes
  - Sampling rate is under 2 minutes

* [www.cloudstatus.com](http://www.cloudstatus.com)
Our Method

Analysis

1. Find out whether variability is present
   • Investigate several months whether the performance metric is highly variable

2. Find out the characteristics of variability
   • Basic statistics: the five quartiles (Q₀-Q₄) including the median (Q₂), the mean, the standard deviation
   • Derivative statistic: the IQR (Q₃-Q₁)
   • CoV > 1.1 indicate high variability

3. Analyze the performance variability time patterns
   • Investigate for each performance metric the presence of daily/monthly/weekly/yearly time patterns
   • E.g., for monthly patterns divide the dataset into twelve subsets and for each subset compute the statistics and plot for visual inspection

Validated Assumption: The performance delivered by production services is variable.
• **Deployment Latency [s]**: Time it takes to start a small instance, from the startup to the time the instance is available
• Higher IQR and range from week 41 to the end of the year; possible reasons:
  • Increasing EC2 user base
  • Impact on applications using EC2 for auto-scaling

---

• **Get Throughput [bytes/s]:** Estimated rate at which an object in a bucket is read
• The last five months of the year exhibit much lower IQR and range
  • More stable performance for the last five months
  • Probably due to software/infrastructure upgrades

AWS Dataset (3/4): SQS

- **Average Lag Time [s]:** Time it takes for a posted message to become available to read. Average over multiple queues.
- Long periods of stability (low IQR and range)
- Periods of high performance variability also exist
AWS Dataset (4/4): Summary

- **All services exhibit time patterns in performance**
- **EC2:** periods of special behavior
- **SDB and S3:** daily, monthly and yearly patterns
- **SQS and FPS:** periods of special behavior
Summary

- Evaluated several commercial alternatives

- **IaaS clouds**: lower performance than theoretical peak
  - Especially CPU (GFLOPS)
  - Some users have started to lease opportunistically: lease many machines, retain only machines with best performance

- **IaaS and PaaS clouds**: high performance variability
  - Difficult to enforce performance guarantees

---


Agenda

1. Everyone is a scientist!
2. Benchmarking: let’s show the numbers

3. Conclusion
Graph Processing: Our Team

Alexandru Iosup
TU Delft
Big Data & Clouds
Res. management Systems, Benchmarking

Dick Epema
TU Delft
Big Data & Clouds
Res. management Systems

Mihai Capota
TU Delft
Big Data apps
Benchmarking

Ana Lucia Varbanescu
U. Amsterdam
Graph processing
Benchmarking

Claudio Martella
VU Amsterdam
Graph processing

Yong Guo
TU Delft
Graph processing
Benchmarking

Marcin Biczak
TU Delft
Big Data & Clouds
Performance & Development
The data deluge: large-scale graphs

Linkedin

Twitter

Yahoo!

Friendster

XFire

TU Delft
Platform diversity

- Platform: the combined hardware, software, and programming system that is being used to complete a graph processing task.
What is the performance of these platforms?

- Graph500
  - Single application (BFS), Single class of synthetic datasets

- Few existing platform-centric comparative studies
  - Prove the superiority of a given system, limited set of metrics

Our vision: a benchmarking suite for graph processing across all platforms
Our Method
A benchmark suite for performance evaluation of graph-processing platforms

1. **Multiple Metrics**, e.g.,
   - Execution time
   - Normalized: EPS, VPS
   - Utilization

2. **Representative graphs** with various characteristics, e.g.,
   - Size
   - Directivity
   - Density

3. **Typical graph algorithms**, e.g.,
   - BFS
   - Connected components


Selection and Design of Performance Metrics for Graph Processing

- Raw processing power
  - Execution time
  - Actual computation time
  - Edges/Vertices per second
- Resource utilization (sys)
  - CPU, memory, network
- Scalability
  - Horizontal vs. vertical
  - Strong vs. weak
- Overhead
  - Data ingestion time
  - Overhead time
- Elasticity (?)
# Dataset Selection: Application Domains

- Number of vertices, edges, link density, size, directivity, etc.

<table>
<thead>
<tr>
<th>Graphs</th>
<th>#V</th>
<th>#E</th>
<th>d</th>
<th>D̄</th>
<th>Directivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>G1 Amazon</td>
<td>262,111</td>
<td>1,234,877</td>
<td>1.8</td>
<td>4.7</td>
<td>directed</td>
</tr>
<tr>
<td>G2 WikiTalk</td>
<td>2,388,953</td>
<td>5,018,445</td>
<td>0.1</td>
<td>2.1</td>
<td>directed</td>
</tr>
<tr>
<td>G3 KGS</td>
<td>293,290</td>
<td>16,558,839</td>
<td>38.5</td>
<td>112.9</td>
<td>undirected</td>
</tr>
<tr>
<td>G4 Citation</td>
<td>3,764,117</td>
<td>16,511,742</td>
<td>0.1</td>
<td>4.4</td>
<td>directed</td>
</tr>
<tr>
<td>G5 DotaLeague</td>
<td>61,171</td>
<td>50,870,316</td>
<td>2,719.0</td>
<td>1,663.2</td>
<td>undirected</td>
</tr>
<tr>
<td>G6 Synth</td>
<td>2,394,536</td>
<td>64,152,015</td>
<td>2.2</td>
<td>53.6</td>
<td>undirected</td>
</tr>
<tr>
<td>G7 Friendster</td>
<td>65,608,366</td>
<td>1,806,067,135</td>
<td>0.1</td>
<td>55.1</td>
<td>undirected</td>
</tr>
</tbody>
</table>
Graph-Processing Algorithms

- Literature survey of metrics, datasets, and algorithms
  - 10 top research conferences: SIGMOD, VLDB, HPDC ...
  - Key word: graph processing, social network
  - 2009–2013, 124 articles

<table>
<thead>
<tr>
<th>Class</th>
<th>Examples</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Statistics</td>
<td>Diameter, PageRank</td>
<td>16.1</td>
</tr>
<tr>
<td>Graph Traversal</td>
<td>BFS, SSSP, DFS</td>
<td>46.3</td>
</tr>
<tr>
<td>Connected Component</td>
<td>Reachability, BiCC</td>
<td>13.4</td>
</tr>
<tr>
<td>Community Detection</td>
<td>Clustering, Nearest Neighbor</td>
<td>5.4</td>
</tr>
<tr>
<td>Graph Evolution</td>
<td>Forest Fire Model, PAM</td>
<td>4.0</td>
</tr>
<tr>
<td>Other</td>
<td>Sampling, Partitioning</td>
<td>14.8</td>
</tr>
</tbody>
</table>

Platforms we have evaluated

- Distributed or non-distributed
- Graph-specific or generic

Scalability: BFS on Friendster

- Using more computing machines/cores can reduce execution time
- Tuning needed, e.g., for GraphLab, split large input file into number of chunks equal to the number of machines
The CPU utilization: computing node

- YARN and Hadoop exhibit obvious volatility
- The CPU utilization of graph-specific platforms is lower
Overhead: BFS on DotaLeague

- The percentage of overhead time is diverse across the platforms, algorithms, and graphs—tuning is only sometimes an option.
Key Findings From the Study of 6 Platforms

• Performance is function of (Dataset, Algorithm, Platform, Deployment)
  • Previous performance studies may lead to tunnel vision
  • Also looked at data structure, for CPU/GPU (submitted to ICPE’15)

• Platforms have their own drawbacks (crashes, long execution time, tuning, etc.)

• Some platforms can scale up reasonably with cluster size (horizontally) or number of cores (vertically)
  • Strong vs weak scaling still a challenge—workload scaling tricky

Graph processing: Possible to get better performance on GPUs than on CPUs

However, Algorithm and Dataset also determine performance
However, data format can also determine performance
Agenda

1. Everyone is a scientist!
2. Benchmarking: let’s show the numbers
3. Conclusion
BTWorld: Our Team

Alexandru Iosup
TU Delft
Big Data & Clouds
Res. management
Systems, Benchmarking

Dick Epema
TU Delft
Big Data & Clouds
Res. management
Systems

Mihai Capota
TU Delft
Big Data apps
Res. management
Benchmarking

Jan Hidders

Tim Hegeman
Vs of big data

- Volume – large scale of data
- Variety – different forms of data
- Velocity – timeliness of data
- Veracity – uncertainty of data

**Vicissitude** – dynamic combination of several big data Vs in processing systems that support the addition of new queries at run-time

**vicissitude** noun \[vɪˈsɪsɪtju\(d\)\]:
a favorable or unfavorable event or situation that occurs by chance; a fluctuation of state or condition
http://merriam-webster.com/dictionary/vicissitude
# Benchmarking MapReduce Systems

## Queries/Jobs, Workload Diversity, Data Set, Data Layout, Data Volume

<table>
<thead>
<tr>
<th>System</th>
<th>Queries/Jobs</th>
<th>Workload Diversity</th>
<th>Data Set</th>
<th>Data Layout</th>
<th>Data Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRBench [15]</td>
<td>business queries</td>
<td>high</td>
<td>TPC-H</td>
<td>relational data</td>
<td>3 GB</td>
</tr>
<tr>
<td>N-body Shop [14]</td>
<td>filter and correlate data</td>
<td>reduced</td>
<td>N-body simulations</td>
<td>relational data</td>
<td>50 TB</td>
</tr>
<tr>
<td>MadLINQ [7]</td>
<td>matrix algorithms</td>
<td>reduced</td>
<td>Netflix [29]</td>
<td>matrix</td>
<td>2 GB</td>
</tr>
<tr>
<td>GridMix [16],</td>
<td>artificial</td>
<td>reduced</td>
<td>random</td>
<td>binary/text</td>
<td>variable</td>
</tr>
<tr>
<td>PigMix [17]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>variable</td>
</tr>
<tr>
<td>HiBench [31],</td>
<td>text/web analysis</td>
<td>high</td>
<td>Wikipedia</td>
<td>binary/text/html</td>
<td>variable</td>
</tr>
<tr>
<td>PUMA [32]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>variable</td>
</tr>
<tr>
<td>WL Suites [12]</td>
<td>production traces</td>
<td>high</td>
<td></td>
<td></td>
<td>variable</td>
</tr>
<tr>
<td>BTWorld</td>
<td>P2P analysis</td>
<td>high</td>
<td>BitTorrent logs</td>
<td>relational data</td>
<td>14 TB</td>
</tr>
</tbody>
</table>

## Non-linear scaling

[Graph showing non-linear scaling with dataset size and query runtime]
Observing BitTorrent: Managing A Typical Global Distributed System
Observing BitTorrent: Managing A Typical Global Distributed System

Most used protocol on Internet, by upload volume [1]
One third (US) to half (EU) of residential upload
Over 100 million users [2]

Observing BitTorrent: Managing A Typical Global Distributed System

Monitor servers instead of users
The BTWorld Use Case (When Long-Term Traces Do Not Exist)

Collected Data

- Ongoing longitudinal study, since 2009
- Data-driven project: data first, ask questions later
- Over 15TB of data, 1 file/tracker/sample
- Timestamped, multi-record files
  - Hash: unique id for file
  - Tracker: unique id for tracker
  - Information per file: seeders, leechers
  - Structured and semi-structured data

Wojciechowski, Capota, Pouwelse, and Iosup. BTWorld: Towards observing the global BitTorrent file-sharing network. HPDC 2010
The BTWorld Use Case (When Long-Term Traces Do Not Exist)

**Analyst Questions**

- How does the number of peers evolve over time?
- How long are files available?
- Did the legal bans and tracker take-downs impact BT?
- How does the location of trackers evolve over time?
- Etc.

These questions need to be translated into queries.

The BTWorld Workflow

BTWorld records

ToT
  └── AT
  └── AS
  └── TKT-L

AH
  └── TKT-G
       └── TKNDH

TKS-L
  └── TKS-G
       └── TKH-G

TKH-L

Query Data path
The BTWorld Workflow

BTWorld records

ToT

AH

TKS-L

TKH-L

TKT-L

AT

AS

TKS-G

TKH-G

TKT-G

TKNHDH
The BTWorld Workload

SELECT tracker, timestamp,
    COUNT(hash) AS hashcount,
    SUM(seeders + leechers) AS sessions,
    AVG(liechers = 0 ?
      seeders : seeders / leechers)
    AS slratio
FROM logs
GROUP BY tracker, timestamp;
The BTWorld Workload

BTWorld records

SELECT timestamp, SUM(hashcount) AS swarms
FROM ToT
GROUP BY timestamp;
MapReduce-based Workflow for the BTWorld Use Case

Query Diversity

- Queries use different operators, stress different parts of system
- This kind of workflow is not modeled well by single-application benchmarks

Active Hashes (AH):

```
SELECT timestamp, COUNT(DISTINCT(hash))
FROM logs
GROUP BY timestamp;
```

Global Top K Trackers (TKT-G):

```
SELECT *
FROM logs
NATURAL JOIN (
    SELECT tracker
    FROM TKTL
    GROUP BY tracker
    ORDER BY MAX(sessions) DESC
    LIMIT k);
```

Hegeman, Ghit, Capotă, Hidders, Epema, Iosup. The BTWorld Use Case for Big Data Analytics: Description, MapReduce Logical Workflow, and Empirical Evaluation. IEEE BigData’13
Cluster configuration—DAS4

<table>
<thead>
<tr>
<th>Cluster size</th>
<th>24 nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map slots</td>
<td>92</td>
</tr>
<tr>
<td>Reduce slots</td>
<td>92</td>
</tr>
<tr>
<td>Memory per task</td>
<td>6 GiB</td>
</tr>
<tr>
<td>Total cluster memory</td>
<td>552 GiB</td>
</tr>
<tr>
<td>Scheduler</td>
<td>FIFO</td>
</tr>
<tr>
<td>HDFS replication</td>
<td>2</td>
</tr>
</tbody>
</table>

Representative for SMEs

Workload Does Not Scale Linearly

![Graph showing workload not scaling linearly with dataset size](image)

Results

• Long vs Short queries
  • Short relatively scale-free
  • Long do not scale super-linearly

• Possible to tune systems to avoid effects of vicissitude
Workflows = Data Vicissitude

Use Case: Monitoring Large-Scale Distributed Computing System with 160M users

Inter-query dependencies

Diverse queries
New queries during project

Hegeman, Ghit, Capotã, Hidders, Epema, Iosup. The BTWorld Use Case for Big Data Analytics: Description, MapReduce Logical Workflow, and Empirical Evaluation. IEEE BigData’13
### Beyond BTWorld

<table>
<thead>
<tr>
<th>BitTorrent</th>
<th>Trackers</th>
<th>Swarms</th>
<th>Hashes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finance</td>
<td>Stock markets</td>
<td>Stock listings</td>
<td>Stocks</td>
</tr>
<tr>
<td>Tourism</td>
<td>Travel agents</td>
<td>Vacation packages</td>
<td>Venues</td>
</tr>
</tbody>
</table>

- Monitoring large scale distributed computer systems
- Benchmarking
Agenda

1. Everyone is a scientist!
2. Benchmarking: let’s show the numbers
3. Conclusion
Elastic MapReduce: Our Team

Bogdan Ghit
TU Delft
Systems
Workloads

Dick Epema
TU Delft
Big Data & Clouds
Res. management
Systems

Alexandru Iosup
TU Delft
Big Data & Clouds
Res. management
Systems, Benchmarking
Dynamic Big Data Processing

Fawkes = Elastic MapReduce via Two-level scheduling architecture

Elastic MapReduce

MapReduce framework
- Distributed file system
- Execution engine
- Data locality constraints

Because workloads may be time-varying:
- Poor resource utilization
- Imbalanced service levels

Grow and shrink MapReduce
- High resource utilization
- Reconfiguration for balanced service levels
- Break data locality

No data locality

Core nodes

- Classical deployment
- Uniform data distribution
- No removal

INPUT/OUTPUT DATA

Transient nodes (TR)

- No local storage
- R/W from/to core nodes
- Instant removal

Performance?

Relaxed data locality

Core nodes

- Classical deployment
- Uniform data distribution
- No removal

Trans-core nodes (TC)

- Local storage, no input
- Only R from core nodes
- Delayed removal

Better performance?

FAWKES in a nutshell

1. Size of MapReduce cluster
   - Changes dynamically
   - Balanced by weight
     \[ s_i = \frac{W_i}{\sum W_j} \]

2. Updates dynamic weights when
   - New frameworks arrive
   - Framework states change

3. Shrinks and grows frameworks to
   - Allocate new frameworks (min. shares)
   - Give fair shares to existing ones

How to differentiate frameworks? (1/3)

By demand – 3 policies:
- Job Demand (JD)
- Data Demand (DD)
- Task Demand (TD)

VS.

How to differentiate frameworks? (2/3)

By usage – 3 policies:
- Processor Usage (PU)
- Disk Usage (DU)
- Resource Usage (RU)

By service – 3 policies:
- Job Slowdown (JS)
- Job Throughput (JT)
- Task Throughput (TT)

Experimental setup

DAS-4 multicluster system:
- 200 dual-quad-core compute nodes
- 24 GB memory per node
- 150 TB total storage
- 20 Gbps InfiniBand

Hadoop deployment:
- Hadoop-1.0 over InfiniBand
- 6 map + 2 reduce slots per node
- 128 MB block size

Overview of experiments:
- Most experiments on 20 nodes
- Up to 60 working nodes
- More than 3 months system time

## MapReduce applications

<table>
<thead>
<tr>
<th>Application</th>
<th>Type</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wordcount (WC)</td>
<td>CPU</td>
<td>200 GB</td>
<td>5.5 MB</td>
</tr>
<tr>
<td>Sort (ST)</td>
<td>Disk</td>
<td>200 GB</td>
<td>200 GB</td>
</tr>
<tr>
<td>PageRank (PR)</td>
<td>CPU</td>
<td>50 GB</td>
<td>1.5 MB</td>
</tr>
<tr>
<td>K-Means (KM)</td>
<td>Both</td>
<td>70 GB</td>
<td>72 GB</td>
</tr>
<tr>
<td>TrackerOverTime (TT)</td>
<td>CPU</td>
<td>100 GB</td>
<td>3.9 MB</td>
</tr>
<tr>
<td>ActiveHashes (AH)</td>
<td>Both</td>
<td>100 GB</td>
<td>90 KB</td>
</tr>
<tr>
<td>BTWorld (BT)</td>
<td>Both</td>
<td>100 GB</td>
<td>73 GB</td>
</tr>
</tbody>
</table>

### Synthetic benchmarks:
- HiBench suite
- Single applications
- Random datasets

### Real-world applications:
- BTWorld workflow
- 14 Pig queries
- BitTorrent monitoring data

Performance of dynamic MapReduce

- TR - good for compute-intensive workloads.
- TC - needed for disk-intensive workloads.

Dynamic MapReduce: < 25% overhead

10 core + 10xTR vs. 10 core + 10xTC

Performance of FAWKES

<table>
<thead>
<tr>
<th>Nodes</th>
<th>45</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frameworks</td>
<td>3</td>
</tr>
<tr>
<td>Min. shares</td>
<td>10</td>
</tr>
<tr>
<td>Datasets</td>
<td>300 GB</td>
</tr>
<tr>
<td>Jobs submitted</td>
<td>900</td>
</tr>
</tbody>
</table>

- **Avg. Slowdown**
  - **None** – Minimum shares
  - **EQ** – EQual shares
  - **TD** – Task Demand
  - **PU** – Processor Usage
  - **JS** – Job Slowdown

Up to 20% lower slowdown

TR nodes deliver good performance for CPU bound workloads.

Speedup when growing (2/2)

(Only) TC nodes deliver good performance for disk-bound workloads

Slowdown when shrinking

Data replicated
- 100 GB
- 50 GB

Fraction of nodes removed [%]

Job slowdown increases linearly with the amount of replicated data

Take-home message

1. Dynamic MapReduce relaxes data locality

2. FAWKES policies can reduce imbalance between frameworks

3. More aggressive policies?

Agenda

1. Everyone is a scientist!
2. Benchmarking: let’s show the numbers
3. Conclusion
The DAS-4 Infrastructure

- Used for research in systems for over a decade
  - 1,600 cores (quad cores)
  - 2.4 GHz CPUs, GPUs
  - 180 TB storage
  - 10 Gbps Infiniband
  - 1 Gbps Ethernet
- Koala grid scheduler

VU (148 CPUs)

UvA/MultimediaN (72)

TU Delft (64)

Astron (46)

Leiden (32)

SURFnet6

UvA (32)
Performance of Resizing using Static, Transient, and Core Nodes

Big Data processing: possible to get better performance using elastic data processing

we understand how for many scenarios (key is balanced allocations)

Sort + WordCount
(50 jobs, 1-50GB)

Conclusion Take-Home Message

- Big Data is necessary and grand challenge
- Big Data = Systems of Systems
  - Big data programming models have ecosystems
  - Stuck in stacks!
  - Many trade-offs, many problems

In this talk
- Predictability challenges: we need to understand workload (modeling) and performance (benchmarking)
- Early steps for benchmarking big data: graph processing, data processing workflows
- Elasticity challenges: dynamic MapReduce
Thank you for your attention! Questions? Suggestions? Observations?

More Info:
- http://www.st.ewi.tudelft.nl/~iosup/
- http://www.pds.ewi.tudelft.nl/
- http://research.spec.org

Alexandru Iosup

A.Iosup@tudelft.nl
(or google “iosup”)
Parallel and Distributed Systems Group
Delft University of Technology

Do not hesitate to contact me...