From Performance Profiling to Predictive Analytics while evaluating Hadoop performance using ALOJA

Nicolas Poggi, Senior Researcher

June 2015
0. About ALOJA
   - DEMO

1. From Performance Profiling to Predictive Analytics
   - Project evolution
   - PA uses and lines of research

2. A Case of Study on Hadoop Benchmark Behavior Modeling Using ALOJA-ML
   - Description of the Machine Learning process and current results

3. A characterization of cost-effectiveness of PaaS Hadoop in the Azure cloud
   - Performance evaluation and scalability of VMs in PaaS
ABOUT BSC’S AND ALOJA
BIG DATA BENCHMARKING PROJECT
Barcelona Supercomputing Center (BSC)

- 22 year history in Computer Architecture research
  - Based at the Technical University of Catalonia (UPC)
  - Long track record in chip Architecture & Parallelism
  - Active research staff with 1000+ publications
  - Large ongoing life science computational projects
  - Mare Nostrum Super Computer

- Prominent body of research activity around Hadoop since 2008
  - SLA-driven scheduling (Adaptive Scheduler), in memory caching, etc.

- Long-term relationship between BSC and Microsoft Research and Microsoft product teams

- Open model:
  - No patents, public IP, publications and open source main focus
  - 90+ publications, 4 Best paper awards

- ALOJA is the latest phase of the engagement
Initial motivation

- The Hadoop implements a complex distributed execution model
  - +100 interrelated config parameters
  - Requires manual iterative benchmarking and tuning

- Hadoop’s price/performance are affected by simple configurations
  - Performance gains SW >3x
  - and HW > 3x

- Commodity HW no longer low-end as in the early 2000’s
  - Hadoop performs poorly on scale-up, or low power

- New Cloud services for Hadoop
  - IaaS and PaaS
  - Direct vs. remote attached volumes

- Spread Hadoop ecosystem
  - Dominated by vendors
  - Lack of verifiable benchmarks
Current scenario and problematic

What is the most cost-effective configuration for my needs?
- Multidimensional problem

And where is my system configuration positioned on each of these axes?
Project ALOJA

Open initiative to explore and produce a systematic study of Hadoop efficiency on different SW and HW
- Both cost and performance
- Including commodity, high-end, low-power, and cloud

Results from a growing need of the community to understand job execution details

Explore different configuration deployment options and their tradeoffs
- Both software and hardware
- Cloud services and on-premise

Seeks to provide knowledge, tools, and an online service
- To make better informed decisions
- Reduce the TCO for their Big Data infrastructures
- Guide the future development and deployment of Big Data clusters and applications
ALOJA Platform components and status

Benchmarking, Repository, and Analytics tools for Big Data

Composed of open-source
- Benchmarking, provisioning and orchestration tools,
- high-level system performance metric collection,
- low-level Hadoop instrumentation based on BSC Tools
- and Web based data analytics tools
  - And recommendations

Online Big Data Benchmark repository of:
- 42,000+ runs (from HiBench), some BigBench and TCP-H
- Sharable, comparable, repeatable, verifiable executions

Abstracting and leveraging tools for BD benchmarking
- Not reinventing the wheel but,
- most current BD tools designed for production, not for benchmarking
- leverages current compatible tools and projects

Dev VM toolset and sandbox
- via Vagrant
Components Big Data Benchmarking

**ALOJA-DEPLOY** Composed of scripts to:
- Automatically create, stop, delete clusters in the cloud
  - From a simple and abstracted node and cluster definition files
  - Both for Linux and Windows
  - IaaS and PaaS (HDInsight)
  - Abstracted to support multiple providers
- Provision and configuration of base software to servers
  - Both for cloud based as on premise
  - Composed of portable configuration management scripts
  - Designed for benchmarking needs
- Orchestrate benchmark executions
  - Prioritized job queues
  - Results gathering and packaging

**ALOJA-BENCH**
- Multi-benchmark support
- Flexible performance counter options
- Dynamic SW and HW configurations
Workflow in ALOJA

Cluster(s) definition
- VM sizes
- # nodes
- OS, disks
- Capabilities

Execution plan
- Start cluster
- Exec Benchmarks
- Gather results
- Cleanup

Import data
- Convert perf metric
- Parse logs
- Import into DB

Evaluate data
- Data views in Vagrant VM
- Or http://hadoop.bsc.es

Historic Repo

PA and KD
- Predictive Analytics
- Knowledge Discovery
ALOJA-WEB Online Repository

- Entry point for explore the results collected from the executions
  - Index of executions
    - Quick glance of executions
    - Searchable, Sortable
  - Execution details
    - Performance charts and histograms
    - Hadoop counters
    - Jobs and task details

- Data management of benchmark executions
  - Data importing from different clusters
  - Execution validation
  - Data management and backup

- Cluster definitions
  - Cluster capabilities (resources)
  - Cluster costs

- Sharing results
  - Download executions
  - Add external executions

- Documentation and References
  - Papers, links, and feature documentation

Available at: [http://hadoop.bsc.es](http://hadoop.bsc.es)
Features and Benchmark evaluations in ALOJA-WEB

**Benchmark Repository**
- Browse executions
- Hadoop Job counters
- PaaS exec details

**Config Evaluations**
- Best execution
- Config improvement
- Parameter evaluation

**Cost/Perf Evaluation**
- Scalability of VMs
- Evaluation of execs
- Evaluation of clusters
- Evaluation of HW configs

**Performance Details**
- Performance Charts
- Performance metrics details
- DBSCAN

**Prediction Tools**
- Modeling data
- Predict configurations
- Config tree
- Anomaly detection
- ...
Entry point for exploring the results collected from the executions,
– Provides insights on the obtained results through continuously evolving data views.

Online DEMO at: http://hadoop.bsc.es
PROJECT EVOLUTION AND
LESSONS LEARNED ALONG THE WAY
Reasons for change in ALOJA

Part of the change/evolution in the project due to focus shift

- To available resources (Cloud)
- Market changes: On-prem vs. Cloud
  - IaaS vs. PaaS
    » Pay-as-you-Go, Pay-what-you-process
  - Challenges
    » From local to remote (network) disks
    » Over 32 types of VM in Microsoft Azure

- Increasing number of benchmarks
  - Needed to compare (and group together) benches of different jobs and systems
  - Deal with noise (outliers) and failed executions
  - Need automation
    » Predictive Analytics and KD

- Expanding the scope / search space
  - From apps and framework
  - Including clusters/systems
  - To comparing providers (datacenters)
Techniques for obtaining Cost/Performance Insights

Profiling
- Low-level
- High Accuracy
- Manual Analysis

Benchmarking
- Iterate configs
- HW and SW
- Real executions
- Log parsing and data sanitization

Aggregation
- Summarize large number of results
- By criteria
- Filter noise
- Fast processing

Predictive Analytics
- Automated modeling
- Estimations
- Virtual executions
- Automated KD

Evaluation of:
- Big Data Apps
- Frameworks
- Systems / Clusters
- Cloud Providers
Initial approach: Low-level profiling

Profiling Hadoop with BSC’s HPC tools

- Preliminary work, relying on over 20 years HPC experience and tools
- Developed the Hadoop Instrumentation Toolkit
  - with custom hooks to capture events
  - Added a network sniffer

HDP processes and communication

CPU

Memory

Page Faults
Overview of HAT and HPC tools

Hadoop Analysis Toolkit and BSC tools

Hadoop + Performance Monitoring Tools

- Extrae
- Hadoop Tools
  - Java
  - Generate Event
- JNI – Java (native)
- extree_wrapper.so
- libextrae.so
- libpcap.so

- Hadoop Events
  - System
  - Networking

- Extrae traces
  - *.mpit

- Paraver Traces
  - *.prv

- Paraver (Visualization and Analysis)

- DIMEMAS (Simulation)

- Paraver Config
  - *.cfg
Hadoop in PARAVER

**Different Hadoop Phases**
- Map
- Reduce
Sort + combine

Detailed work done by Hadoop
- Sort / Combine

Flush
SortAndSpill
Sort
Combine
CreateSpillIndexFile
Network communications

Communications between processes...

... or between nodes
Network: low-level

Low level details
- TCP 3-way handshake

Data analysis tool:
Low-level profiling

Pros

- Understanding of Hadoop internals
- Useful to improve and debug Hadoop framework
- Detailed and accurate view of executions
- Improve low-level system components, drivers, accelerators

Cons

- Non-deterministic nature of Hadoop
- Not suitable for finding best configurations
- Not suitable to test different systems
  - And Big Data platforms (re implement)
- Virtualized environments introduces challenges for low-level tools
- On PaaS you might not have admin user (root)
Extensive benchmarking effort iterating SW and HW config

- Different cluster architectures
  - On-prem and Low-power

Objectives:

- Understand Hadoop executions
- Capture results for analysis/research

Led to the online repository

- You can compare, side by side, all execution parameters:
  - CPU, Memory, Network, Disk, Hadoop parameters….
Benchmarking use case: IB vs ETH

- InfiniBand (IPoIP) performance increase in
  - Terasort (100G) and InfiniBand

- Cluster minerva-100
  - 12 real cores, 64MB RAM, Ubuntu Server 14.04
  - 8-datanodes + 1-headnode
  - 5 SATA drives
  - 2 SSD drives (provided by Sandisk)

- Evaluations (Speedups and Cost-effectiveness)
  - Disk combinations
    - 1-5 SATA drives as JBOD (HDD, HD2, HD3, HD4, HD5)
    - 1-2 SSD drives as JBOD (SSD, SS2)
    - 5 SATA drives JBOD and Hadoop /tmp to 1 SSD (HS5)
  - Network bandwidth to disk configs
    - InfiniBand (IPoIB) vs Ethernet (1-GbE)
  - Hadoop max slots (mappers) speedup by Network and Disk combination

All data online and accessible at [http://hadoop.bsc.es/](http://hadoop.bsc.es/)
Hadoop Execution phases: IB vs ETH for Terasort and DFSIOE

Terasort

IB Slightly faster for Terasort

DFSIOE Read

IB Significantly faster than ETH for DFSIOE

URL Terasort
http://hadoop.bsc.es/perfcharts?execs%5B%5D=84766&execs%5B%5D=84746&metric=Memory&hosts=Slaves&aggr=AVG&detail=1

URL DFSIOE Read
http://hadoop.bsc.es/perfcharts?benchmarks_length=1&execs%5B%5D=85088&execs%5B%5D=85776
Network MB/s: IB vs ETH for Terasort and DFSIOE

Terasort

IB not fully utilized in Terasort
22 MB/s max

DFSIOE Read

IB reaches 100 MB/s for DFSIOE

URL Terasort
http://hadoop.bsc.es/perfcharts?execs%5B%5D=84766&execs%5B%5D=84746&metric=Memory&hosts=Slaves&aggr=AVG&detail=1

URL DFSIOE Read
http://hadoop.bsc.es/perfcharts?benchmarks_length=1&execs%5B%5D=85088&execs%5B%5D=85776
Disk IOPS: IB vs ETH for Terasort and DFSIOE

**Terasort**

With IB, almost 10,000 IOPS for DFSIOE

**DFSIOE**

Read

Slightly higher IOPS for Terasort
Benchmarking problems

As number of results grew, manually analyzing low-lever results was no longer feasible
  – Either for the HPC tools
  – Or manually revising them

Cons
  – It became a Big Data problem in it self
  – Cloud introduces more uncertainty
  – Manual sampling was required
  – Search space kept growing

For this we relied in aggregation and summarization of data
  – Grouping results from different executions
  – Lost accuracy, but gained in processing time and abstraction
Aggregation and summaries

Once data is imported into a DB data aggregation and summarizations becomes simple

As data is immutable, aggregation only have to be done once

This produces metadata
  - That is small in size
  - Can be queries online

Slightly different executions can be grouped together

Works well with public cloud executions

Noise get filtered out

Examples in the Web app at the
  - Config Evaluations
  - Cost/Performance menus
### Best terasort configuration

<table>
<thead>
<tr>
<th>Run details</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark</td>
<td>terasort</td>
</tr>
<tr>
<td>Execution time</td>
<td>327.00s</td>
</tr>
<tr>
<td>Running cost $</td>
<td>0.91$</td>
</tr>
<tr>
<td>Net</td>
<td>Ethernet</td>
</tr>
<tr>
<td>Disk</td>
<td>SSD drive</td>
</tr>
<tr>
<td>Number of hadoop mappers</td>
<td>10</td>
</tr>
<tr>
<td>IO Sort File Factor</td>
<td>10</td>
</tr>
<tr>
<td>Number of replicas</td>
<td>1</td>
</tr>
<tr>
<td>IO File Buffer</td>
<td>65536KB</td>
</tr>
<tr>
<td>Compression algorithm</td>
<td>None</td>
</tr>
<tr>
<td>Block size</td>
<td>64MB</td>
</tr>
<tr>
<td>Cluster</td>
<td>r1-33</td>
</tr>
<tr>
<td>PARAVER</td>
<td>PARAVER</td>
</tr>
</tbody>
</table>

### Best wordcount configuration

<table>
<thead>
<tr>
<th>Run details</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark</td>
<td>wordcount</td>
</tr>
<tr>
<td>Execution time</td>
<td>633.00s</td>
</tr>
<tr>
<td>Running cost $</td>
<td>1.76$</td>
</tr>
<tr>
<td>Net</td>
<td>Ethernet</td>
</tr>
<tr>
<td>Disk</td>
<td>2 Remote volumes(s)</td>
</tr>
<tr>
<td>Number of hadoop mappers</td>
<td>8</td>
</tr>
<tr>
<td>IO Sort File Factor</td>
<td>10</td>
</tr>
<tr>
<td>Number of replicas</td>
<td>1</td>
</tr>
<tr>
<td>IO File Buffer</td>
<td>65536KB</td>
</tr>
<tr>
<td>Compression algorithm</td>
<td>None</td>
</tr>
<tr>
<td>Block size</td>
<td>64MB</td>
</tr>
<tr>
<td>Cluster</td>
<td>r1-30</td>
</tr>
<tr>
<td>PARAVER</td>
<td>PARAVER</td>
</tr>
</tbody>
</table>
Impact of SW configurations in Speedup

Number of mappers

<table>
<thead>
<tr>
<th>Number of Mappers</th>
<th>No comp.</th>
<th>ZLIB</th>
<th>BZIP2</th>
<th>snappy</th>
</tr>
</thead>
<tbody>
<tr>
<td>4m</td>
<td>0.565</td>
<td>0.816</td>
<td>0.816</td>
<td>0.816</td>
</tr>
<tr>
<td>6m</td>
<td>0.618</td>
<td>0.816</td>
<td>0.816</td>
<td>0.816</td>
</tr>
<tr>
<td>8m</td>
<td>0.816</td>
<td>0.96</td>
<td>1.013</td>
<td>1.04</td>
</tr>
<tr>
<td>10m</td>
<td>0.805</td>
<td>0.994</td>
<td>1.08</td>
<td>1.08</td>
</tr>
</tbody>
</table>

Speedup (higher is better)

Compression algorithm

<table>
<thead>
<tr>
<th>Compression Algorithm</th>
<th>No comp.</th>
<th>ZLIB</th>
<th>BZIP2</th>
<th>snappy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.89</td>
<td>0.903</td>
<td>0.96</td>
<td>0.895</td>
</tr>
<tr>
<td></td>
<td>0.89</td>
<td>1.263</td>
<td>1.266</td>
<td>1.311</td>
</tr>
</tbody>
</table>

Speedup (higher is better)
Impact of HW configurations in Speedup

Disks and Network

Cloud remote volumes

Speedup (higher is better)
Parameter evaluation example (lower is better)

Increasing RAM improvement:
Data sizes for 42K executions

- Profile traces: ~57 TB
- Perf counters: 1.2 TB
- Hadoop logs: 11 GB
- Metadata: 15 MB
- PA model: ~0.4 MB

* Estimated size, profiles only ran on selected execs
** Only includes exec config and exec time
*** Model for predicting exec times and compressed on disk
Encompasses statistical and Machine Learning (ML) techniques
- To make predictions of unknown events
  - Forecast and foresight
  - From historical data

Implemented them as an extension to the platform
- Mainly R code, that can be called from a Web frontend
Welcome to the ALOJA project,

ALOJA is an initiative of the BSC-MSR research centre in Barcelona to explore Hadoop's performance. You can find introductory Slides and Papers in the ALOJA Reference menu.

This site is under constant development and it is in the process of being documented. For site updates, browse the site, the code, and send inquiries, feature requests or bug reports to: hadoop@bsc.es

If you’re curious about the name of the project, visit ALOJA at

Site’s content:

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Video DEMO of ALOJA</strong></td>
<td>Brief video showcasing ALOJA’s main online features (a bit outdated)</td>
</tr>
</tbody>
</table>

**Benchmark Executions**

This sections presents the benchmark execution repository. It features more than 30,000 executions and counting. This tool allows you to browse, filter, search, and select distinct executions to compare and analyse its execution details.

**Hadoop Job Counters**

The Hadoop Job Counters sections allows to browse the counters output at each of the Hadoop executions, filter them, and order by a specific counter the selected runs (or all). The section presents the summary of all the Job execution counters, Map and Reduce specific counters, and the I/O subsystem counters. It also features the details by task: to understand the running time of each Map or Reduce process.
The ALOJA Predictive Analytics tool-set

1. **Modeling and Prediction**
   - From ALOJA dataset →
   - Find a model for
   - \( \langle \text{Workld,Conf} \sim \text{Exe.Time} \rangle \)

2. **Configuration recommendation**
   - Rank (un)seen confs. for a benchmark expected Exe.Time

3. **Anomaly detection (outliers)**
   - Statistic + Model-based detection of anomalous executions

4. **Behavior observation and statistic information**
   - Aggregate variables around the ones we want to observe
   - Show frequency, percentiles and other useful information from ALOJA datasets
Rank and Recommend Configurations

- Predict a range of configurations, previously seen or unseen
- Order them by predicted execution time and rank configurations
- Compare also predicted execution times vs. observed execution times, if there are any.
Perf profiling vs. PA Benchmarking in ALOJA

From perf profiling
- Detailed
  - Low-level (HPC-tools)
  - Debug info
  - Specific
- Application centric (Hadoop)
  - Improve application
  - Hadoop configuration
- Constrained approach
- Big Data
- Exposes internal components
- High accuracy
- Susceptible to noise and variations
- Manual analysis

To Predictive Analytics
- Summaries (metadata)
  - High-level
  - Insights
  - General / Tendencies
- System centric (HW for BD)
  - Improve systems
  - Cluster topology
- Unbounded search space
- Metadata
- Black-box approach
- Estimations and tendencies
- Some noise and failures are acceptable
- Automated KD
# Summary of techniques

<table>
<thead>
<tr>
<th></th>
<th>Profiling</th>
<th>Benchmarking / Importing</th>
<th>Aggregation</th>
<th>Predictive A.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Datasizes</strong></td>
<td>Very large</td>
<td>Large</td>
<td>Small</td>
<td>Very small</td>
</tr>
<tr>
<td><strong>Processing</strong></td>
<td>Medium</td>
<td>Medium</td>
<td>Fast</td>
<td>Slow</td>
</tr>
<tr>
<td></td>
<td>• Large data</td>
<td>• Medium data</td>
<td>• SQL based (group by)</td>
<td>• Large RAM</td>
</tr>
<tr>
<td></td>
<td>• Match timestamps</td>
<td>• Uncompress</td>
<td>• Data does not change</td>
<td>• CPU time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Convert formats</td>
<td></td>
<td>• Parallelization problems</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Import formats</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Main focus</strong></td>
<td>App</td>
<td>Framework</td>
<td>Comparing systems and HW confs</td>
<td>Cloud providers Datacenters</td>
</tr>
<tr>
<td></td>
<td>• Hadoop phases (App)</td>
<td>• Hadoop parameters</td>
<td></td>
<td>• + previous</td>
</tr>
</tbody>
</table>
Summary and conclusions

- Described the evolution of the project
  - Highlighting the technical and market motivations

- Data reduction
  - 99% of the data in low-level details
  - Faster information and insights from meta-data and summaries
  - Simplify management
  - Share results
  - DEV VM and toolbox

- PA is our current frontier
  - to save in execution times and costs
  - Relies on metadata and summaries
  - Knowledge Discovery

Next steps
- Predictions everywhere
- Guided executions
- Low-level dynamic Instrumentation for Hv2
Online repository and tools available at:
- [http://hadoop.bsc.es](http://hadoop.bsc.es)


- Project description on:
  - "ALOJA: a Systematic Study of Hadoop Deployment Variables to Enable Automated Characterization of Cost-Effectiveness"

- Upcoming:
  - SIGKDD15
    - ALOJA-ML: Predictive analytics tools for benchmarking on Hadoop deployments
Extending and collaborating in ALOJA

1. Install prerequisites
   - vagrant

2. git clone https://github.com/Aloja/aloja.git

3. cd aloja

4. vagrant up

5. Open your browser at: http://localhost:8080
Thanks!

Q&A

Contact: hadoop@bsc.es