INTEGRATING LHCB OFFLINE ACTIVITIES ON SUPERCOMPUTERS:
State of Practice

CHEP 2023

May 9th 2023

Alexandre F. Boyer
alexandre.boyer@cern.ch

European Organization for Nuclear Research
Meyrin, Switzerland
The LHC produces an increasing amount of data over time (x10 with the HL-LHC)

- The WLCG resources will be limited to process the data - and simulated data - coming from the next LHC runs in real time.
- Experiments are constantly looking for new opportunistic resources to expand their computing capacity: clouds, supercomputers...

Supercomputers provide massive computing power

- Funding agencies encourage us to exploit them but they are not easily accessible.
- Running LHCb software on such infrastructures requires a significant amount of work.

Would supercomputers be able to manage the LHCb offline activities?
LHCb Offline Activities: Computing Resource Requirements
Review of LHCb offline activities on WLCG in 2022

Highlights

- 92.4% of the capacity is dedicated to MC simulation.
- The remaining 7.6% represents the other activities:
  - Analysis
  - Reconstruction
  - ...
- The more real data we get, the more MC simulations have to be processed: this is not linear.

We are going to focus on MC simulation in this presentation.
Simulating the collisions with Gauss

Goal

• Better understand the experimental conditions and performance of the experiment.

Properties

• (Almost) no input data.
• CPU-intensive task.
• 1 logical core and 2Gb of RAM is needed.

Gauss is "easy" to export on remote computing resources.
More MC simulations: Considered strategies

Developing more efficient and flexible applications

- Gauss-on-Gaussino: multi-threaded version of Gauss (not validated in production yet)
- Gauss on ARM (not validated in production yet).
- Other approaches: simulating less detector (RICHLess), simulating less event (ReDecay) ...

Use (efficiently) more computing resources

- A few ongoing collaborations with supercomputer centers:
  - Piz Daint in Switzerland
  - Marconi-100 in Italy
  - Santos Dumont in Brazil
  - Mare Nostrum IV in Spain
  - ...
  - They provide massive computing power but are very restrictive.

This is the approach that we are going to describe in the following sections.
LHCB & SUPERCOMPUTERS
Submitting jobs in WLCG resources: DIRAC

Brief presentation

• Middleware used to submit jobs to remote, shared and heterogeneous computing resources.

• Open source and generic tool developed by LHCb and used in many different contexts: EGI, Belle II, CTA...

• Further details this afternoon, in a presentation dedicated to DIRAC developments: https://indico.jlab.org/event/459/contributions/11468/
DIRAC Workload Management System & WLCG resources

1. Jobs arrive in WMS queue
2. Pilot-Factory generates & submits pilots
3. Pilots wait in the Site queues
4. Pilots evaluate their environment
5. Pilots fetch jobs
6. Jobs run

DIRAC Services
1. Jobs arrive in WMS queue
2. Pilot-Factory generates & submits pilots
3. Pilots wait in the Site queues
4. Pilots evaluate their environment
5. Pilots fetch jobs
6. Jobs run

LRMS Queues

Grid Site

Waiting Jobs

Pilot-Factory (SiteDirector)
DIRAC Workload Management System & Supercomputers?
Challenges

• Software has to be flexible. Supercomputers may include non-x86 CPUs and accelerators.

• The DIRAC Workload Management System (and operators) needs to provide the software requirements. We will focus on that aspect in the following sections.

⇒ Supercomputers are very heterogeneous: it is impossible to produce a generic and unique solution that would work for all of them.

⇒ Goal: exploiting x86 CPU resources by building small software blocks that can be added to each other to generate a tailored solution.
TECHNICAL SOLUTIONS
# Solutions based on features

## Features

- 1 feature directly affects the chosen paradigm:
  + Do the worker nodes have an external connectivity? Yes (or only via the head node), no.

- Other features generate some technical adjustments around the chosen paradigm:
  + Is CVMFS mounted on the worker nodes? yes, no.
  + Is the Batch System accessible from outside? yes, no.
  + What type(s) of allocations can we request? Single core, multi-core, multi-node.
Software solutions: Complete access to the supercomputer & single-core allocations

Similar to a WLCG grid site

**PizDaint**

- Uncommon for a supercomputer.
- Require a close collaboration with the system administrator of the supercomputer.
Software solutions: Complete access to the supercomputer & multi-core allocations

Supercomputers tend to favor multi-core allocations...

Node partitioning

- One pilot-job for many cores on 1 node.
- Repeats the following operations until all the cores are occupied: fetch a job from the DIRAC services and execute it on the node.

**SantosDumont**  **DIRAC**
... And even multi-node allocations.

**Sub-Pilots**

**SantosDumont**  **DIRAC**

- Use of `srun` to install 1 pilot-job per node in parallel.
- The pilot-jobs share the same identifier, status and logs.
- Possibilities to request elastic allocations (e.g. between 1 and 5 nodes).
By default, supercomputers do not provide access to CVMFS.

**CVMFS-exec**

- Client installed on the shared file system of the supercomputer.
- Mounts CVMFS as an unprivileged user.
- Requires actions from a DIRAC operator.
Some supercomputers can only be accessed via a VPN (No CE, no direct SSH access).

**Pilot factory installed on a head node** **DIRAC**

- Pilot-Jobs are directly submitted from the supercomputer.
- Requires actions from both a system administrator of the supercomputer (getting the certificate, authorizing cron jobs), and a DIRAC operator (installing the Pilot factory).
Software solutions: No external connectivity...

Some supercomputers do not allow jobs to access external services.

PushJobAgent
- MareNostrum
- DIRAC

- Works as a Pilot-Job that would be executed outside of the supercomputer.
- Fetches jobs, manages their input and output data, and solely submit the application (Gauss) to the supercomputer.
- Requires a direct access to the Batch System.
In this context, we cannot leverage CVMFS-exec.

Subset-CVMFS-Builder
MareNostrum

- Generic solution to create and deploy subsets of CVMFS.
- Takes the form of a Python package and a continuous integration pipeline.
- Example: extracting Gauss dependencies (a few GB) in 2h30: [https://gitlab.cern.ch/lhcb-dirac/subcvmfs-builder-pipeline](https://gitlab.cern.ch/lhcb-dirac/subcvmfs-builder-pipeline)
RESULTS
Available supercomputers process 300 jobs/hour on average vs WLCG grid resources process 14,000 jobs/hour on average.
CONCLUSION
Generic solutions exist and can be adapted to other Supercomputers: we are ready to scale up.

Main contribution

• Methods and software blocks to integrate MC simulations tasks on supercomputers (constrained environments).

• May benefit to VOs using DIRAC, LHC experiments, and more broadly, to any community working with distributed, shared and heterogeneous computing resources.
Questions? Comments?
BACKUP
### SDumont, LNCC: Development

**Features**

- Ranked 462<sup>th</sup> of the Top500 (1.85 PFlop/s - Nov. 2022)
- Opportunistic resources.
- 24 cores and 64Gb of RAM per node.

**Implementing the following solutions**

- Sub-pilots and node partitioning.
- Test: Pilot factory installed on one of the head node.
Results

- A Gauss job on every logical cores available per allocation.
- Elastic allocation: we request a time interval and a variable number of nodes.

Problems & Considered approaches

- Inaccurate CPU work estimates: a lot of our jobs run out of time.
Features

- Ranked 88th of the Top500 (6,470 PFlop/s - Nov. 2022)
- 4-month allocations of CPU hours.
- 48 cores and 96Gb of RAM per node.

Implementing the following solutions

- PushJobAgent to push jobs.
- Subset-CVMFS-Build to generate and deploy up-to-date subsets of CVMFS with Gauss dependencies.
Mare Nostrum, BSC: Status

Résultat

- One Gauss job per single-core allocation.
- 300 jobs in parallel.
- Using 500Kh/750Kh allocated (4 months).
- The subset of CVMFS is regularly updated: no major issue so far.

Problems & Considered approaches

- **PushJobAgent** is simple but consumes a lot of memory: cannot scale.
- Reducing the memory consumption implies important changes within DIRAC.