Optimization of opportunistic utilization of the ATLAS High-Level Trigger farm for LHC Run 3

Ivan Glushkov (UTA), Chris Lee (Stoney Brook U), Alessandro Di Girolamo (CERN), Rodney Walker (Munich LMU), Carlo Alberto Gottardo (CERN)

on behalf of the ATLAS Computing Activity
Opportunistic usage of the ATLAS Trigger and Data Acquisition High Level Trigger (HLT) computing farm for offline data processing workflows

Hypervisors
- Total: 119744 Cores
- 1216 x AS-2124BT-HNTR
  - 2 x AMD EPYC 7302 16-core
  - 64 HT cores
  - 125 GB RAM
- 440 x S2600TPR
  - Intel Xeon E5-2660 v4 @ 2.00GHz
  - 56 HT cores
  - 62 GB RAM
- 360 x S2600KPR
  - Intel Xeon E5-2650 v4 @ 2.20GHz
  - 48 HT cores
  - 62 GB RAM

VMs
- Total: 111680 Slots
- x1216
  - 60 slots
  - 1.63 GB/slot
- x440
  - 52 slots
  - 0.94 GB/slot
- x360
  - 44 slots
  - 1.11 GB/slot

Less than the typically required 2 GB RAM/slot
Isolation of offline environment from detector control
- Network: VLAN on data link layer from P1* to CERN
- WN: VMs

VMs
- Switch / VM Creation is triggered by the TDAQ** shifter
- Puppet triggers VM creation. It is configured to run at random times within an hour to avoid network overload
- Libvirt VMs from CernVM image + config disk + ephemeral disk
- Config disk - prepare CVMFS, ATLAS env, join batch

Number of VM slots available for Sim@P1 Site

Number of slots occupied by running GRID jobs

* P1 - Abbreviation of “Point 1” - the physical location of the ATLAS experiment on the LHC accelerator ring
** TDAQ - Trigger Data Acquisition System
Sim@P1: Contribution to ATLAS Distributed Computing

- **Optimal for simulation workflow**
  - Limited RAM
  - Limited Network
- **Urgent Run2 Reprocessing was run in the beginning of 2022**
  - Not optimal

---

**Sim@P1 Resource Utilization**

**Sim@P1 Running Slots**

**ATLAS Simulated Events**
Sim@P1 - Open questions

- When should we switch to Sim@P1?
  - What is the minimum LHC inter-fill time (IFT) from which ATLAS can benefit?
- What is the typical IFT to be expected?
- What should be optimized and where in order to best utilize the resources?
  - Workflow Management (Panda)
  - Pilot submission (Harvester)
  - Local Batch (HTCondor)
  - WN configuration
- What metrics shows that we are using Sim@P1 better?
  - Speed of releasing resources to Sim@P1
  - Speed of filling
  - Quality of filling
  - Job parameters
  - Number of events produced
- Let’s test!
Test Setup

- **Data**
  - Large FastSim LHC Run2 standard proton-proton collision dataset

- **Schedule**
  - 06.09.2022 - 20.09.2022
  - During LHC technical stop
  - Period with low simulation load on the system

- **Hardware**
  - “Only” 98.1k CPUs were available for the test (the rest were used by HLT)
No need to improve it if we cannot fill the resources fast enough with offline workflows, right?

1. Submission rate is lower than resource release rate

2. Long delay before filling with jobs

Solutions:
1. 
   a. Increase the amount of queued workers in Harvester
   b. Increase the corecount on the jobs submitted
2. 
   a. Use dedicated tasks
      i. Reduces the job brokerage time
Resource utilization rate is limited by resource availability rate

Average filling rate: ~1500 cores/min
Average time to fill all slots: ~80 min

Resource increase expected if it was limited only by submission infrastructure

Irreducible delay due to job initialization: ~10 min
TDAQ: Forcing Puppet

Before

1.5k cores/min
or
77 mins for 117k cores

After

12k cores/min
Or
9.78 mins for 117k cores

x8 faster

Tried to leave this setting “on” in production, BUT:

○ This leaves the TDAQ monitor system in bad condition and it took a while to recover. We rely on monitoring to ensure system functionality.

○ We cannot assure that a reboot of the other, critical netboot nodes (including ROS, Felix, SWROD) would work properly during the massive puppet run on the TPUs.
Resource Filling Efficiency

Less cores, better filling
FastSim Job Properties

Higher core count:

- Shorter jobs
- Lower CPU/WALL efficiency
  Better for shorter IFT

Lower core count:

- Longer jobs
- Higher CPU/WALL efficiency
  Better for longer IFT

Older CPUs (i.e. Intel):

- Up to 31% of jobs
- Up to 70% longer

Sim@P1, Ivan Glushkov, CHEP2023
Resource Filling Shape

- **Distribution Shape**
  - Matches job walltime minus the initialization phase

- **Due to**
  - Too similar jobs (i.e. job lengths) all initializing at the same time
  - Non-zero job initialization time
  - Two different hardware types

- **Solution**
  - Mix of different jobs
FullSim* vs FastSim* (indicative!)

- FullSim measurements are ONLY indicative due to:
  - Production job mix
  - Low statistics
- No dedicated tests with FullSim:
  - Looking for the shortest jobs
  - FastSim is expected to be the standard simulation production for Run3

* FullSim - Full ATLAS detector simulation
** FastSim - Fast ATLAS detector simulation combining parametrization, machine learning, and modelling of physics object to minimize CPU utilization

FullSim vs FastSim jobs: On average: x2 longer

- FullSim Sim@P1 Preferred corecount configuration:
  - Consistent with FastSim configuration
  - Higher core count for short IFT
  - Lower core count for long IFT
Sim@P1: Proposed Configuration

- **Proposed setup**
  - MC: Dedicated low priority FastSim samples
  - WFMS: Constantly queued workers
  - Batch queue: 44 + 8 cores
  - TDAQ: Forced Puppet run at TDAQ > Sim@P1 change

- **Advantages**:
  - Perfect ramp-up
  - Faster ramp-up
  - Many 44-core jobs finishing in case of short IFT
  - Running 8-core jobs in case of longer IFT

- **Results**:
  - FastSim: 1h - 400k events simulated

- **Work**:
  - MC production
  - TDAQ
    - Not clear how much work would that imply
    - Priority is datataking!
    - Manpower
Sim@P1: Operation Scenarios

- **Currently**
  - LHC is not yet in stable operation mode - no interfills to talk about

- **Parameters**
  - A = 5 mins
  - B
    - 77 mins - currently
    - 9.8 mins - if Puppet is enforced
  - C
    - 10 mins - Job initialization, Input data download.
  - D - minimal values @44 cores
    - 43 mins (FastSim)
    - 85 mins (FullSim)

- **Scenarios**
  - Best:
    - Enforced puppet, dedicated FastFim
    - E = A+D = 5 + 43 >= 48 mins
  - "Worst":
    - No puppet changes, FullSim, random production
    - E = A+D = 5 + 85 + ? >= 90 min
  - Current:
    - == Worst
      - All WFMS related settings were optimized
      - Further improvement can be expected when we have low priority FastSim available at mass

A - Switching from TDAQ to Sim@P1. 5 mins.
B - Ramp-Up. Configuration of VMs
C - Job initialization time
D - Job length
E - Total minimal IFT needed to get any jobs finished
• **We can do (almost) anything**
  ○ Without significant changes we can get useful events from IFT as short as 2 hours
  ○ Potentially we can even go down to 1 hour

• **The parameter phasespace is huge**
  ○ How long is the IFT going to be?
    ■ During data taking
      ● 1-2 hours - only FastSim can succeed.
      ● 3 hours and more - any simulation is fine
    ■ Technical Stops / Machine Development
      ● Just another GRID site running simulation
  ○ How much do we want to tailor the jobs?
    ■ Many convoluted parameters - no silver bullet to efficient resource utilization
    ■ Requires person-hours

• **More tests can be done to optimize even better**
  ○ ...but we have a solid idea already
  ○ The first-approximation parameters were adjusted - good enough for the time being

• **More experience during data taking would be welcome**
  ○ Switch to Sim@P1 every time possible, please.
  ○ Data taking is paramount! Switching to Sim@P1 is up to Run Coordination and TDAQ Coordination