Running a mixture of ATLAS jobs with widely ranging resource requirements at IT4Innovation

M. Svatoš, J. Chudoba, P. Vokáč

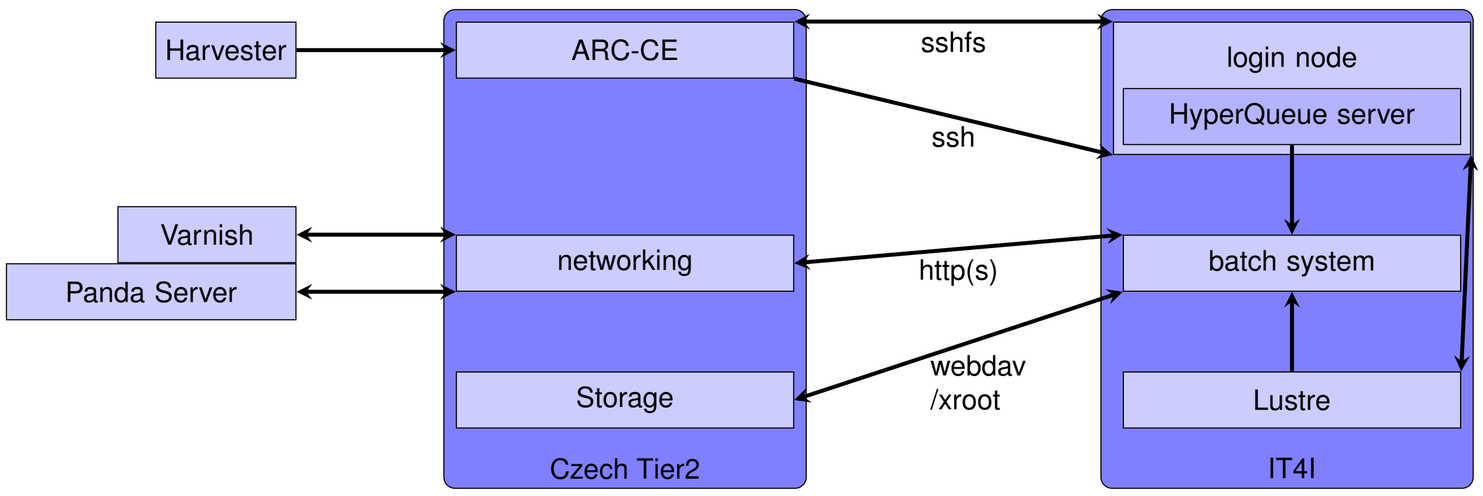
9th Users Conference of IT4Innovations

30.-31.10.2025

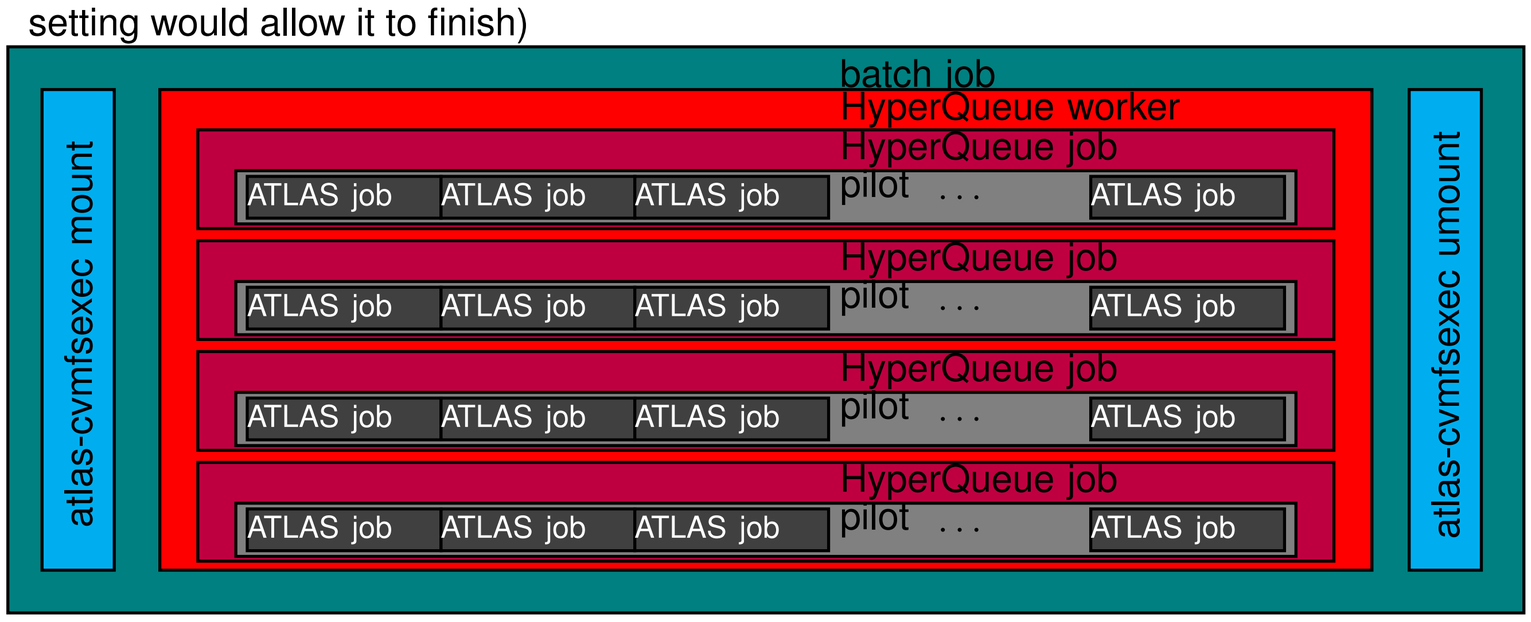
Introduction

For years, the ATLAS experiment has been running a selection of its workflows (mostly Monte Carlo simulations, which have small inputs and outputs and can run on tens of cores for hours) at IT4Innovations. As the experiment and its computing evolve, so does the constitution of its workflows. Recently, we are increasingly getting into a situation where there is an insufficient number of these jobs, while there are many other jobs from other workflows. The most common example is event generation, which has small or no input and small output but, unlike simulations, runs on only one core. An insufficient number of runnable jobs means leaving available resources idle. To address this, the submission system was updated to allow, for example, jobs using one CPU core to run next to jobs requiring 30+ CPU cores. This allows running more workflows.

Submission system

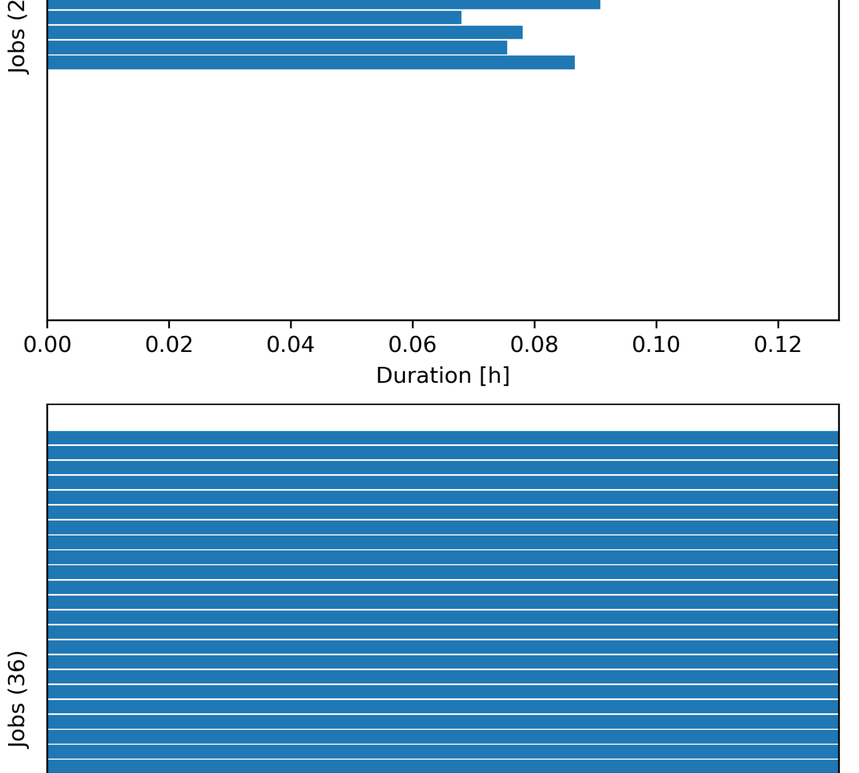


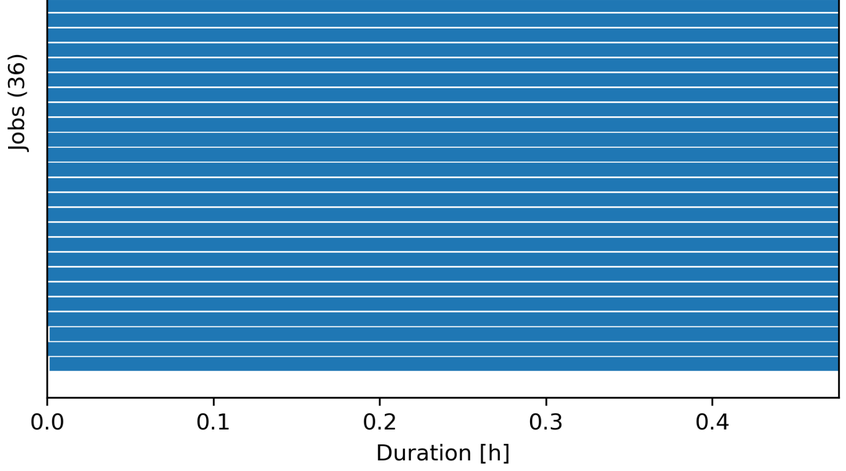
- the ARC-CE shares storage with the Lustre via sshfs connection through a login node and communicates with the batch system via ssh connection (through the same login node)
- when the ARC-CE receives a job, it translates the job description into a script that can be run in the batch system, puts necessary files into a folder within sshfs shared area and submits the job via ssh connection to the HyperQueue server running on a login node
- the HyperQueue server buffers the jobs and when there are enough of them, it submits jobs into the batch system
- when the batch job starts, HyperQueue jobs start in it (in sufficient numbers to fill the worker node - if available) after cvmfsexec (software repository) was mounted
- in each HyperQueue job, pilot wrapper starts, launching the pilot (see the schematic below - example from Karolina)
- requests to panda server made by pilot are routed through Czech Tier2 networking to receive a payload (as there are only few open ports)
- when it receives the payload, it gets input file from the Czech Tier2 storage via xroot or webdav
- then it starts the calculation
- if the calculation requires conditions data, the requests routed through Czech Tier2 networking to external varnish cache
- when the payload finishes, it sends outputs to the Czech Tier2 storage via xroot or webdav
- when this is finished, pilot will request another payload (if it can expect that the batch queue setting would allow it to finish)

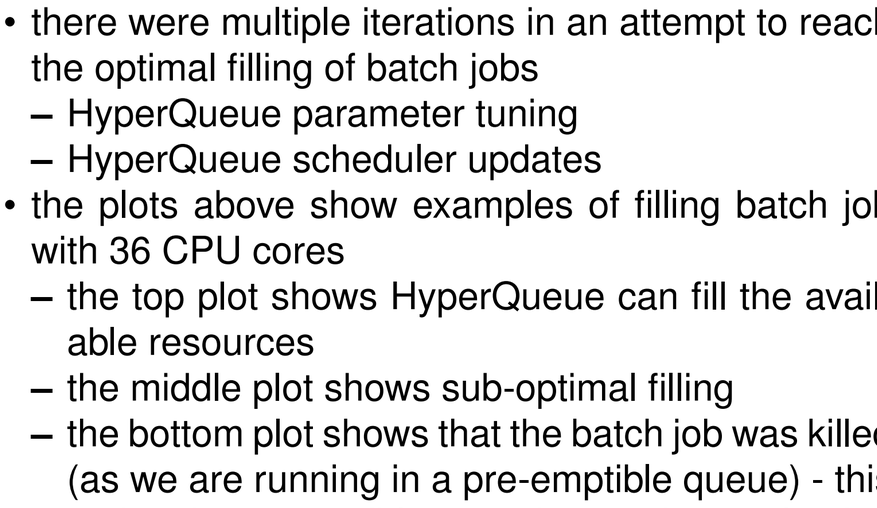


Job mixture testing

- currently, Barbora is used for testing as there are only two distinct cases
  - one job using all 36 CPU cores
  - 36 jobs, each using one CPU core (examples showing when each job starts and ends within one batch job are on plots below)
- on Karolina, the mixture of jobs with different requirements would occur inside a single batch job







- there were multiple iterations in an attempt to reach the optimal filling of batch jobs
  - HyperQueue parameter tuning
  - HyperQueue scheduler updates
- the plots above show examples of filling batch job with 36 CPU cores
  - the top plot shows HyperQueue can fill the available resources
  - the middle plot shows sub-optimal filling
  - the bottom plot shows that the batch job was killed (as we are running in a pre-emptible queue) - this makes it not possible to make precise evaluation of performance and efficiency
- this effort is a work-in-progress

Acknowledgement

This work and computing resources at FZU were co-financed by project CERN-INV (CZ.02.01.01/00/23_015/0008198) from EU funds and MŠMT and project CERN-CZ (LM2023040).
This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254).