Readiness of WL benchmark candidates
ATLAS

Martina Javurkova[1], Lorenzo Rinaldi

[1]University of Massachusetts-Amherst

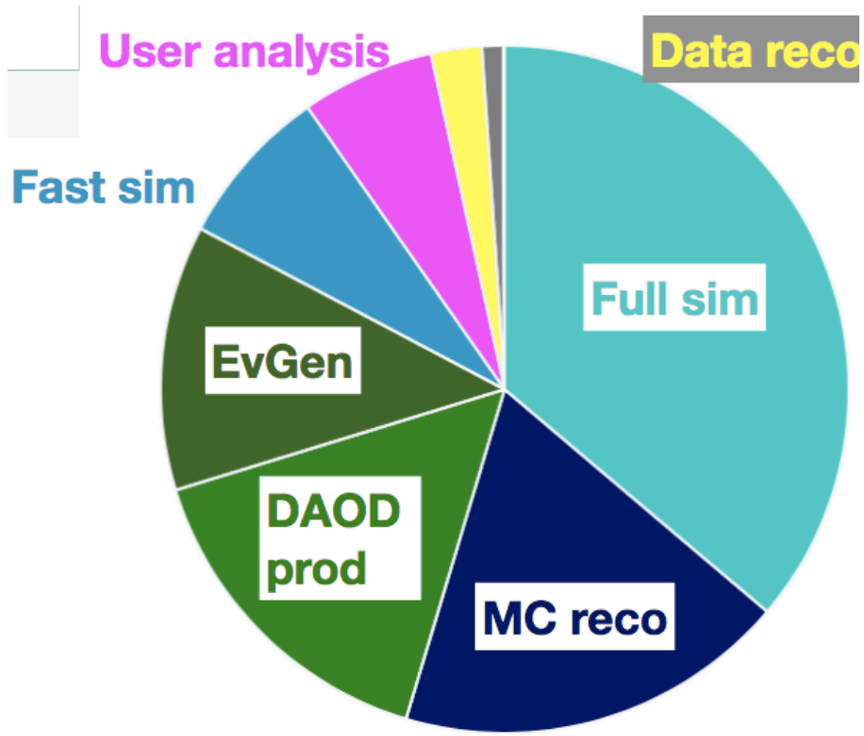
pre-GDB - Benchmarking

08/10/2019

ATLAS workloads

- the actual production workloads of the ATLAS experiment consists of three steps with difference performance

  1. **Event generation**: Generate_tf.py and Athena
  2. **Simulation**: Sim_tf.py and AthenaMP
  3. **Digi-reco**: Reco_tf.py and AthenaMP



- <u>Idea</u>
  - Build standalone containers
  - Parse the outputs and calculate scores [ev/s]
  - Validate (Many thanks to Manfred Alef, Michele Michelotto, Jean-Michel Barbet, …)

Output report

The results are produced in JSON document with all the important information

- Configuration parameters
  - #copies, #threads, #events
- Measurements (from PerfMonSD)
  - Benchmark score [evt/s]
  - Statistics: mean, median, min, max , etc
  - Additional metrics for performance studies
    - Memory and CPU usage
- Container info
  - Version, description, checksum


```
{
  "ncopies": "4",
  "threads_x_copy": "1",
  "events_x_thread": "5",
  "status": {
    "status_copy": [⟺],
    "successful/all": "4/4"
  },
  "events_proc_Athena(MP)": [⟺],
  "CPU_score": {
    "score_proc": [⟺],
    "score": "106.1713",
    "weighted_score": "106.1713",
    "min": "26.0586",
    "max": "27.0270",
    "avg": "26.5428",
    "median": "26.5428",
    "unit": "evt/s"
  },
  "CPU_proc": {
    "cpuError": [⟺],
    "cpuSys": [⟺],
    "cpuAvg": [⟺],
    "evtMaxCPU": [⟺]
  },
  "Memory_proc": {
    "vmem": [⟺],
    "swap": [⟺],
    "RSS": [⟺]
  },
  "log": "ok",
  "app": {
    "version": "v0.4",
    "description": "ATLAS Event Generation based on version based
      on version 19.2.5.5",
    "checksum": "d51b26b336426b8c4c16193c4831cb4d"
  }
}
```

atlas-gen-bmk WL

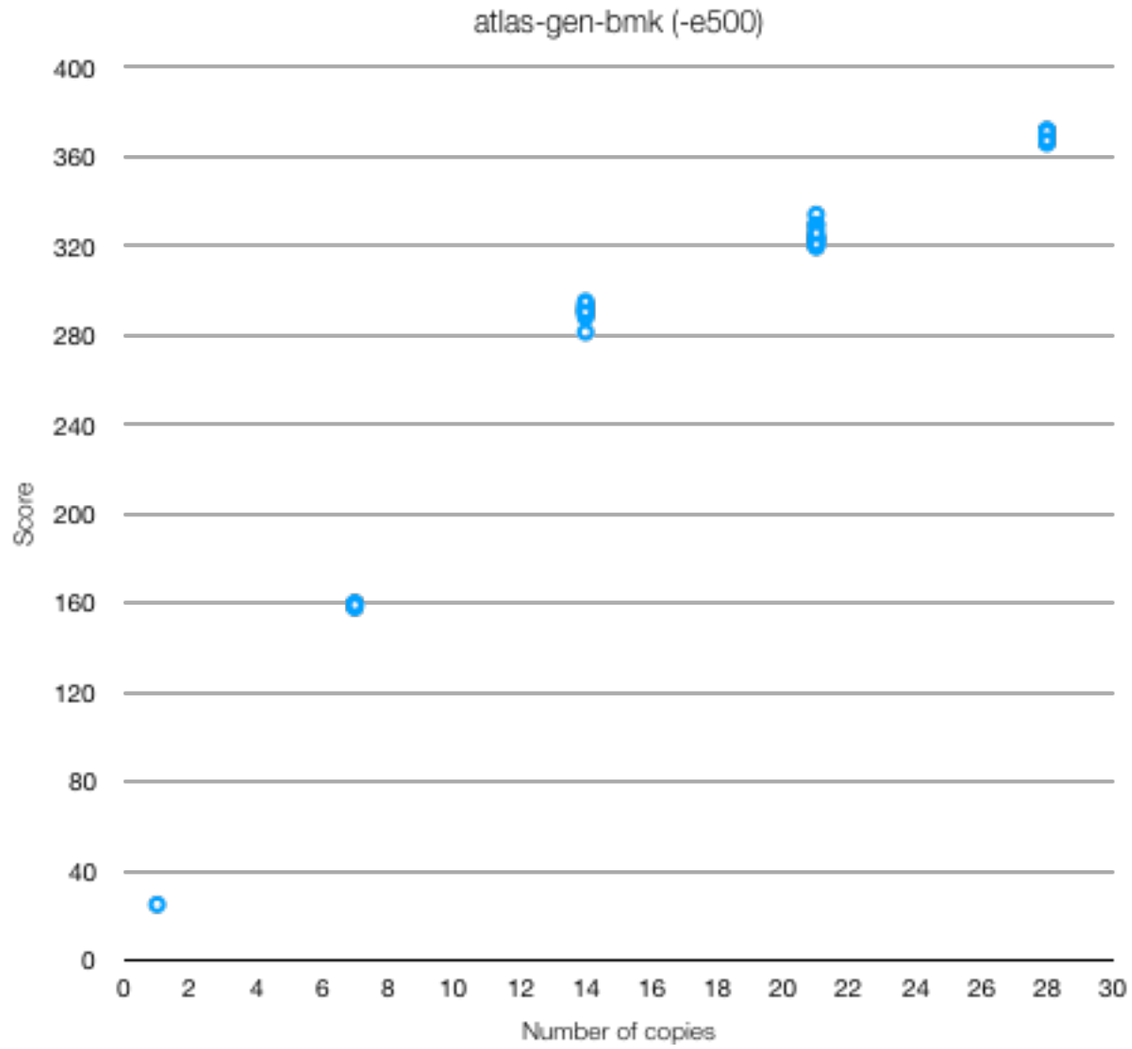
Generation of ttbar MC events. The application is single-process (**Athena**) and requires no input data. The score consists of throughput (events per second).

- Generate_tf.py
  - job options: MC15.410501.PowhegPythia8EvtGen_A14_ttbar_hdamp258p75_nonallhad.py
- Default input parameters
  - NEVENTS_THREAD=500
  - NTHREADS=1 # cannot be changed by user input
  - NCOPIES=$(nproc) # saturated mode

atlas-gen-bmk WL: validation

The **robustness** and **accuracy** of this standalone HEP benchmark are being carefully examined (see Manfred's talk).

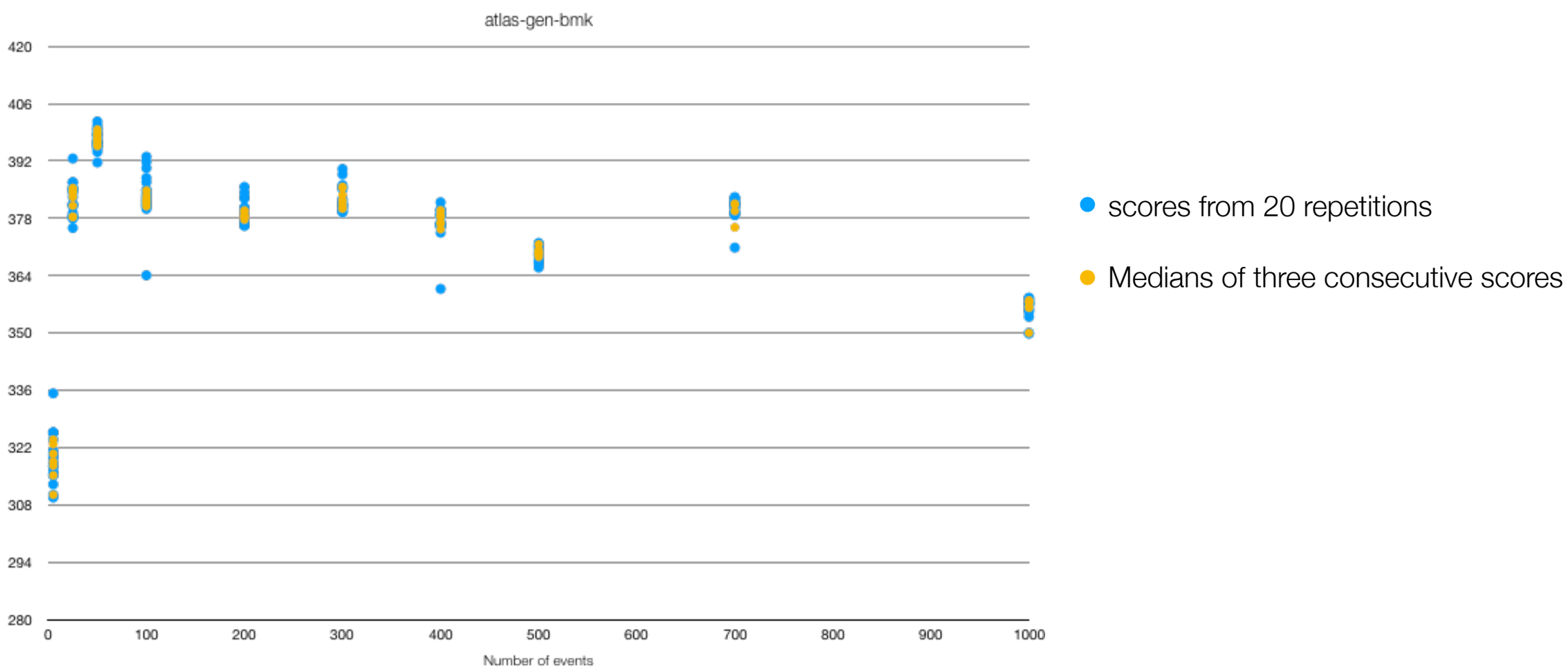
- ✓ Runtime
- ✓ Robustness
- ✓ Memory usage
- ✓ Linearity test[1]
- ◉ Spread



[1] Test performed on the host Intel Xeon E5-2680 (2.4 GHz) with 28 vCPUs running on a CPU socket of 14 physical cores and SMT enabled

atlas-gen-bmk WL: validation (# events)

- Spread (https://its.cern.ch/jira/browse/BMK-41)
  - Studies performed to find the best value of # events:-e5/25/50/100/200/300/400/500/700/1000
  - Spread[1] > 5% caused by **one outlier** for -e5/100/400
  - Spread (medians[2]) < 5% for -e∀



[1]Spread=(Max-Min)/Mean. [2] Median of three consecutive runs

atlas-gen-bmk WL: validation (wall time)

- Wall time (https://its.cern.ch/jira/browse/BMK-211)
  - Idea is to move from cpu time to wall time: more representative
  - Spread is smaller for cpu time than for wall time
  - In some cases the wall time value is equal to zero for -e5
    - Fixed by changing input parameters

—firstEvent="47060001" and —skipEvents="17060000" ⇒ —firstEvent="1"

atlas-sim-bmk WL

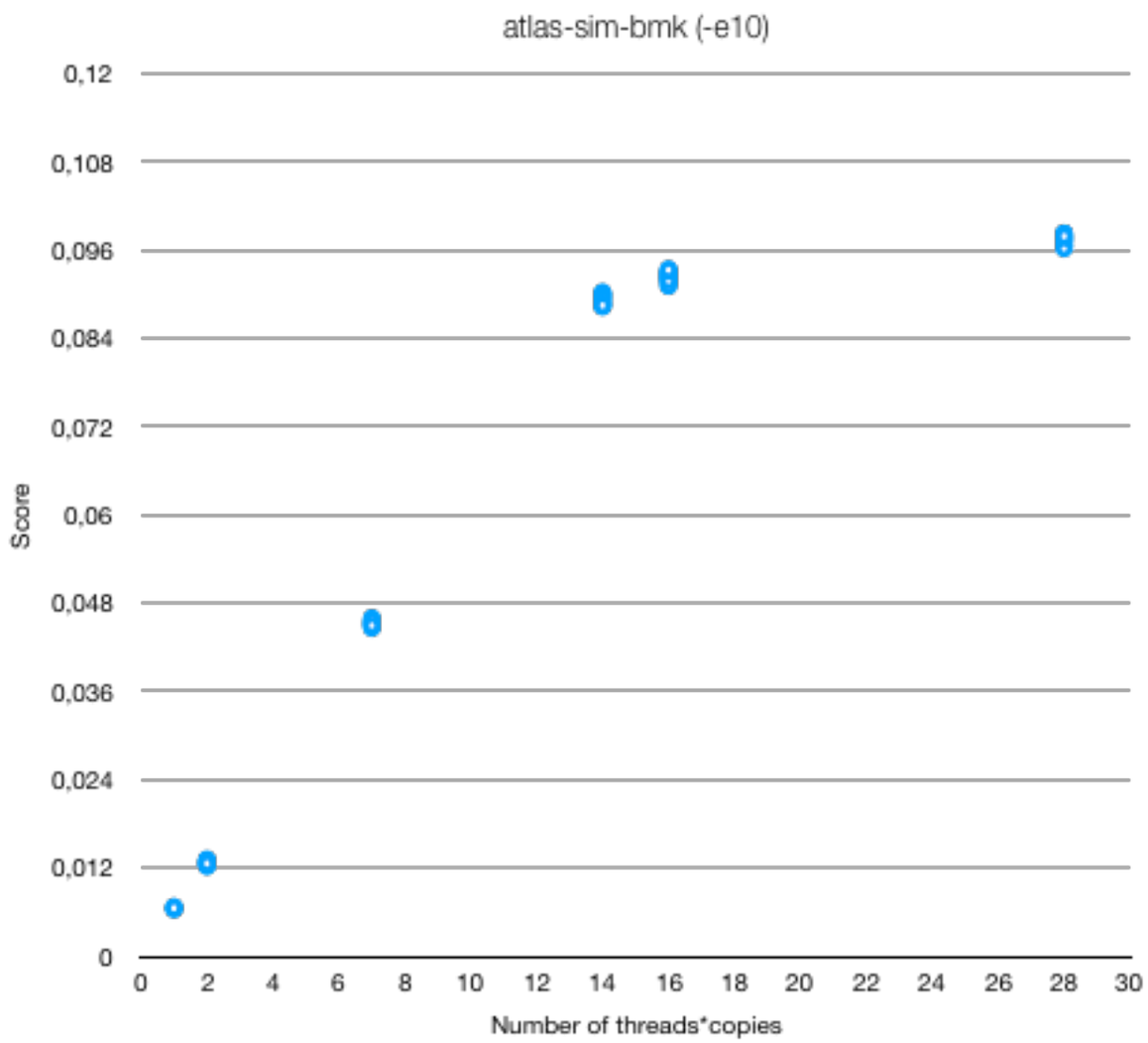
Simulation of MC events that have been generated by a generation workload. The application is multi-process (**AthenaMP**) and reads an input data file containing 10'000 generated events. The score consists of throughput (events per second).

- Sim_tf.py
  - input files (publicly available on CVMFS)
    - EVNT.root from event generation
    - DBRelease: 100.0.2 (Custom, already used for SIM in HPC centers)
- Default input parameters
  - NEVENTS_THREAD=20
  - NTHREADS=4 # in order to match the most possible hardware models
  - NCOPIES=$((`nproc`/$NTHREADS)) # saturated mode

atlas-sim-bmk WL: validation

The **robustness** and **accuracy** of this standalone HEP benchmark are being carefully examined (see Manfred's talk).

- ✓ Runtime
- ✓ Robustness
- ✓ Memory usage
- ✓ Linearity test[1]
- ✓ Spread



[1] Test performed on the host Intel Xeon E5-2680 (2.4 GHz) with 28 vCPUs running on a CPU socket of 14 physical cores and SMT enabled

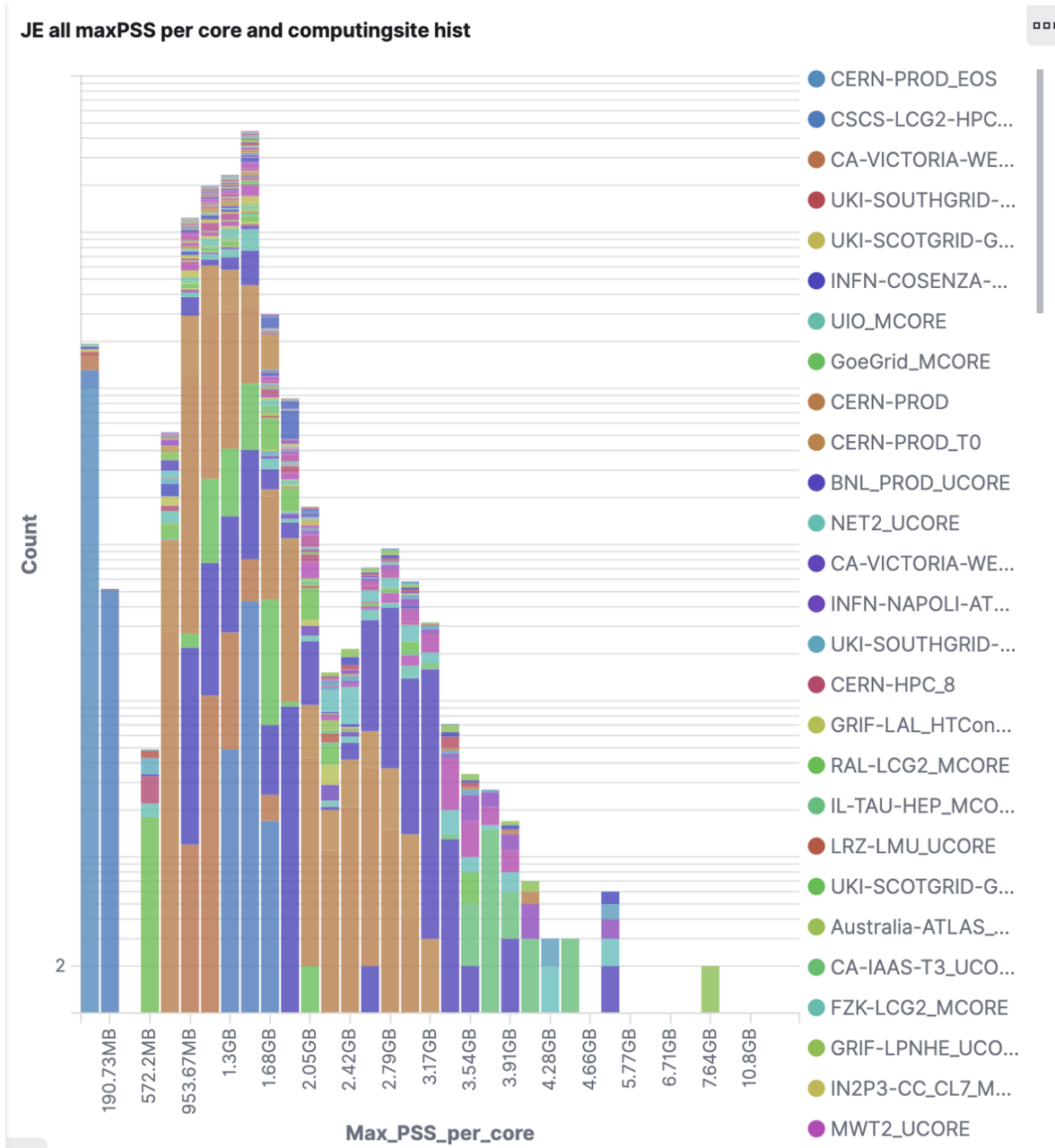
atlas-digi-reco-bmk WL

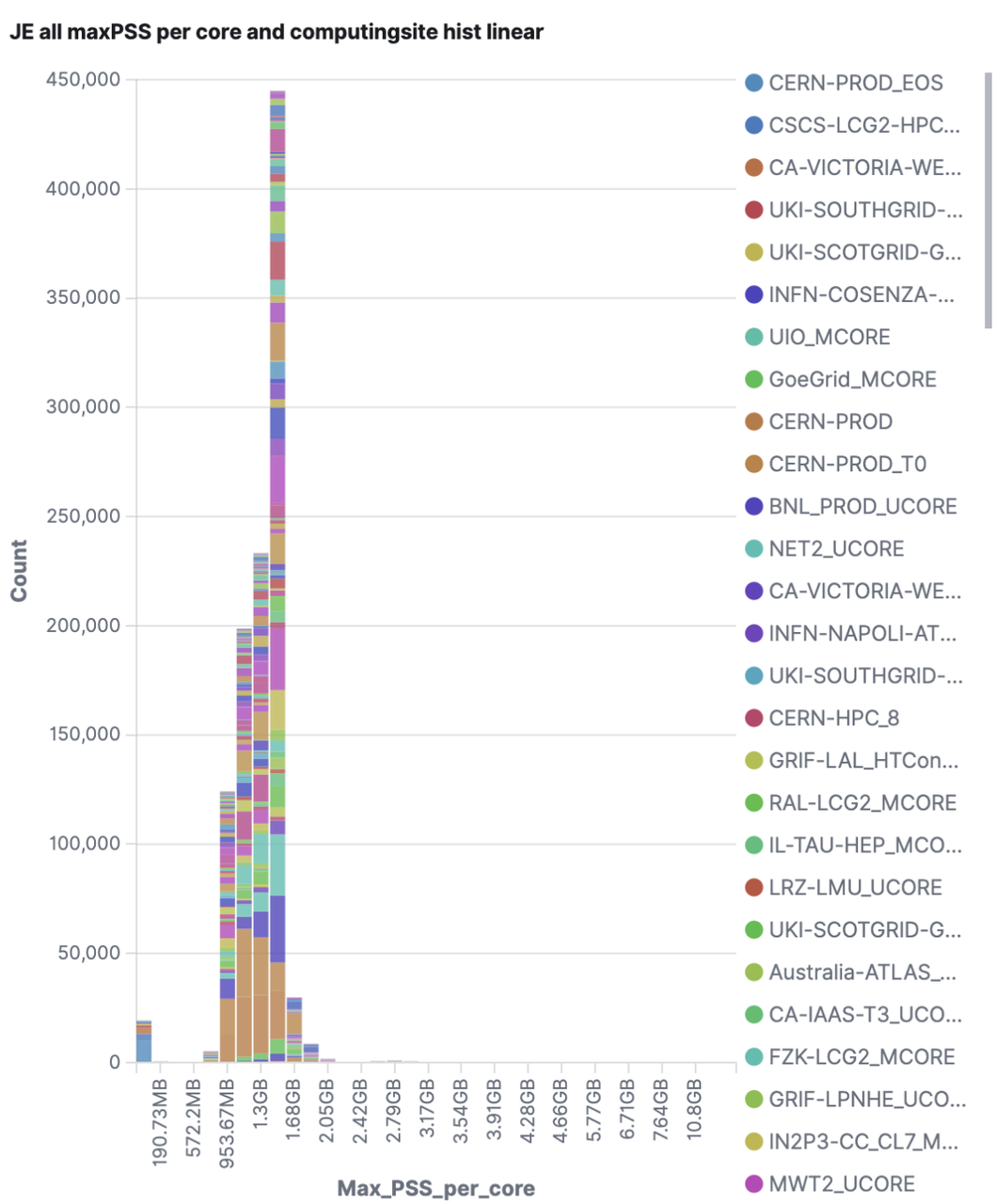
Digitisation, trigger and reconstruction of MC events simulated by a simulation workload. The application is multi-process (**AthenaMP**) and reads input data file(s) containing simulated events. The score consists of throughput (events per second).

- Consists of 4 consecutive steps
  - digitisation: HITtoRDO and RDOtoRDOTrigger
  - reconstruction: RAWtoESD and ESDtoAOD
  - NB: everything in one step RAWtoALL can run only on T0 due to high memory requirements
- Default input parameters
  - NEVENTS_THREAD=10
  - NTHREADS=4 # in order to match the most possible hardware models
  - NCOPIES=$((`nproc`/$NTHREADS)) # saturated mode

atlas-digi-reco-bmk WL: memory intensive WL

- maxPSS/core distribution of all MC digi-reco jobs in the past months





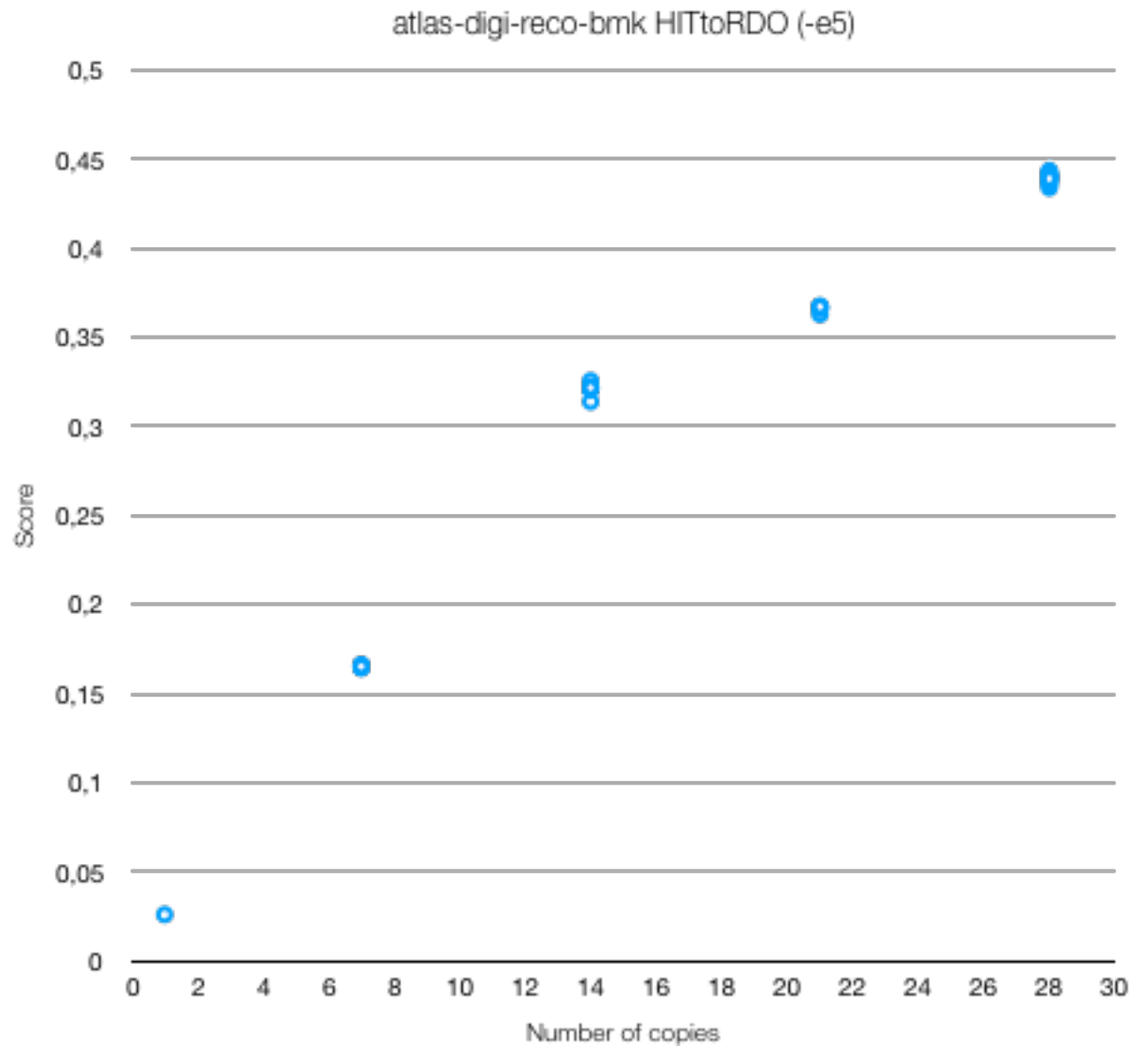
atlas-digi-reco-bmk WL: 1st version

- Standalone single-process (**Athena**) application
  - Not representative because digi-reco jobs are multi-threaded
  - Available only for testing the robustness
- Reco_tf.py
  - Input files (publicly available on CVMFS):
  - Hard-scattering events: HITS.root from simulation
  - Pile-up events: HITS.minbias_lowpt and HITS.minbias_highpt
  - Custom SQLite file for Conditions

atlas-digi-reco-bmk WL: 1st version

Validation

- ✓ Runtime
- ✓ Robustness
- ✓ Linearity test[1]
- ✓ Spread
- ◉ Memory usage



- Sometimes consumes more than a standard 2GB/core WLCG WN (https://its.cern.ch/jira/browse/BMK-171)
- Usually not the problem when enabling swapping but swapping biases the performance scores

[1] Test performed on the host Intel Xeon E5-2680 (2.4 GHz) with 28 vCPUs running on a CPU socket of 14 physical cores and SMT enabled

atlas-digi-reco-bmk WL: 2nd and 3rd versions

- 2nd version
  - Standalone multi-process (**AthenaMP**) application
  - Similar to the first version (used in production)
  - Representative but swapping when running in saturated mode
- 3rd version
  - Standalone multi-process (**AthenaMP**) application
  - Self-contained AMI tag -q221
    - Used for Athena release testing (not used in production)
    - Gives a reasonable memory representation but not representative in terms of I/O

▸ Both versions need to be containerised and carefully validated

atlas-digi-reco-bmk WL: 2nd and 3rd versions

- Local tests on 4-core machine with 7GB mem + 5GB swap (total memory 12 GB)
  - 2nd version: -t4 -c1 (tested with 5GB swap (total memory =12 GB))
  - 3rd version: -t2 -c1 and -t4 -c1 (tested with 1GB swap (total memory = 8GB))

maxPSS (maxSwap) [GB]	2nd version -t4 -c1	3rd version -t2 -c1	3rd version -t4 -c1
HITtoRDO	4.35 (0)	1.71 (0)	2.10 (0)
RDOtoRDOTrigger	6.19 (6.42)	3.62 (0)	4.80 (0)
RAWtoESD	6.15 (0.80)	3.76 (0)	5.60 (0)
ESDtoAOD	3.23 (0)	4.00 (0)	6.37 (0.31)

atlas-digi-reco-bmk WL: proposed solution

- Running high-memory classes of jobs

✓ In the production

  - some jobs exceeding the 2GB while at the same time other jobs are consuming less than 2GB

◉ In the new HEP benchmark

  - All jobs are running concurrently, the performance scores will be biases by swapping

Proposed memory solution

- Implement a default configuration of NTHREADS (ATHENA_PROC_NUMBER) not exceeding 2 GB per process

- Run this particular benchmark in relaxed mode, i.e. with one instance per physical core instead of two instances

- ATLAS will use premixed pile-up overlay soon (instead of using the heavy I/O low/high pt minbias mixing) - this will reduce the disk I/O significantly

Conclusion

gen: **ready but some fine-tuning required**

sim: **ready and validated**

digi-reco: **memory issue understood, container under construction**

- https://gitlab.cern.ch/hep-benchmarks/hep-workloads/tree/qa/atlas
- https://gitlab.cern.ch/hep-benchmarks/hep-workloads/container_registry
- Currently two ATLAS docker images available in the registry
  - gitlab-registry.cern.ch/hep-benchmarks/hep-workloads/atlas-gen-bmk
  - gitlab-registry.cern.ch/hep-benchmarks/hep-workloads/atlas-sim-bmk
- Image for atlas-digi-reco-bmk coming very soon for testing

Backup

Score calculation [ev/s]

- Gen and Sim

$$\text{score} = \sum_{i}^{\text{copies}} \sum_{j}^{\text{threads}} \frac{1000}{< \text{cpu}_{ij} >}$$

- Digi-reco

$$\text{score} = \sum_{i}^{\text{copies}} \sum_{j}^{\text{threads}} \frac{1000}{< \text{cpu}_{ij}^{\text{step1}} > + < \text{cpu}_{ij}^{\text{step2}} > + < \text{cpu}_{ij}^{\text{step3}} > + < \text{cpu}_{ij}^{\text{step4}} >}$$