Hewlett Packard
Enterprise

PERFORMANCE-AWARE BUILD SYSTEM FOR HPC AND AI CONTAINERS

Hybrid HPC Solution Group

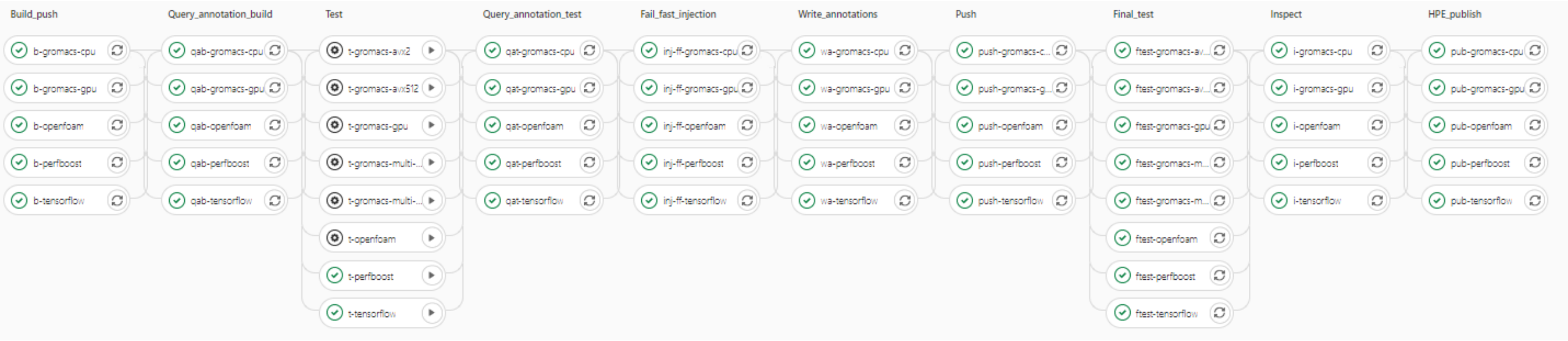
April 15, 2022

INTRODUCTION

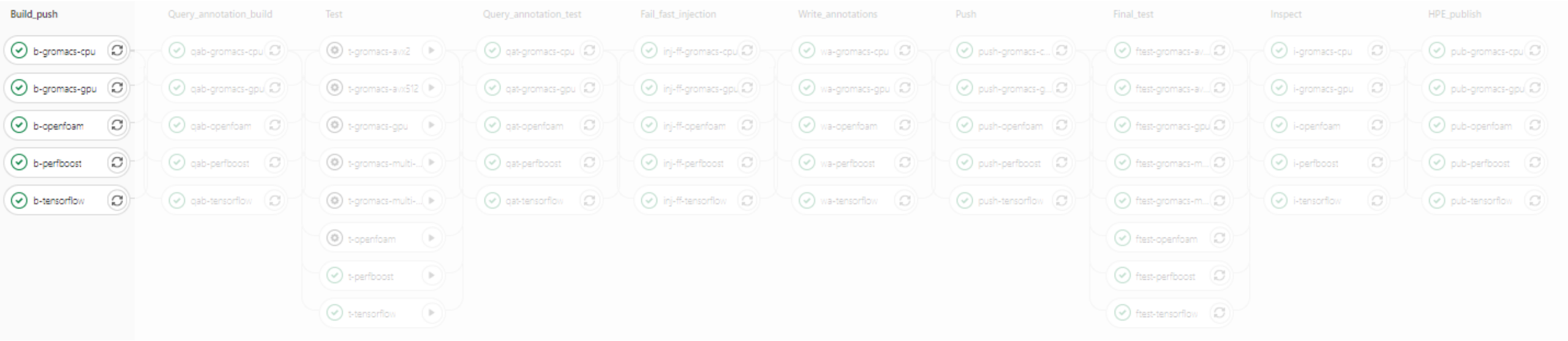
Once we have a curated, high quality container image from our container factory, it's easy to execute the containerized application on different platforms, from a Cray Supercomputer to public cloud, bringing the question on where and how to run optimally? We propose to extend our existing container factory CI/CD build system to add a benchmark step at the end of the pipeline.

The benchmark step will execute micro-benchmarks and the application embedded inside the container with pre-defined dataset(s) and parameters on multiple HPE and public cloud clusters, recording the execution times of each run.

CONTAINER FACTORY EXISTING PIPELINE

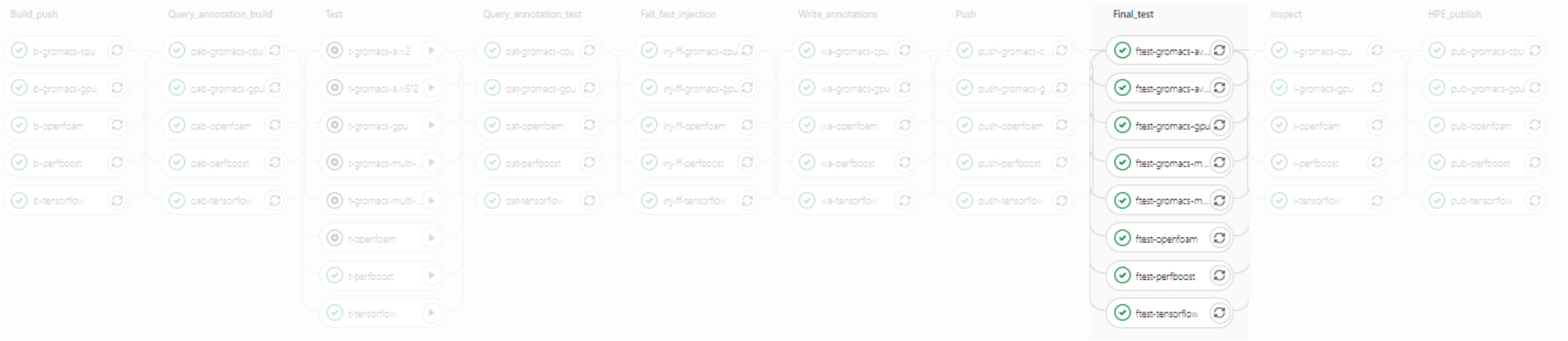


FIRST STEP: BUILD PUSH



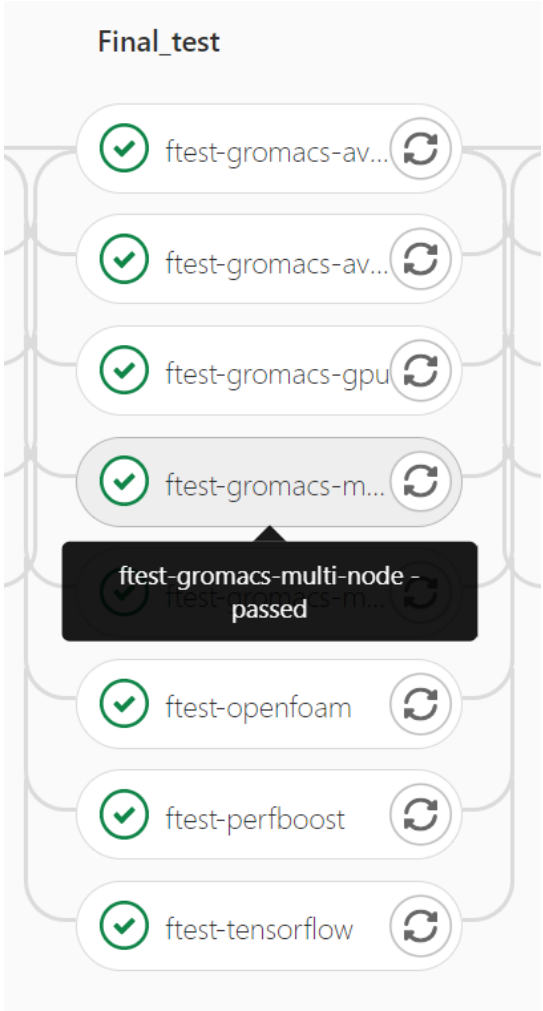
- Build the image from recipe files (Dockerfile)
- OCI format
- Push the image using a temporary tag based on the merge request and stage

FINAL TEST



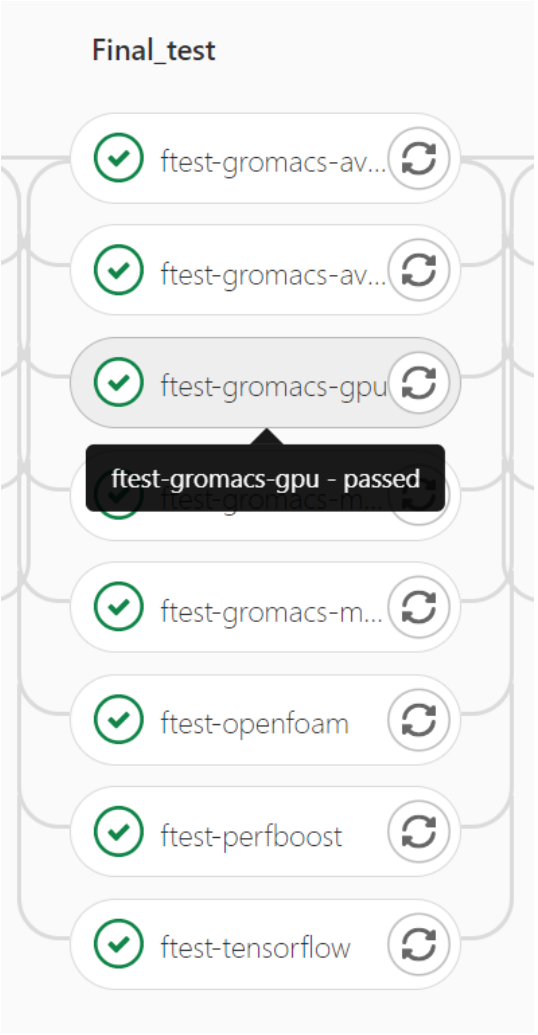
- Validate the final image by running the application
- Execute a performance check (**pipeline build fails on performance loss**), including:
  - Multi-node executions
  - Micro-benchmarks (embedded into the image) e.g., OSU

GROMACS BUILD TIME MULTI-NODE PERFORMANCE CHECK (SINGULARITY)



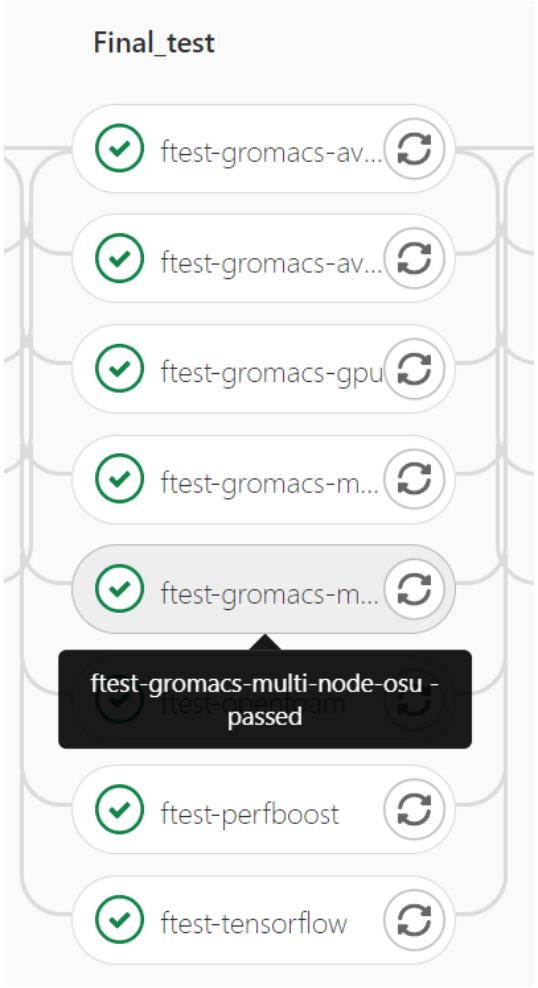

```
290                Core t (s)   Wall t (s)        (%)
291        Time:    16446.380      102.792    15999.7
292                  (ns/day)    (hour/ns)
293 Performance:        6.726        3.568
294 GROMACS reminds you: "If I Were You I Would Give Me a Break" (F. Black)
295 Gromacs performance acceptance threshold 5.38 ns/day
296 Gromacs observed performance 6.726 ns/day
297 Passed Gromacs performance test
298 CF Job Succeeded.
299 Removing /nfs-shared/gitlab-runner/ci_job_tmp_74c3876e-a08c-46ed-b46d-9085c4428fa2
301 Job succeeded
```

GROMACS BUILD TIME GPU PERFORMANCE CHECK (DOCKER)



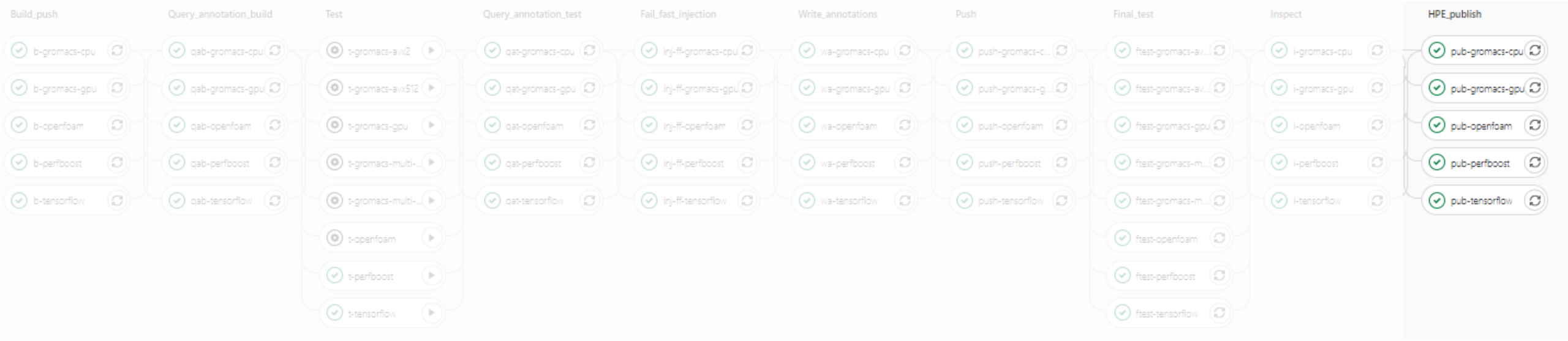

```
405                Core t (s)   Wall t (s)        (%)
406        Time:     4454.779       92.808     4800.0
407                  (ns/day)    (hour/ns)
408  Performance:        7.449        3.222
409  GROMACS reminds you: "You Got to Relate to It" (A.E. Torda)
410  Gromacs performance acceptance threshold 6.0 ns/day
411  Gromacs observed performance 7.449 ns/day
412  Passed Gromacs performance test
413  Passed all performance checks.
414  CF Job Succeeded.
415  Removing /nfs-shared/gitlab-runner/ci_job_tmp_855c8d99-3f87-494b-9e7f-d020bd72cbcb
417  Job succeeded
```

GROMACS BUILD TIME MULTI-NODE OSU CHECK (SINGULARITY)




```
204  524288              12169.50
205  1048576             12225.59
206  2097152             12226.48
207  4194304             12212.41
208  # OSU MPI Bandwidth Test v5.8
209  OSU bandwidth acceptance threshold 10000 MB/s
210  OSU observed bandwidth 12212.41 MB/s
211  validate_osu_bw_performance.sh passed.
212  CF Job Succeeded.
213  Removing /nfs-shared/gitlab-runner/ci_job_tmp_b8a9212c-4a73-4b71-8552-1683639d05b3
215  Job succeeded
```

LAST STEP: PUBLISH THE IMAGE



- Publish the final image to HPE registry or customer private registry

AUTO-BENCHMARK STEP

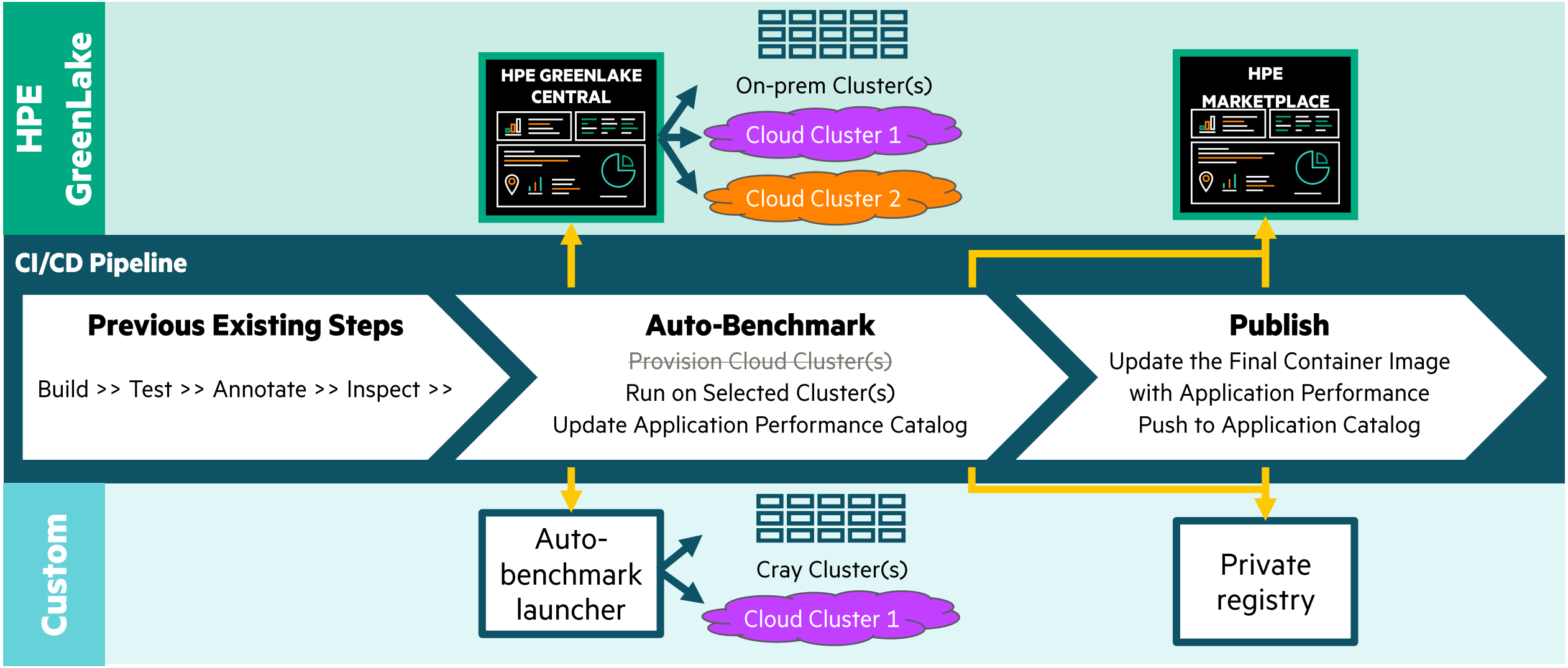
Since we already validate the performance on multiple machines, CPUs/GPUs included, the natural step is to save the performance results to support the decision on where to better run the application

To support that decision, we need to add more tests and machines to the final test pipeline step, including scalability tests

A rich set of experiments during the build time can provide great insights on where and how to run the application, examples:

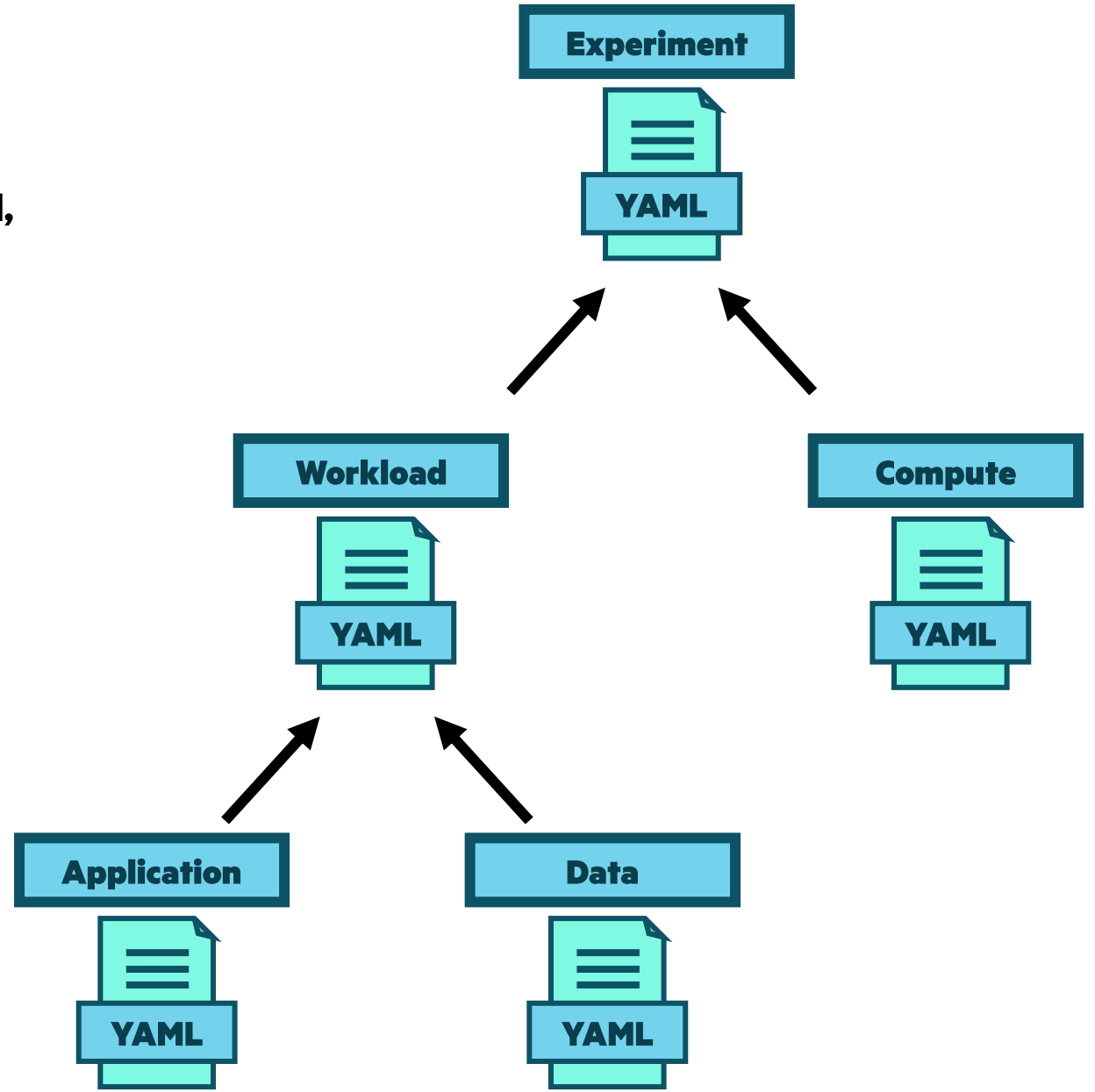
- **CPU or GPU ?**
- **AMD or Intel ?**
- **Does it scale well, how many nodes ?**
- **How many OpenMP threads ?**

AUTO-BENCHMARK STEP

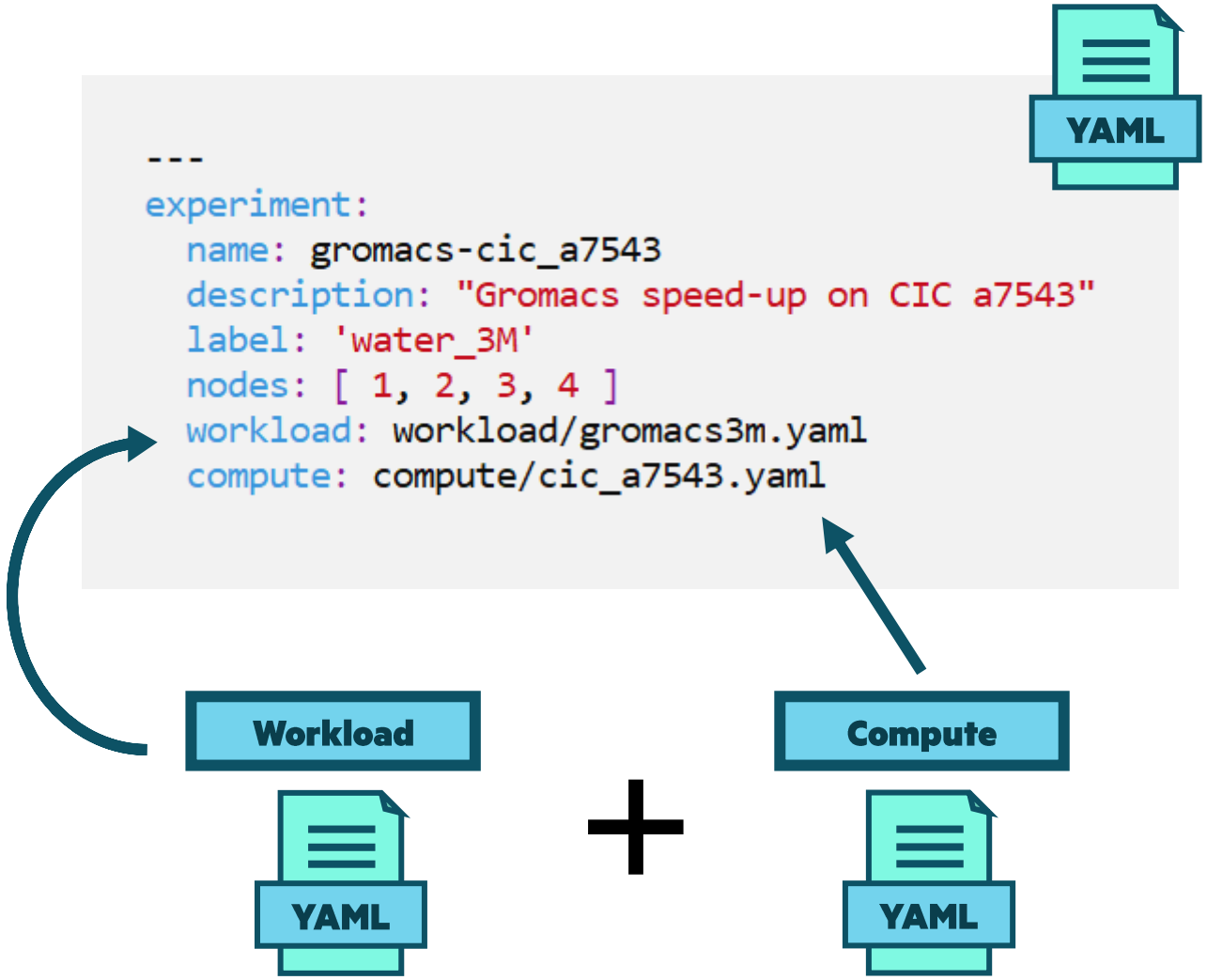


AUTO-BENCHMARK LAUNCHER

- The benchmark launcher is a Python standalone tool, not dependent on the CI/System
- It's also used to execute benchmarks outside the CI, with and without containers
- I uses a set of YAML files to define an experiment
- An experiment contains one workload and one compute
- A workload contains one application and one data



EXPERIMENT CONFIGURATION



WORKLOAD CONFIGURATION

Default processes per node

Pre-process on head-node before the job submission

Pre-process on compute node during job execution

Time measure window during job execution

Performance parser after job on head-node

Performance unit

```yaml
---
workload:
  name: Gromacs_3M
  application: application/gromacs.yaml
  data: data/gromacs.yaml
  runtime: singularity # EMPTY(no-container), singularity, docker, podman, ...
  ppn: 16 # default number of tasks per node

  before_job: |
    cd water-cut1.0_GMX50_bare/3072 # check data before job submission

  job: |
    cd water-cut1.0_GMX50_bare/3072
    singularity run -B \$(pwd) $AUTOBENCH_IMAGE_PATH gmx_mpi grompp -f pme.mdp
    $AUTOBENCH_START_TIME
    mpirun $AUTOBENCH_MPIFLAGS singularity run -B \$(pwd) \
    $AUTOBENCH_IMAGE_PATH gmx_mpi mdrun -resethway -noconfout \
    -nb cpu -nsteps 8000 -v -pin on >../../job_stdout.txt 2>../../job_stderr.txt
    $AUTOBENCH_FINISH_TIME

  after_job: |
    grep --after-context=1 "(ns/day).*(hour/ns)" job_stderr.txt | grep "^Performance:" | \
    cut -d: -f2- | sed -e "s/\ \ */,/g"| cut -d, -f2 > $AUTOBENCH_PERFORMANCE_FILE

  performance:
    unit: 'ns/day'
```

COMPUTE CONFIGURATION

High level description

Resource manager

MPI runtimes and how to activate

Container runtimes And how to activate

Access type

Parallel FS / NFS

Global environment

```yaml
---
compute:
  name: cic_a7532
  description: 'CIC partition cpu-a7532'
  nodes: 4
  arch: x86_64 # uname -m
  os_release: 'SUSE Linux Enterprise High Performance Computing 15 SP2'
  kernel_version: '5.3.18-22-default'
  memory: 515705 # memory per node in MB
  cores_per_node: 64
  htcores_per_node: 64
  cpu_model: 'AMD EPYC 7543 32-Core Processor'
  resource_manager:
    type: slurm
    launcher: 'sbatch -p cpu-a7532'
  mpi:
    - vendor: intel
      version: 2021.3.1
      env: |
        set +eu
        source /opt/intel/oneapi/mpi/2021.3.1/env/vars.sh
  container:
    - type: singularity
      version: 3.7.3
      env: |
        export PATH="/opt/singularity-3.7.3/bin:$PATH"
    - type: docker
      version: 20.10.6-ce
  access:
    type: ssh
    user: cirunner
    host: 10.1.2.3
  globalfs: /bench/benchmarks
  env: |
    export SLURM_NOFEEDBACK=1
```

GROMACS SPEED-UP EXPERIMENT ON COMPUTE CIC-A7532

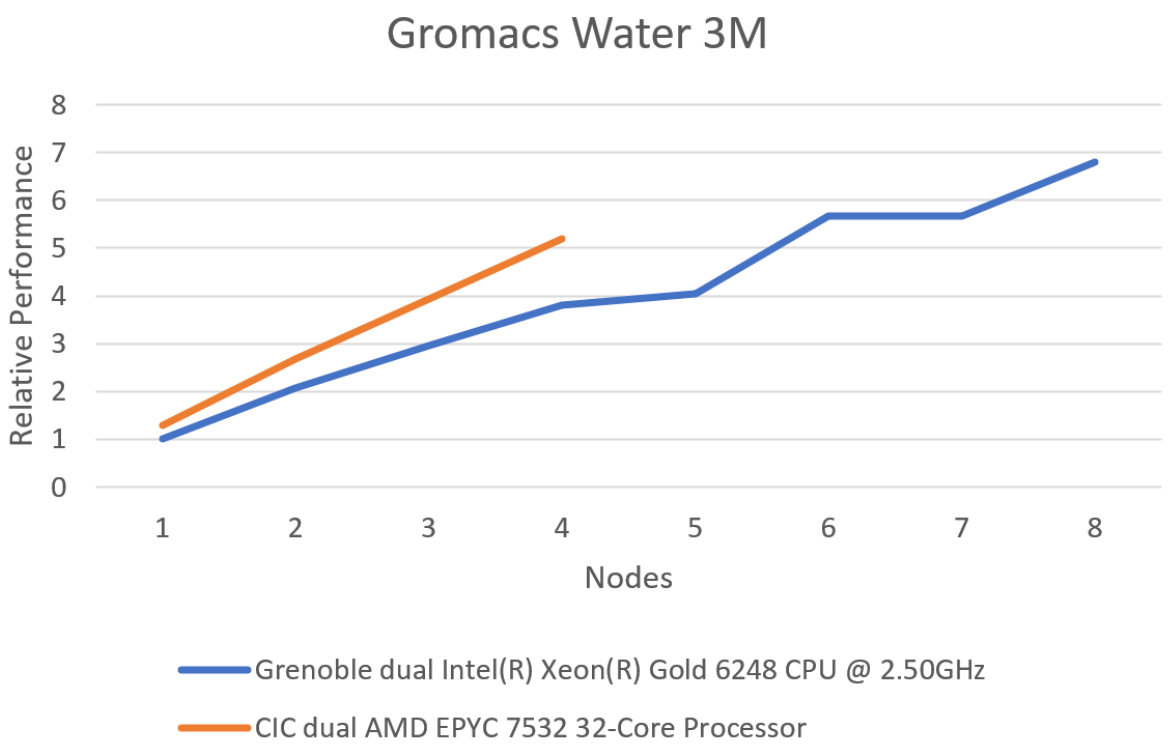

```
./launch.py --experiment experiment/gromacs3m-cic_a7532.yaml
```

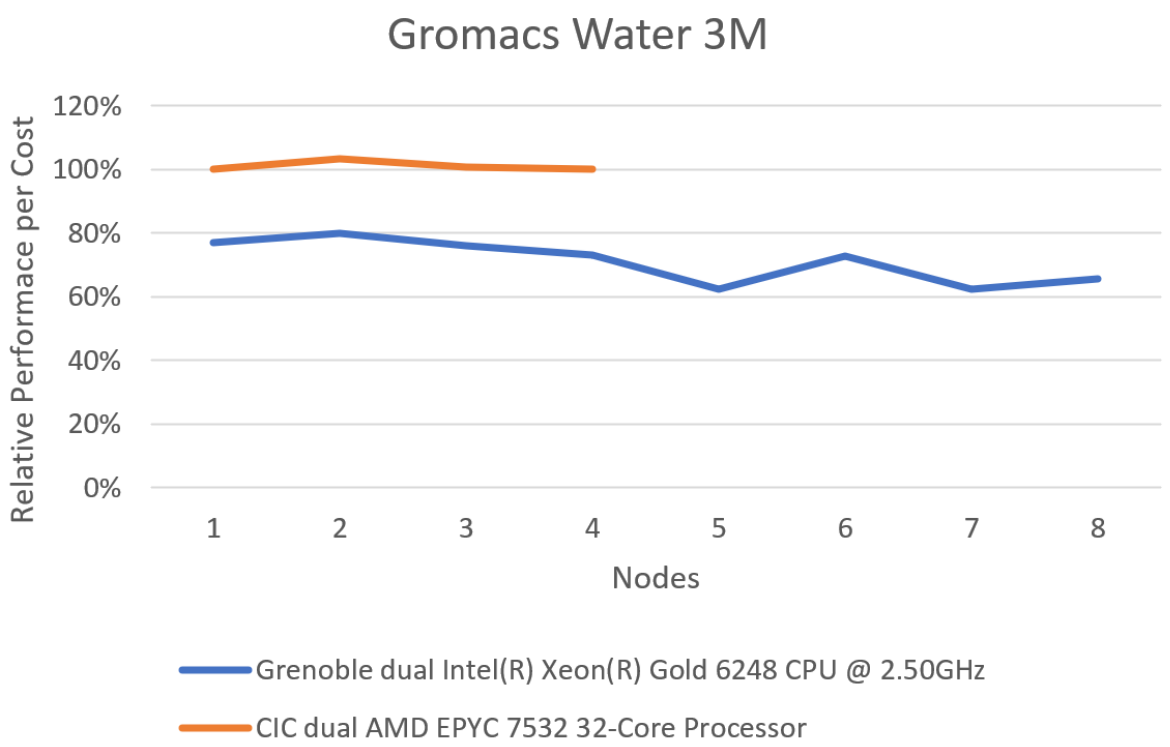
```yaml
---
experiment:
  name: gromacs-cic_a7543
  description: "Gromacs speed-up on CIC a7543"
  label: 'water_3M'
  nodes: [ 1, 2, 3, 4 ]
  workload: workload/gromacs3m.yaml
  compute: compute/cic_a7543.yaml
```

```
cat results/* | grep 'compute,cic_a7532' | grep 'label,water_3M,'

app,Gromacs,dataset_tag,water_GMX50_bare,label,water_3M,compute,cic_a7532,nodes,1,walltime,514.823674434,ns/day,2.699
app,Gromacs,dataset_tag,water_GMX50_bare,label,water_3M,compute,cic_a7532,nodes,2,walltime,251.725320976,ns/day,5.575
app,Gromacs,dataset_tag,water_GMX50_bare,label,water_3M,compute,cic_a7532,nodes,3,walltime,173.869243033,ns/day,8.153
app,Gromacs,dataset_tag,water_GMX50_bare,label,water_3M,compute,cic_a7532,nodes,4,walltime,132.75647342,ns/day,10.8
```

GROMACS PERFORMANCE COMPARISON





Insights:

Better performance on the AMD system (orange)

Good scalability (small 8-node test)

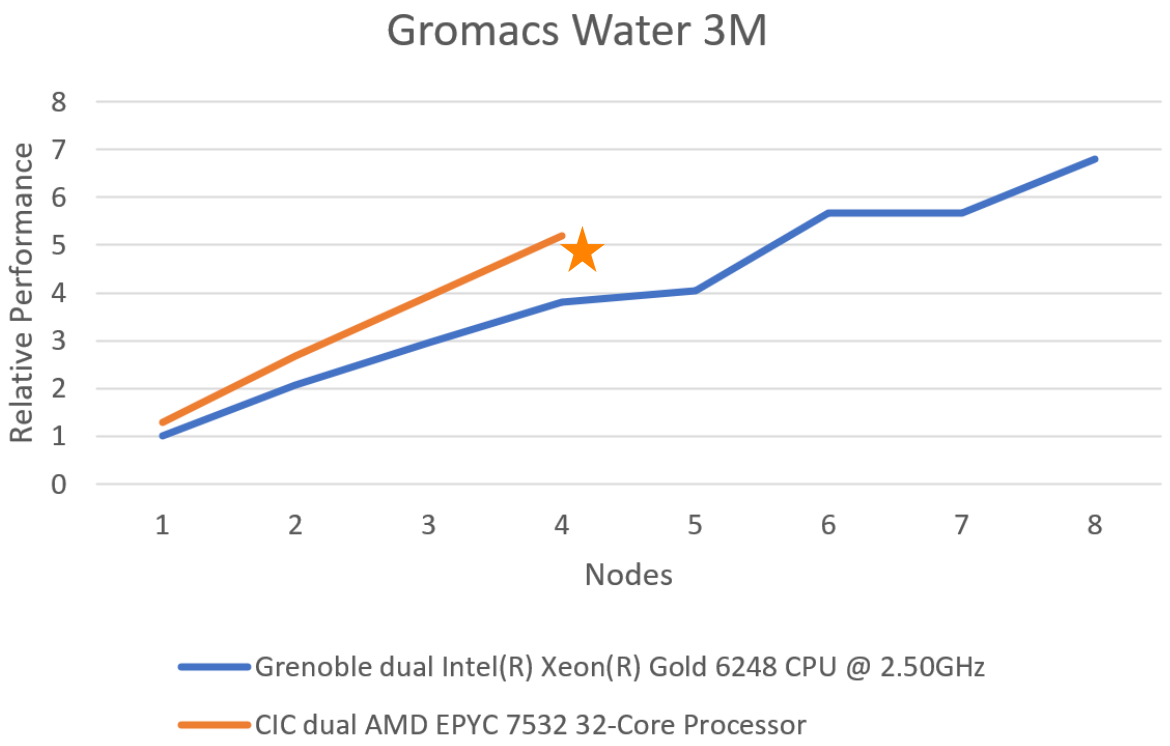
GROMACS FINE TUNING EXPERIMENT ON COMPUTE CIC-A7532

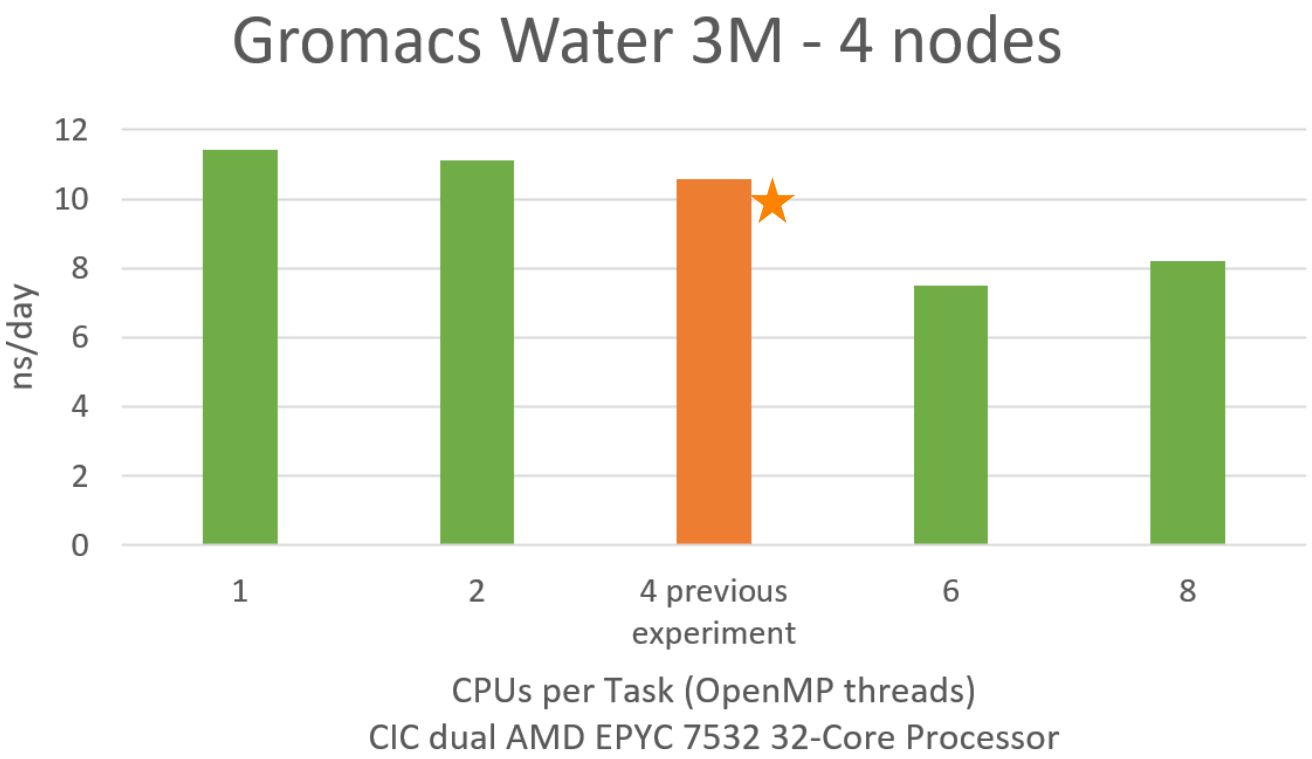
- Find the optimal balance between the number of MPI and OpenMP
- Fixed the compute and the number of nodes (4)
- Executed with 1,2,4 and CPUs per task

- **Best result with one MPI rank per core: OpenMP threads = 1**

```
# cat results/* | grep ',cpus_per_task_'
app,Gromacs,dataset_tag,water_GMX50_bare,label,cpus_per_task_1,compute,cic_a7532,nodes,4,walltime,140.597032139,ns/day,11.426
app,Gromacs,dataset_tag,water_GMX50_bare,label,cpus_per_task_2,compute,cic_a7532,nodes,4,walltime,131.74967541,ns/day,11.109
app,Gromacs,dataset_tag,water_GMX50_bare,label,cpus_per_task_4,compute,cic_a7532,nodes,4,walltime,135.157591446,ns/day,10.587 (previous experiment)
app,Gromacs,dataset_tag,water_GMX50_bare,label,cpus_per_task_6,compute,cic_a7532,nodes,4,walltime,190.782735396,ns/day,7.503
app,Gromacs,dataset_tag,water_GMX50_bare,label,cpus_per_task_8,compute,cic_a7532,nodes,4,walltime,176.969342525,ns/day,8.203
```

GROMACS FINETUNING PERFORMANCE COMPARISON





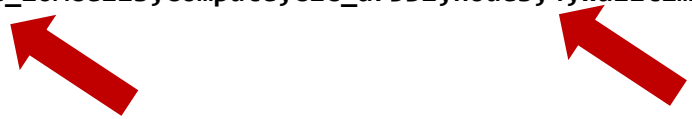
Finetuning Insights:

Better performance with less OpenMP threads (1)

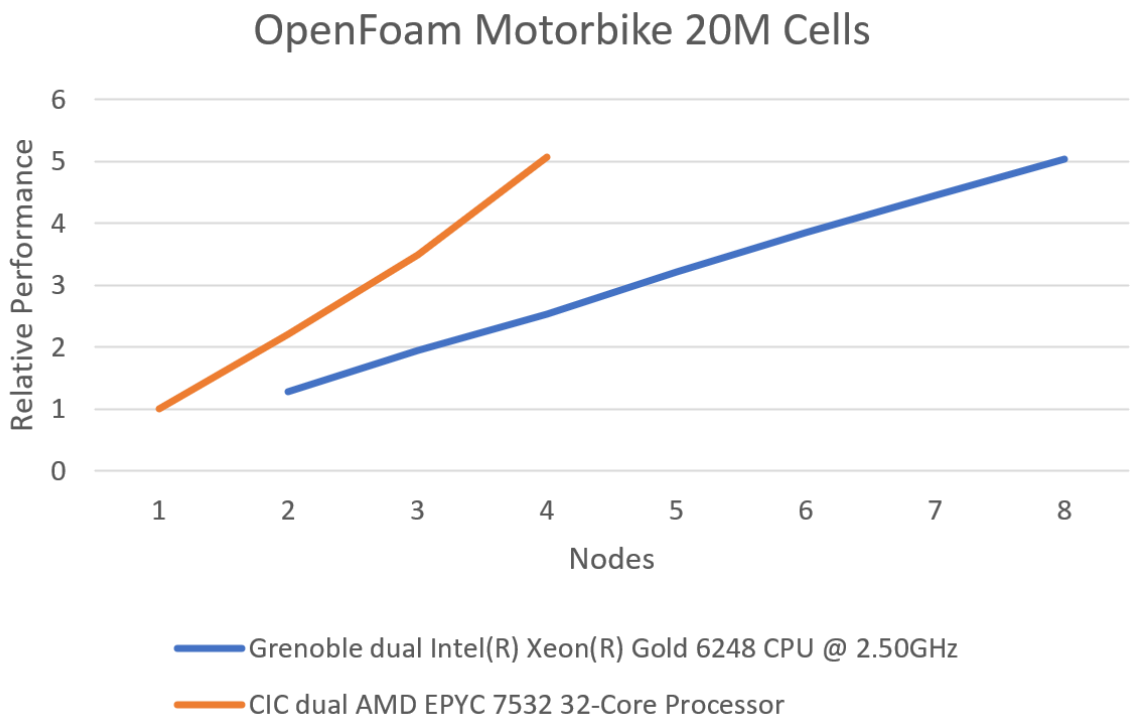
OPENFOAM SPEED-UP EXPERIMENT ON COMPUTE CIC-A7532

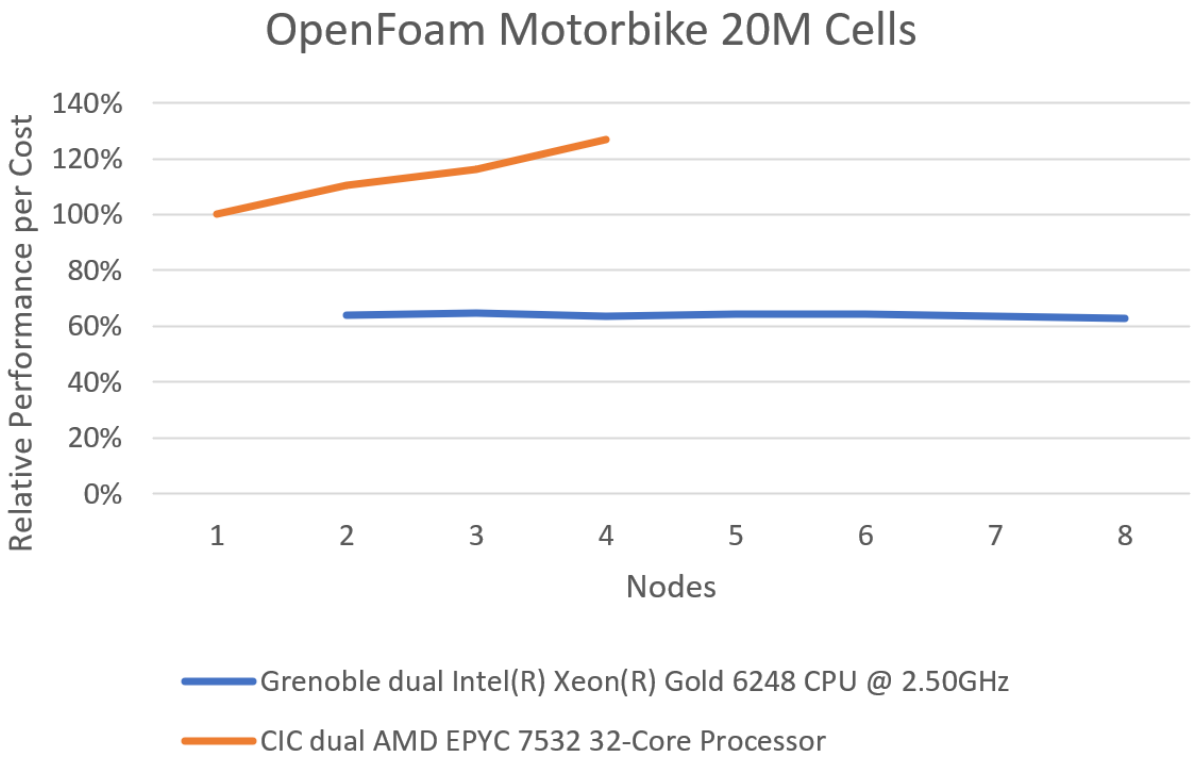
```
./launch.py --experiment experiment/openfoam-cic_a7532.yaml
```

```
# cat results/*csv | grep "label,motorbike_20Mcells.*Mcells/s"
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,grenoble,nodes,2,walltime,581.280796606,Mcells/s,8.710801393728223
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,grenoble,nodes,3,walltime,400.521802543,Mcells/s,13.192612137203167
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,grenoble,nodes,4,walltime,311.77981192,Mcells/s,17.24137931034483
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,grenoble,nodes,5,walltime,265.879917243,Mcells/s,21.83406113537118
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,grenoble,nodes,6,walltime,228.572317809,Mcells/s,26.17801047120419
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,grenoble,nodes,7,walltime,199.83679097,Mcells/s,30.303030303030305
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,grenoble,nodes,8,walltime,195.877308044,Mcells/s,34.24657534246575
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,cic_a7532,nodes,1,walltime,745.289153965,Mcells/s,6.802721088435374
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,cic_a7532,nodes,2,walltime,344.631562758,Mcells/s,15.015015015015015
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,cic_a7532,nodes,3,walltime,224.014489963,Mcells/s,23.696682464454977
app,OpenFoam,dataset_tag,motorbike_20Mcells,label,motorbike_20Mcells,compute,cic_a7532,nodes,4,walltime,160.169417786,Mcells/s,34.48275862068966
```



OPENFOAM PERFORMANCE COMPARISON





Insights:

Better performance on the AMD system (orange)

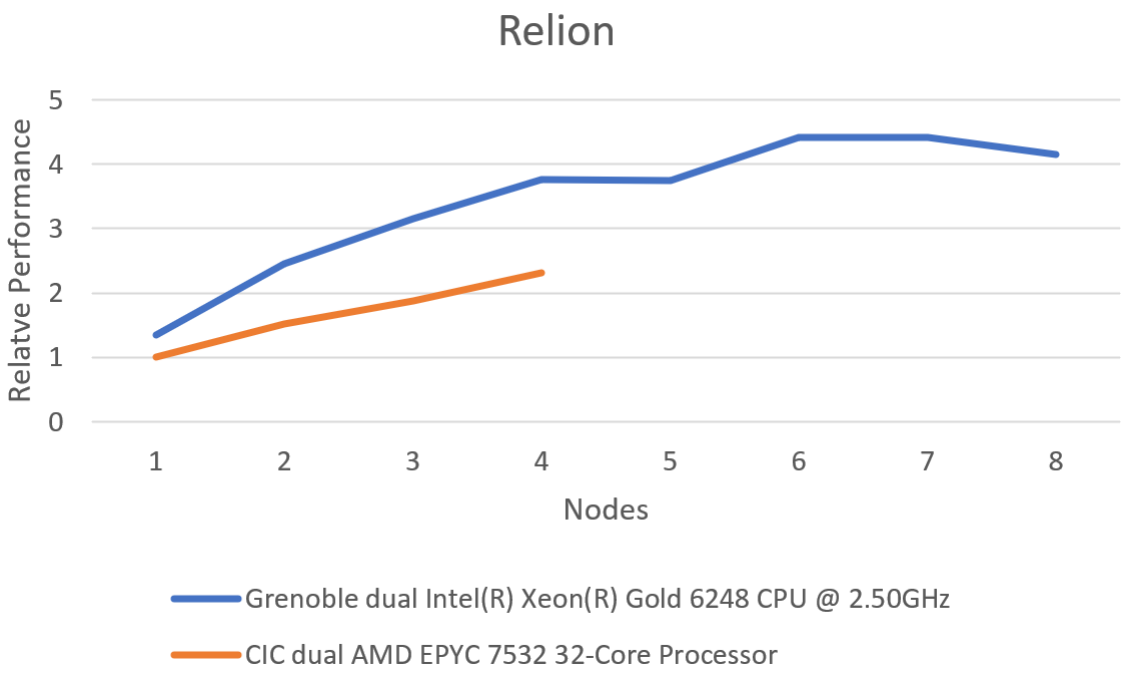
Good scalability (small 8-node test)

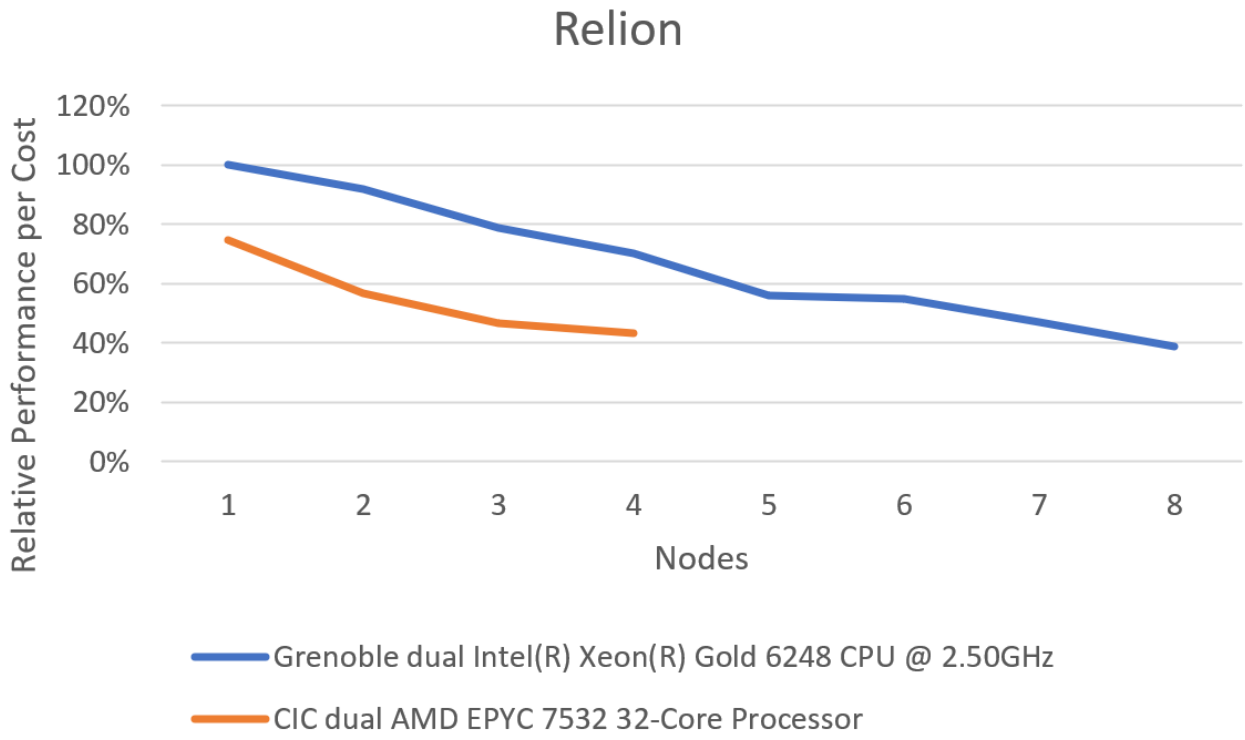
RELION SPEED-UP EXPERIMENT ON COMPUTE CIC-A7532

```
./launch.py --experiment experiment/relion-cic_a7532.yaml
```

```
# cat results/*.csv | grep "label,j8.*compute,cic_a7532"
app,Relion,dataset_tag,relion_benchmark,label,j8,compute,cic_a7532,nodes,1,walltime,2834.917614888,jobs/day,30.47707614015211
app,Relion,dataset_tag,relion_benchmark,label,j8,compute,cic_a7532,nodes,2,walltime,1861.323023747,jobs/day,46.4185952130273
app,Relion,dataset_tag,relion_benchmark,label,j8,compute,cic_a7532,nodes,3,walltime,1512.731822962,jobs/day,57.115212814671104
app,Relion,dataset_tag,relion_benchmark,label,j8,compute,cic_a7532,nodes,4,walltime,1229.025106321,jobs/day,70.29962167219863
```

RELION PERFORMANCE COMPARISON





Insights:

Better performance on the Intel system (blue)

Poor scalability

CONCLUSIONS

In addition to validate the application for correct execution is also important to validate the performance during the build process.

The extra step to execute on multiple systems with multiple parameters were able to provide useful insights on where and how to run the application.

If you add the machine costs, the performance per cost metric will provide even better information on where to run

THANK YOU