UL HPC School 2015

PS2: HPC workflows with sequential jobs

H. Cartiaux

University of Luxembourg, Luxembourg

Latest versions available on Github:

UL HPC tutorials: https://github.com/ULHPC/tutorials

UL HPC School: http://hpc.uni.lu/hpc-school/

PS2tutorial sources:

https://github.com/ULHPC/tutorials/tree/devel/basic/sequential_jobs

Summary

1. Objectives
2. Pre-requisites
3. Exercise 1: parametric execution of Gromacs
4. Exercise 2: Watermarking images in Python
5. Exercise 3: Advanced use case, using a Java program: "JCell"
6. Conclusion

Summary

1 Objectives

2 Pre-requisites

3 Exercise 1: parametric execution of Gromacs

4 Exercise 2: Watermarking images in Python

5 Exercise 3: Advanced use case, using a Java program: "JCell"

6 Conclusion

Objectives of the PS

- Run sequential, parametric programs on the clusters
- Learn how-to use our set of launcher scripts
- Submit jobs, and use the cluster monitoring tools
  - ↪ drawgantt
  - ↪ monika
  - ↪ ganglia

Read the full subject of this PS here

- http://git.io/5cYmPw

Summary

1 Objectives

2 Pre-requisites

3 Exercise 1: parametric execution of Gromacs

4 Exercise 2: Watermarking images in Python

5 Exercise 3: Advanced use case, using a Java program: "JCell"

6 Conclusion

MPI tasks: getting started

1. Connect to the cluster(s)

```
(node)$> ssh chaos-cluster
(node)$> ssh gaia-cluster
```

2. Transfer files

```
(node)$> rsync -avz directory chaos-cluster:
```

3. Submit jobs:

```
(node)$> oarsub -I
(node)$> oarsub ./program
```

Follow this part: "Connect the the cluster and set-up the environment for this tutorial"

- http://git.io/5cYmPw

Summary

1 Objectives

2 Pre-requisites

3 **Exercise 1: parametric execution of Gromacs**

4 Exercise 2: Watermarking images in Python

5 Exercise 3: Advanced use case, using a Java program: "JCell"

6 Conclusion

Gromacs

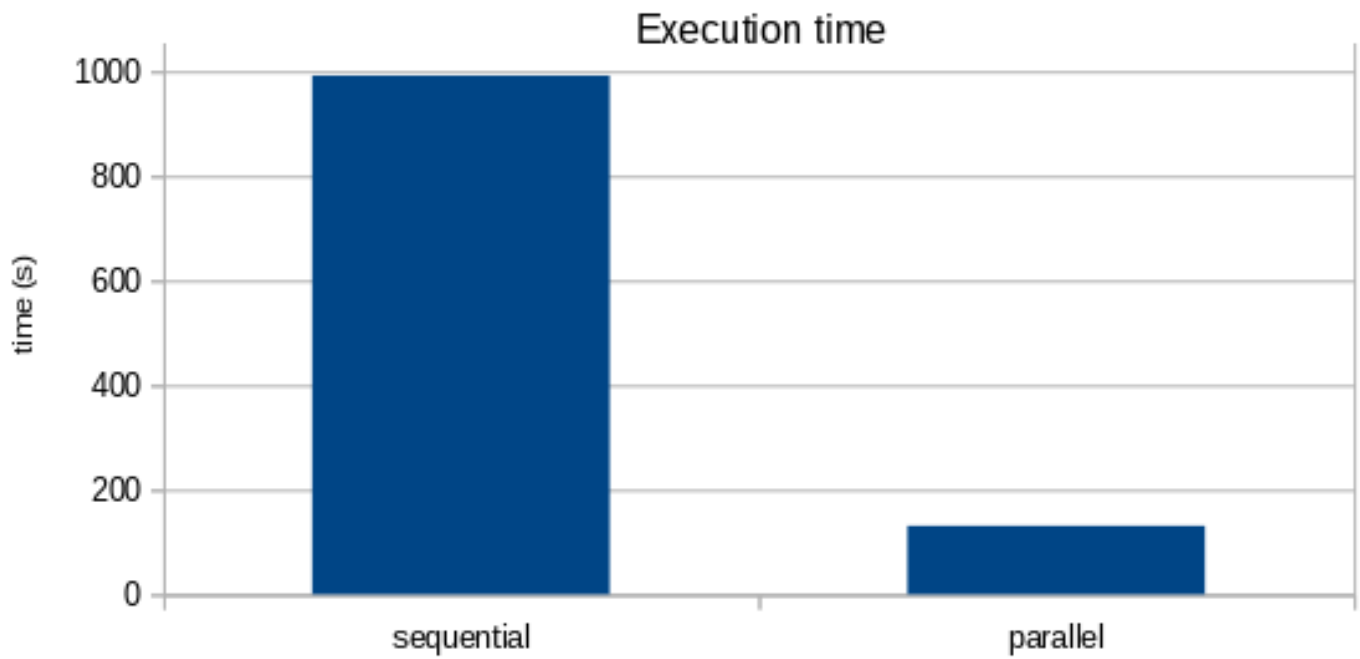
GROMACS: GROningen MAchine for Chemical Simulations
versatile package for molecular dynamics, primarily designed for biochemical molecules

- very large codebase: 1.836.917 SLOC
- many applications in the package, several parallelization modes
- **mdrun**: computational chemistry engine, performing:
  - ↪ molecular dynamics simulations
  - ↪ Brownian Dynamics, Langevin Dynamics
  - ↪ Conjugate Gradient
  - ↪ L-BFGS
  - ↪ Steepest Descents energy minimization
  - ↪ Normal Mode Analysis
- **mdrun** - parallelized using MPI, OpenMP, pthreads and with support for GPU acceleration

Exercise 1: parametric execution

2 approaches

- Sequential (loop)
- Parallized (with GNU parallel)



Summary

1 Objectives

2 Pre-requisites

3 Exercise 1: parametric execution of Gromacs

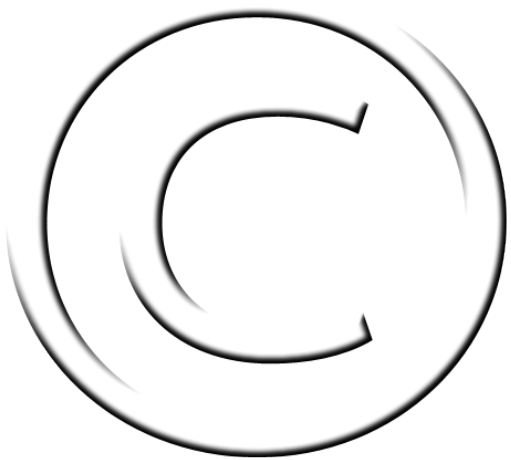
4 **Exercise 2: Watermarking images in Python**

5 Exercise 3: Advanced use case, using a Java program: "JCell"

6 Conclusion

Task: apply a watermark to a given set of pictures

- Simple Python script
- Generic parallel launcher
- Distribute the work on several nodes



Summary

1 Objectives

2 Pre-requisites

3 Exercise 1: parametric execution of Gromacs

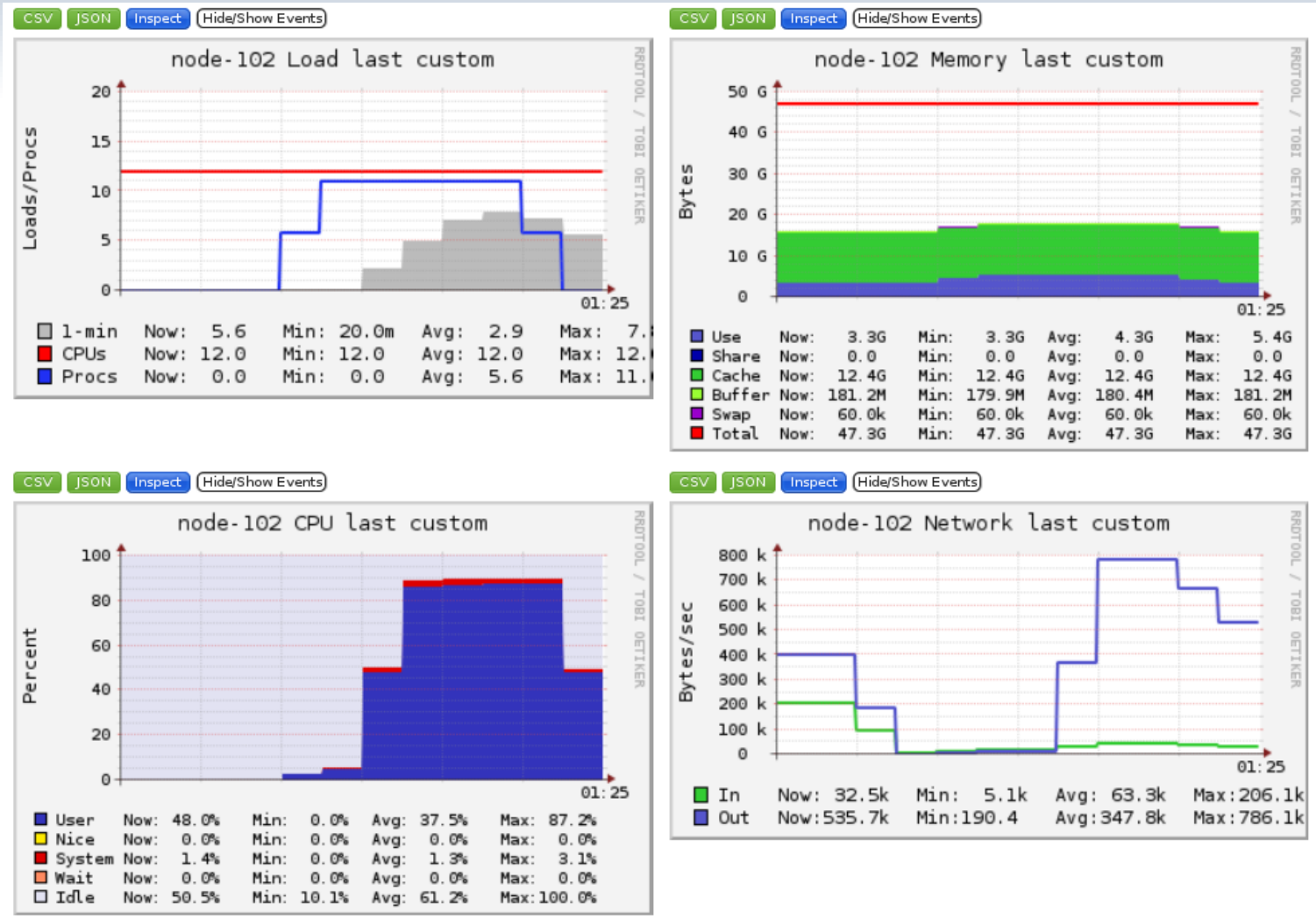
4 Exercise 2: Watermarking images in Python

5 **Exercise 3: Advanced use case, using a Java program: "JCell"**

6 Conclusion

Subject

- **JCell**: a Java framework for working with genetic algorithms
- Example: Generational algorithm for the Combinatorial ECC problem
- Test the variations of these parameters: *Mutation probability* and *Crossover probability*


CSV JSON Inspect Hide/Show Events
node-102 Load last custom
Loads/Procs
20
15
10
5
0
01:25
RRDTOOL / TOBI OETIKER
1-min Now: 5.6 Min: 20.0m Avg: 2.9 Max: 7.
CPUs Now: 12.0 Min: 12.0 Avg: 12.0 Max: 12.
Procs Now: 0.0 Min: 0.0 Avg: 5.6 Max: 11.
CSV JSON Inspect Hide/Show Events
node-102 Memory last custom
Bytes
50 G
40 G
30 G
20 G
10 G
0
01:25
RRDTOOL / TOBI OETIKER
Use Now: 3.3G Min: 3.3G Avg: 4.3G Max: 5.4G
Share Now: 0.0 Min: 0.0 Avg: 0.0 Max: 0.0
Cache Now: 12.4G Min: 12.4G Avg: 12.4G Max: 12.4G
Buffer Now: 181.2M Min: 179.9M Avg: 180.4M Max: 181.2M
Swap Now: 60.0k Min: 60.0k Avg: 60.0k Max: 60.0k
Total Now: 47.3G Min: 47.3G Avg: 47.3G Max: 47.3G
CSV JSON Inspect Hide/Show Events
node-102 CPU last custom
Percent
100
80
60
40
20
0
01:25
RRDTOOL / TOBI OETIKER
User Now: 48.0% Min: 0.0% Avg: 37.5% Max: 87.2%
Nice Now: 0.0% Min: 0.0% Avg: 0.0% Max: 0.0%
System Now: 1.4% Min: 0.0% Avg: 1.3% Max: 3.1%
Wait Now: 0.0% Min: 0.0% Avg: 0.0% Max: 0.0%
Idle Now: 50.5% Min: 10.1% Avg: 61.2% Max: 100.0%
CSV JSON Inspect Hide/Show Events
node-102 Network last custom
Bytes/sec
800 k
700 k
600 k
500 k
400 k
300 k
200 k
100 k
0
01:25
RRDTOOL / TOBI OETIKER
In Now: 32.5k Min: 5.1k Avg: 63.3k Max: 206.1k
Out Now: 535.7k Min: 190.4 Avg: 347.8k Max: 786.1k

Summary

1 Objectives

2 Pre-requisites

3 Exercise 1: parametric execution of Gromacs

4 Exercise 2: Watermarking images in Python

5 Exercise 3: Advanced use case, using a Java program: "JCell"

6 Conclusion

Conclusion

- We have covered the most common workflow: **parametric jobs**
- Our launchers can be improved!

Perspectives

- Checkpoint/Restart mechanism
- Best effort jobs
- OAR array jobs

Questions?

1. Objectives
2. Pre-requisites
3. Exercise 1: parametric execution of Gromacs
4. Exercise 2: Watermarking images in Python
5. Exercise 3: Advanced use case, using a Java program: "JCell"
6. Conclusion