Exercises for High Performance Computing (MA-INF 1108) WS 2023/2024

E. Suarez, M. Wolter and B. Kostrzewa
Tutors: O. Vrapcani and N. Pillath

2 Running Jobs

In this exercise you will learn about the software environment and slurm job submission on the JUWELS Cluster.

1: Software Modules documentation
To facilitate HPC users when building their software, the JSC systems provides a module environment, containing hundreds of the most frequently used software components (compilers, libraries, etc.) Check the software modules available in JUWELS, and how to use them in:
`https://apps.fz-juelich.de/jsc/hps/juwels/software-modules.html`

Open a Jupyter notebook with access to the **JUWELS Cluster**. Open a terminal (located in the `launcher` under `other`). Use the above software documentation and the terminal to reply following questions on the eCampus checklist:

a) Which command tells you the software modules available on JUWELS?

b) Which version(s) of the software `Julia` is(are) available?

c) Which version(s) of the profiler tool `Vampir` is(are) available?

d) Which command tells you the software modules loaded on JUWELS?

e) Which software stage is loaded per default?

2: Compiling Parallel Programs
One of the goals of the practical exercises in the HPC lecture is to learn good practises on Research Software Engineering (RSE), including the use of code repositories. This is why we will use GiHub Classroom to provide the assignments. For any RSE project, it is extremely important to keep a clean structure in your software repositories. This will help you and your future collaborators finding the right information. A clean folder structure allows immediately identifying where to find which kind of files. Typical structure of a software repository:

- `README`: a text file explaining any important information about the content of the repository or how to use it.
- `./src`: for source code
- `./include`: for header files (if needed)

- `./script`: for scripts to compile and/or submit jobs
- `./bin`: for binaries (executables)
- `./out`: for output files
- `./err`: for error files
- `./solutions`: in the case of exercises this is the folder into which you should copy your solutions.
- `./tests`: for verification tests. It is a very good practice to include verification tests for each function. The idea is that each function is tested individually, and then in conjunction with others. This tremendously facilitates debugging large projects. In several exercises of the lecture automatic validation tests will be included to facilitate the work of tutors checking correct execution.

Find on eCampus the link to GitHub Classroom and accept the assignment *02_running*. It contains some incomplete code examples in the `src` folder, and a some incomplete Slurm scripts in the folder `script`. Clone the GitHub repo on Jupyter or copy its information to your `home` directory on JUWELS. Follow the instructions. **Do not forget to push your changes to the GitHub Classroom at the end of the exercise**
Important Notes: (1) Do NOT change the name of the given files, or the verification tests will not pass and the exercise will be failed. (2) On the JSC systems, software is compiled on the login node, but executed on the compute nodes (via Slurm), as you will learn in following exercises.

a) **(1pt)**: Open "hi.c" and write therein a simple "hello world" program. Compile it with the GCC compiler and store the output executable "hi" in the folder `bin`.

b) **(2pt)**: Complete the source code "hi_omp.c" as indicated, and compile to an executable "hiomp" using `gcc -fopenmp`.

c) **(2pt)**: Complete the source code "hi_mpi.c" as indicated, and compile to an executable "himpi" using `mpicc`. If the command is not found, load the necessary software (GCC and OpenMPI). Try to compile again.

d) **(2pt)**: Complete the source code "hi_hybrid.c" as indicated, and compile to an executable "hihyb" using `mpicc -fopenmp`.

e) **(2pt)**: Complete the source code "hi_cuda.cu" as indicated, and compile it to an executable "hicuda" using `nvcc`.

3: Batch system documentation

Check the batch system documentation at:
`https://apps.fz-juelich.de/jsc/hps/juwels/batchsystem.html`
Reply following questions on the eCampus checklist:

a) Which are the batch partitions on the JUWELS Cluster?

b) Which are the batch partitions on the JUWELS Booster?

c) What do the following Slurm options mean:

- `--account`
- `--nodes`
- `--ntasks`
- `--ntasks-per-node`
- `--cpus-per-node`

4: Slurm interactive allocation

For this practical exercise, you should copy your solutions into the folder `"solutions"` and give to your output files names starting by the exercise number in a specific format.

- For example, the solution file for exercise number 4.b of the exercise sheet 2 should be called: ex02040b_... The dots can be substituted by an information indicative of the content, e.g. ex02040b_hi_n2_t1.out
- Equivalently, if there would be an exercise sheet number 10, containing 12 exercises with 5 subsections, the output of the very last exercise in these sheet could be called: ex10120e_solution.out

Open an interactive session, allocating 2 node on the developer partition (*devel*) of the JUWELS Cluster:

```
salloc --partition=devel --nodes=2 --account=training2325 --time=00:10:00
```

Note that if you take more than 10 minutes to solve the exercise you'll need to retype the above command to get a new allocation.

a) Use `srun` to run the executable "hi" (from Ex.2.2.a) with 1 task and 1 task per node, and pipe (save) the output to `ex02040a_hi_n1_t1.out` Check the warning message that you receive on the screen.

b) Run now with 2 tasks and 1 task per node and save the output to `ex02040b_hi_n2_t1.out`.

c) Run now "himpi" with 4 tasks and 1 task per node and pipe the output to `ex02040c_himpi_n4_t1.out`. Check the warning message that you receive on the screen.

d) Use now `srun` to run "himpi" with 4 tasks and 2 tasks per node and pipe the output to `ex02040d_himpi_n4_t2.out`

5: Slurm batch mode

The interactive sessions are Ok for small tests, but production runs should be submitted in batch mode, which ensures the resources are well shared and scheduled fairly.
Check the provided scripts under `script`, and solve the TODO assignments for each of the following exercises:

a) Complete `slurm_mpi.sh` to run ”himpi” (from Ex2.2.c) on 2 nodes, 4 tasks, and 2 tasks per node. Run it with sbatch. The output of your run should be saved into the `out` folder as a file called ”ex02050a_out.XXXXXX”, with the ending being the Id-number of your job. Check the output file.

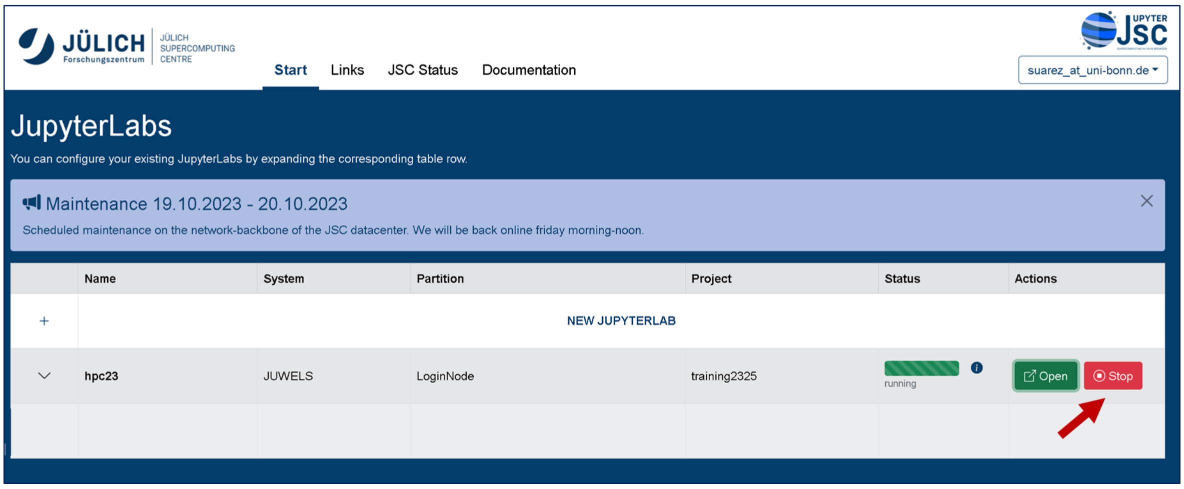
b) Type `squeue --help` to find out how to find information about running jobs. Reply the related question on the eCampus checklist. Rerun then the script and immediately afterwards query the status of your new job with the appropriate command. Get also more information from your job using `scontrol show job <YOURJOBID>`

c) Adapt the `slurm_omp.sh` script to execute the program `hiomp` (from Ex.2.2.b) using 1 node and 8 OpenMP threads. To run an OpenMP job you need to include additional lines to set the number of OpenMP threads (`cpus-per-task`) and define the environmental variable `OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK`. Check the output file.

d) Adapt the `slurm_hyb.sh` script to execute the program `hihyb` (from Ex.2.2.d). Run the hybrid job on 2 nodes, with 1 MPI process per node, and 8 threads per MPI process. Check the output file.

e) Adapt the `slurm_cuda.sh` script to execute the program `hicuda` (from Ex.2.2.e). You need to add a line to determine the number of GPUs to be used (`gres=gpu:<NGPUS>`). Note that this time you also need to change the partition to `develgpus`, because the `devel` partition does not have GPUs. Run the CUDA job on 1 node, using 1 GPU. Check the output file.

Remember to: (1) Push your solved exercises to the GitHub Classroom repository. (2) Close your Jupyter session and stop JupyterLabs (see figure 1).



JupyterLabs

You can configure your existing JupyterLabs by expanding the corresponding table row.

Maintenance 19.10.2023 - 20.10.2023

Scheduled maintenance on the network-backbone of the JSC datacenter. We will be back online friday morning-noon.

	Name	System	Partition	Project	Status	Actions
+	NEW JUPYTERLAB					
	hpc23	JUWELS	LoginNode	training2325	running	Open Stop

Figure 1: Stopping JuppyterLabs