Example scripts

In this section, we provide you a set of examples which will be useful to you to configure scripts depending on your research needs. If you are interested in taking a look at them, they are all available in */share/user_codes/* directory.

Also, this examples are in our repository https://github.com/hpciter/user_codes.git

In order to run the scripts, copy them to your *data* folder.

```
$ tree /share/user_codes/
├──
│   ├── 01-2node-4cpupertask
│   │   ├── 01-2nodes-4cpupertask.sbatch
│   │   └──
│   ├── 02-1node-4cpupertask
│   │   ├── 02-1node-4cpupertask.sbatch
│   │   └──
│   └── 03-1node-6cpupertask-ilk
│       ├── 03-1node-6cpupertask.sbatch
│       └──
├──
│   ├── 00-sandy-nodes.sbatch
│   ├── 01-icelake-nodes.sbatch
│   └──
├──
│   ├──
│   ├──
│   └──
├──
│   └──
├──
└──
```

Single and Multi-core Jobs

Depending on resources consumption, the simulation could need more than one processor. In the bash script, it is possible to specify how many cores are requested to run the job.

If the simulation uses more than one core, you have to add `--cpu-per-task` or `-c` parameter with the number of CPU cores used per task. Without this option, the controller will allocate one core. Remember to specify the tasks number with `--ntasks` or `-n` parameter.

Single core	Multi-core
#SBATCH --ntasks=X where X => 1	#SBATCH --ntasks=X where X => 1 #SBATCH --cpus-per-task=Y where Y > 1

OMP, MPI and Hybrid Jobs

It is possible to find different types of parallelism: OpenMP, OpenMPI or a hybrid solution combining both of them. On one hand, you will use OpenMP for parallelism within a multi-core node.

Since it is a multithreading implementation, an *OMP_NUM_THREADS* variable has to be defined. On the other hand, if the parallelism is between nodes, you will use OpenMPI.

So, in our sbatch script, it will be necessary to specify the number of nodes, the number tasks on each node and each CPU.

OpenMP	**OpenMPI**	**Hybrid**
#SBATCH --cpus-per-tasks=X where X>1 export OMP_NUM_THREADS=X where X>1	#SBATCH --ntasks=X where X => 1 #SBATCH --cpus-per-task=Y where Y>1 #SBATCH --nodes=Z where Z =>2 #SBATCH --ntasks-per-node=W where W>1 #SBATCH --ntasks-per-socket=U where U>1 module load OpenMPI/2.0.2-GCC-6.3.0-2.27	Combine both options

Array Jobs

Using an array, you are able to execute multiple jobs with the same parameters. In your sbatch script, you have to specify the --array option.

Array
#SBATCH --array=1-X where X > 1

GPU Jobs

It is essential to use --gres parameter to reserve a GPU resource and load the CUDA module.

Nvidia A100	Nvida Tesla T4
#SBATCH --gres=gpu:a100:1 module load CUDA/8.0.61	#SBATCH --gres=gpu:t4:1 module load CUDA/8.0.61