Prof. Dr. O. Rheinbach
Dipl.-Math F. Röver
Institut für Numerische Mathematik und Optimierung
TU Bergakademie Freiberg

# 8th Exercise in HPC

**Exercise 1**
Try to learn from `https://urz.tu-freiberg.de/urz/hpc/` how the PBS Queuing system works.

Create a file script.pbs

```
#!/bin/sh
#PBS -N MPI
#PBS -l select=2:ncpus=1:mem=2gb:mpiprocs=1
#PBS -l walltime=00:02:00
#PBS -m abe
#PBS -o pbs_out.txt
#PBS -e pbs_err.txt
module add gnu/gcc/7.1.0 openmpi/gcc/7.1.0/2.1.1
PBS_O_WORKDIR=$HOME
cd $PBS_O_WORKDIR
cat $PBS_NODEFILE
echo
echo $OMP_NUM_THREADS
echo
```

and submit it

```
 qsub -q hpcopt script.pbs
```

Vary the value of `select`, `ncpus`, `mpicprocs` and check in `pbs_out.txt` on which nodes your program would be started (`cat $PBS_NODEFILE`) and what is the value of `$OMP_NUM_THREADS` (`echo $OMP_NUM_THREADS`).

**Exercise 2**
Try to run one of your old pure MPI programs using a PBS script. For this add `mpirun ./a.out` as the last line of the script (without `-np`).

**Exercise 3**
Try to write a first MPI/OpenMP program, but using 2 MPI ranks and only 1 thread per MPI rank (which usually makes no sense) by using

```
 module add gnu/gcc/7.1.0 openmpi/gcc/7.1.0/2.1.1
 mpicc -fopenmp mpi_omp_first.c
```

Create a `run.pbs`

```
#!/bin/bash
#PBS -N MPI
#PBS -l select=2:ncpus=1:mem=2gb:mpiprocs=1
#PBS -l walltime=00:03:00
#PBS -m abe
#PBS -o pbs_out.txt
#PBS -e pbs_err.txt
module add gnu/gcc/7.1.0 openmpi/gcc/7.1.0/2.1.1
PBS_O_WORKDIR=$HOME
cd $PBS_O_WORKDIR
cat $PBS_NODEFILE
echo
echo $OMP_NUM_THREADS
echo
mpirun ./a.out
```

Queue the job by

```
 qsub -q hpcopt run.pbs
```

and after that inspect your output in `pbs_out.txt`.

**Exercise 4**
Try to write a first MPI/OpenMP program, but using only 12 threads (one node) per MPI rank, i.e., using

```
#PBS -l select=2:ncpus=12:mem=2gb:mpiprocs=1
```

Make sure that MPI functions are only called in `master` regions. Here, on each node one MPI rank is executed each spawning 12 OpenMP threads, i.e., we have a total of 2 MPI ranks and 24 threads.
Also try

```
#PBS -l select=2:ncpus=2:mem=2gb:mpiprocs=1
```

which should result in 2 MPI ranks each spawning 2 OpenMP threads running on a single node.

Remark: Also openmpi/intel should be compiled with threading support (level 2) now:

```
module add intel/compiler/16.0_u2 openmpi/intel/16.0_u2/2.1.1
```

Explanation: In

```
#PBS -l select=2:ncpus=12:mpiprocs=12
```

*select* is the number of chunks. For every chunk ncpus is the number of cores requested and (also for every chunk) *mpiprocs* MPI processes are started. Here, a total of 24 cores are requested and 24 MPI processes are run.