Access to the infrastructure

To access the cluster enter the following command:

ssh student<number>@ gw.ipp.acad.bg

Enter your account name (student<number>) and your account password.

cp -r /opt/exp_software/documentation/Training/* /home/student<number>/

In our first example we will submit a simple job which will print **Hello World** in its output. Navigate to **/home/student<number>/Examples/Hello** and enter the following command.

cat hello.sh

```
#!/bin/bash
#PBS -q lifesci

echo "Hello World!"
```

To submit the shell script to the cluster use the following command:

qsub hello.sh

This command will print the following output:

<Job_ID>.torq.hpcg

Your job will be completed almost instantly and the standard output and the standard error output will be in **hello.sh.o<Job_ID>** and **hello.sh.e<Job_ID>** respectively. Enter the following commands.

cat hello.sh.o<Job_ID>

cat hello.sh.e<Job_ID>

Our second example will print the resource granted to your job. Navigate to **/home/student<number>/Examples/Resources**. Print the shell script **res.sh** (use the command **cat res.sh**).

```
#!/bin/bash

#PBS -q lifesci
#PBS -l nodes=2:ppn=4

cat $PBS_NODEFILE
```

The #PBS -l nodes=2:ppn=4 denotes that our job will run on 2 nodes and will use 4 logical cores from each node. This can be modified from the by modifying the **res.sh** or by using the **-l** option of the **qsub** command. Submit the script using following command:

qsub -l nodes=1:ppn=16 res.sh

Open the output:

cat res.sh.o<Job_ID>

The third example will familiarize you with the commands used to monitor and terminate your job. Navigate to **/home/student<number>/Examples/Loop**. Here you will see the script **loop.sh**:

cat loop.sh

```
#!/bin/bash

#PBS -q lifesci
#PBS -l nodes=1:ppn=1

while(true)
do
    a=1+1
done
```

Now let's submit **loop.sh**. Again we will use the **qsub** command, but we will specify an e-mail on which we will receive status updates on our job:

qsub -M <your_e_mail_address> -m abe loop.sh

Such options to qsub can be specified on command line, but alternatively they can be specified in the shell script.

Use the following commands to view the information for your job:

qstat <Job_ID>

qstat -f <Job_ID>

qstat -n <Job_ID>

The last command will print a list of the nodes and cores granted to your job. The first entry in the list (**wn<number>.hpcg**) is the host node. To navigate to it:

ssh wn<number>

Notice that you don't have to specify password. Once you are logged to the execution node enter the **top** command. You will see a list of all the running processes sorted by their CPU usage. You can notice

that your process is on top. To close the **top** application press **q**. Now return to **wn02** by entering the **exit** command.

You can also get information about all jobs of a given user using the **qstat** command. Enter the following command:

qstat -u <user_name>

Using **qstat** you can also get information about the queues. Enter the following commands and consider their ouputs:

qstat -q will print information about all the queues.

qstat -q lifesci will print the information about the **lifesci** queue.

qstat -Q will print more detailed information about all the queues.

qstat -Q lifesci

qstat -Qf

qstat -Qf lifesci

Now it is time to terminate our job:

qdel <Job_ID>

If you don't remember your **Job_ID** you can either get a list of all your current jobs using the **qstat -u <user_name>** command or use the command **qdel ALL** which will try to delete all jobs, but will delete only yours and will produce a lot of error messages because (caused by your unauthorized deletion requests).

Before we continue with **MPI** we must choose an MPI compiler. Enter the following command:

mpi-selector-menu

From the list of options choose **openmpi_gcc-1.3.3**. Select this setting to be **per-user** (**u**) and overwrite any existing settings (**y**). Next exit the **mpi-selector-menu** (press **Q**). Close the terminal and reopen it again (you will have to log on to **gw.ipp.acad.bg** using ssh again).

Next navigate to **/home/student<number>/Examples/MPI**. Before we continue with using

In the directory there are two files – **helloMPI.c** and **helloMPI.sh**. View them by using the following commands:

cat helloMPI.c

Enter the following command:

mpicc helloMPI.c -o helloMPI

This will build your program. To run it on the login node (only useful during testing):

mpirun -np 10 ./helloMPI

This will run 10 instances of your program. Alternatively you can submit your job using the **qsub** command. Print the shell script using **cat helloMPI.sh**

Submit your job using the following command:

qsub helloMPI.sh

Check your job's status or enter **ls** to check if your results are present.

cat helloMPI.sh.o<Job_ID>

Next navigate to **/home/student<number>/Examples/OMP**. Again, there are two files in the directory – **helloOMP.c** and **helloOMP.sh**. View them using

cat helloOMP.c

cat helloOMP.sh

Build your program using the command

gcc -fopenmp helloOMP.c -o helloOMP

To run it on the login node (only useful during testing), enter

./helloOMP <number_of_threads>

To submit it enter **qsub helloOMP.sh**

Navigate to **/home/student<number>/Examples/MPInOMP**. There are two files in the directory – **helloMPInOMP.c** and **helloMPInOMP.sh**. Print each of them. Build the program using the following command:

mpicc -fopenmp helloMPInOMP.c -o helloMPInOMP

To run in on your machine use:

mpirun -np 10 ./helloMPInOMP 3

This will run the 10 instances of the application with 3 threads each.

Print the **helloMPInOMP.sh**. Submit it using the **qsub** command and review its output.