Introduction to Parallel Computing — Matthias K. Gobbert
Wintersemester 2011/2012 — Universität Kassel
Homework 2 — due on Tuesday, December 13, 2011

Each programming problem needs to start out with text explaining what you did; computer code, tables, or plots should *never* be the first page of any problem! This text should verbally introduce, explain, and interpret all other material including tables, plots, and computer code, in this order. Include relevant portions of code as part of your discussion for maximum clarity.

1. [4 points.] Write a serial "Hello, world!" program, compile it, and run it on the head node of the cluster in the IT Servicezentrum at the Uni Kassel. The point of this problem is to ensure that you can operate effectively under Linux on this cluster, can create and edit a file, can compile, and run in serial. I will demonstrate this in the lab. Ask questions as needed.

2. [6 points.] You should read Chapter 3 of Pacheco as reading assignment. As preparation for the excercises, recall Homework 1, Problem 3 (b) "Obtain the source code ...." on downloading the source code for all examples in the textbook. Transfer that downloaded file to the cluster in the IT Servicezentrum and expand it, so that you can actually access each file. Ask questions as needed. I post this problem explicitly here so that we can ensure that you know how to tar and compress/expand 'tar balls.'

   (a) Chapter 3, Exercise 1. in Pacheco.

   (b) Chapter 3, Exercise 2. in Pacheco.

   (c) Chapter 3, Exercise 3. in Pacheco.

   Just report what you did here and discuss to the questions in the exercises. Include listings of your codes in your report.

3. [6 points.] Download the entry for this lab in the updated detailed schedule the qsub submission script `mpich2_pgi.qsub` and the parallel "Hello, world!" program `hello_parallel.c`.

   (a) Before using them, create a suitable directory structure, such as a directory for this course (e.g., `math627`) and sub-directories for homeworks and labs (e.g., `labs`), with more numbered sub-directories (e.g., `lab01`). This is an excercise in learning how to operate the operating system and shell commands effectively on the cluster.

   (b) Now place the downloaded files here. Compile the program using `mpicc`; to control the name of the resulting output file (the name of the executable), use the `-o` option, so use the compile command

   ```
   mpicc hello_parallel.c -o hello_parallel
   ```

   The name of the executable is important, because the supplied qsub submission script refers to this executable name under `mpijob`.

(c) Try running the code now with

```
qsub mpich2_pgi.qsub
```

What happens? Notice that stdout is retained in the file qsub.out and stderr is retained in the file qsub.err. You should now look through both qsub.err and qsub.out. In one of them, you should find, among other things, an error message that instructs you to create a file `.mpd.conf` (notice the leading period "." in its name) containing the single line `MPD_SECRETWORD=` with a 'secret word' of your choosing after the equal sign; follow the instructions given, including the change of permissions to 600, and try running again.

(d) Experiment with different choices of values `X` and `Y` in the line

```
#PBS -l nodes=X:Opteron2435:ppn=Y
```

near the beginning of the submission script. Which values for them are legal on this cluster (namely specifically on the nodes with the AMD Opteron 2435 processors)? Report your experiences.

4. [4 points.] Chapter 3, Programming Assignment 1. in Pacheco. State clearly what task this assignment assigns, that is, state the programming problem clearly. Discuss clearly how you expect the program to behave. How can you be sure that your code is actually correct? Include a printout / printouts of your code(s) in your report.