Informatik und Angewandte Kognitionswissenschaft
Lehrstuhl für Hochleistungsrechnen

Rainer Schlönvoigt
Thomas Fogal
Prof. Dr. Jens Krüger

High-Performance Computing

http://hpc.uni-duisburg-essen.de/teaching/wt2013/pp-nbody.html

Exercise 0 (10 Points)

All assignments are to be uploaded to Moodle. Assignments are due at midnight on the due date. No late assignments will be accepted.

All assignments must include a `Makefile` for compiling your assignments. The assignment specification should include what the default target of your makefile should be. Assignments which do not compile will receive 0 points. Sometimes, we provide sample inputs and outputs; assignments which do not satisfy these test inputs will receive very few points.

Please do not include additional output other than what was requested by the assignment details. *Hint: if you want more debugging output, use a 'debug' flag in your program's arguments and have it only print when that flag is active.*

Your assignment will be graded on the `duecray.uni-due.de` supercomputer. It does not matter if your program runs correctly on another machine; it must run correctly on `duecray` to receive credit.

1 Assignment (10 Points)

In this assignment you will run a simple job on the cluster.

1.1 PBS

You can ssh to the cluster using the hostname '**duecray.uni-due.de**'. Job submission is via the PBS scheduler. Basically, you must write a shell script to run your program, which includes some additional metadata to PBS detailing job parameters (such as number of processors to use). PBS uses these parameters to decide *when* to run your job, which is not necessarily immediate. A very simple job submission script is given in Listing 1:

```
#PBS -N tjf_loves_hpc
#PBS -l mppwidth=4
#PBS -l mppnppn=2
#PBS -l mppdepth=1
#PBS -l walltime=0:30:00
#PBS -o stdout
#PBS -e stderr

cd $PBS_O_WORKDIR
aprun -n 4 /homes/adc701q/test-assignments/hw
exit 0
```

Listing 1: Simple PBS job submission script.

This creates a job named 'tjf_loves_hpc' in the queue. Names do not have much purpose, other than helping you to identify your jobs later, when you query the system wondering if they have run yet.

1.2 Hello World

Write a simple MPI-based 'Hello World' program. Your program should output:

```
  Hello, my name is %s on process %u
```

where '%s' should be replaced by your actual name, and '%u' should be replaced by the rank of the MPI process performing the print.

Run this program on 4 processors and save the output.

1.2.1 Example MPI Program

As we have not yet formally gone over MPI, you may find this C99 example useful.

```
#include <stdio.h>
#include <mpi.h>

int main(int argc, char* argv[]) {
  MPI_Init(&argc, &argv);
  int rank = -1;
  if(MPI_Comm_rank(MPI_COMM_WORLD, &rank) != MPI_SUCCESS) {
    fprintf(stderr, "grabbing rank failed!\n");
    MPI_Abort(MPI_COMM_WORLD, -1);
  }
  printf("Hello, my name is Tom Fogal on process %u\n", (unsigned)rank);
  MPI_Finalize();
}
```

1.2.2 Example Makefile

We will go over how to write simple makefiles in class. If you'd like to get started earlier, you can start with Listing 2 and modify to suit your program.

```
hw: hello.c
  CC $^ -o $@
```

Listing 2: Sample Makefile

Note that the indentation before `CC` *must* be a tab (spaces will not work).

As we mentioned in class, we will use the GNU compilers for this course. However, `duecray`'s default environment sets up the PGI compiler suite. To change this, you need to run:

```
module swap PrgEnv-pgi PrgEnv-gnu
```

before you compile your programs. Note that, for complicated reasons, `module` commands cannot be put into makefiles; you must execute this *manually* on every login!

1.3 Tarball Layout

Your tarball submission should include:

- Your 'Hello World' program source, in a file named `hello` with an extension appropriate to your language of choice.

- The PBS script you used for submission, in a file named `hworld.qsub`.

- A file containing the standard output log of your run, named `stdout` or `stdout.txt`.

- A file containing the standard error log of your run, named `stderr` or `stderr.txt`. Note that this file should be empty!

These should extract to a directory named 'hello-world'. A sample tarball listing is given in Listing 3:

```
$ tar ztvf tom.fogal-hpc0.tar.gz
drwxr-xr-x adc701q/ing-ing1  0 2013-10-22 11:49 hello-world/
-rw-r--r-- adc701q/ing-ing1 355 2013-10-22 11:41 hello-world/hello.c
-rw-r--r-- adc701q/ing-ing1 205 2013-10-22 11:41 hello-world/hworld.qsub
-rw------- adc701q/ing-ing1 422 2013-10-22 11:41 hello-world/stdout
-rw------- adc701q/ing-ing1   0 2013-10-22 11:41 hello-world/stderr
-rw-r--r-- adc701q/ing-ing1  69 2013-10-22 11:49 hello-world/makefile
```

Listing 3: Tarball listing of correctly-organized submission.

2 *sine qua non*

Pack your file into a single compressed tarball for submission. The tarball should extract all files into a single subdirectory. Upload your tarball to Moodle.