15-418 Spring 2011
Tutorial on using `blacklight` and `pople`

1 About the Machines

We will be using two parallel machines that support OpenMP this semester: `blacklight` and `pople`, both at the Pittsburgh Supercomputing Center (PSC). Details about both these machines can be found in the following web sites:

```
http://www.psc.edu/machines/sgi/uv/blacklight.php
http://www.psc.edu/machines/sgi/altix/pople.php
```

2 Connecting to the Machines

2.1 Logging In

You should use `ssh` to login to `blacklight` or `pople`. You can use the following command from your terminal:

```
$ ssh username@pople.psc.edu
$ ssh username@blacklight.psc.teragrid.org
```

This will give you access to the interactive machines in the respective clusters where you can test and debug your program.

2.2 Transferring files

You will have a local home directory on both `blacklight` and `pople`. In order to transfer files to these machines, you should use `scp`. You can use the following command to transfer files from your machine:

```
$ scp localfile username@pople.psc.edu:remotedir/
```

If you are a windows user, you can use `WinSCP` (available at `http://winscp.net`). This tool has a GUI to transfer files to the remote machines.

3 Compiling Your Programs

The OpenMP standard is supported on both `blacklight` and `pople`. Hence you don't need to modify your source code to run on either platform. However, you do need to compile your code on these machines before running them. The following command allows you to compile your program on either of the machines:

```
$ icc -opemmp -O prog.c -o prog
```

4 Running Your Programs

4.1 Testing/Debugging

In both `blacklight` and `pople`, you can directly run your program on the frontend machines (the ones you login to). You can do this by simply typing the program name from the command line with appropriate arguments (like you will do in any standard unix machine). This is a very useful way to test and debug your program and also get some rough performance numbers. Note that there is an upper limit on the number of cores you can use in the frontend machines.

4.2 Running Batch Jobs

However, to get proper performance measurement on either machine, you need to submit your program to the batch queueing systems. When your program reaches the head of the batch queue, a dedicated portion of the machine (corresponding to the number of processors that you requested) will run your program exclusively until it either completes or runs out of time (more on the latter below).

To submit these batch jobs, you will use the Portable Batch Scheduler (PBS) system . To submit a job using PBS, you need to create a job script. The following webpages contain information about how to create the job script:

```
http://www.psc.edu/machines/sgi/uv/blacklight.php#running
http://www.psc.edu/machines/sgi/altix/pople.php#running
```

Job scripts are regular shell scripts with additional PBS directives at the top, immediately after the specification of the shell to use. Notice the PBS directives at the top of the script. The above-mentioned web pages provide a detailed explanation of what each directive means.

It is important to have a reasonable estimate for the walltime directive. This time is a hard limit on how long your job is permitted to run. Once the time elapses, your job will be killed if it has already not terminated. Therefore you should surely not underestimate the runtime of your job. On the other hand, shorter jobs are given a higher priority. So grossly over-estimating the runtime will result in slower service.

5 Debugging

Both `blacklight` and `pople` have the `gdb` (GNU Debugger) tool available. You can use it to debug your program. Man pages for the debugger is available in both these machines. If you have trouble using them, you can always contact one of the TAs or Prof. Mowry for help.