Due Date: Friday, January 20, 2015 - 5:00 pm
Your name: Devsirme Janissaries

INTRODUCTION

The purpose of this homework is to get you some exposure to git, python, and working on a cluster. The homework is composed of three parts. In the first part you will create a git archive that you will use for the remainder of the class. The second task will involve modifying a python script that parses a webserver access file. The final part, only for those taking the class for more than two units, involves modifying a python script that submits jobs to the grid engine.

PROBLEM 1: GIT [5 PTS]

Git is a revision control system that can be very useful in code development. For the remaining homework assignments you will be required to store your assignments in a repository that you will create during this assignment. Begin by logging into cees-rcf.stanford.edu.

A git archive has already been created for you. This is a protected archive that only you, the TA, and I will be able to access. Your first step is to `clone` the archive. I would suggest first changing into your work directory on `/data/cees/gp257/username`, once there, enter the git command:

```
git clone git@zapad.Stanford.EDU:GP257/lab_USERNAME  USERNAME
```

where `USERNAME` is your Stanford ID. This command will copy your Git archive from the CEES gitlab server to the directory `USERNAME`.

After cloning the archive you should see lab1 in your `USERNAME` directory. In that directory you will find:

- This TEXfile and the associated PDF file
- A makefile to compile and run various parts of this lab

- A python script that you need to edit that parses the webserver logfile (part 2)
- The python script that we worked on in class that could run parallel jobs on a single machine (part 3)
- The source code to estimate pi
- An example script for submitting to the grid engine

Next lets download a logFile used in the next part using (this should be on one line)

```
curl -o logFile
http://sepwww.stanford.edu/data/media/public/sep/bob/download/logFile
```

This isn't really source code so lets tell git to not track changes to this file. We can do this by editing our `.gitignore`. The `.gitignore` using the regex syntax used in the second part of this lab. To excluded this file we will simply use the file name. Using your favorite text editor add the line `logFile` to your `.gitignore` file.

PROBLEM 2: PARSING A LOG FILE [5 PTS]

In this section you will modify a working Python code to be more memory efficient. Currently the code reads in the log file, storing the access information in a dictionary. Each element of the dictionary is referenced by its web address, and contains an array of all of the requests for the given page. These requests are stored within the class structure page access. When executed the first argument of the command line is used to search the database of web pages. All pages that match the command line string are printed.

The memory requirements of using this approach can be significant. Each reference to each page is stored in memory. For large logfiles this often becomes cost prohibitive. Your assignment is to modify the script so that only web addresses that fit the string that the user requests to search for are added to the dictionary. In addition I want you to

organize the search results. Printing out all of the lines associated with a given file together. You can run the current script by typing `make script_test` in your working directory. (check out the `Makefile` for the underlying workflow).

PROBLEM 3: SUBMITTING JOBS TO THE GRID ENGINE [10 PTS]

The final part of the homework is only required if you are doing the class for more than two units. Begin by compiling the executable by typing `make piCalc.x`. Look at `sub.sh`. This is an example script used to submit jobs to the cluster. This script is written to run `piCalc.x` program on one core of the cluster. We can submit the job to the cluster by typing `qsub sub.sh`. You can then run `qstat` to see all of the jobs running on the cluster. You can see just your job by running `qstat |grep USERNAME |grep piCalc.x` where USERNAME is your username. The result should be six fields. The job ID, the name we gave our script piCalc.x, your username, how long it has been running, the job's status (R-running, C-completed, W-waiting to run), and the queue the job has been assigned to (in this case default).

The goal of this portion of the assignment is to build upon the script for running parallel jobs on a single node. Your assignment is to

- Derive a new class, `gridParallelJob` from `runParallelJob`. You will need the following functions to work with grid engine. You will need to override the `startJob` and `checkJobRunning` routines. Hints:
  - You will need build a submit script similar to the one included in the lab for each job and then execute `qsub`.
  - You will need to parse the `qstat` command to check the job status
  - You should use the process ids to keep track of the jobs.
  - You will need to edit where `piCalc` writes stdout so it is in your data directory.

- You should inherit the `piParallelGrid` class from your `gridParallelJob`. This class should be nearly identical to the original `piParallel` class.

SUBMITTING THE ASSIGNMENT

Remove the executable (`make clean`). Then add all the files you have created and want to share with `git add`, and commit the final version of your lab with `git commit -m "Finished lab1"`. Check that if all updates are committed by looking at the output of `git status`. Finally pushed those changes onto the server with `git push`.