Lab Orientation – ENG EC527

Authors: Dan Collins (TF 2010, 2011), **Robert Munafo** (TF 2018-2021)

Welcome to the HPC Lab in PHO 305/307 (and elsewhere)!

First, why 305/307? For consistency. This is a course about creating/testing/measuring/evaluating/optimizing programs on real machines and this means dealing with much noise, performance variation, etc. So to properly evaluate your evaluations, we must have a stable baseline – for which 305/307 is used. You can still develop code wherever you like, but don't leave it to the last minute to verify that it works on the lab machines.

As you may find out, these are not always the most stable machines, so I advise you to bring a fair amount of patience and **SAVE OFTEN WITH BACKUPS**. The machines in PHO 305 and PHO 307 are part of the "Eng-GRID" also called just "GRID." Later on in the semester we'll start using machines on the SCC (also called MGHPCC) -- more info on that is in Assignment 7.

Remote Access

1. VPN. To use the machines remotely requires being connected to the BU's part of the internet. If you are in a BU building (including WiFi in the dorms) this is probably already true. If you are outside of BU, you can use the BU "VPN" (virtual private network) connection utility.

When this document was written, VPN instructions were available at
www.bu.edu/tech/services/cccs/remote/vpn/
If that link does not work, do a Google search for "VPN bu.edu".

You *MUST* use the Boston University VPN, not some other VPN. You *MUST* use your own BU login to connect to the VPN, not somebody else's.

2. Active Directory / WebNew. Some students who have never used file servers at BU may need to do the "WebNew" application first. This is an online form to request that the IT department give you an Active Directory (file server) account linked to your Kerberos (normal BU login) account. You won't notice any change to other things you already do with a BU login, but you gain access to some new services.

Even if you have been at BU for many years you may have do WebNew for the ENG Grid to work. There is no harm in using the WebNew tool multiple times so please complete the process again if you are unsure.

3. SSH. Next you need a way to make an ssh connection to `eng-grid.bu.edu` Instructions for this should be at `www.bu.edu/engit/knowledge-base/grid/coursework/` or if that link does not work, search for "Eng GRID ssh connect bu.edu". The SSH username and password are the same as your BU email ID and password.

4. Actual Lab Machine. After the first SSH login you are only on a "gateway" machine. It will say "**`tl;dr: Run "qlogin" right now.`**" Do as it suggests, and use the same password. Then use the `hostname` command to find out what machine it logged you in to. This should be "`signals`N" or "`vlsi`N" where N is a number.

If it is slow or prints a message like "*timeout expired while waiting on socket*" or "*Your qlogin request could not be scheduled*", then try a direct SSH instead. Randomly pick a number from 10 to 60 and do an SSH command like:

```
ssh -X vlsi27
```

If this works, then you can proceed and get work done. If the random machine you picked is too busy (using "`w ; who`" commands as described below) log out and pick a different one.

5. Assignment Files. Now you need to get files from Blackboard into the Lab machine. File transfer instructions are in:

`www.bu.edu/engit/knowledge-base/grid/instructions/`
`www.bu.edu/engit/knowledge-base/mountingengnas/`

In-Person Access

To use the machines in person, make sure that you have room access to PHO 305 and 307. Go to `www.bu.edu/eng/zaius` and log in with your BU email ID and password, select "Apply for Access" on the left, put "*307*" into the search (lower right), select the item "PHO 307 STUDENT", in the reason for access box put "*I am a student in EC527*" and then hit the button "Submit this request". Repeat for room 305.

Before actually going to the room in person, you probably should verify that you can log in to the machines, using the remote instructions above. If it doesn't work, it is most likely because of the "WebNew" step.

When you use these computers in person, just as in the libraries and Ingalls, you can use a USB "thumb drive" to transfer files. If you have files in your Active Directory ("ad"), for example, from using the computers in a library or the Ingalls Resource Center, it should be visible after you log in ("home" folder right below "Computer" icon in upper-left of desktop).

To get a terminal window (for typing the `gcc` command and lots of other steps in the labs): Applications menu > System Tools > Terminal.

If you have some favorite Linux programming / debugging environment (`vim`, `emacs`, or whatever) try it on a simple "Hello World" program (at least) to confirm things work.

Fixing problems

Sometimes you may encounter problems on a lab machine -- for example, MATLAB or LibreOffice sometimes gives an error or won't start up. We have found that these problems are often random and depend on what machine you are connected to.

The gateway "`eng-grid.bu.edu`" is not a single machine, it randomly selects one of two gateways. Similarly, when you do a "`qlogin`" it selects a "vlsi" or "signals" machine at random (and usually one that's idle).

So, if you're having a problem, definitely let a TA know, but also please try these steps:

- Disconnect back to the first machine (where it said "`tl;dr: Run qlogin right now`", and then do `qlogin` again.
- If that doesn't work, try doing SSH directly to a randomly-chosen machine as described above.
- If that doesn't work, disconnect completely and repeat the original ssh command to `eng-grid.bu.edu`
- If that doesn't work, quickly disable your WiFi, re-enable and reconnect to the VPN, then connect to the grid again.

Doing the Labs

- These are weekly labs. Most of the work in this course is in the programming assignments and final project. There will be programming assignments almost every week, so plan accordingly!
- Due dates: Labs are due electronically about one week after they are assigned.
- The due date and time this semester will *usually* be Monday at 23:59. But always confirm the due date/time by looking on Blackboard in Course Information > Schedule-Calendar.
- Getting help. This is a "mezzanine" (upper level) class with *little TA/TF support*. We will try to fix system

issues promptly, but in general you are expected to work independently. The TA/TF will *not* look at a non-working program and figure out what might be wrong.

- △ Please note that since the labs involve interaction with architecture and system (whole point of the course!) this requires some programming maturity. As the syllabus (given on first day of class) states, the prerequisites include "*Computer Organization (EC413 of equivalent)*" and ability "*to learn new programming tools from professional documentation*".
- ⇒ There is a document in Blackboard (in the same area as this document) called "`Debugging your program.txt`". Use its advice if things are not working.
- Timings *must be taken from lab machines*. This is a course in performance, which means measurement. To keep this consistent, you must collect your performance data (run times, etc.) by running your programs on the machines specified. You can still develop code wherever you like, but don't leave it to the last minute to verify that it works on the lab machines.

Lab groups

- You may work with at most one other person in a lab group . You should submit one write-up per group. Both of your names should be on the submission file. You should *both* submit the file to Blackboard.
- If you are working by yourself, you may (mutually) cooperate with at most one other person. In that case, you will have separate write-ups, but you should have the cooperating classmate's name at the top of your write-up.

Submitting Labs

Again, this class is given very limited TA/TF support, so we need to be efficient. Please --

- Use Blackboard (`learn.bu.edu`) to upload your completed assignments.
- Include all code files (`.c`, `.h`, etc.) files and raw output files (`.csv`, `.xls`). Code should have detailed comments so that anybody reading it can quickly understand what is going on.
- Answering questions. Many assignments have questions interspersed with the coding. Some questions are trivial and just meant for guidance, but most require some thought and should be answered with complete sentences (or a short paragraph) plus references to evidence (e.g., graphs) that you have generated and embedded in your report.
- In your report, include a write-up describing what you did, any problems you had, and interpretations and explanations of the data you recorded. In most cases, your data should be in graphs in the write-up. Also, include a description of each of the program files you have included. The write-up should be a single PDF file with the graphs and tables embedded.
- **Archive all of your files into a .zip file named <last_name>_<first_name>_lab<x>.zip before uploading.** Do *not* use `.tar`, `.rar`, `.7z`, `.bz2` or any other archive format the cool kids are using these days. The graders do not have access to all decompression utilities to extract these, and will probably not notice the problem until past the deadline.

Other Lab Pointers

- Running simulations will sometimes take a few hours or more. Take this into consideration when deciding when to start labs. But especially, make sure that they do not take so long in the first place! Do this by taking care with what you are simulating. ***Be sure to understand your code thoroughly and to debug and test it thoroughly before you start your runs.***
- Design and consider experiments before you invest much time in a run. In any experimental science, there should be some amount of exploration before you put serious work/effort/time into an experiment.
- In this course it is very rare that the best method is to run a single script. Most often the experiments take planning and interaction, the latter as you focus on particular parameters and their ranges.
- If you are working in person in PHO 305/307, you are ***not*** allowed to leave a workstation unattended,

even if you have a long simulation running. However you ***can*** connect to the GRID remotely (as described earlier) and start something running. You might be on the same machine with someone else, but you're not taking up the chair/screen/keyboard. Be aware however that students in the room sometimes hard-reboot workstations, particularly if it cannot be used because it is locked (someone else logged in, then walked away).

- Other classes use the labs. If you are physically in the room, be sure to check the schedule posted on or next to the door, to be aware of when you may be asked to leave the room.
- Getting stuck? Some questions may be very difficult, but also skippable. If you get stuck anywhere for too long, you might want to bypass and return later.
- Need to skip a section? Not understanding a whole lab is a big deal. But don't get too thrown if you occasionally need to skip a section altogether.

Connecting to the lab machines (Eng GRID) Remotely

See the "Remote Access" section at the very beginning of this document.

Basic Terminal Commands in UNIX/LINUX

The supported environment for this course is BU Linux. Your programs must compile and run in this environment (for check-off by the TAs). For those of you who are not familiar with a UNIX or Linux terminal (command line), here are some basic commands. The $ represents the prompt; you do not type it.

$ **mv <file> <dest>**
This will move the specified file to the specified destination. If the destination is a folder, the file will have the same name in the new directory. If the destination is a filename in a folder, the file will be renamed to the specified filename in the new folder. If the destination is only a filename, this function will rename the file to the new filename in the same folder.

$ **cp <file> <dest>**
This behaves in the same way as mv, except the file is copied, not moved.

$ **cd <dir>**
This changes the current directory (folder) of the terminal. The destination can be a relative folder (ex. "images") or an absolute folder (ex. "/home/<user>/images"). Specifying the destination as "~" will change the directory to your home folder. Specifying the destination as "../" will change the directory to your current directory's parent folder.

$ **mkdir <name>**
This command creates a new folder.

$ **htop**
This command starts an interactive program that displays the most resource intensive programs currently running on the workstation.

$ **w ; who**
This is two commands (`w` and `who`) that print mostly the same information: who is connected to the machine, and a command they are running. This is a quicker way to find out if the machine is busy, to decide if you should maybe try a different machine.

$ **ps aux**
This command prints all the running processes and their PID

`$ kill <pid>`
This command kills the process with the specified PID

`$ killall -I <name>`
This command kills all instances of the specified process name.

Pressing CTRL+Z will suspend the running process in your current terminal window, allowing you to return to the shell prompt. If you have done this, you may then use the command **`fg`** to continue the suspended process where it left off.

Graphing Your Data

For many of the assignments you will be asked to try setting different values for a parameter, then trying this setting for different memory sizes. The quickest way to batch run these combinations is usually to modularize your program in a function, then by calling the function with various parameters using double for each loop.

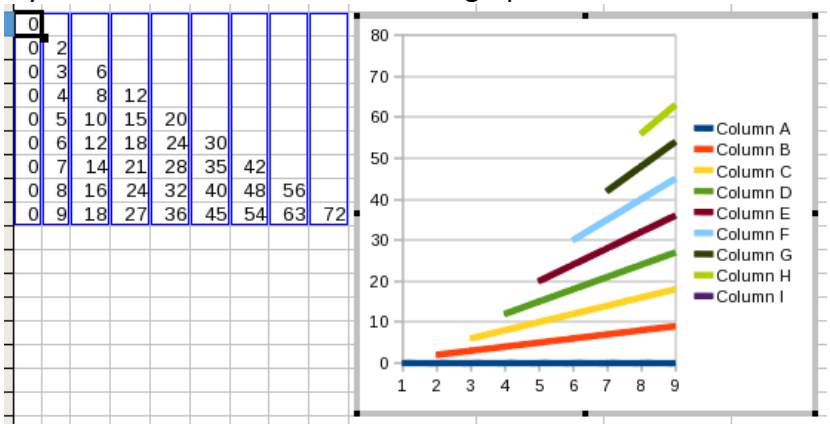
The easiest way to plot your data is to format your program's output in a CSV format, save it to a file, and import it to either OpenOffice Calc or LibreOffice Calc (two open source alternatives to Excel) or MATLAB.

There are two demo functions showing how to quickly format your output inin the first assignment, in `plotting.c`. Note that MATLAB does not like asymmetrical CSV files, so you need to pad your CSV file with 'NaN' values.

You can run your compiled program as:

`$ ./plotting | tee plotting.csv`

Where `plotting` is the program name and `plotting.csv` is the file where the output is saved.
You can open OpenOffice calc under the Start Menu's 'office' submenu. From there, File>Open the csv file. It will prompt you to confirm the import format. After doing so, highlight the table, then open the graphing tool. This tool is very much like excel to fine tune the graph. The result will look something like this:



If you are familiar with MATLAB, you can use it for plotting. Launch it from the terminal with

`$ matlab &`

Please allow a few minutes for it to start up. Once the program is started, navigate to and double click your csv file in the left explorer pane. It will ask you to confirm the import format and variable name for the matrix. You can then plot the matrix using either plot(var) or surf(var). The result should look like one of these:

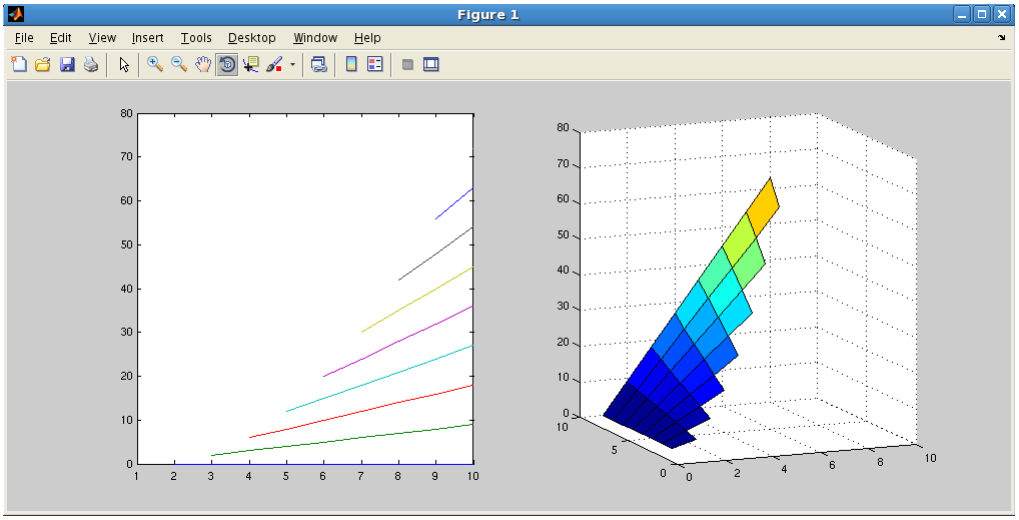


Figure 1
File Edit View Insert Tools Desktop Window Help
80
70
60
50
40
30
20
10
0
1 2 3 4 5 6 7 8 9 10
80
70
60
50
40
30
20
10
0
10
5
0
0 2 4 6 8 10