Parallel Programming Pre-Assignment

Setting up the Software Environment

Authors: B. Wilkinson and C. Ferner. Modification date: Aug 21, 2014
(Minor correction Aug 27, 2014.)

Software

The purpose of this "pre-assignment" is to set up the software environment on your computer that will be used in subsequent assignments.[1] You will not be able to do the subsequent assignments as written without this environment. Hence it is essential that you complete it by the posted deadline. ***Note the posted date to report issues that stop you from progressing. This is a very important to enable any issues to be resolved quickly.***

In the course assignments, various parallel programming tools and programs will be tried out on your own computer (as well as on remote servers). Some of the software requires a C compiler and an MPI message-passing environment. Using an IDE such as Eclipse is an advantage. As a convenience, a virtual machine with everything you need has been created as an .ova file that can be imported into VirtualBox.[2] Virtualbox is a virtualization software that enables "guest" operating systems to be installed on the underlying "host" OS without disturbing the host. It will install on all host platforms (Windws, Mac, and Linux). Our guest operating system will be the Linux distribution Ubuntu.

Installing everything from source. If you do not want to use the provided virtual machine (.ova file), and want to create your own virtual machine or install the software onto an existing Linux distribution, see instructions on course home page under assignment "Pre-assignment".

Part 1 Installing VirtualBox

You will have to first install Virtual box and then import the .ova file. Go to:

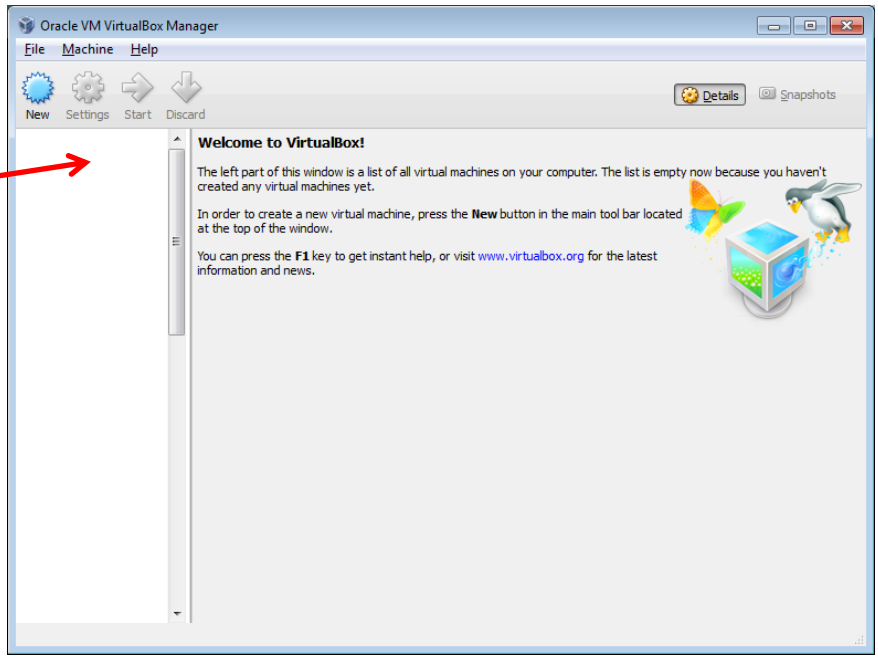
https://www.virtualbox.org/wiki/Downloads

and download VirtualBox to suit your computer[3] and run the installation package (double-clicking the downloaded executable file). Our tested installation uses VirtualBox virtual machine version 4.3.12.

[1] You can use a lab computer is desired.
[2] and probably other virtualization products as it uses the industry standard Open Virtualization Format (OVF)
[3] It does not matter whether you have a 64 bit or 32 bit platform although nowadays one would usually be running a 64 bit processor/OS. VirtualBox supports 32-bit hosts with 64-bit guest OS's.

The package installs in **C:\Program files\Oracle\VirtualBox** by default on Windows.

Starting VirtualBox for the first time, one gets:



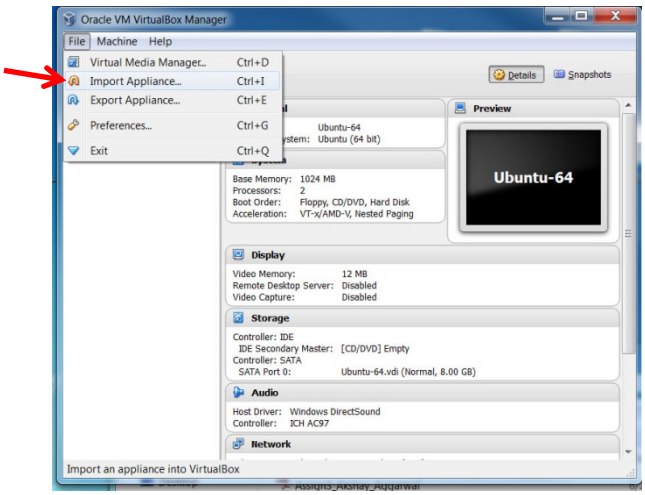
Blank, no OS's installed yet

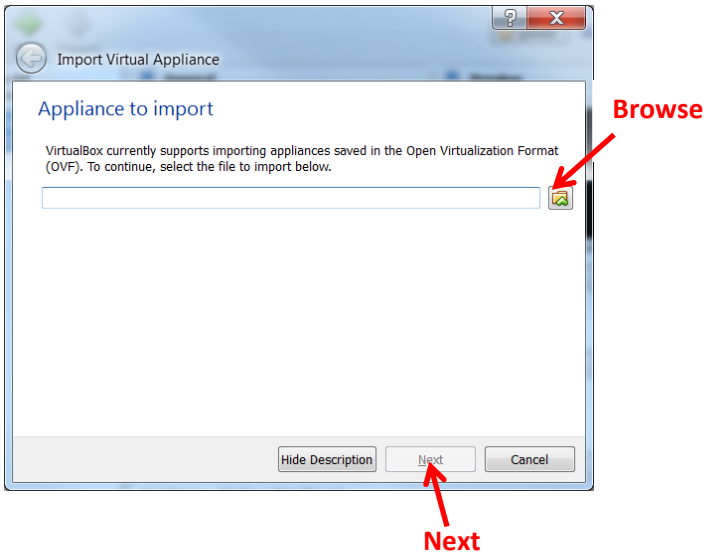
Now you have to import the virtual machine.

Importing Preconfigured Virtual Machine

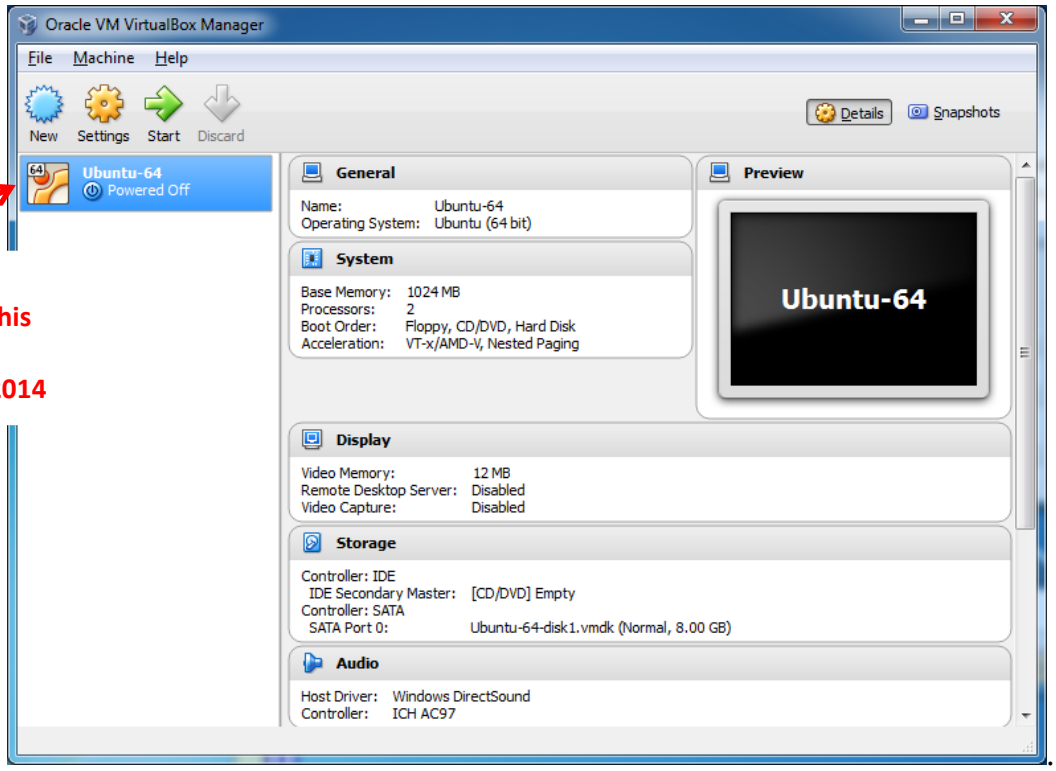
The .ova file is provided on a link on the course home page under the Pre-assignment link “VM Software”. Download the .ova file.

Go to **File > Import Appliance...** and browse for the downloaded .ova file (probably in the Download directory). Click **Next** and **Import**.



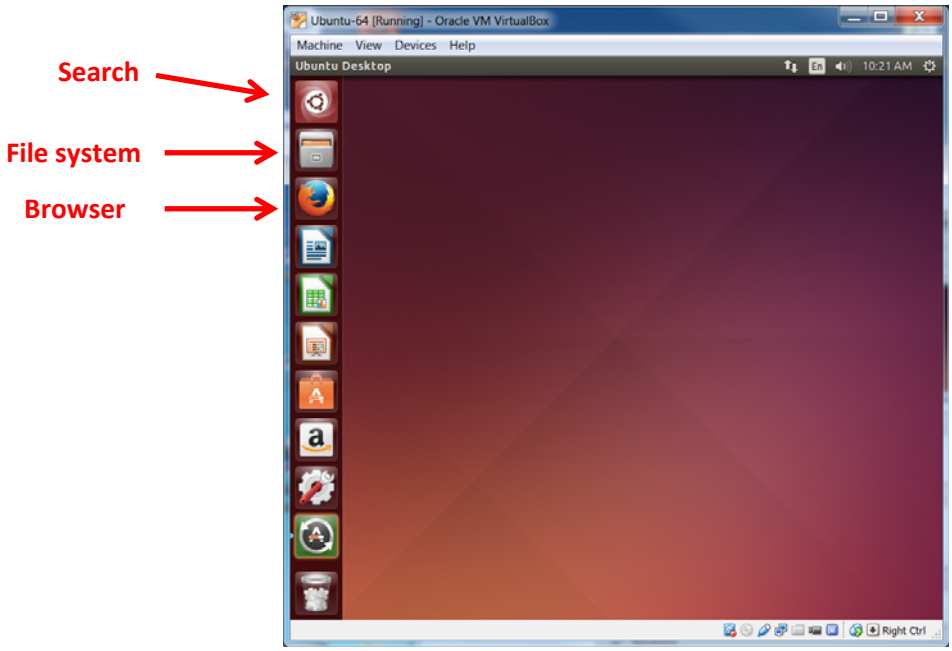


Once imported, you should then see the Virtual Machine:



On the supplied virtual machine this is called ParalleProg_Fall2014

Double click to start. [4]

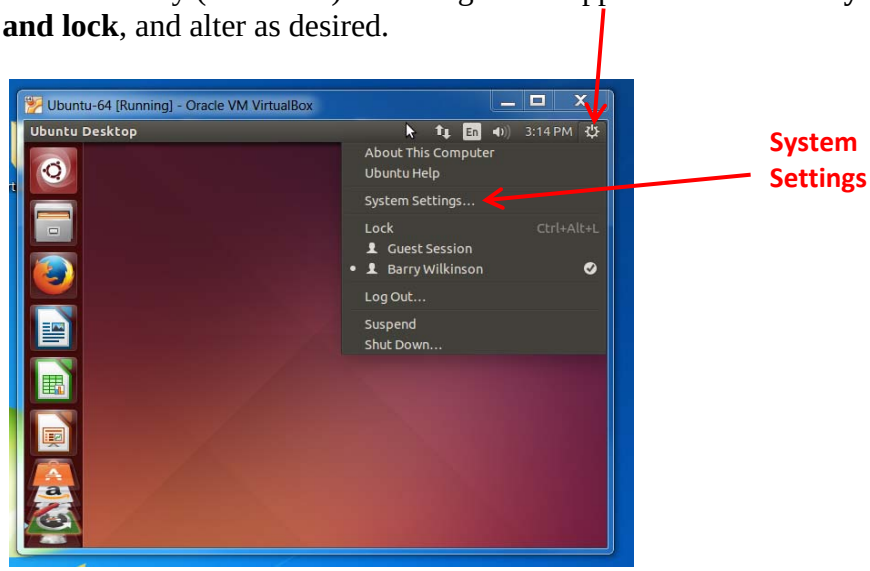


Ubuntu default installation does not allow root logon. Use **sudo** for commands needing root permissions.

User name: **abw**
Password: **abc123**

Note the home directory is **/home/abw/** and is referred to as ~/.

Screen locking. For convenience on a private computer, you may wish to change the setting for locking the screen after user inactivity (5 minutes). Go to right side upper corner icon > **Systems Settings** > **Brightness and lock**, and alter as desired.



[4] If you do not see the virtual machine or it will not start, first check the assignment FAQ page for any known issues.

Copy-and-Paste across Host and Guest OS.

The virtual machine is installed with VirtualBox "guest additions" that enable a number of important features such as copy-and-paste across host and guest OS and desktop scaling. This makes your life much easier. After guest additions are installed, copy-and-paste across host and guest OS is then enabled by first selecting the machine icon (e.g. "ParallelProg_Fall2014") and then setting from **Machine > Settings > Advanced > Shared Clipboard "Bidirectional" and Drag 'n' Drop "Bidirectional"**. Note this only applies to a particular machine. If you have multiple virtual machines installed, you would need to do it for each machine.

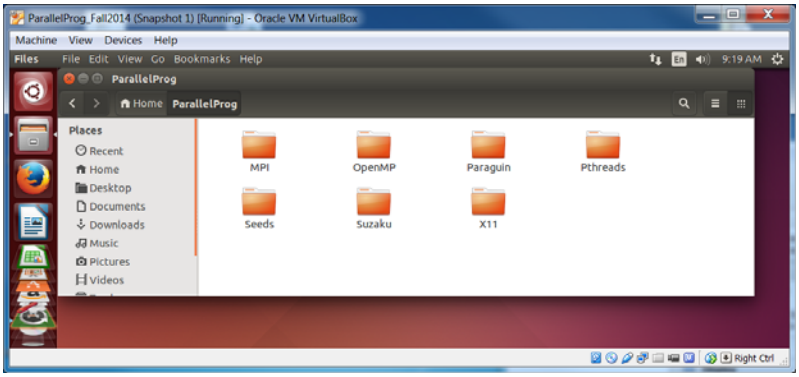
Closing the Machine. Note: When you close a virtual machine you have the option of "*Save the machine state"* or *"Power off the machine"*. *"Power off the machine"* is necessary if you need to reboot.

Software
Within that virtual machine is the following software stack:[5]

- 64-bit Ubuntu Linux OS version 14.04, with VirtualBox guest additions
- OpenMPI version 1.8.1
- Java version 1.7
- Eclipse version 4.3.2 (Kepler) with plug-ins for both Java and C programming including the parallel tools platform (PTP) for MPI and OpenMP

Within the directory **~/ParallelProg**, there is one directory for each programming environment we may need in assignments:[6]

- OpenMP
- MPI
- Paraguin
- Seeds
- Suzaku



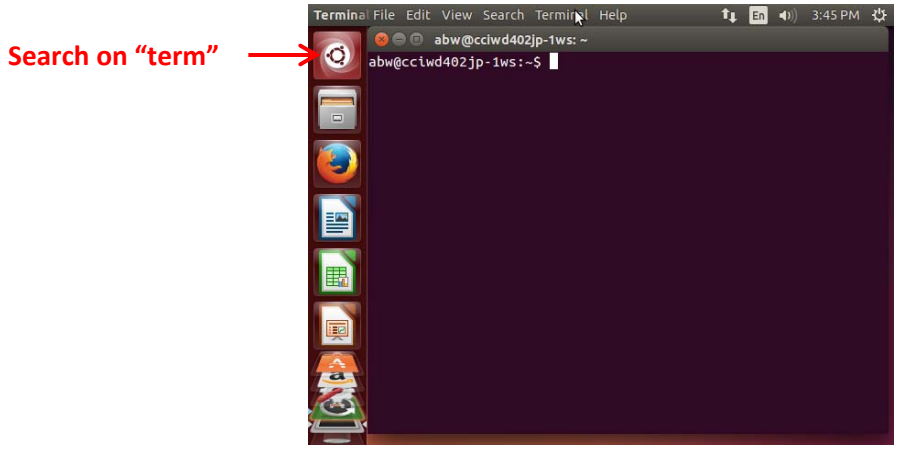
[5] The versions of the various software packages may be updated over time, and generally you can use more recent versions as they become available.
[6] The other directories provide sample X11 and Pthreads code.

Within each of these directories are sample programs and test files for the corresponding environment (e.g. **hello.c**, **matrixmult.c**, etc.) and a **workspace** directory used for compiling and running the programs as Eclipse projects.

Part 2 Testing environment

Your environment will be tested with the provided sample programs. The environment and its use will be described in more detail in subsequent assignments

Terminal window. Click the search icon (to left on icon ribbon) to search on "Term" to find the terminal and start it:



Command line. Many of the assignments ask you to compile and execute programs using a command line. So let's just try a really simple C program.

If it does not exist, create a directory within **~ParallelProg** called **C**, and cd into that directory. Create a C program called **hello.c** in this directory with the contents:

```
#include <stdio.h>
#include <stdlib.h>
int main(void) {
     printf("Hello World\n");
     return 0;
}
```

using an editor (typically **gedit** in Ubuntu). Compile this program with the command:

cc hello.c –o hello

Execute the program with the command:

./hello

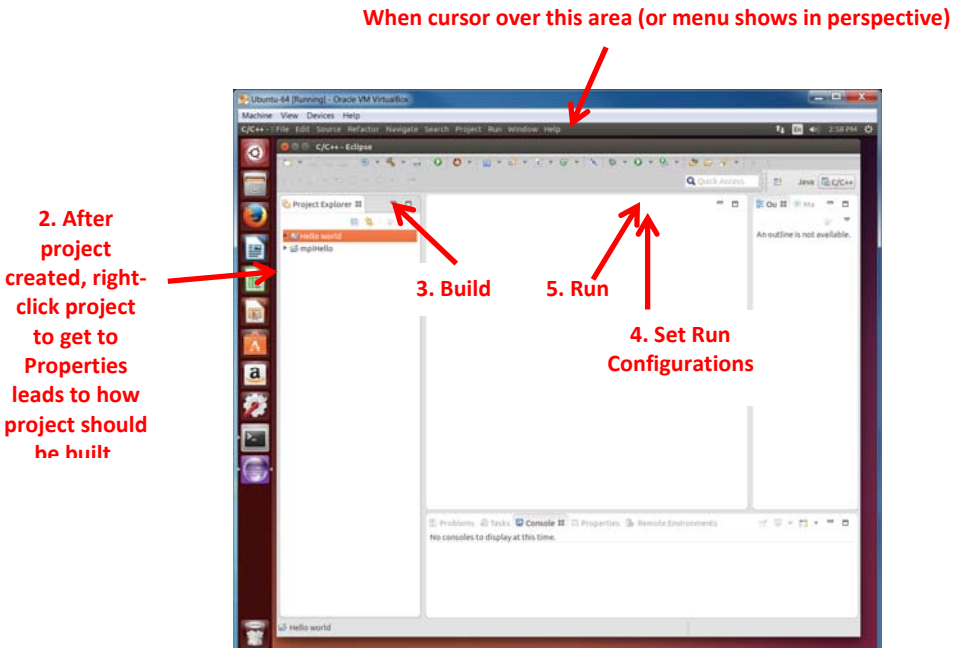
Note **./** (the current directory) is usually necessary. Save a screenshot of the output for your submission document.

Eclipse

Certain assignments also ask you to use the Eclipse IDE, so let try that.

General. Eclipse has environments called "Perspectives" for different programming languages. It is installed in the virtual machine with both Java and C perspectives, including for parallel programming tools. Here we will use the C perspective. The basic steps to compile and execute a program are:

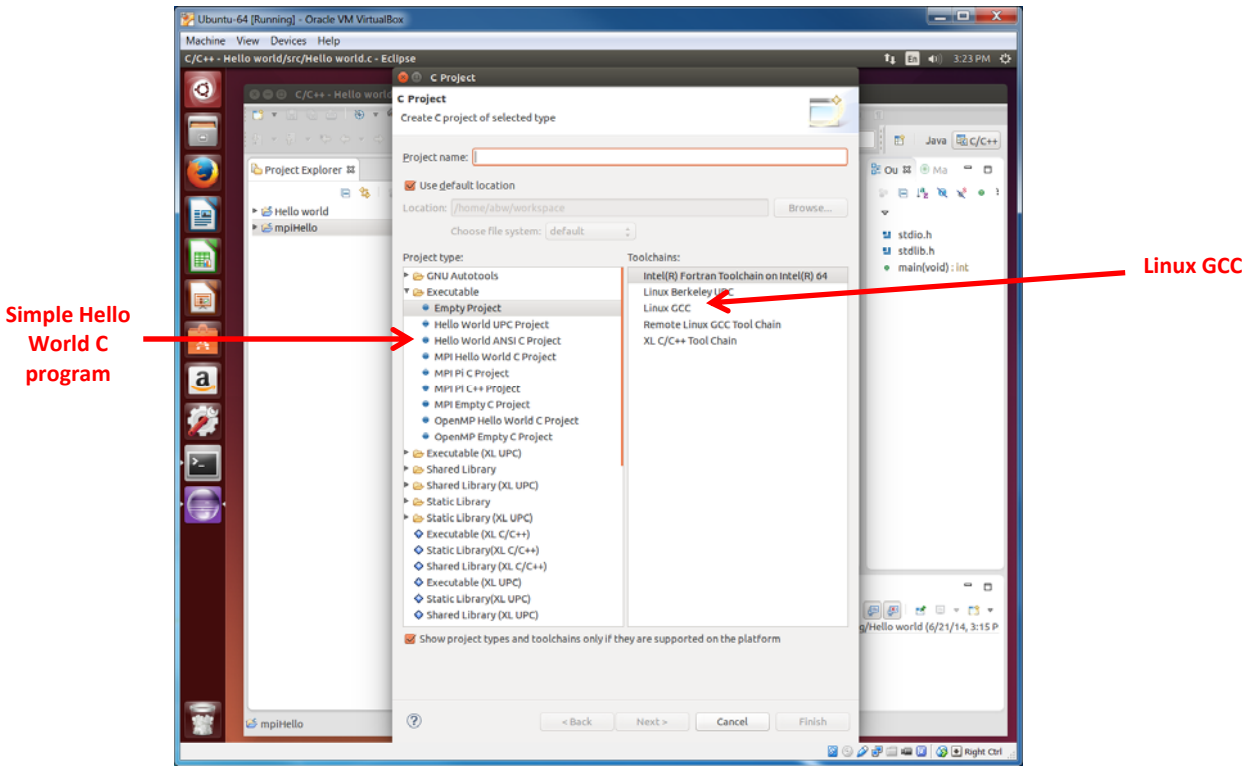
1. Create a project with the source file and any required libraries
2. Set how to build (compile) project in **Properties > ... Build**
3. Build project (compile to create executable)
4. Set how to execute compiled program in **Run Configurations**
5. **Run** (execute) using the specified run configurations



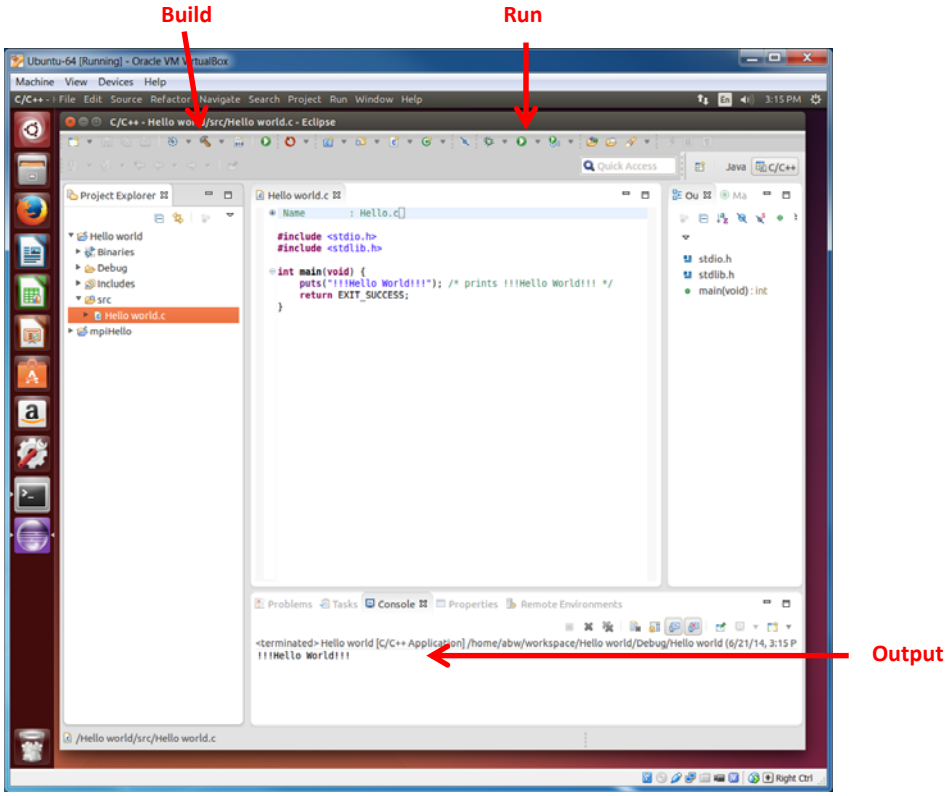
Using Eclipse. Start Eclipse on the command line by typing:

eclipse

Create a new workspace called **~/ParallelProg/C/workspace** and go to the workbench. We will test the environment using the sample C program (Hello World) given in Eclipse. Create a new C project (**File > New > C Project**), and select the "**Hello World ANSI C** Project" type and **gcc** compiler:



Create the “Hello world” C project with a name, built it, and run:



Parallel programs (OpenMP, MPI, etc) will be tested in subsequent main assignments.

This part can only be done once you receive your credentials (user name and password) for a remote server.

Part 3 Connecting to remote servers

After programs have been tested on your own computer, you may be asked to connect to a remote cluster at UNC-C or UNC-W and test you programs there using multiple processors. First carefully review the notes on each cluster posted on the course home page:

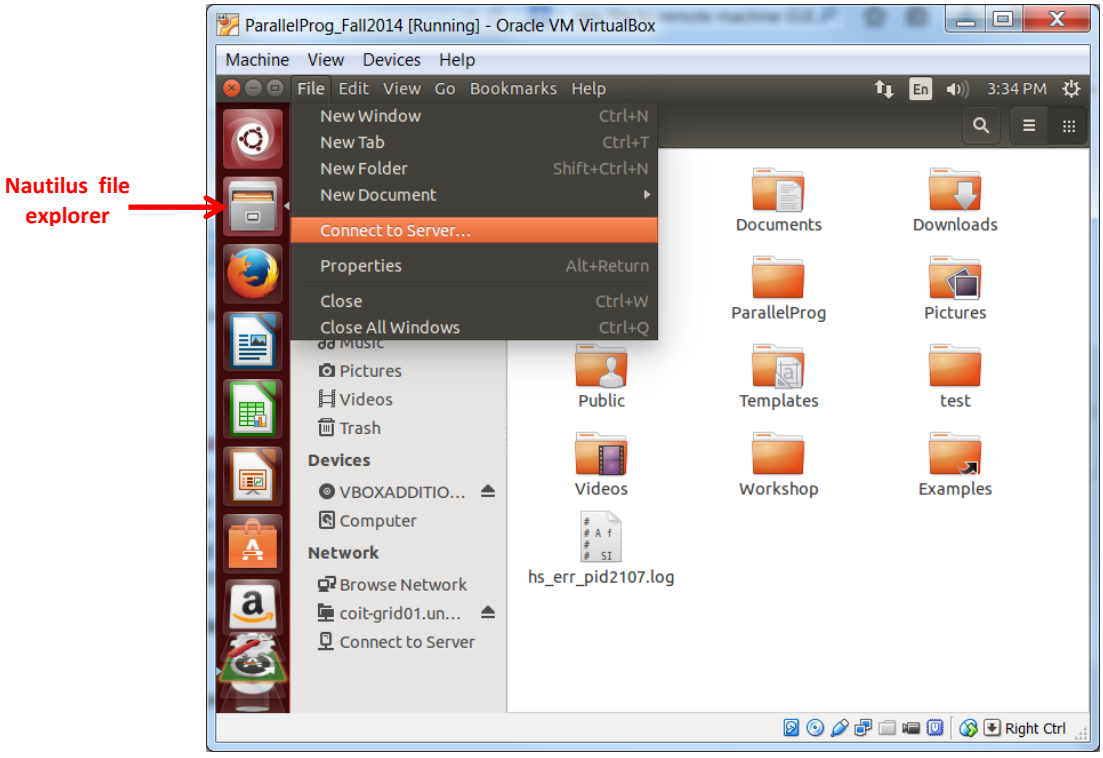
- "Using UNC-C cci-gridgw.uncc.edu cluster"
- "Using UNC-W babbage.cis.uncw.edu cluster"

Command line. While it is possible to use clients such as Putty and WinSCP on Windows systems, the most convenient approach, especially for file transfers to and from the virtual machine and the remote server, is to make the connection directly from the virtual machine. A virtual machine terminal can be used to access remote severs using the command:

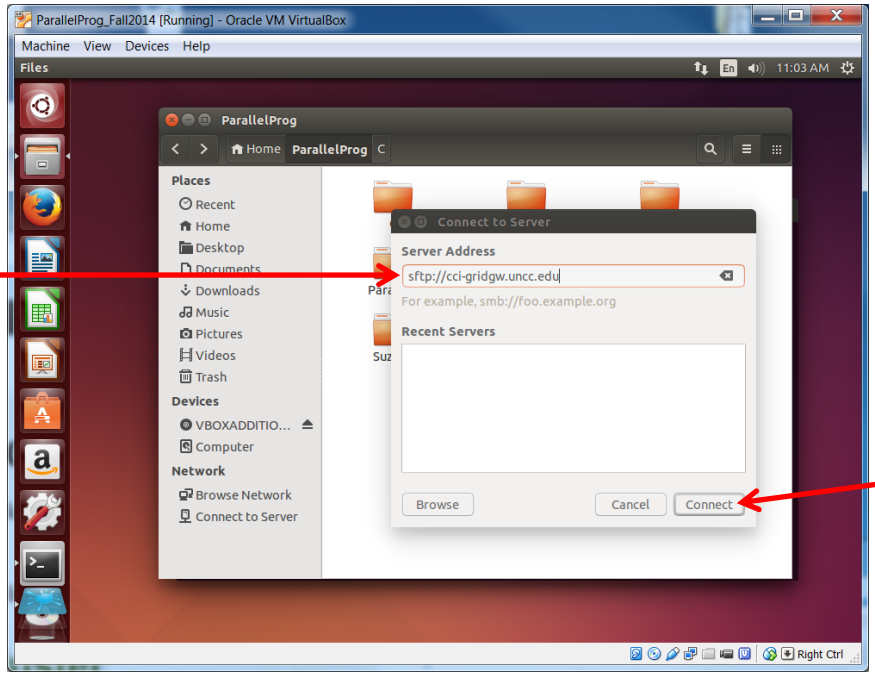
ssh –l *username* cci-gridgw.uncc.edu

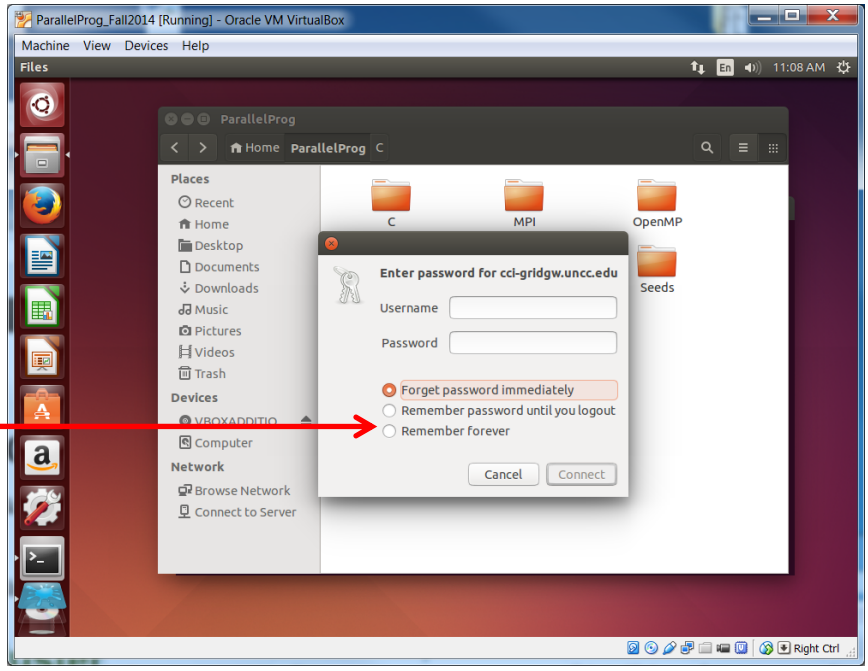
where ***username*** is your user name on the server and **cci-gridgw.uncc.edu** is the server name. Notice the **–l** option is necessary as your username on the remote server will be different to that on you virtual machine (or you could use ***username*@cci-gridgw.uncc.edu**). You will be prompted for your password on the remote server. Then Linux commands can be issued, see the section "**Commonly Used Linux Commands**."

For file transfers to and from the virtual machine one could use Linux commands such as **scp** but a more convenient approach is to use the Ubuntu (Nautilus) file explorer. Go **to File > Connect to Server**:



Enter the servers address with the **sftp** protocol: **sftp://cci-gridgw.uncc.edu.**





Once you have entered your credentials (username and password), you should see your home directory on the remote server and be able to move files between the remote server and the local machine.

If you have any problems at this stage, check the Assignment FAQ for known issues.

Command Line. Create, compile, and execute the **hello.c** C program on the remote server (using the same commands you used on your local virtual machine).

Using Eclipse. It is possible to use Eclipse to make a remote connection and compile and execute programs on the remote server. It is even possible to synchronize local and remote copies of files (WinSCP also has this feature). More on that later.

Assignment Submission

What to submit

Your submission document should include the following:

Your name, whether you are an undergraduate or graduate student, and your institution.

Part 1

(a) A screenshot showing you have installed the virtual machine.
(or have installed a native installation).

Part 2:

(a) A screenshot of the sample C program executing on your computer using the command line
(b) A screenshot of the sample C program executing on your computer using Eclipse

Part 3: If you have received your username and password for a remote server

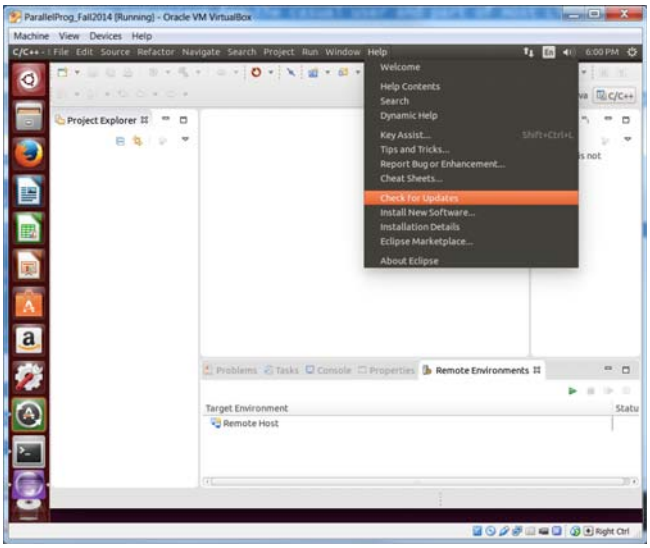
(a) A screenshot demonstrating that you can connect to a remote server.
(b) A screenshot of the sample C program executing on the remote server using the command line

Submit **ONE PDF** file containing your submission on the UNC-C Moodle-2 by the due date as described on the course home page.

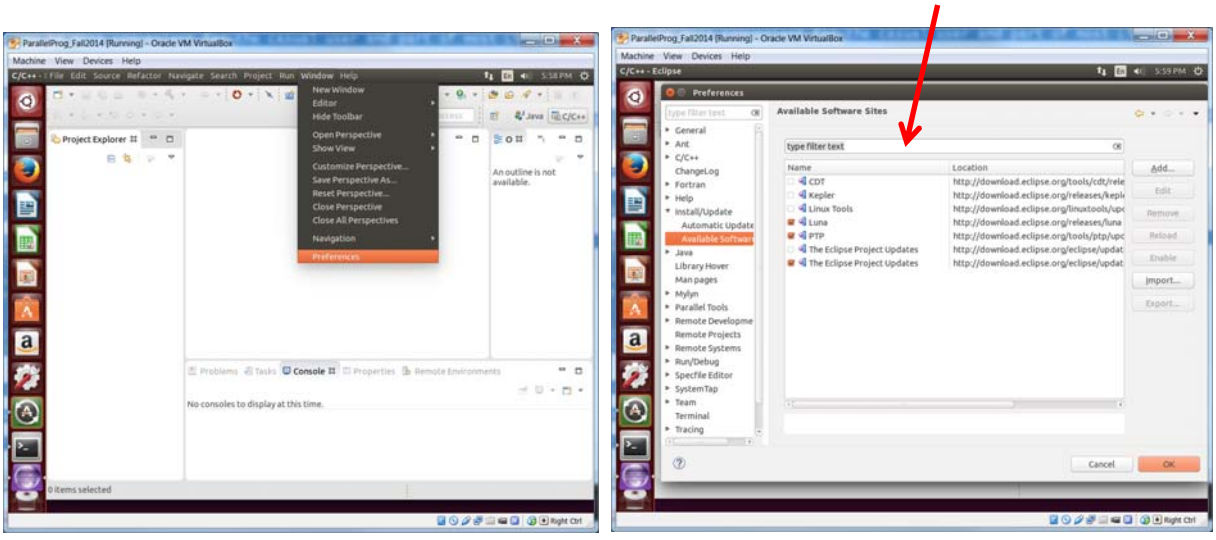
More Information

Updating Eclipse

Go to to **Help > Check for Updates**, select and install updates:



For a major release. It is not necessary to re-install Eclipse to update Eclipse for a major release, for example from Kepler (version 4.3) to Luna (version 4.40). First update the repository URL. Start Eclipse. Go to **Window > Preferences > Install/Update > Available Software Sites** and enter and select the new repository URL. e.g. http://download.eclipse.org/releases/luna.

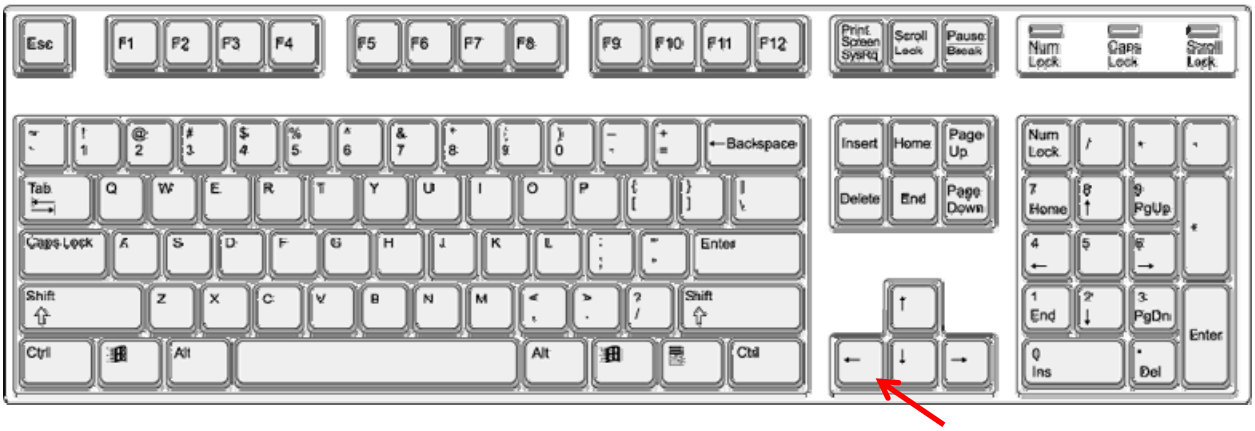


Next go to to **Help > Check for Updates**, select and install updates. You will need to reboot the system completely to get Eclipse to function correctly after a major upgrade.
For more information on VirtualBox see https://www.virtualbox.org/manual/ch04.html
Ubuntu Documentation Eclipse IDE: https://help.ubuntu.com/community/EclipseIDE

FAQ about VirtualBox

1. **Where is the “host” key?**

   “Right control” key, that is, *the control key on the right side of the keyboard* (not the control key on the left side of the keyboard).



The key for “host key” can be changed under **File > Preferences > Input > Virtual Machine**.

2. **Why is there no the machine menu at the top?**

   You are in the scale mode. Switch out of the scale mode with “host key” + C.

3. **How do I make the view bigger?**

   Switch scaled mode (“host key” + C)

Commonly Used Linux Commands

Viewing and Navigating Directories

Frequently used commands for viewing/navigating directories.

ls -a	List files in current directory. **-a** says list all files including hidden files (those starting with a period)
pwd	Print the full path of the current directory
cd *directory*	Change user's directory to that specified, or if none specified, to user's home directory
cd ..	Move up one directory

Changing Password

Once you have logged on, you may need to change your password as your account may have been set up with a password that was sent to you in an insecure way (say by email). The command is

passwd	Change user password, which will prompt for the current password and then the new password.

Creating Directories and Files

Program files are usually created using an editor. The standard GUI Linux editor on Ubuntu is **gedit**, which can be invoked with the command **gedit *filename***, or selected after searching on "gedit". To use **gedit** on a remote server, you will need to connect with X11 forwarding using the **–X** option:

ssh –X –l *username* cci-gridgw.uncc.edu

Non-GUI editor. For remote servers without using X11 graphics, one simple editor that may be convenient for the casual user and part of most Linux installations is called **nano**. To invoke **nano**, type **nano** *filename* or without a file name if file not yet created. One simply types into the window to change the contents the file. Use **control-O** to save and **control-X** to exit. Available commands are given at the bottom of the window.

Manipulating Directories and Files

Common commands for manipulating directories and files:

mkdir *directories*	Create one or more directories
cat *files*	Displays contents of files.
more *file*	

less *file*	Both display file one page at a time (space bar to go forward one page, 'b' to go backward one page, 'q' to quit)
cp *file1 file2*	Copy *file1* to *file2*
cp *file1 directory*	Copy *file1* to *directory*
mv *sources destination*	Move or rename files and directories
rm *files*	Delete one of more files
rmdir *directories*	Delete one of more directories. Must be empty.
rmdir -r *directories*	Remove non-empty directories (recursive remove)

Getting information, killing jobs and logging out

man *command*	List information about command, arguments etc. Type **q** to leave before end.
kill *n*	Kills process *n* where *n* is the process number, which can be obtained from **ps** command
ps	Display information about processes.
exit	Terminate your login shell