Evening Lab: BCCD/LittleFe

Blue Waters Petascale Institute 2017

Day 8 (Week 2, Wednesday, June 7) 7:00pm-9:00pm

Author: Skylar Thompson, University of Washington

Materials

- VirtualBox
  - You downloaded VirtualBox 5.0 before you came to the institute
  - Also make sure to download the extension pack
- Bootable Cluster CD (BCCD)
  - You downloaded BCCD v3.3.3 before you came to the institute
  - BCCD documentation

Introduction

LittleFe and BCCD are free, open source hardware and software solutions, respectively, for computational science education. They are often used as an intermediate step between running simulations on a laptop and running on a campus cluster, XSEDE resources, and/or Blue Waters. They can be used separately, or together, and many curriculum modules are already written assuming the use of one or both of them.

LittleFe is a multi-node (currently, 6) low-footprint computational cluster that can be assembled in less than a day by someone with only basic aptitude with basic tools. Institutions can also order LittleFe units to be assembled and shipped to them, for a nominal fee. The cluster has a built-in Ethernet switch, and the entire unit can run off a single AC outlet. One node in the cluster has both a SATA hard drive for persistent storage and a Wi-Fi card and USB Ethernet NIC for external connectivity. The hardware manifests for the various versions of LittleFe are available online. There are over 100 LittleFe units in the world today.

BCCD (the Bootable Cluster CD) is a Debian-based Linux distribution that automates the setup of a computational cluster. Despite its name, it can run on both regular hard drives (or their virtual equivalents) and also USB drives. It can run on any standard x86 (i.e. Intel or AMD) hardware and does not need to run on dedicated cluster hardware (common targets are school computer labs, or a couple virtual machines). Its initial operating mode is to run out of RAM, so the existing contents of the hard drive (if one is available) are never touched. An optional operating mode copies the contents of RAM (including any customizations) to the hard drive, **but will destroy the contents of the hard drive.** After booting the first node, it takes less than five minutes to get it configured, and a default configuration can be selected by pressing **`Enter`** at every prompt.

In this lab, we will set up a BCCD cluster of virtual machines running on our laptops.

Setup

Before beginning, please:

1. Make sure to disable sleep/power management for the duration of the lab.
2. Make sure that you have VirtualBox and the extension pack installed for your platform. Mac OS X and Microsoft Windows systems just have an executable, while Linux systems will have a package. Feel free to take the defaults when installing.

Configure VirtualBox

1. Start VirtualBox. You should see a welcome window appear.
2. Click *New* to create a virtual machine.
3. In the *Name and operating system* window
    a. Set *Name* to **`BCCD`**
    b. Set *Type* to **`Linux`**
    c. Set *Version* to **`Debian (64-bit)`**
4. In the ***Memory size*** window, accept the default memory size (or less, down to 512MB, if you wish)
5. In the ***Hard disk*** window, select **`Do not add a virtual hard disk`**. It will complain, but BCCD is designed to operate without a hard disk.
6. After the VM is created, there are some more customizations to do.
    a. Click on the *System* link, and then click on the *Processor* tab. Change the number of *Processor(s)* to the number of physical cores on your system.
        i. If the Processors slider is grayed out, leave it set to the default of 1 (not all platforms support multi-CPU VMs)
        ii. Note that the green part of the slider will show you the maximum number of cores it is recommended to use. This would correspond to one virtual core per physical core, unless your system has hyperthreading enabled, in which case the green part of the slider will allow more than the physical number of cores on your system.
    b. Click on the ***Storage*** link, and then click on the ***Empty*** optical drive icon in the Storage tree. Over on the right hand side, click the CD icon, and then select the BCCD ISO image that you downloaded previously. After selecting the ISO image, check the box for the ***Live CD/DVD***.
    c. Click on the ***Network*** link. In the Adapter 1 tab, change the ***Attached To*** drop down to Bridged Adapter (it probably says NAT). Make sure to select the **non-wireless** NIC to bridge onto.
    d. Open the ***Advanced*** section of the ***Network*** link. Select the **`PCnet-Fast III`** driver and set **Promiscuous Mode** to **`Allow All`**.
7. You are now ready to start playing with BCCD!

Booting the BCCD

Your first task with the BCCD will be to boot it up.

1. Elect one person in your group to run through the following process, prior to anyone else (this person's computer will be the "head node," which gives IP addresses and hostnames to the other nodes).
2. Select BCCD in the VM list, and click Start.
    - If you are on a Linux system and get a message about the kernel drivers being unavailable:
        i. Make sure to install the kernel header package:
        `sudo apt-get -y install linux-headers-$(uname \ -r)<ENTER>`
        (for Debian-based systems), or
        `sudo yum -y install kernel-devel<ENTER>`
        (for Red Hat-based systems)
        ii. Run the setup script:
        `sudo /etc/init.d/vboxdrv setup<ENTER>`
        iii. Try starting the BCCD again
3. You should now be at a BCCD boot prompt. Either wait a bit, or speed things along by pressing *Enter.*
    - As the system boots, take note of the penguin count in the upper left. It will be equal to the number of Processor(s) that you specified when you set up the virtual machine. If you were running on real hardware, it would be equal to the number of CPU cores on the system itself (unless the CPU is doing hyperthreading, in which case there will be more penguins).
4. You should now be at a password prompt. Enter whatever password comes to mind; you will rarely need this, but if you need to execute an administrative command you will have to provide it. The password we usually choose for this lab is just **`bccd`** (which is not very secure, but since we are on a local network with no outside connections, and we are only on there temporarily, we are safe). Note that as you type, there will be no keystrokes shown.
5. After entering your password a second time to confirm it, the system will try to discover other BCCD systems on the same network. The first BCCD system to come up will start a network configuration server that will answer only to other BCCD systems; these other systems will find it and configure themselves to be on a private network running alongside whatever other network happens to be operating. You should always be sure one BCCD system has booted completely before booting the others; otherwise, multiple systems will try to run the network configuration server (this is why we had only one person's computer serve as the "head node").
6. After the automatic configuration is done, you will be at a BCCD desktop. Two terminals will be running, each of which will show your node number (e.g. **`node000`**).
7. Once the first person completes the process, everyone else can boot (in parallel, no more waiting needed).

Running BCCD applications

The BCCD ships with a number of applications to illustrate parallel concepts. Let's start with GalaxSee, which is an MPI-based *n*-body simulation for galaxy formation, written by Dave Joiner at Kean University.

An aside on MPI: many supercomputers (i.e. Blue Waters **aprun**) will integrate MPI with the job scheduler. This has the advantage of making MPI easier to use for the researcher, but separates you from the actual cluster in that you don't actually know which nodes you will be running on. While BCCD ships with a job scheduler (Torque), most curriculum modules do not use it. Rather, you generate a *machines* file using the **bccd-snarfhosts** command, which is a list of nodes combined with their core count. The BCCD machines files will be placed in your home directory, with the suffix of the MPI implementation they were generated for (i.e. **~/machines-openmpi**). Not using a job scheduler makes it simpler for people to start using a cluster, and means no waiting around for **qsub**!

An aside on BCCD storage: Unlike a supercomputer, BCCD does not have a shared storage system (unless you "liberate" it; that is, install it on a hard drive). Every BCCD instance has its own local storage in memory, which includes the BCCD user's home directory. In order to make a piece of software available to all nodes, you must copy it to every node you will run on. You can use the **bccd-syncdir** command, which accepts a directory to copy and a list of nodes in a MPI machines file. The given directory will be copied to every node in the **/tmp/node*nnn*-bccd** directory, where *nnn* is your local node's number (see your shell prompt). You can view more BCCD-specific commands here.

1. Generate a MPI machines file nodes:
   bccd-snarfhosts -s<ENTER>
   a. The **-s** tells `bccd-snarfhosts` to sort the file by node name, which is needed for X11 forwarding
2. Examine the machines file:
   cat ~/machines-openmpi<ENTER>
3. Enter the GalaxSee directory:
   cd GalaxSee<ENTER>
4. Compile the software:
   make<ENTER>
5. Copy the software to every node in the machines file:
   bccd-syncdir . ~/machines-openmpi<ENTER>
6. Try running it with 4 processes, specifying your own node number below, and timing how long it takes to run:
   time mpirun -np 4 -machinefile ~/machines-openmpi
   /tmp/node<Your node number>-bccd/GalaxSee.cxx-mpi 1000
   500 400<ENTER>

    a. The three numbers at the end are the number of bodies, mass of each body, and the number of time steps, respectively. The number after the **-np** (in this case 4) is the number of MPI processes (similar to **-n** for **aprun**).
7. Try playing around with the number of processes and the runtime parameters (number of bodies, mass of each, number of time steps) to see how the model works and figure out how well it speeds up as you scale up the number of processes. If it doesn't speed up perfectly, can you figure out why not?

Different MPI implementations

One advantage that BCCD has over some clusters is that it ships with two MPI implementations. The default is OpenMPI, which is what you used above. BCCD also ships with MPICH2. You can toggle between them with the **module switch** command:

```
module switch openmpi mpich2<ENTER>
```

Note that when you switch, you must remember to recompile and recopy software and re-run **bccd-snarfhosts**.

Let's try this with GalaxSee:

1. Enter the GalaxSee directory:
   ```
   cd ~/GalaxSee<ENTER>
   ```
2. Switch to mpich2:
   ```
   module switch openmpi mpich2<ENTER>
   ```
3. Generate a MPICH2 machines file:
   ```
   bccd-snarfhosts -s<ENTER>
   ```
4. Examine the new machines file, noting the different format:
   ```
   cat ~/machines-mpich2<ENTER>
   ```
5. Recompile GalaxSee:
   ```
   make clean && make<ENTER>
   ```
6. Re-sync:
   ```
   bccd-syncdir . ~/machines-mpich2<ENTER>
   ```
7. Run with MPICH2:
   ```
   time mpirun -machinefile ~/machines-mpich2 -np
   4 /tmp/node<Your node number>-bccd/GalaxSee.cxx-mpi 1000
   500 400<ENTER>
   ```
8. Do some experiments to determine the speedup as you scale the number of processes and/or number of bodies. Do you notice any performance differences between the OpenMPI and the MPICH2 version?

Feel free to poke around the other modules in the BCCD home directory. Over time, we will also include more modules from recipients of LittleFe buildout events.