The Bootable Cluster CD for High Performance Computing Education

Paul Gray
Dept. of Computer Science
University of Northern Iowa
gray@cs.uni.edu

Sarah Miller
Dept. of Computer Science
University of Northern Iowa
sarahm@uni.edu

Abstract

This paper confronts the issue of High Performance Computing (HPC) platforms and their suitability to support much-needed education of HPC topics at the undergraduate level. One of the most demanding aspects of teaching HPC topics is the commensurate demand in administrative tasks—from maintenance of user accounts and configuring software components, to keeping ahead of the latest security advisories. We focus on the Bootable Cluster CD (BCCD) for its breadth of HPC software, ease of configuration, and ease of administration.

The BCCD is a self-contained clustering environment in a bootable CD format. The project was motivated by the need for a drop-in clustering environment for parallel computing education at community colleges and primarily undergraduate institutions (PUI's). Unlike research institutions, where dedicated clustering resources have the benefits of full-time support and administration staff, the support for hardware and the ongoing administration of clustering environments is a significant resource drain on community colleges and PUIs.

However, these days student labs with Windows operating systems are becoming universally available for class-time use. The Bootable Cluster CD project was put forward to leverage these environments so as to provide a convenient, non-destructive (no installation required), pre-configured clustering environment—providing PVM [6], MPICH, OpenMosix [1], compilers, and graphical debugging tools—and to "leave no trace"' when the host computer system is allowed to reboot normally.

In this way, the BCCD project offers a drop in solution that requires no dual-boot configurations, minimal system configuration, no ongoing administration, and an extremely robust on-demand development environment. This paper will describe the flexibility inherent in building the BCCD, describe its use at SuperComputing for supporting workshops on High Performance Computing Education, and tips for effective usage in concert with the IEEE/ACM Curricular Guidelines for Computer Science programs [4].

Introduction

The BCCD is a bootable CD image that boots up into a pre-configured distributed computing environment targeted toward HPC Education (HPCE). The BCCD project was motivated by numerous factors that are present at most undergraduate institutions; namely lack of available clustering resources locally, lack of funds to secure a dedicated clustering resource, and lack of staff to administrate and support the additional aspects of a true clustering environment (such as schedulers, image synchronization, etc.). The BCCD image was not put together for the intention of building HPC environments, but rather environments for HPC education.

Based upon the motivations above, the intended use of the BCCD is to be able to walk into an open laboratory of computer systems[1], boot up the CD, and instantly have a pre-configured distributed computing laboratory. The instructions for classroom use of the BCCD image are relatively straight forward:

1. burn it to a CD
2. put it in your CD tray
3. reboot
4. follow on-screen instructions
5. repeat steps 1—4 for all machines that you'd like to have in your lab or cluster

This image boots up into a self-contained, “non-destructive” environment that allows the user to build virtual machines from other systems that are similarly booted, not to mention build and run all types of parallel paradigms (described later in this document). The non-destructive aspect of the BCCD means that the software is never installed to the workstation's hard drive and that no multi-booting operating systems is required. From an administrative standpoint, this is a win-win situation for supporting a clustering environment.

Built around the GNU/Linux operating system and distributed under the version two of the Gnu Public License, the BCCD system is able to support a wide breadth of hardware support and a wide selection of applications. A wide breadth of hardware is supported at this time, and additional hardware support is added regularly. The BCCD contains over 1400 applications ranging from a wide selection of editors, web browsers, mathematical tools, graphing programs, networking utilities, and even a game or two. In addition, there is an abundance of compilers, debuggers, profiling libraries, and visualizing applications for parallel programs. It is this breadth of applications geared toward HPC that makes the BCCD unique.

A BCCD image is built dynamically using predefined Makefiles, GNU make, and web-fetched source files. In this way, the standard BCCD image can be customized to suit the needs of one's local institution by simply modifying the corresponding Makefiles. All of the sources required to build a BCCD from scratch are available through CVS checkout or source tar file from the project web site http://bccd.cs.uni.edu.

1 Currently, only i386-based systems are fully supported. A PPC (Macintosh) is also under development, and should be released by third quarter 2004.

The remaining portion of this document describes the design goals, the makeup, the similarities and differences with other HPC platforms, and the historical usage of the BCCD in the context of national workshops for HPCE. The descriptions and figures used to illustrate the functionality of the BCCD clustering environment in this document were taken from development snapshots of the BCCD version 2.2 CVS tree.

BCCD Goals & Objectives

The primary design goals of the BCCD are

i. Breadth of hardware support.
ii. Acclimation to arbitrary network environments.
iii. Ease of configuration through customized scripts and pre-configured software packages.
iv. Ample packaged examples and demonstrations for extensive investigations into core distributed computing concepts.
v. Ease of customization to tailor the image for local needs.

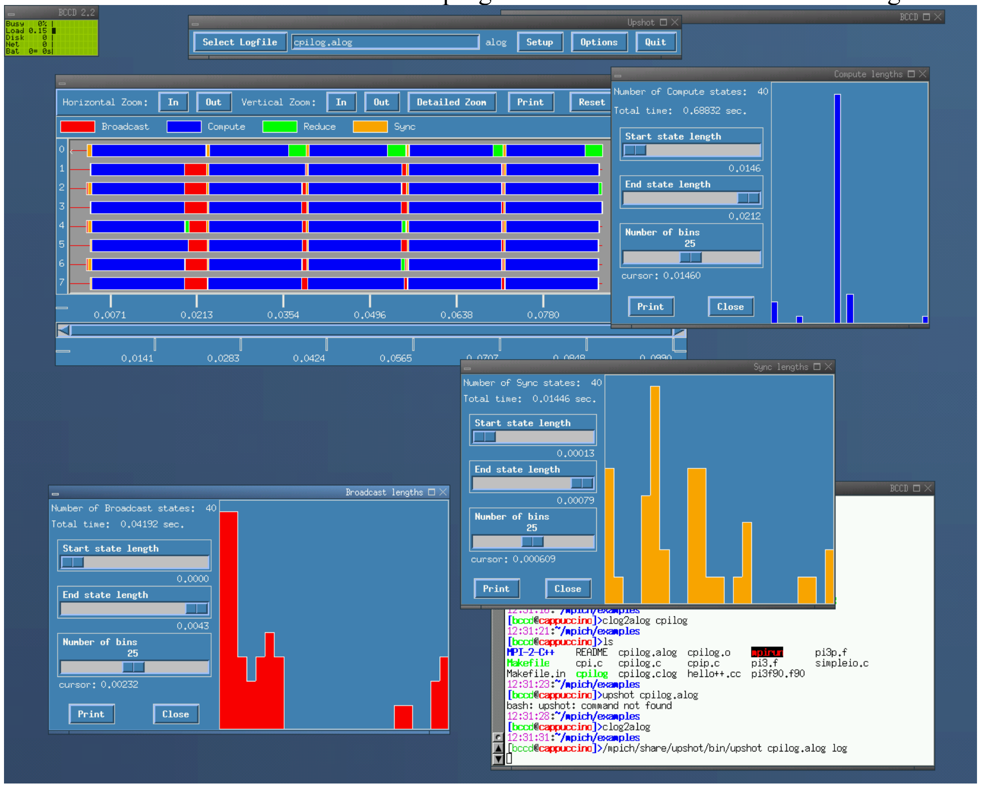
The BCCD derives its breadth of hardware support from the extensive support of the GNU/Linux operating system. Automatic configuration of hardware is accomplished through the *discover* and *hotplug* hardware configuration utilities. These utilities are similar to the plug-and-play paradigm which automatically configure hardware during the boot-up process. By default, the BCCD uses the VESA frame buffer compatibility mode to support high-resolution (non-accelerated) graphics. This allows support for the most common video mode supported by standards-compliant video cards.

Networking infrastructures vary from institution to institution. In order to support the most arbitrary of networking environments, the BCCD includes a multitude of networking utilities. The environment most amenable to configuration of a BCCD clustering environment is one that has pre-configured DHCP and DNS services with resolvable reverse-lookup support (which is required by the MPICH and LAM [3] clustering environments). However, in the absence of these services, the BCCD supplies, among numerous other networking services, configurable DNS and DHCP services.

The most arduous aspects related to the utilization of HPC environments in an educational setting are configuration, administration (of software packages), and security of the environment. Clustering applications and environments, such as PVM, MPICH [7], and the openMosix tools, boot up pre-configured and ready to use. Customized scripts automate the tasks of connecting together your BCCD-based workstations. In general, customization and configuration are part of the build processes instead of the run-time system, which saves time and energy in administration. In turn, this translates to more time spent on HPC education than on environmental concerns. And, unlike a dedicated clustering resource, once the system on which the BCCD resides on is rebooted, it reverts back to the operating system of the original host.

On every BCCD image, there exists several pre-packaged examples of parallelism. From Fortran-based PVM programs to C++ based visualizations in MPI, the default user environment contains illustrated programs that can be immediately built with a 'make' command and subsequently run in the respective parallel environment. Moreover, there are several visualization applications for displaying, debugging, and viewing the interactions between the components of these example programs. Figure 1 below shows an example of *upshot* being used to visualize messages, CPU time, communication, and the overhead of an MPI-based program found on the default BCCD image.

Figure 1: The *upshot* program being used to visualize messages, CPU time, communication, and the overhead of an MPI-based program found on the default BCCD image.



```
[bccd@cappuccino]>clog2alog cpilog
12:31:21:~/mpich/examples
[bccd@cappuccino]>ls
MPI-2-C++    README  cpilog.alog  cpilog.o   mpirun      pi3p.f
Makefile     cpi.c   cpilog.c     cpip.c     pi3.f       simpleio.c
Makefile.in  cpilog  cpilog.clog  hello++.cc pi3f90.f90
12:31:23:~/mpich/examples
[bccd@cappuccino]>upshot cpilog.alog
bash: upshot: command not found
12:31:28:~/mpich/examples
[bccd@cappuccino]>clog2alog
12:31:31:~/mpich/examples
[bccd@cappuccino]>/mpich/share/upshot/bin/upshot cpilog.alog log
```

Finally, the BCCD project leverages the 'gar' [9] system for building an entire BCCD image completely from net-fetched sources obtained during the compilation process. 'Gar' is a mixture of the BSD 'ports' system, Linux From Scratch (LFS), and the Gentoo Linux 'emerge' process. Using *make*, *autoconf*, and *wget* as the main building tools, gar can put together an entire BCCD image from Makefiles available from the BCCD CVS tree. This allows end-users to customize their images during the build process for specific needs of the runtime system.

BCCD and Other Paradigms

There are several clustering paradigms and implementations to choose from when deciding on a parallel computing environment for HPC education. They range from the overly simple to the overly complex in terms of application support, ease of configuration, the installation process, and maintainability of the environment. The three main solutions involve building a cluster from scratch, clustering distributions, and dynamic (bootable) environments.

A modest, viable approach to setting up a clustering environment that continues to be available to researchers and educators is to purchase commodity off-the-shelf (COTS) components and to build a cluster entirely from scratch. After sufficient testing, troubleshooting, and tuning, this often produces a very capable computational environment. Others, the authors included, have sought out capable "second-hand"' systems that have reached the "end-of-life" in terms of production use. Systems that have performed well in production for one to three years still possess considerable *educational* value for HPC education. The drawback of this approach is the considerable time and effort involved from start to the final environment. Even with the final environment in place, updates and maintenance of the environment require considerable amounts of time.

Clustering distributions, such as OSCAR [5] and NPACI-Rocks [11], add on or integrate with standard Linux distributions. The benefit of this approach derives from pre-configured packages and cluster administration tools. Like the build-from-scratch method, this approach requires dedicated resources.

Finally, the dynamic environments, which include the BCCD as well as CHAOS [8], PlumpOS [13], ClusterKnoppix [12], and LOAF (Linux On A Floppy) [2], allow for a drop-in clustering environment. These drop-in clustering environments are non-invasive, in the sense that there is no installation of software to the hard drive. Each environment exists only while the temporary media is booted. In general terms, CHAOS, PlumpOS, and LOAF provide dedicated openMosix environments, and one cannot introduce new software or clustering applications to the environment. The BCCD, on the other hand, provides a multitude of parallel environments. In addition to openMosix, the BCCD includes MPICH, LAM-MPI, PVM, and the C3 tools. Another distinguishing feature about the BCCD is that it supports unique user accounts.

Components of the BCCD

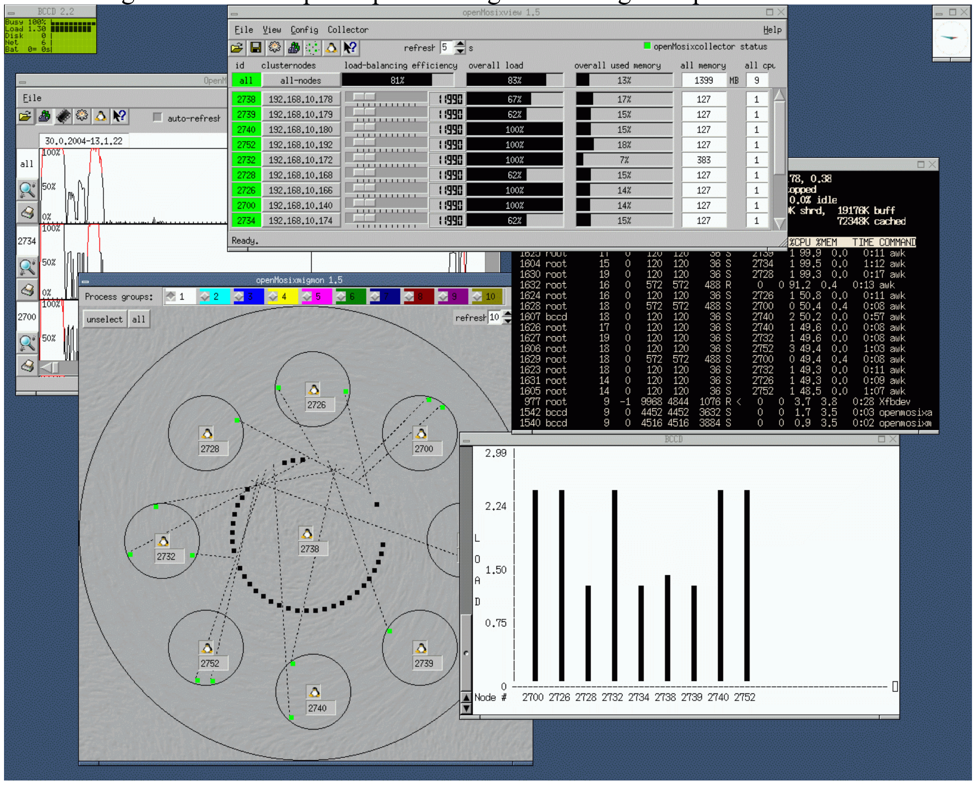
This section describes the software and applications that are available on the BCCD. For starters, the BCCD version 2.2 includes:

- openMosix version 0.3.5 and openMosixView [10] version 1.5
- PVM version 3.4.4
- X-PVM version 1.2.5
- MPICH version 1.2.5.2 (MPICH 2.0 available as an option)
- LAM-MPI version 7.0.4
- C3-tools version 4.0.1
- gcc, g++, gcj, g77, the GNU C, C++, Java and Fortran compilers
- mpicc, mpi77, mpiCC, mpic++
- hcc, hcp, hf77, and various lam tools
- xmpi version 2.2.3b8

- upshot being used on the BCCD to show a parallelized ring-based communication
- XFree86 version 4.3.0
- many other network diagnostic tools

OpenMosix, the openMosix user-land tools, and openMosixView together form an environment that extends kernel-level process-based parallelism. The openMosix environment is based upon a process-based, farming model of parallelism. Process migration is transparent and managed by the kernel. Figure 2 shows an example of process migration using the openMosix model.

Figure 2: An example of process migration using the openMosix model.



id	clusternodes	load-balancing efficiency	overall load	overall used memory	all memory		all cpu
all	all-nodes	81%	83%	13%	1399	MB	9
2738	192.168.10.178	1990	67%	17%	127		1
2739	192.168.10.179	1990	62%	15%	127		1
2740	192.168.10.180	1990	100%	15%	127		1
2752	192.168.10.192	1990	100%	18%	127		1
2732	192.168.10.172	1990	100%	7%	383		1
2728	192.168.10.168	1990	62%	15%	127		1
2726	192.168.10.166	1990	100%	14%	127		1
2700	192.168.10.140	1990	100%	14%	127		1
2734	192.168.10.174	1990	62%	15%	127		1

```
78, 0.38
opped
0.0% idle
K shrd,   19176K buff
          72348K cached

                                              %CPU %MEM   TIME COMMAND
1625 root     17   0   120  120    36 S   2739   1 99.9  0.0   0:11 awk
1604 root     15   0   120  120    36 S   2734   1 99.5  0.0   1:12 awk
1630 root     19   0   120  120    36 S   2728   1 99.3  0.0   0:17 awk
1632 root     16   0   572  572   488 R      0   0 91.2  0.4   0:13 awk
1624 root     16   0   120  120    36 S   2726   1 50.8  0.0   0:11 awk
1628 root     18   0   572  572   488 S   2700   0 50.4  0.4   0:08 awk
1607 bccd     18   0   120  120    36 S   2740   2 50.2  0.0   0:57 awk
1626 root     17   0   120  120    36 S   2740   1 49.6  0.0   0:08 awk
1627 root     19   0   120  120    36 S   2732   1 49.6  0.0   0:08 awk
1606 root     18   0   120  120    36 S   2752   3 49.4  0.0   1:03 awk
1629 root     18   0   572  572   488 S   2700   0 49.4  0.4   0:08 awk
1623 root     18   0   120  120    36 S   2732   1 49.3  0.0   0:11 awk
1631 root     14   0   120  120    36 S   2726   1 49.3  0.0   0:09 awk
1605 root     14   0   120  120    36 S   2752   1 48.5  0.0   1:07 awk
 977 root      9  -1  9968 4844  1076 R <    0   0  3.7  3.8   0:28 Xfbdev
1542 bccd      9   0  4452 4452  3632 S      0   0  1.7  3.5   0:03 openmosixa
1540 bccd      9   0  4516 4516  3884 S      0   0  0.9  3.5   0:02 openmosixm
```

PVM and the accompanying graphical visualization tool XPVM are based upon the process collection model of parallel computing, where interprocess communication is supported through socket-based connection. Users can interact with the PVM environment through the PVM console or from the command line for PVM-aware applications. Figure 3 shows PVM being used on the BCCD to join together four BCCD-based nodes in an illustration of PVM's group-based communication model. The graphical representation of the process is conveyed using XPVM. XPVM shows interprocess communication, processes blocking for messages, message queues, process computation, PVM overhead, and many additional PVM-related aspects of the parallel computation.

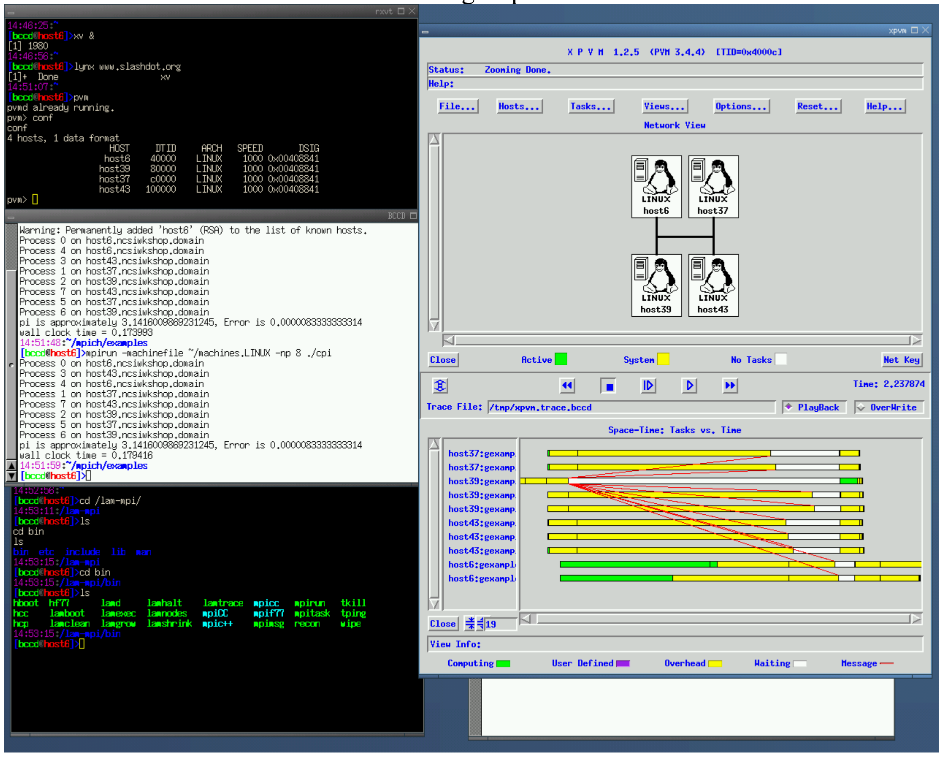


Figure 3: PVM being used on the BCCD to join together four BCCD-based nodes in an illustration of PVM's group-based communication model.

```
14:46:25:~
[bccd@host6]>xv &
[1] 1980
14:46:56:~
[bccd@host6]>lynx www.slashdot.org
[1]+  Done                    xv
14:51:07:~
[bccd@host6]>pvm
pvmd already running.
pvm> conf
conf
4 hosts, 1 data format
                    HOST     DTID     ARCH   SPEED       DSIG
                   host6    40000    LINUX    1000 0x00408841
                  host39    80000    LINUX    1000 0x00408841
                  host37    c0000    LINUX    1000 0x00408841
                  host43   100000    LINUX    1000 0x00408841
pvm> 
```

```
Warning: Permanently added 'host6' (RSA) to the list of known hosts.
Process 0 on host6.ncsiwkshop.domain
Process 4 on host6.ncsiwkshop.domain
Process 3 on host43.ncsiwkshop.domain
Process 1 on host37.ncsiwkshop.domain
Process 2 on host39.ncsiwkshop.domain
Process 7 on host43.ncsiwkshop.domain
Process 5 on host37.ncsiwkshop.domain
Process 6 on host39.ncsiwkshop.domain
pi is approximately 3.1416009869231245, Error is 0.0000083333333314
wall clock time = 0.173993
14:51:48:~/mpich/examples
[bccd@host6]>mpirun -machinefile ~/machines.LINUX -np 8 ./cpi
Process 0 on host6.ncsiwkshop.domain
Process 3 on host43.ncsiwkshop.domain
Process 4 on host6.ncsiwkshop.domain
Process 1 on host37.ncsiwkshop.domain
Process 7 on host43.ncsiwkshop.domain
Process 2 on host39.ncsiwkshop.domain
Process 5 on host37.ncsiwkshop.domain
Process 6 on host39.ncsiwkshop.domain
pi is approximately 3.1416009869231245, Error is 0.0000083333333314
wall clock time = 0.179416
14:51:59:~/mpich/examples
[bccd@host6]>
```

```
14:52:56:~
[bccd@host6]>cd /lam-mpi/
14:53:11:/lam-mpi
[bccd@host6]>ls
cd bin
ls
bin  etc  include  lib  man
14:53:15:/lam-mpi
[bccd@host6]>cd bin
14:53:15:/lam-mpi/bin
[bccd@host6]>ls
hboot  hf77      lamd     lamhalt    lamtrace  mpicc   mpirun   tkill
hcc    lamboot   lamexec  lamnodes   mpiCC     mpif77  mpitask  tping
hcp    lamclean  lamgrow  lamshrink  mpic++    mpimsg  recon    wipe
14:53:15:/lam-mpi/bin
[bccd@host6]>
```

X P V M 1.2.5 (PVM 3.4.4) [TID=0x4000c]

Status: Zooming Done.

Help:

Time: 2.237874

Trace File: /tmp/xpvm.trace.bccd

View Info:

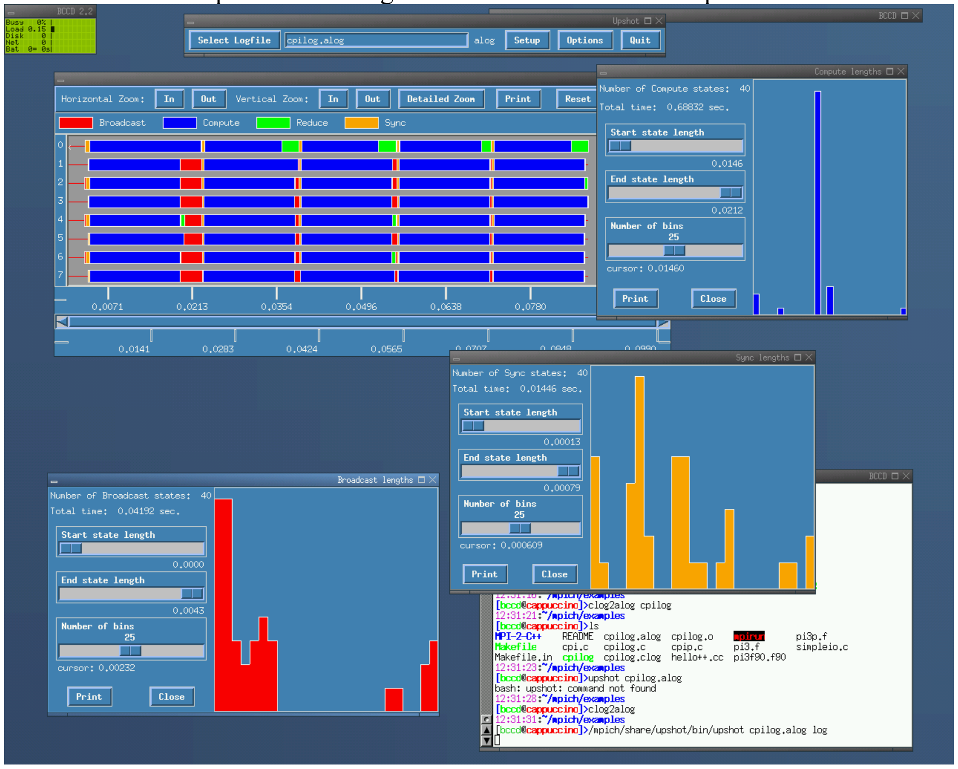
MPICH has a similar processed-based model of parallelism. Graphical tools to visualize MPICH programs include *upshot*. *Upshot* is based on tcl/tk. While the newer version of *upshot, jumpshot,* is part of the MPE package (the message-passing extensions of MPICH), it requires a specific version of Java. However, the standard JDK cannot be included on the BCCD due to licensing restrictions. Figure 4 shows *upshot* being used on the BCCD to show a parallelized ring-based communication example.

While MPICH is the default MPI environment of the BCCD, LAM-MPI is also available for those that prefer a daemon-based MPI environment. The BCCD includes XMPI, which is maintained by LAM-MPI developers. Scripts are provided on the BCCD image to switch between MPI environments so those that prefer one environment over the other are not constrained by the default choices. Figure 5 shows a LAM-MPI process as visualized through XMPI.

The Cluster Command and Control (C3) tools consist of numerous cluster administration scripts written in Python. Examples of the C3 tools include *cexec* (used to execute a process on all nodes in the cluster) and *cpush* (used to push files out to the nodes of the cluster). These tools help to main consistency across the nodes of the cluster when modifications are made to the environment.

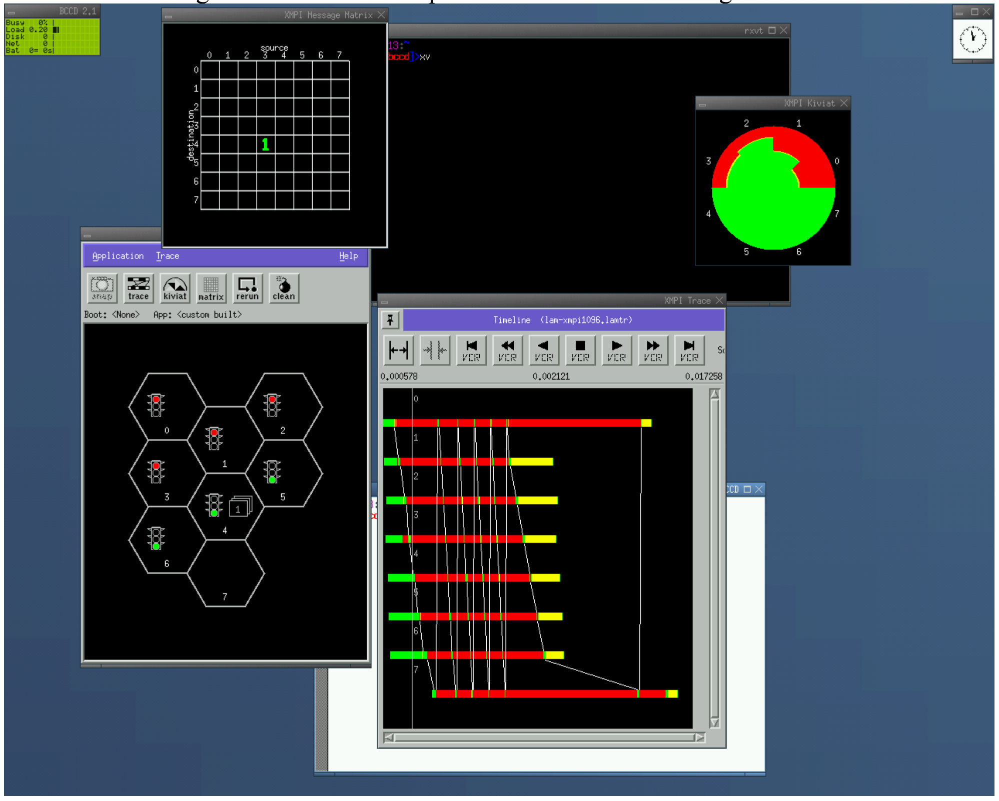
The BCCD includes the latest version of gcc 3.2.3. Included in the gcc package are the g77 (Fortran), gcj (Java), and g++ (C++) compilers. Optionally, the Ada compiler can be included if desired. These compilers provide the foundation for the MPI-based wrapper functions hcc, hcp, hf77, mpicc, mpi77, mpiCC, and mpic++.

Figure 4: *upshot* being used on the BCCD to show a parallelized ring-based communication example.



```
[bccd@cappuccino]>clog2alog cpilog
12:31:21:~/mpich/examples
[bccd@cappuccino]>ls
MPI-2-C++    README  cpilog.alog  cpilog.o     mpirun       pi3p.f
Makefile     cpi.c   cpilog.c     cpip.c       pi3.f        simpleio.c
Makefile.in  cpilog  cpilog.clog  hello++.cc   pi3f90.f90
12:31:23:~/mpich/examples
[bccd@cappuccino]>upshot cpilog.alog
bash: upshot: command not found
12:31:28:~/mpich/examples
[bccd@cappuccino]>clog2alog
12:31:31:~/mpich/examples
[bccd@cappuccino]>/mpich/share/upshot/bin/upshot cpilog.alog log
```

Figure 5: A LAM-MPI process as visualized through XMPI.



Usage of the BCCD

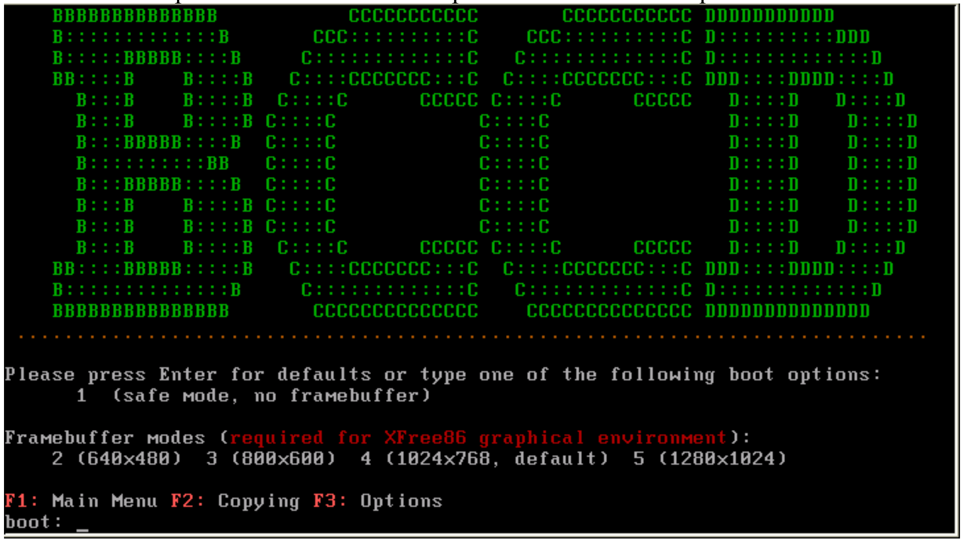
The BCCD has been used for several workshops on HPCE over the past year. Week-long workshops, supported by the National Computational Science Institute, were held at Washington University in St. Louis, Prairie View Texas A&M, and most recently during SuperComputing 2003 in Baltimore, Maryland. These workshops, sponsored by the National Computational Science Institute (NCSI), are targeted at instructors from PUI's and the upper-level high school grade levels. In the upcoming year, NCSI workshops in cluster computing education will take place at Bethune Cookman College, the University of Oklahoma, and SuperComputing 2004 (Pittsburgh).

The flexibility and reliability of the BCCD image was put to the test at SuperComputing 2003. Several afternoon workshops and tutorials on MPI programming were scheduled, but the MPI environment had not been configured on the machines being used in the tutorial rooms. These machines were laptops, configured to dual boot between Windows XP and RedHat 9.0. The difficulty in setting up the MPI environment centered around incompatibilities and missing features in the MPICH RPM packages and the RedHat 9.0 distribution. However, using the BCCD approach, these laptops were used to support a distributed-computing environment within minutes using the wireless networking infrastructure of the Baltimore Conference Center. During these workshops, the laptops were often reconfigured and the environment was repartitioned so as to support "miniclusters" of four and eight nodes. This repartitioning was used to isolate participant applications and to facilitate performance benchmarking. The wide variety of applications, including text editors, compilers, debuggers, graphical tools, customized scripts, and networking applications, made the BCCD approach an ideal environment for the SuperComputing 2003 tutorials.

The common usage scenario for using the BCCD as an instructional environment begins with a single instructor and a classroom of students, one per workstation. Each student begins by booting their workstation to the CD drive. If the system is not configured to boot to the CD drive, it is necessary to modify the system's BIOS. Or, if the system permits booting from a floppy drive, the student can create a floppy image that will boot the system to the CD drive. At the initial "splash screen" of the boot process, the user is offered a boot prompt. At this boot prompt, users can customize the boot-up behavior; specifying screen resolutions, kernel parameters, and different initialization levels. In most situations the default parameters are acceptable and the system boots automatically. Figure 6 shows the initial splash screen where users can enter these kernel parameters.

One aspect of the BCCD that is unique among the dynamic clustering environments mentioned above is the support of a user account. The BCCD environment does not enroll all capable hosts automatically. Rather, individual users must take a number of steps in order to enroll their resources into the clustering environment. This is especially significant for the instructor's resources. By adopting an opt-in approach to system resource utilization, users are not able to access or enroll the participation of the instructor's system (or each others systems) until explicitly permitted to do so. To support the separation of user privileges, the user is prompted for a password as part of the system initialization process. This password-protects the local system user account. In addition to password protection, unique DSA and RSA keys are generated for the user. These keys will subsequently be used to support the distribution of applications over the parallel environment. Figure 7 shows the generation of both the public and private key pairs and the solicitation of the user's password.

Figure 6: The initial splash screen where users may enter kernel parameters and indicate special initialization sequences .



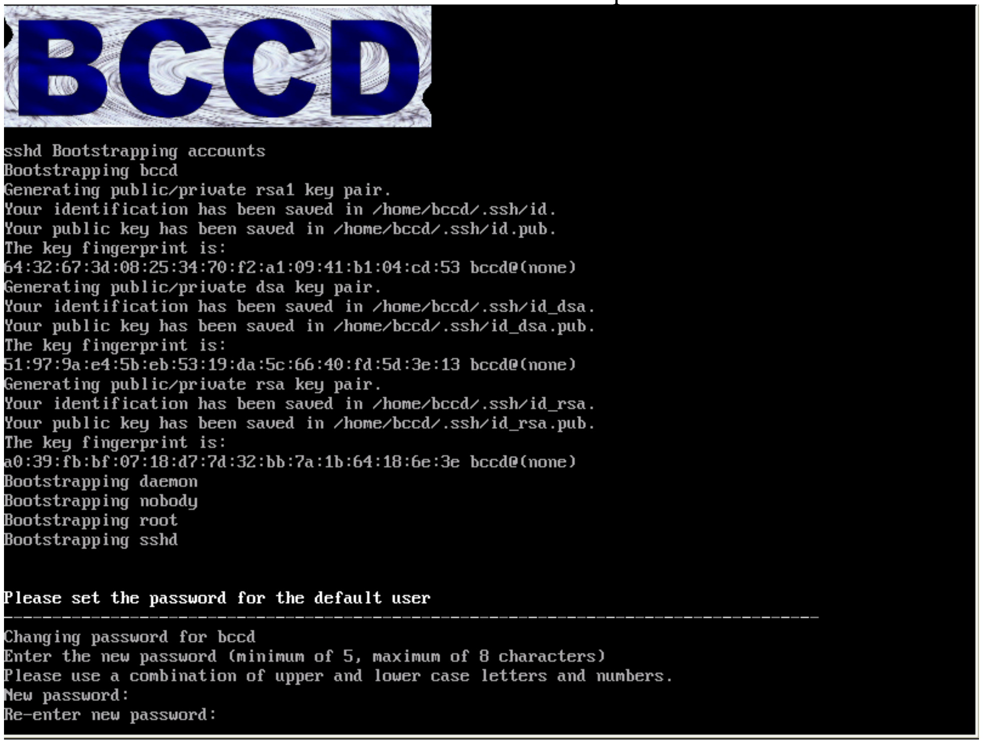
```
    BBBBBBBBBBBBBB           CCCCCCCCCCC       CCCCCCCCCCC DDDDDDDDDDD
    B:::::::::::::B       CCC::::::::::C    CCC::::::::::C D::::::::::DDD
    B:::::BBBBB::::B     C:::::::::::::C   C:::::::::::::C D:::::::::::::D
    BB::::B    B::::B   C::::CCCCCCC:::C  C::::CCCCCCC:::C DDD::::DDDD::::D
      B:::B    B::::B  C::::C      CCCCC C::::C      CCCCC   D::::D   D::::D
      B:::B    B::::B C::::C            C::::C               D::::D    D::::D
      B:::BBBBB::::B  C::::C            C::::C               D::::D    D::::D
      B::::::::::BB   C::::C            C::::C               D::::D    D::::D
      B:::BBBBB::::B  C::::C            C::::C               D::::D    D::::D
      B:::B    B::::B C::::C            C::::C               D::::D    D::::D
      B:::B    B::::B C::::C            C::::C               D::::D    D::::D
      B:::B    B::::B  C::::C      CCCCC C::::C      CCCCC   D::::D   D::::D
    BB::::BBBBB:::::B   C::::CCCCCCC:::C  C::::CCCCCCC:::C DDD::::DDDD::::D
    B::::::::::::::B     C:::::::::::::C   C:::::::::::::C D:::::::::::::D
    BBBBBBBBBBBBBBB       CCCCCCCCCCCCCC    CCCCCCCCCCCCCC DDDDDDDDDDDDDD
 .............................................................................

Please press Enter for defaults or type one of the following boot options:
      1  (safe mode, no framebuffer)

Framebuffer modes (required for XFree86 graphical environment):
    2 (640x480)  3 (800x600)  4 (1024x768, default)  5 (1280x1024)

F1: Main Menu F2: Copying F3: Options
boot: _
```

Figure 7: Generation of both the public and private key pairs and the solicitation of the user's password.



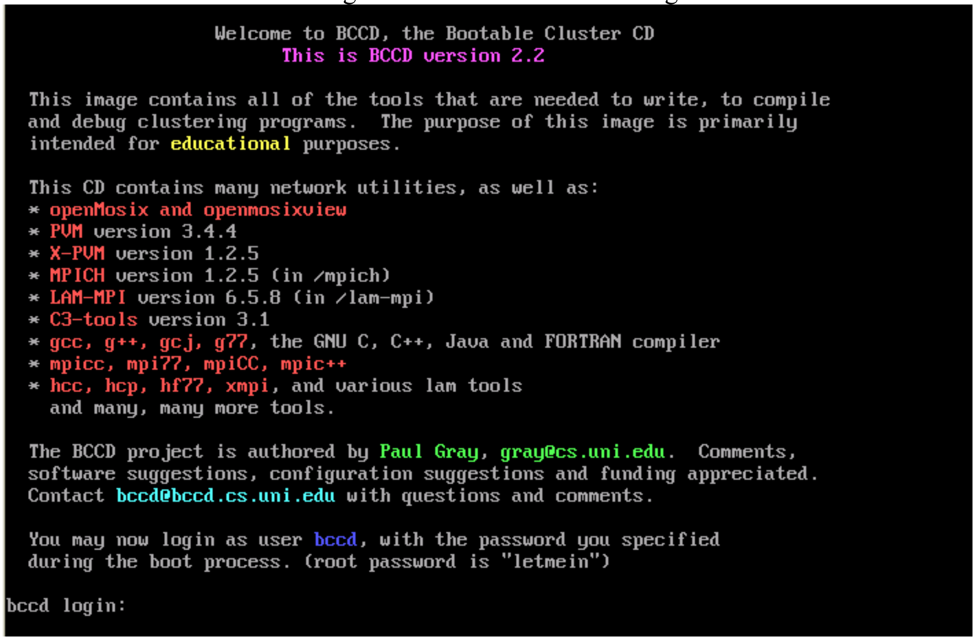
```
sshd Bootstrapping accounts
Bootstrapping bccd
Generating public/private rsa1 key pair.
Your identification has been saved in /home/bccd/.ssh/id.
Your public key has been saved in /home/bccd/.ssh/id.pub.
The key fingerprint is:
64:32:67:3d:08:25:34:70:f2:a1:09:41:b1:04:cd:53 bccd@(none)
Generating public/private dsa key pair.
Your identification has been saved in /home/bccd/.ssh/id_dsa.
Your public key has been saved in /home/bccd/.ssh/id_dsa.pub.
The key fingerprint is:
51:97:9a:e4:5b:eb:53:19:da:5c:66:40:fd:5d:3e:13 bccd@(none)
Generating public/private rsa key pair.
Your identification has been saved in /home/bccd/.ssh/id_rsa.
Your public key has been saved in /home/bccd/.ssh/id_rsa.pub.
The key fingerprint is:
a0:39:fb:bf:07:18:d7:7d:32:bb:7a:1b:64:18:6e:3e bccd@(none)
Bootstrapping daemon
Bootstrapping nobody
Bootstrapping root
Bootstrapping sshd


Please set the password for the default user
--------------------------------------------------------------------------------
Changing password for bccd
Enter the new password (minimum of 5, maximum of 8 characters)
Please use a combination of upper and lower case letters and numbers.
New password:
Re-enter new password:
```

Another part of the initialization process revolves around configuration of the network, which is a critical component to support of distributed computing. Following configuration of the network, the user is prompted to log in with the previously-specified password. Upon logging in for the first time, the user is prompted to start the BCCD *pkbcast* (public key broadcast) program as part of the cluster pre-configuration process.

Once logged in, the user has numerous example programs already located in the home directory. If a graphical mode is desired, the user may issue *startx* at the command line to bring up the X Window system. In the X Window system, numerous graphical editors and distributed systems profiling applications are available through the menu system with a simple mouse click on the desktop. Figure 8 shows the login prompt along with the '/etc/issue' message of the standard BCCD image.

Figure 8: The login prompt along with the '/etc/issue' message of the standard BCCD image.



```
                   Welcome to BCCD, the Bootable Cluster CD
                         This is BCCD version 2.2

  This image contains all of the tools that are needed to write, to compile
  and debug clustering programs.  The purpose of this image is primarily
  intended for educational purposes.

  This CD contains many network utilities, as well as:
  * openMosix and openmosixview
  * PVM version 3.4.4
  * X-PVM version 1.2.5
  * MPICH version 1.2.5 (in /mpich)
  * LAM-MPI version 6.5.8 (in /lam-mpi)
  * C3-tools version 3.1
  * gcc, g++, gcj, g77, the GNU C, C++, Java and FORTRAN compiler
  * mpicc, mpi77, mpiCC, mpic++
  * hcc, hcp, hf77, xmpi, and various lam tools
    and many, many more tools.

  The BCCD project is authored by Paul Gray, gray@cs.uni.edu.  Comments,
  software suggestions, configuration suggestions and funding appreciated.
  Contact bccd@bccd.cs.uni.edu with questions and comments.

  You may now login as user bccd, with the password you specified
  during the boot process. (root password is "letmein")

bccd login:
```

By default, a user's workstation resources are isolated. No user is permitted to access the local resources until the local user extends access permissions. Access permissions are granted through public/private key pairs as mentioned earlier. Distribution of the public keys occurs through the *pkbcast* program started upon the user's initial logon. There are several BCCD-provided scripts that automate this process of extending access rights to remote users. In this way, very diverse clustering configurations are possible.

Once the environment has been established, the BCCD environment is completely ready for utilization.

Summary

In summary, with its vast hardware support, acclimation to various networks, simple configurability, excellent pre-configured demonstrations, and ease of customization, the BCCD is an all-inclusive, easy solution to the task of teaching HPC concepts. At a community college or PUI where lack of dedicated clustering tools and lack of full-time support and administration exist, the BCCD is an easy, non-invasive, drop-in solution for support of HPC education.

References

1. Bar, M. (2003). *Linux Clusters State of the Art.* Online document available at http://openmosix.sourceforge.net.
2. Blomgren, M. and Mota, M. Jr. (2004) *openMosixLoaf.* Information available at the project web site http://openmosixloaf.sourceforge.net.
3. Burns, G. Daoud, R. and Vaigl, J. (1994) *LAM: An Open Cluster Environment for MPI*, Supercomputing Symposium. Toronto, Canada.
4. Chang, C. Et. Al. (2001) *Computing Curricula 2001; Computer Science, Final Report*. IEEE/ACM Curricular Standards available for download at http://www.sigcse.org/cc2001.
5. Des Ligneris, B., Scott, S., Naughton, T. and Gorsuch, N. (2003) *Open Source Cluster Application Resources (OSCAR): Design, implementation and interest for the [computer] scientific community.* Proceedings of the First OSCAR Symposium (Sherbrook, May 2003).
6. Geist, G. A., and Sunderam, V. S. (1991) *The PVM System: Supercomputer Level Concurrent Computations on a Heterogeneous Network of Workstations.* Proceedings of the Sixth Distributed Memory Computing Conference. IEEE pp 258—261.
7. Gropp, W. Lusk, E. and Skjellum, A. (1996) *A High-Performance, Portable implementation of the MPI Message Passing Interface Standard.* Parallel Computing 22, No. 6, pp 789—828.
8. Latter, I (2003) *Running ClusterKnoppix as a Head Node to a CHAOS Drone Army.* Tech Report, Macquarie University, AU. ITSecurity Group Publication.
9. Moffit, N. Et. Al. (2004) *The LNX-BBC Project*. Information available at the project web site http://lnx-bbc.org.
10. Rechenburg, M. (2003) *openMosixview* Online document available at http://www.openmosixview.com
11. SDSC (UCSD) and Millennium Group (Berkeley) (2003) *NPACI Rocks*. Information available at the project web site http://www.rocksclusters.org/Rocks/.
12. Vandersmissen, W. (2004) *ClusterKnoppix.* Information available at the project web site http://bofh.be/clusterknoppix.
13. Willis, P. (2004) *PlumpOS*. Information available at the project web site http://plumpos.soureceforge.net.

Acknowledgements

This work has been supported in part by a grant from the National Computational Science Institute (NCSI).