Introduction To HPCC

Faculty Seminars in Research and Instructional Technology

May 6, 2014

https://wiki.hpcc.msu.edu/x/m4E_AQ

Dirk Colbry

colbrydi@msu.edu

Director, High Performance Computing Center

Institute for Cyber-Enabled Research
All Day Agenda

• Introduction to HPCC
  – Introduction to iCER
  – The Seven Steps to using the HPCC

• Advanced HPCC, Doing more faster
  – Powertools
  – Pleasantly Parallel
  – Shared Memory Parallelization
  – Shared Network Parallelization
How this workshop works

• We are going to cover some basics. Lots of hands on examples.
• When you get tired of listening to me talk, skip ahead to an exercise and give it a try.
• Exercises are denoted by the following icon in your notes:
Red and Green Flags

• Use the provided sticky notes to help me help you.
  – **NO Sticky** = I am working
  – **Green** = I am done and ready to move on
  – **Red** = I am stuck and need more time and/or I could use some help
What is Advanced Computing Hardware?

• Anything more advanced than your desktop
• Local resources
  – Lab, Department, Institution (HPCC)
• National resources
  – NSF (XSEDE, Blue Waters), DOE (Jaguar), Others
• Commercial Resources (cloud computing)
  – Amazon, Azure, Liquid Web, Others
Why use Advanced Computing Hardware?

- Science takes too long
- Computation runs out of memory
- Access to software
- Need advanced interface (visualization)
Types of problems

• CPU bound
  – Lots of computing (think simulation)

• Memory bound
  – Requires lots of memory (think genomics)

• I/O bound
  – Requires lots of data (think astronomy)

(many problems fall in more than one category)
1957 MISTIC Mainframe

- MSU’s first mainframe
- Hand built by grad students
  - Dick Reid
  - Glen Keeney
After MISTIC

• 1957 MISTIC
• 1963-1973 CDC 3600
• 1967 Computer Science Department
• 1968 CDC 6500
• 1971 MERIT
• 1978 Cyber 750
• 2004 HPCC
• 2009 ICER
2004 HPCC

• Provide a level of performance beyond what you could get and reasonably maintain as a small group
• Provide a variety of technology, hardware and software, that would allow for innovation not easily found
2005
- **amd05**: 512 core, 128 node cluster installed in 2005. Each node contains four 2.2 GHz AMD Opteron cores, 8 GB of RAM, and 146 GB of local disk.

2005
- **green**: 128 core, SMP system purchased in 2005 and upgraded in 2007 to its current configuration of 128 processors, 512GB RAM, and 6.4 TB of high-speed disk.

2007
- **intel07**: 1024 core, 128 node cluster installed in 2007. Each node contains eight 2.3 GHz Intel Xeon cores, 8 GB of RAM, and 250 GB of local disk.

2009
- **Gfx10**: 256 core, 32 node graphics cluster installed in 2010. Each graphics node contains two nVidia Tesla M1060 GPGPU accelerators with 240 GPU cores and 4GB GPU Ram each, eight 2.4 GHz Intel Xeon cores, 18 GB of RAM, and 250 GB of local hard disk.

2010
- **intel10**: 1504 core, 188 node cluster installed in 2010. Each node contains eight 2.4 GHz Intel Xeon cores, 24 GB of RAM, and 250 GB of local disk.

2011
- **intel11**: Mixed 512GB - 2TB RAM nodes with 32-64 core 2.66 GHz Xeon E7-8837 processors.
The Institute for Cyber Enabled Research (iCER) at Michigan State University (MSU) was established to coordinate and support multidisciplinary resource for computation and computational sciences. The Center's goal is to enhance MSU's national and international presence and competitive edge in disciplines and research thrusts that rely on advanced computing.
Bigger Science

• The goal of iCER is **NOT**:  
  – Kflops  
    (floating point operations per second)

• Instead, the goal of iCER **IS**:  
  – KSciences / second

• Doing More Science, Faster  
  – Reducing the “Mean time to Science”

• iCER is designed to help researchers do their science and when appropriate scale them up to one of the national labs
HPC Systems

• Large Memory Nodes (up to 2TB!`)
• GPU Accelerated cluster (K20, M1060)
• PHI Accelerated cluster (5110p)
• Over 540 nodes, 10000 computing cores
• Access to high throughput condor cluster
• 363TB high speed parallel scratch file space
• 50GB replicated file spaces
• Access to large open-source software stack and specialized bioinformatics VMs
Free Access to software

- Compiled open-source software stack
  - over 500 titles!
- Optimized Math/Communications libraries
- Some commercial software available
  - E.g. Ansys, MATLAB (+many toolboxes), Stata, Gaussian, SAS
Buy-In Opportunities

- We will maintain your computers for you
- Researchers get exclusive use of their nodes within 4 hours of submitting a job
- Buy-in jobs will automatically overflow into the general resources.
2014 Cluster Buy-in

- intel14 Option 1 (20 core 64gb base node): $3805.82
- intel14 Option 2 (20 core 256gb large memory node): $5338.46
- intel14 Option 3 (20 core 128gb K20 GPU node): $7899.15
- intel14 Option 4 (20 core 128gb Phi Node): $9042.91
- intel14 Chassis: $1216.44

For ~$10,000 we can also upgrade any of these options to 512gb of memory

More information: https://wiki.hpcc.msu.edu/x/dwH3
2014 Large Memory Buy-in

- Large Memory Option 1 (1TB RAM, 48 cores): $29,979
- Large Memory Option 2 (1.5 TB RAM, 48 cores): $34,989
- Large Memory Option 3 (3 TB RAM, 48 cores): $60,995
- Large Memory Option 4 (6 TB RAM, 96 cores): $142,772

- Limited time only

- https://wiki.hpcc.msu.edu/x/L4E_AQ
What if I want more?
Online Resources

• icer.msu.edu - iCER Home
  – hpcc.msu.edu – HPCC Home

• wiki.hpcc.msu.edu – HPCC User Wiki
iCER Lightening Talks

• May 19, 2014
  – 10:00-12:00 Biomedical and Physical Sciences Building (1445A)
  – Lots of short talks about new services provided by iCER.
VSCSE

- Harness the Power of GPU’s: Introduction to GPGPU Programming
  - June 16-20, 2014
- Data Intensive Summer School
  - June 30 – July 2, 2014

- $100 per class ($50 for MSU affiliated individuals)
- https://wiki.hpcc.msu.edu/x/Z4E_AQ
Seven Steps to using the HPCC (The Basics)

http://www.softwarecarpentry.org/
Steps in Using the HPCC

1. Get an account
2. Install needed software (SSH, SCP, X11)
3. Transfer input files and source code
4. Compile/Test programs on a developer node
5. Write a submission script
6. Submit the job
7. Get your results and write a paper!!
Accounts

• PIs must request accounts for students:
  – http://www.hpcc.msu.edu/request

• Each user has 50Gigs of backed-up personal hard drive space.
  – /mnt/home/username/

• Users have access to 363TB of high speed parallel scratch space.
  – /mnt/scratch/username/

• Shared group space is also available upon request.
Steps in Using the HPCC

1. Get an account
2. Install needed software (SSH, SCP, X11)
3. Transfer input files and source code
4. Compile/Test programs on a developer node
5. Write a submission script
6. Submit the job
7. Get your results and write a paper!!
URLs for Software (windows)

- PuTTY:
  - http://www.chiark.greenend.org.uk/~sgtatham/putty/
- Xming:
  - http://www.straightrunning.com/XmingNotes/
- Xming install:
  - https://wiki.hpcc.msu.edu/x/swAk
- WinSCP:
  - http://winscp.net
MobaXterm

• Complete toolbox for remote computing:
  – Multi-tab terminal
  – X11 server
  – SSH
  – File transfer
  – More

• Opensource
• http://mobaxterm.mobatek.net/
Exercise: Portable HPCC

- Plug in your USB thumb drive
- Open the thumb drive folder and select – PortableApps
- You should see a new menu in your system tray for navigating
Exercise: Connect to HPCC

• Step 1: Use Xming on your thumb drive to log into gateway.hpcc.msu.edu

• Step 2: ssh into a dev node (developer node)
  > ssh dev-intel10
Command Line Interface

• Command Line Interface (CLI)
• Shell
  – Program to run Programs
• Bash (Bourne Again Shell)
• Use it because:
  – many tools only have command-line interfaces
  – allows you to combine tools in powerful new ways
Shell Navigation

• Basic Navigation commands:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pwd</code></td>
<td>print working directory</td>
</tr>
<tr>
<td><code>cd</code></td>
<td>change working directory</td>
</tr>
<tr>
<td><code>ls</code></td>
<td>list directory</td>
</tr>
</tbody>
</table>

• Use the following symbols to indicate special directories:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>current directory</td>
</tr>
<tr>
<td>..</td>
<td>parent directory</td>
</tr>
<tr>
<td>~</td>
<td>home directory</td>
</tr>
<tr>
<td>-</td>
<td>previous directory</td>
</tr>
</tbody>
</table>
Exercise – Shell Navigation

• Show the path to the current directory
  > pwd

• Change to the scratch directory
  > cd /mnt/scratch/

• List the contents of the current directory:
  > ls

• Change back to home
  > cd ~
Directories

/mnt/home/coblrydi
Man Pages

• “Manual” pages.
• Type “man” and then the command name
• Example:
  \texttt{>man qsub}
• Use “q” key to quick out of the man program
Example: File Manipulation

- **Try Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mkdir</code></td>
<td>make directory</td>
</tr>
<tr>
<td><code>cp</code></td>
<td>copy file</td>
</tr>
<tr>
<td><code>cat</code></td>
<td>display contents of text file</td>
</tr>
<tr>
<td><code>rm</code></td>
<td>remove file</td>
</tr>
</tbody>
</table>

- **See the contents of your ".bashrc" file**

  `> cat .bashrc`

- **Make a directory called "hpccworkshop", change to that directory and list the contents.**

  `> mkdir hpccworkshop`
  `> cd ./hpccworkshop`
Available Software

• Center Supported Development Software
  – Intel compilers, openmp, openmpi, mvapich, totalview, mkl, pathscale, gnu, ...

• Center Supported Research Software
  – MATLAB, R, fluent, abaqus, HEEDS, amber, blast, ls-dyna, starp...

• Customer Software
  – gromacs, cmake, cuda, imagemagick, java, openmm, siesta...

  – For a more up to date list, see the documentation wiki:
    • http://wiki.hpcc.msu.edu/
Module System

- To maximize the different types of software and system configurations that are available to the users, HPCC uses a Module system

- Key Commands
  - `module avail` – show available modules
  - `module list` – list currently loaded modules
  - `module load` modulename – load a module
  - `module unload` modulename – unload a module
  - `module spider keyword` – Search modules for a keyword
Exercise – Module

• List loaded modules
  >>> module list

• Show available modules:
  >>> module avail

• Try an example (Shouldn’t work):
  >>> powertools
Exercise: getexample

• Load a newly available module:
  > module load powertools

• Show powertools (should work now):
  > powertools

• Run the “getexample” powertool
  > getexample

• Download the helloMPI example
  > getexample helloworld
Standard in/out/err and piping

• You can redirect the output of a program to a file using “>” greater than character:
  – myprogram > output.txt

• You can also cause the output of the program to be the input of another program using the “|” pipe character:
  – myprogram | myotherprogram
Exercise: Redirection and Piping

- Change to the helloworld directory:
  > cd ~/hpccworkshop/helloworld
  > ls -la

- Redirect the output of the ls command:
  > ls -la > numOfLines
  > cat numOfLines

- Pipe Commands together

Easy command to calculate the number of lines of code in your programs
Steps in Using the HPCC

1. Get an account
2. Install needed software (SSH, SCP, X11)
3. **Transfer input files and source code**
4. Compile/Test programs on a developer node
5. Write a submission script
6. Submit the job
7. Get your results and write a paper!!
SCP/SFTP – Secure File transfer

• WinSCP for Windows
• Command-line “scp” and “sftp” on Linux
• Many other scp and sftp clients out there as well
• Functions over SSHv2 protocol, very secure
Exercise: Transfer a file

• Make a file called `minlines` using notepad++ on your thumb drive
• Put in the following line:
  
  ```
  wc -l * | sort -n | head -1
  ```

• Open WinSCP on your thumb drive
• Copy the file `minlines` to the `helloworld` directory
## File Permissions

<table>
<thead>
<tr>
<th></th>
<th>user</th>
<th>group</th>
<th>all</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>read</strong></td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td><strong>write</strong></td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td><strong>execute</strong></td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
Permissions

- Common Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chmod</td>
<td>Change permissions (change mode)</td>
</tr>
<tr>
<td>ls -al</td>
<td>List all long (including permissions)</td>
</tr>
</tbody>
</table>
Example: permissions

- Change to the helloMPI directory
  
  \texttt{cd ~/hpccworkshop/helloworld}

- Show current permissions
  
  \texttt{ls -la}

- Make the \texttt{minlines} file executable
  
  \texttt{chmod u+x minlines}

- Check permissions again
  
  \texttt{ls -la}

- Now you can run minlines as a command
  
  \texttt{./minlines}
Environment Variables

- Scripts also let you use environment variables
- These variables can be used by your script or program
- Use “export” and = to set a variable
- Use the $ and {} to display the contents of a variable
Example: Environment Variables

• Display all environment variables
  ➜ `env`

• Display specific environment variable
  ➜ `echo ${MACHTYPE}`

• Make a new variable
  ➜ `export MYVAR="Hello World"`

• Use your variable
  ➜ `echo ${MYVAR}`
Steps in Using the HPCC

1. Get an account
2. Install needed software (SSH, SCP, X11)
3. Transfer input files and source code
4. **Compile/Test programs on a developer node**
5. Write a submission script
6. Submit the job
7. Get your results and write a paper!!
Running Jobs on the HPC

• Submission scripts are used to run jobs on the cluster
• The developer (dev) nodes are used to compile, test and debug programs

• However, the developer nodes are powerful systems too. **We don’t want to waste their compute power.**
Advantages of running Interactively

• You do not need to write a submission script
• You do not need to wait in the queue
• You can provide input to and get feedback from your programs as they are running
Disadvantages of running Interactively

• All the resources on developer nodes are shared between all users.
• Any single process is limited to 2 hours of cpu time. If a process runs longer than 2 hours it will be killed.
• Programs that overutilize the resources on a developer node (preventing other to use the system) can be killed without warning.
## Developer Nodes

<table>
<thead>
<tr>
<th>Name</th>
<th>Cores</th>
<th>Memory</th>
<th>Accelerators</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>dev-intel07</td>
<td>8</td>
<td>8GB</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>dev-gfx08</td>
<td>4</td>
<td>8GB</td>
<td>3 x M1060</td>
<td>Nvidia Graphics Node</td>
</tr>
<tr>
<td>dev-intel10</td>
<td>8</td>
<td>24GB</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>dev-intel14</td>
<td>20</td>
<td>64GB</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>dev-intel14-phi</td>
<td>20</td>
<td>128GB</td>
<td>2 x Phi</td>
<td>Xeon Phi Node</td>
</tr>
<tr>
<td>dev-intel14-k20</td>
<td>20</td>
<td>128GB</td>
<td>2 x K20</td>
<td>Nvidia Graphics Node</td>
</tr>
</tbody>
</table>
Compiling

• Most users use the developer nodes for developing their software.

• If you are using a makefile you can compile using more processors with the –j option.
  – make –j32
  – Will make with 32 core threads
    • (use this on dev-amd09)
Compilers

• By default we use the gnu compilers. However, lots of other compilers are available including Intel and Portland compilers.

• The module system always sets environment variables such that you can easily test with other compilers.
  – ${CC}$
  – ${FC}$
  – Etc.
Exercise: Compile Code

• Make sure you are in the helloworld directory:
  > pwd

• Run the gcc compilers:
  > ${CC} -O3 -o hello hello.c

• Run the program:
  > ./hello
Running in the background

- You can run a program in the background by typing an “&” after the command.
- You can make a program keep running even after you log out of your ssh session by using “nohup command”
- You can run an entire session in the background even if you log in and out of your ssh session by using the “screen” or “tmux” commands
- All three of these options are common to linux and tutorials can be found online
CLI vs GUI

• CLI – Command Line Interface

• GUI – Graphical User Interface
What is X11?

- Method for running Graphical User Interface (GUI) across a network connection.

![Diagram showing SSH and X11 connections between a personal computer running x11 server and a cluster.](image_url)
What is needed for X11

- X11 server running on your personal computer
- SSH connection with X11 enabled
- Fast network connection
  - Preferably on campus
Graphical User Interface

• X11 Windows: Install Xming
  – Installation instructions at: https://wiki.hpcc.msu.edu/x/swAk
• ssh -X username@hpc.msu.edu
• Turn on x11 forwarding

• Note: Mac Lion Users should use XQuartz
  http://xquartz.macosforge.org/
Exercise: Transfer a file

- Try one of the following Commands

<table>
<thead>
<tr>
<th>xeyes</th>
<th>Test X11</th>
</tr>
</thead>
<tbody>
<tr>
<td>firefox</td>
<td>Web browser</td>
</tr>
</tbody>
</table>

> xeyes

> firefox &

> ps <- Find the process ID #### for firefox

> kill ####
Programs that can use X11

• R - statistical computing and graphics
• firefox – Web browser
• totalview – C/C++/fortran debugger
• gedit, gvim, emacs – Text editors
• And others...
HPCC Portals

• (coming soon) Evaluation Node portal
• (coming soon) New X11 Portal using XRDP
• (limited, more coming soon) Galaxy Portals
• (limited, more coming soon) Virtual Compute Lab (VCL)
• (coming soon) Material Studio
• (Available now) Medea
Steps in Using the HPCC

1. Get an account
2. Install needed software (SSH, SCP, X11)
3. Transfer input files and source code
4. Compile/Test programs on a developer node
5. **Write a submission script**
6. Submit the job
7. Get your results and write a paper!!
Resource Manager and scheduler

Moab \rightarrow Torque

Not First In First Out!!
Schedulers vs Resource Managers

• Scheduler (Moab)
  – Tracks and assigns
    • Memory
    • CPUs
    • Disk space
    • Software Licenses
    • Power / environment
    • Network

• Resource Manager (PBS/Torque)
  – Hold jobs for execution
  – Put the jobs on the nodes
  – Monitor the jobs and nodes
Common Commands

• **qsub** `<Submission script>`
  – Submit a job to the queue
• **qdel** `<JOB ID>`
  – Delete a job from the queue
• **showq** `–u <USERNAME>`
  – Show the current job queue
• **checkjob** `<JOB ID>`
  – Check the status of the current job
• **showstart** `–e all <JOB ID>`
  – Show the estimated start time of the job
Submission Script

1. List of required resources
2. All command line instructions needed to run the computation
Typical Submission Script

```
#!/bin/bash -login
#PBS -l walltime=10:00:00,mem=3Gb,nodes=10:ppn=1
#PBS -j oe

cd ${PBS_O_WORKDIR}
./myprogram -my input arguments
qstat -f ${PBS_JOBID}
```
Example: Submit a job

• Go to the top helloworld directory
  
  > cd ~/hpccworkshop/helloworld

• Create a simple submission script
  
  > nano hello.qsub

• See next slide for what to type...
#!/bin/bash -login
#PBS -l walltime=00:01:00
#PBS -l nodes=1:ppn=1,feature=gbe

cd ${PBS_O_WORKDIR}
./hello

qstat -f ${PBS_JOBID}
Steps in Using the HPCC

1. Get an account
2. Install needed software (SSH, SCP, X11)
3. Transfer input files and source code
4. Compile/Test programs on a developer node
5. Write a submission script
6. Submit the job
7. Get your results and write a paper!!
Submitting a job

• `qsub` –arguments `<Submission Script>`
  – Returns the job ID. Typically looks like the following:
    • 5945571.cmgr01

• Time to job completion
Example: Submit a job, cont.

• Submit the file to the queue
  > qsub hello.qsub

• Record jobid number (#####) and wait at most 30 seconds

• Check the status of the queue
  > showq
Example: Monitor a job

- Submit the file to the queue:
  
  ```bash
  > qstat -f #######
  ```

- When will a job start:
  
  ```bash
  > showstart -e all #######
  ```
Scheduling Priorities

• Jobs that use more resources get higher priority (because these are hard to schedule)

• Smaller jobs are backfilled to fit in the holes created by the bigger jobs

• Eligible jobs acquire more priority as they sit in the queue

• Jobs can be in three basic states:
  — Blocked, eligible or running
## Cluster Resources

<table>
<thead>
<tr>
<th>Year</th>
<th>Name</th>
<th>Description</th>
<th>ppn</th>
<th>Memory</th>
<th>Nodes</th>
<th>Total Cores</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007</td>
<td>intel07</td>
<td>Quad-core 2.3GHz Intel Xeon E5345</td>
<td>8</td>
<td>8GB</td>
<td>126</td>
<td>1008</td>
</tr>
<tr>
<td>2009</td>
<td>amd09</td>
<td>Sun Fire X4600 (Fat Node) AMD Opteron 8384</td>
<td>32</td>
<td>256GB</td>
<td>3</td>
<td>96</td>
</tr>
<tr>
<td>2010</td>
<td>gfx10</td>
<td>NVIDIA CUDA Node (no IB)</td>
<td>8</td>
<td>18GB</td>
<td>32</td>
<td>256</td>
</tr>
<tr>
<td>2010</td>
<td>intel10</td>
<td>Intel Xeon E5620 (2.40 GHz)</td>
<td>8</td>
<td>24GB</td>
<td>191</td>
<td>1528</td>
</tr>
<tr>
<td>2011</td>
<td>intel11</td>
<td>Intel Xeon 2.66 GHz E7-8837</td>
<td>32</td>
<td>512GB</td>
<td>2</td>
<td>64</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1TB</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2TB</td>
<td>2</td>
<td>128</td>
</tr>
<tr>
<td>2014</td>
<td>intel14</td>
<td>Intel Xeon E5-2670 v2 (2.6 GHz)</td>
<td>20</td>
<td>64GB</td>
<td>128</td>
<td>2560</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>256GB</td>
<td>24</td>
<td>480</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 NVIDIA K20 GPUs</td>
<td>20</td>
<td>128GB</td>
<td>40</td>
<td>800</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 Xeon Phi 5110P</td>
<td>20</td>
<td>128GB</td>
<td>28</td>
<td>560</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td></td>
<td><strong>7512</strong></td>
</tr>
</tbody>
</table>
System Limitations

• Scheduling
  – 5 eligible jobs at a time
  – 512 running jobs
  – 1000 submitted jobs

• Resources
  – 1 week of walltime
  – 500 cores (nodes * ppn)
  – ppn=64
  – 2TB memory on a single core
  – ~200 GB Hard Drive
Job completion

• By default the job will automatically generate two files when it completes:
  – Standard Output:
    • Ex: jobname.o5945571
  – Standard Error:
    • Ex: jobname.e5945571
• You can combine these files if you add the join option in your submission script:
  – “#PBS -j oe”
• You can change the output file name
  – #PBS -o /mnt/home/netid/myoutputfile.txt
Other Job Properties

- resources (-l)
  - Walltime, memory, nodes, processor, network, etc.

- #PBS –l feature=gpgpu,gbe
- #PBS –l nodes=2:ppn=8:gpu=2
- #PBS –l mem=16gb

- Email address (-M)
  - Ex: #PBS –M colbrydi@msu.edu

- Email Options (-m)
  - Ex: #PBS –m abe

Many others, see the wiki:
http://wiki.hpcc.msu.edu/
Requesting local disk

• Sometimes (not often) local disk is faster than scratch

• Users can use the following resource to request temporary local disk space:
  – #PBS -l file=10gb

• The directory to access this disk space is determined by the one time use environment variable
  – ${TMPDIR}
Advanced Environment Variables

• The scheduler adds a number of environment variables that you can use in your script:
  – PBS_JOBID
    • The job number for the current job.
  – PBS_O_WORKDIR
    • The original working directory which the job was submitted

Ex:

```bash
mkdir ${PBS_O_WORKDIR}/${PBS_JOBID}
```
Steps in Using the HPCC

1. Get an account
2. Install needed software (SSH, SCP, X11)
3. Transfer input files and source code
4. Compile/Test programs on a developer node
5. Write a submission script
6. Submit the job
7. Get your results and write a paper!!
Steps in Using the HPCC

1. Get an account
2. Install needed software (SSH, SCP, X11)
3. Transfer input files and source code
4. Compile/Test programs on a developer node
5. Write a submission script
6. Submit the job
7. Get your results and write a paper!!
Getting Help

• Documentation and User Manual – wiki.hpcc.msu.edu
• Contact HPCC and iCER Staff for:
  – Reporting System Problems
  – HPC Program writing/debugging Consultation
  – Help with HPC grant writing
  – System Requests
  – Other General Questions
• Primary form of contact - http://contact.icer.msu.edu/
• HPCC Request tracking system – rt.hpcc.msu.edu
• HPCC Phone – (517) 353-9309
• HPCC Office – 1400 PBS
• Open Office Hours – 1pm Monday (PBS 1440)