UNIVERSITY of CALIFORNIA · IRVINE

BioLinux on HPC

Bio: Jenny Wu
jiew5@uci.edu

Linux: Harry Mangalam
harry.mangalam@uci.edu

Linux: Adam Brenner
aebrenne@uci.edu

Good Judgment comes from Experience

Experience comes from Bad Judgment

Before We Begin

- You know Linux at a user level
- You're bright: can Google, and read further by yourself.
- You know how to tell useful info from pure fantasy.
- If I speak to fast; let me know
- Questions, ASK THEM, but I may not answer them immediately. – "*You don't know what you don't know*"

Computing Philosophy

- Be lazy.
- Copy others.
- Don't invent anything you don't have to.
- Re-USE, re-CYCLE, DON'T re-invent.
- Don't be afraid to ask others.
- Resort to new code only when absolutely necessary.
  - Add comments to your code - ALWAYS

Philosophy – Take Away

- You're not CS, not programmers
- Don't try to be them
- But! Try to think like them, at least a bit
- Google is your friend

Getting Help

- Fix IT Yourself with Google <goo.gl/05MnTi>
- Listservs, forums, IRCs are VERY useful for more involved questions
- The HPC HOWTO <goo.gl/kzlqI>
- Us – Jenny, Adam, Harry, Joseph.
- BUT!! Unless you ask questions intelligently, you will get nothing but grief.

How to Ask Questions

- Reverse the situation: if you were answering the question, what information would you need?
- Not Science, but it is Logic.
- Include enough info to recreate the problem.
- Exclude what's not helpful or ginormous (use <pastie.org> or <tny.cz>)
- Use text, not screenshots if possible.

Bad Question

Why doesn’t “X” work?

Good Question

I tried running the new podunk/2.8.3 module this morning and it looks like I can't get it to launch on the Free64 queue. My output files aren't helping me figure out what is wrong.

I am working out of
/bio/joeuser/RNA_Seq_Data/M_sexta_RNAseq
and the qsub script is 'job12.sh'

When I submit the job, it appears to go thru the scheduler but then dies immediately when it hits the execution node.

I can't find any output to tell me what's wrong.

HELP US HELP YOU

- the directory in which you’re working (*pwd*)

- the machine you’re working on (*hostname*)

- modules loaded (*module list*)

- computer / OS you’re connecting from

- the command you used and the error it
  Caused (in text)

- much of this info is shown by a decent prompt

On to HPC

What is the H$_{\text{igh}}$ P$_{\text{erformance}}$ C$_{\text{omputing}}$ Cluster?

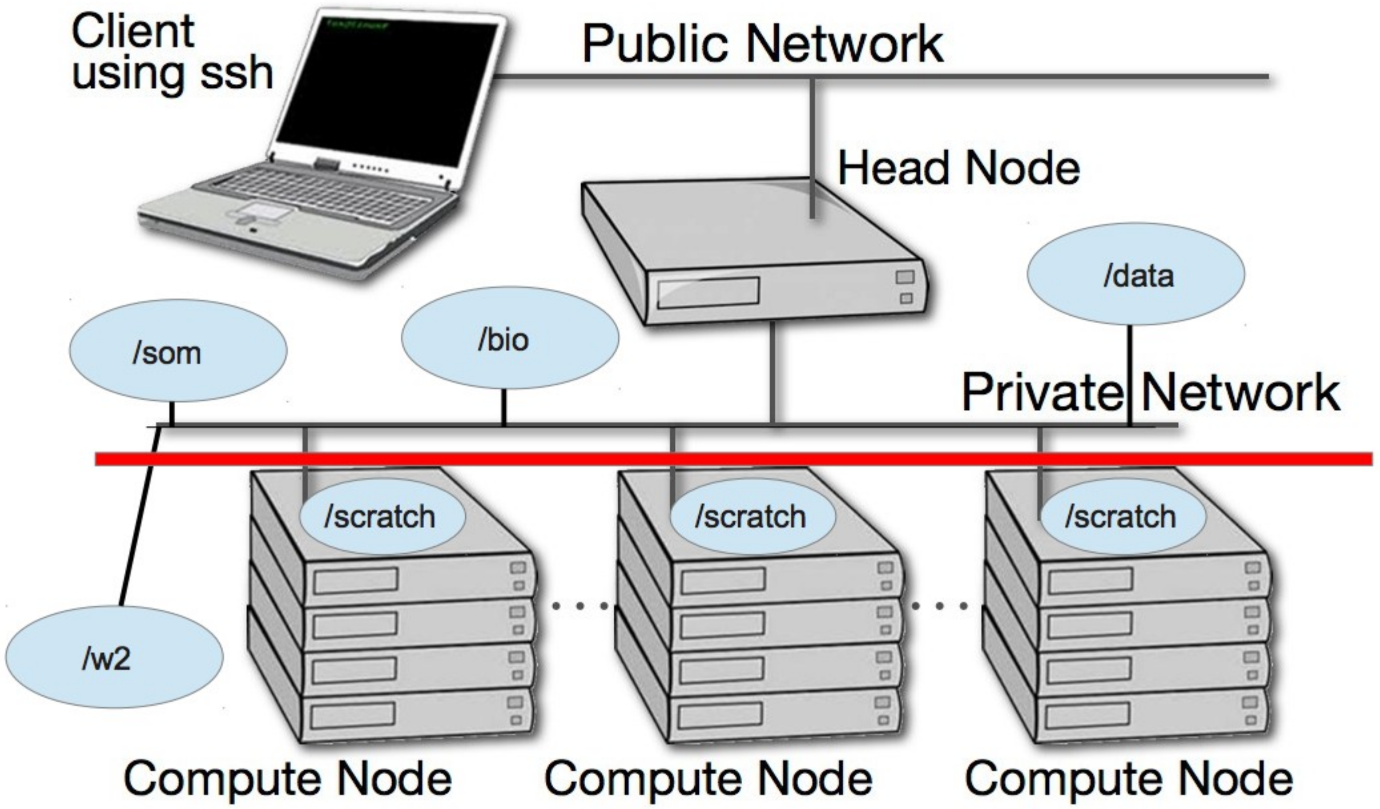
and…

Why do I need HPC?

What is a Cluster?

- bunch of big general purpose computers
- running the Linux Operating System
- linked by some form of networking
- have access to networked storage
- that can work in concert to address large problems
- by scheduling jobs very efficiently

Overview


Client
using ssh
Public Network
Head Node
/data
/som
/bio
Private Network
/scratch
/scratch
/scratch
/w2
Compute Node
Compute Node
Compute Node

HPC @ UCI in Detail

- ~5500 64b Cores – Mostly AMD, few Intel
- ~14TB aggregate RAM
- ~1PB of storage (1000x slower then RAM)
- Connected by 1Gb ethernet (100MB/s)
- Connected by QDR IB (800MB/s)
- Grid Engine Scheduler to handle Queues
- > 650 users, 100+ are online at anytime

What HPC is NOT

- NOT: your personal machine – shared resource
- NO DATA IS BACKED UP – WHAT SO EVER
- Well secured from mischief and disasters – not an invitation

DATA IS NOT BACKED UP

- NO DATA IS BACKED UP – WHAT SO EVER - Agitate to your PIs to get us more $ if you want this.
- Most data is stored on RAID6
- BUT! Any of that can disappear at any moment
- IF ITS VALUABLE, back it up elsewhere --- or the code that generated it.

Linux FileSystem Layout

/

bin	critical executables
boot	kernel image and init files
dev	device file
etc	config files
home	usually where your files live
lib	critical library files
lib32	32bit libs
lib64	64bit libs
lost+found	what it sounds like
media	where removable disks get mounted
mnt	where temporary other devices devices get mounted
opt	optional package installs
proc	process tracking dir, system config files
root	home for the root user
run	keeps track of running processes (locks, IDs)
sbin	system binaries
selinuxugh.	Secure linux config (usually empty on a usable system)
srv	service-specific files (some distros)
sys	system-specific files (some distros)
tmp	where anyone can write temporay files
usr	most of the system files live here
var	'varying' files for keeping track of various system processes.

HPC FileSystem Layout

Orange – Cluster Wide
Black – Node Specific

```
/
├── data/        NFS Mount
     |─apps      All Programs are installed here
     |─users     Users home directory                                           – 20GB LIMIT PER USER
├── w1/          Public NFS Server         – No Enforced Disk Limit             – 14TB Space
├── w2/          Public NFS Server         – No Enforced Disk Limit             – 40TB Space
├── bio/         Gluster Space for BIO group   ~400TB
├── som/         Gluster Space for SOM group ~160TB
├── cbcl/        Gluster Space for CBCL group
├── ffs/         Fraunhofer FileSystem – Experiential File System          ~170TB Space
├── scratch      Node specific temporary storage per job (faster then all above)     ~1TB – 14TB of Space
├── /tmp         Same as scratch
```

Disk Space / Quotes / Policies

- You can only have so much space
- 20GB for /data/ (home directory)
- 6months or older without use – please remove from cluster
- More for Condo owners or Groups who have bought extra disk space.
- Regardless, NO DATA IS BACKED UP

Data Sizes

- Your data will be BIG – “BigData”
- BigData is somewhat 'dangerous' due to its bigness.
- Think before you start. You can't predict everything, but you can predict a lot of things – more on this later

Example Data Sizes

- 1,000 b (KB) – an email
- 2MB – Size of a 3 ½ " floppy
- 250MB – Human Chr 1
- 1,000,000,000b (1GB) – 30X Story of Civilization
- 4GB – Size of DVD
- 1,000,000,000,000b (1TB) – 1/15th Lib of Congress (256 DVDs)
- 5 TB – primary data fr. Illumina HiSeq2K
- 1,000,000,000,000,000b (1PB) – 100X Lib of Congress (262,144 DVDs)

How to: Login with SSH

- SSH is an encrypted protocol so that info over the connection can't be deciphered by others.
- You MUST use SSH to connect to HPC – think command line
- Underlies 'scp' (secure copy), sftp
- Also 'sshfs' which allows you to attach your filesystem to HPC (or vice versa).

Command Line Cons

- The tyranny of the blank page
- No visual clues
- Type vs click
- Have to know what to type
- HOW DO YOU KNOW WHAT TO TYPE???

Command Line Pros

- It doesn't get much worse than this
- When you do learn it, you'll know it and it probably won't change for the rest of your life, unless they perfect mind control..
- It's a very efficient way of interacting with the computer (which is why it's survived for 50+yrs).
- You can use it to create simple, but very effective pipelines and workflows.

Keeping SSH Session Alive

- If you need to maintain a live connection for some reason, use 'byobu or screen'.
- It allows you to multiplex and maintain connections in a single terminal window.
- Somewhat unintuitive interface but very powerful.
- You know about cheatsheets (Google!!)

GUI with SSH and HPC

- Linux uses X11 for graphics
- X11 is very chatty, high bandwidth, sensitive to network hops/latency.
- If you need graphics programs on HPC, use x2go vs native X11.
- x2go is described in the Tutorial & HOWTO, also GOOGLE

How to: SSH & The Shell

- Once logged in to HPC via SSH you are now using the *Shell,* which is..
  - A program that intercepts and translates what you type, to tell the computer what to do.

- What you will be interacting with mostly.

- HPC shell is '***bash***', altho there are others.

Follow Along

- Take a few moments to login to cluster (Harry and Adam will help if needed)

- After logged in follow me on screen

- Ref:
- http://moo.nac.uci.edu/~hjm/biolinux/Linux_Tutorial_12.html

Know the shell, Embrace the Shell

- If you don't get along with the shell, life will be hard.
- Before you submit anything to the cluster via qsub, get it going in your login shell.
- You're welcome to start jobs in on the IO node, type: `qrsh`
- "*DO NOT, UPON THE PAIN OF DEATH, RUN JOBS ON THE LOGIN NODE*"

How to know if I am on Login Node?

- Look at your shell!
- [aebrenne@hpc ~]$ ← 'HPC' is the login node
- [aebrenne@compute-6-1 ~]$ ← On compute 6-1
- May also use the command `hostname`

Command Line Editing

- Since you'll be spending a lot of time fighting with the cmd line, make it easy on yourself.
- Learn cmd line editing to edit previous cmds
- Up/Down arrow keys scroll thru cmd history
- L/R arrow keys scroll by 1 char
- `^` means CONTROL Key
- `^` makes L/R arrow jump by a word (usually)
- Home, End, Insert, Delete keys work (except Macs lack 'Delete' keys (because … Steve Jobs)
- `^u` kills from cursor left; `^k` kills from cursor to right
- Tab for auto complete

STDIN, STDOUT, STDERR

- STD = Standard
- STDIN is usually the keyboard, but...
- STDOUT is usually the screen, but...
- STDERR is *also* usually the screen, but...
- All can be redirected all over the place
- to files, to pipes, combined, split (by 'tee'), etc
- More on this later.

File & Directories

- Files & folders much like on Mac & Win
- Except...
- Names are case-sensitive, 256 char long
- 'Folders' → 'Directories' , separated by '/'
- No spaces in names(*)
- . means 'in this dir’
- .. means parent dir
- ~ means 'home dir'
- A leading '/' means 'from the root dir'

Foreground & Background Jobs

- Foreground (fg) jobs are connected to the terminal. You kill a fg job with `^C`.
- Background (bg) jobs have been disconnected from the terminal.
- Send a job to the bg by appending `&`
- Recall a job to the fg with `fg`.
- Send a fg job to the bg with '^z' (suspend), then 'bg'.

Pipe |

- Works with STDIN/OUT/ERR to create 'pipelines'
- Very similar to plumbing; can add 'tee's to introduce splits
- STDOUT of one program goes to the STDIN of another command whose STDOUT goes to the STDIN of another program ad infinitum.
- Sooooo......

Pipe Example

```
w |cut -f1 -d ' ' | egrep -v "(^$|USER)" | sort | uniq -c | wc
```

w spits out who is on the system right now

cut -f1 -d ' ' chops out the 1st field (the user), based on the space token

egrep -v "(^$|USER)" filters out both blank lines and lines with 'USER'

sort sorts the usernames alphabetically

uniq -c counts the unique lines

wc -l word-counts that output.

Example: Now on HPC!

General Commands

- `cmd -h`
- `cmd -help`
- `man cmd`
- `info cmd` (but you hope not)
- And ….. Google...

Some Useful Commands

- ls [many options] = list fil<tab><tab>
- cd [up or down] = change directory
- find [from] -name [name] = find files
- locate [name] = where is this file?
- tree [options] = show the dir tree
- file [name(s)] = what is this?
- du = disk usage
- df = disk free
- less [names] = view files
- cols [file] = view file in columns

Creative / Destructive Commands

- mkdir [name] – make a dir
- rmdir [name] – remove a dir
- mv [from] [to] = move or rename
- cp [from] [to] = copy file(s)
- rm [file] = delete file(s)
- wget [URL] = get a file from the Internet
- curl -O [URL] = ditto, but on steroids

More Useful Commands

- mc = Midnight Commander
- [ah]top = top CPU-using processes
- time [command] = how long does it take?
- [aef]grep [regex] [files] = find regex in files
- cat [files] = print the files to STDOUT
- head/tail [files] = dump the top / bottom of files

Regular Expressions

- Among the most powerful concepts in pattern matching
- Simple in concept, NASTY in implementation
- Among the ugliest / most confusing things to learn well
- But pretty easy to learn the simple parts.
- But you NEED to learn it – it's central to computers and especially biology

Regexes

- Simplest form is called globbing (a*)
- Mix it up (a*.txt)
- A bit more (a*th.txt)
- Can be MUCH more complex:
- [aeiou] = any of 'aeiou'
- F{3,5} = 3-5 'F's
- H+ = 1 or more 'H's
- . = any character
- Also classes of characters (#s, alphabetic, words)

Archiving / Compression

- tar = std archive format for Linux
- zip = common archive format, from Windows
- gzip/unzip = common compressed format
- bzip2/bunzip2 = another compressed format
- pigz = parallel gzip (for large files)
- pbzip – parallel bzip2 (ditto)

Editing Files: simple & complex

- Text-based:
  - nano, joe, vi/vim, emacs

- GUI-based:
  - gedit, nedit, kate, jedit, emacs

(choose one and learn it, *well*)

Customize Your Environment

- Change your prompt to something useful to you (and to us):

- PS1="\n\t \u@\h:\w\n\! \$ "
- Set aliases (`alias nu=”ls -lt |head -22”`)
- Set Environment Variables (`export EDITOR=vim`)

- All done via `.bash_profile` & `.bashrc` files in your *home directory*

Move Data to / from HPC

- Covered in detail in HPC USER HOWTO, which references: goo.gl/XKFEp
- scp, bbcp, netcat/tar on Mac, Linux.
- WinSCP, Filezilla, CyberDuck,FDT on Win
- Everyone should know how to use rsync. Not the easiest to learn, but very powerful.
- rsync GUIs for Linux, Windows, MacOSX

Using Software on HPC

- 3 main sets of programs
  - Your personal set (later)
  - The default system utilities (already covered)
    - cut, egrep, ls, mv, cp, rm, cd, etc…
  - The module system programs

How to Find Software

- locate <partial search term>
- apropos <search term>
- na<tab><tab> → name
- yum search <search term>   # CentOS
- module avail (will dump all modules)
- Google
- Ask us.

The Module System

- The modules system is primarily how you will use software on HPC
- `module avail` shows all installed software
- `module load R/3.01` loads program R version 3.01 – DOES NOT START THE PROGRAM
- `module unload` unloads the specified program
- `module purge` removes all loaded programs

The Scheduler (GE)

- Just another program that juggles requests for resources
- Make sure a program is working on a small set of test data.
- Need a short bash script (aka ***qsub script***) to tell the GE what your program needs to run.
- Can improve the performance of your program in a variety of ways (staging data, running in parallel, using array jobs, etc)

The Scheduler: QSUB vs QRSH

- *qrsh* will log you into an Interactive Node (IO Node).
- IO Nodes are useful for GUI programs (X11 & x2go needed) or testing / running SMALL and FAST data sets

- qsub script is just a series of bash commands that sets up your resource requirements, PATHs, executes your jobs, and does the post-processing. – NO USER INVOLVEMENTENT during the process

GE Useful Commands

- qstat - Queue Status
- queue / q – What queues you have access to
- qdel – Delete/Stop your job
- qhost – Show all nodes and their status

- Use `man  cmd` to find out more information on above
- http://hpc.oit.uci.edu/PHPQstat

- Ref:
- http://hpc.oit.uci.edu/running-jobs
- http://hpc.oit.uci.edu/PHPQstat/

Sample QSUB Script

- Visit:
- <http://hpc.oit.uci.edu/guides/qsub-biolinux.html>

- Ref:
- <http://goo.gl/hrcXBg>

GE – Request Node Resources

- Use Case: You know your program requires at least
  - 24GB Memory
  - 16 CPU Cores
- You need to tell the scheduler
- `#$ -pe openmp 16`
- `#$ -l mem_free=24G`
- This does not make your program run faster or use all cores – you simply reserve this amount

GE – Queues

- As you noticed, the scheduler uses queues to handle your job.
- Some queues have higher priority then others.
- Type `queue` or `q` to see what you have access to
- `#$ -q som, free*`

GE – Free and All Queue

- The free* queue allows anyone to use CPU cycles when they are not in use on any queue/node cluster wide
  - When owners want to use their nodes, free* jobs are suspended

- Similar to the free* queue, the all queue is group specific: abio, asom, etc.

Big Data

- Volume
  - Scary sizes, and getting bigger
- Velocity
  - Special approaches to speed analysis
- Variety
  - Domain-specific standards (HDF5, bam/sam, FITS), but often aggregations of unstructured data
- BigData Hints for Newbies
  - http://moo.nac.uci.edu/~hjm/biolinux/BigData4Newbies.html

Big Data – How Big is Big?

# Bytes	Byte name / Abbriev'n	Approximation
1/8	bit (b)	0 or 1: the smallest amount of information.
1	Byte (B)	8 bits, the smallest chunk normally represented in a programming language.
2^{10}	1,024 B (1 KB)	a short email is a few KBs
2^{20}	1,048,576 B (1 MB)	a PhD Thesis ; Human Chr 1 is ~250 MB
2^{30}	1,073,741,824 B (1 GB)	the Human Genome is 3,095,693,981 B (optimized, ~780 Mb @ 2b/base) ; a BluRay DVD holds 25GB per layer (most movie BluRays are dual-layer = 50GB); a Genomic bam file is ~150GB
2^{32}	4,294,967,296 (4GB)	**fuzzy border between SmallData (32b) and BigData (64b)**
2^{40}	1,099,511,627,776 B (1 TB)	1/10th Library of Congress (LoC); the primary data fr. an Illumina HiSeq2K is ~5 TB
2^{50}	1,125,899,906,842,624 B (1 PB)	100X LoC; ~HPC's aggregate storage; ~100 PB is the yearly storage requirements of YouTube.
2^{60}	1,152,921,504,606,846,976 B (1 EB)	the est. capacity of the NSA's data facility is ~12 EB

Integer Byte Sizes

word size	#bits	range of variable
byte or char	8	256
int	16	65,536
long int	32	4,294,967,296
long long int	64	1.84467440737e+19

Inodes and ZOT Files

- Inodes contain the metadata for files and dirs
- Inodes are pointers to the data
- Regardless of size, a file needs at least one inode to locate it.
- A file of 1 byte takes up the same minimum inode count as a file of 1TB
- DO NOT USE ZOTFILES!! – Zillions of Tiny Files

Streaming Reads & Writes

- Demo Time!
  - Pack of Cards

Pointless Data Replication

- How informative is this?
- How informative is this?
- How informative is this?
- How informative is this?
- How informative is this?
- vs
- How informative is this? [5x]
- __LINE__

Editing Big Data

- Use format-specific utilities to view such files and hash values to check if they're identical to what they should be.

- Try not to be the member of your class who tried to open a 200GB compressed data file with nano/vim/joe/emacs, etc.

[De]Compression

- If your applications can deal with compressed data, KEEP IT COMPRESSED.
- If they can't, try to use pipes (|) to decompress in memory and feed the decompressed stream to the app.
- Use native utilities to examine the compressed data (zcat/unzip/gunzip, grep, archivemount, Vitables, ncview, etc.

Move BigData

- Don't.
- Otherwise, plan where your data will live for the life of the analysis, have it land there, and don't move it across filesystems.
- Don't DUPLICATE DUPLICATE DUPLICATE BigData
- See: http://moo.nac.uci.edu/~hjm/HOWTO_move_data.html
  - rsync for modified data
  - bbcp for new transfers of large single files, regardless of network
  - tar/netcat for deep/large dir structures over LANs
  - tar/gzip/bbcp to copy deep/large dir structures over WANs

Checksums

- They work. Choose one and use it.
- md5sum / jacksum
- Use MANIFEST files & copy them along with the data files.
- See Checksum example
    - http://moo.nac.uci.edu/~hjm/biolinux/BigData4Newbies.html#checksums

Processing BigData

- Files (HDF5, bam/sam) and specialized utilities (nco/ncview, [Py/Vi]tables, R, Matlab)
- Relational Dbs (SQLite, Postgres, MySQL)
- NoSQLs (MongoDB, CouchDB)
- Binary Dumps (Perl's Data::Dumper, Python's pickle)
- Non-Storage (pipes, named pipes/FIFOs, sockets)
- Keep it RAM-resident.

Big, but not forever

- HPC is not backed-up.
- Cannot tolerate old, unused BigData.
- RobinHood is looking for your old BigData.
- Please help us by doing your own data triage.
- Ask your PIs to bug our boss to provide more resources so we can provide more resources.