Research Computing at UC

Introduction to UC's Advanced Research Computing (ARC) Resources: HPC Computing/Analysis Tools For Research And Education

George Turner, Chief Systems Architect
turnerg@iu.edu
Pervasive Technologies Institute, UITS/RT, Indiana University

Contact information concerning any issue related to today's topic: **arc_info@uc.edu**

First things first

- This is an introductory tutorial; if you are interested in something more advanced please start the conversation by sending an email to **arc_info@uc.edu**
- To login to the ARCC2 you must be on the UC campus or using the UC VPN
- People are standing by to help
- While you are logging in, we will cover basic terms, history, etc.

Logging into the ARCC2

- ssh *USERNAME*@arcc2.uc.edu
  - Your USERNAME will be your UC 6+2 for UC individuals or as identified in an email you received yesterday.
  - If you have problems connecting, instead of arcc2.uc.edu try the IP address 10.127.5.249

Terms

- **ARC** : Advanced Research Computing group. The team working to establish a centralized research computing cyberinfrastructure at UC
- **ARCC** : the Advanced Research Computing Clusters
  - More on this to follow
- **Storage**
  - Home directories
  - Workbench/scratch
  - Project space (future acquisition)
  - Archive (future acquisition)
- **Networks**
  - Switch fabrics; OmniPath, Infiniband (100 gb/s)
  - External connectivity (10 gb/s)

Some Introductory Tutorials

- Basic Linux introduction
  - https://training.linuxfoundation.org/training/introduction-to-linux
- Beginner's Guide to the Bash Terminal
  - https://www.youtube.com/watch?v=oxuRxtrO2Ag
- Introduction to High Performance Computing Systems, ADACS
  - https://www.youtube.com/watch?v=7zJUceJiYxQ
- Slurm
  - Introduction to Slurm, ADACS
    - https://www.youtube.com/watch?v=K_JIPrcPHCg
    - There are multiple YouTube videos in this series
  - Slurm Basics, Center for High Performance Computing, University of Utah
    - https://www.youtube.com/watch?v=49DzPT9HFJM
    - There are multipe YouTub videos in this series
  - Introduction to Slurm Tools, BYU Supercomputing
    - https://www.youtube.com/watch?v=U42qlYkzP9k

Terms (continued)

- **SLURM** – cluster resource manager and scheduler
- **ssh** – a command line interface used to connect between servers
  - Native to Linux and Macs
  - putty for Windows (https://www.putty.org)
- **#** - comment in Linux; i.e. whatever follows the hashtag will not be executed
- **stdin, stdout, stderr** : input and out put IO streams; e.g. keyboard for input, terminal for output and error messages
- **PATH** : a list of directories where applications are; to see the current path do `echo $PATH`

ARC --The project

- https://www.research.uc.edu/arc
- Five+ years in the making
- Jane Combs and Amy Latessa driving force
- Larry Schartman, Kurt Roberts, Adam Steele, James Dusing, Himakar Ganti, Aditya Kavalur, Rob Ogden, and others!!!!
- Faculty Advisory Committee – Dr Prashant Kare, Chair
- UC Office of Research, VPR Limbach, Michael Dunaway, Philip Taylor
- XSEDE Capabilities and Resource Integration (XCRI) Steve Bird, Eric Coulter
- IU Pervasive Technologies Institute, Research Engagement, George Turner
- Engagement and Performance Operations Center (EPOC) report
  - Jennifer Schopf (GlobalNOC), Jason Zurawski (ESnet), Has Addleman (GlobalNOC), Doug Southworth (GlobalNOC)
  - University of Cincinnati Campus-Wide Deep Dive https://escholarship.org/uc/item/6t58p052
  - Watershed moment; well worth a read; others should contribute case studies!!!

ARC Clusters (ARCC)

- **ARCC-1** - a.k.a. the *Pilot* cluster
  - Explore what is involved in establishing research cyberinfrastructure
  - Acquired in three phases
    - Started with Dr Prashant Khare's startup cluster
    - Addition funded by the Office of Research
    - Expansion of Dr. Khare's original cluster
  - Dell PowerEdge 36 C6420, dual Intel Gold 6148 2.4 GHz, 20 core CPU (40 total cores), 192 GB RAM
  - Dell PowerEdge R740xd Storage node, 96 TB raw
  - Dell PowerEdge R740 GPU server dual Intel Gold 6148 2.4 GHz, 20 core CPU (40 total cores), 192 GB RAM, Nvidia V100-32G GPU
  - 100 Gb/s OmniPath switch fabric
  - Initial integration Dec-2018 and moved to the UC Data Center Jan-2019

ARC Clusters (continued)

- **ARCC-2** will be UC's first production cluster
  - Funded in part by an NSF MRI grant with Dr Prashant Khare as the project PI
  - Additional resources provided by UC Office of Research
  - Architected for longevity and expandability
  - Integrating compute, networking, storage into cohesive environment
  - The cluster we will be using today

ARC Clusters (continued)

- **ARCC-2** (continued)
- *Excuse our Dust : Under construction*
  - Currently 19 compute nodes and one Nvidia V100 GPU node
  - HPE Apollo 2000, dual AMD EPYC 7452, 32 cores (64 total) 2.3GHz, 256 GB RAM, Gen4 PCIe bus
  - HPE Proliant DL 385 Gen10+ GPU nodes, dual AMD EPYC 7452, 32 cores (64 total), 1024 GB RAM, dual Nvidia A100-40 GPUs , Gen4 PCIe bus
  - One login node, one management node
  - Storage purchase is in the works
  - More compute and GPU nodes after data center upgrades

Some basic Linux commands

Note: Faculty Development Workshop: Linux 101, next Friday, 22-Jan-2021

- **man** : the “help” for Linux; e.g. man bash, man ls, man salloc, etc.
- **pwd** : present working directory; what directory am I sitting in
- **ls** : show me the files in this directory
- **cd** : change Directory
  - cd dd2020-hpc - move down into the dd20202-hpc directory
  - cd .. - move up one directory
- **cp** : copy a file
  - cp –rp /opt/ohpc/dd2020-hpc .
    - Recursively copy the directory at /opt/ohpc/dd2020-hpc to my current directory

Some basic Linux commands (continued)

- **cat** : stream a whole file to stdout
- **more** : stream a file one page at a time to stdout
- **less** : like more but you can move up and down in the file using the arrow keys; exit with q
- **module** : change your Linux environment; e.g. add applications
- **echo** : repeat the argument to stdout (the terminal)

Editors available on Linux

- **nano**
  - Easist to use but not always available
  - Menu at the bottom of the screen
  - Use control-x (^x) to exit
- **vi**
  - always available om a Linux system
  - enter insert mode with `i`
  - Exit insert mode with Esc key
  - Exit editor with Esc-wq to save work
  - Exit editor with Esc-q! to not save work

Editors available on Linux (continued)

- **emacs**
  - very complicated
  - very powerful
  - Written in lisp; very extensible
- GUI editors
  - Works great if you have X11 working via the ssh connection
    - To see if X11 is working, do **echo $DISPLAY**
  - **gedit**
  - **nedit**

SLURM – Simple Linux Utility for Resource Management

- Development started 2002
- Prior resource managers PBS (Portable Batch System), TORQUE, LSF, Sun Grid Engine (SGE)
- Open Source under the GPL v2
- Commercial support
- Plugins for Auth, MPI support, Checkpoint/restart, network and memory topologies, & many more
- Combined resource manager and scheduler

SLURM – Simple Linux Utility for Resource Management

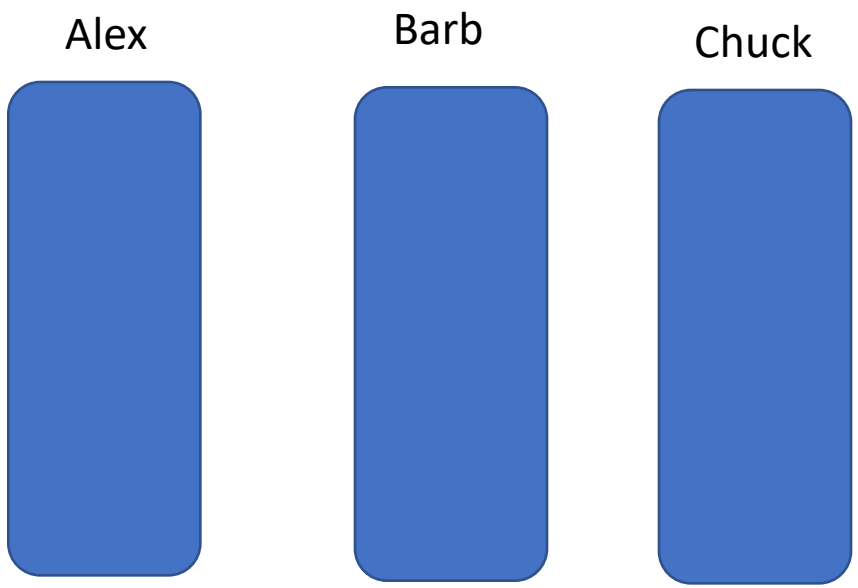
- Combined resource manager and scheduler
- Resource manager
  - Inventory resources (nodes, sockets, cores, memory, licenses, GPUs, etc.)
- Scheduler
  - Map job requirement onto available resources
  - Predict when resources will become available
  - complex scheduling algorithms e.g. optimize for memory or network topologies, fairshare scheduling, advanced reservations, preemption resource limits; e.g. access restrictions, user & group quotas, etc

SLURM – Scheduling – Calculating a job's dispatch priority

- **FIFO** – First In, First Out; no longer used
- **Fairshare** : the more you use the system, the lower your fairshare drops; it rebuilds over time. Documented at https://slurm.schedmd.com/fair_tree.html
- **Backfill** : small jobs can run sooner while resources are being gathered for larger jobs
- **Expansion** : allows small jobs to gain priority so that they can get in and run. E.g. you don't want to wait days to run a ten minute job

SLURM – Scheduling (continued)

- Fairshare example

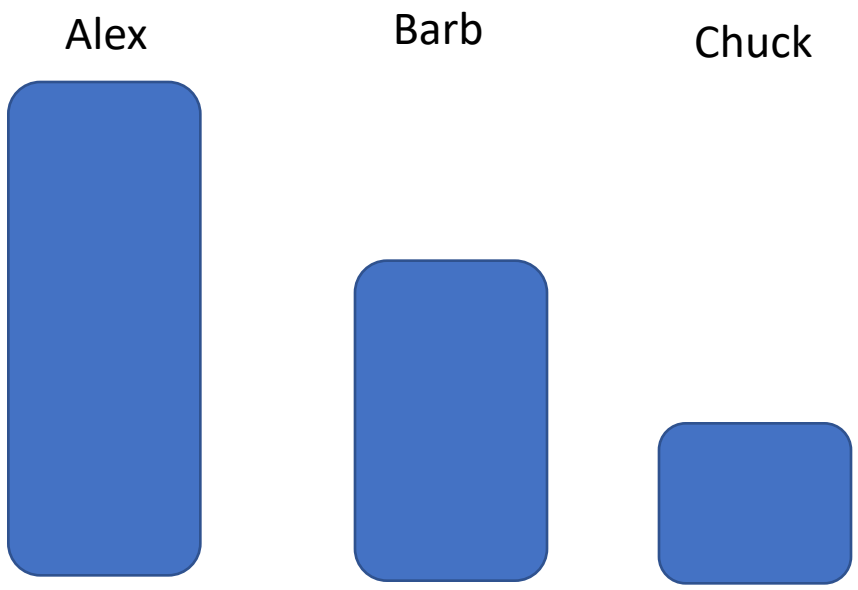


Alex uses no resources
Barb uses #cores for Xtime
Chuck uses twice the resources; it could be twice the number of cores or ran for twice as long

Starting failshare

SLURM – Scheduling (continued)

- Fairshare example

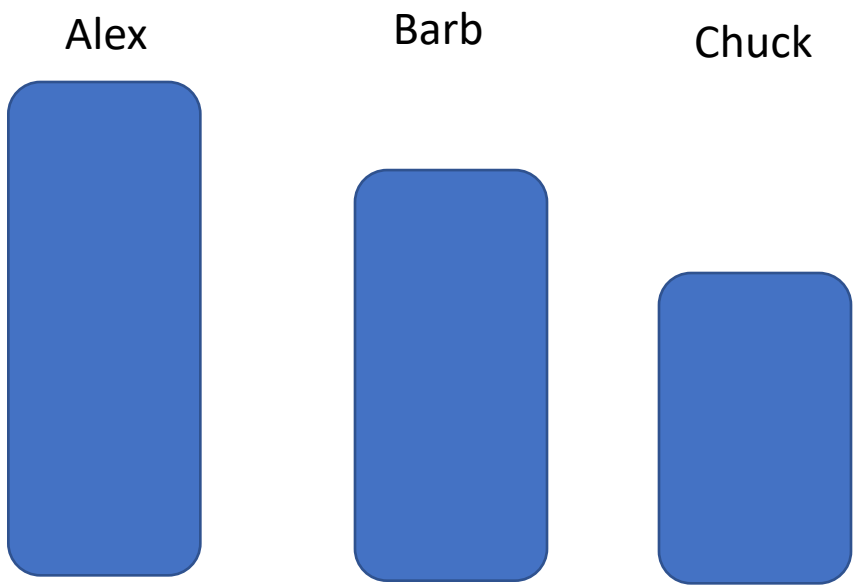


- Alex uses no resources
- Barb uses some #cores for some length of time
- Chuck uses twice the resources; it could be twice the number of cores or ran for twice as long or some combination therein
- If Alex, Barb, and Chuck were to submit new jobs, all things being equal, Alex would run first, Barb second, and Chuck last.

After Barb and Chuck utilize resources

SLURM – Scheduling (continued)

- Fairshare example

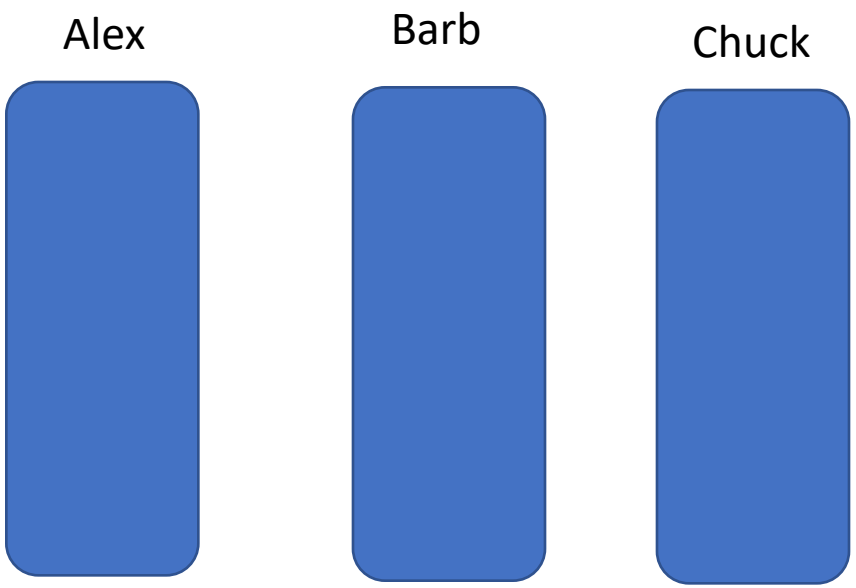


Over time, if no resources are used, they replenish and all three users would have an equal dispatch priority

After Barb and Chuck utilize resources

SLURM – Scheduling (continued)

- Fairshare example



Over time, if no resources are used, they replenish and all three users would have an equal dispatch priority

After a period of time using no resources

Some basic Slurm commands

Slurm Cheat sheet available at https://slurm.schedmd.com/pdfs/summary.pdf

- **salloc** : allocate resources for an interactive job
- **sbatch** : queue a batch script and allocate resources for the job when they become available
- **srun** : utilize resources allocated for a job (or job step) via the salloc or sbatch commands
- **squeue** : check the status of queued jobs
- **sprio** : view factors comprising a jobs priority
- **sinfo** : view the state of the system
- **scancel** : cancel queued or running jobs
- **sattach** : attach stdin/out/err to a running job
- **sbcast** : copy a file to local storage (e.g. /tmp) on all the nodes allocated to a job

Once logged in, prep our environment…

```
# set up out environment
module avail
module spider
module load gnu9 openmpi4
module list

# copy over the files needed for the tutorial
cp –rp /opt/ohpc/training/fdw-hpc-210114 .
cd fdw-hpc-210114
ls -al
```

Compile and run our hello world example

```
# start with the worlds simplest C program
less simple.c
gcc -o simple simple.c
./simple
# doesn’t look like much; but, it does everything that a basic program does
```

Compile and run our hello world example

```
# protocol dictates that we start with a hello world example program
less hello.c
mpicc -o hello hello.c
salloc --tasks=2
mpirun hello
exit
```

That was simple, now let’s do something useful

```
# now try a slightly more sophisticated  example
mpicc -o connectivity connectivity.c

salloc --tasks=2
mpirun connectivity
exit
```

Exploring --nodes and --ntasks

```
salloc --nodes=2
mpirun connectivity
exit

salloc --nodes=2 --tasks=2
mpirun connectivity
exit

salloc --nodes=2 --tasks=4
mpirun connectivity
exit

salloc --nodes=2 --tasks-per-node=2
mpirun connectivity
exit
```

Interactive verses batch

```
# using sbatch to queue jobs for running in the background
# look for the output in files named slurm-NN.out

less hello.slurm
sbatch --nodes=2 --tasks-per-node=2 hello.slurm
less slurm-XX.out            # where XX is the job ID number

less connectivity.slurm
sbatch --nodes=2 --tasks-per-node=2 connectivity.slurm
less slurm-XX.out            # where XX is the job ID number
```

References & Other Tutorials

XSEDE/Container_Tutorial
https://github.com/XSEDE/Container_Tutorial/tree/master/Gateways2020

Sylabs (developers of singularity) tutorial videos
https://sylabs.io/videos

Questions, Comments, Suggestions?

Project website: https://research.uc.edu/arc/
Project email: arc_info@uc.edu
Direct email: turnerg@iu.edu

License Terms