Dell C8220 cluster, ~200 nodes, 2x since last year
  - 50x GPU nodes, each with dual NVIDIA K20 GPUs
  - 150x CPU nodes, each with dual 2.6GHz 8-core Intel Xeon CPUs, and 64/128/256GB of RAM
  - 2TB, 48-core ‘hugemem’ node
- FDR Infiniband, 2-to-1 oversubscription
- Dell NSS NFS server, ~120TB
- Dell / Terascale HSS Lustre storage system, ~250TB
Priority modeling

- Complicated priority model due to funding model
- Using priority/multifactor plugin
- Scheduler will try to balance groups runtime over a longer period, so they match their priority targets
- Top-level: contributing schools, colleges, research centers
  - Second-level: research groups, departments
    - Third: separate research groups in department
      - Fourth: individual users in group
## Fairshare model - top tier

```bash
# sacctmgr show assoc tree format="account,fairshare" | editing
```

<table>
<thead>
<tr>
<th>Account</th>
<th>Fairshare</th>
</tr>
</thead>
<tbody>
<tr>
<td>root</td>
<td>1</td>
</tr>
<tr>
<td>c1</td>
<td>parent</td>
</tr>
<tr>
<td>cbi</td>
<td>1880</td>
</tr>
<tr>
<td>ccas</td>
<td>10080</td>
</tr>
<tr>
<td>other</td>
<td>1960</td>
</tr>
<tr>
<td>seas</td>
<td>4320</td>
</tr>
<tr>
<td>sphhs</td>
<td>720</td>
</tr>
</tbody>
</table>

- **c1** account - support folks, give them top ranking
- “**other**” - 10% open to anyone (10pts per node)
- **Everyone else** - 90pts per node
Fairshare model - secondary tier

# sacctmgr show assoc tree format="account,fairshare" | editing

---------------

<table>
<thead>
<tr>
<th>account</th>
<th>fairshare</th>
</tr>
</thead>
<tbody>
<tr>
<td>root</td>
<td>1</td>
</tr>
<tr>
<td>ccas</td>
<td>10080</td>
</tr>
<tr>
<td>astrophysics</td>
<td>17</td>
</tr>
<tr>
<td>ccas-other</td>
<td>31</td>
</tr>
<tr>
<td>economics</td>
<td>1</td>
</tr>
<tr>
<td>qcd</td>
<td>16</td>
</tr>
</tbody>
</table>

- at the second tier, set 1 point per node
- fairshare is calculated separately at each level, ratio between accounts at each level is all that matters
Useful commands:

- sshare
- sprio
- sreport

Force users to specify timelimit:

JobSubmitPlugins="job_submit/require_timelimit"
Level 3 support contract with SchedMD
Filed one bug in the past year - #532
  … I was doing things out of order. My fault.
Patch to prevent this committed within a week
Feature requests

- Independent fairshare hierarchies for specific partitions
  - Our current model implies GPU and CPU nodes cycles are interchangeable
    - No one has complained about this… yet
- sreport support for partitions
  - same rationale as above
  - using ugly/slow bash scripts to aggregate statistics
Novel (ab)uses of Slurm

- Two non-traditional uses of the Slurm scheduler
  - Fastscratch - dynamic SSD scratch space allocation
  - Backups

- “If all you have is a hammer, everything looks like a nail.”
  - Slurm is a pretty good good hammer...
Motivation
- Genomic sequencing apps
- "Big data"
- Random I/O with mixed read+writes
  - Our NFS and Lustre fileservers hate this
  - SSDs handle this much better than disks, but...
    - We can’t afford to install large SSDs into all nodes
    - And don’t want a "special" set of nodes just for them
    - And this wouldn’t deal with shared access across nodes

Possible solution - build fileserver with ~3 TB of SSDs
  - And allocate space to jobs on demand
Fastscratch

- Need to manage space and assign to active jobs only
- First approach… some way to use GRES?
  - GRES works with devices in nodes, this is a separate system
  - Jobs shouldn’t have access to the fileserver
    - Let alone launch jobs on it

- There’s a second mechanism in Slurm that tracks resources: Licenses
Fastscratch

- `slurm.conf`: "Licenses=fastscratch:3000"
- One license == 1 GB of space

- Want 400GB? Add this to your job scripts
  
  ```bash
  #SBATCH --licenses=fastscratch:400
  ```
Use a new PrologSlurmcltd script to
- SSH into fileserver to allocate space
  - XFS project quotas currently
  - /fastscratch/$jobid
- Set permissions, adjust exports
- Adjust Prolog scripts
  - Mount /fastscratch/$jobid on assigned nodes
Fastscratch - Implementation

- Epilog
  - Unmount /fastscratch
- EpilogSlurmctld
  - Remove directory and exports

- Jobs are responsible for moving data in/out of assigned /fastscratch space
  - Same as if they were using local /scratch space, although this is available across multiple nodes
Backups

- Disk backup servers
  - Located in a separate datacenter
  - ~100TB usable in 4RU.
- ZFS on FreeBSD, uses zfs snapshots
- Dedicated transfer node located within cluster - “syncbox”
- Goal: run rsync on separate user directories in parallel.
- Bottlenecks are
  - (1) SSH encryption speed (limited by core speed),
  - (2) TCP throughput between locations, then
  - (3) disk I/O.
- Running in parallel lets us get past (1) and (2).
Backups

- Create special partition:
  ```
  PartitionName=syncbox Nodes=syncbox MaxTime=7-0
  RootOnly=YES Hidden=YES Shared=YES:8
  ```

- Run each rsync as separate job:
  ```
  #!/bin/bash
  #SBATCH -p syncbox --share -t 1-0

  # sync
  rsync -avd /home/$1 backup1:/storage/home

  # snapshot
  ssh backup1 zfs snapshot storage/home/$i@$(date +%Y%m%d)
  ```
Thank You

Documentation: http://colonialone.gwu.edu
Twitter: @GWColonialOne
Support Email: hpchelp@gwu.edu

Tim Wickberg
wickberg@gwu.edu
Funicolare Monte Brè

Half-hour ride each way, we’re trying to take the 5:45pm ride up, 7pm ride down (or may walk down to a bus route instead)

Meet in front of the convention center by 5:10pm

15 minute walk to the station from here

CH 25 for round trip ticket