Habanero - Technical Information

- **Hardware**
  - Execute Nodes
  - Storage
  - Network
  - Management Nodes
  - Visualization Server
- **Scheduler**
  - Fair Share
  - Partitions

## Hardware

### Execute Nodes

Habanero has 302 execute nodes.

#### Standard Nodes

<table>
<thead>
<tr>
<th>Model</th>
<th>HP Enterprise XL170r</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>E5-2650v4</td>
</tr>
<tr>
<td>Number of CPUs</td>
<td>2</td>
</tr>
<tr>
<td>Cores per CPU</td>
<td>12</td>
</tr>
<tr>
<td>Total Cores</td>
<td>24</td>
</tr>
<tr>
<td>Memory</td>
<td>128 GB</td>
</tr>
<tr>
<td>Network</td>
<td>EDR Infiniband</td>
</tr>
</tbody>
</table>

#### High Memory Nodes

Habanero's 41 high memory nodes have 512 GB of memory. They are otherwise identical to the standard nodes.

#### GPU Nodes

Habanero has types of GPU nodes, both of which have the same CPUs and system memory (128 GB) as standard nodes.

- 14 GPU nodes each have two Nvidia K80 GPUs.
- 13 GPU nodes each have two Nvidia P100 GPUs.

## Storage

640 TB of GPFS parallel file system storage is used for scratch space and home directories.
Network

Management Nodes

Visualization Server

Habanero has a visualization server which allows web browser GUI access to Habanero storage and uses GPU acceleration to allow remote visualization of data.

Scheduler

Habanero uses the Slurm scheduler to manage jobs.

Fair Share

Resource allocation on our cluster is based on each group's contribution to computing cores. The Slurm scheduler uses fair share targets and historical resource utilization to determine when jobs are scheduled to run. Also, within-group priority is based on historical usage such that heavier users will have a lower priority than light users. Slurm uses all of a job's attributes - such as wall time, resource constraints, and group membership - to determine the order in which jobs are run.

Using job data such as walltime and resources requested, the scheduler can start other, lower-priority jobs so long as they do not delay the highest priority jobs. Because it works by essentially filling in holes in node space, backfill tends to favor smaller and shorter running jobs more than larger and longer running ones.

There is no preemption in the current system; a job in the queue will never interrupt or stop a job in run state.

Partitions