Transition to Summit

Peter Ruprecht
peter.ruprecht@colorado.edu

www.rc.colorado.edu

Outline

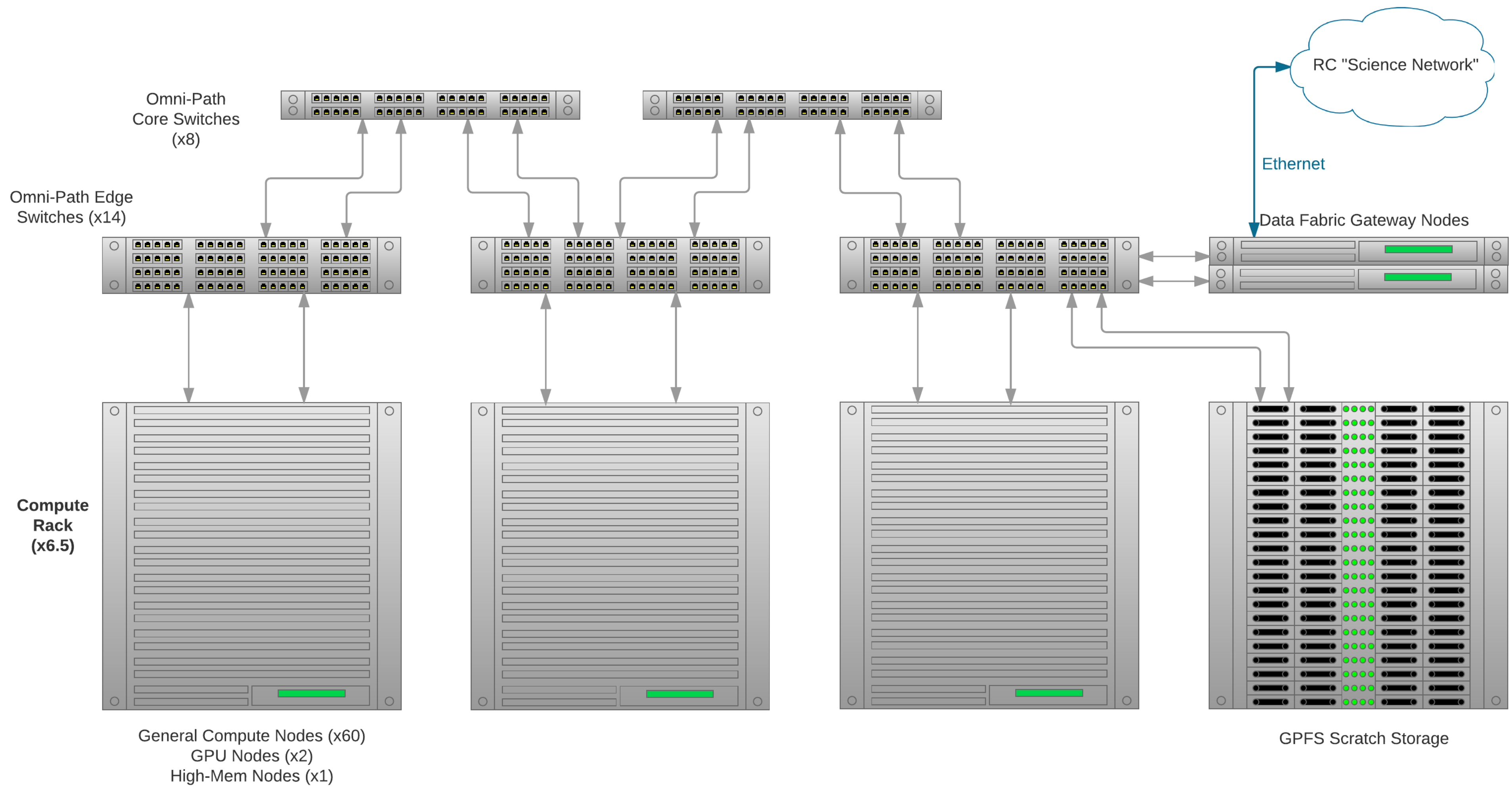
- Refresher on Summit's general architecture
- Storage
- Getting the most out of Summit
- Moving your application and workflows to Summit
- Scheduling and allocations

- Questions and answers

- One-on-one consulting

SUMMIT SCHEMATIC

Peter Ruprecht | July 15, 2016



Summit: Node Types

- 380 general compute nodes
  - 24 real cores (Intel Haswell)
  - 128 GB RAM => 5 GB/core
  - local SSD

- Measured performance (High-Perf LINPACK):
  - .76 TFLOPS/node (average)
  - Compare to about .15 TFLOPS/node on Janus

Summit: Node Types

- 10 GPGPU/visualization nodes
  - Same CPU and RAM as general
  - 2x NVIDIA K80 card (effectively 4 GPUs)
- Measured performance (hybrid HPL)
  - 3.87 TFLOPS/node
  - 38.7 TFLOPS aggregate
  - 36.8 TFLOPS 10-node HPL run (Rmax)

Summit: Node Types

- 5 high-memory nodes
  - 48 real cores (Intel Haswell)
  - 2048 GB RAM => 42 GB/core
  - 12-drive local RAID
- Measured performance (High-Perf LINPACK)
  - 1.46 TFLOPS/node
  - 7.3 TFLOPS aggregate

Omni-Path (OPA) Interconnect

- Cutting-edge network product from Intel
- Same role as the InfiniBand fabric on Janus
  - Also carries NFS traffic from /home, /projects, /work ...
- 100 Gb/s bandwidth
- Extremely low latency for MPI performance (1.5us)
- "Islands" of 28 - 32 nodes fully non-blocking
- 2:1 blocking factor between islands

GPFS Scratch Storage

- 1.2 PB of high-performance scratch storage
- GPFS for parallel access and improved small-file performance
- Measured >21 GB/s parallel throughput and >18K file creations or deletions per second

- ***Files created more than 90 days in the past are automatically purged!!***

GPFS Scratch Storage

- Available on login nodes and data-transfer nodes, but not on non-Summit compute nodes
- Mounted as /scratch/summit
- To copy data over from /lustre/janus_scratch, you can currently use cp or rsync on a login node.
- Globus transfers available “soon”

- **10 TB quota per user.** Request an increase by emailing rc-help@colorado.edu.

Why Optimize for Summit?

- Summit has fewer nodes than Janus, thus we can allocate fewer core-hours (SU)
- Summit is shared between CU, CSU, and RMACC, thus we can only allocate a fraction of core-hours to each

- If you take advantage of Summit's performance features, your allocation will go much farther, so
  - You will get more results
  - You will publish more papers
  - You will take fewer years to graduate
  - You will get { postdoc | awesome faculty position | tenure | Nobel prize }

Getting the most out of Summit

- Any application that you have built for Janus will need to be recompiled
- Applications ideally should take advantage of multi-core / many-core architectures (ie, parallelize)
- Performance improvements in latest processors are mainly through more cores and SIMD* rather than faster clock speed
- Applications should be parallelized and “vectorized” … otherwise Summit may *seem* slower than older systems

*single instruction on multiple data

Other New Features w/Summit

- OS is RedHat Enterprise Linux 7 (vs RHEL6 on Janus)
- RC login nodes will also be updated to RHEL7
- Slurm job scheduler will have some new features
  - Different QoS and partition names
  - Job requests must provide more detail
  - Allowing shared nodes, for single-core jobs
- Allocations will be provided as shares rather than “credit limits”; you’ll get higher queue priority if you have a larger share
- Running on different node types will “cost” different amounts

Building your software

- *Applications that were built on Janus need to be recompiled for Summit!*

- Build applications on a Summit compile node.
  - From login node, `ssh scompile`

- You must use the “new” hierarchical modules: `/curc/tools/utils/switch_lmod.sh`

- Run `ml avail` to see compiler options, load one, run `ml avail` to see MPI options, load one, run `ml avail` to see dependent modules, load one or more

- `module spider` helps search for dependent modules

Allocations - overview

- An allocation of CPU time will be required in order to run on Summit
  - Based on required core-hours, scaled to Service Units (SU)
- Janus allocations do not carry forward to Summit

- Three basic types of allocation accounts:
  - General – low priority; doesn't require a proposal
  - Project – higher priority based on needs and on quality of proposal
  - Condo – for researchers who buy Summit nodes; no proposal required

Allocations - implementation

- **Fairshare:**

- Slurm scheduler's "fairshare" setting will be configured with a target number of core-hours for each allocation
  - This target corresponds to the number of core-hours awarded
- Slurm changes each job's queue priority in an attempt to reach the target
  - Lower recent usage implies a priority boost
  - Higher recent usage results in reduced priority in order to allow other projects to reach their targets

Allocations – fairshare tree

- Root share 100%

- CU-Boulder share (67.5%)
  - General (20% of 67.5%)
  - Projects (80% of 67.5%)
    - UCB1
    - UCB2 …

- CSU share (22.5%)
  - General (20%)
  - Projects (80%)
    - CSU1
    - CSU2 …

- RMACC share (10%)
  - General (20%)
  - Projects (80%)
    - RMACC1
    - RMACC2 …

- Condo shares (variable)

Submitting jobs with Slurm

- On login node, `module load slurm/summit`
- On compile node, Slurm module not required
- Need to load modules in the job script – *modules and environment variables will not be propagated from your login or compile node!*

Slurm Partitions and QoSes

Partition	Node type	Max Wall Time	Billing factor
shas	General compute (Haswell)	24 hr	1
sgpu	GPU-enabled	24 hr	2.5
smem	High-memory	7 day	6
sknl	Phi (Knights Landing)	24 hr	TBD

QoS	QoS Time Limit	Priority Direction	Max Jobs/User	Max Nodes/User
normal	partition max	0	1000	256
long	7 day	+	100	22
debug	1 hr	++	1	32
condo	partition max	+	1000	256

Thank you!   Questions?

CU-Boulder Research Computing

www.rc.colorado.edu   rc-help@colorado.edu

Link to survey on this topic:
http://tinyurl.com/rcpresurvey