Supercomputing: An Overview

Shelley Knuth
shelley.knuth@colorado.edu

Peter Ruprecht
peter.ruprecht@colorado.edu

www.rc.colorado.edu

Questions? #RC_Meetups

Link to survey on this topic: http://goo.gl/forms/8VidcwOhRT

Slides: https://github.com/ResearchComputing/Final_Tutorials

What Is a Supercomputer?

- A supercomputer is one large computer made up of many smaller computers and processors
- Each different computer is called a node
- Each node has processors/cores
  - Carry out the instructions of the computer
- With a supercomputer, all these different computers talk to each other through a communications network
  - Example - InfiniBand

Computers and Cars - Analogy

Computers and Cars - Analogy

-

Why Use a Supercomputer?

- Supercomputers give you the opportunity to solve problems that are too complex for the desktop
  - Might take hours, days, weeks, months, years
  - If you use a supercomputer, might only take minutes, hours, days, or weeks
- Useful for problems that require large amounts of memory

World's Fastest Supercomputers

www.top500.org June 2016

Rank	Site	Name	TeraFlops
1	National Supercomputing Center (Wuxi, China)	Sunway	125435.9
2	National Super Computer Center (Guangzhou, China)	Tianhe-2	54902.4
3	Oak Ridge National Laboratory (United States)	Titan	27112.5
4	DOE/NNSA/LLNL (United States)	Sequoia	20132.7
5	RIKEN Advanced Institute for Computational Science (Japan)	K	11280.4
6	DOE/Argonne National Lab (United States)	Mira	10066.3
7	DOE/NNSA/LANL/SNL (United States)	Trinity	11078.9
8	Swiss National Supercomputing Centre (Switzerland)	Piz Daint	7788.9
9	HLRS - Höchstleistungsrechenzentrum Stuttgart (Germany)	Hazel Hen	7403.5
10	King Abdullah University of Science and Technology (Saudi Arabia)	Shaheen II	7235.2

What Does It Mean to Be Fast?

- Titan can do 27 trillion calculations per second
- A regular PC can perform 17 billion per second
- Researchers can get access to some of these systems through XSEDE (The Extreme Science and Engineering Discovery Environment)

Different Node Types

- Login nodes
  - This is where you are when you log in
  - No heavy computation, interactive jobs, or long running processes
  - Script or code editing, minor compiling
  - Job submission
- Compute/batch nodes
  - This is where jobs that are submitted through the scheduler run
  - Intended for heavy computation

Storage Spaces

- System variations
- **Home Directories**
  - Store source code
  - Not for direct computation
  - Small quota (~5 GB)
  - Backed up
- **$WORK Space**
  - Mid level quota (~300 GB)
  - Large file storage
  - Not backed up

- **Scratch Directory**
  - Much larger – depends on system
  - Output from running jobs should go here
  - Files generally purged at some point

What is Job Scheduling

- Supercomputers usually consist of many nodes
- Users submit jobs that may run on one or multiple nodes
- Sometimes these jobs are very large; sometimes there are many small jobs
- Need software that will distribute the jobs appropriately
  - Make sure the job requirements are met
    - Reserve nodes until enough are available to run a job
    - Account for offline nodes
- Also need software to manage the resources
- Integrated with scheduler

Job Scheduling

- On a supercomputer, jobs are scheduled rather than just run instantly at the command line
  - People “buy” time to use the resources
  - Shared system
  - Request the amount of resources needed and for how long
  - Jobs are put in a queue until resources are available
  - Once the job is run they are “charged” for the time they used

Job Scheduling - Priority

- What jobs receive priority?
  - Can depend on the center
  - Can arrange for certain people who “pay more” receive priority
  - Generally though based on job size and time of entry
- Might have different queues based on different job needs
- Can receive priority on a job by creating a reservation

Job Schedulers - Slurm

- Jobs on supercomputers are managed and run by different software

- Simple Linux Utility for Resource Management (Slurm)
  - Open source software package

- Slurm is a resource manager
  - Keeps track of what nodes are busy/available, and what jobs are queued or running
- Slurm is a scheduler
  - Tells the resource manager when to run which job on the available resources

Running Jobs

- What is a “job”?
- Interactive jobs
  - Work interactively at the command line of a compute node
- Batch jobs
  - Submit job that will be executed when resources are available
  - Create a text file containing information about the job
  - Submit the job file to a queue
- Load the Slurm module!

Queues

- There are several ways to define a “queue”
- Clusters may have different queues set up to run different types of jobs
  - Certain queues might exist on certain clusters/resources
  - Other queues might be limited by maximum wall time
- Slurm can use a “quality of service” for each queue
  - aka “QOS”
- Also can use a “partition” (or set of nodes) that corresponds to a queue

Submit Batch Job example

```
#!/bin/bash
#SBATCH –N 2                                  #No. nodes
#SBATCH --ntasks-per-node=12                  #No. cores
#SBATCH --time=1:00:00                        #Max walltime
#SBATCH --job-name=SLURMDemo                  #Job name
#SBATCH --output=SLURMDemo.out                #Output file name
###SBATCH -A <account>                        #Allocation
###SBATCH --mail-type=end                     #Send Email completion
###SBATCH --mail-user=<your@email>            #Email address

ml intel
ml openmpi/1.8.5

mpirun ./hello
```

Submit Batch Job example

- Have to make sure the slurm module is loaded!
- Submit the job, and specify the queue:

```
sbatch --qos janus-debug slurmSub.sh
```

  - Demonstrates that you can add slurm functions at the command line or in the bash script

- Check job status in the janus-debug queue:

```
squeue –q janus-debug
```

- Check output:

```
cat SLURMDemo.out
```

Your Turn

- Submit a slurm job with the following instructions:

1. The job should run the Unix "hostname" command
   - Hint – the command "srun" will run commands in slurm
2. The job should be submitted from a bash script named practice.sh
   - Don't forget to make it executable!
3. The job should run for 5 minutes in the default queue
4. The job should be run on 1 node
5. The output should be put in a file called hostname.txt

Your Turn - Solution

Bash Script practice.sh:

```
#!/bin/bash
#SBATCH -N 1                      # No. of nodes
#SBATCH --time=0:05:00            # Walltime
#SBATCH --output=hostname.txt     # Output file name

srun hostname
```

Submit the job:

```
sbatch practice.sh
```

Questions?

- Email rc-help@colorado.edu
- Twitter: CUBoulderRC

- Link to survey on this topic: http://goo.gl/forms/8VidcwOhRT

- Slides: https://github.com/ResearchComputing/Final_Tutorials