ECE 574 – Cluster Computing Lecture 4

Vince Weaver

http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

31 January 2019

Announcements

- I'll post Homework #2, will send e-mail, submit via e-mail

- Barrows Hall power shutdown this weekend (likely Sunday 9am-noon) so don't plan on doing your homework then. Hopefully the machine will come back up.

Cluster Accounts

- Log in to weaver-lab. Be sure to use port 2131 or it will try to connect to the wrong machine. (Why?)

- Behave. No hacking/cracking/spamming/irc-bots Also be responsible with disk usage, as I don't have disk quota set up.

- Change your password first thing.

- If you find a security bug, great! Let me know! Don't

go deleting things or impersonating people or installing root kits, or other stuff.

Things from last time

- Performance counters in the cloud. Why we are doing assignments on my computer rather than on the UMaine supercomputer.

- Performance counters and interpreted code. Can you use it to help find bugs in java or python etc? Trickier.

Speedup / Parallel Efficiency Example

- Reminder
  - Speedup $= S_p = \frac{T_s}{T_p}$
    where p=# of processes (threads)
    $T_s =$ execution time of sequential code
    $T_p =$ execution time of parallel with p processes
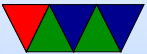   For ideal, $S_p = p$
  - Parallel Efficiency
    $E_p = \frac{S_p}{p} = \frac{T_s}{pT_p}$
    Ideal linear speedup E_p=1

- Examples where serial code takes 120s, p=2
  - $T_2 = 150s$, $S_p = \frac{120}{150} = 0.8$, $E_p = \frac{.8}{2} = .4$
  - $T_2 = 120s$, $S_p = \frac{120}{120} = 1$, $E_p = \frac{1}{2} = .5$
  - $T_2 = 60s$, $S_p = \frac{120}{60} = 2$, $E_p = \frac{2}{2} = 1$
  - $T_2 = 30s$, $S_p = \frac{120}{30} = 4$, $E_p = \frac{4}{2} = 2$

Running Linpack

- HPL solves linear system of equations, Ax=b. LU factorization.
- Download and install a BLAS. ATLAS? OpenBLAS? Intel?
  Compiler? intel? gcc? gfortran?
- Download and install MPI (we'll talk about that later). MPICH? OpenMPI?
- Download HPL. Current version 2.2?
  Modify a Makefile (not trivial) make sure links to proper

BLAS. make arch=OpenBLAS
- Above step, might need to create a link from hpl in your home directory to actual location for reasons
- Create a a bin/OpenBLAS with default HPL.dat file
- Run it `./xhpl` Or if on cluster `./mpirun -np 4 ./xhpl` or similar.
- Result won't be very good. Need to tune HPL.dat
- N is problem size. In general want this to fill RAM. Take RAM size, squareroot, round down. NxN matrix. Each N is 8 bytes for double precision.
- NB block size, can be tuned

- PxQ, if on cluster can specify machine grid to work on. Linpack works best with as square as possible.
- Fiddle with all the results until you get the highest.

Commodity Cluster Layout
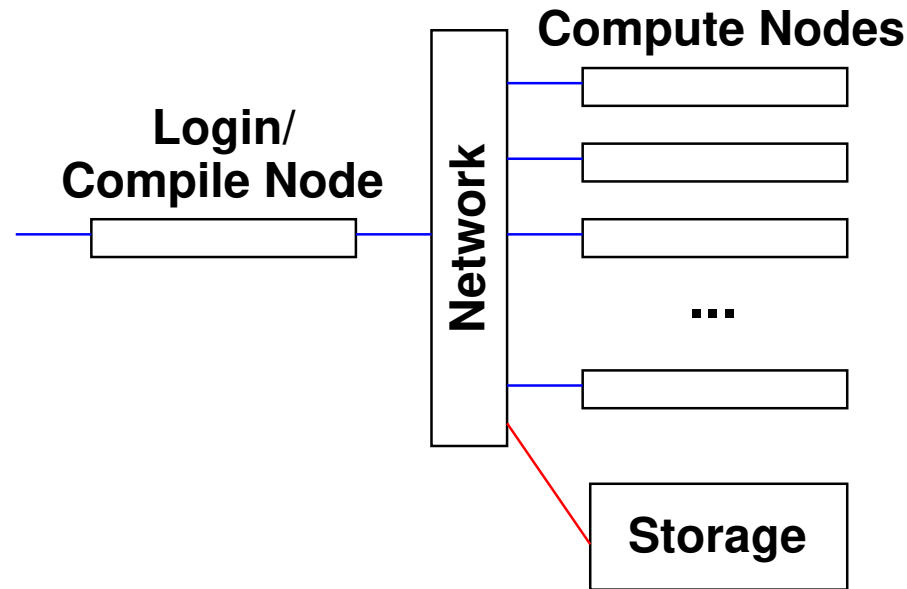
- Simple cluster like the pi-cluster, or older ones I've made
- Commodity cluster design is a combo of ECE331/ECE435 more than anything else

- Why have a head node?
- What kind of network? Ethernet? Inifiniband? Something fancier?
- Operating system? Do all nodes need a copy of the OS? Linux? Windows? None?
- Booting: network boot, local disk boot.
- Network topology? Star? Direct-connect? Cube? Hyper-cube?
- Disk: often shared network filesystem. Why? Simple: NFS (network file system). More advanced cluster filesystems available.

- Don't forget power/cooling
- Running software?

Job Schedulers

- On a big cluster, how do you submit jobs?
- If everyone just logged in to nodes at random, would be a mess
- Batch job scheduling
- Different queues (high priority, long running, etc)
- Resource management (make sure don't over commit, use too much RAM, etc)
- Notify you when finished?
- Accounting (how much time used per user, who is going

to pay?)

Scheduling

- Different Queues Possible – Low priority? Normal? High priority (paper deadline)? Friends/Family?

- FIFO – first in, first out

- Backfill – bypass the FIFO to try to efficiently use any remaining space

- Resources – how long can run before being killed, how many CPUs, how much RAM, how much power? etc.

- Heterogeneous Resources – not all nodes have to be same. Some more cores, some older processors, some GPUs, etc.

Common Job Schedulers

- PBS (Portable Batch System) – OpenPBS/PBSPro/TORQU

- nbs

- slurm

- moab

- condor

- many others

Slurm

- `http://slurm.schedmd.com/`

- Slurm Workload Manager
  Simple Linux Utility for Resource Management
  Futurama Joke?

- Developed originally at LLNL

- Over 60% of top 500 use it

sinfo

provides info on the cluster

```
PARTITION AVAIL   TIMELIMIT   NODES   STATE NODELIST
debug          up    infinite       1    idle haswell-ep
general*       up    infinite       1    idle haswell-ep
```

srun

start a job, but interactively

sbatch

submit job to job queue

```
#!/bin/bash

#SBATCH -p general          # partition (queue)
#SBATCH -N 1                    # number of nodes
#SBATCH -n 8                    # number of cores
#SBATCH -t 0-2:00        # time (D-HH:MM)
#SBATCH -o slurm.%N.%j.out # STDOUT
#SBATCH -e slurm.%N.%j.err # STDERR
export OMP_NUM_THREADS=4
./xhpl
```

  Notes: sbatch -N 24 - -ntasks-per-node=4 ./time_
To run on all 96 cores of pi-cluster

Can set up to e-mail you when done (though only locally).

squeue
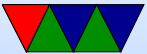
```
JOBID PARTITION      NAME      USER ST       TIME   NODES NODELIST(REASON)
   63    general time_hpl ece574-0 PD       0:00       1 (Resources)
   64    general time_hpl ece574-0 PD       0:00       1 (Resources)
   65    general time_hpl ece574-0 PD       0:00       1 (Resources)
   62    general time_hpl ece574-0  R       0:14       1 haswell-ep
```

scancel

kills job

```
scancel 65
```

Average Machine Speeds

- Look up my top40 list. Green and regular. Compare with top and bottom of top500. Also Pi-cluster

- Computers we might use in class:
  haswell-ep server 436 GFLOPS, 16/32 cores, 80GB, 2.13GFLOP/W
  power8 machine 195 gflops, 8/64 cores, 32GB, ??
  pi-cluster, 15.4 GFLOPS, 96 cores, 24GB RAM, 0.166 GFLOP/W
  pi-3B 3.62 GFLOPS, 4 cores, 1GB RAM, 0.813

GFLOP/W (higher possible)

Reminder, top machine, 93 PFLOPS, sunway, 6GFLOPS/W (top 10 3-9 GFLOPS/W)

- First list, June 1993. Top machine 1024 cores, 60 GFLOPS, 131kW
  Pi cluster would have been #7

- I ran HPCG benchmark on Haswell-EP machine. Theoretical: 16DP FLOP/cycle * 16 cores * 2.6GHz = 666 GFLOPS

Linpack/OpenBLAS: 436 GFLOPS (65% of peak),
HPCG: 0.7 GFLOPS (0.1% of peak)