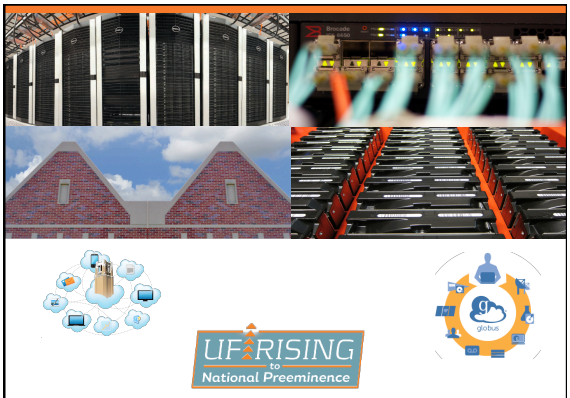
Optimizing Batch Submission and Job Performance

Matt Gitzendanner: magitz@ufl.edu

2/25/15

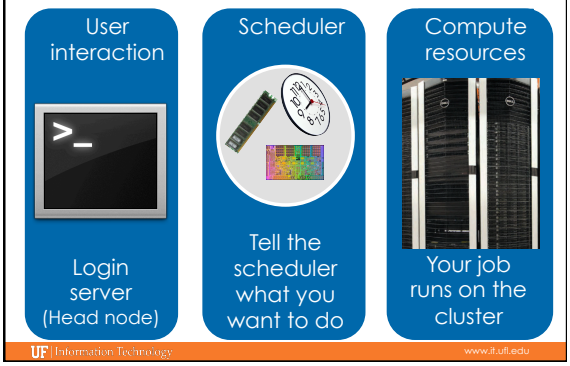


HiPerGator

The University of Florida Supercomputer for Research

- 16,384 cores (total of ~20,000 today)
- Infiniband interconnect
- >3PB fast, high-availability, storage

Cluster basics



Scheduling a job

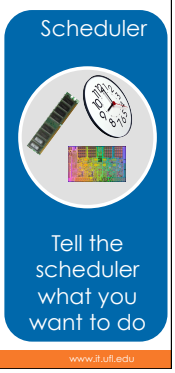
- Need to tell scheduler what you want to do
  - **How many CPUs** you want and how you want them grouped
  - **How much RAM** your job will use
  - **How long** your job will run
  - The commands that will be run

UF Research Computing

- Submission Script

```
#!/bin/bash
#PBS -N My_Job_Name
#PBS -M Joe_Shmoe@ufl.edu
#PBS -m abe
#PBS -o My_Job.log
#PBS -e My_Job.err
#PBS -l nodes=1:ppn=1
#PBS –l pmem=900mb
#PBS -l walltime=00:05:00

cd $PBS_O_WORKDIR
date
module load test_app
test_app -i file.txt
```



How do you know???

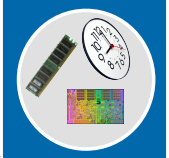
- Experience
- Trial and Error
- Run on a development node:

```
$ ssh dev1
$ module load my_app
$ my_app -i file1.txt &
$ top
```

```
top - 15:30:21 up 196 days, 21:26, 44 users,  load average: 1.09, 4.71, 8.34
Tasks: 1874 total,   2 running, 1864 sleeping,   8 stopped,   0 zombie
Cpu(s):  1.6%us,  0.2%sy,  0.0%ni, 98.2%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  264523424k total, 180940792k used, 83582632k free,   386552k buffers
Swap:  8388600k total,  2031204k used,  6357396k free, 72171180k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
45338 jmeyer    20   0 92.6g  85g 4640 R 99.2 33.9   2454:48 mira
24915 nalex     20   0  138m 9824 2192 S  6.9  0.0   8:03.62 perl
   41 root      20   0     0    0    0 S  2.6  0.0   1121:42 ksoftirqd/9
40467 magitz    20   0 18476 2672  976 R  2.0  0.0   0:00.61 top
   37 root      20   0     0    0    0 S  1.6  0.0   1157:49 ksoftirqd/8
26491 greulich  20   0 18480 2716  988 S  1.6  0.0   1:33.53 top
40493 arnnh     20   0  144m 5908 2180 S  1.3  0.0   0:00.04 vim
```

Scheduler



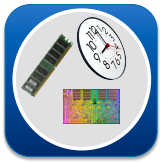
Tell the scheduler what you want to do

How do you know???

- Start general and refine

```
#PBS –M magitz@ufl.edu
#PBS -m abe
#PBS -l nodes=1:ppn=1
#PBS –l pmem=4gb
#PBS -l walltime=24:00:00
```

- Look at the ending or abort email for time and ram usage and adjust



```
Job deleted at request of root@moab.ufhpc
job 4189661 exceeded MEM usage hard limit (2325 > 2252)
```

(MB)

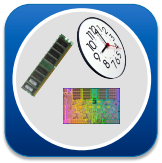
End-of-job emails:

```
#PBS -M Joe_Shmoe@ufl.edu
#PBS -m abe
```

```
PBS Job Id: 358634.moab.ufhpc
Job Name:   NR.25.nex
Exec host:  c7a-s1/60
Execution terminated
Exit_status=0
resources_used.cput=07:16:09
resources_used.mem=251348kb
resources_used.vmem=318916kb
resources_used.walltime=07:16:52
```


How do you know???

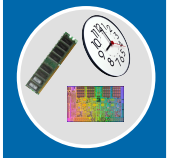
- **Common misconceptions**
  - More cores (processors) will make my application run faster
  - More RAM will make my application run faster
  - *The University of Florida Supercomputer for Research* will run my application faster than my laptop



Scheduling a job

- Need to tell scheduler what you want to do
  - **How many CPUs** you want and how you want them grouped
  - **How much RAM** your job will use
  - **How long** your job will run
  - The commands that will be run

Scheduler



Tell the scheduler what you want to do

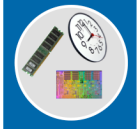
Walltime

```
#PBS –l walltime=00:50:00
```

- Backfill- use ~ accurate walltimes
- MAXPS: 3days*10*investment
  - Long running jobs won't start in "burst" capacity
- Last 3 days of use factors into priority

	Maximum	Short	Long
Investor	31 days	<12 hrs	7 days
Other	7 days	<12 hrs	3 days

Scheduler



Tell the scheduler what you want to do

RAM

#PBS –l pmem=900mb

- Lots to consider, but do your best at estimating RAM needed for job
- Over about 4GB of RAM, "costs" toward CPU allocation

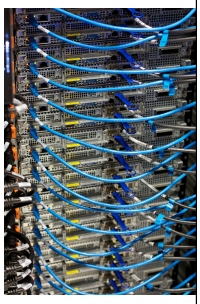
Wasted RAM leads to idle CPUs and low job throughput

Nodes and processors

Single processor apps:
#PBS –l nodes=1:ppn=1

Threaded (& MPI) apps:
#PBS –l nodes=1:ppn=4

MPI apps:
#PBS –l nodes=2:ppn=32

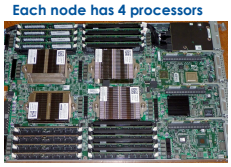


Nodes—Processors—"Cores"

A compute node or server

Most HiPerGator nodes have 64 cores or ppn and 256GB of RAM

Each node has 4 processors



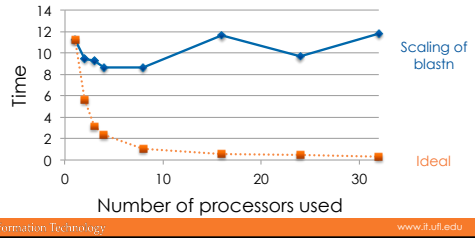
The scheduler uses "processors" where most think of "cores".

The processor request (ppn) is what most people think of as cores.

Each processor has 16 "cores"

Processor Requests

- Is your application parallel?
- Can it use CPUs on multiple nodes?
- How well does it scale?



HiPerGator

The University of Florida Supercomputer for Research

- ***64 cores per node***
  - If RAM allows, MPI jobs under ~32 cores, should use nodes=1:ppn=##
  - Some older nodes have 4-16 cores

Passing variables in qsub

```
#!/bin/bash
#PBS -N My_Job_Name
#PBS -M Joe_Shmoe@ufl.edu
#PBS -m abe
#PBS -o My_Job.log
#PBS -e My_Job.err
#PBS -l nodes=1:ppn=1
#PBS –l pmem=900mb
#PBS -l walltime=00:05:00

cd $PBS_O_WORKDIR
date
echo Running with input file: $infile
module load test_app
test_app -i $infile
```

- `qsub test.pbs -v infile=myfile1.txt`
- `-v x=3,y=5`

Task Arrays

- `#PBS -t 1-1000` Runs 1000 tasks all submitted at once
- `#PBS -t 1-1000%20` Will throttle to run 20 tasks at at time
- `$PBS_ARRAYID`

```
#PBS -t 1-100%10
cd $PBS_O_WORKDIR
module load my_app
file=`ls *.txt | head -n $PBS_ARRAYID | tail -n 1`
my_app -i $file
```


Task Arrays

- The "[]" is part of the job number
  - `qdel 1234[]`
  - `qdel -t 45 1234[]`
  - `qdel -t 1,10,50-100 1234[]`
  - `qstat -t 1234[]`
- Resource requests are per task
  - `#PBS -l nodes=1:ppn=4`
  - Each task gets 4 processors
- Can't assume tasks are executed in order
- Limit total tasks to 2,000-3,000

Checking resources

- How many jobs are running in my group?
  - `showq -w group=<group_name>`
- How many resources will this job take?
  - `pbs_info -f script_file.pbs`
- Why isn't this job running?
  - checkjob -v <job_id>
  - `NOTE: job violates constraints for partition base (job 46226XX violates active HARD MAXPE limit of 310 for group XXXX partition ALL (Req: 1 InUse: 310))`

Some helpful environment variables

- `$PBS_O_WORKDIR` : the directory where you typed qsub
- `$PBS_JOBID` : the unique job id: e.g. 24461774.moab.ufhpc
- `$TMPDIR` : temporary directory for each job on compute node's local disk, good for jobs with lots of small I/O
- `$PBS_NUM_PPN` : Number of processors for single node job, use this when starting a threaded application to tell it how many processors to use. Prevents needing to change in multiple places. E.g. nodes=1:ppn=4 , `blastn -num_threads $PBS_NUM_PPN`
- `$PBS_JOBNAME` : Name your gave your job with #PBS –N