CENTER FOR
GENOME RESEARCH &
BIOCOMPUTING

"Introduction to Unix/Linux"
INX_U18, Day 9, 2018-08-13

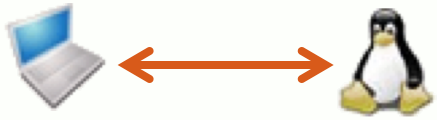
Sun Grid Engine (SGE) (Oracle Grid Engine (OGE)), q* commands, resources

Learning Outcome(s):
Submit batch jobs to a computational infrastructure, i.e.,
submitting batch jobs to the Sun Grid Engine (SGE) queuing system
to run (noninteractively) on cluster nodes

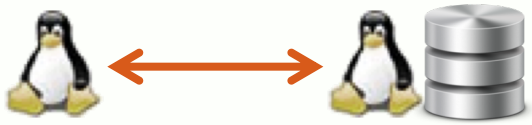
Matthew Peterson, OSU CGRB, matthew@cgrb.oregonstate.edu

Building upon: Clients and Servers

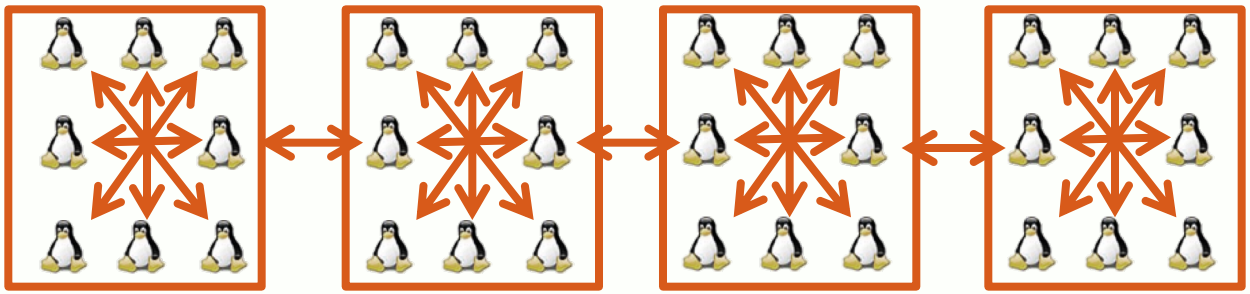
- **Servers** and **clients**, e.g.



  - Your laptop connects (via SSH) to the shell server
  - The shell server connects to your `$HOME` directory from another file server.



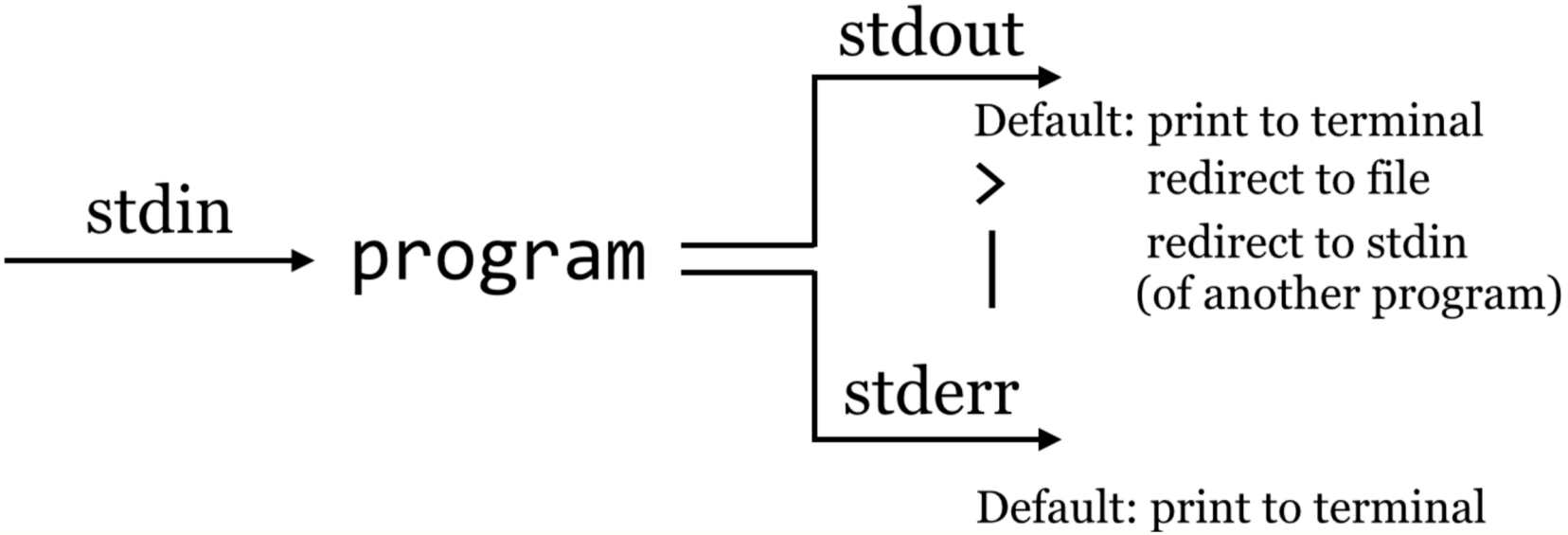
  - There are 100s of servers in the infrastructure, which all "talk" to each other.



Oregon State UNIVERSITY

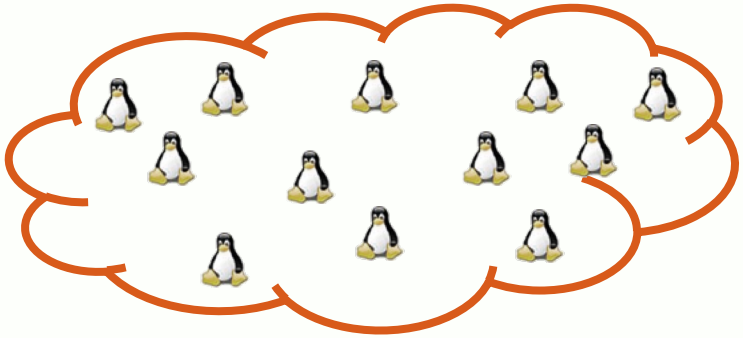
Building upon: Streams

- When you submit "jobs" to run on the infrastructure it handles the data streams for you, i.e., **stdout** and **stderr**



High Performance Computing

- **HPC** is filled with "buzzwords" over the years:
  - 1990s: **Cluster**
  - 2000s: **Grid**
  - 2010s: **Cloud**



- For **HPC** experts these terms are distinct but for us the refer to *approximately* the same idea:
  - A group of computers we can access to store or process data, without having to think much about them individually.

Oregon State UNIVERSITY

Batch Queuing Systems

- May institutions have a **batch queuing system** to "**submit jobs**" to its servers in its "***cloud***."
- There are many different batch queuing systems:
  - Sun Grid Engine (**SGE**) – What the CGRB uses
    - aka Oracle Grid Engine (**OGE**)
    - aka Univa (commercial version)
  - Portable Batch System (**PBS**)
  - Torque (a derivative of PBS)
  - Condor

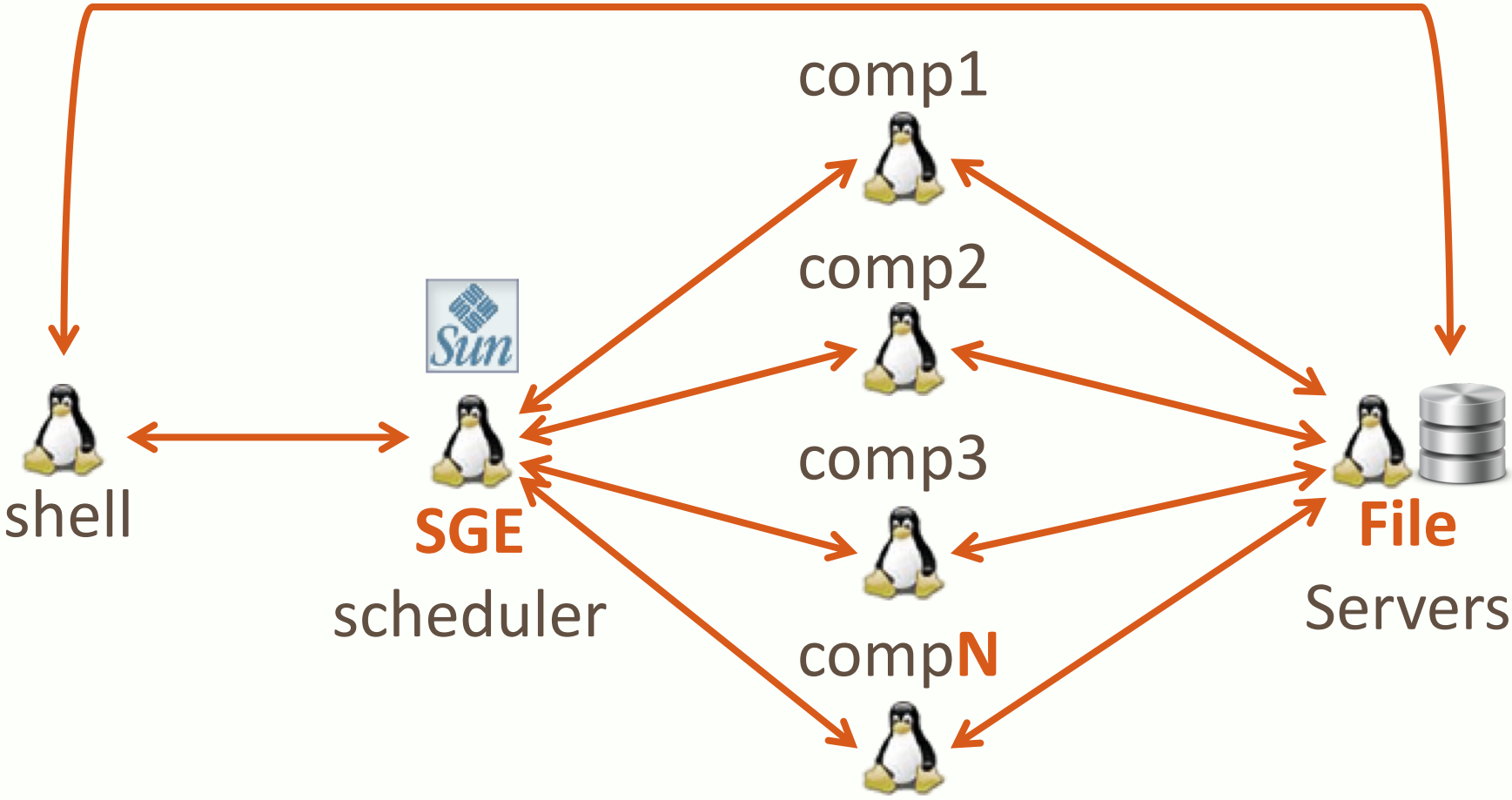
Oregon State UNIVERSITY

Other Distributed Computing

- There are other ways to do **distributed computing** than **batch queuing systems**, e.g.,
  - Google Map-Reduce
  - Apache Hadoop
  - Amazon AWS Lambda
- Not all software can take advantage of the parallelization offered by these distributed computing solutions.
  - Custom programming may be required.

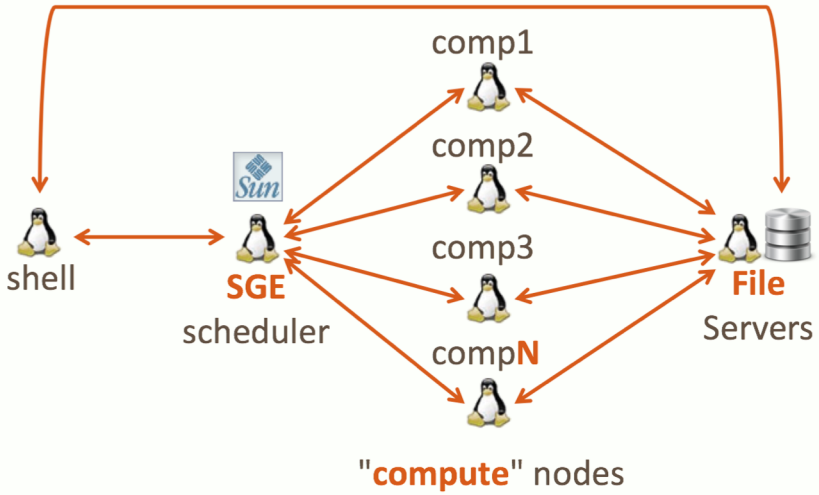
Oregon State UNIVERSITY

SGE layout


comp1
comp2
comp3
compN
shell
SGE
scheduler
File
Servers
"compute" nodes

Submitting "jobs"

- The **SGE** scheduler waits for "**jobs**" from shell
- Servers wait for "**jobs**" from the **scheduler**, e.g.,
  - `hmmsearch p450s.profile db.fasta`
  - How much free RAM do you have?
- Compute nodes access the same file servers



Example "job" submission

```
blastn -query q.fasta -db nt -numthreads 4 -out result.txt
```

1. User logs into front-end machine, e.g., shell
2. User runs a **client** command requesting the:
   - Program to run: **`blastn`** (above)
   - Resources: 4 CPUs, 5 GB RAM, kill >100GB files
3. The **client** sends this request to the **scheduler**, which assigns a **job #** and puts it in the **queue**.
4. The scheduler monitors the compute nodes if they are "free" and sends jobs there to run.
5. The compute node runs the **job**.

Oregon State UNIVERSITY

Running a "job"

Compute node runs a **job** assigned by the **scheduler**

- Runs the command, e.g., **`blastn`**
- Creates the output files, e.g., `result.txt`, which are written to the shared file servers.
  - These files are immediately viewable on shell in the **job**'s output directory, e.g., in your `$HOME`
- The **stdout** and **stderr** of your program are written to files in the **job**'s output directory.
  - They do not appear on your terminal!

Oregon State UNIVERSITY

`qsub` Submit a batch job

- To submit a **job** to the queue "*the hard way*"
  - This submission takes a shell script (that you write) of the command(s) you want run
  - Also takes the resources you are requesting, e.g., number of CPU cores, amount of RAM
- The CGRB has written a nice "wrapper" script (around **`qsub`**) to help you submit jobs!
  - We will cover **`SGE_Batch`** a bit later…

Oregon State UNIVERSITY

qsub Example submission

```
cat sgetest.sh
```

```
#!/bin/tcsh
echo "SGE Test!"
sleep 5
```

```
qsub sgetest.sh
Your job 3631185 ("sgetest.sh") has
been submitted
```

Oregon State UNIVERSITY

qsub Example output in $HOME

```
ls sgetest.sh.*
sgetest.sh.e3631185
sgetest.sh.o3631185

# Any stderr written to file; none in this case
wc -l sgetest.sh.e3631185
0 sgetest.sh.e3631185

# stdout written to file; may be warnings about job
cat sgetest.sh.o3631185
SGE Test!
```

Oregon State UNIVERSITY

`qstat` Status of "running" jobs

```
qstat                # Shows all of your "running"
qstat -u '*'         # Shows all jobs on system
```

Will show the **state** of the job:

- `r` Job is running on a compute node
- `qw` Waiting in queue to be matched to a node
- `eqw` Error, delete job and try resubmitting

Jobs that have completed **r**unning will no longer show up in `qstat`!

Oregon State UNIVERSITY

`qdel` Delete a submitted job

`qdel` `<job number>`

Delete a specific job number

`qdel -u` `<username>`

Delete all jobs for a given user
(You cannot delete other's jobs)

Oregon State UNIVERSITY

`qhost` Show resources available

`qhost`

Show resources for all available machines

`qhost -q`

Show resources for all available machines "by **`queue`**" (we will cover queues later).

`SGE_Avail`

A CGRB **`qhost`** "wrapper" script showing machine resources available to you.

Oregon State UNIVERSITY

`qrsh` Login to a node

`qrsh -l h=``"<hostname>"`

Interactively login to a machine you have access to, *reserves* "1 CPU" from the scheduler.

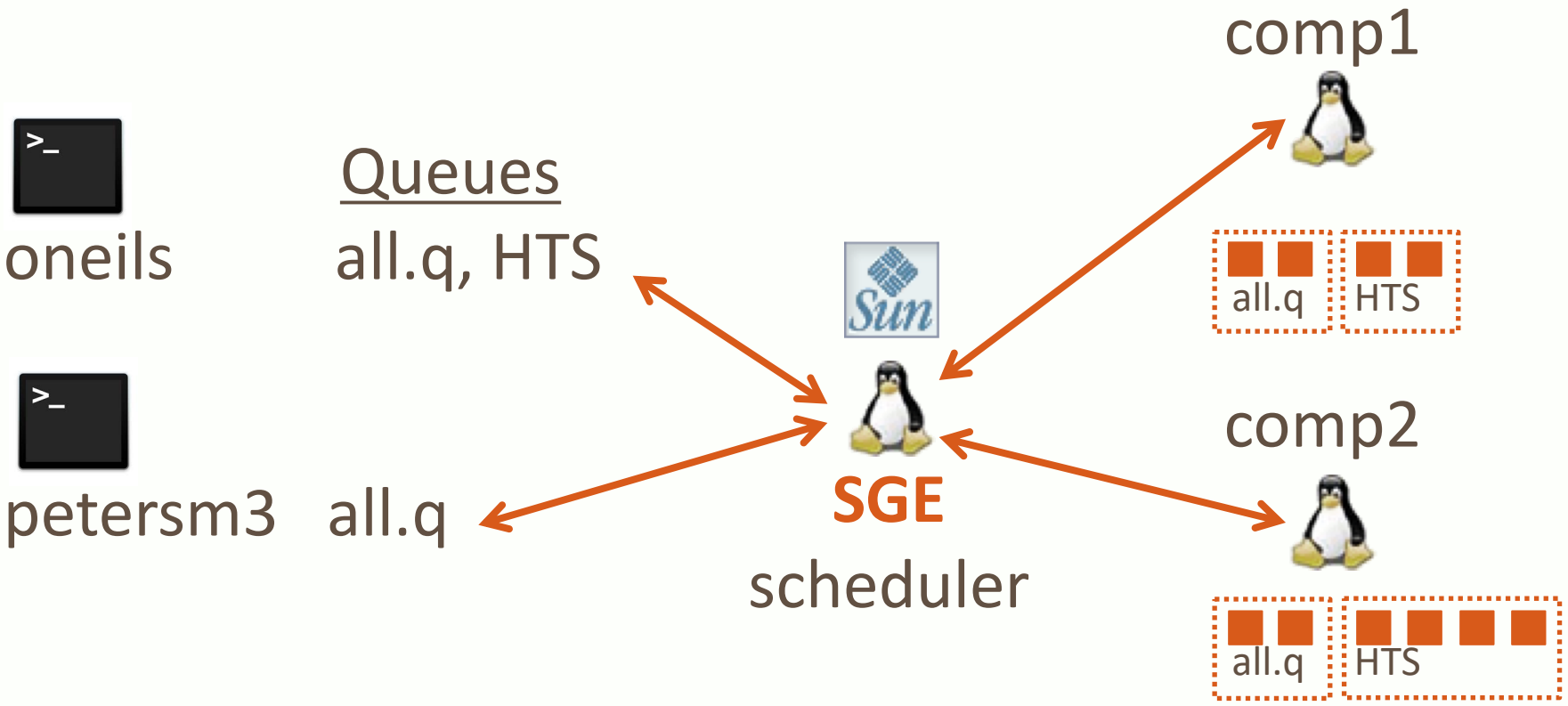
`qrsh -pe thread 4` `-l h="<hostname>"`

Reserves "**4** CPUs" from the scheduler.

NOTE: This is turned off on most nodes!
You *may* be granted access if your lab or department owns the compute node.

Oregon State UNIVERSITY

Sharing resources example



oneils requests 4 CPUs => HTS@comp2
oneils requests 2 CPUs => HTS or all.q @comp1 or 2
petersm3 requests 4 CPUs => Wait in queue forever
petersm3 requests 2 CPUs => all.q@comp1 or comp2

Oregon State UNIVERSITY

Queue types

Running **`SGE_Avail`** or **`qhost -q`** shows that the CPUs on a compute node can have 3 properties (under "ARCH" or "QTYPE"):

- **`B`** Allow for "batch" submission (we'll cover this later) via **`qsub`** or **`SGE_Batch`**
- **`I`** Allow for interactive use with **`qrsh`**

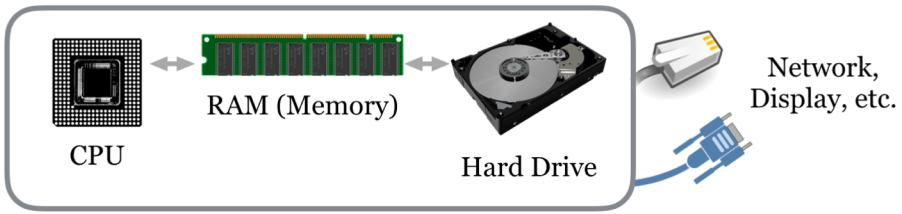
  Note: all.q CPUs do not have this turned on!
- **`P`** Allow for multiple-CPU jobs (almost all of them)

Oregon State UNIVERSITY

Computational bottlenecks

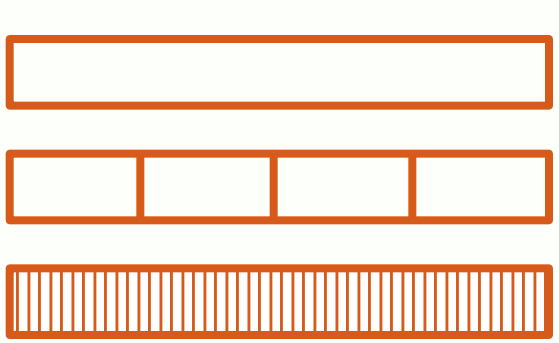
4 limitations on computers (that cost money):

- **RAM** (Physical **R**andom **A**ccess **M**emory)
  - Use it all and the machine *effectively* crashes.
- **File space** (disk)
  - Storage for your files
- **Network bandwidth**
  - File servers have finite bandwidth for N nodes
- **CPUs**
  - Number of CPU cores for processing.



Should I ask for more CPU cores?

- Compute nodes have a finite number of CPU cores (this varies from 8 to 64 or more).
- Assume you had a server with 1 CPU core and you submitted:
  - 1 job => 1 hour
  - 4 jobs => 4 hours each*
  - 1000 jobs => 1000 hours each*



* Plus "overhead" to time-share the CPU!

Oregon State UNIVERSITY

SGE_Batch

```
$ SGE_Batch

                SGE QSUB MENU
        (items with a "*" are required)

        *Command to Run:                c
         Max Memory (512M, 4G):         m
         Array Job Range (eg 1-100):            t
         Array Job Maximum Task Concurrency (max array jobs to run simultaneously) (default: 50):   tc
         Free Memory (512M, 4G):                f
         Max File Size (512M, 4G):              F
         Num. Processors (1,2,4 etc):   P
         QUEUE to Use:                  q
        *Run ID / Output:               r
         Job Priority:                  p
         Email Address:                 M
         Change Shell:                  S
         Show Setup Info:               h
         Submit Job:                    s
         Exit:                          e

   Please enter a letter corresponding to an option:
```

Oregon State UNIVERSITY

`SGE_Batch` Requestable resources

- `-P` Slots/CPUs
  - Run job on a node with this many CPUs *reserved*.
- `-f` Free RAM
  - Run job on node with this much free RAM at the time the job starts.
- `-m` Max RAM
  - If my job uses more than max RAM, kill it.
  - Ideally set Free RAM = Max RAM
  - Killed jobs create a core.<jobid> file

Oregon State UNIVERSITY

`SGE_Batch` Requestable limits

- `-F` Max file size
  - If any single file is created by the job that exceeds this size then kill the job.
- `-q` Queue to use
  - By default it uses the all.q

Oregon State UNIVERSITY

SGE_Batch Configurations

- **-c** Command (or script) to run *
  - A command wrapped in single quotes, e.g.,

```
'blastn -query p450s.fasta -db nt -num_threads 4 -out result.txt'
```

  - Or a script with your command(s) in it
- **-r** Run ID / Output *
  - Log directory to put your **stdout** and **stderr** in
  - Directory placed in the `$PWD` you start the job in
- **s** submit job

* Required

Oregon State UNIVERSITY

SGE_Batch Script submission

```
cat sgetest.sh
```

```
#!/bin/tcsh
echo "SGE Test!"
sleep 5
```

```
SGE_Batch -c 'sgetest.sh' -r MYLOG
```

Oregon State UNIVERSITY

SGE_Batch Command-line

```
makeblastdb -in pz_cDNAs.fasta -dbtype nucl
SGE_Batch -c 'blastn -query pz_cDNAs.fasta
-db pz_cDNAs.fasta -max_target_seqs 2 -max_hsps 1
-evalue 1e-6 -outfmt 7 -num_threads 4
-out pz_blastn_pz_top2.txt'
-P 4 -f 5G -m 5G -F 100G -r pz_blastn_pz_logs
```

- **-P 4** request 4 CPU slots
- **-f 5G** request 5 GB free RAM
- **-m 5G** kill at 5 GB RAM limit
- **-F 100G** kill at 100 GB file size limit
- **-r** `pz_blastn_pz_logs` put logs here

Oregon State UNIVERSITY

SGE `.sh` Log File

`pz_blastn_pz_logs_1_sge.sh`

`.sh` File (**shell script**) lists:

- The bash script created by **`SGE_Batch`** and submitted with **`qsub`**

SGE .o Log File

`pz_blastn_pz_logs.o3631395`

.o File (**stdout**) lists:

- Machine (node) it ran on
- Start time
- **stdout** (unless you redirected the output to a file)
- End time

Oregon State UNIVERSITY

SGE .e Log File

`pz_blastn_pz_logs.e3631395`

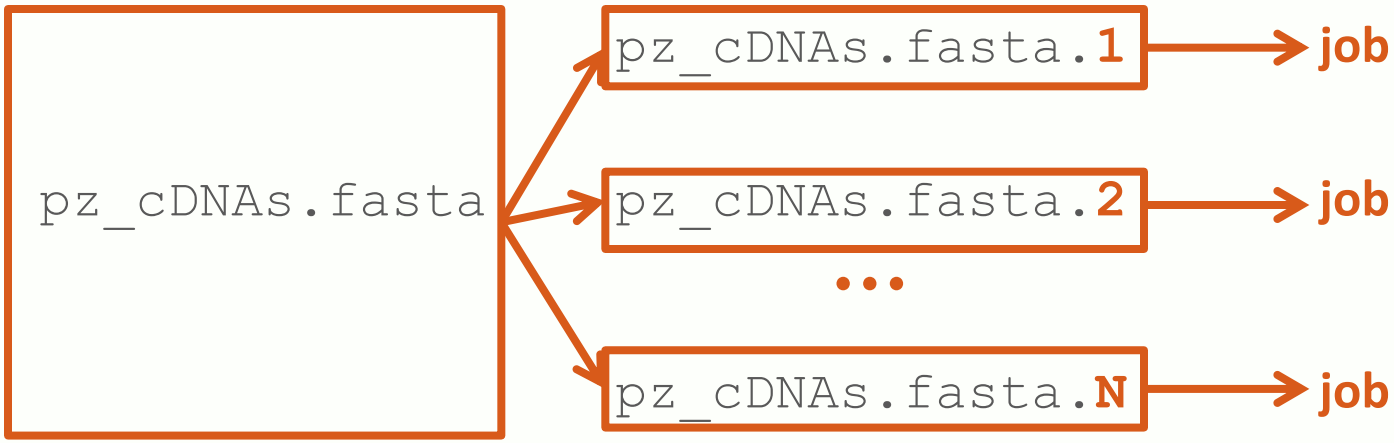
.e File (**stderr**) lists:

- **stderr** of the command (unless you redirected the output to a file)
- Amount of RAM used
- Amount of time taken

Oregon State UNIVERSITY

Array jobs with SGE_Batch

- Submitting large jobs may take a long time tor run
- Requesting more CPUs may result in a longer wait for a more powerful machine to become available
- Each BLAST query is "independent" so we can "break up" our job into smaller FASTA files



Array jobs BLAST goal

- Split our FASTA file into 20 individual pieces
- BLAST **query** `pz_cDNAs.fasta.X`
  - Produce `pz_blastn_pZ_top2.txt.X`
    - (**X** = 1, 2, … 20)
- We will only run small jobs asking for:
  - 1 CPU from BLAST via **`-numthreads 1`**
  - *Reserve* 1 CPU from SGE via **`-P 1`**

Oregon State UNIVERSITY

`SGE_Batch` can do Array jobs

1) Instead of specifying **X** for each FASTA file we can use the variable (in our script for both BLAST `-query` and `-out`): `$SGE_TASK_ID`
2) In `SGE_Batch` we set the `$SGE_TASK_ID` via: `-t 1-20`

- If running `SGE_Batch` from the command line (vs. inside a script) you want '$SGE_TASK_ID' in single quotes (so it's not expanded by the shell).
- You can do 2, 4, 6, ... 20 via `-t 2-20:2`

Oregon State UNIVERSITY

Array jobs output

- Logs in `pz_blastn_pz_logs` contain **stdout** (`.o`) and **stderr** (`.e`) files for each of the 20 **jobs**.
- Individual BLAST results, i.e., `pz_blastn_pz_top2.txt.N` (**N** = 1 .. 20)
- Since we used BLAST output format 7, which does not have a header, we can merge all of our outputs into a final result!

```
cat pz_blastn_pz_top2.txt.* \
    > ALL-pz_blastn_pz_top2.txt
```

Oregon State University

SGE_Plotdir

SGE_Plotdir pz_blastn_pz_logs

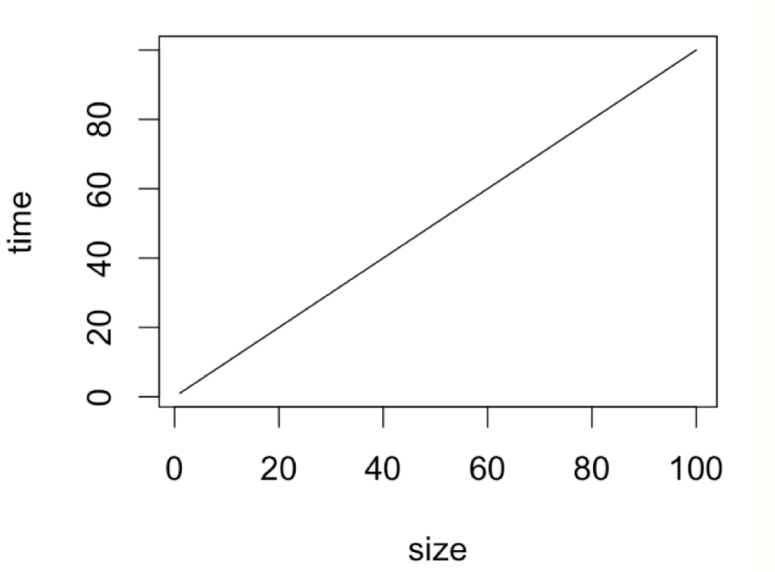
Displays:

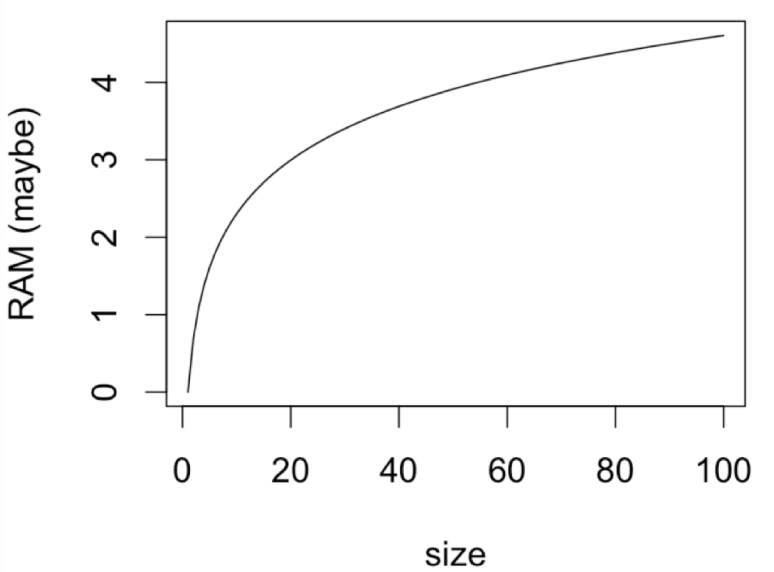
- RAM usage (for each **Array job**)
- Execution time (for each **Array job**)
- Will produce an ASCII plot if you submitted your job though **SGE_Batch** using Array jobs.

Oregon State UNIVERSITY

`SGE_Plotdir` Plots

- Running **Array jobs** (with different sized data sets) *may* reveal that **time** scales linearly with input size, but **RAM** usage *may* scale logarithmically.





- **"10/100/1000 rule"** – run tests on small subsets of data to extrapolate usage!

Oregon State UNIVERSITY

Command / Concept Review

- qsub / SGE_Batch
- qstat
- qdel
- qhost / SGE_Avail
- qrsh
- SGE_Plotdir
- $SGE_TASK_ID

Queues

BIP

.sh Log file

.o Log file

.e Log file

Array jobs

Oregon State UNIVERSITY