HPC at the Smithsonian

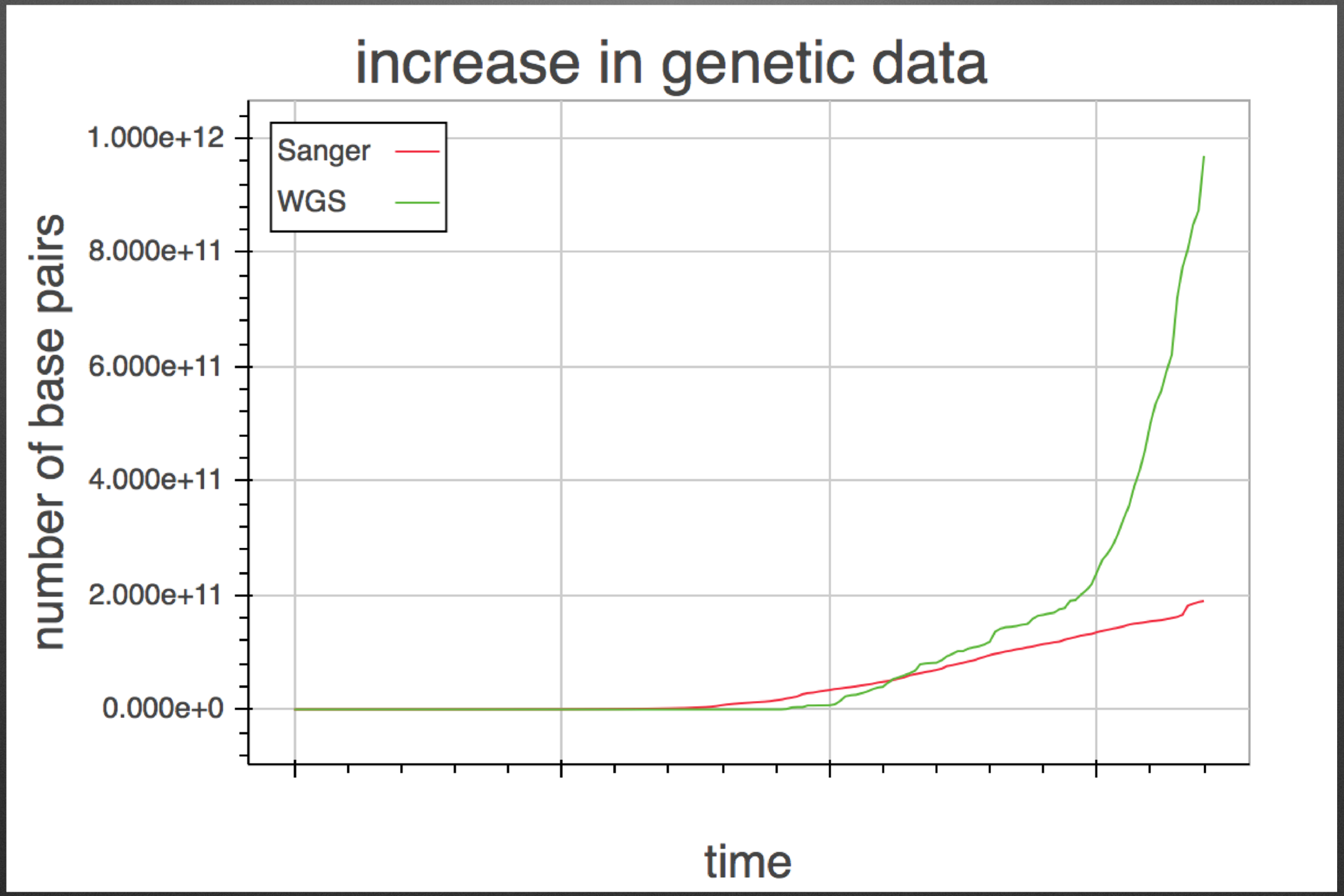
Introduction to Hydra

Overview

- Why HPC?
- SI Computing Cluster
- Support
- Cluster Architecture
- Submitting Jobs
- Best Practices
- Tips

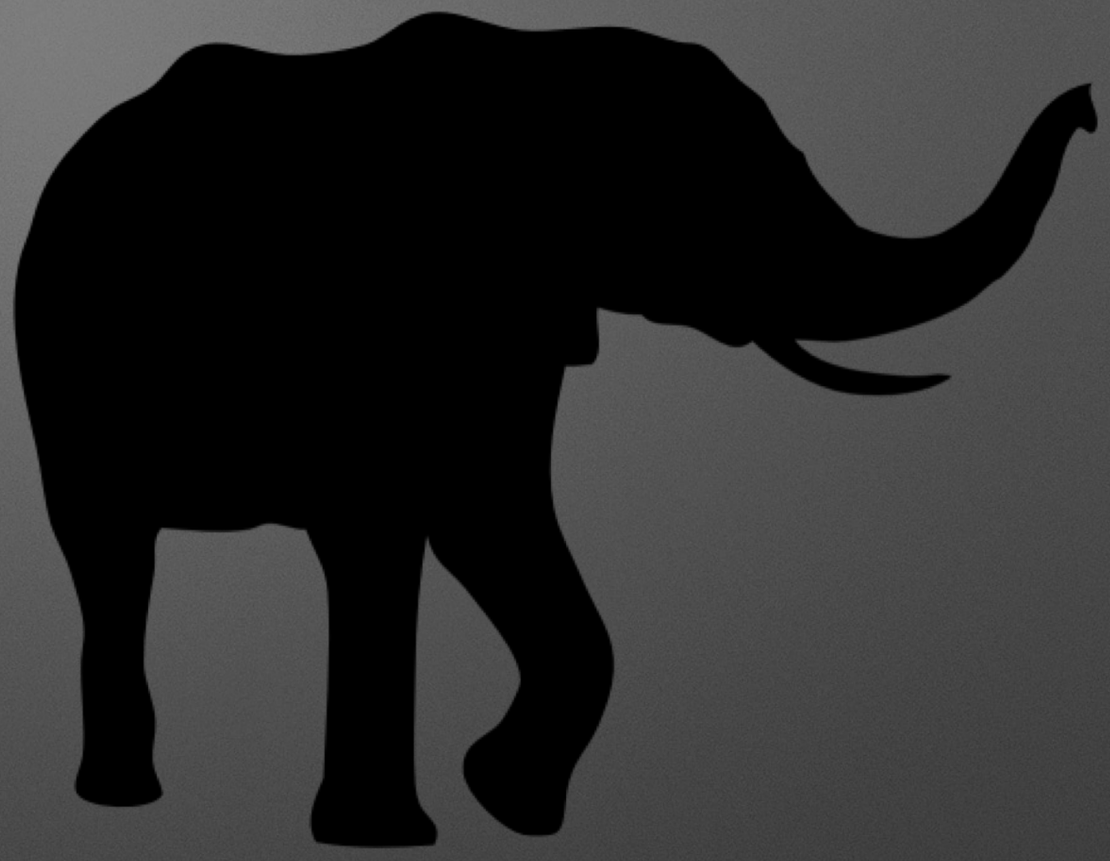
Why High Performance Computing (HPC)?

Why HPC?


increase in genetic data
Sanger
WGS
1.000e+12
8.000e+11
6.000e+11
4.000e+11
2.000e+11
0.000e+0
number of base pairs
time

Why HPC?

Size

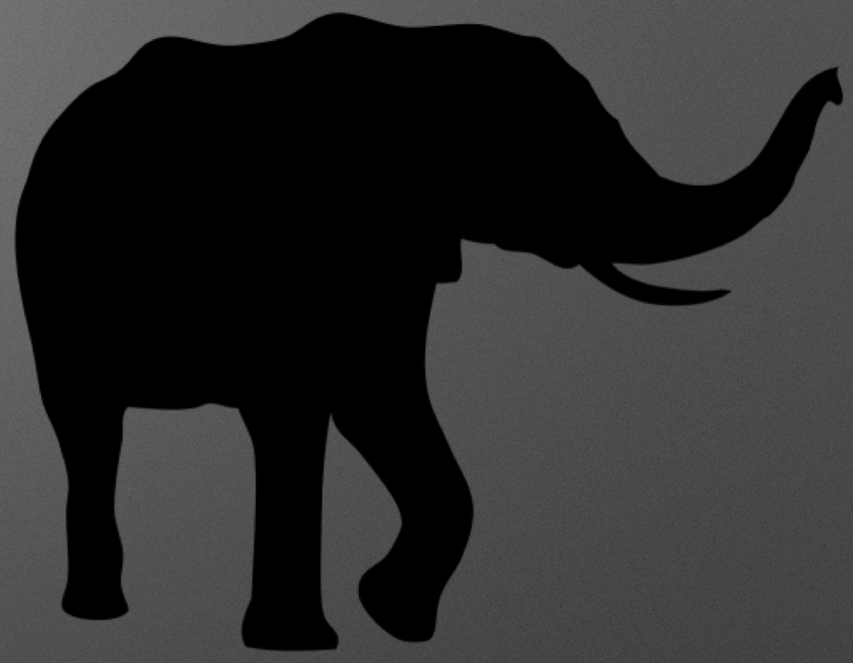


&

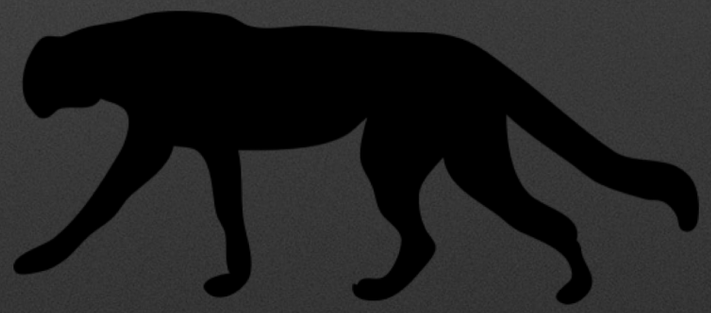
Speed

Why HPC?

- Size: some analyses use too much memory to be completed on a standard desktop/laptop, e.g. *de novo* genome assembly



- Speed: all about parallel computing! A problem that might take months on a desktop can take hours on a HPC, e.g. BLAST searches



SI Computing Cluster

Hydra

- Smithsonian HPC cluster
- 3,950 CPUs, 26 TB of RAM
- 16 high memory nodes (14 512GB; 2 1TB RAM; 1 2TB RAM)
- Many high CPU nodes with 64 CPUs

Cluster Architecture

Cluster Architecture

- Many computers connected to each other via an interconnection network (interconnect) with software that controls where jobs are run
- Each computer in the cluster is called a node
  - Login nodes: users log into these to access the cluster and launch jobs, they perform no real computation (hydra-login01.si.edu, hydra-login02.si.edu)
  - Compute nodes: actually run your program
  - Head node: runs the software that controls the cluster and the jobs

Queuing System/Job Scheduler

- Software that schedules the jobs and submits them to the compute nodes.
- Hydra uses the “(Sun) Grid Engine” queuing system.
- Most jobs should be run in batch mode. To submit jobs, you use the command "`qsub`"
- There is also an interactive mode for short jobs. To initiate, type “`qrsh`”

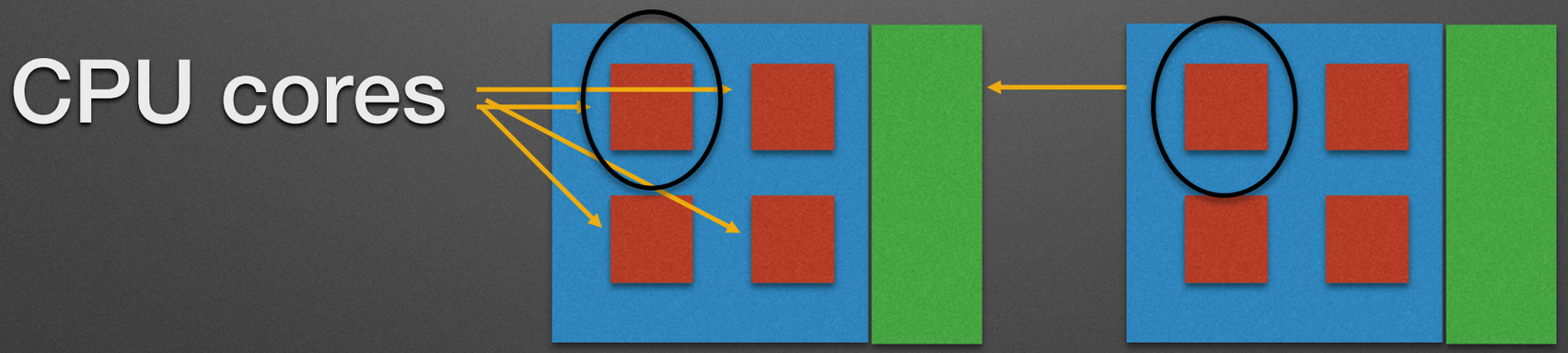
Submitting Jobs

Job Submission: You need to know…

- Number of CPUs/type of parallel environment
- Time you job will take
- Memory you would like to reserve
- Job specific information (module required, job specific commands)

Number of CPUs: Parallelization

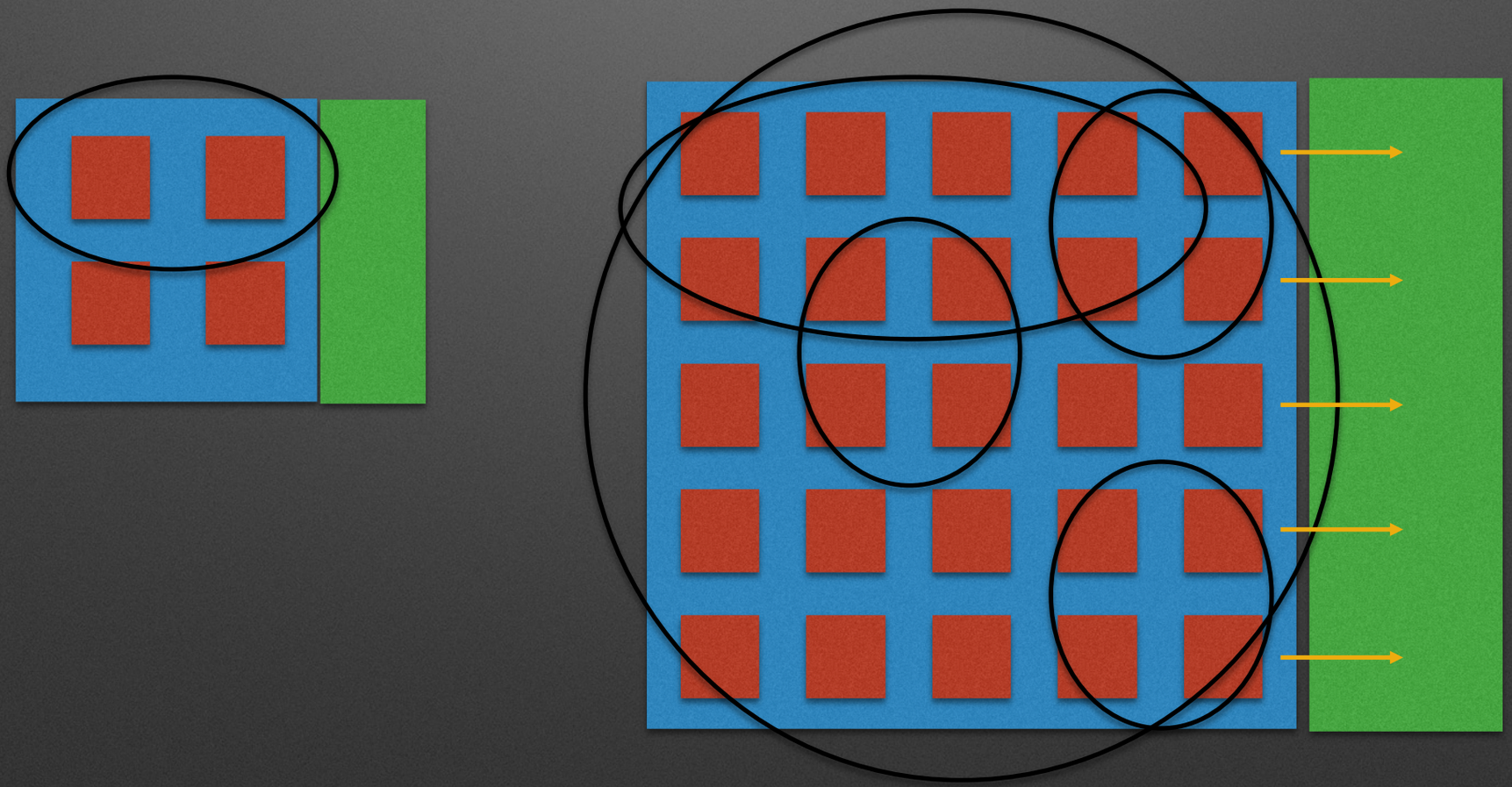
- Naive parallelization (embarrassingly parallel)- one large job easily split into separate jobs.



- E.g. many sequential blast searches at once, splitting bootstrap reps.

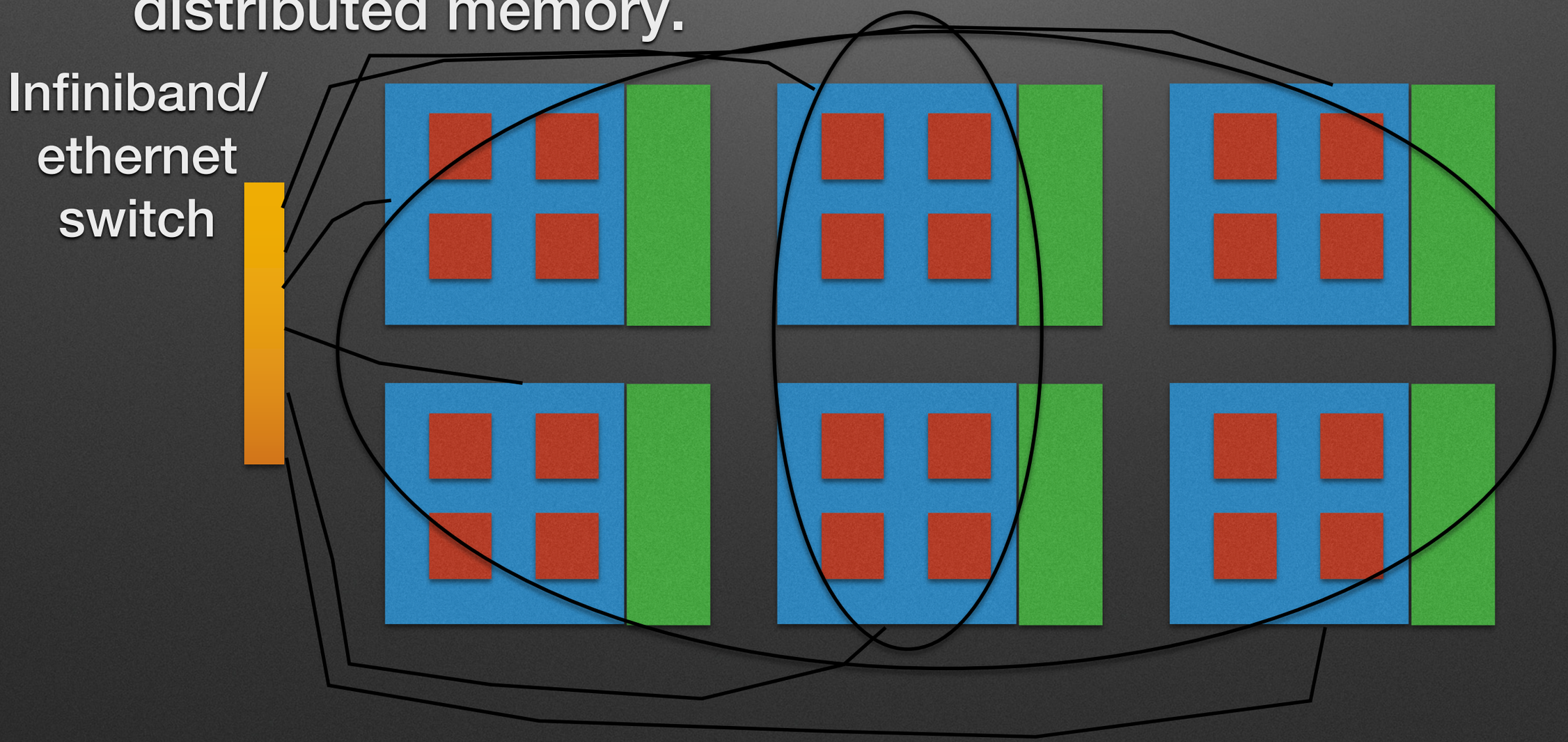
Number of CPUs: Parallelization

- Multi-threading (pthreads, OpenMP)-multiple CPUs on a single node. This method uses shared memory.



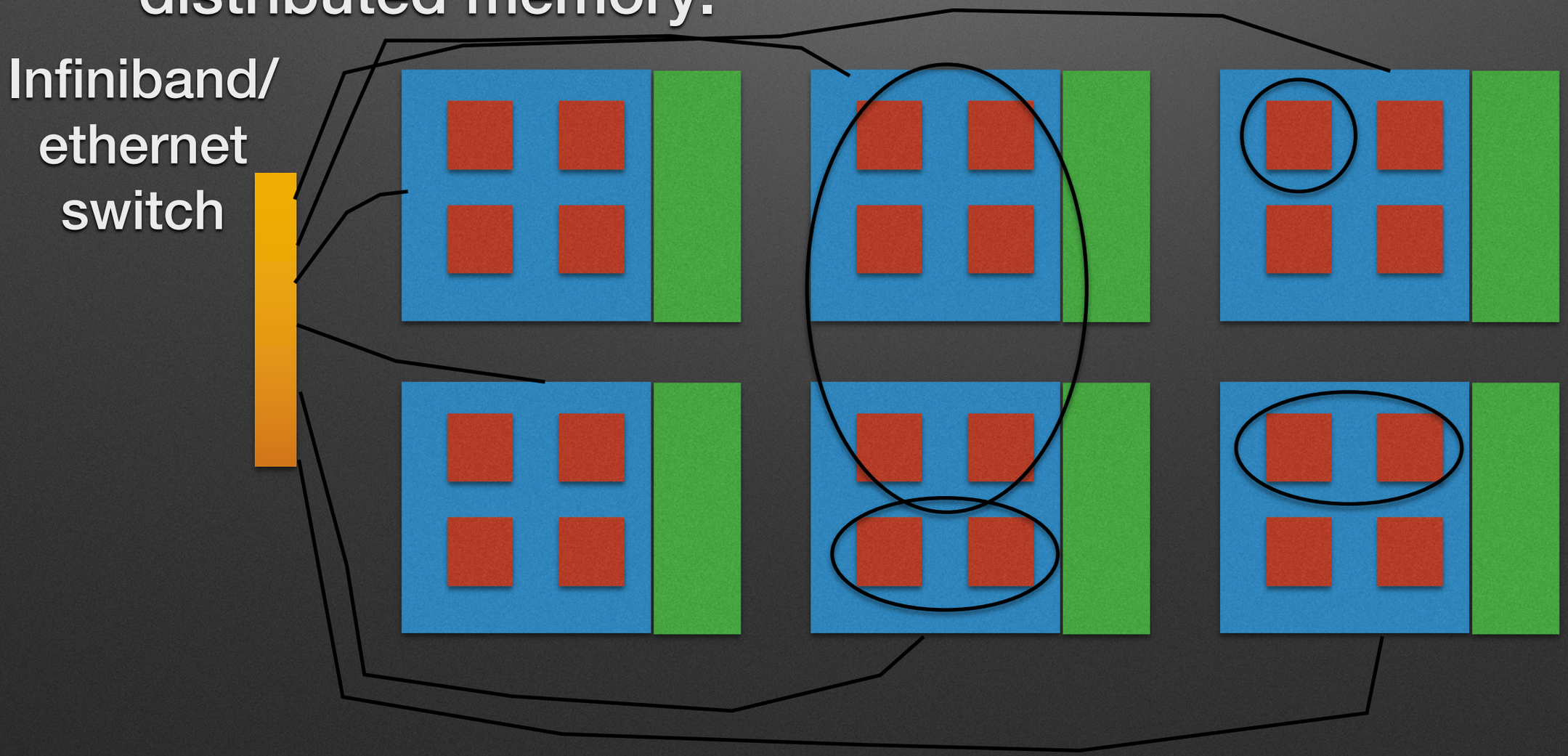
Number of CPUs: Parallelization

- MPI (Message Passing Interface)-multiple CPUs across multiple nodes. Messages are passed between nodes via the interconnect. This method uses distributed memory.



Number of CPUs: Parallelization

- MPI (Message Passing Interface)-multiple CPUs across multiple nodes. Messages are passed between nodes via the interconnect. This method uses distributed memory.



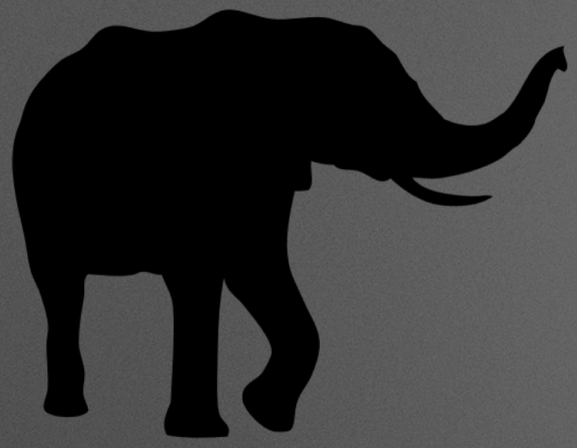
Number of CPUs: Parallelization

- How do I know which one to use?

- *This is determined by the type of program you wish to run. Check the program documentation or check with us.*

Time

- The cluster needs to be told estimates of run time
- Short jobs: 7 hours
- Medium: 3 days
- Long: 30 days
- Unlimited (there are very few of these…anything submitted to this queue should have some checkpointing capability)



Memory use

“Memory is both an expensive and scarce resource.”
-- Sylvain Korzennik

- Important to give a good estimate!
- How to estimate?
  - Check with others, program documentation
  - Monitor test runs, try smaller run and check how memory use scales up

`qsub` flags

- How you tell the scheduler about your job
- `-cwd` runs job from current working directory
- `-j y` joins the stderr to the stdout
- `-o` sampleOut.log if you have chosen to join stderr and stdout, your job will write these to the sampleOut.log file
- `-N` The name of your job. Use something informative!
- `-m abe, -M example@si.edu` email with job status

`qsub` flags

- Parallel enviromnent (`-pe`)
  - MPI (CPUs spread across nodes) `-pe orte 10` or `-pe mpich 10`
  - Threads (multiple CPUs, one node) `-pe mthread 16`
- Run Time `-q <q-name>` or `-l s_cpu <time>`

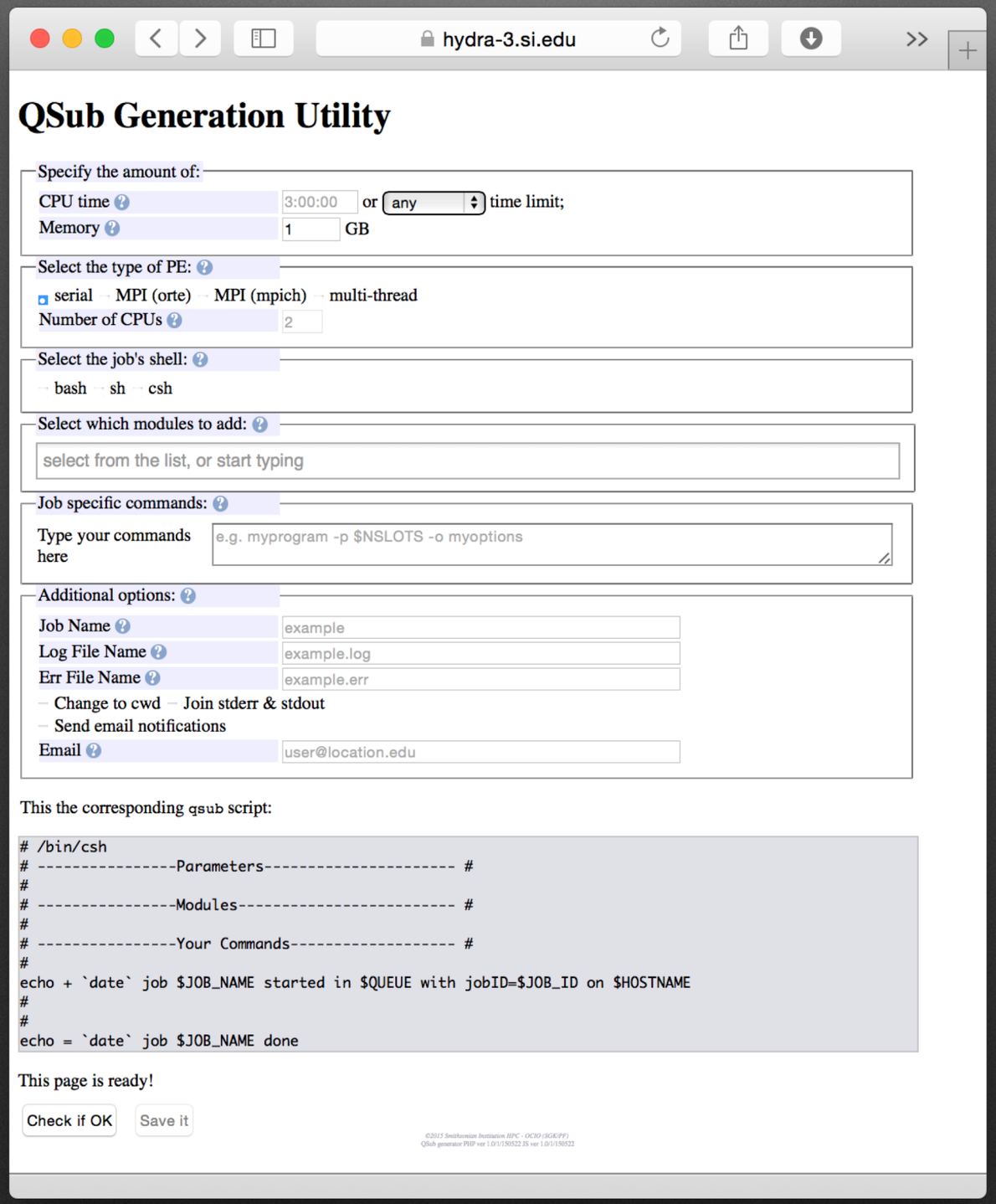
qsub flags

- Memory use specification:
  - `-l memres=4G,h_data=4G,h_vmem=4G`
    - `memres`: “Memory Reserved” This is for the cluster to track memory use and prevent other from grabbing memory you will need. Corollary, it prevents other from using that memory. *This is a shared resource*.
    - `h_data,h_vmem`: Tells the system to terminate your job if it uses too much RAM
    - For large memory jobs, you must indicate that you wish to use the high memory queue by adding `-l himem` this can be added to the other variable using the `-l` flag.


MEMORY IS PER CPU NOT PER JOB

- 16 threads with 4GB RAM each means 64GB total

Putting it all together



Best use practices

- Ramp up your use slowly to monitor the resources used by your jobs (CPU, memory, disk space)
- Run all analyses through `qsub`, i.e. NO analyses other than simple scripting/moving around should be done on the login nodes.
- Upload your files directly to the /pool drives, not your home directory.
- Check available disk space before you upload large files.
  - do this with:
    `$ df -h /pool/genomics` (`df`=disk free)
- Cleanup disk use following analysis
  - Remove files after transferring them back
  - Avoid leaving lots of small files (use tar!!)
- Understand the usage requirements of the programs that you are running and use reasonable `-l` flags for memory and compute time.