Introduction to Bioinformatics

Topic 9
Cluster Computing

Lecturers
Gregory R. Grant

Fall, 2023

Teaching Assistants

Gregory R. Grant

Genetics Department

ggrant@pennmedicine.upenn.edu

ITMAT Bioinformatics Laboratory

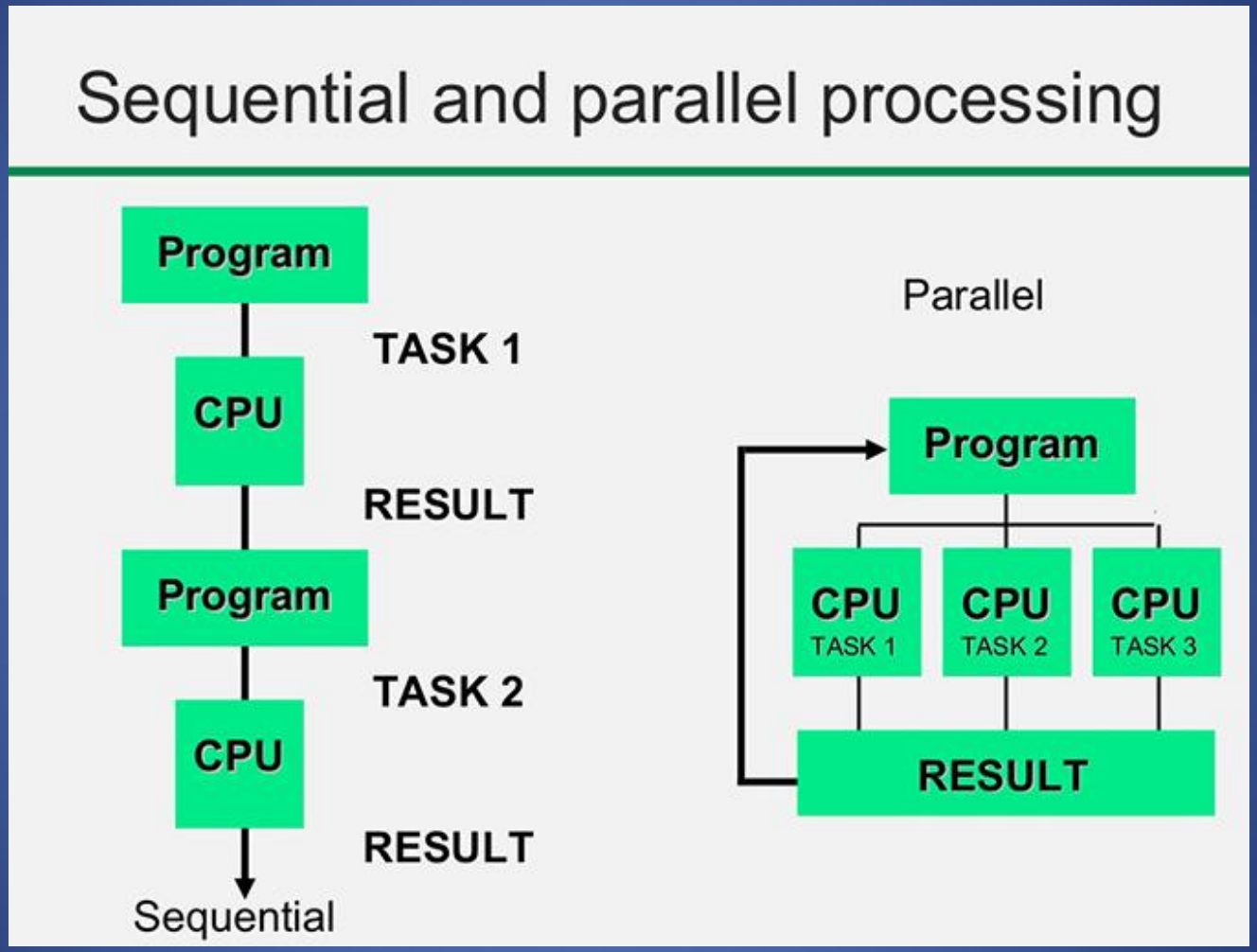
University of Pennsylvania

Cluster Computing

A compute cluster is what you can imagine. A room full of computers.

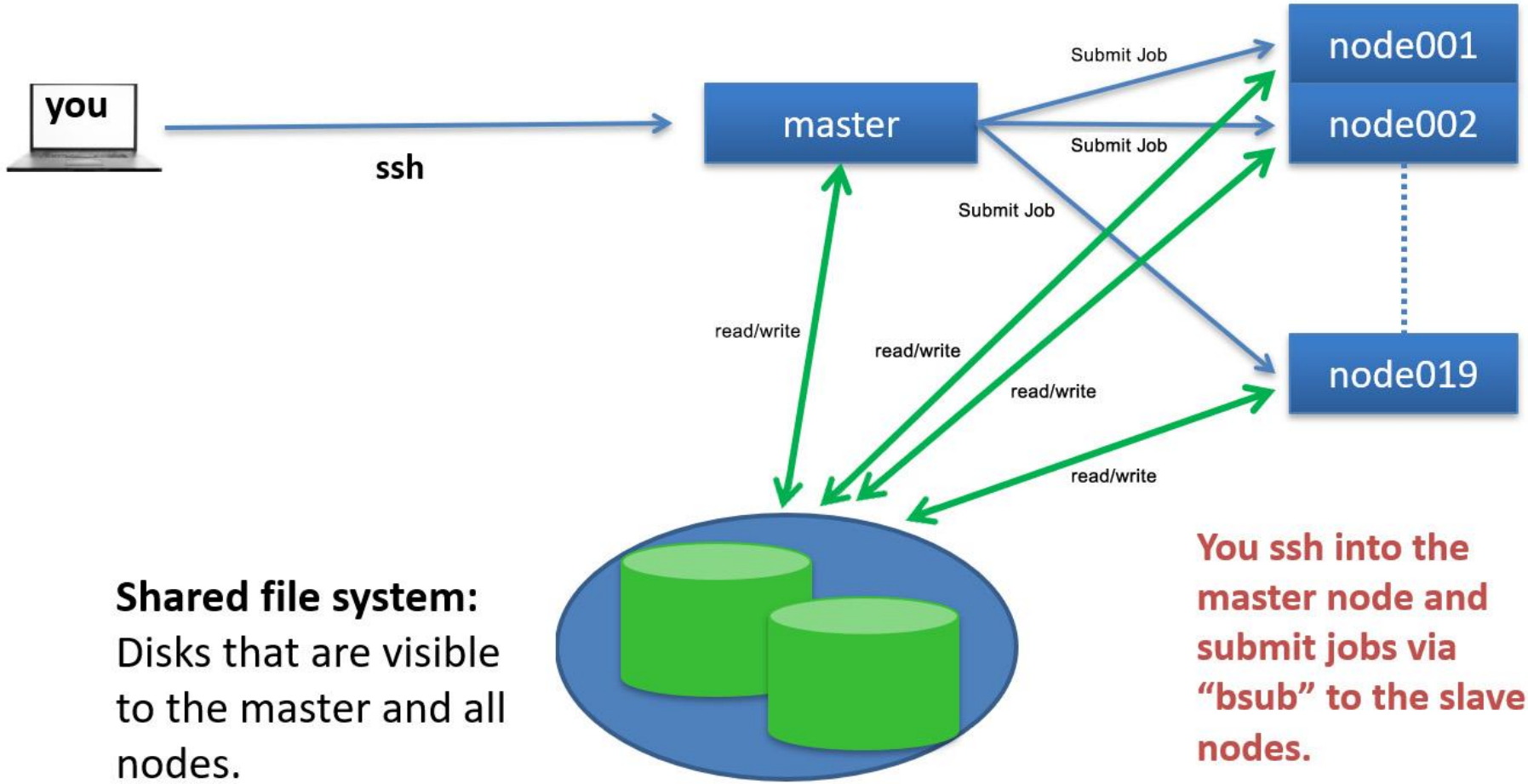
But no monitors or keyboards.

Sequential vs. Parallel



- Of course, this only works if Task 2 does not depend on the output of Task 1 and Task 3 does not depend on Task 1 or 2, etc.

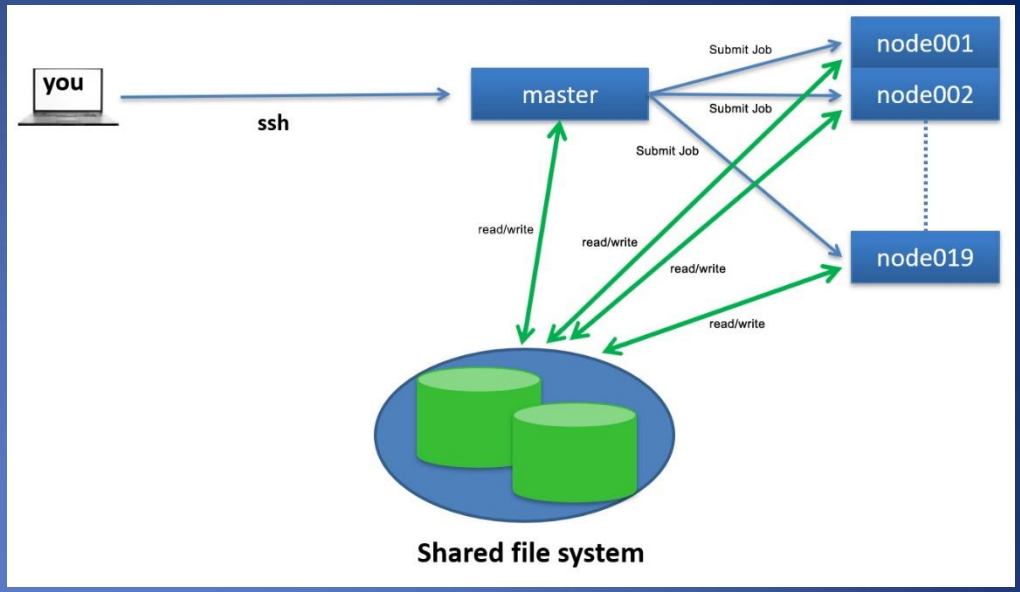
Cluster Compute Architecture



Shared file system:
Disks that are visible to the master and all nodes.

You ssh into the master node and submit jobs via "bsub" to the slave nodes.

Cluster Computing



- One master machine
- Multiple slave machines
  - AKA nodes, CPU's, processors, computers, etc.
- And one shared file system visible to all nodes.
- ➢ Every node can be used for computation except the master node.
- ❑ You log in to the master node.
- ❑ You run jobs on the slave nodes.
- There are special Unix command enabled on the master node to submit jobs to the slave nodes.
  - As well as commands to monitor jobs, etc.
  - These commands are called the "architecture" of the cluster.
  - There are two main architectures you are likely to encounter: SGE (SunGridEngine) and LSF (Load Sharing Facility).

Shared and Unshared Memory

- Disk space (the file system) is universal and shared by all nodes.
- But each node has its own Random Access Memory (RAM) which is internal to the node and is not shared across nodes.
  - RAM is fast, but transient, memory. It is forgotten once the job is completed.
  - Multiple CPUs or Cores within a node may share a nodes RAM, but it is not shared across nodes.
- When you submit a job, you typically have to tell it how much RAM you will need, so the scheduler can make sure it's available on that node when your job is executed.

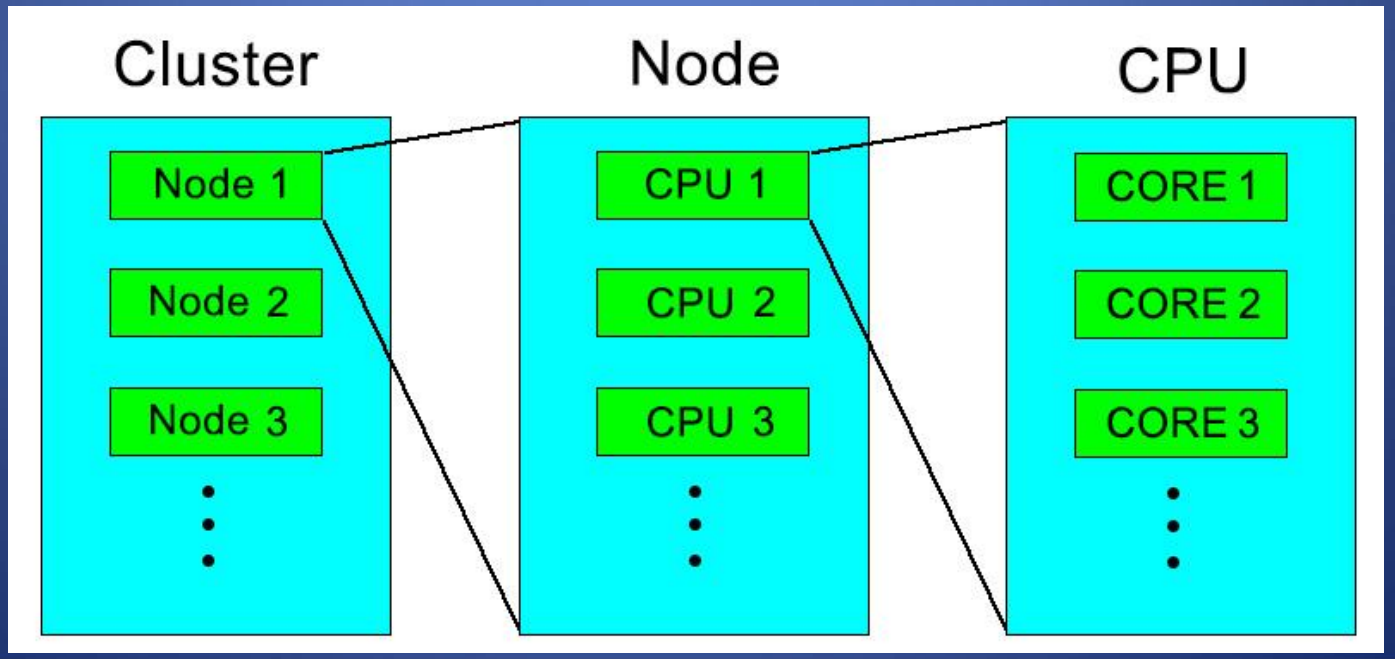
Where to do cluster computing?

- If your institution does not have a cluster, you can check one out by the hour from Amazon.
  - UPenn has a powerful cluster called DART (used to be PMACS), running the LSF architecture.
  - But we will not be using DART.

- You can also check out a cluster from Amazon.
  - Amazon offers many options; you can even make one that works identically to DART.
  - We have set up an Amazon cluster for this class, that you can access through the same interface we've been using for Unix.

Terminology

- A **node** is basically a computer, without a monitor or keyboard.
  - A node consists of multiple **CPU**s and a CPU consists of multiple **Cores**.
- A **core** is a serial processing unit that can only one job at a time.



Parallelization

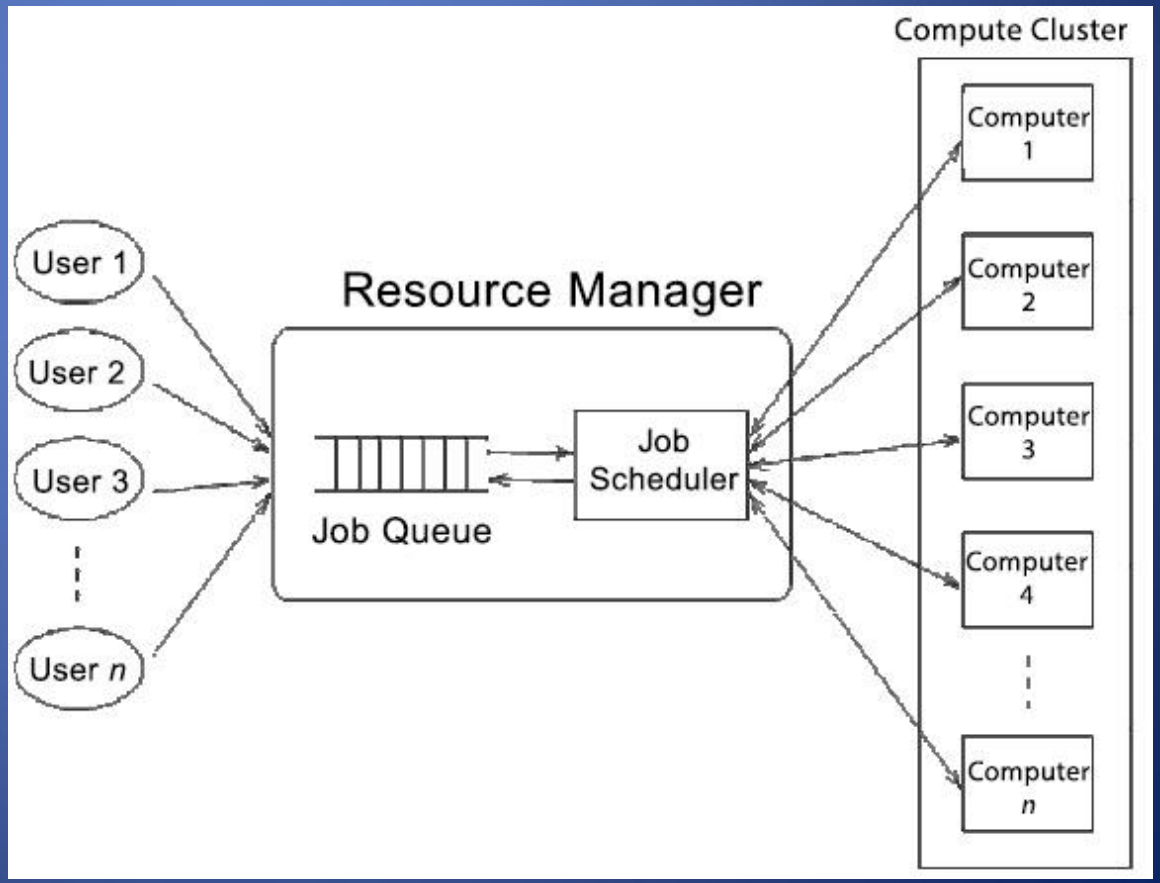
- When working with big data, we get jobs done by distributing them across nodes on a compute cluster.
  - A job that takes 1,000 hours of serial compute can be done in one hour if distributed to 1,000 nodes.
- But this only works if the job can be parallelized.
  - Not all jobs can.
- If you have a pipeline, for example, where each step depends on the previous, then you cannot run the steps in parallel.
  - Each job must wait for the previous one to finish.

Big Data in Biology

- Fortunately, analysis of big data in biology tends to be amenable to parallelizable.
- For example, alignment of high-throughput sequencing data.
  - Aligners do not use information from the alignment of one read to align any other.
- If you have one billion reads, you could in theory distribute that to one billion nodes.
  - But no cluster has a billion nodes.
  - More likely you'd distribute 100 million reads to 10 nodes, or 10 million reads to 100 nodes, depending on how fast you need it and how many nodes you have available.

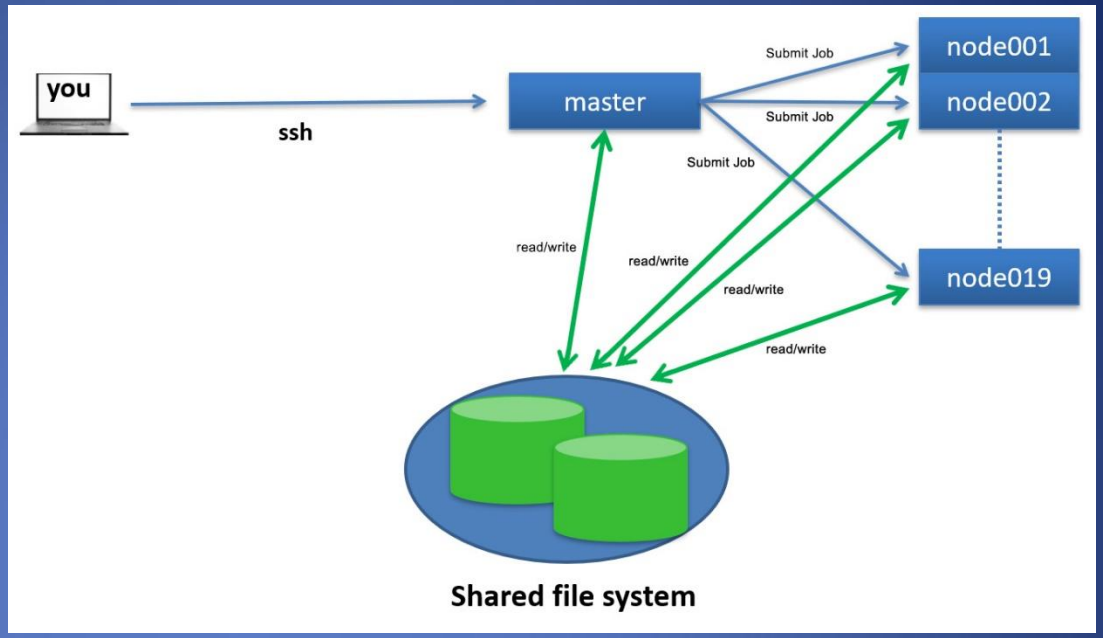
Queues and Schedulers

- Clusters are shared, usually by many users.
  - Unless you're privileged enough to have a cluster all to yourself.
- The number of nodes available at any given time depends not only on the size of the cluster, but on the current demand being put on it by users.
- Thus, you do not submit jobs directly to a node.
  - Instead, you submit jobs to "the queue".
  - Your jobs sit on the queue waiting their turn.
- Management of the queue is handled by software called a 'scheduler'.



File System Collisions

- A cluster has multiple nodes, but only one file system.



- Therefore, you can have multiple processors trying to write to the same file at the same time.
- This is not a problem if you're aware of it and handle it gracefully.

Levels of Parallelization

- Parallelization can happen at many levels.
- You may break a study up by samples.
  - E.g., align each sample in its own node.
    - All reads from one sample go to the same node.
- Or you may break one sample up by reads.
  - E.g., align chunks of 100,000 reads each in their own nodes.
    - 100,000 reads per node.
- A node can itself have multiple CPUs.
  - So really the scheduler assigns your jobs to individual CPU's not individual nodes.
  - One job may go to one CPU in one node while your next job may go to a CPU in a different node.

Levels of Parallelization

- You can only submit jobs to a single CPU.
  - Not to a single core within a CPU.
- But parallelization can happen even within the single CPU.
- The program running on a single CPU, may still parallelize the job to the multiple cores in the CPU it has been assigned.
  - That level of parallelization is not much of our concern. It's the concert of the programmer who wrote the app.
  - However, how many cores the app is allowed to use may be an option to the application. So, it's still important to be aware of it.

Parallelization Flow

- In general, no matter how you are parallelizing a job, two things need to be carefully considered.

1. How to divide the job into independent parts.
   - Each part must be able to complete without getting any information from the other parts.
2. How to merge the output of all the individual nodes into one final output for the whole job.

Parallelization Flow

- Often this is easy. Consider two examples:
  1. Break a study into its individual samples.
  2. Chop up a file of reads into chunks.
- Merging can also be easy.
  1. Each sample gives one column in a spreadsheet.
  2. Concatenate the alignments

- It can also be difficult, sometimes extremely difficult, to split and/or merge information.
  - Fortunately, bioinformatics tasks tend to be easily parallelizable.

BIOL4536 Cluster

- We have attached a cluster to our BIOL4536 Unix server.
- The command to send a job to the cluster is “batch”.
- Run it with no params to see the usage.

```
ggrant@workstation:~$ batch
usage: batch [-h] {submit-job,list-jobs,cancel-job,terminate-job,describe-jobs} ...

Using Batch, you can run batch computing workloads on the Amazon Web Services Cloud. Batch computing is a common means
for developers, scientists, and engineers to access large amounts of compute resources.

positional arguments:
  {submit-job,list-jobs,cancel-job,terminate-job,describe-jobs}
    submit-job          Submit a job.
    list-jobs           Returns a list of Batch jobs. Only the last 10 jobs are returned by default.
    cancel-job          Cancels a job. Jobs that are in the SUBMITTED, PENDING, or RUNNABLE state are canceled. Jobs
                        that have progressed to STARTING or RUNNING aren't canceled, but the API operation still
                        succeeds, even if no job is canceled. These jobs must be terminated with the terminate job
                        operation.
    terminate-job       Terminates a job. Jobs that are in the STARTING or RUNNING state are terminated, which causes
                        them to transition to FAILED. Jobs that have not progressed to the STARTING state are
                        cancelled.
    describe-jobs       Describes a list of Batch jobs.

options:
  -h, --help            show this help message and exit
ggrant@workstation:~$
```

submit-jobs

- The option 'submit-jobs' is used to submit a job to the scheduler.
- Run it with the –h option to see the usage of the submit-jobs option.

```
ggrant@workstation:~$ batch submit-job -h
usage: batch submit-job [-h] --job-name JOB_NAME --command COMMAND [--gpu GPU] [--vcpu VCPU] [--memory MEMORY]

options:
  -h, --help            show this help message and exit
  --job-name JOB_NAME   The name of the job. It can be up to 76 letters long. It can contain uppercase and lowercase
                        letters, numbers, hyphens (-), and underscores (_).
  --command COMMAND     The command to execute.
  --gpu GPU             The number of physical GPUs to reserve for the job. (default: 0)
  --vcpu VCPU           The number of vCPUs reserved for the job. (default: 1)
  --memory MEMORY       The memory hard limit (in MiB) present to the job. If your job attempts to exceed the memory
                        specified, the job is terminated. (default: 2048)
ggrant@workstation:~$
```

submit-jobs

- Here's an example.
- Once the job is submitted, run '`batch list-jobs`' to see the status. They'll go through several statuses before eventually arriving at SUCCEEDED or FAILED.
- You do not need to wait for one to finish before submitting another.
  - That's the whole point, they'll all run in parallel.

```
ggrant@workstation:~$ batch submit-job --job-name test1 --command 'water -gapopen 10 -gapextend .5 -outfile test_EPAM10.t
xt -datafile EPAM10 seq1.fa seq2.fa'
{
  "jobName": "test1",
  "jobId": "927668aa-2a82-4303-a203-accf3e43e2d0"
}
ggrant@workstation:~$ batch list-jobs
```

jobId	jobName	createdAt	stoppedAt	status
927668aa-2a82-4303-a203-accf3e43e2d0	test1	Mon Aug 28 15:29:24 2023	N/A	RUNNABLE
4f233ac9-3786-495e-aa81-ddd01d18104c	test1	Mon Aug 28 15:14:22 2023	Mon Aug 28 15:18:34 2023	SUCCEEDED

```
ggrant@workstation:~$
```