Introduction to High-Performance Computing (HPC)

http://tinyurl.com/HPC-bmi713

Radhika Khetani, PhD

Training Director @ Harvard Chan Bioinformatics Core (HBC)

Email: rkhetani@hsph.harvard.edu, Twitter: @rs_khetani

With contributions from Meeta Mistry and Kristina Holton

Computer components

CPU : Central Processing Unit

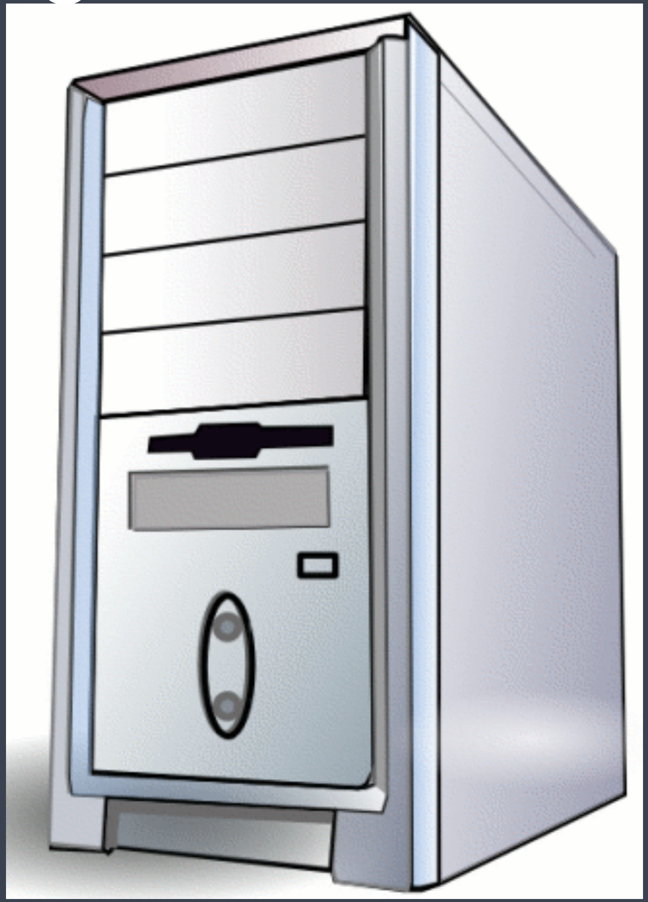
cores : individual processing units within a CPU

Storage : Disk drives

HDD : Hard Disk Drive

SSD : Solid State Drive

Memory : small amount of volatile or temporary information storage



Computer components (my Macbook Pro)

Model Name:	MacBook Pro
Number of Processors:	1
Total Number of Cores:	4
Memory:	16 GB
Data storage:	1 TB

“High Performance Computing most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business.”

http://insidehpc.com/hpc-basic-training/what-is-hpc/

Computer resources required for NGS Data Analysis

100s of cores for processing!

100s of Gigabytes or even Petabytes of storage!

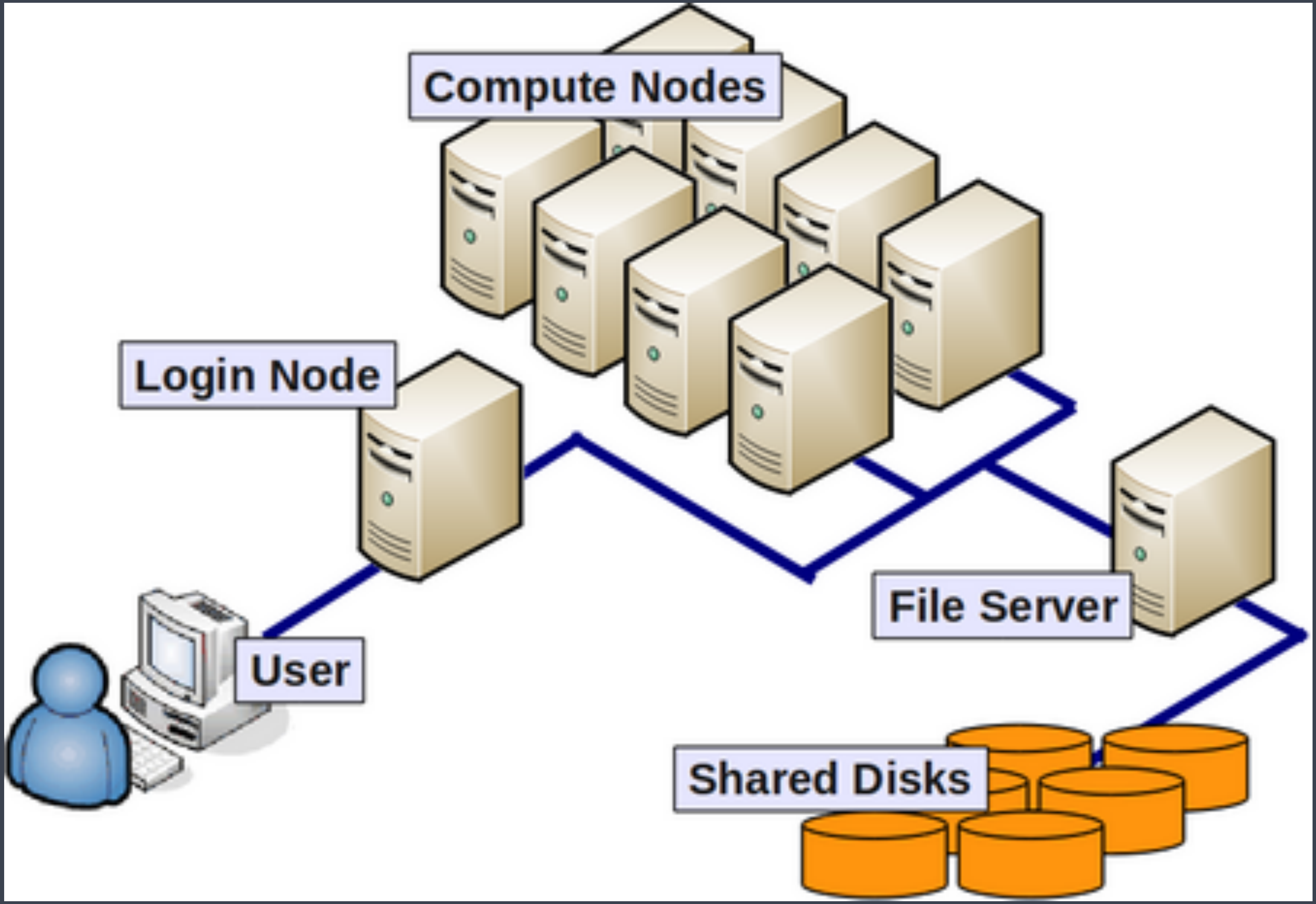
100s of Gigabytes of memory!

High-Performance Computing

Provides all the resources to run the desired Omics analysis in one place.

Provides software that is unavailable or unusable on your computer/local system

HPC cluster structure



HPC cluster components

Nodes: Individual computers in the cluster

Cores (threads): individual processing units available within each CPU of each Node

e.g. a “Node” with eight “quad”-core CPUs = 32 cores for that node.

Shared disk: storage that can be shared (and accessed) by all nodes

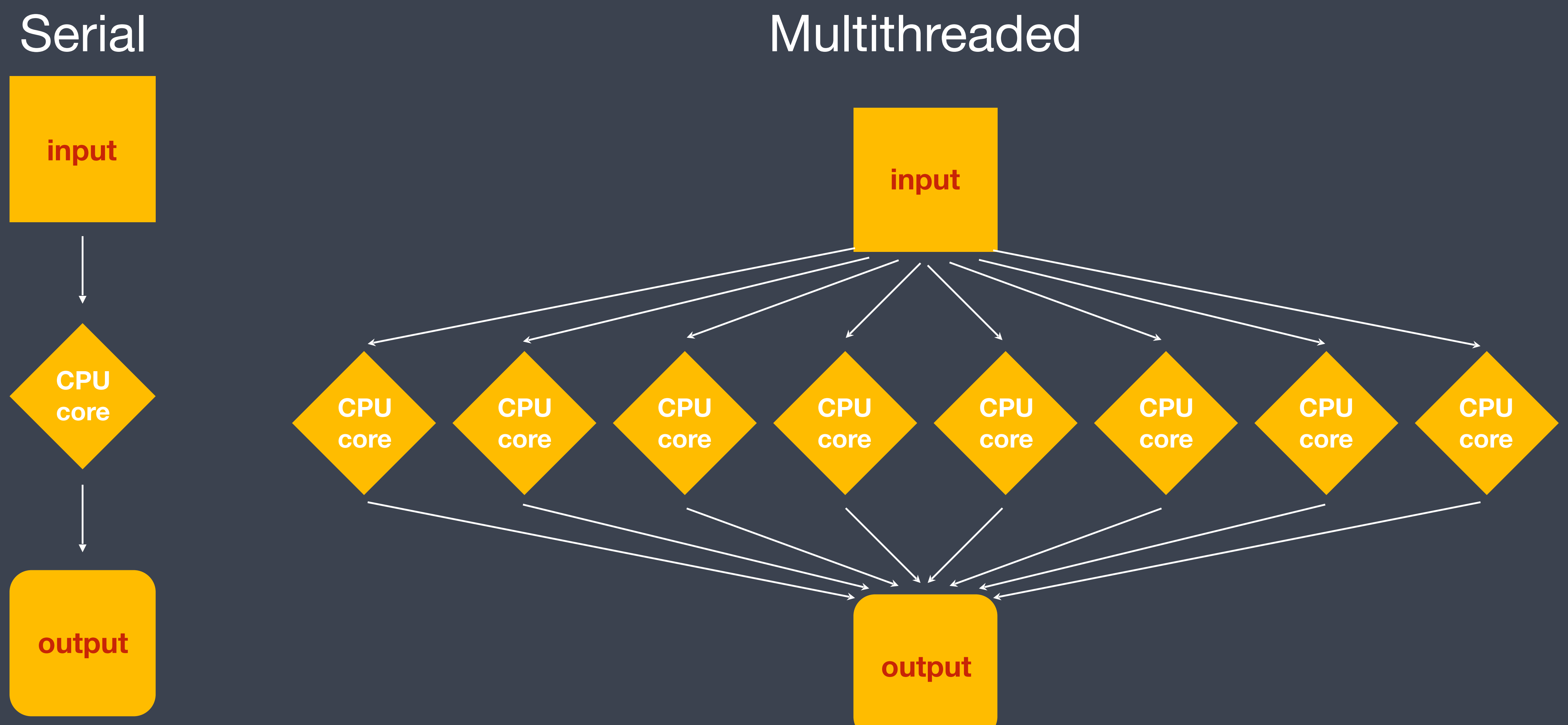
Parallel Computing

“Parallel computing is a form of computation in which many calculations are carried out simultaneously, operating on the principle that large problems can often be divided into smaller ones, which are then solved concurrently ("in parallel”)."

http://en.wikipedia.org/wiki/Parallel_computing

For 1 sample

High-Performance Computing

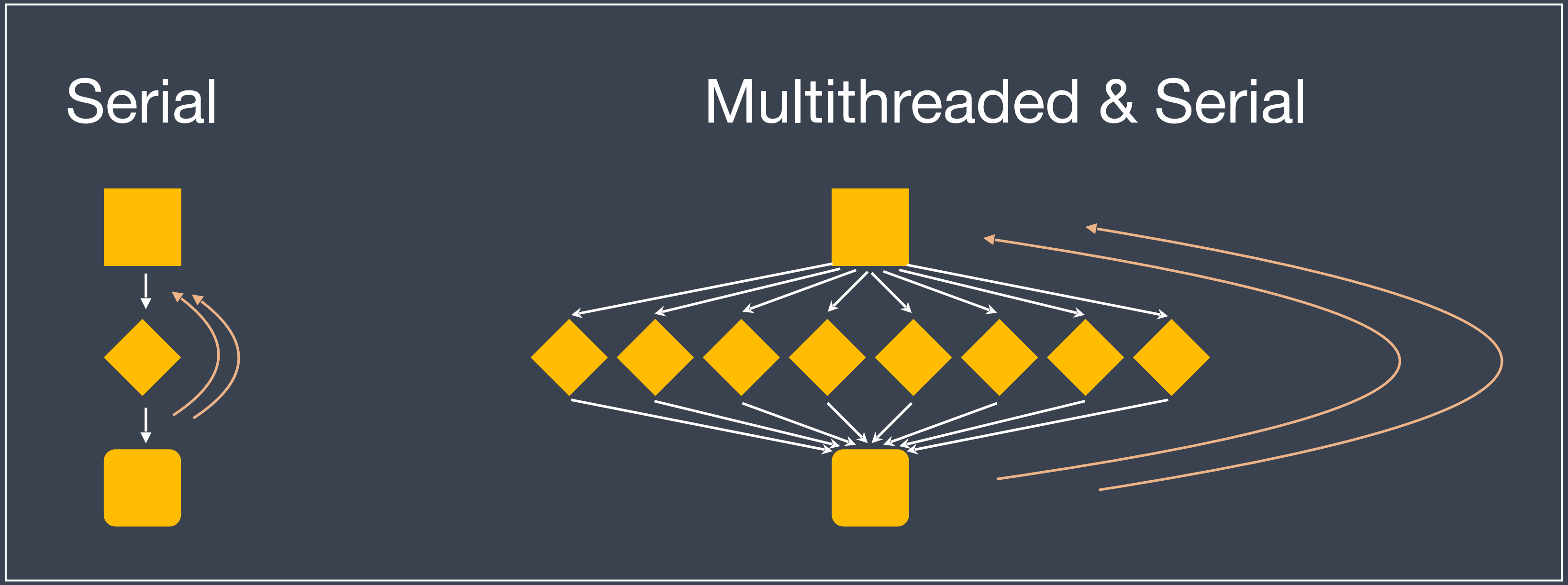


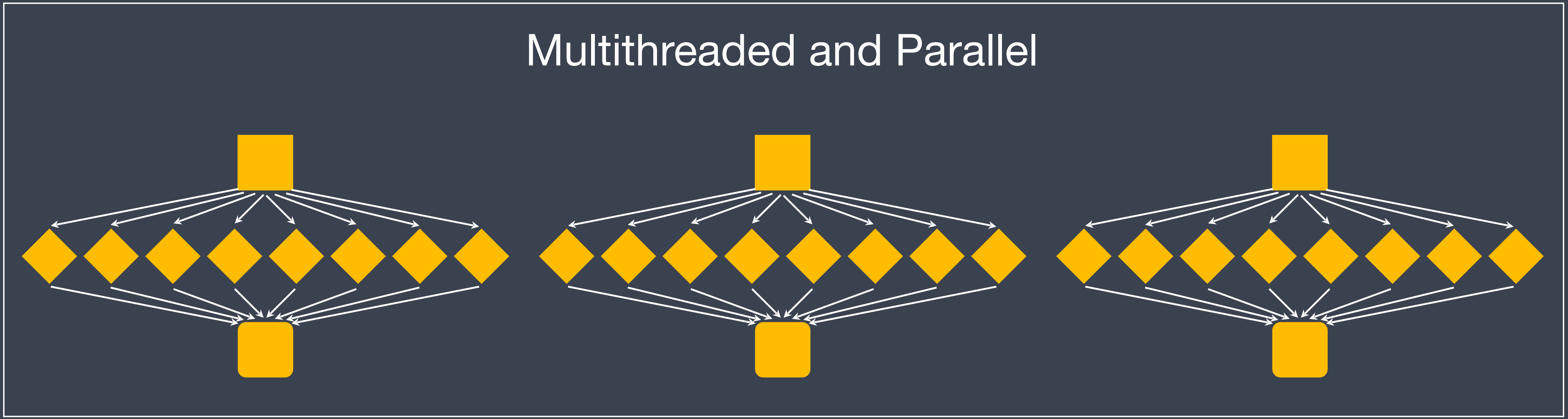
Faster and more efficient…

NGS data analysis is very amenable to this strategy

High-Performance Computing

For 3 samples



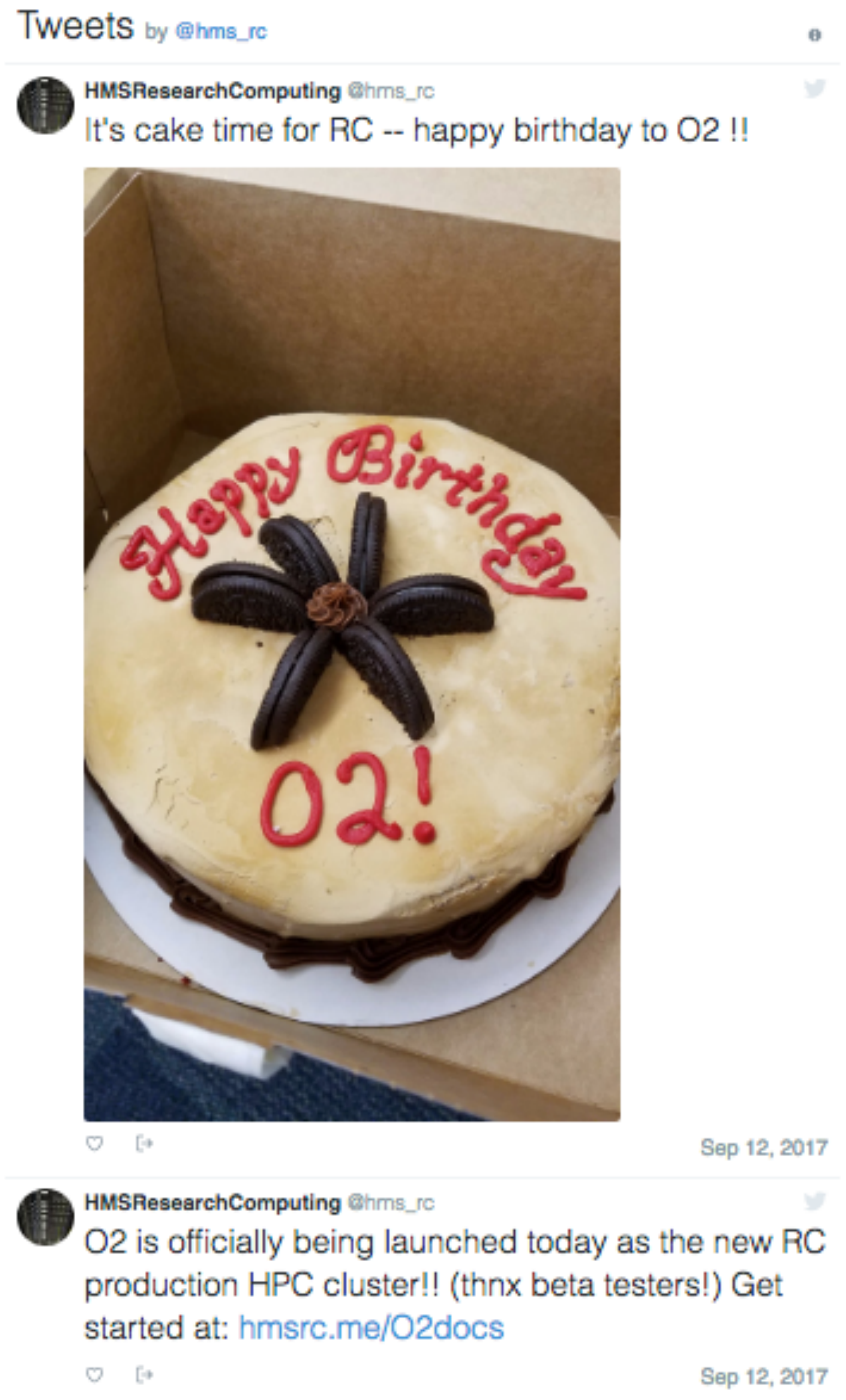


HPC Cluster

- multi-user, shared resource
- lots of nodes = lots of processing capacity + lots of memory
- a system like this requires constant maintenance and upkeep, and there is an associated cost

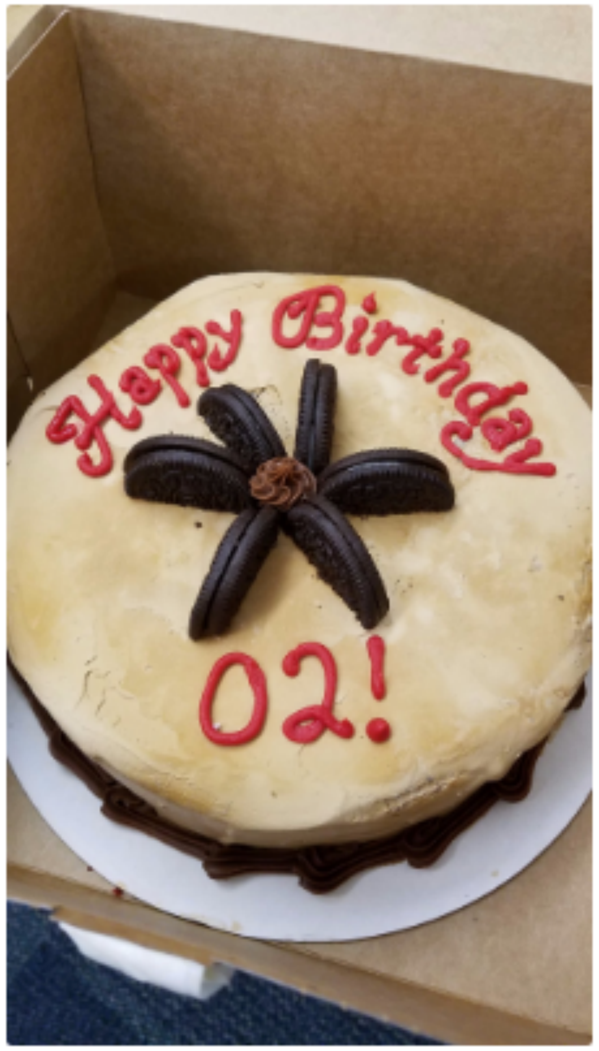
Wiki page: hmsrc.me/O2docs

Orchestra
HIGH PERFORMANCE COMPUTING CLUSTER



Tweets by @hms_rc

HMSResearchComputing @hms_rc
It's cake time for RC -- happy birthday to O2 !!



Sep 12, 2017

HMSResearchComputing @hms_rc
O2 is officially being launched today as the new RC production HPC cluster!! (thnx beta testers!) Get started at: hmsrc.me/O2docs

Sep 12, 2017

Introduction to High Performance Computing and O2 for New Users

HMS Research Computing

(*Slides courtesy of Kris Holton at HMS-RC*)

Welcome to O2!

- HMS Research Computing's second High-Performance Compute cluster to enhance the compute capacity available to HMS Researchers
- Homogeneous environment of newer, faster cores with high memory allocation to facilitate multi-core and parallelized workflows
- SLURM scheduler to efficiently dispatch jobs

O2 Tech Specs

- 8000 cores
- 32, 28, or 20 cores per node
- 256-160GB RAM (memory) per node (8-9GB/core)
- 756GB RAM high memory nodes
- 24 GPUs (K80, M40)
- CentOS 7 Linux
- SLURM job scheduler

Using O2!

1. Logging in to remote machines (securely)

- When logging in we used the “ssh” command,
  ssh stands for Secure SHell
- **ssh** is a protocol for data transfer that is secure, i.e the data is encrypted as it travels between your computer and the cluster (remote computer)
- Commonly used commands that use the **ssh** protocol for data transfer are, **scp** and **sftp**

Logging Into Orchestra 2

- Open a terminal and connect to O2 using the following command:

```
ssh YourECommons@o2.hms.harvard.edu
```

Welcome to O2!

- Where are you in O2?

```
mfk8@login01:~$
```

- You are logged into a “**shell login server**”, **`login01-05`**. These are not meant for heavy lifting!

```
mfk8@login01:~$ pwd
```

- You are in your home directory. This is symbolized by the “tilde ” (~). This is shorthand for: `/home/eCommons`

Interactive Sessions

- The login servers are not designed to handle intensive processes, and CPU usage is throttled.
- Start by entering your first job! This will (usually) log you into a "**compute** node!"

```
mfk8@login01:~$ srun --pty –p interactive –t 0-12:00
              ––mem 8G bash
```

"`srun ––pty`" is how interactives are started

"`-p interactive`" is the partition

"`-t 0-12:00`" is the time limit (12 hours)

"`--mem 8G`" is the memory requested

```
mfk8@compute-a:~$
```

2. Using & installing software

LMOD: Software Modules

- Most “software” on Orchestra2 is installed as an environment module.
- LMOD system adds directory paths of software into `$PATH` variable, to make sure the program runs without any issues.
- Allows for clean, easy loading, including most dependencies, and switching versions.

LMOD: Software Modules

Most software is compiled against gcc-6.2.0 — so load this first

```
$ module load gcc/6.2.0
$ module avail  #to see software now available
$ module spider  #verbose software currently available
```

Loading/Unloading Modules

- Loading modules

```
$ module load bowtie2/2.2.9
```

- Which module version is loaded (if at all)?

```
$ which bowtie2
```

- Need help with the module?

```
$ module help bowtie2/2.2.9
```

- Unloading modules

```
$ module unload bowtie2/2.2.9
```

- Dump all modules

```
$ module purge
```

3. The Job Scheduler, SLURM

Submitting Jobs

- In an "interactive session", programs can be called directly.

```
mfk8@compute-a:~$ bowtie -c 4 hg19 file1_1.fq
file1_2.fq
```

- What if you wanted to run the program and come back later to check on it?
  - a program is **submitted to O2 via a job (sbatch)**

```
mfk8@compute-a:~$ sbatch mybowtiejob.sh
```

Simple Linux Utility for Resource Management (SLURM)

- Fairly allocates access to resources (computer nodes) to users for some duration of time so they can perform work
- Provides a framework for starting, executing, and monitoring batch jobs
- Manages a queue of pending jobs; ensures that no single user or core monopolizes the cluster

Choosing the proper resources for your job with the appropriate `SBATCH` options

The “sbatch"

```
$ sbatch –p short –t 0-1:00 --wrap=“cp file.txt ..”
```

- Necessary to specify:
  - - p (partition)
  - - t 0-1:00 (time)
  - - - wrap (write it all in one line)

Lots of other options you can specify!

sbatch options

```
#SBATCH -p #partition
#SBATCH -t 0-01:00 #time days-hr:min
#SBATCH -n X #number of cores
#SBATCH -N 1 #confine cores to 1 node, default
#SBATCH --mem=XG #memory per job (all cores), GB
#SBATCH -J name_of_job (default = name of job script)
#SBATCH -o %j.out #out file
#SBATCH -e %j.err #error file
#SBATCH --mail-type=BEGIN/END/FAIL/ALL
#SBATCH —-mail-user=mfk8@med.harvard.edu
```

Partitions -p

Partition	Priority	Max Runtime	Max Cores	Limits
short	12	12 hours	20	
medium	6	5 days	20	
long	4	30 days	20	
interactive	14	12 hours	20	2 job limit
priority	14	30 days	20	2 job limit
mpi	12	5 days	640	20 core min
highmem	12	5 days	8	750G
gpu		120 GPU hours	20 cpu	
transfer		5 days	4	

Runtime: -t

- To specify how long you estimate your job will run for
- -t days-hours:minutes, e.g. 0-12:00
- -t hours:minutes:seconds, e.g. 12:00:00
- Aim for 125% over
- Subject to maximum per partition
- Excessive runlimits (like partition max) take longer to dispatch, and affect fairshare

Cores: -c

- -c X to designate cores: max 20
- -N X to constrain all cores to X nodes
- CPU time: wall time (-t) * (-c) cores used
- Unable to use cores not requested (no overefficient jobs): cgroups constraint
- Adding more cores does not mean jobs will scale linearly with time, and causes longer pend times

Memory: --mem

- Only 1G is allocated by default
- --mem XG #total memory over all cores
- --mem-per-cpu XG #total memory per CPU requested, use for MPI
- No unit request (G) defaults to Megabytes
  - 8G ~= 8000

Job submission scripts

Creating a job submission script

```
#! /bin/sh                    #Always at the top of the script

#SBATCH -p short
#SBATCH –n 4
#SBATCH --mem=8G
#SBATCH –o %j.out
#SBATCH –e %j.err
#SBATCH -J bowtie2_run1
#SBATCH --mail-type=ALL
#SBATCH –-mail-user=mfk8@med.harvard.edu

module load seq/bowtie2/2.2.9

bowtie -n 4 hg19 file1_1.fq file1_2.fq
```

Save as myJobScript.run

Run as `$ sbatch myJobScript.run`

**O2 will notify you when the job is done, or if there is an error

Job Priority

- Dynamically assigned
- Factors contributing: Age, Fairshare, Partition, QOS, Nice
- Fairshare: 0-1 scale

Managing jobs and getting information about submitted/running jobs

Job Monitoring

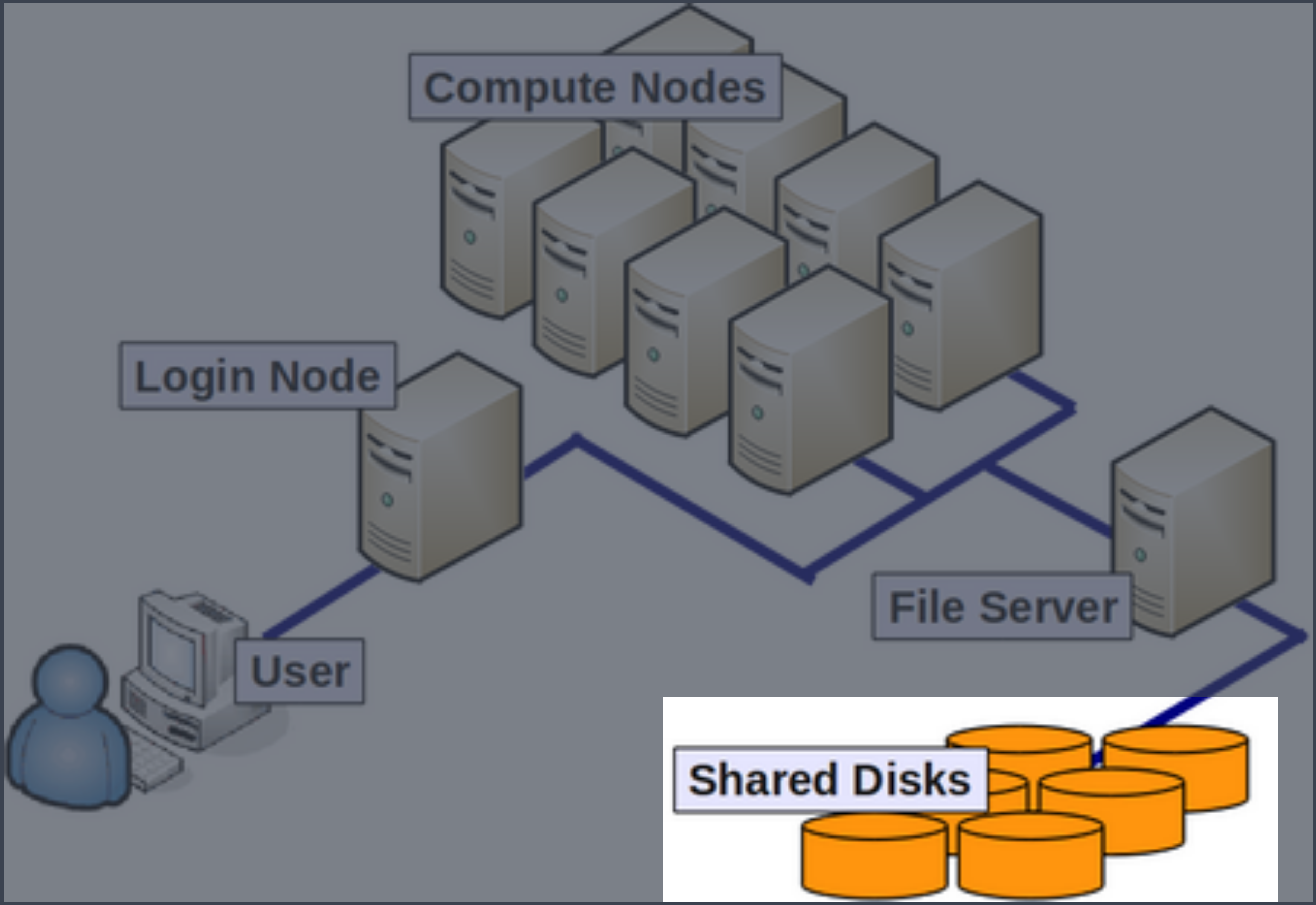
- `$ O2squeue #HMS wrapper`
- `$ squeue –u eCommons –t RUNNING/ PENDING`
- `$ squeue –u eCommons –p partition`
- `$ squeue –u eCommons --start`
- Detailed job info:
  `$ scontrol show jobid <jobid>`
- Completed job statistics:
  `$ O2sacct #HMS wrapper`

Cancelling/Pausing Jobs

- `$ scancel <jobid>`
- `$ scancel –t PENDING`
- `$ scancel --name JOBNAME`
- `$ scancel jobid_[indices]` #array indices
- `$ scontrol hold <jobid>` #pause pending jobs
- `$ scontrol release <jobid>` #resume
- `$ scontrol requeue <jobid>` #cancel and rerun

4. Filesystems and storage

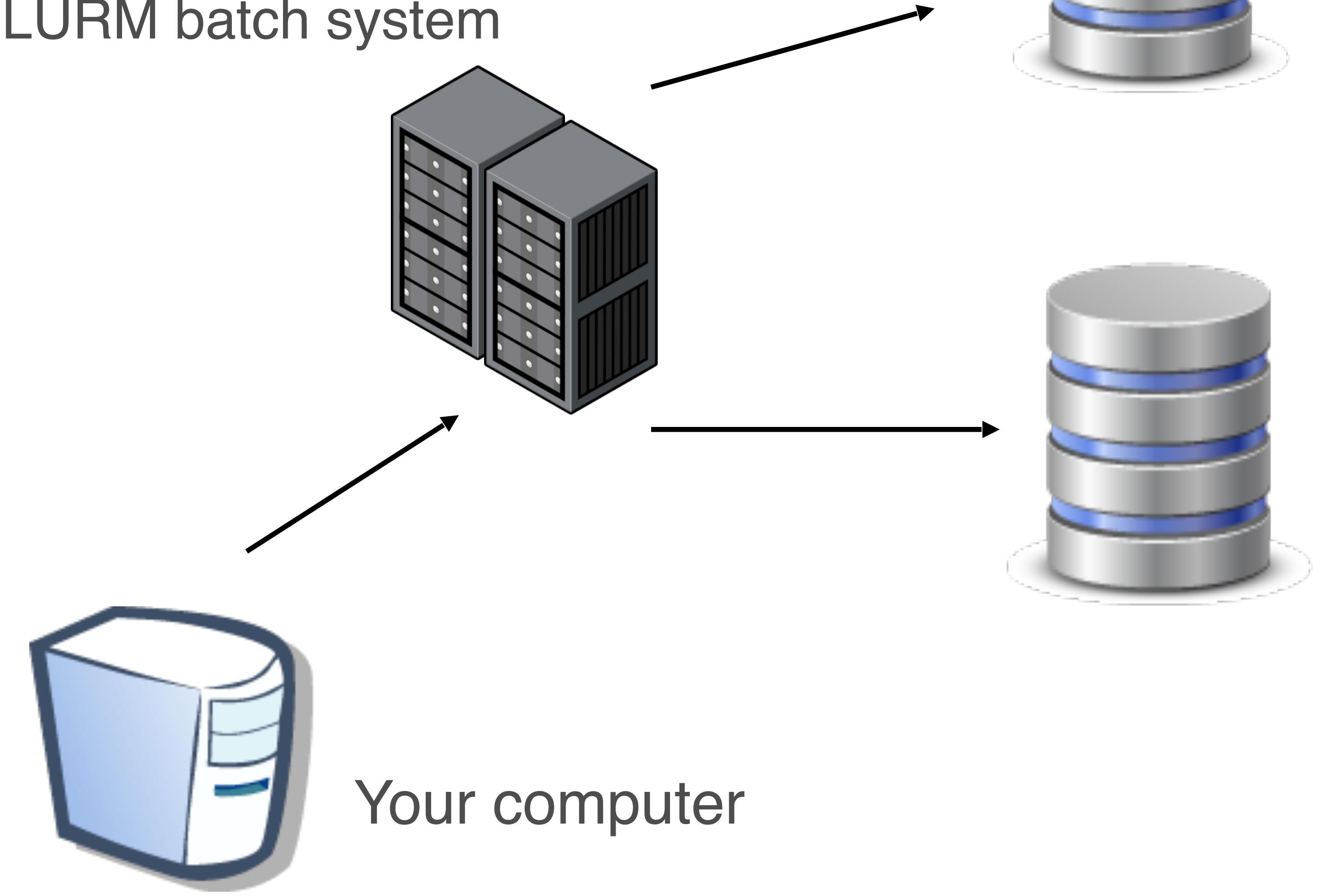
Filesystems and storage



Filesystems and storage

- Storage on HPC systems is organized differently than on your personal machine
- Physical disks are bundled together into a virtual volume; this volume may represent a single filesystem, or may be divided up, or partitioned, into multiple filesystems
- Filesystems are accessed over the internal network

O2 Primary Storage



O2 Cluster

- 5000+ cores
- SLURM batch system

/home

- `/home/user_id`
- quota: 100GB per user
- Backup: extra copy & snapshots:
- daily to 14 days, weekly up to 60 days

/n/data1, /n/data2, /n/groups

- `/n/data1/institution/dept/lab/your_dir`
- quota: expandable
- Backup: extra copy & snapshots:
- daily to 14 days, weekly up to 60 days

Temporary "Scratch" storage

Lustre
Scalable Storage

- `/n/scratch2`
- **For data only needed temporarily during analyses.**
- **Each account can use up to 10 TB and 1 million files/directories**
    - **Lustre** --> a high-performance parallel file system running on DDN Storage.
    - More than 1 PB of total shared disk space.
    - No backups! Files are automatically deleted after unaccessed for 30 days, to save space.
    - More info at: http://hmsrc.me/O2docs

Checking storage usage

- Isilon: `/n/data1, /n/data2,/n/groups`
  - `$ quota`
  - Breaks down per user, directory
- `/n/scratch2`
  - `$ lfs quota –h /n/scratch2`
  - 1 million files/folders, 10TB limit

For more direction

- http://hmsrc.me/O2docs
- http://rc.hms.harvard.edu
- RC Office Hours: Wed 1-3p Gordon Hall 500
- rchelp@hms.harvard.edu

These materials have been developed by members of the teaching team at the Harvard Chan Bioinformatics Core (HBC). These are open access materials distributed under the terms of the Creative Commons Attribution license (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.