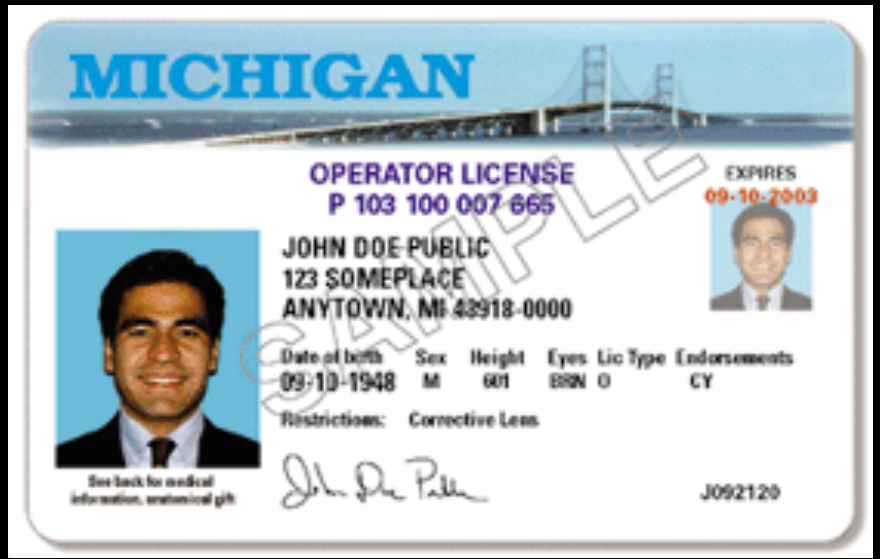
MODULE 03 – HIGH PERFORMANCE COMPUTING

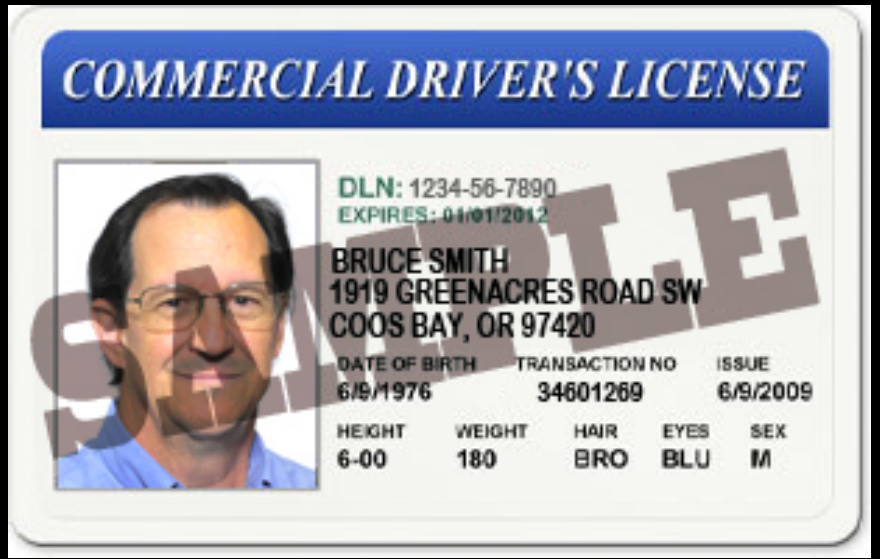
HIGH-PERFORMANCE COMPUTING CLUSTERS

HYUN MIN KANG @ UNIVERSITY OF MICHIGAN

Personal Computers vs. Clusters

… Require Different Trainings





Questions for Today

- What is HPC?
- When do we need to run analysis in HPC environment?
- What additional knowledge and experiences are needed for analysis of HPC?
- What about GPU, Cloud, Grid Computing, etc?

If a job is too slow to run on your laptop...

- **Plan A : Buy (or obtain access to) a server computer with higher computing power**
  - For example, server machines at scs.itd.umich.edu have 32 CPUs, 256GB of RAMs, with terabytes of disks

- **If the plan A is still too slow...**
  - Plan B : Supercomputer
  - Plan C : High Performance Compute Cluster
  - Plan D : Cloud Computing

Evolution of processing powers

- **FLOPS : Floating-point operations per second**

Year	Name	FLOPS	Cost	Notes
1946	ENIAC	5×10^2	$6.1M	Earliest general-purpose computer
1959	IBM 7090	1×10^5	$23M	Second-generation supercomputer
1982	IBM AT (286)	6×10^5	~$1,000	One of the first personal computers
1982	NEC V20	2×10^6	~$200	Home video game console
2007	iPhone	4×10^8	~$700	First successful smartphone
2016	iPhone 7	1×10^{10}	~$650	Latest Dual core smartphone
2016	Macbook Pro	5×10^{10}	~$2,000	15-inch Late 2016
2016	Titan	2×10^{16}	$97M	Initially built in 2005
2016	HP CL2100	3×10^{11}	$50K	Similar to what was used in Michigan's server

CPU vs. GPU programming

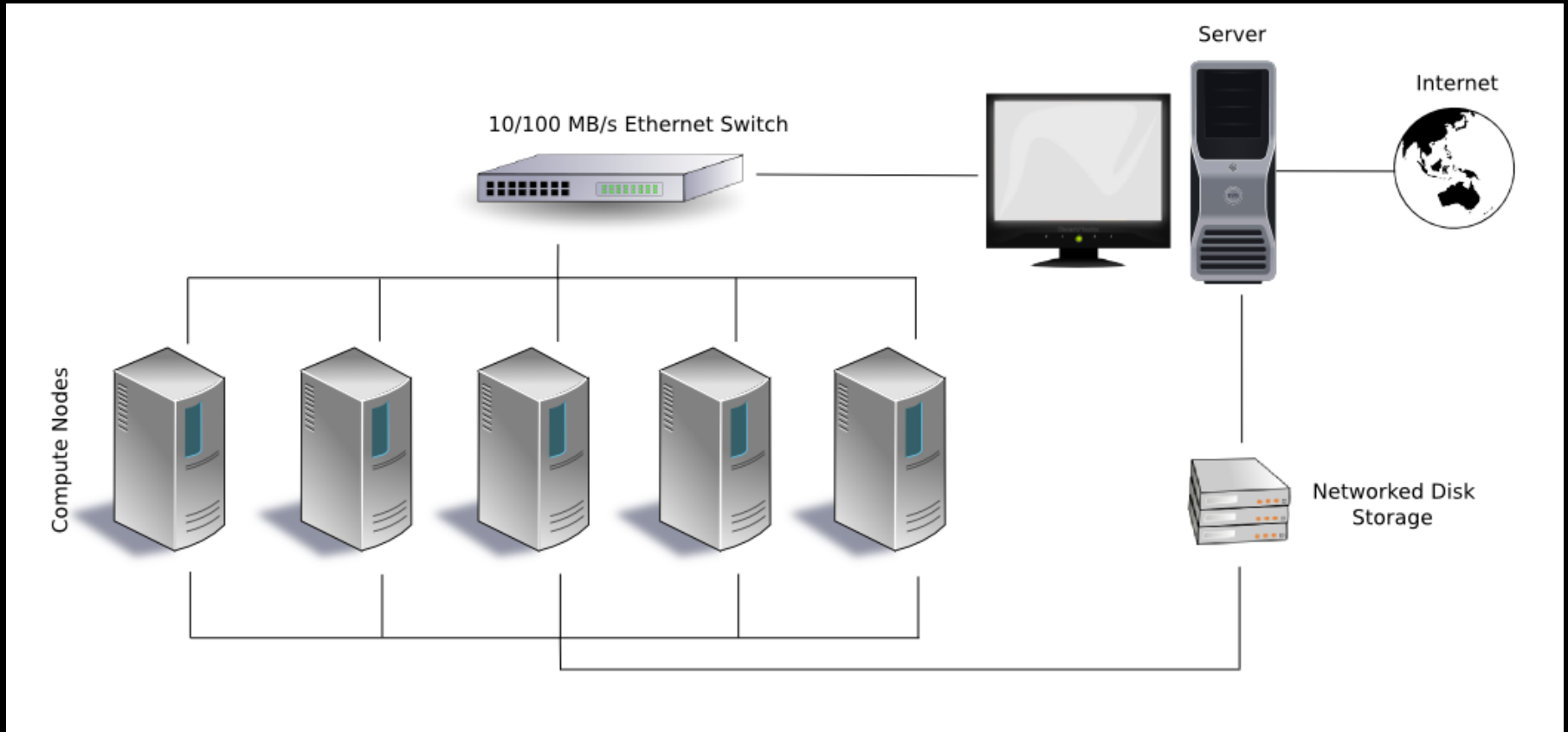
CPU

- Conventional
- Easier to program
- Compatible with widely used languages/libraries
- Parallelizable into tens of processes
- Better for memory-intensive,

GPU

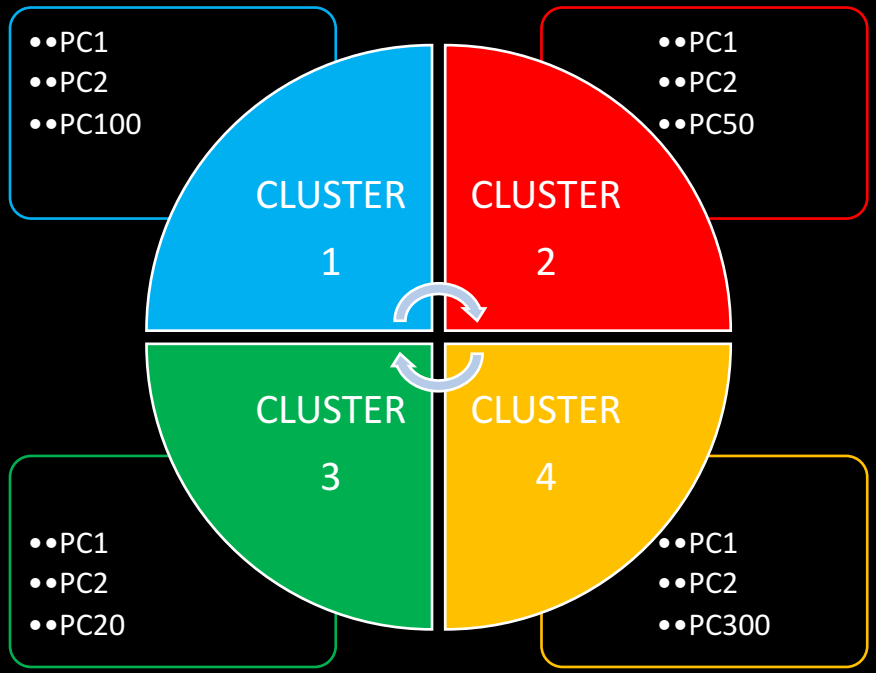
- Newer
- Higher FLOPS
- Requires specialized programming interface
- Massively parallelized into thousands of small processes
- Better for high-compute, low shared memory jobs

High-Performance Compute Clusters



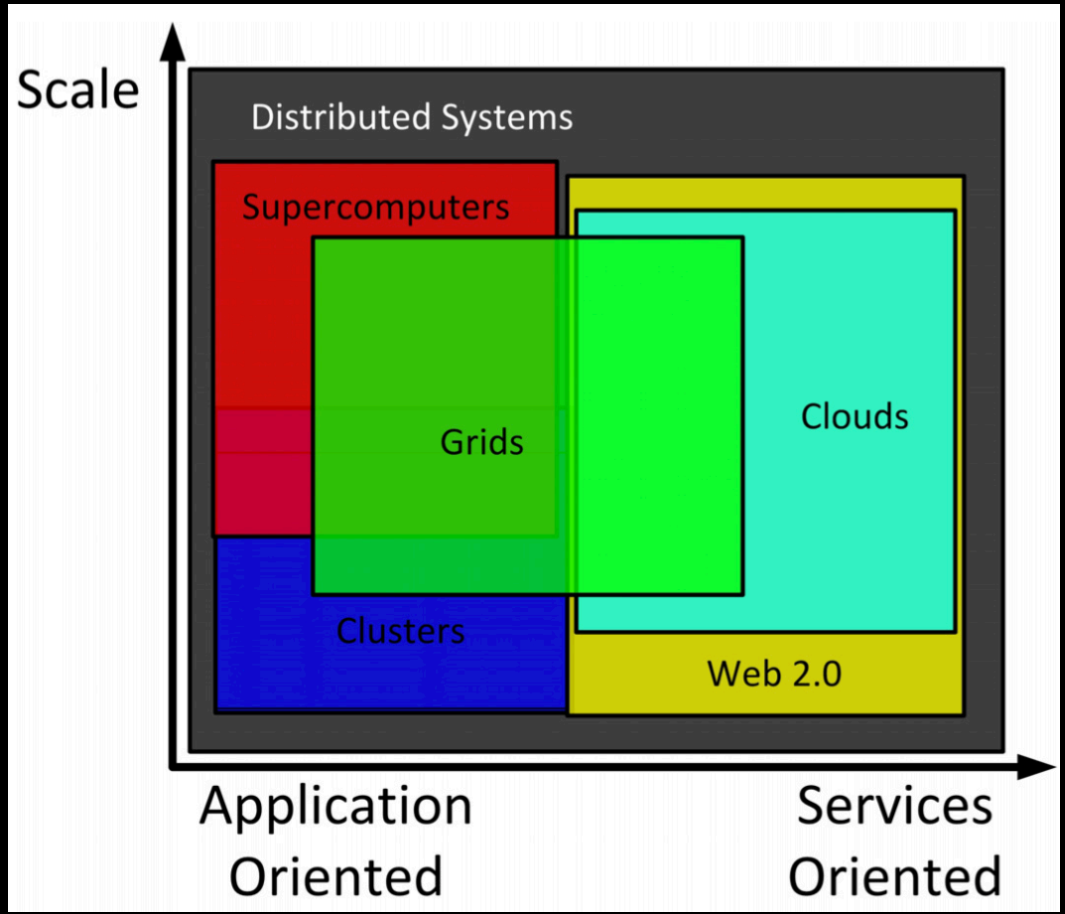
Grid Computing

- Combines multiple clusters into a single shared resources to achieve a common goal
- Multiple administrative domains are loosely connected to each other



Courtesy by Hasibur Rashid

Clusters, Grids, and Clouds

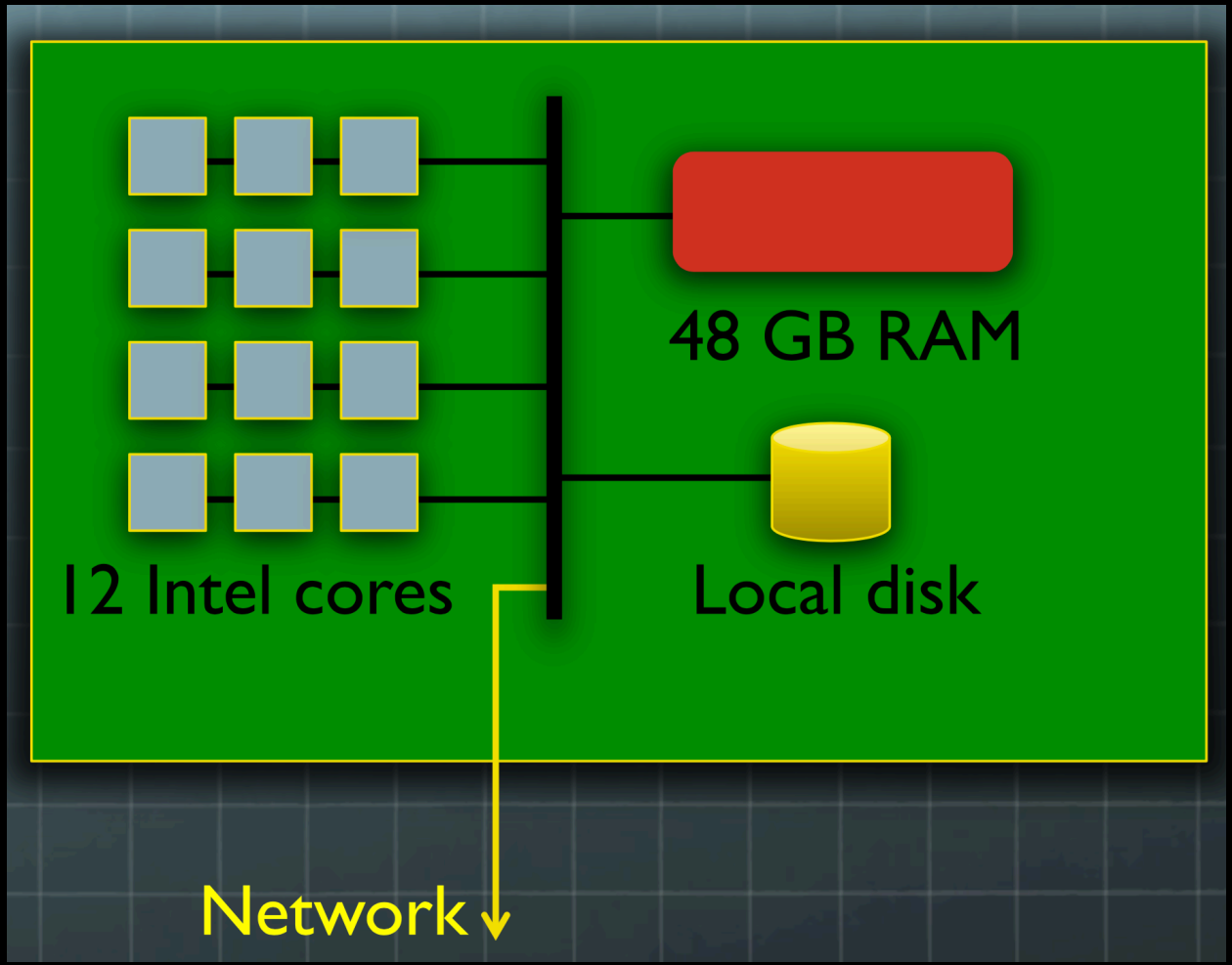


Foster et al. (2008)

Flux

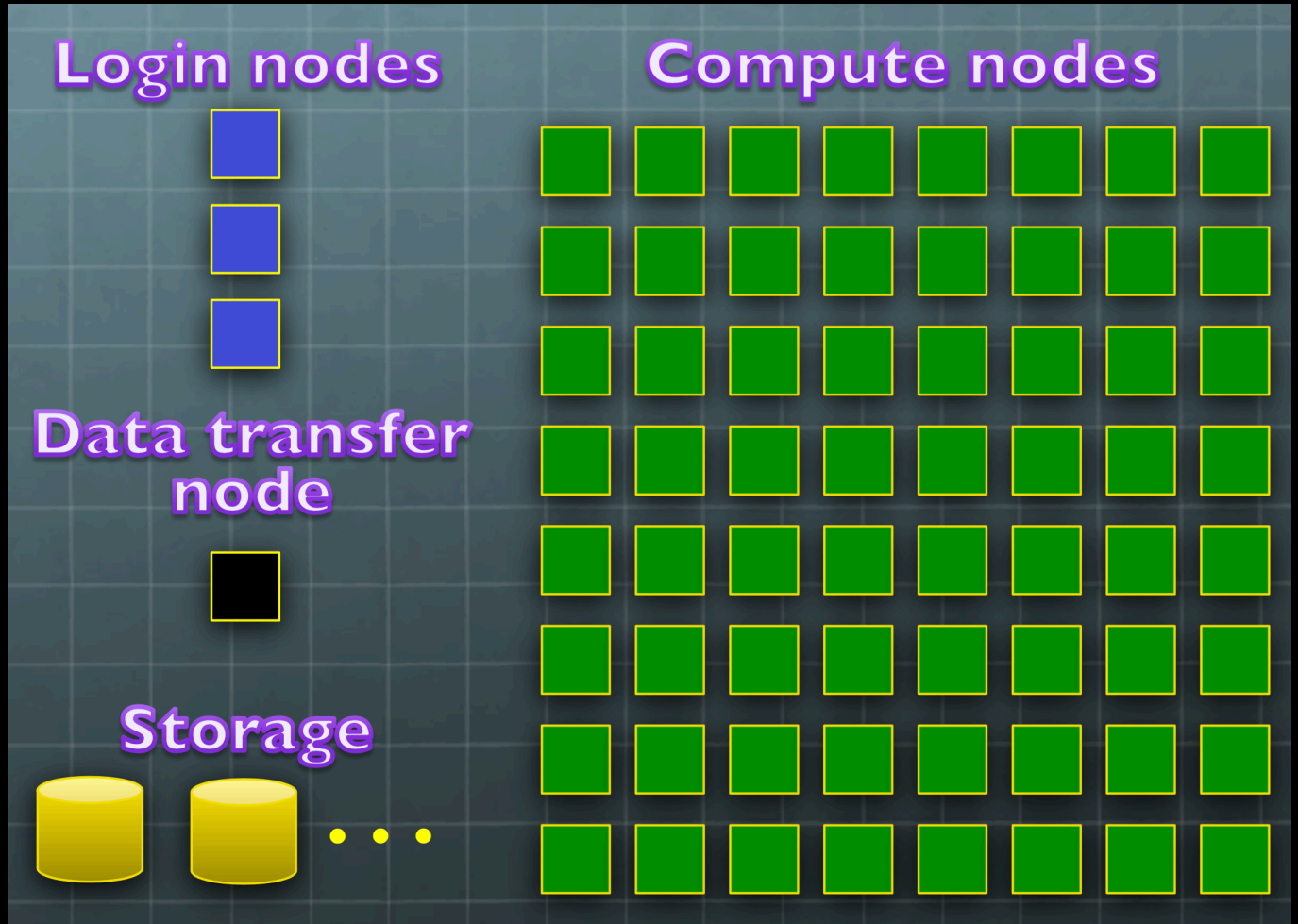
- **A linux-based HPC cluster at U Michgan**
  - 27,000 CPU cores & 1,372 compute node

A flux node



Ken Weiss

The flux cluster



Ken Weiss

To login the FLUX cluster…

- **You need to be on-campus network**
  - If you're off-campus, use VPN
    http://its.umich.edu/enterprise/wifi-networks/vpn
- **You need to enroll to Duo two-factor authorization**
  http://its.umich.edu/accounts-access/uniqnames-passwords/two-factor-authentication
- **Follow these steps:**
  1. Login to `[uniqname]@flux-login.engin.umich.edu` via SSH
  2. Type your Kerberos password
  3. Complete login through duo two-factor authorization

Running jobs on compute node

- **You need to write a PBS script to describe the jobs you want to run, and submit the script to queue your job**

- **You will need to describe..**
  - How many CPUs you will need
  - The upper-bound of memory you will need
  - The upper bound of wall-clock time of your job
  - The account which you are authorized to submit the jobs with
  - (Optional) The amount of node-specific disk space to use

- **The scheduler will execute the jobs considering all of these factors**

Why do we need to write PBS script?

- The scheduler need to optimize the resources to maximize the efficiency across all running jobs.

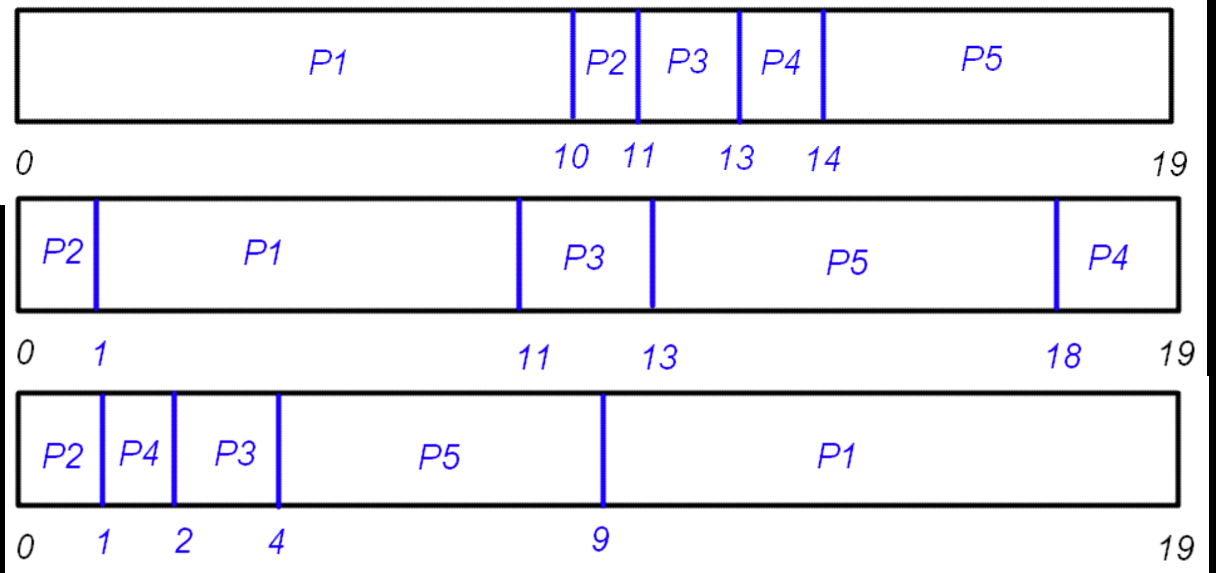
- For example, if many memory-intensive jobs are submitted into the same node, all jobs will be extremely slowed down (thrashing).

- Job scheduler uses the job description to find out the optimal schedule (using dynamic programming)

Example of job scheduling

	Burst Time	Priority
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	3

Given these processes, draw a Gantt Chart showing the execution.



Notes on PBS job descriptions

- **Flux allows allocations by core & months**

- **If you specify too little resources (e.g. small memory, small wall-clock time), your jobs will have risk of being aborted.**

- **If you specify too much resources, you may not be able to fully utilize the resources allocated to you**
  - If you specify >4GB of max memory, it will be considered as equivalent to use more than one cores
  - Specifying too long wall-clock time may reduce the priority of your jobs

- **Using /usr/time –v, you can figure out the peak memory and wall-clock time of your running jobs**

Three ways to run jobs on Flux

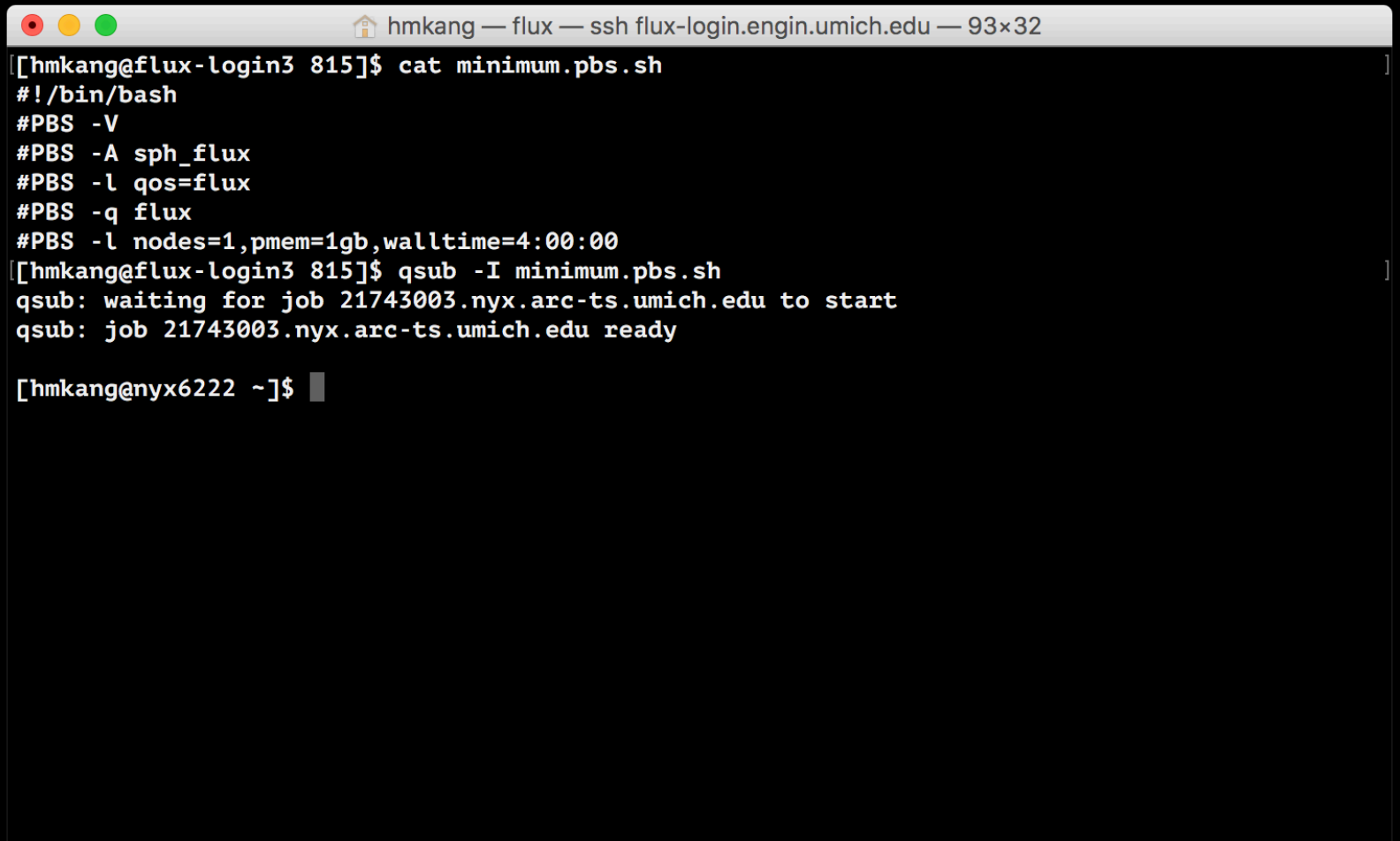
1. **Very short jobs – use the login node**
   - For light and simple commands (e.g. ls, cp, ..) use the head node as if you're using a regular UNIX server.
   - After a certain amount of time (e.g. ~30 minutes), your long-running jobs will be terminated
   - Running too many jobs on the login nodes may cause warnings.
2. **Long, interactive jobs**
   - Use PBS interactive mode to enter a UNIX console to a compute node, and run your jobs from there.
   - Suitable if you want to run R interactively on FLUX
3. **Batch jobs**
   - Most common use. Create batch scripts and submit them

Writing PBS batch script

- **Start from the guide**
  http://arc-ts.umich.edu/flux-user-guide/
  http://www.youtube.com/watch?v=SW8Lu1-JaSM
  http://arc-ts.umich.edu/software/torque/pbs-template/

- **At the minimum you will need**
  # PBS -V (to inherit the environment variables)
  # PBS -A bios815_flux (account info)
  # PBS -l qos=flux (quality of service, use flux for standard jobs)
  # PBS -q flux (queue of the jobs, use flux for standard jobs)
  # PBS -l nodes=1,pmem=4gb,walltime=4:00:00
  (specify the resources you need)

Running interactive jobs



Running batch jobs

- The commands after the preambles will run in a serial manner
- Once jobs are queued, the scheduler will execute the jobs in the order of scheduler's priority.
- Simulation, MCMC are good example jobs to run in batch in a massively parallel way

Some examples we’re going to work on..

- Power simulation
- MCMC / Gibbs sampler
- Large number of statistical tests