Research Tips & Tricks

Arthur Delarue

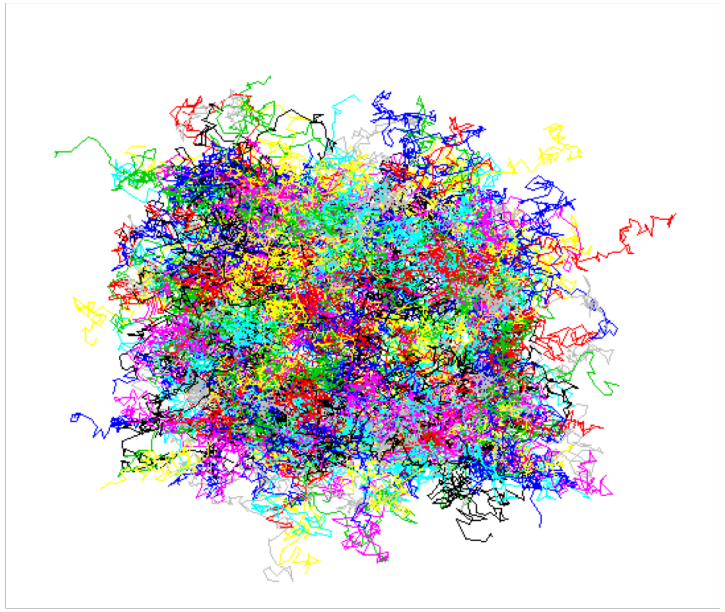
Using materials by Jackie Baek, Phil Chodrow, Sebastien Martin and Chris McCord

Outline

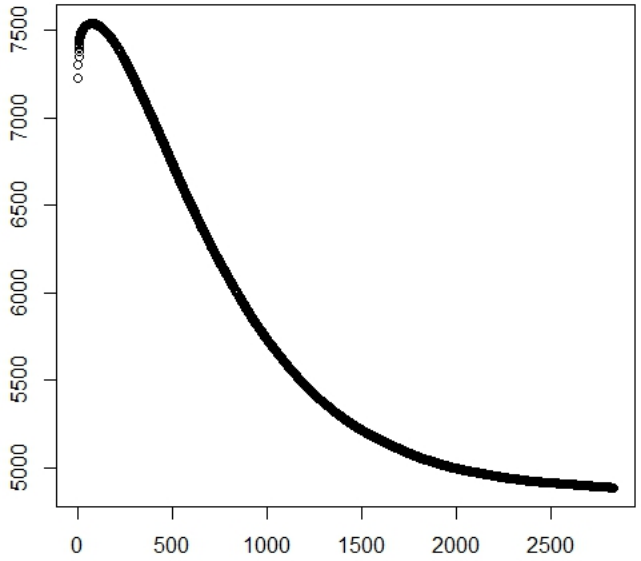
1. Compute *like a boss*
2. Write up your results *like a boss*
3. Become a knowledgeable and engaged citizen in your field *like a boss*

Distributed computing for Optimization & Stats

In optimization and stats, we often need a lot of computational **power**:



Machine learning on a large dataset



Simulations that require many iterations to converge

GUROBI OPTIMIZATION

Hard optimization problems

OPERATIONS RESEARCH CENTER

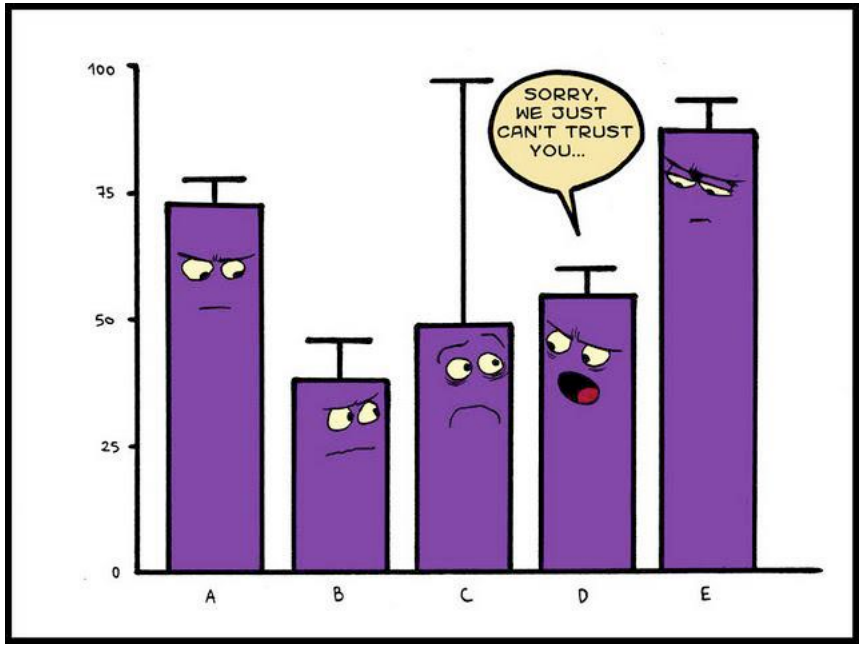
Distributed computing for Optimization & Stats

We often need to perform **repetitive** computational tasks:

Tuning parameters

"You can't keep adjusting the data to prove that you would be the best Valentine's date for Scarlett Johansson."

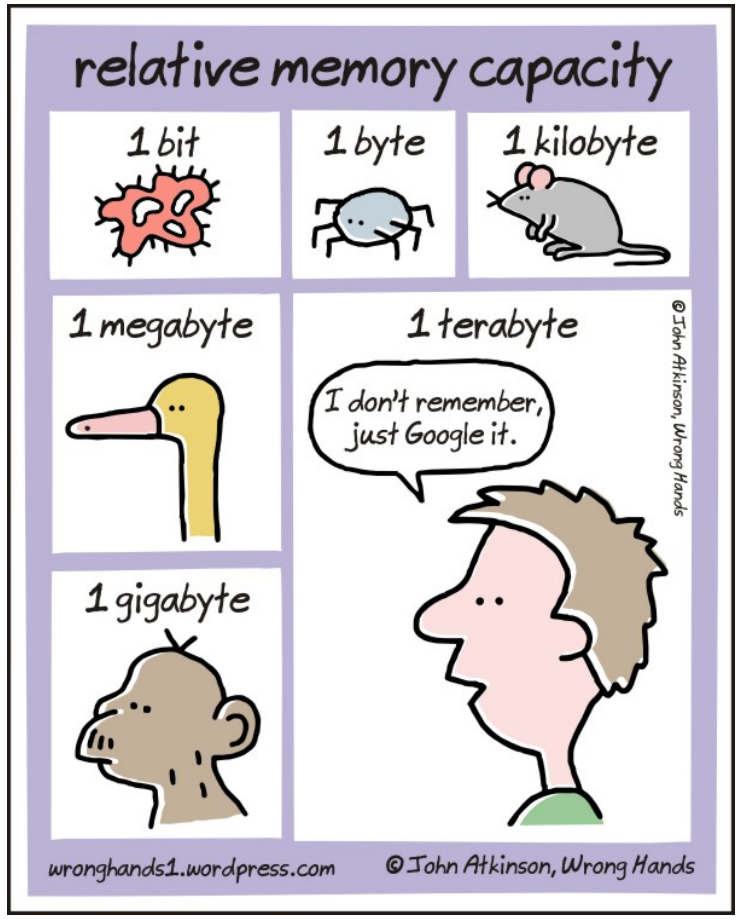
Benchmarking



OPERATIONS RESEARCH CENTER

Why your personal computer might not cut it

Limited memory



Limited processing power/cores

Limited time

Available resources

If you need more computational power, there are several ways you can access a remote computer or cluster:

- Another (bigger) personal computer
- Athena (at MIT)
- Resources of your department/lab (Engaging at Sloan)
- Cloud computing service (Amazon AWS, Microsoft Azure, Google Cloud Computing…)

Beyond the buzzwords

Remote computing: execute commands on a remote machine

Parallel computing: execute multiple commands simultaneously

Distributed computing: parallel computing without shared memory

Why should I use shared/parallel computing?

Why should I use this?

In your research:

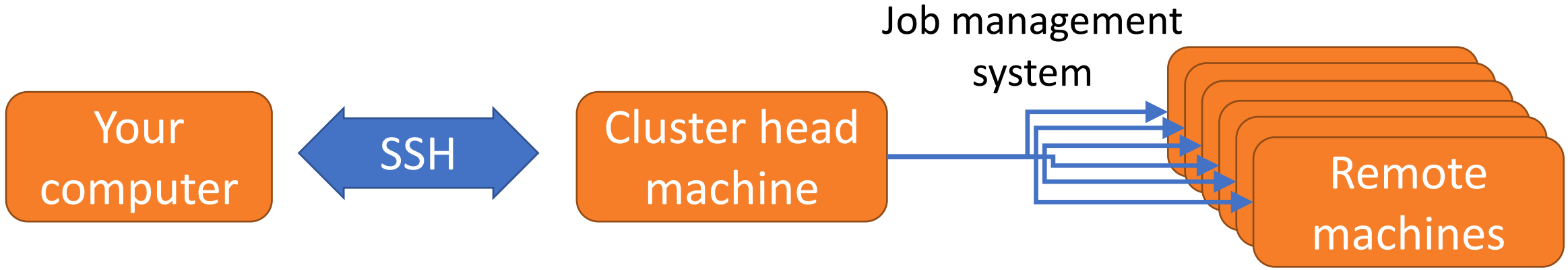
- Tackle bigger datasets
- Parallelize (and speed up) your computations
- Run simulations overnight or for a whole week!
- Can use all of the tools we learned about in the class (Rstudio…)

In general: it’s pretty dang useful!

How it works: using a remote computer

- We use **SSH** (see lecture 1) to run commands on a remote machine through our computer.
- We can use the **shell/terminal** to do almost anything on the remote computer: create files, run a program, use Julia...
- It is also possible to use GUI applications like RStudio or Matlab.
- It works with any computer, including your own machines.

How it works: using a cluster



When a computing cluster is available, we can run multiple "jobs" thanks to a job management system.

- Different job management systems exist, we will use the example of Slurm, which the cluster Engaging uses.
- More complex to use, but far more powerful.

Engaging

- Engaging is a powerful computing cluster for MIT Sloan affiliates, request an account at stshelp@mit.edu.
- Today we will work with its job management system, Slurm.

Hostname	eosloan.mit.edu
Username	MIT Kerberos Username
Password	MIT Kerberos Password

https://wikis.mit.edu/confluence/display/sloanrc/Engaging+Platform

Athena

- Any MIT affiliate can access the Athena distributed computing environment, you can use it to follow if you do not have access to Engaging.

Hostname	athena.dialup.mit.edu
Username	MIT Kerberos Username
Password	MIT Kerberos Password

http://web.mit.edu/dialup/www/ssh.html

Let’s connect to Engaging!

We can connect using SSH:

```
ssh <username>@eosloan.mit.edu
```

File Transfer

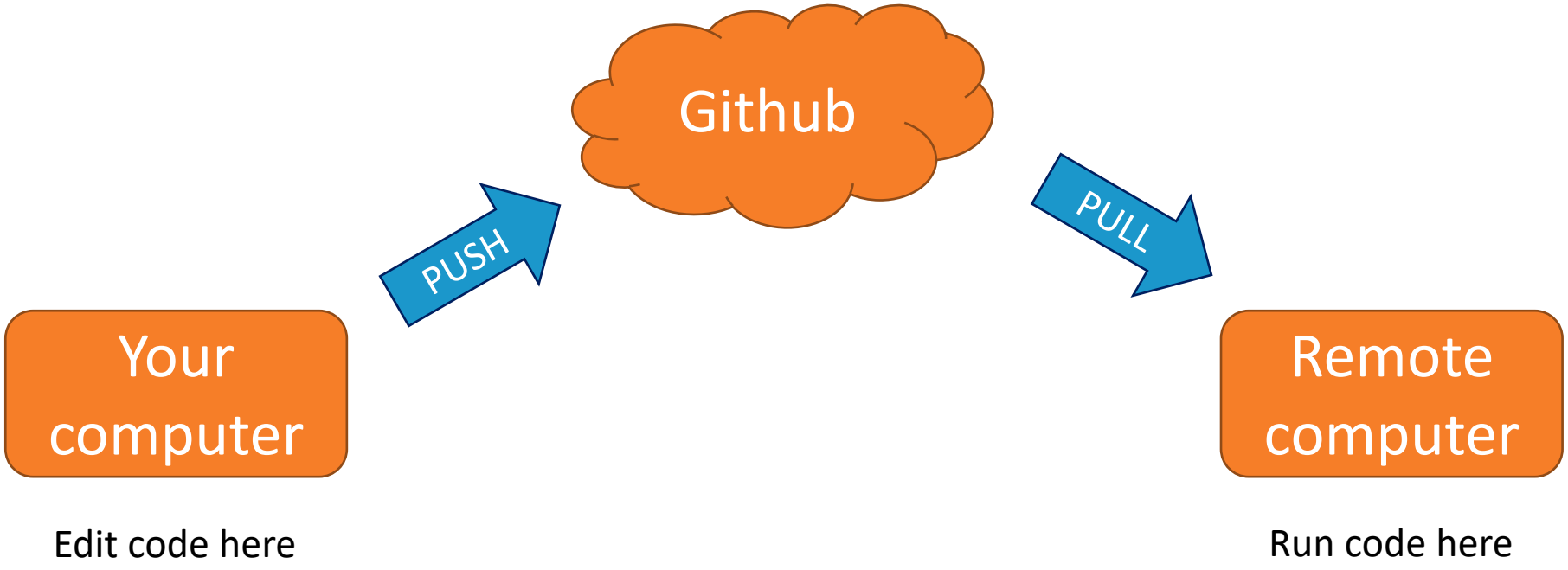
- Downloading from the Internet (useful to install software, or download datasets): **wget**

- Transferring files between the local and remote machines: Secure Copy **scp**

```
scp file.csv <username>@eosloan.mit.edu:file.csv
```

A Git/Github workflow for the cluster

- This is an elegant workflow to run code on the remote machine:

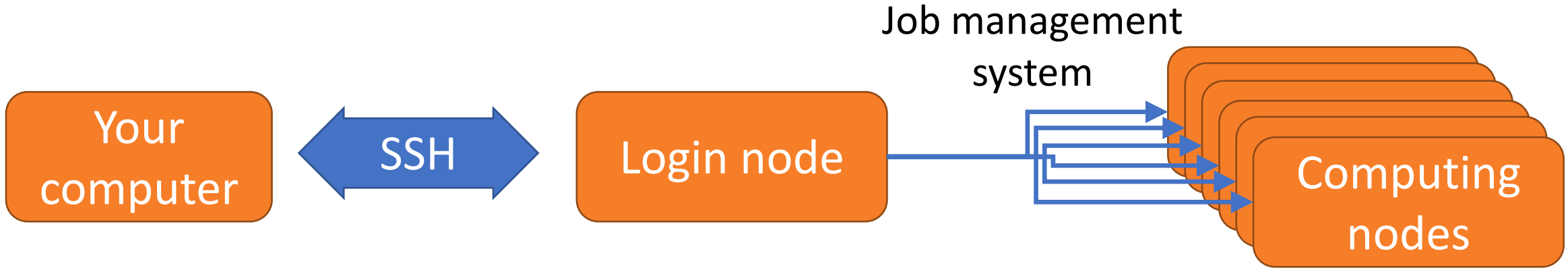


For advanced users

Feel free to read up if you're interested!

- **sshfs** allows you to access the files of the remote machine as if they were on your own computer (feels like magic)!
- **screen** allows you to keep your session active after you disconnect.
- **tmux** (terminal multiplexer) is like screen with lots of additional functionalities (panes...)

How it works: the job management system



- When we login to Engaging, we are actually logging into a special **login node**.
- The actual work will be done by the **computing nodes**.

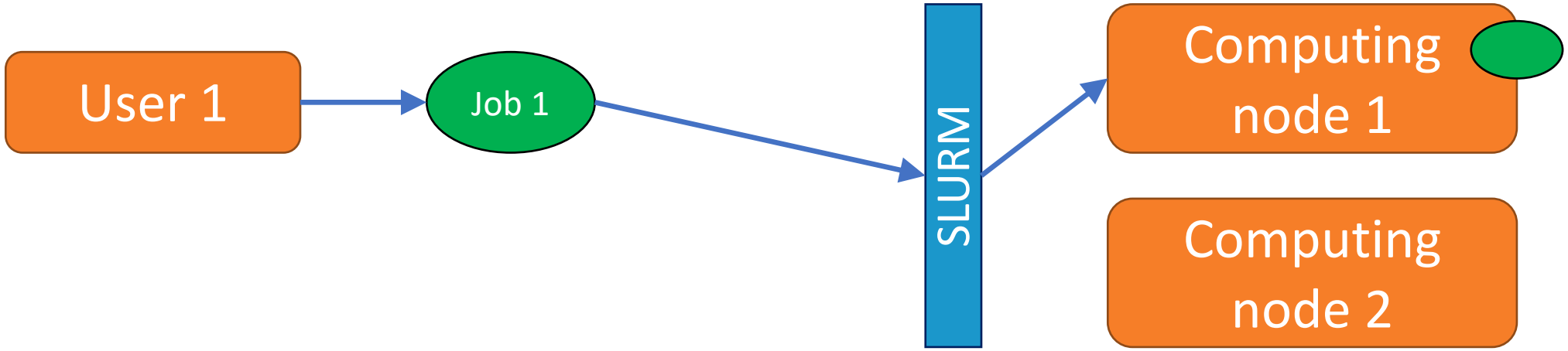
Never run programs on the login node!

It has very little computing power and you WILL paralyze it for other users who just want to submit their jobs.

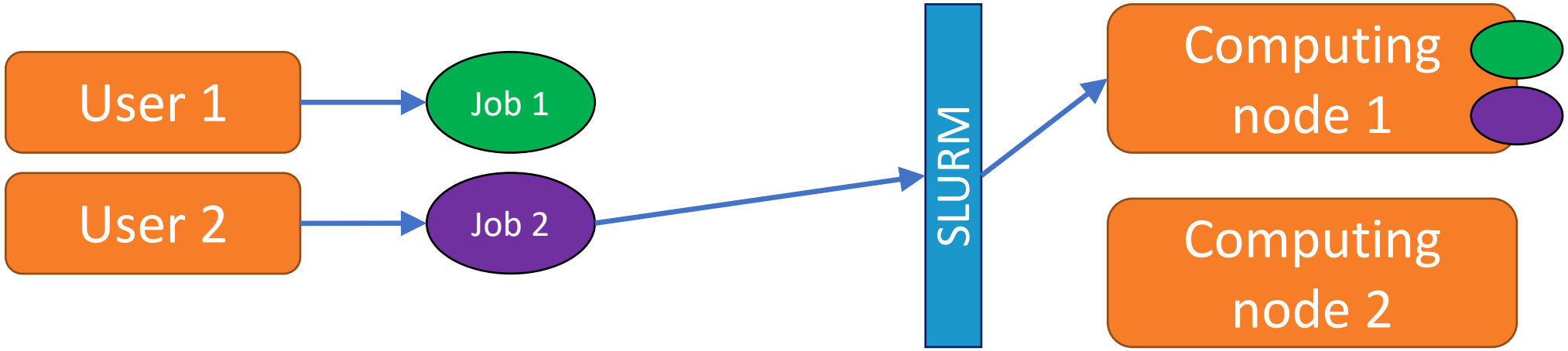
Submitting jobs

Job	A program to be executed (running a R/Julia source file, a bash script...)
Resources	Resources required to run the job (memory, cpus, time, software requirements...)

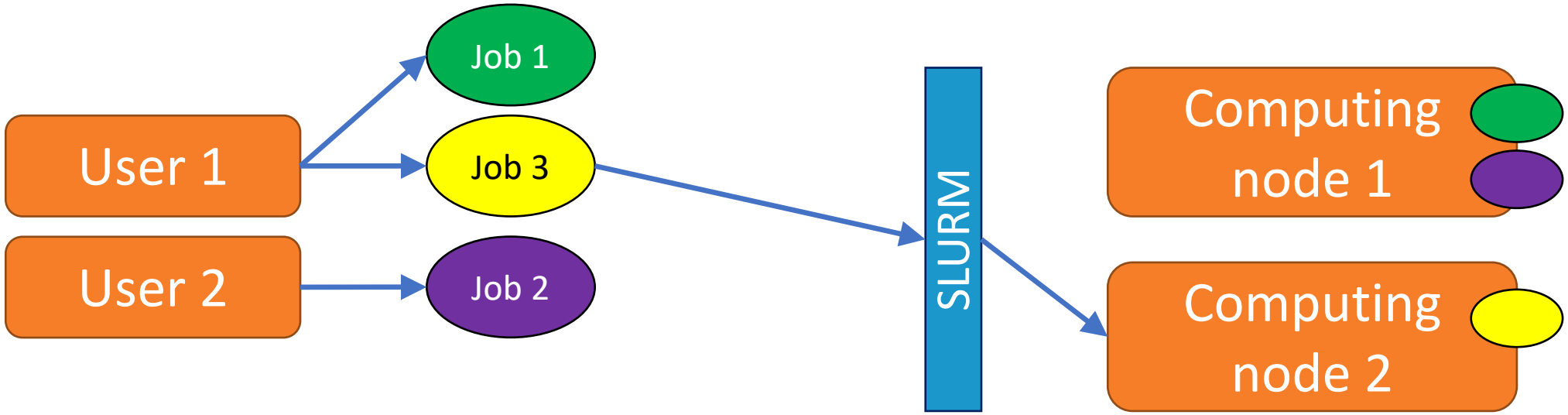
Submitting jobs



Submitting jobs



Submitting jobs



Module loading

Many packages are installed on the cluster. To use them, we use the following special commands.

- Show all modules: `module avail`
- Search for modules: `eo-module-find Julia`
- Load a module: `module load sloan/julia/1.0.0`

Submitting a job on Engaging

To submit a job to SLURM, we use the command srun. The main resource parameters are:

Memory	`--mem-per-cpu=1G`
CPUs	`--cpus-per-task=1`
Time	`--time=1-12:00`
Partition	`--partition=sched mit sloan interactive`

- The partition is the name of the sub-cluster you want to use. The two main partitions are `sched_mit_sloan_interactive` and `sched_mit_sloan_batch`.
- When running interactive jobs, we should use the `--pty` flag to start the session in the current window.

Monitoring your jobs

- You can check if your job is running or in the queue:

```
eo-show-myjobs
```

- You can cancel a job by using

```
scancel <JOB_ID>
```

A hierarchy of parallel computing

- Easy to parallelize
  - Multiple instances of the same program with different inputs
  - Each instance can run independently, e.g. grid search for hyperparameter tuning
- Parallel computing with shared memory
  - Commands execute simultaneously on separate CPUs, but work on shared memory
  - Computation cannot (easily) be split across separate nodes because communication between processes is necessary
  - e.g. matrix multiplication, parallel SGD
- Advanced topics
  - Distributed computing with a message passing interface (MPI)
  - GPU computing