Sharing research compute

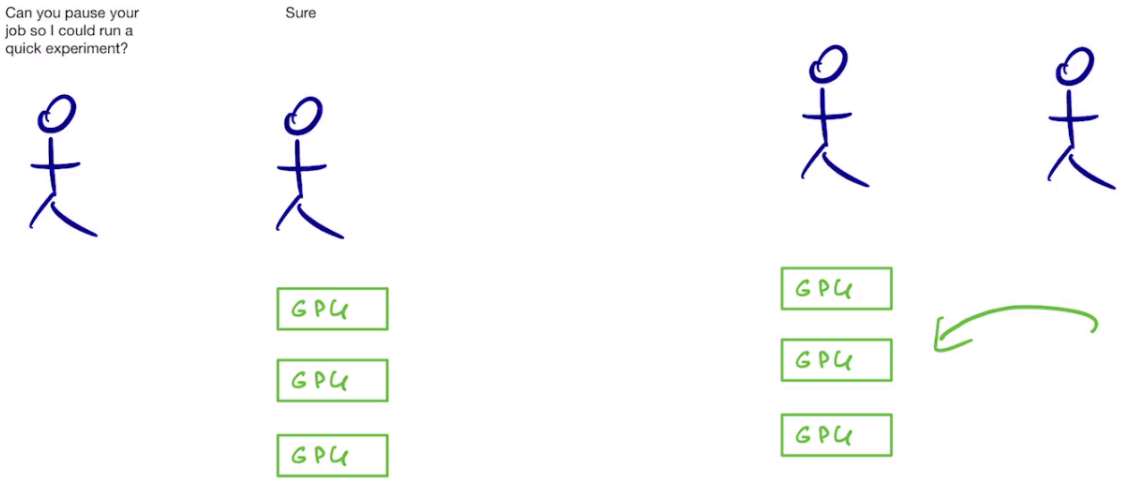
[Sharing research compute](#)
- [Compute sharing](#)
- [Resource separation](#)
- [Antipattern of who calls the shots](#)
  - [Sys-admins](#)
  - [Program managers](#)
  - [Researchers](#)

[Overall](#)

Compute sharing

Small scale, informal

Easiest case is two people sitting next to each other



Everyone has an approximate understanding of requirement/value for each task, so near-optimal balancing happens naturally

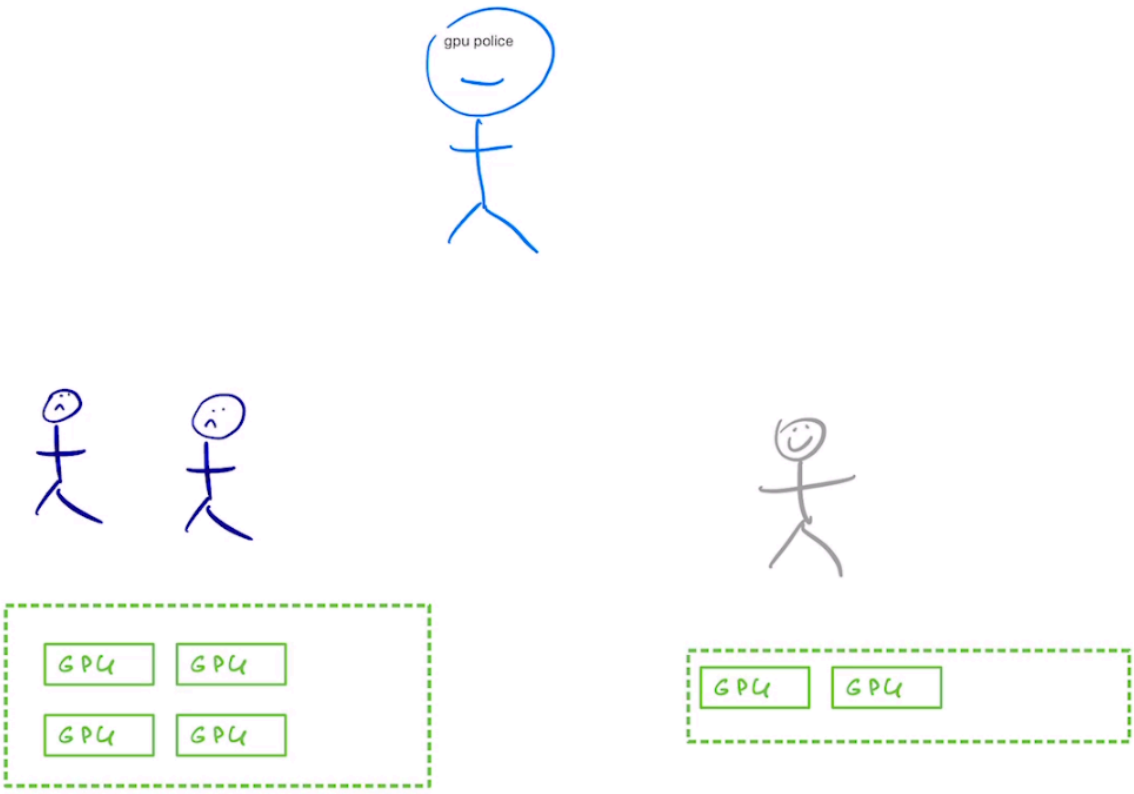
At some point not everyone knows who needs what, and it becomes "my GPUs vs. others"

Intern invasion

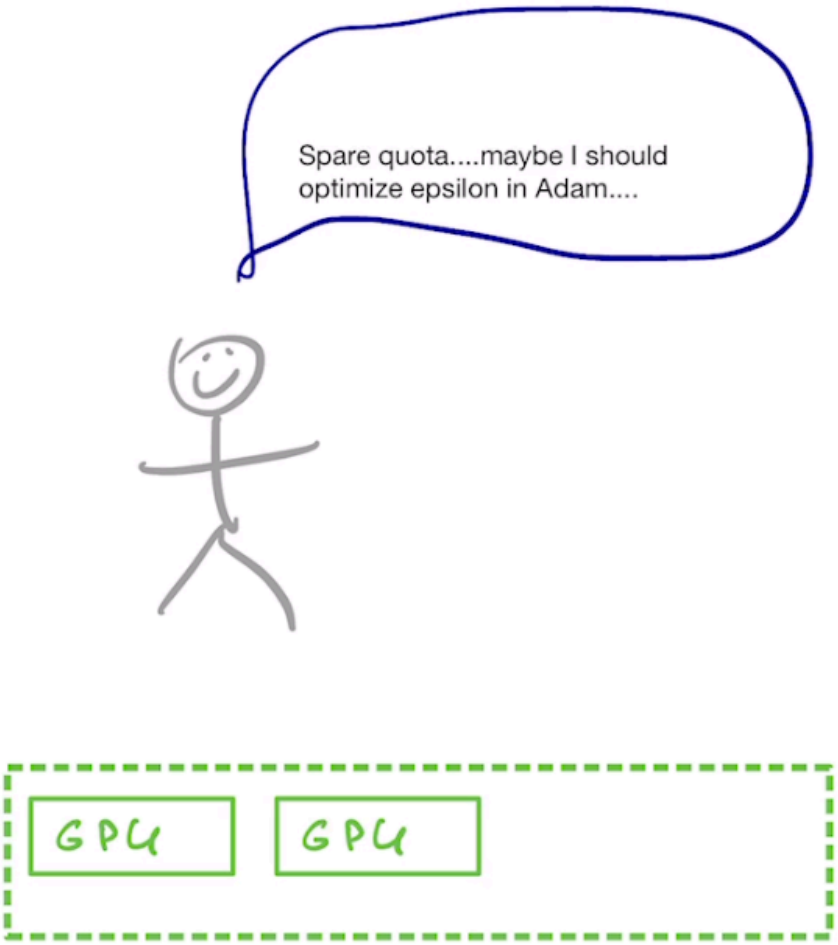
Common special case is intern season -- interns have no sense of what the cluster is used for. Interns can just keep adding "0"s to their reservation until their run fails to schedule.

Large Scale, Program Managers

At such times the job of balancing the needs is delegated to the program manager. They get priorities from the leadership and enforce various groups not using more than allocated by using a quota system.



The downside is that quota information is imperfect so delegation is suboptimal. Someone may underuse their quota and then fill the rest with low value hyper-parameter sweep.



An anti-pattern is when people are punished by underutilizing their quota. For instance, if the future quota of the team is estimated based on current usage -- "use it or lose it". Google occasionally launched "utilization code yellow" sprints where PMs encouraged people to increase their compute usage to fill their allocation. Locally optimal (100% utilization of quota, OKRs are met), globally suboptimal.

An alternative is to give people an incentive to go under their quota. Example from Google air travel expenses -- you have a hard upper quota, going under the quota earns you points that you can use for future travel upgrades.

Still may need some kind of quota system because some tasks are inherently more resource costly. IE 7B finetining vs 405B.

Resource separation

People sharing machines can step on each other. Slurm provides some resource separation using cgroups. Occasionally conflicts still occur (ie, bad config, two users sharing QPI link).

Removing sudo prevents users from breaking config.

The flip side of this is someone could have a use-case they don't know how to get working under resource separation constraint. Restricting this use-case without first educating them can be demotivating.

As a bridge -- log all users using SSH and ask why they need it

A: What are you using SSH for?
B: I wanted to upgrade things so I did "sudo apt update sudo apt-get remove '^nvidia'"
A: Isn't that going to break other users of the cluster?
B: I didn't think of that....

A. why SSH?
B. getting weird behavior, want to rule out Slurm cgroup interaction
A. make sure nobody else is using the machine at the time, use Slurm "--exclusive" to reserve

Antipattern of who calls the shots

Systems evolve in different ways depending on which groups ends up with the most political influence

Sys-admins

users breaking each other increases sys-admin work, hence they want to add maximum restriction

Pro: no resource conflicts
Con: bureocracy, long-approval times to install some package, demotivated researchers

Program managers

Get priorities, instill quota system, shift quotas around in response to utilization
Pro: easy
Con: inefficient use of resource, "use it or lose it" mentality

Researchers

Use cluster machines like a personal machine
Pro: freedom
Con: doesn't scale past 2 researchers

Overall

- Hard quota based on underlying requirements
- An incentive for groups to go under their quota
- Unused compute available for ad hoc experiments
- Incentive to be efficient/conservative with ad hoc runs

- Usage logged. Visibility into reserved+unutilized GPUs.
- Some visibility into experiments without having to attend 100x meetings

Perhaps Special handling for interns (perhaps no SSH access by default, their host decides)