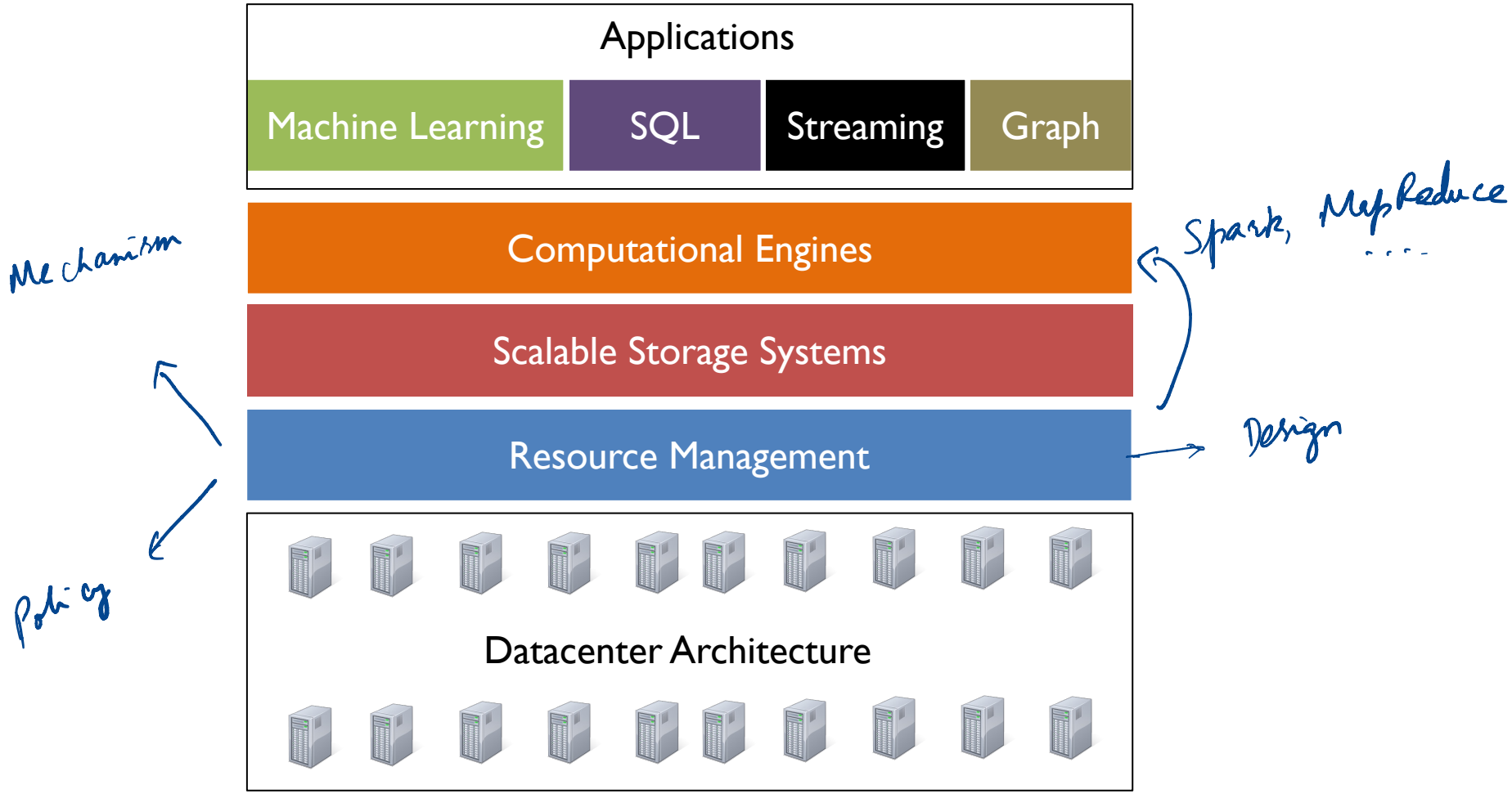
CS 744: MESOS

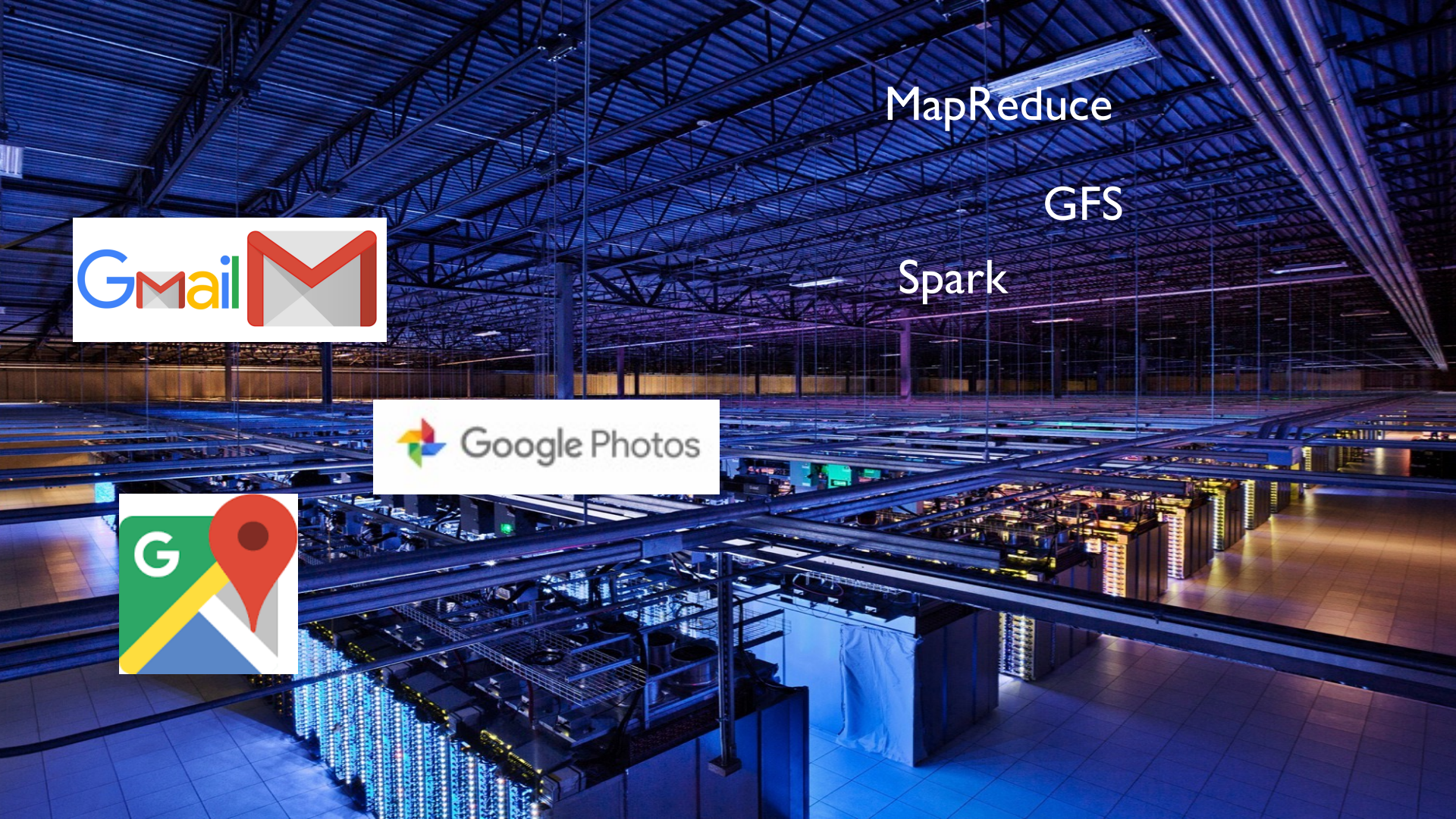
Shivaram Venkataraman

Fall 2022

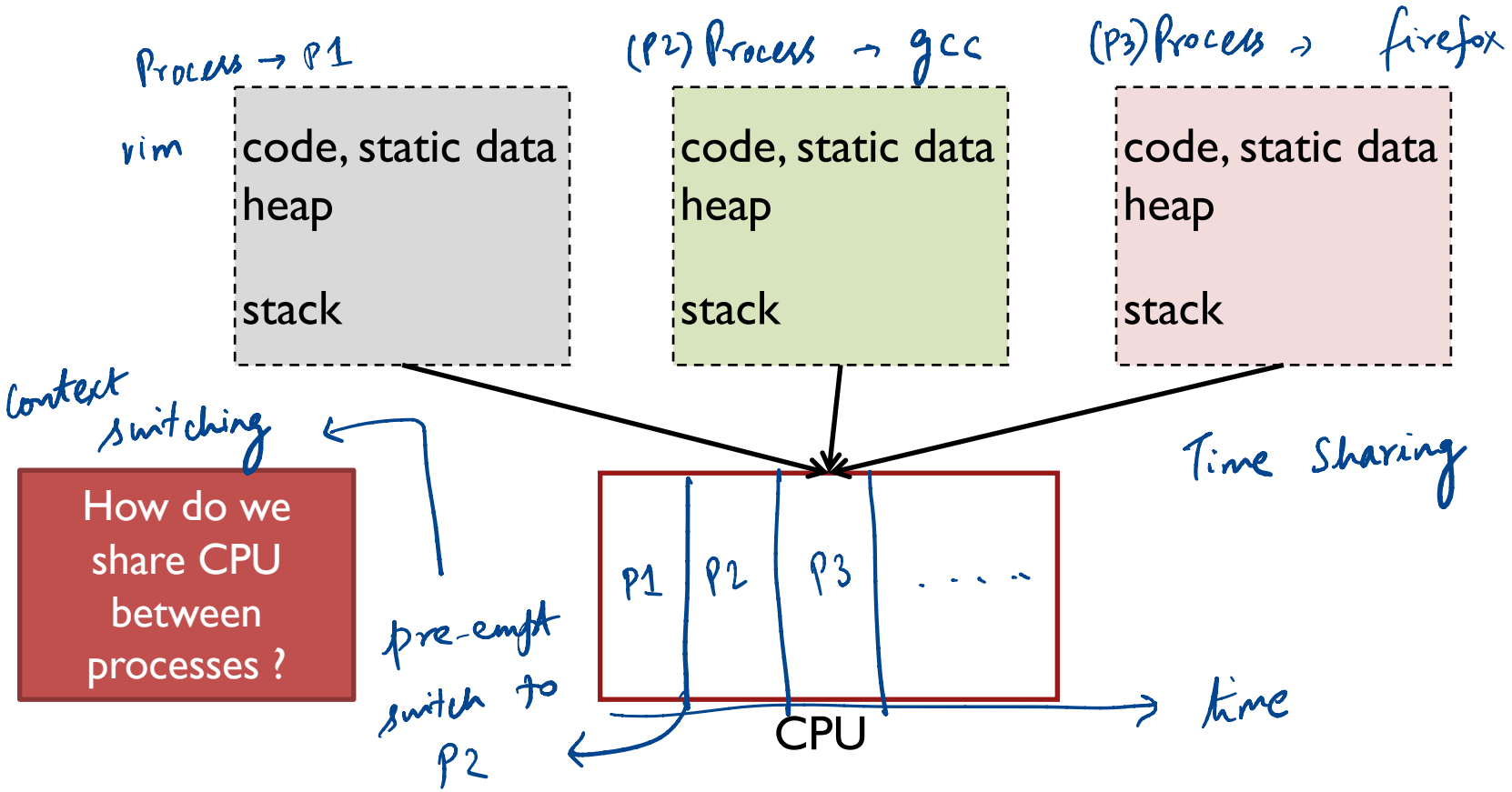
ADMINISTRIVIA

- Assignment 1: Due Sep 28th at 11am! →
- Assignment 2 out soon!
- Project details
  - Create project groups
  - Bid for projects/Propose your own
  - Work on Introduction
  - Final report / poster presentation


Applications
Machine Learning
SQL
Streaming
Graph
Computational Engines
Scalable Storage Systems
Resource Management
Datacenter Architecture
Mechanism
Policy
Spark, MapReduce
....
Design


MapReduce
GFS
Spark
Gmail
Google Photos

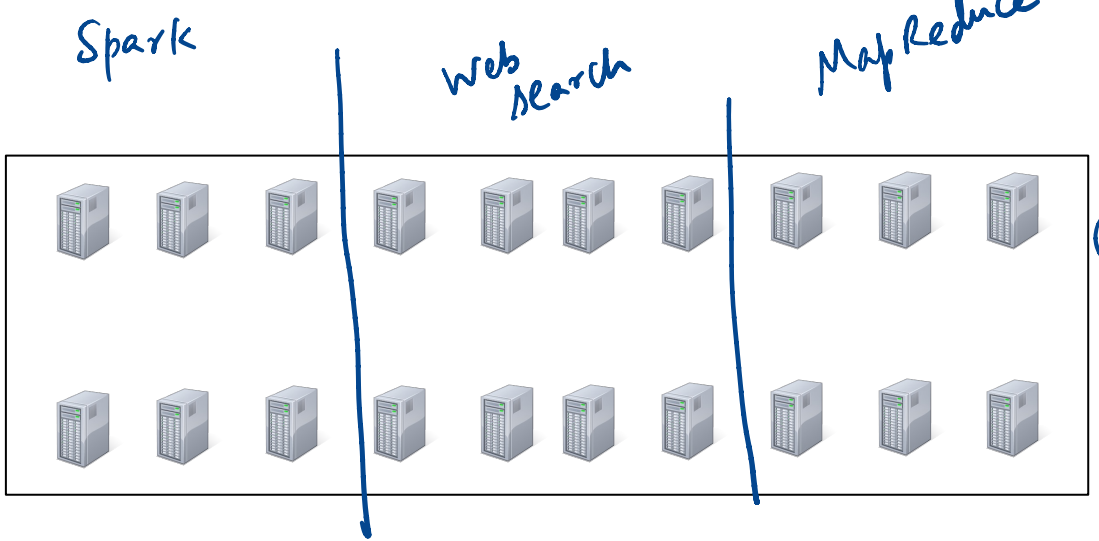
BACKGROUND: OS SCHEDULING



CLUSTER SCHEDULING

Space Sharing

- Partition resources many applications run at the same time



1. Locality of frameworks to resources
2. Minimize wasted work, but ensure frameworks get sufficient resources
3. Applications are complex and diverse

TARGET ENVIRONMENT

→ ~2006 - 2009 Hadoop releases

Multiple MapReduce versions

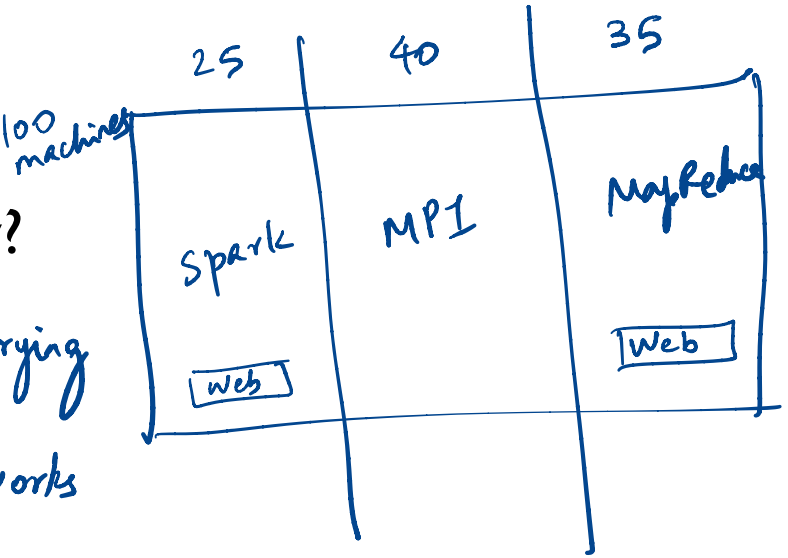
→ share the clusters

Mix of frameworks: MPI, Spark, MR

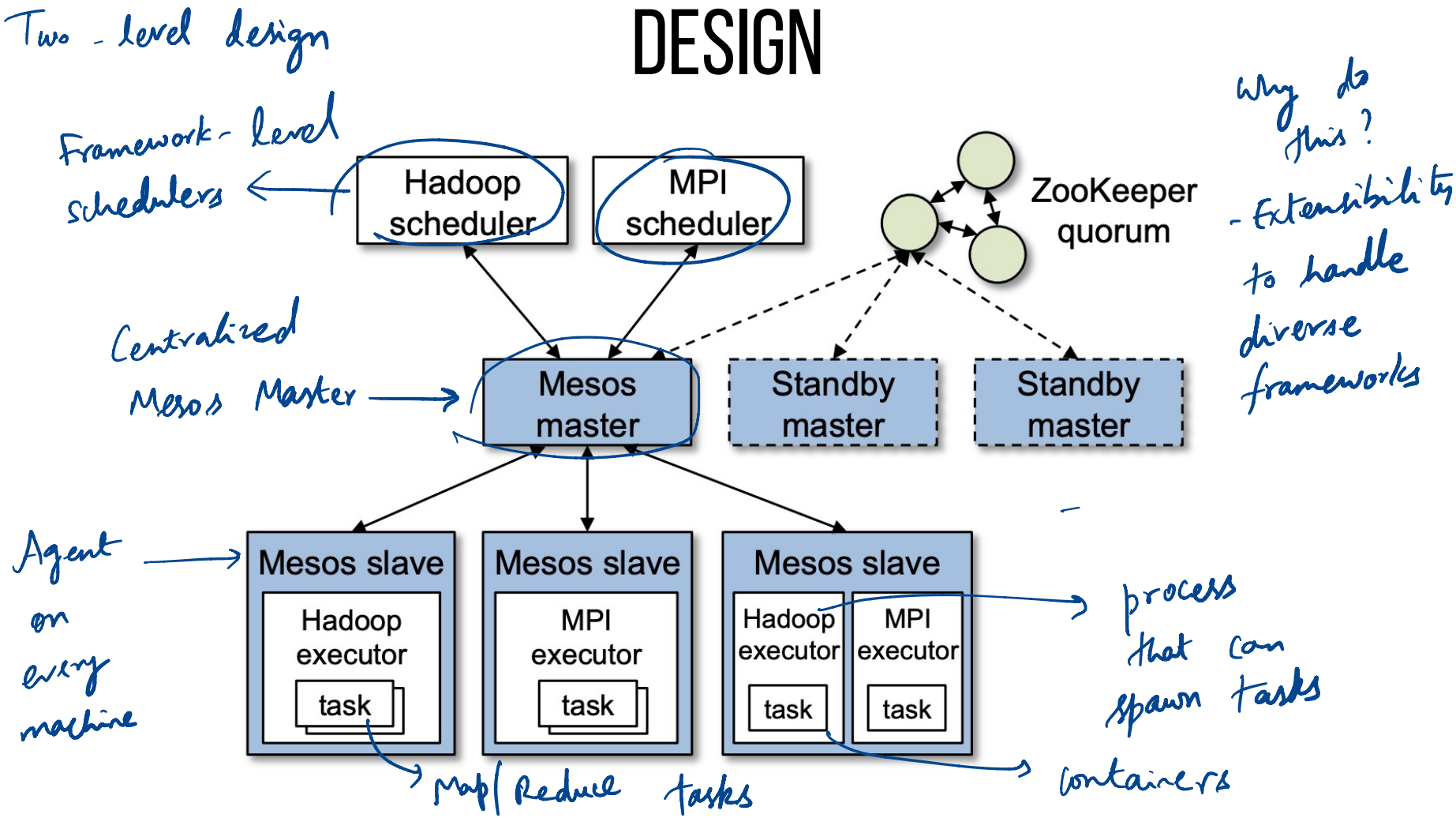
Avoid per-framework clusters. Why?

→ Under utilization
  ↳ workload time varying

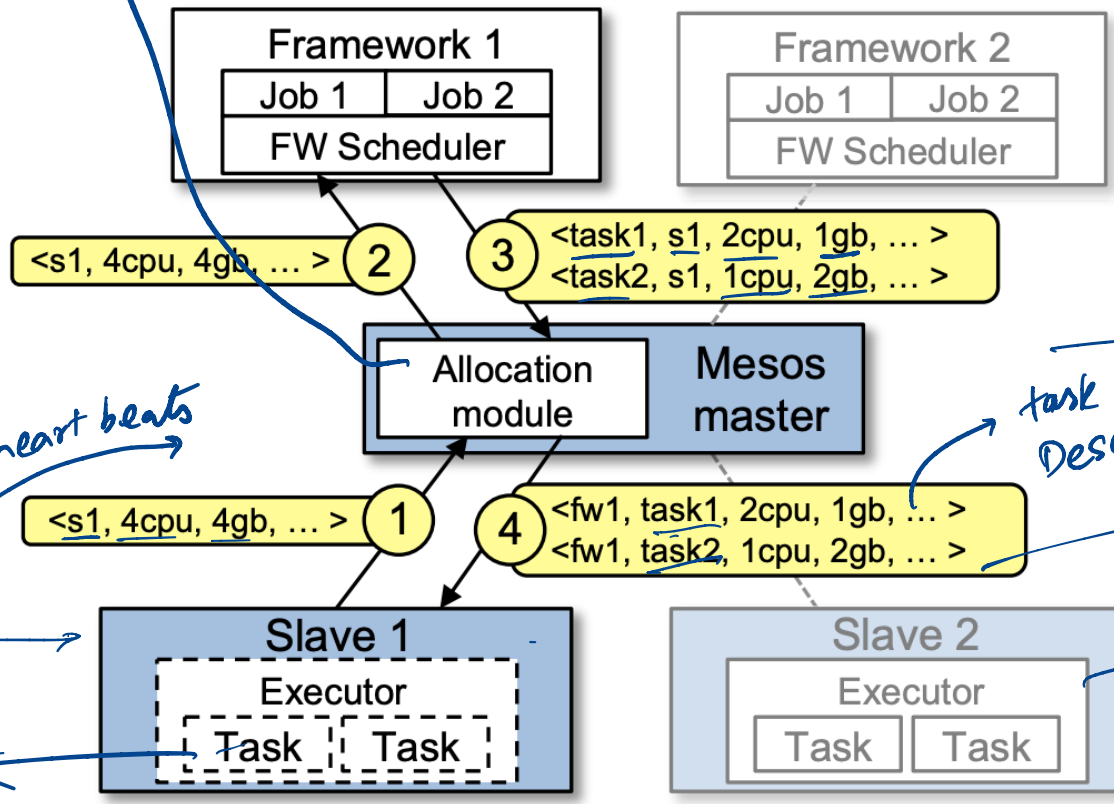
→ Data sharing, composing frameworks



DESIGN


Two - level design
Framework - level schedulers
Hadoop scheduler
MPI scheduler
ZooKeeper quorum
Why do this?
- Extensibility to handle diverse frameworks
Centralized Mesos Master
Mesos master
Standby master
Standby master
Agent on every machine
Mesos slave
Hadoop executor
task
Mesos slave
MPI executor
task
Mesos slave
Hadoop executor
task
MPI executor
task
process that can spawn tasks
Map/Reduce tasks
containers

RESOURCE OFFERS



CONSTRAINTS

Examples of constraints

- Data locality → soft constraint
- GPU machines → hard constraint

only want resources that have your data

Constraints in Mesos:

- Applications can reject offers
- Optimization: Filters
  - ↳ only resources that pass this filter are offered.

DESIGN DETAILS

Allocation:

- Tasks are short, allocate when they finish
- Long tasks? Revocation beyond guaranteed allocation

Isolation

- Containers (Docker)

→ example: if you allocate 1 CPU, 1GB to Spark to run a task

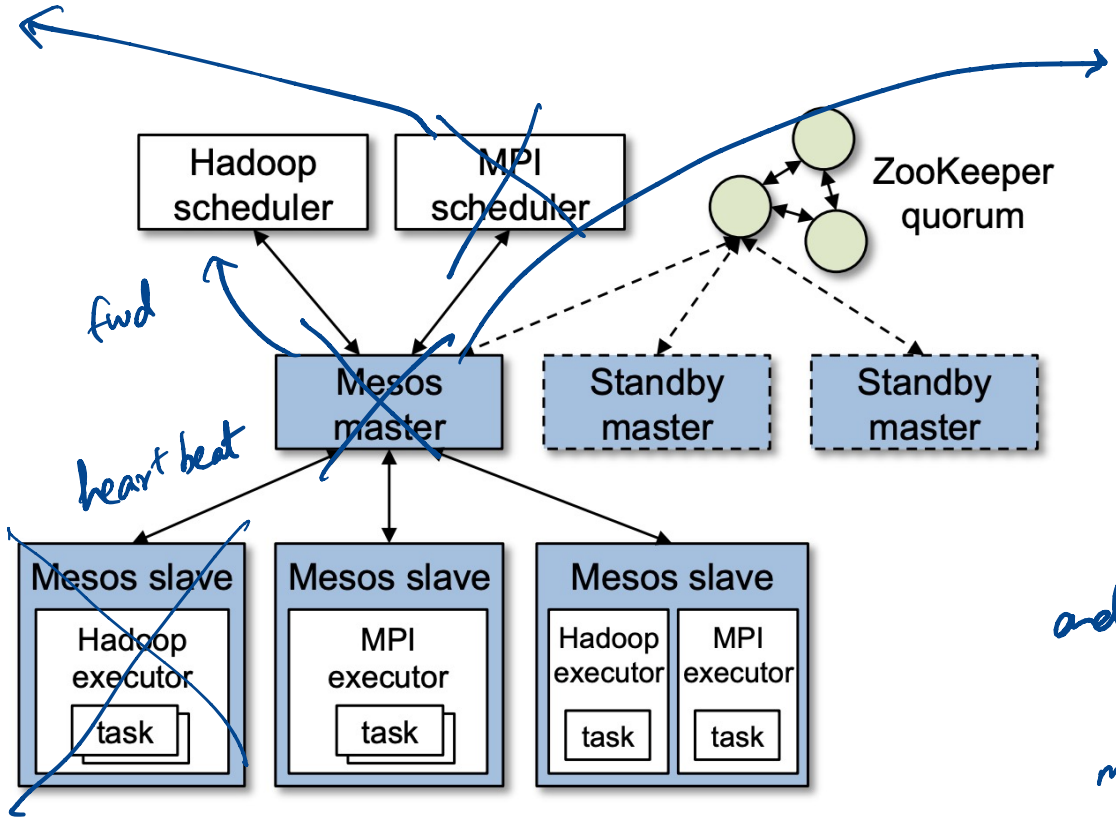
→ pre-emption

Policy can say that framework is exceeding its guaranteed allocation

excess

FAULT TOLERANCE

relaunch it?



soft-state can be reconstructed by talking to framework schedulers and Mesos agents on machines

tasks running fail

HANDLING PLACEMENT PREFERENCES

↳ Two-level sched. vs centralized sched.

What is the problem?

More frameworks have preferred nodes than available

Who gets the offers?

How do we do allocations?

Lottery scheduling – offers weighted by num allocations

weighted offers based on job size → larger jobs get a fraction

CENTRALIZED VS DISTRIBUTED

Framework complexity → Mesos every framework needs to write their scheduler

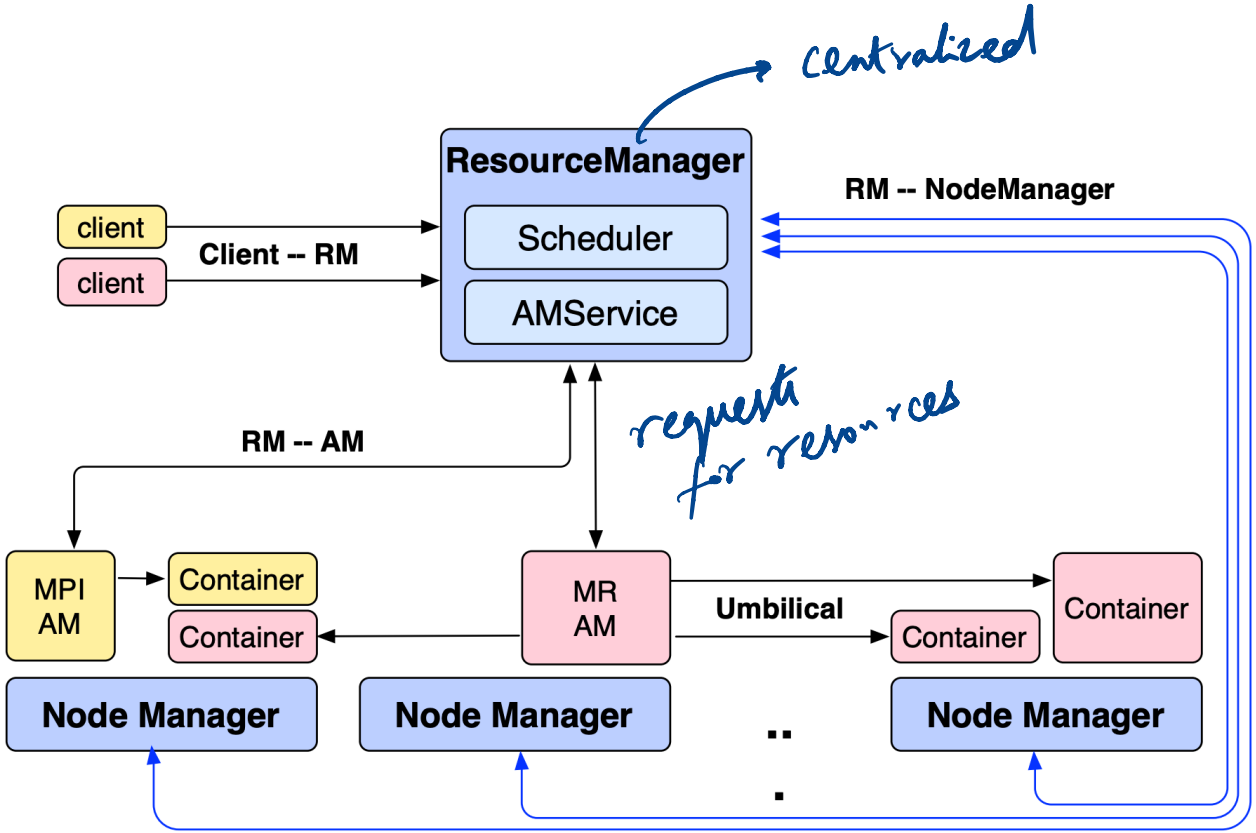
Fragmentation, Starvation

↳ if you a mix of small tasks and very large tasks

Inter-dependent framework

↳ cannot enforce such constraints

COMPARISON: YARN → Apache



Per-job scheduler

↓

Application Manager

AM asks for resource

RM replies

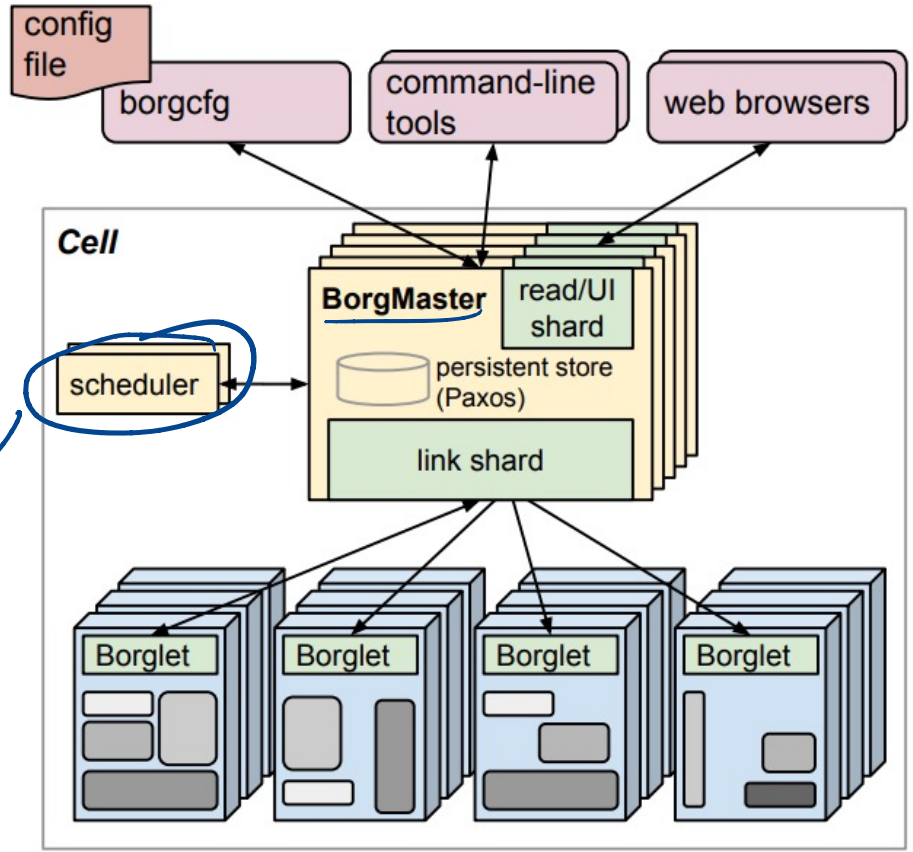
COMPARISON: BORG → Kubernetes

Single centralized scheduler

Requests mem, cpu in cfg
Priority per user / service

Packing of applications

Support for quotas / reservations



SUMMARY

- Mesos: Scheduler to share cluster between Spark, MR, etc.
- Two-level scheduling with app-specific schedulers
- Provides scalable, decentralized scheduling
- Pluggable Policy ? Next class!

DISCUSSION

https://forms.gle/D1sqfzD3GqxQC4Y97

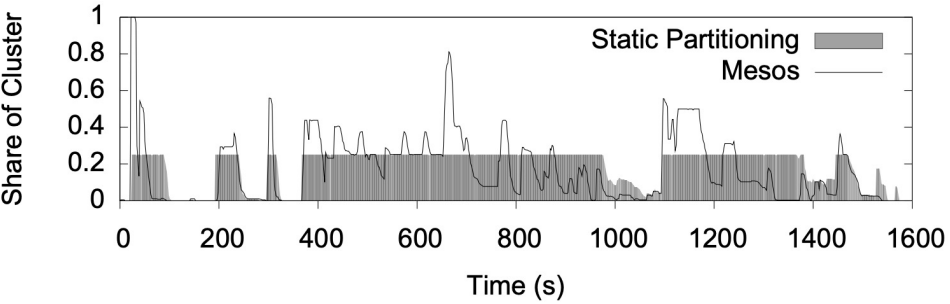
What are some problems that might arise if you wanted to use Mesos with frameworks that had very low latency tasks (e.g., for interactive analytics)

- Mesos master needs to make offers frequently
- Could lead to fragmentation / starvation
- Tasks might get slowed down waiting for offers

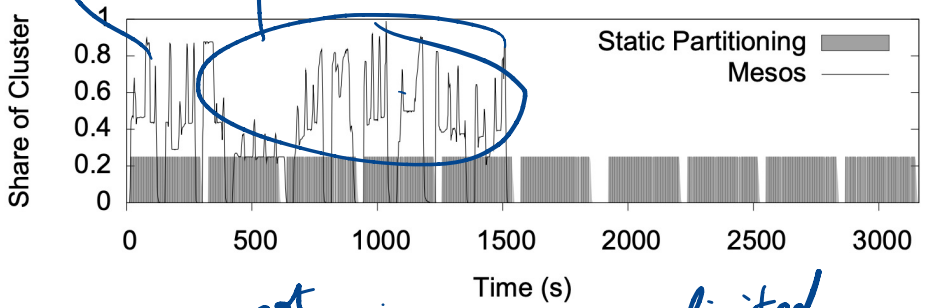
Cluster utilization is higher

If framework is elastic get benefits

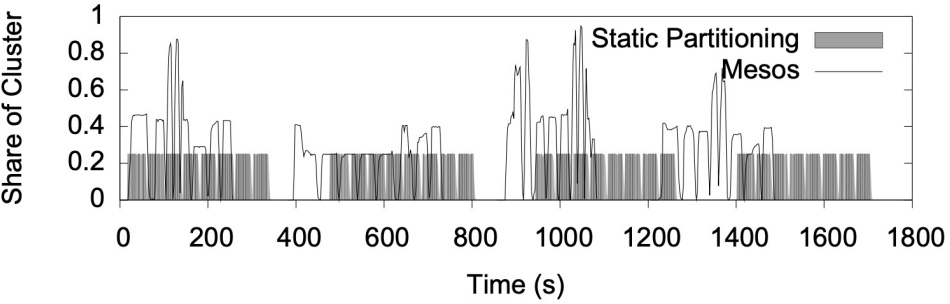
(a) Facebook Hadoop Mix



(b) Large Hadoop Mix



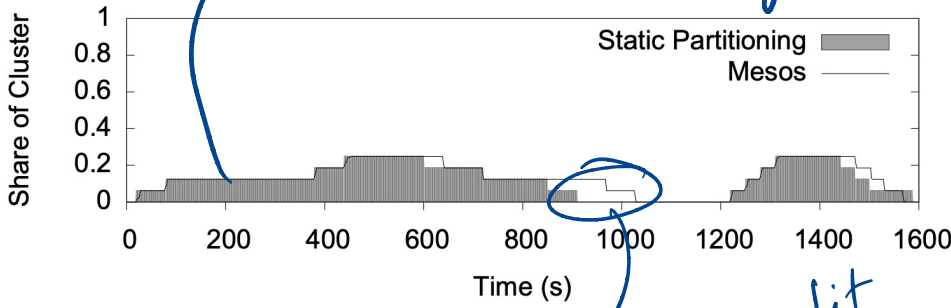
(c) Spark



not elastic

limited benefits

(d) Torque / MPI



Mesos is bit worse

NEXT STEPS

Next class: Scheduling Policy

Further reading

- https://www.umbrant.com/2015/05/27/mesos-omega-borg-a-survey/
- https://queue.acm.org/detail.cfm?id=3173558