Hello!

CS 744: DRF

Shivaram Venkataraman

Spring 2024

ADMINISTRIVIA

- Assignment 2 done! → Assignment 1 grading
  ↳ Wed / Thu
- Course Project
  - Form groups and submit project bids by tonight!
    ↳ preferences
  - Assigned projects by March 1
  - Introductions due March 8

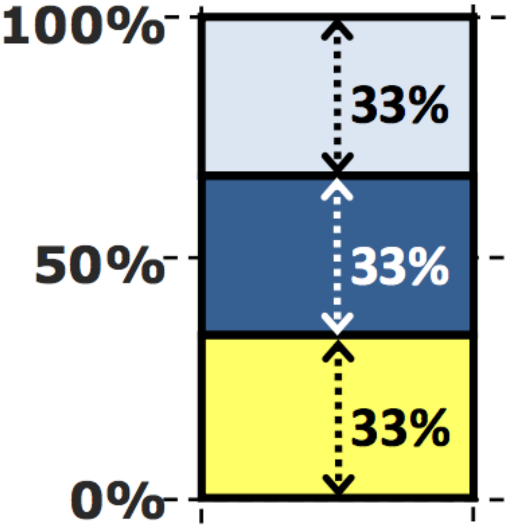
→ Come up at the end of the class

SETTING: FAIR-SHARING

3 clients or users

1 resource = CPU

Equal Share

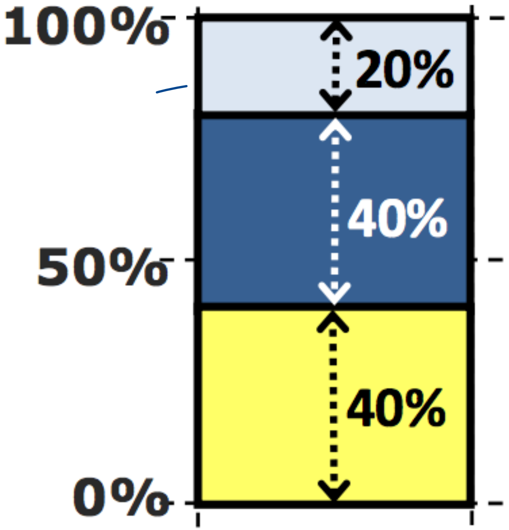


Max-Min Share

Maximize the allocation for "most poorly treated" users

Maximize the minimum

demands of some users is greater than others



late 1980 network
mid 1990s

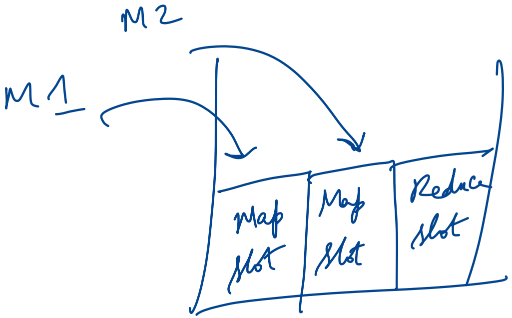
SLOT-BASED MODEL

Slot: Fixed quantity of CPU and memory → state of the art in 2009 / 2010

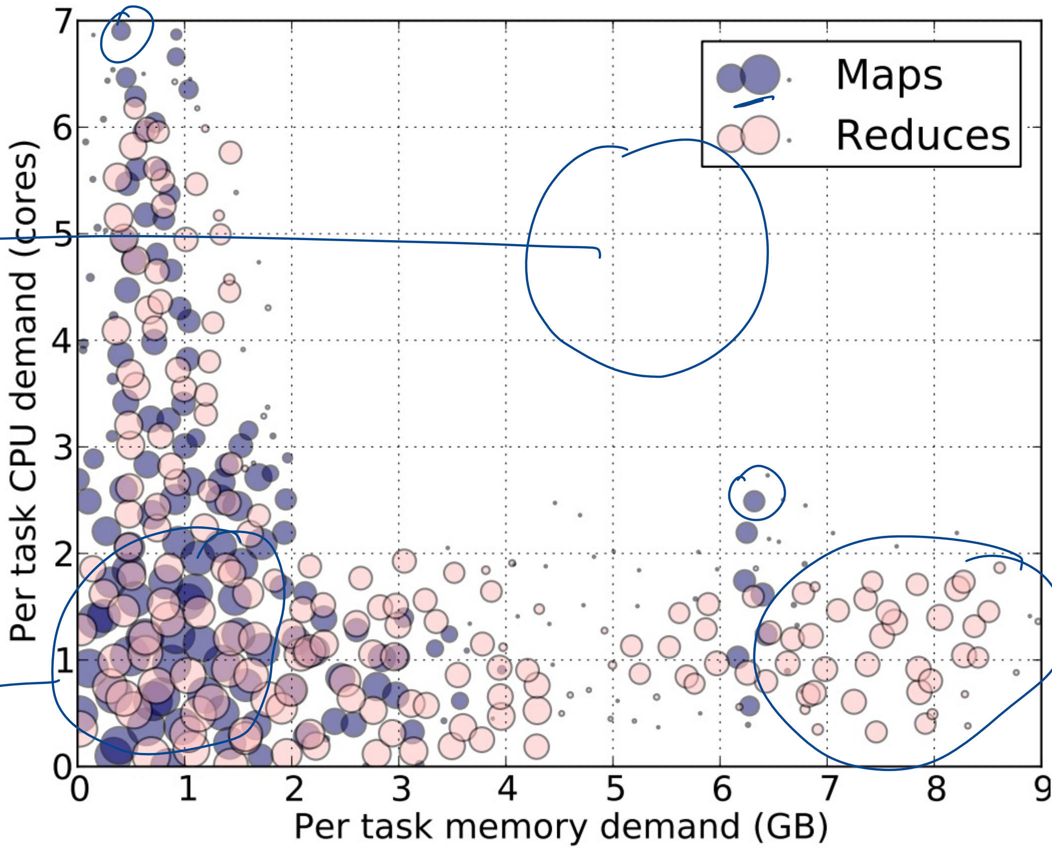
Example: Hadoop MapReduce

- Mapper: 2 CPU and 1 GB → use up 1 map slot
- Reducer: 1 CPU and 2 GB → 1 reduce slot

Allocate in units of slots



MOTIVATION: MULTI RESOURCES



empty

diff map tasks can have very varied resource needs

most tasks are here

Reduce tasks more memory intensive

DRF: MODEL

Users have a demand vector → generalization for multiple resources

<2, 3, 1> means user's task needs 2 R1, 3 R2, 1 R3

R1 → CPU, R2 → Mem, R3 → Disk

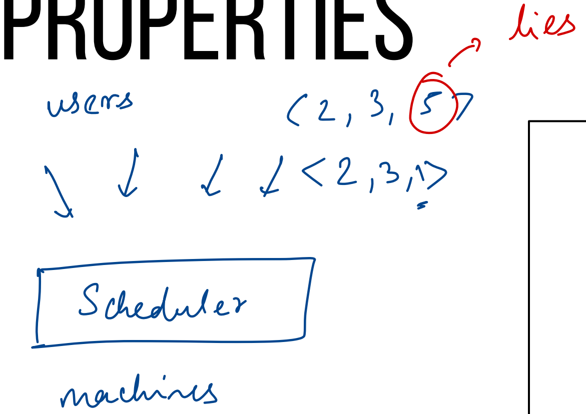
Resources given in multiples of demand vector

i.e., users might get <4,6,2>

10 map tasks in my job

<20, 30, 10> as my overall resource requirement

PROPERTIES



Sharing Incentive

At least as well off as having a partitioned cluster

Strategy Proof

Users cannot benefit from lying about resources required

Utilization ←

Pareto Efficiency

Not possible to allocate resources to one user without removing from another user.

Envy free

No user should desire another user's share

PROPERTIES

Sharing Incentive

User is no worse off than a cluster with

1/n resources

Strategy Proof

User should not benefit by lying about demands

Pareto Efficiency

Not possible to increase one user without decreasing another

Envy free

User should not desire the allocation of another user

DRF: APPROACH

Dominant Resource

Resource user has the biggest share of

Dominant Share

Fraction of the dominant resource user is allocated

Total: <10 CPU, 4 GB> Cluster

User 1: <1 CPU, 1 GB>

Dominant resource is memory

1/10 th CPU

1/4 th memory → Dominant

E.g., for User 1 this is 25% or 1/4

↓

Dominant share

DRF: APPROACH

Objective → Equalize the dominant share of users

Total: <9 CPU, 18 GB>

User1: <1 CPU, 4 GB>
dom res: mem

User2: <3 CPU, 1 GB>
dom res: CPU

<9 CPU, 14GB>
< 18 GB

cluster is full

User	Allocation	Dominant Share
User1	<0 CPU, 0 GB> <1 CPU, 4 GB> <2 CPU, 8 GB> <3 CPU, 12GB>	0 4/18 = 2/9 8/18 = 4/9 12/18 = 2/3
User2	<0 CPU, 0 GB> <3 CPU, 1 GB> <6 CPU, 2 GB>	0 3/9 = 1/3 6/9 = 2/3

DRF: APPROACH

Total: <9 CPU, 18 GB>

User1: <1 CPU, 4 GB> per task

<3 CPU, 12 GB> for 3 tasks

dom res: mem

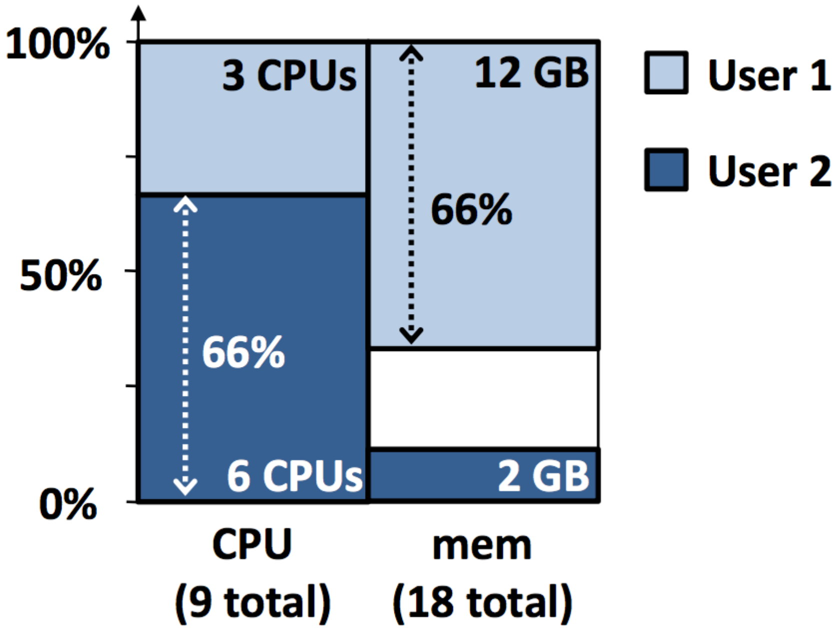
dom share: 12/18 = 2/3

User2: <3 CPU, 1 GB>

<6 CPU, 2 GB> for 2 tasks

dom res: CPU

dom share: 6/9 = 2/3



DRF ALGORITHM

Whenever there are available resources:

Schedule a task to the user with smallest dominant share

↓

new users who arrive are given resources when running tasks complete

DRF ALGORITHM

Algorithm 1 DRF pseudo-code

$R = \langle r_1, \cdots, r_m \rangle$ ▷ total resource capacities
$C = \langle c_1, \cdots, c_m \rangle$ ▷ consumed resources, initially 0
$s_i \ (i = 1..n)$ ▷ user i's dominant shares, initially 0
$U_i = \langle u_{i,1}, \cdots, u_{i,m} \rangle \ (i = 1..n)$ ▷ resources given to user i, initially 0

pick user i with lowest dominant share s_i
$D_i \leftarrow$ demand of user i's next task
if $C + D_i \leq R$ **then**
  $C = C + D_i$ ▷ update consumed vector
  $U_i = U_i + D_i$ ▷ update i's allocation vector
  $s_i = \max_{j=1}^{m}\{u_{i,j}/r_j\}$
else
  return ▷ the cluster is full
end if

picks user with lowest dominant share from S

is the cluster full

recompute dominant share

COMPARISON: ASSET FAIRNESS

Violating sharing incentive

Asset Fairness: Equalize each user's sum of resource shares

Consider total of 70 CPUs, 70 GB RAM

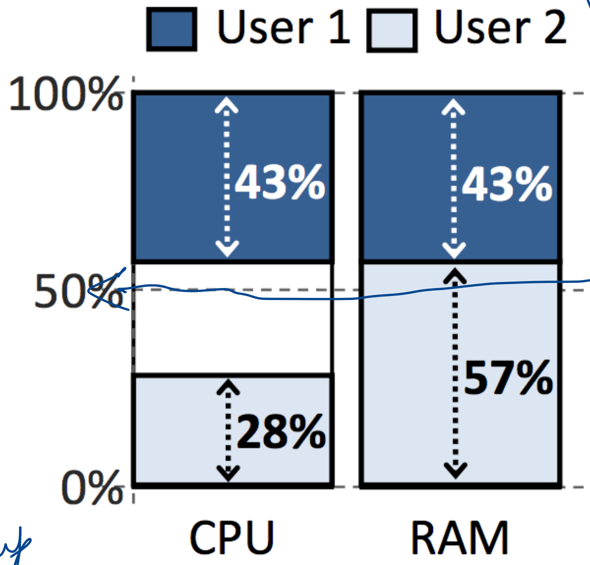
U1 needs <2 CPU, 2 GB RAM> per task

U2 needs <1 CPU, 2 GB RAM> per task

Asset Fair Allocation:

U1: 15 tasks: 30 CPU, 30 GB (Sum = 60)

U2: 20 tasks: 20 CPU, 40 GB (Sum = 60)



add up CPU & memory used by this user

COMPARISON: ASSET FAIRNESS

Asset Fairness: Equalize each user's sum of resource shares

Violates Sharing Incentive

Consider total of 70 CPUs, 70 GB RAM
U1 needs <2 CPU, 2 GB RAM> per task
U2 needs <1 CPU, 2 GB RAM> per task

Sharing incentive?
Half of the cluster is 35 CPU, 35 GB RAM
U1: 17 tasks to get 34 CPU, 34 GB RAM → 17 tasks > 15 tasks with Asset Fairness
U2:

COMPARISON: CEEI

CEEI: Competitive Equilibrium from Equal Incomes

- Each user receives initially 1/n of every resource,
- Subsequently, each user can trade resources with other users in a perfectly competitive market
- Nash solution: Maximize product of utilities across users

exchange 1GB for 1CPU

number of tasks that a user gets to run

equilibrium

COMPARISON: CEEI

Total: <9 CPU, 18 GB>　User1: <1 CPU, 4 GB>　User2: <3 CPU, 1 GB>

x tasks　　y tasks

$$\max\,(x \cdot y)$$ → product utilities

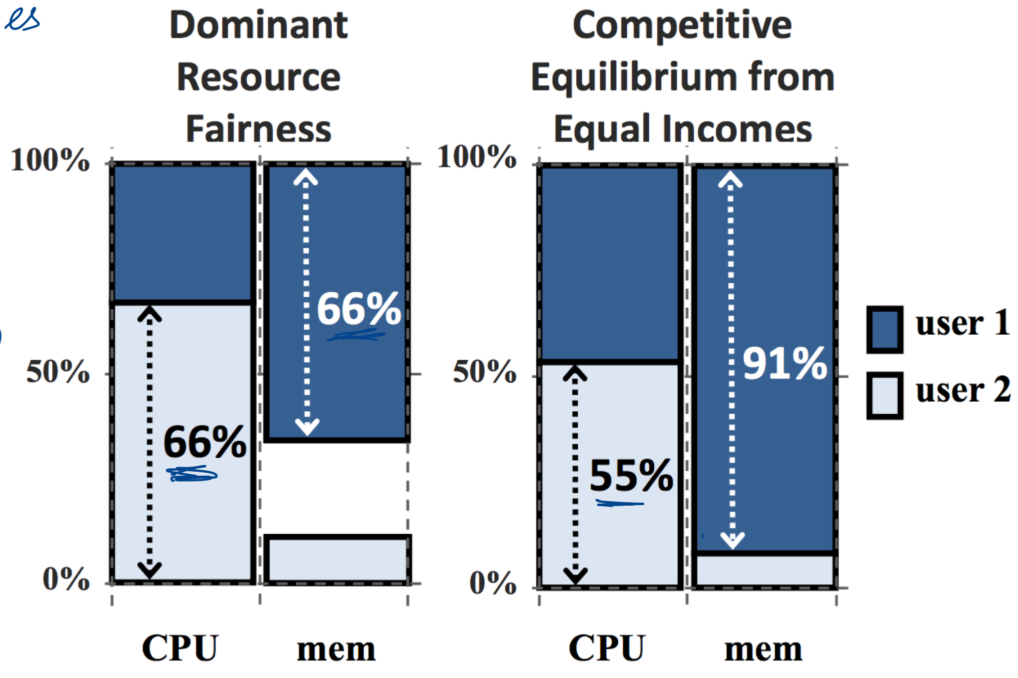
subject to

$$\begin{aligned} x + 3y &\leq 9 \quad \text{(CPU)} \\ 4x + y &\leq 18 \quad \text{(memory)} \end{aligned}$$

x = 4.05

y = 1.62

→ rounding?



CEEI: STRATEGY PROOFNESS

By lying about non dominant resource user 2 gets more dominant resource

Total: <9 CPU, 18 GB>

User2 Before:
CEEI: 55% CPU, 9% mem

Total: <9 CPU, 18 GB>
User1: <1 CPU, 4 GB>
~~User2: <3 CPU, 1 GB>~~
User2: <3 CPU, 2 GB>

lies about memory req

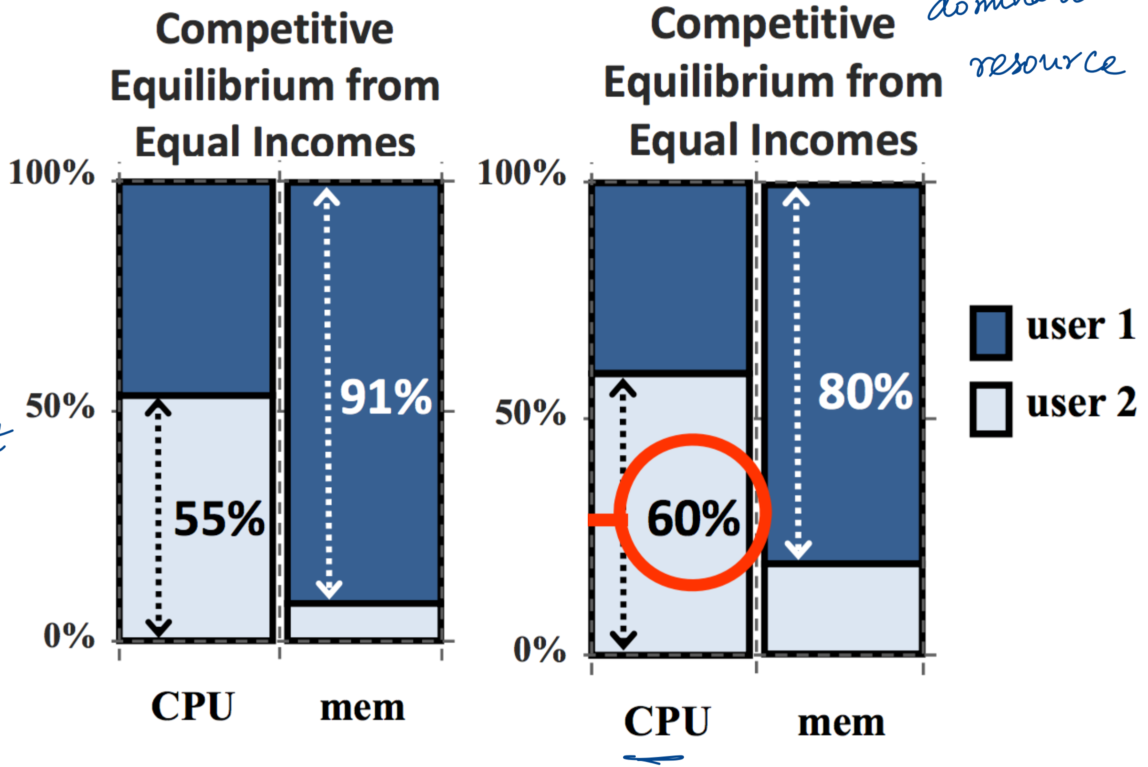
max $x \cdot y$

$x + 3y \le 9$

$4x + 2y \le 18$

$x = 3.6$

$y = 1.8$



COMPARISON

Property	Allocation Policy: Asset	CEEI	DRF
Sharing Incentive		✓	✓
Strategy-proofness	✓		✓
Envy-freeness	✓	✓	✓
Pareto efficiency	✓	✓	✓
Single Resource Fairness	✓	✓	✓
Bottleneck Fairness		✓	✓
Population Monotonicity	✓		✓
Resource Monotonicity			✗

as you add more resources to the cluster

Table 2: Properties of Asset Fairness, CEEI and DRF.

SUMMARY

DRF: Dominant Resource Fairness

Allocation policy for scheduling

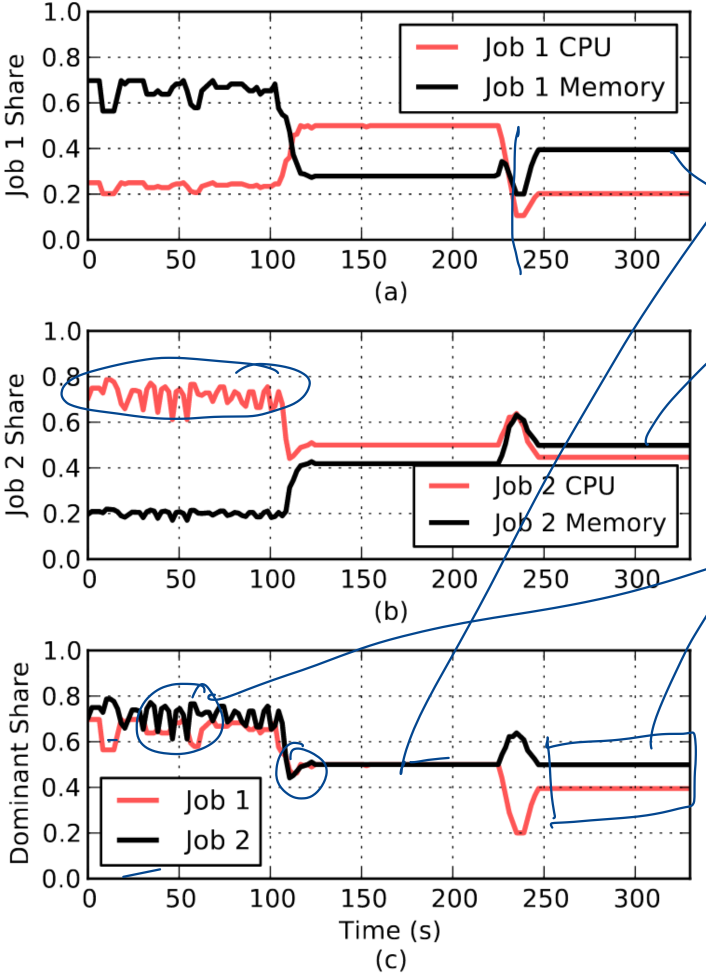
Provides multi-resource fairness

Ensures sharing incentive, strategy proofness

↳ DRF was integrated Mesos, YARN

DISCUSSION

https://forms.gle/75faGZ4quQgWSYRQ8



CPU is dominant share for both jobs

Mem dominant resource fragmentation / discrete allocations

Happens when jobs are very diff.

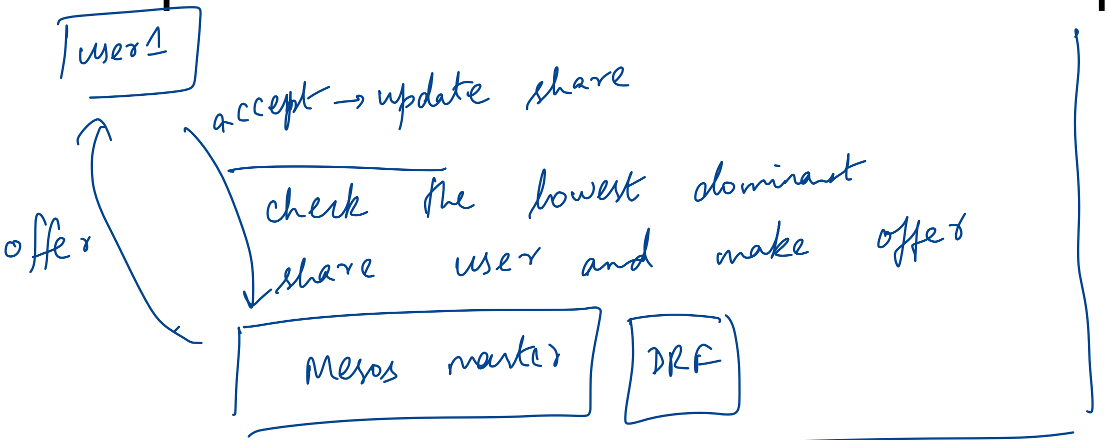
Both dominant shares are ~0.7

DRF quickly handles change in job requirements

↳ ~10s from this figure

→ task length → Pre-emption vs. waiting for tasks

What could be one workload / cluster scenario where DRF implemented on Mesos will NOT be optimal?



- DRF can only adjust shares after tasks finish
- Resources that fail back and come back
  frequent changes to running jobs

- Constant stream of "small" jobs starve the larger jobs

- <8 CPU, 1GB>
  <5 CPU, 4GB>

  · Resource demands that flip very often
  → might not converge

- Locality.

NEXT STEPS

Next class: Machine Learning Schedulers