Resource sharing in data centers

Thomas Bonald & Jim Roberts

Workshop on
Performance and Control of Large-Scale Networks

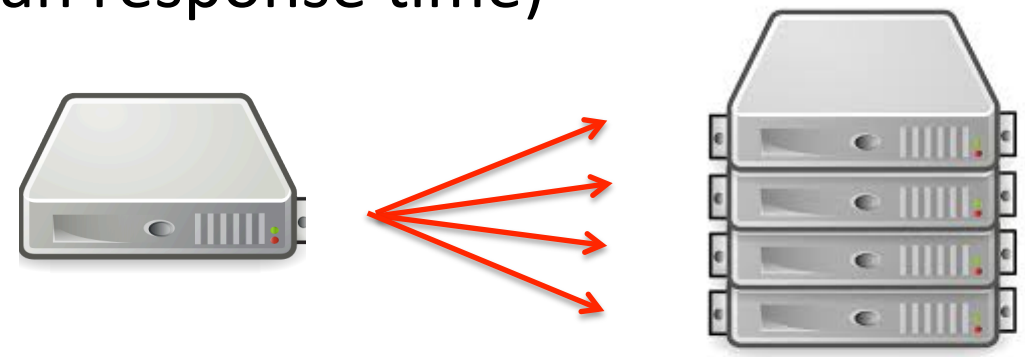
Eindhoven, 2014

Introduction

Key features of (big) data centers:

- massively parallel jobs (à la MapReduce)
- heterogeneous requirements (CPU, RAM, I/O)
- high power consumption

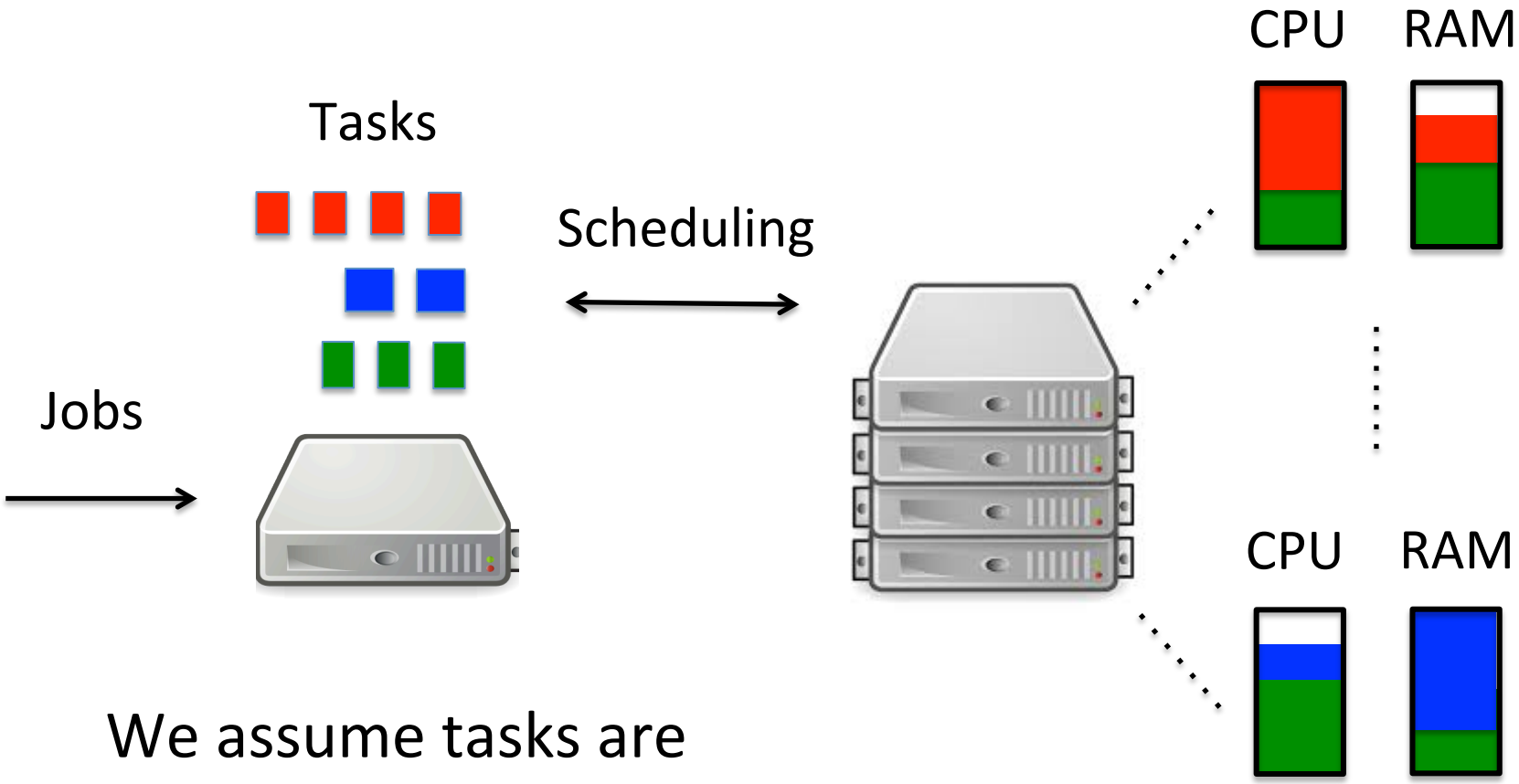
Strong impact of job scheduling on performance (e.g., mean response time)



Outline

- Model
- Objectives
- Algorithms
- Performance
- Conclusion

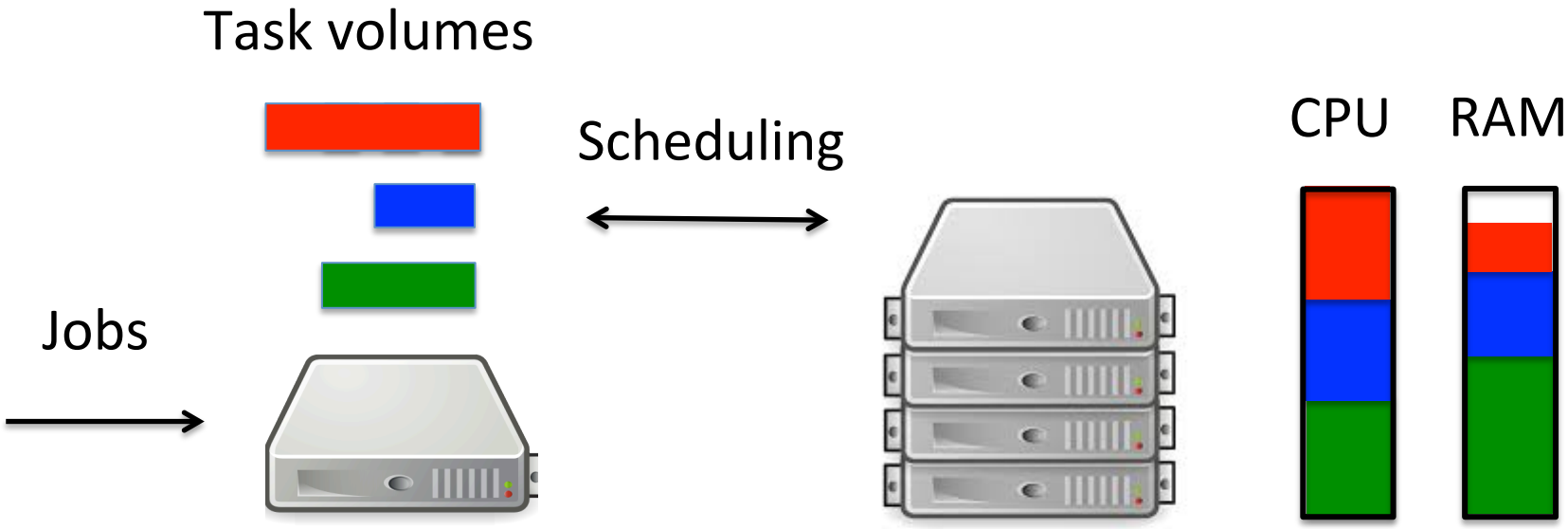
Model



We assume tasks are

- independent,
- homogeneous per job,
- infinitely small

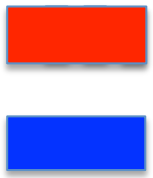
Resource pooling



Capacity constraints:

$$\varphi A \leq C \quad \Longleftrightarrow \quad \varphi a \leq 1$$

Example

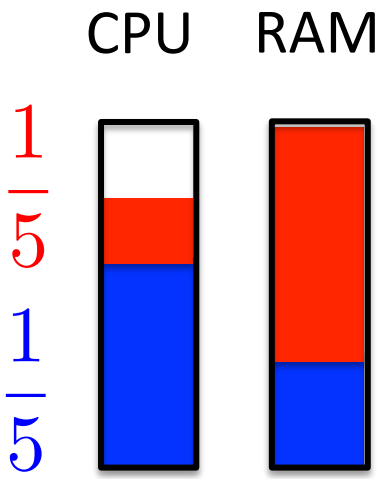


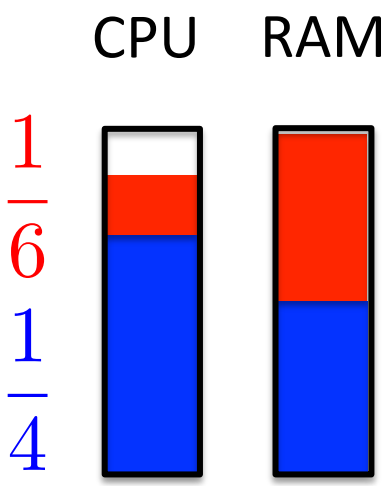
CPU	RAM
1	3
3	2

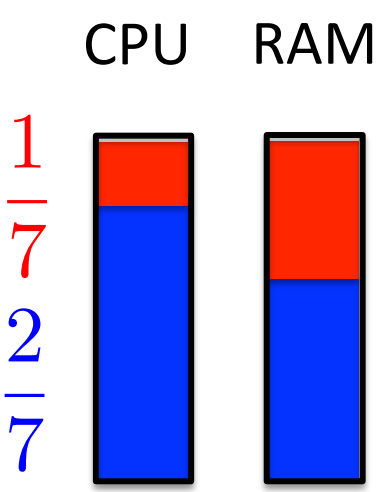
$$\iff$$

$$\varphi a \leq 1$$

$$\varphi_1 + 3\varphi_2 \leq 1$$
$$3\varphi_1 + 2\varphi_2 \leq 1$$

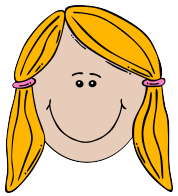
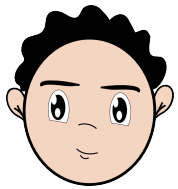






Buffet sharing

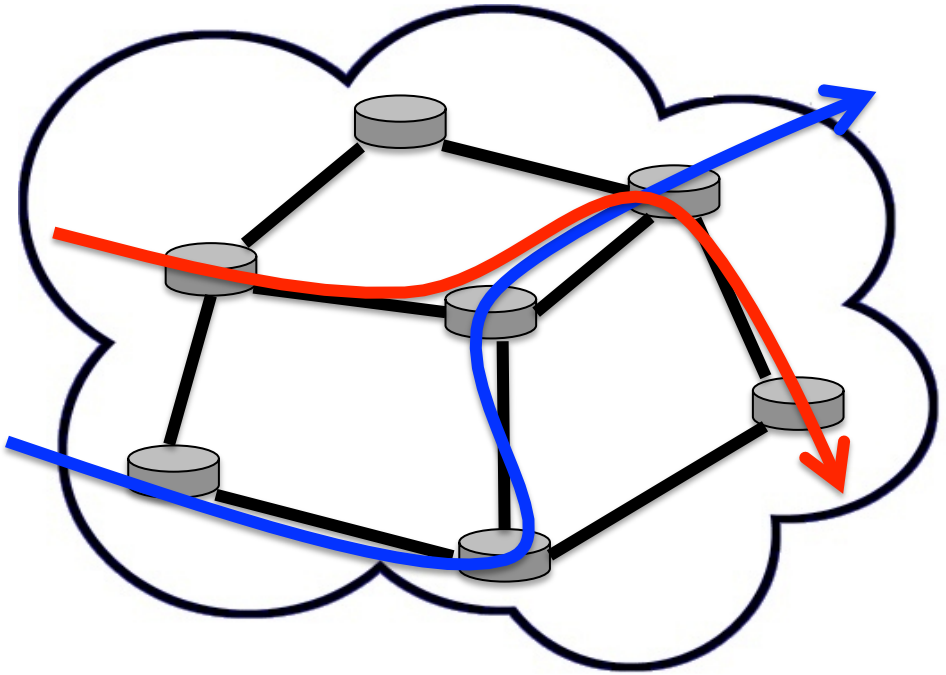


2	1	3
3	4	1

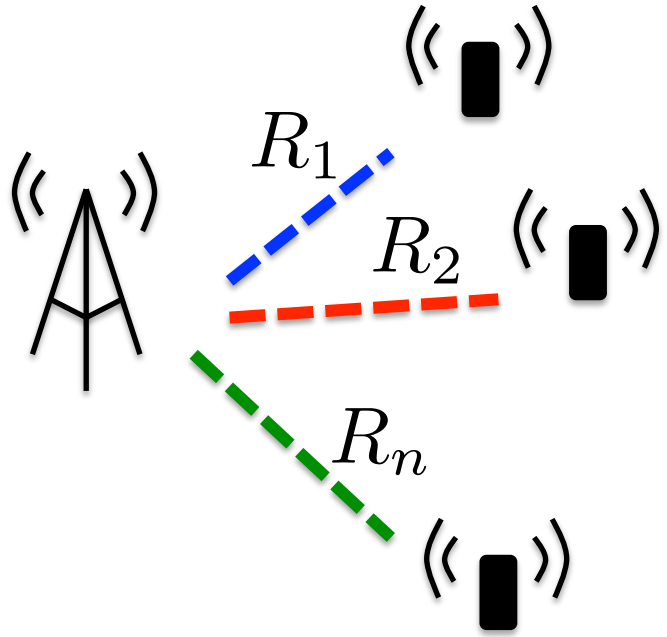
Leontief utilities

Data networks



Capacity constraints: $\varphi A \leq C$

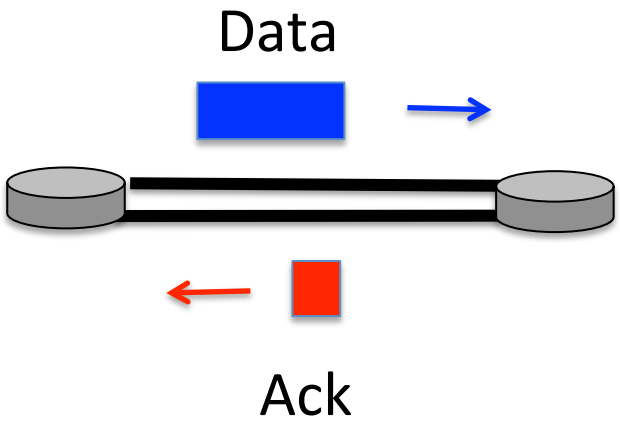
Wireless access



Capacity constraint:

$$\sum_{i=1}^{n} \frac{\varphi_i}{R_i} \leq 1$$

TCP Acks



Capacity constraints:

$$\sum_{i \in \mathrm{Data}_j} \varphi_i + \sum_{i \in \mathrm{Ack}_j} \varphi_i r_i \leq C_j$$

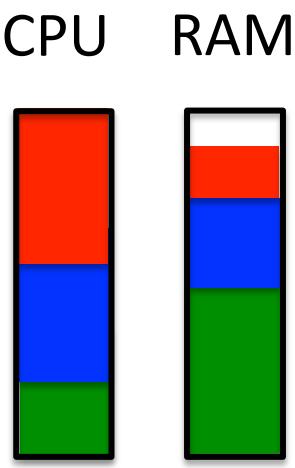
Outline

- Model
- Objectives
- Algorithms
- Performance
- Conclusion

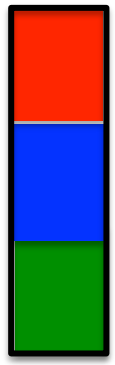
Some desirable properties

The allocation should be...

- Pareto efficient
- sharing incentive
- envy free
- scale invariant



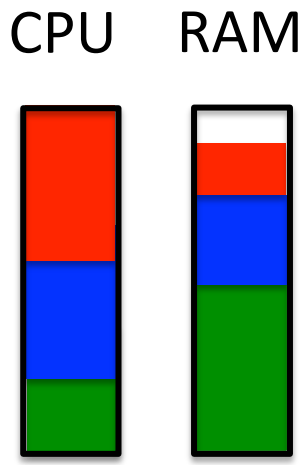
Note: Equal shares for a single resource

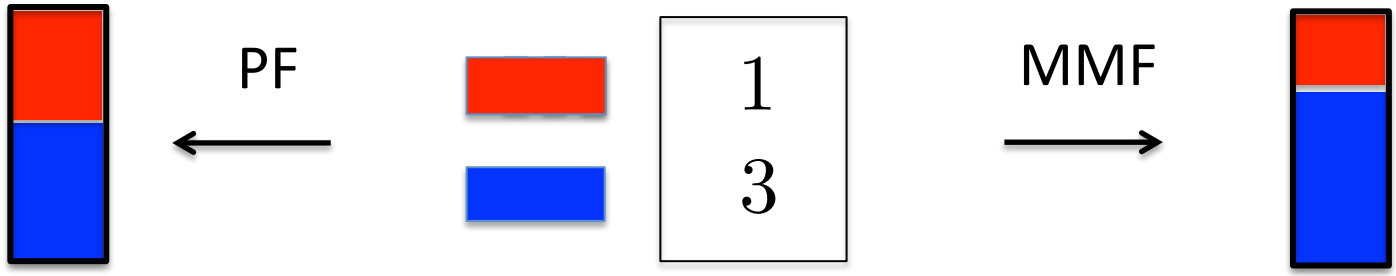


Some objectives

The allocation could be...

- proportional fair
- max-min fair
- dominant-resource fair
- bottleneck-max fair





Dominant resource fairness

- Weighted max-min fairness

$$w_i = \frac{1}{\max_j a_{ij}}$$

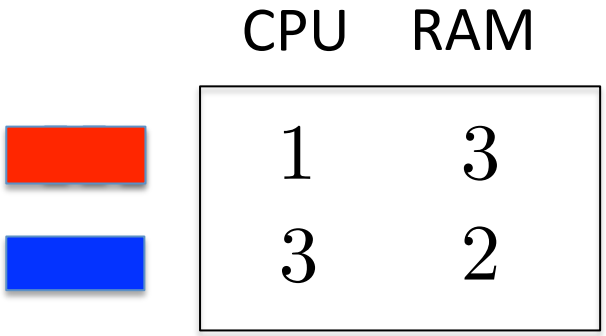
dominant resource of job i

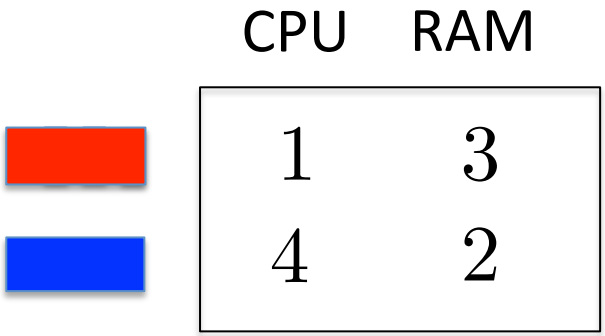
Ghodsi et. al. 2011

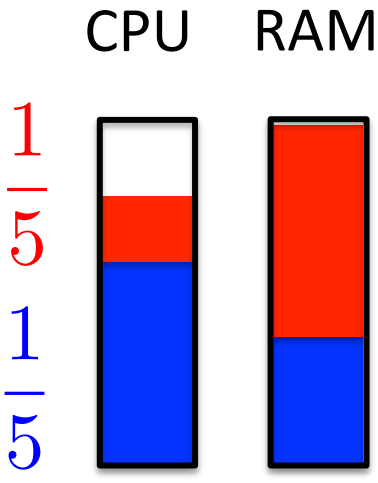
Notes:

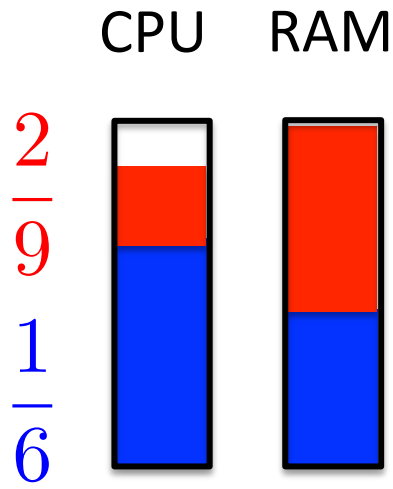
- strategy proof
- implemented in Hadoop
- only one saturated resource in general

Example









Proportional fairness

$$\max \sum_{i=1}^{n} \log \varphi_i \quad \text{with } \varphi a \leq 1$$

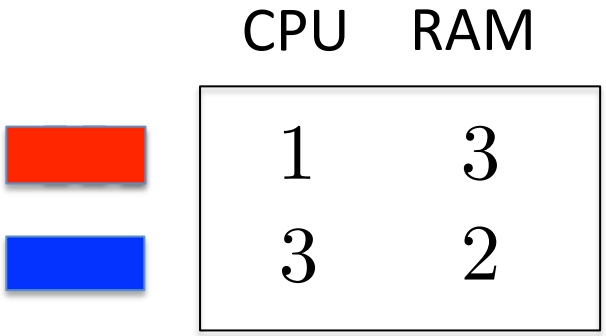
Notes:

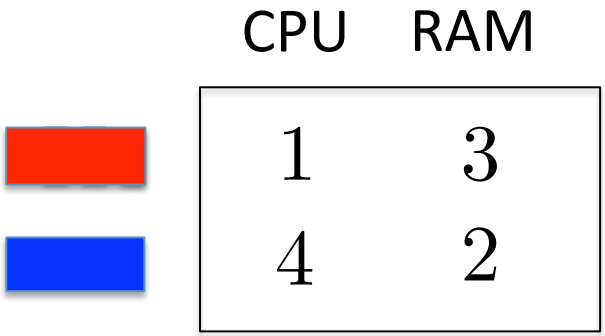
- Equal shares for a single bottleneck
- For two resources,

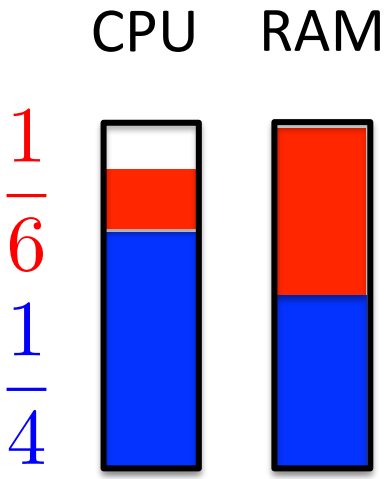
$$\sum_{i=1}^{n} \varphi_i a_{i1} = 1 \iff \sum_{i=1}^{n} \frac{a_{i1}}{a_{i2}} \geq n$$

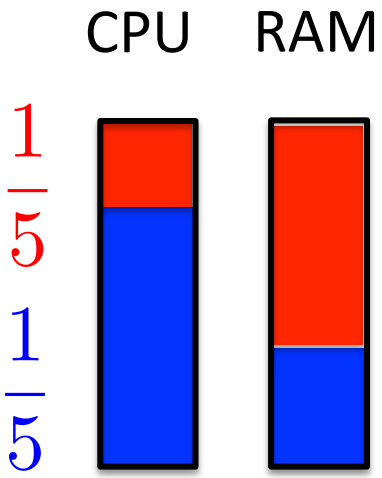
$$\sum_{i=1}^{n} \varphi_i a_{i2} = 1 \iff \sum_{i=1}^{n} \frac{a_{i2}}{a_{i1}} \geq n$$

Example









Bottleneck-max fairness

Definition: Each job has the maximum resource share on at least one bottleneck

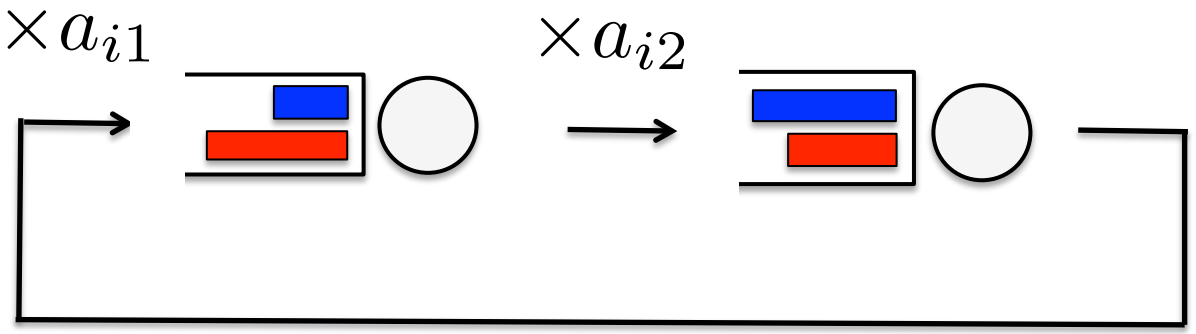
Notes:

- Equal shares for a single bottleneck
- For two resources,

$$\sum_{i=1}^{n} \varphi_i a_{i1} = 1 \iff \sum_{i=1}^{n} \frac{a_{i1}}{a_{i2}} \geq n$$

$$\sum_{i=1}^{n} \varphi_i a_{i2} = 1 \iff \sum_{i=1}^{n} \frac{a_{i2}}{a_{i1}} \geq n$$

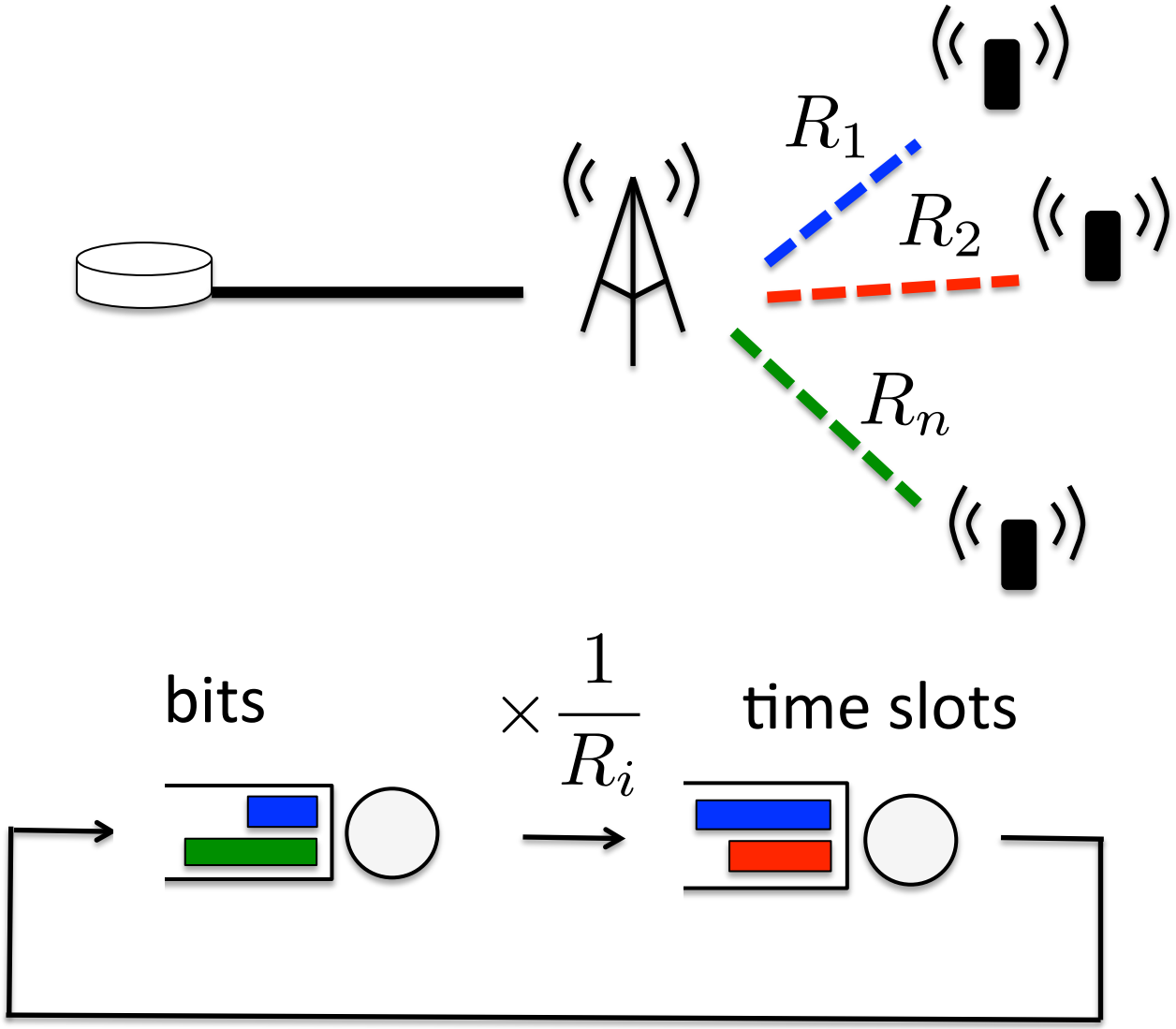
Existence



$$\dot{Q}_{i1} = R_{i2}\frac{a_{i1}}{a_{i2}} - R_{i1}$$

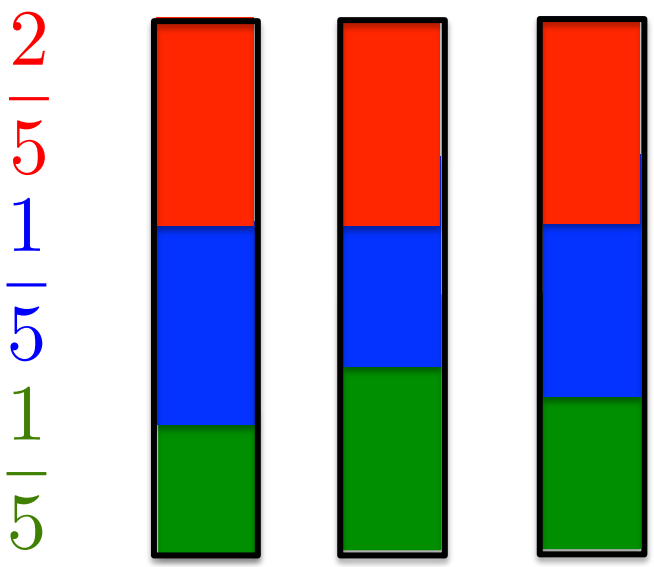
$R_{i1} =$ max-min fair rates
with rate limits $R_{i2}\dfrac{a_{i1}}{a_{i2}}$ if $Q_{i1} = 0$

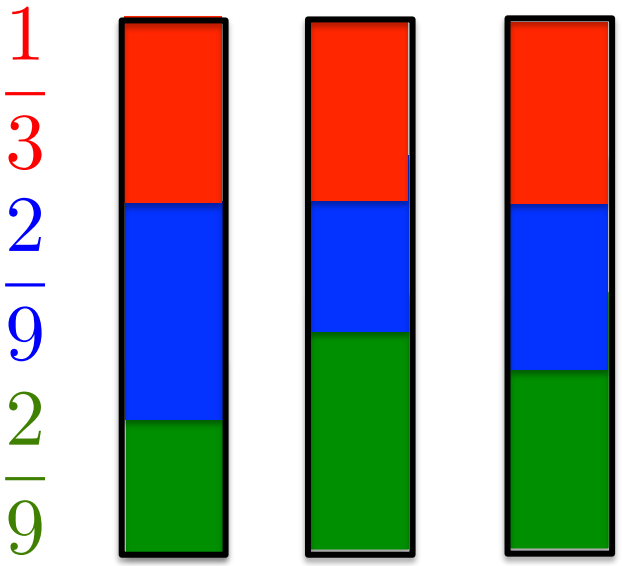
Example



Non-uniqueness

1	1	1
2	1	3/2
1	2	3/2

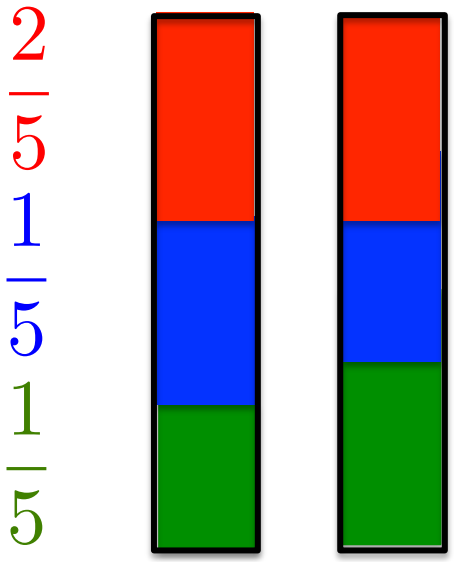




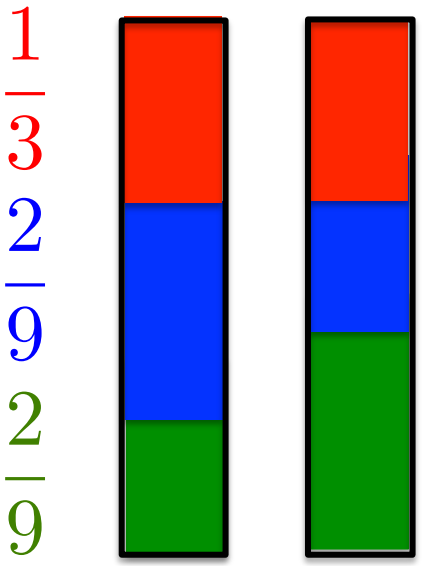
Difference with PF

$$\begin{matrix} 1 & 1 \\ 2 & 1 \\ 1 & 2 \end{matrix}$$

BMF



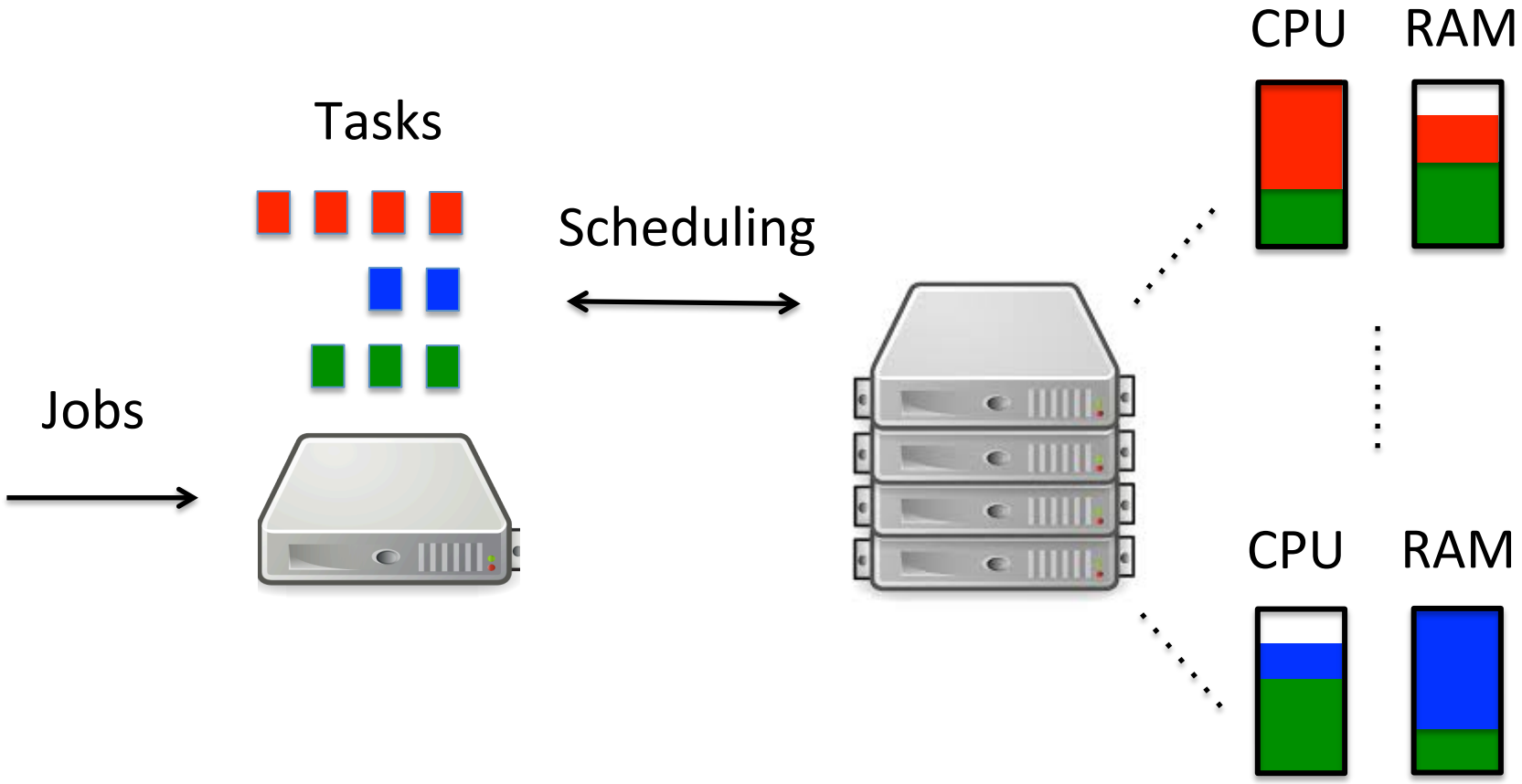
PF



Outline

- Model
- Objectives
- Algorithms
- Performance
- Conclusion

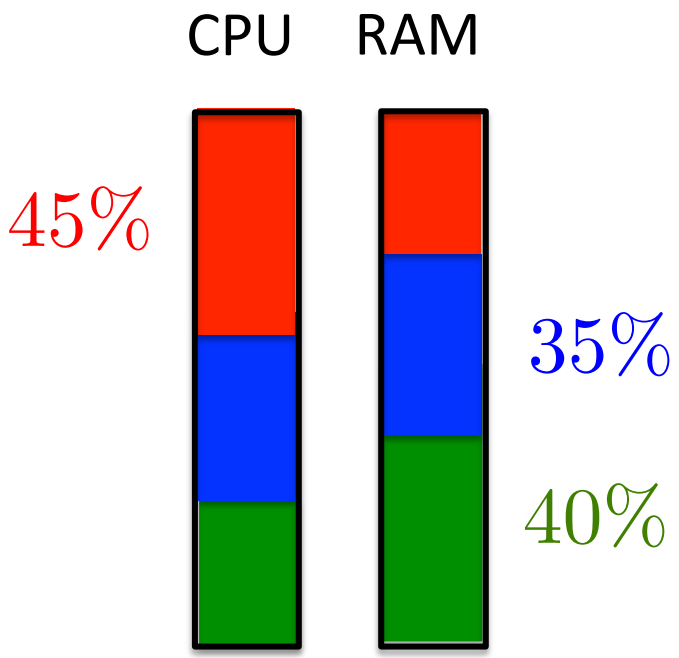
Serving the "most deprived" job



Note: Head-of-line blocking

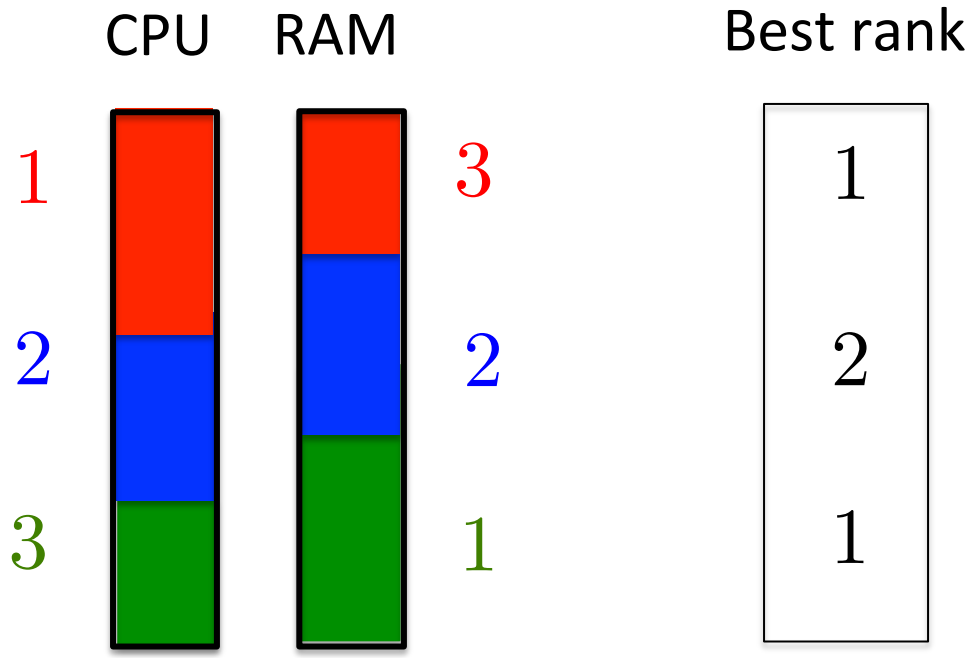
Dominant resource fairness

Algorithm: Serve the job with the smallest dominant resource share



Bottleneck-max fairness

Algorithm: Serve the job with the lowest best rank of bottleneck shares

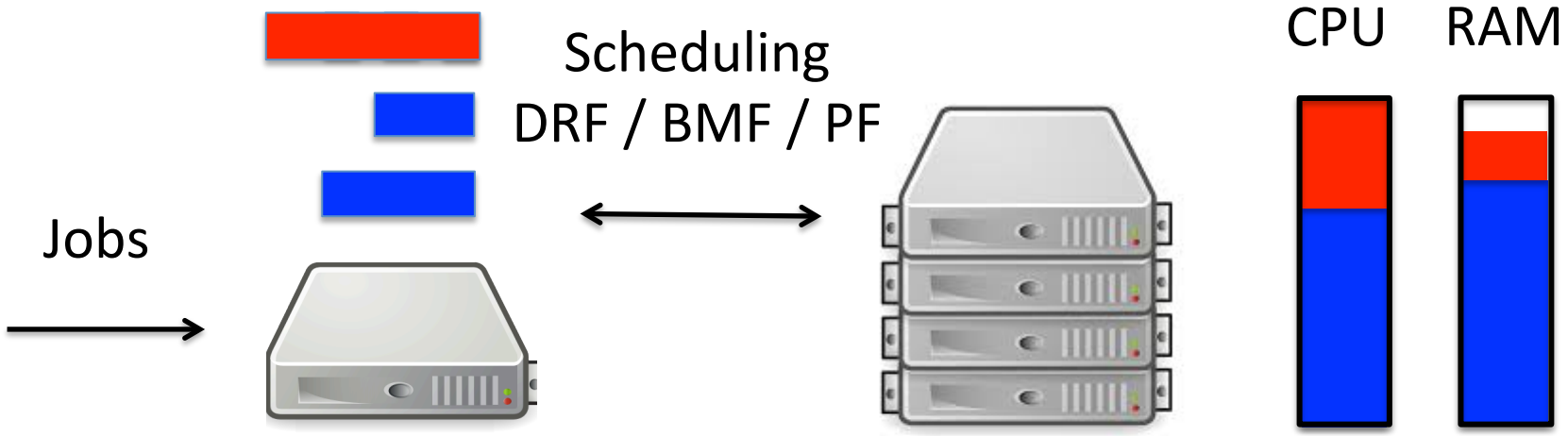


Outline

- Model
- Objectives
- Algorithms
- Performance
- Conclusion

Accounting for random traffic

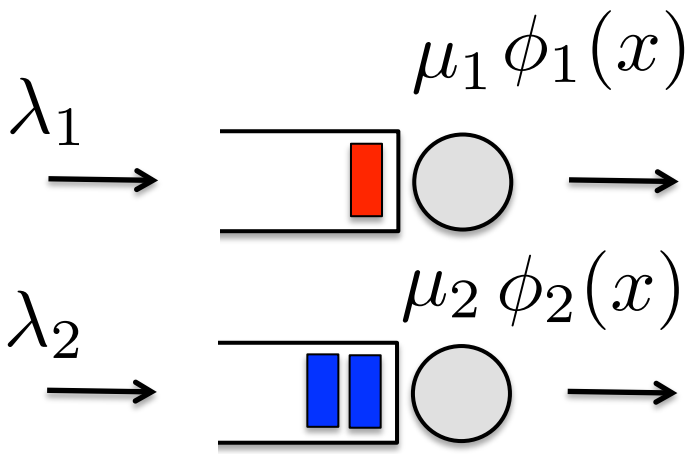
N classes of jobs

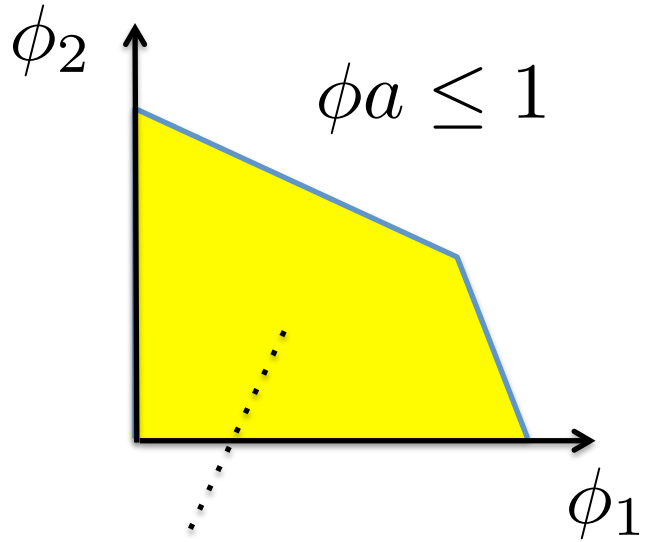


Capacity constraints:

$$\phi a \leq 1$$

A coupled queuing system





Capacity region

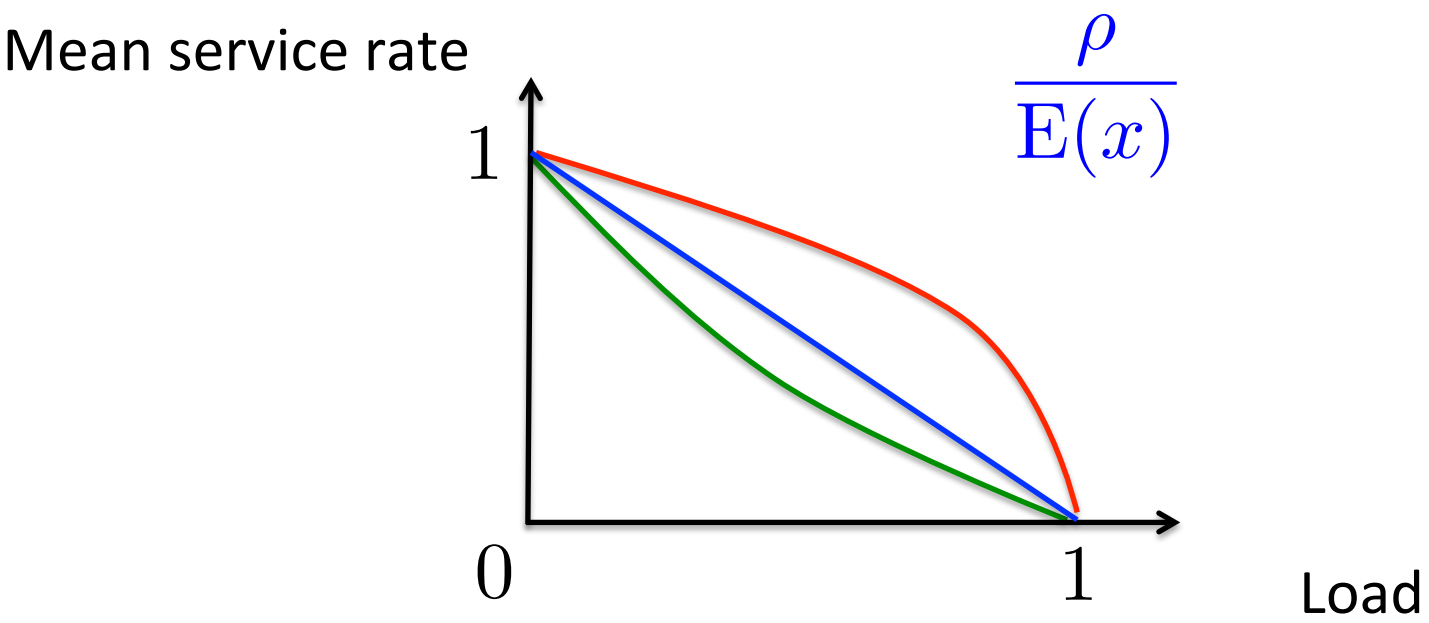
Stability condition:

$$\rho a < 1 \qquad \rho = \frac{\lambda}{\mu}$$

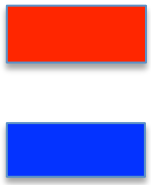
Performance metric

Mean service rate =

- Mean number of tasks of an active job
- Ratio of mean job size to mean job duration



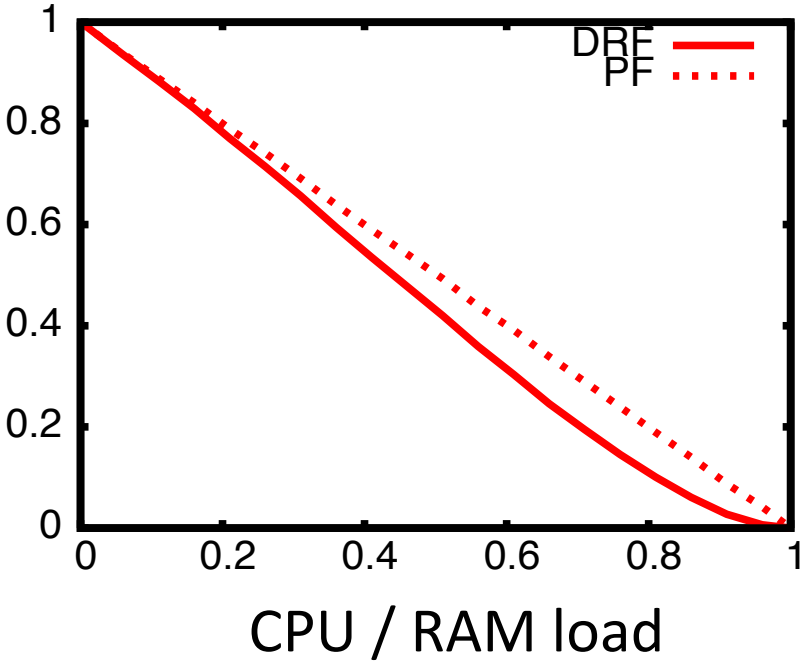
Example



CPU	RAM
10	1
1	10

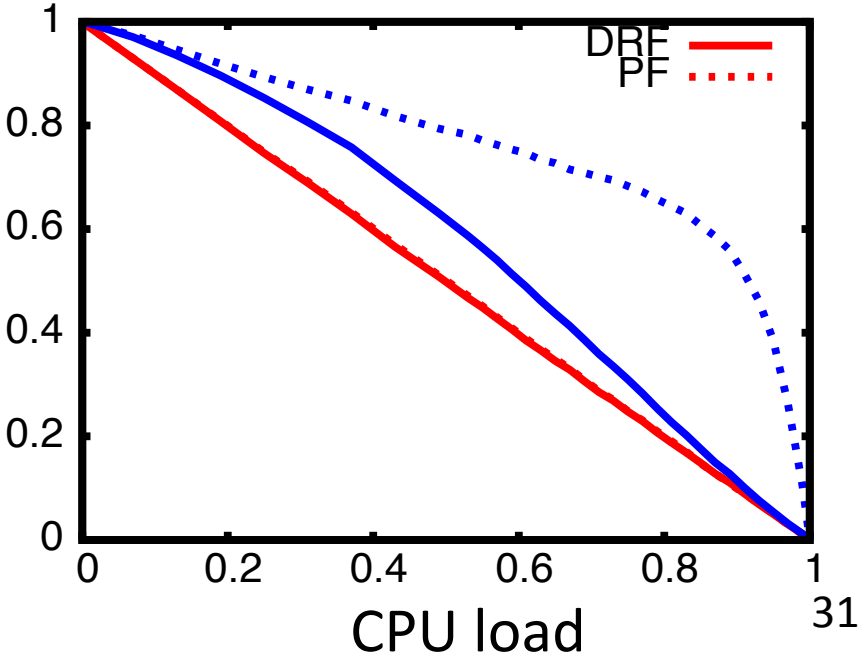
Balanced load

$\rho_1 = \rho_2$

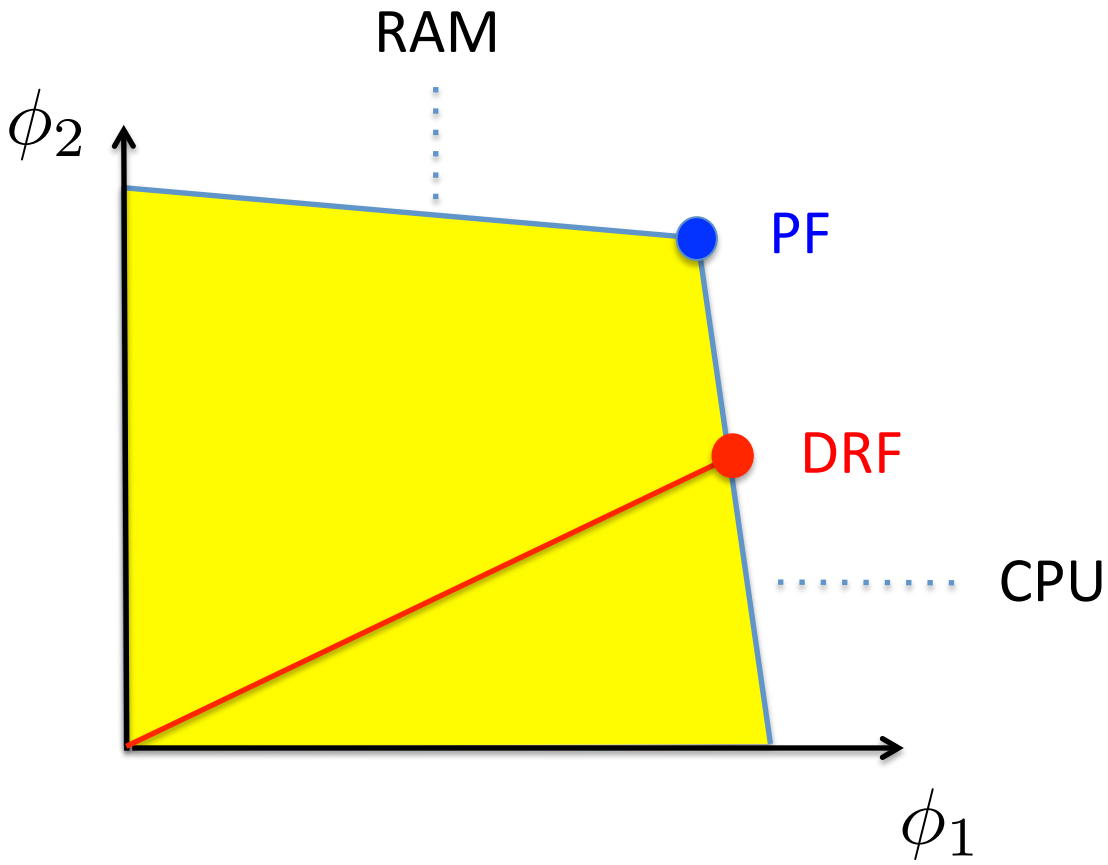


Unbalanced load

$\rho_1 = 3\rho_2$

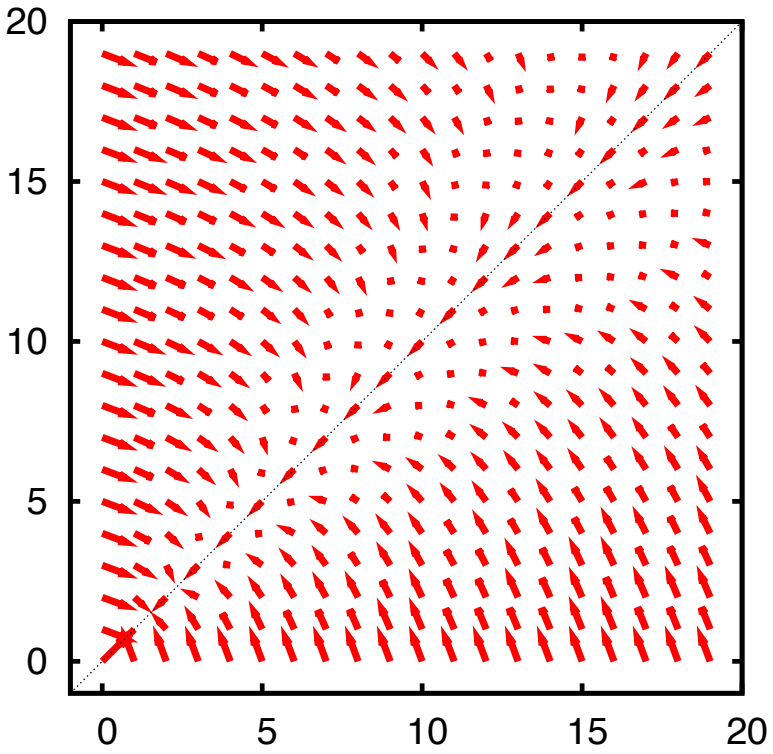


Capacity region

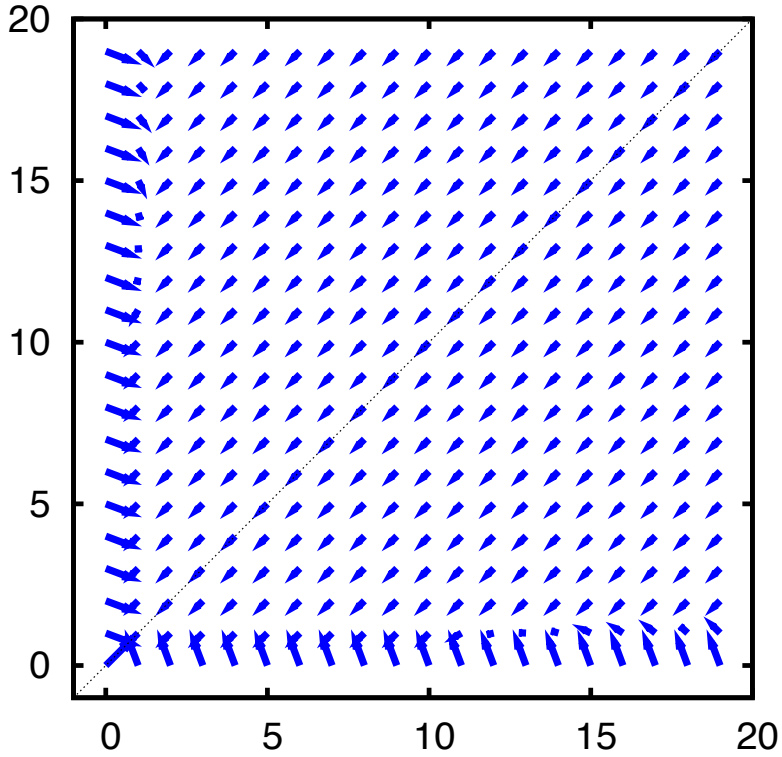


Drift for balanced load

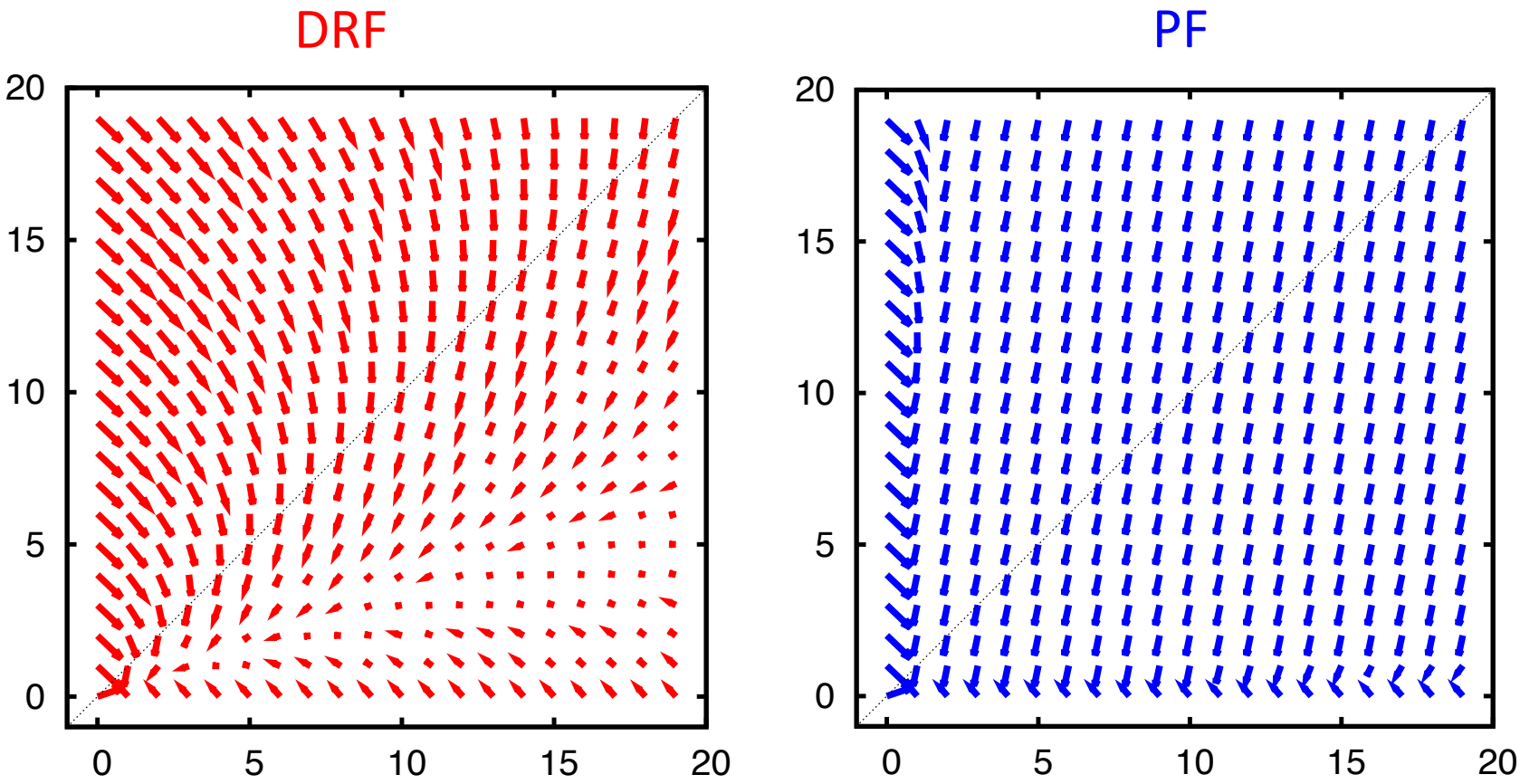
DRF



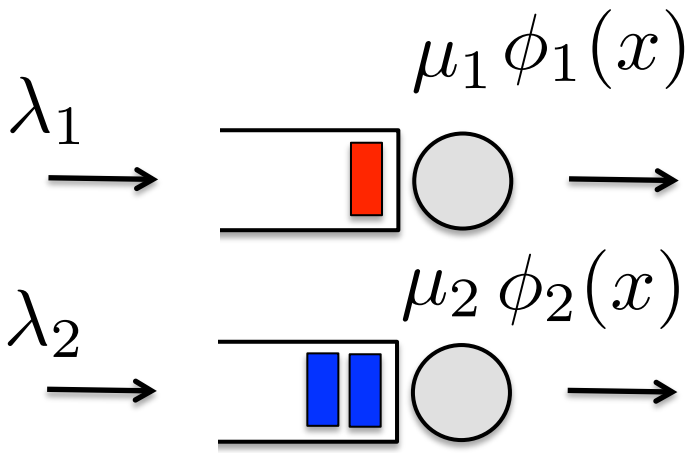
PF

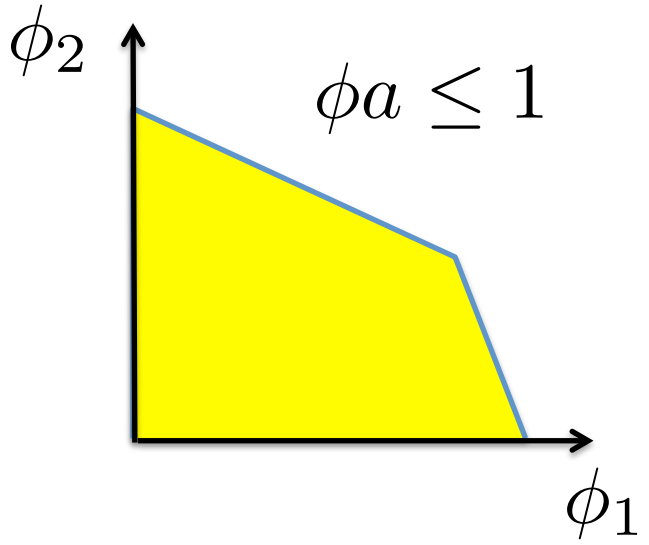


Drift for unbalanced load



Balanced fairness





$$\begin{cases} \phi_1(x) = \dfrac{\Phi(x - e_1)}{\Phi(x)} \\ \phi_2(x) = \dfrac{\Phi(x - e_2)}{\Phi(x)} \end{cases}$$

$$\pi(x) = \pi(0)\Phi(x)\rho_1^{x_1}\rho_2^{x_2}$$

A useful bound for BF

Mean duration of job i

$$\leq 1 + \sum_j a_{ij} \frac{\varrho_j}{1 - \varrho_j}$$

⋮

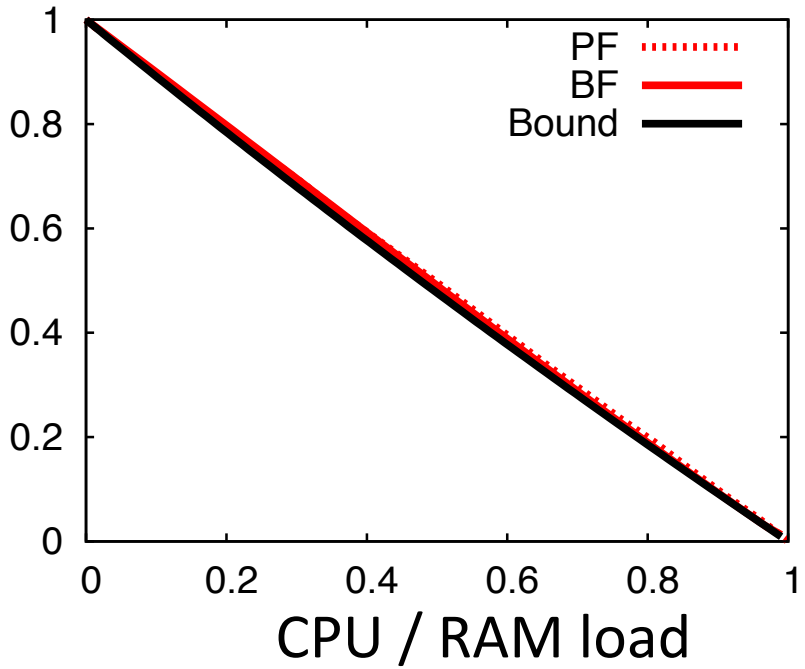
Load of resource j

Example

CPU	RAM
10	1
1	10

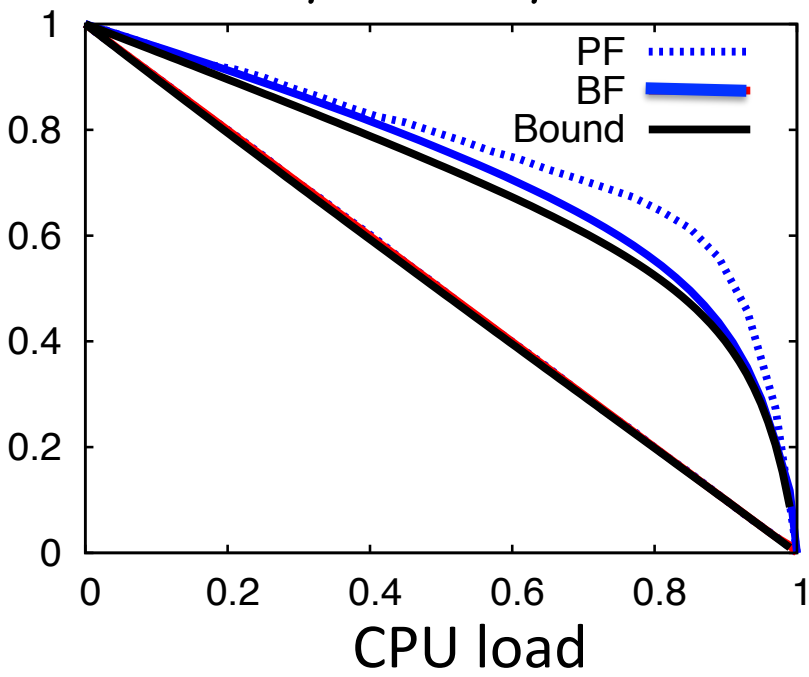
Balanced load

$\rho_1 = \rho_2$

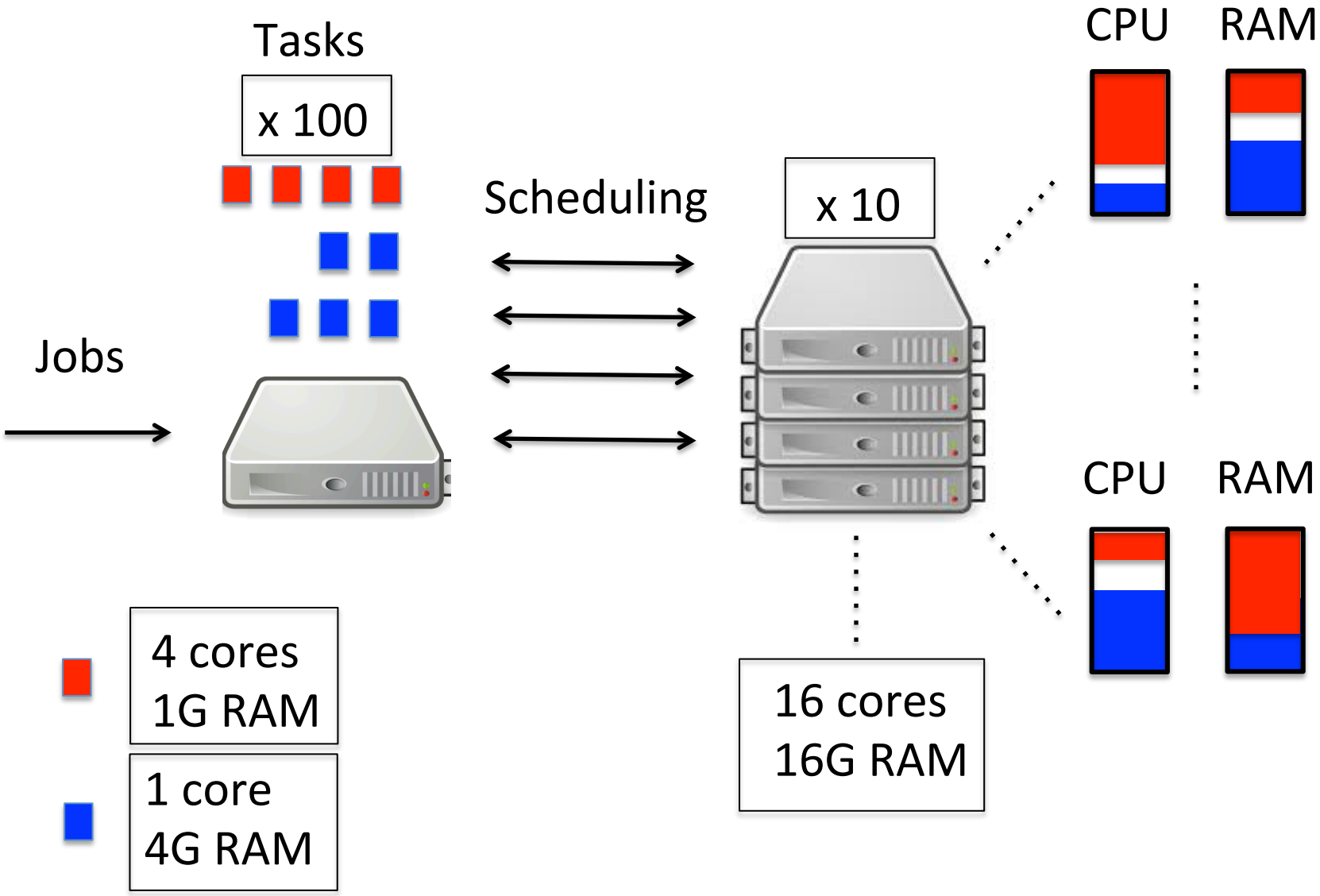


Unbalanced load

$\rho_1 = 3\rho_2$



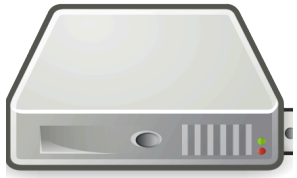
Validation


Tasks
x 100
Jobs
Scheduling
x 10
CPU
RAM
CPU
RAM
4 cores
1G RAM
1 core
4G RAM
16 cores
16G RAM

Results

CPU	RAM
4	1
1	4

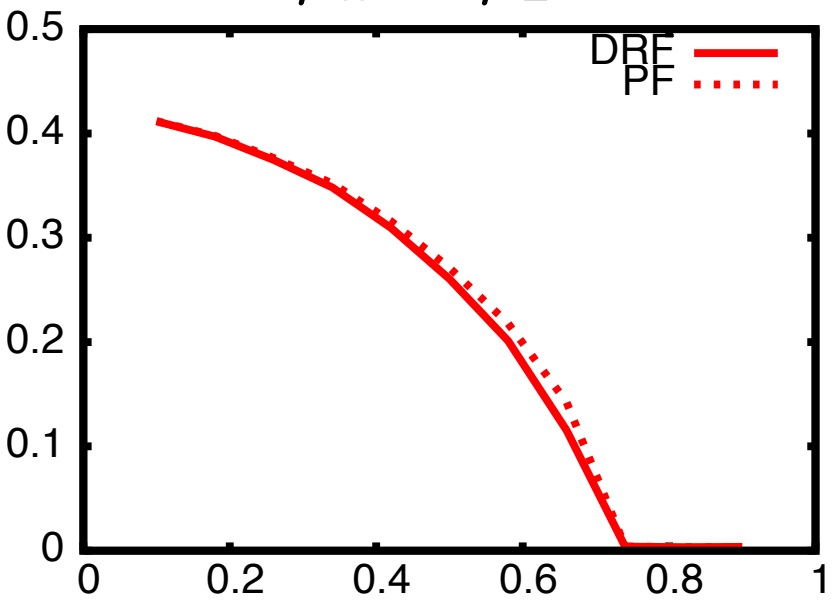
16 cores
16G



x 10

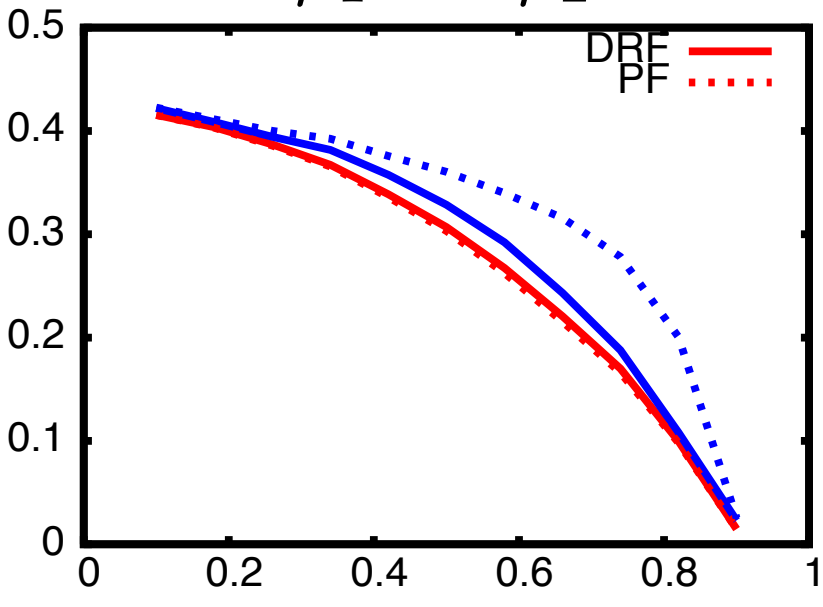
Balanced load

$\rho_1 = \rho_2$



Unbalanced load

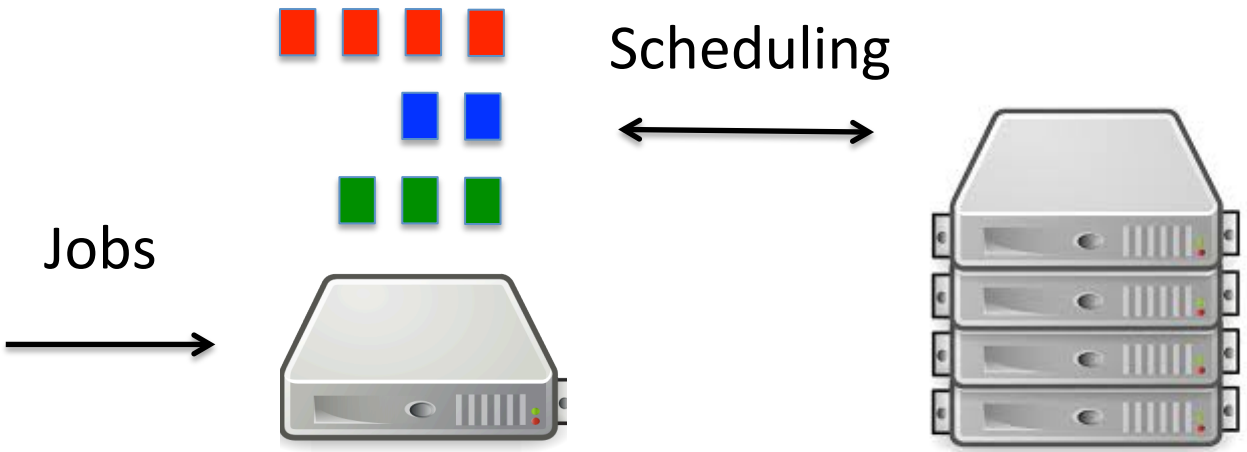
$\rho_1 = 3\rho_2$



Summary

Arguments for bottleneck-max fairness

- performance similar to PF, much better than DRF
- simple algorithm based on "best rank"



Future work

Conjectures

- Existence of BMF
- Stability of BMF
- PF = BF in heavy traffic

Extensions

- Load balancing
- Bin packing
- Heterogeneous jobs