Προχωρημένη Κατανεμημένη Υπολογιστική

ΗΥ623

Διδάσκων –
Δημήτριος Κατσαρός

@ Τμ. ΜΗΜΥ
Πανεπιστήμιο Θεσσαλίας

Διάλεξη 4η

Dominant Resource Fairness – DRF

Fair allocation of multiple resource types

Introduction

- Resource allocation is a key building block of any shared computer system
- One of the most popular allocation policies proposed so far has been max-min fairness, which maximizes the minimum allocation received by a user in the system
- The focus has so far been primarily on a single resource type, or allocate resources at the level of fixed-size partitions of the nodes, called slots

Introduction

- The problem of fair allocation of multiple types of resources to users with heterogeneous demands
- Dominant Resource Fairness (DRF), a generalization of max-min fairness for multiple resources
- For example: if user A runs CPU-heavy tasks and user B runs memory-heavy tasks, DRF attempts to equalize user A's share of CPUs with user B's share of memory
- The strength of DRF lies in the properties it satisfies

Motivation

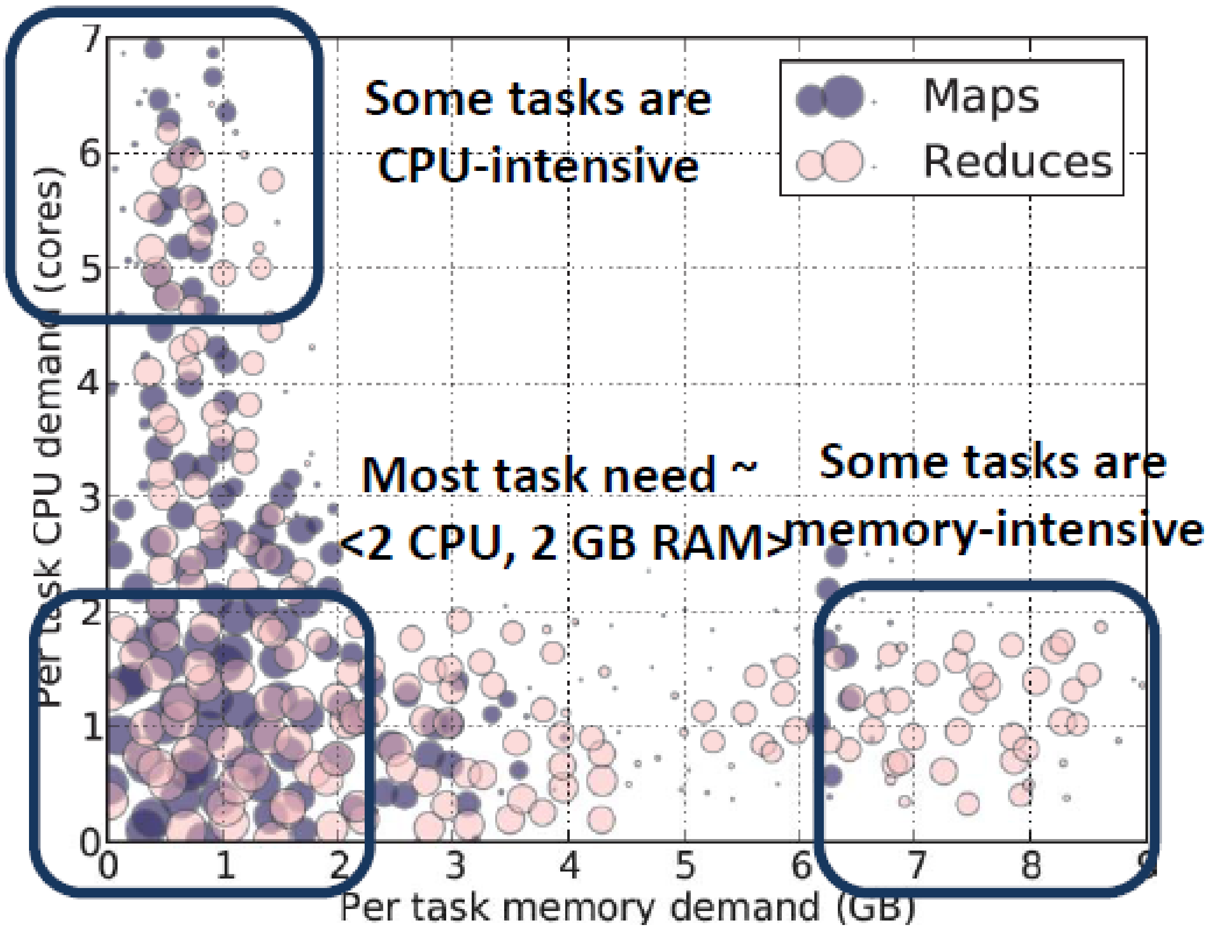


Figure 1: CPU and memory demands of tasks in a 2000-node Hadoop cluster at Facebook over one month (October 2010). Each bubble's size is logarithmic in the number of tasks in its region.

Motivation

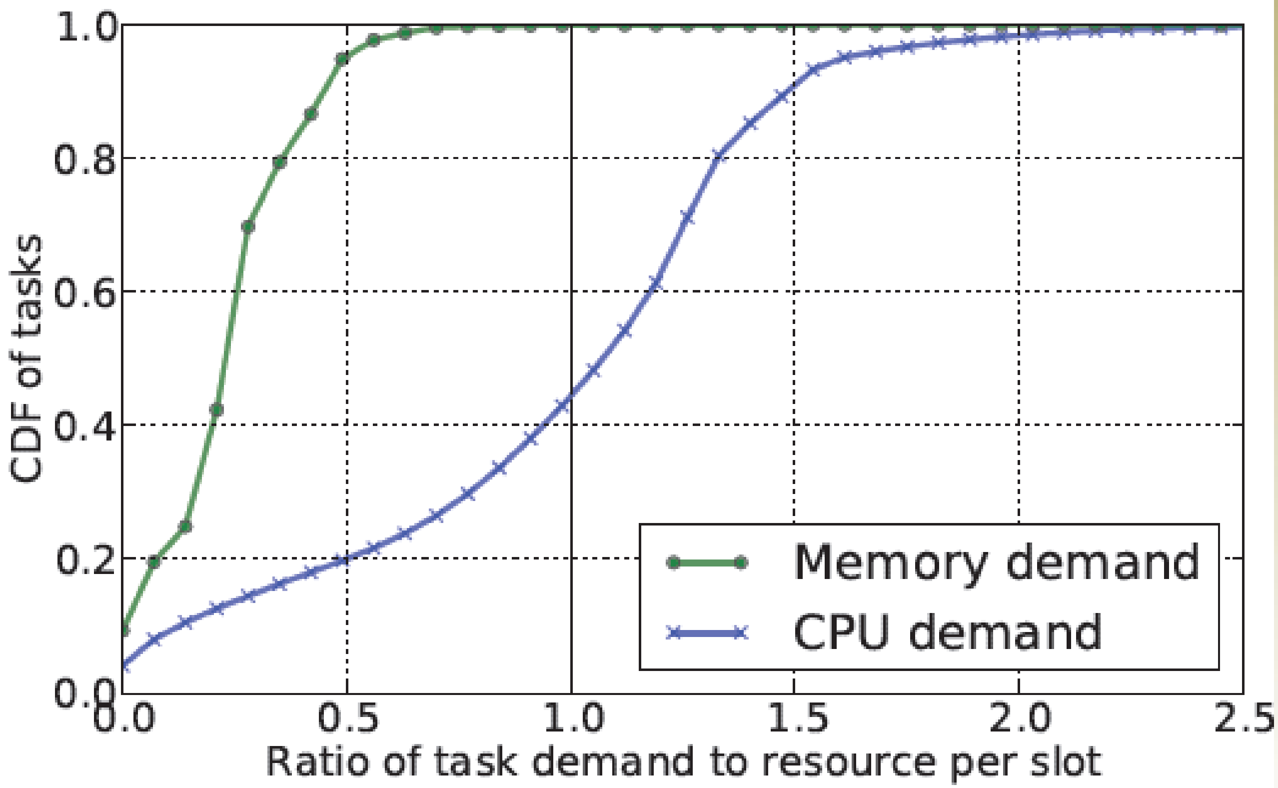


Figure 2: CDF of demand to slot ratio in a 2000-node cluster at Facebook over a one month period (October 2010). A demand to slot ratio of 2.0 represents a task that requires twice as much CPU (or memory) than the slot CPU (or memory) size.

Motivation

- Figure1 shows that though the majority of tasks are CPU-heavy, there exist tasks that are memory heavy as well, especially for reduce operations
- Figure 2 shows that most of the tasks either underutilize or overutilize some of their slot resources

Allocation Properties

- 1. ***Sharing incentive*:**

  Each user should be better off sharing the cluster, than exclusively using her own partition of the cluster

  Consider a cluster with identical nodes and n users. Then a user should not be able to allocate more tasks in a cluster partition consisting of 1/n of all resources

Allocation Properties

- **2. Strategy-proofness:**

  Users should not be able to benefit by lying about their resource demands. This provides incentive compatibility, as a user cannot improve her allocation by lying

Allocation Properties

- **3. Envy-freeness:**
  A user should not prefer the allocation of another user

- **4. Pareto efficiency:**
  It should not be possible to increase the allocation of a user without decreasing the allocation of at least another user

Allocation Properties

- Strategy-proofness and sharing incentive properties are of special importance in datacenter environments
- For example, one of Yahoo!'s Hadoop MapReduce datacenters has different numbers of slots for map and reduce tasks. (strategy-proofness)
- Another big search company provided dedicated machines for jobs only if the users could guarantee high utilization. (strategy-proofness)

Some other nice-to-have properties

- **Single resource fairness:**

  For a single resource, the solution should reduce to max-min fairness

- **Bottleneck fairness:**

  If there is one resource that is percent-wise demanded most of by every user, then the solution should reduce to max-min fairness for that resource

Some other nice-to-have properties

- **Population monotonicity:**

  When a user leaves the system and relinquishes her resources, none of the allocations of the remaining users should decrease

- **Resource monotonicity:**

  If more resources are added to the system, none of the allocations of the existing users should decrease

Dominant Resource Fairness (DRF)

- For every user, DRF computes the share of each resource allocated to that user
- The maximum among all shares of a user is called that user's **dominant share**
- The resource corresponding to the dominant share is called the **dominant resource**
- <u>DRF equalizes the dominant shares of the users</u>
- We assume a "pool" of resources, i.e., *all-in-one*

An Example

- A system with of 9 CPUs, 18 GB RAM
- Two users, where user A runs tasks with demand vector <1 CPU, 4 GB>, and user B runs tasks with demand vector <3 CPUs, 1 GB> each
- Dominant share:
  A:2/9 (memory) B:1/3 (CPU)
- With this allocation, each user ends up with the same dominant share, i.e., user A gets 2/3 of RAM, while user B gets 2/3 of the CPUs

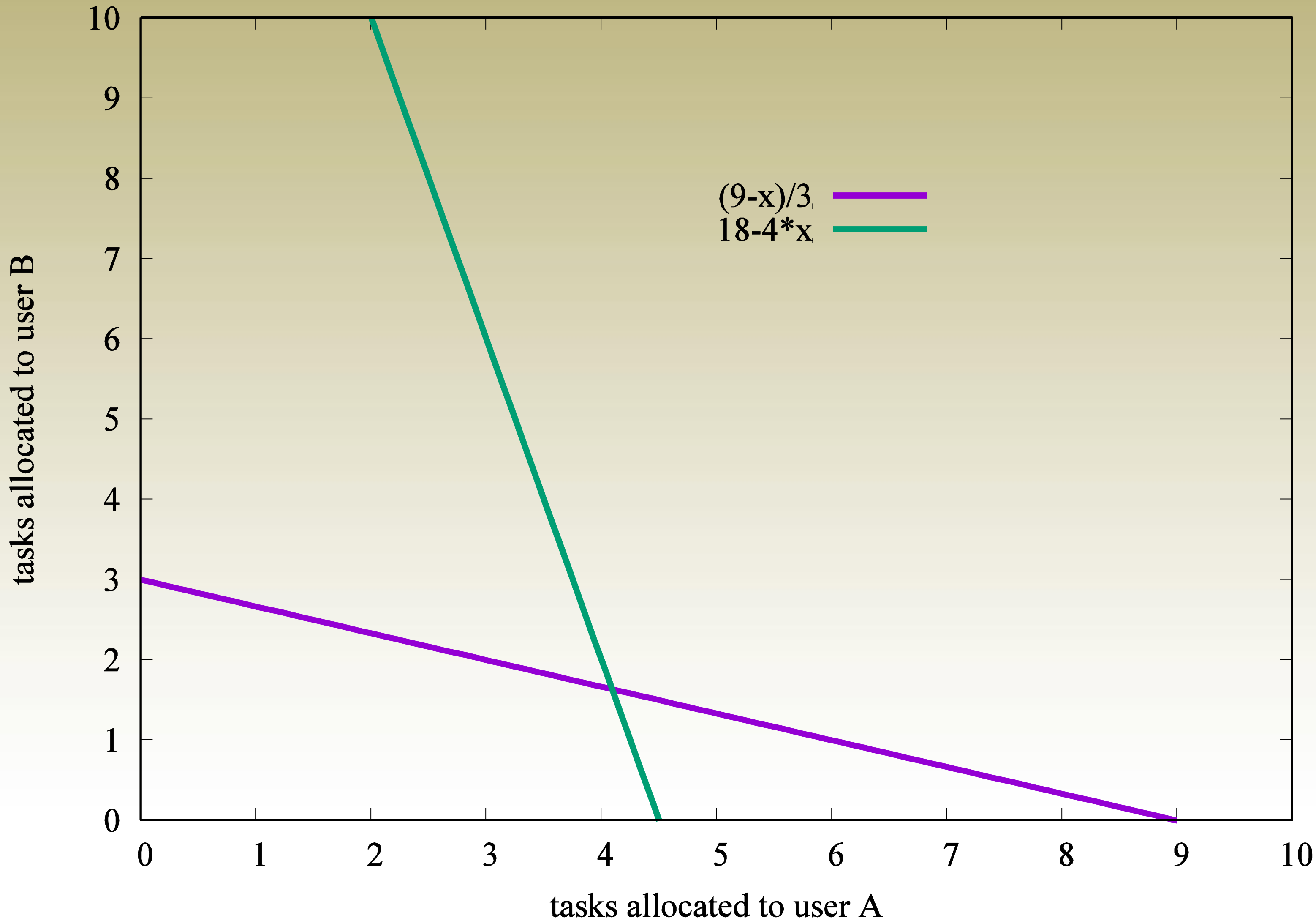
An Example

- The allocation can be computed mathematically:
- Let **x** and **y** be the number of tasks allocated by DRF to users A and B. It is obvious that x,y ≥ 0

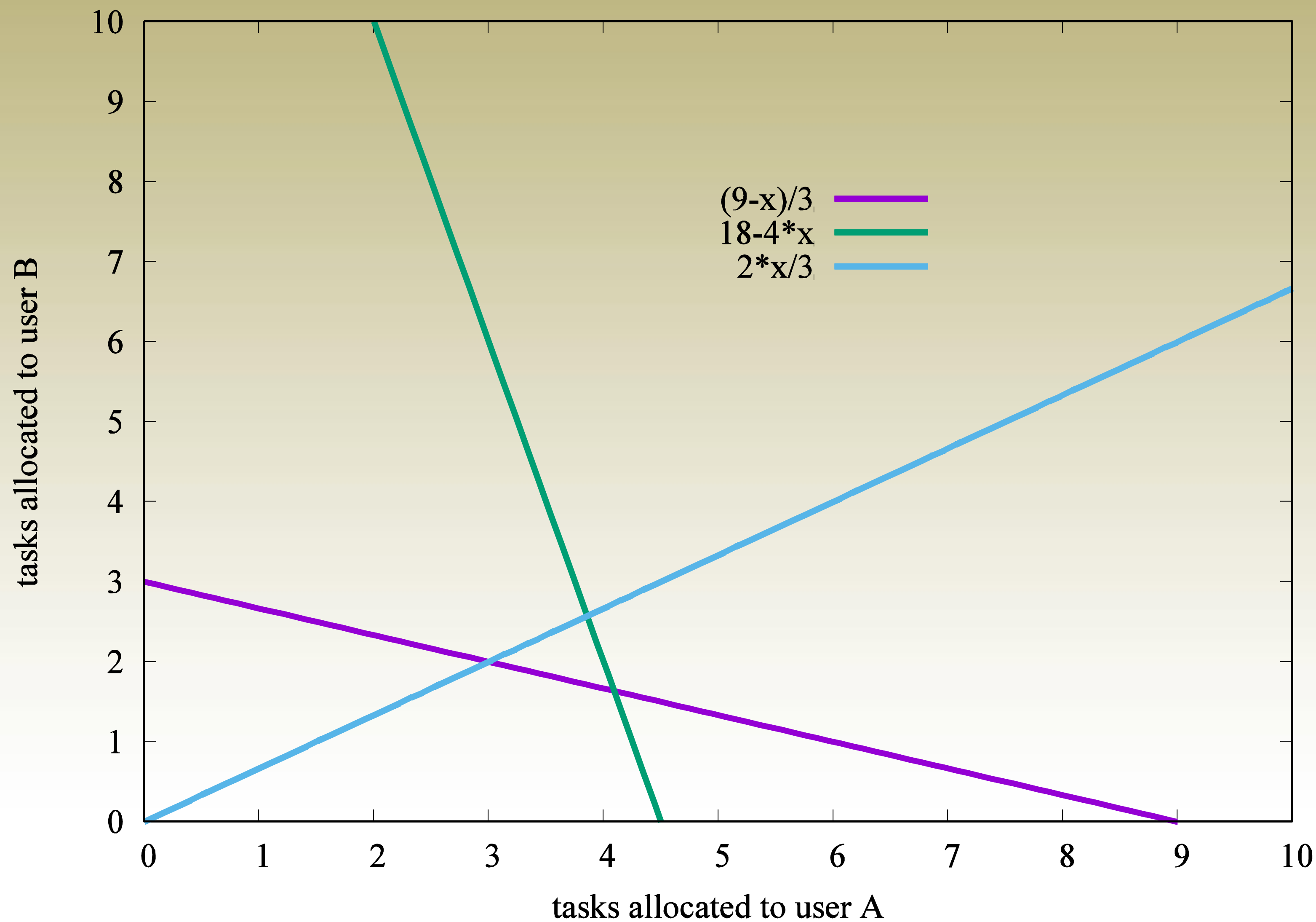
$$
\begin{aligned}
&\max(x, y) && \text{(Maximize allocations)} \\
&\quad \text{subject to} && \\
&\quad x + 3y \le 9 && \text{(CPU constraint)} \\
&\quad 4x + y \le 18 && \text{(Memory constraint)} \\
&\quad \frac{2x}{9} = \frac{y}{3} && \text{(Equalize dominant shares)}
\end{aligned}
$$

- Since x,y ≥ 0, it means that we must find: *max{x+y}*
- Solving this problem yields: x = 3 and y = 2
- User A gets <3 CPU, 12 GB> and B gets <6 CPU, 2 GB>

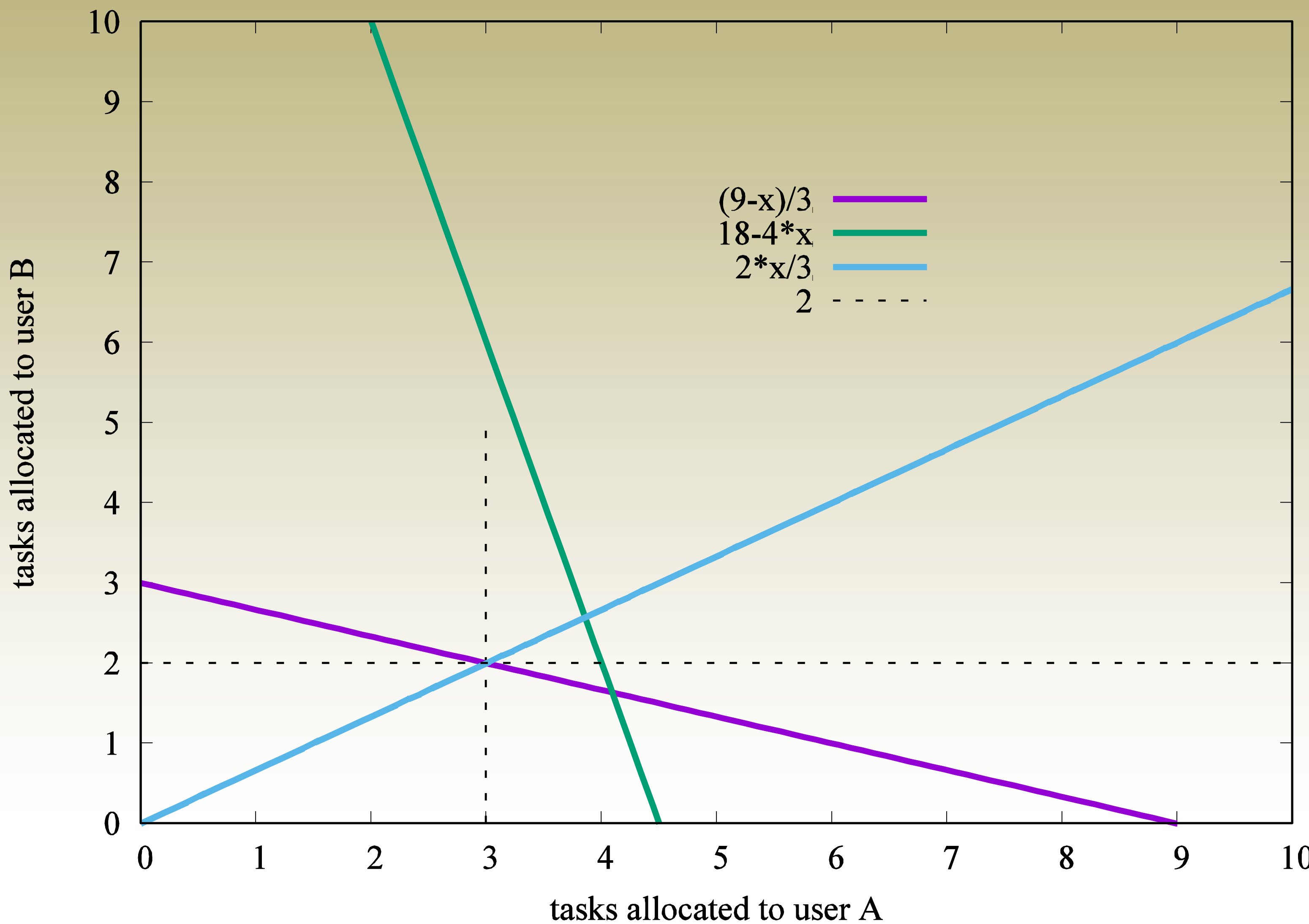
Geometrical Interpretation: Linear progr.



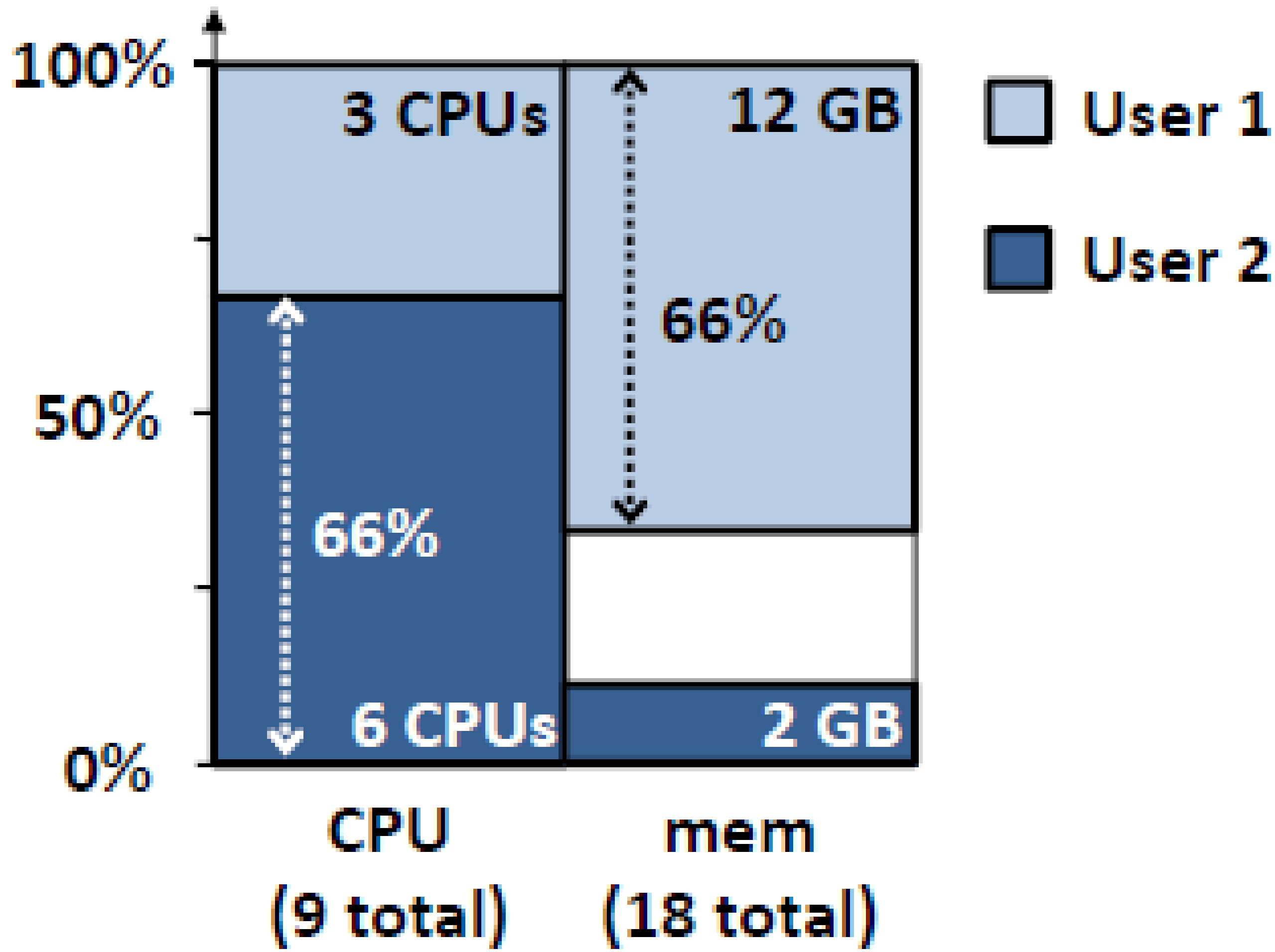
Geometrical Interpretation: Linear progr.



Geometrical Interpretation: Linear progr.



DRF allocation for the Example



An Example

Schedule	User A		User B		CPU	RAM
	res. shares	dom. share	res. shares	dom. share	total alloc.	total alloc.
User B	$\langle 0, 0 \rangle$	**0**	$\langle 3/9, 1/18 \rangle$	1/3	3/9	1/18
User A	$\langle 1/9, 4/18 \rangle$	**2/9**	$\langle 3/9, 1/18 \rangle$	1/3	4/9	5/18
User A	$\langle 2/9, 8/18 \rangle$	4/9	$\langle 3/9, 1/18 \rangle$	**1/3**	5/9	9/18
User B	$\langle 2/9, 8/18 \rangle$	**4/9**	$\langle 6/9, 2/18 \rangle$	2/3	8/9	10/18
User A	$\langle 3/9, 12/18 \rangle$	**2/3**	$\langle 6/9, 2/18 \rangle$	**2/3**	1	14/18

Table 1: Example of DRF allocating resources in a system with 9 CPUs and 18 GB RAM to two users running tasks that require $\langle$1 CPU, 4 GB$\rangle$ and $\langle$3 CPUs, 1 GB$\rangle$, respectively. Each row corresponds to DRF making a scheduling decision. A row shows the shares of each user for each resource, the user's dominant share, and the fraction of each resource allocated so far. DRF repeatedly selects the user with the lowest dominant share (indicated in bold) to launch a task, until no more tasks can be allocated.

Asset Fairness

- The idea behind Asset Fairness is that equal shares of different resources are worth the same, i.e., that 1% of all CPUs worth is the same as 1% of memory and 1% of I/O bandwidth
- Asset Fairness then tries to equalize the aggregate resource value allocated to each user
- Consider the previous example: since there are twice as many GB of RAM as CPUs (i.e., 9 CPUs and 18 GB RAM), one CPU is worth twice as much as one GB of RAM
- Supposing that one GB is worth \$1 and one CPU is worth \$2, it follows that user A spends \$6 for each task, while user B spends \$7

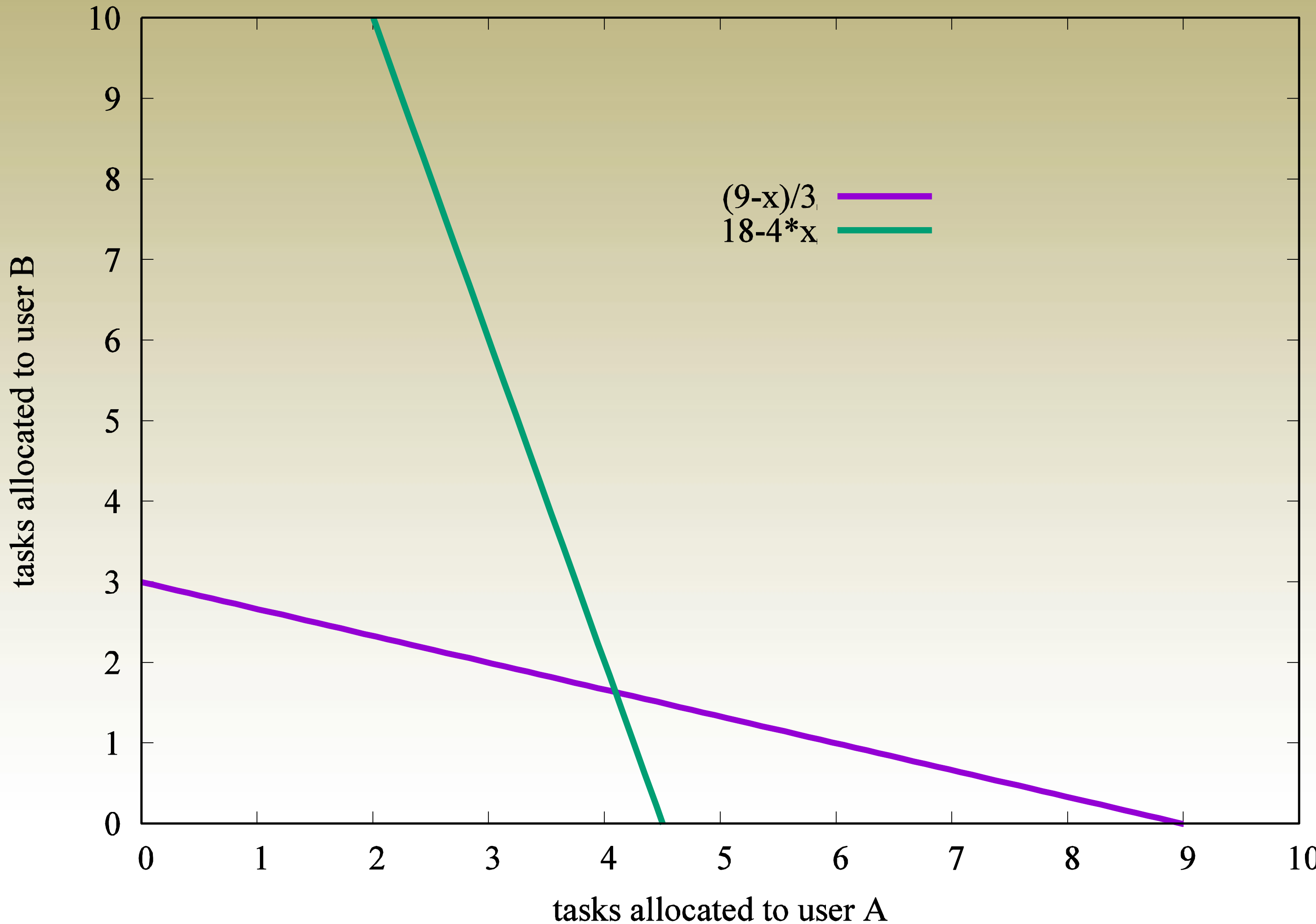
Asset Fairness

- The allocation can be computed mathematically:
- Let **x** and **y** be the number of tasks allocated by DRF to users A and B. It is obvious that x,y ≥ 0

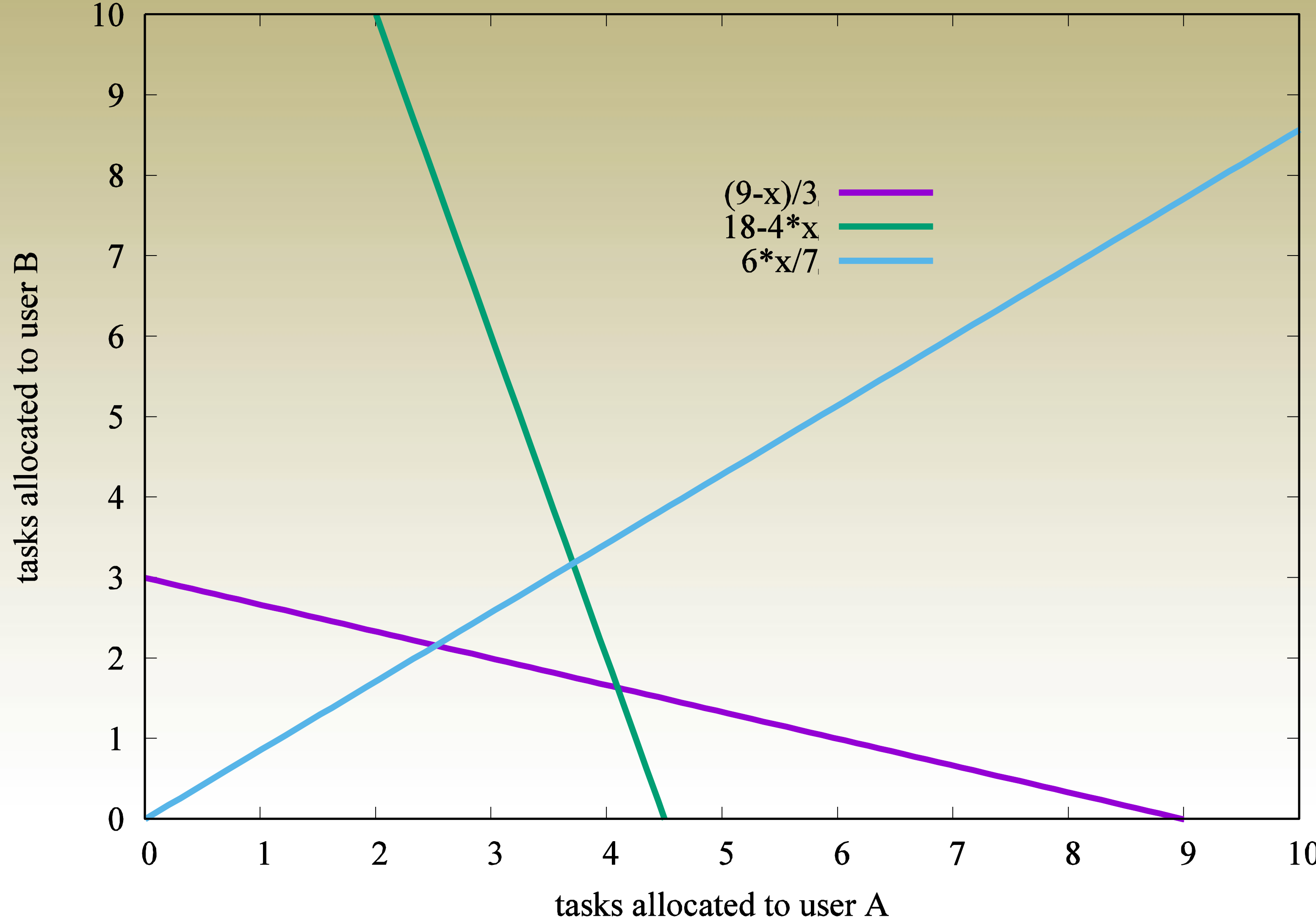
$$
\begin{aligned}
\max\,(x, y) & \qquad \text{(Maximize allocations)} \\
\text{subject to} & \\
x + 3y &\leq 9 \ \ \text{(CPU constraint)} \\
4x + y &\leq 18 \ \text{(Memory constraint)} \\
6x &= 7y \ \text{(Every user spends the same)}
\end{aligned}
$$

- Since x,y ≥ 0, it means that we must find: *max{x+y}*
- Solving this problem yields: x = 2.52 and y = 2.16
- User A gets <2.5 CPU, 10.1GB>, and B gets <6.5 CPU, 2.2GB>

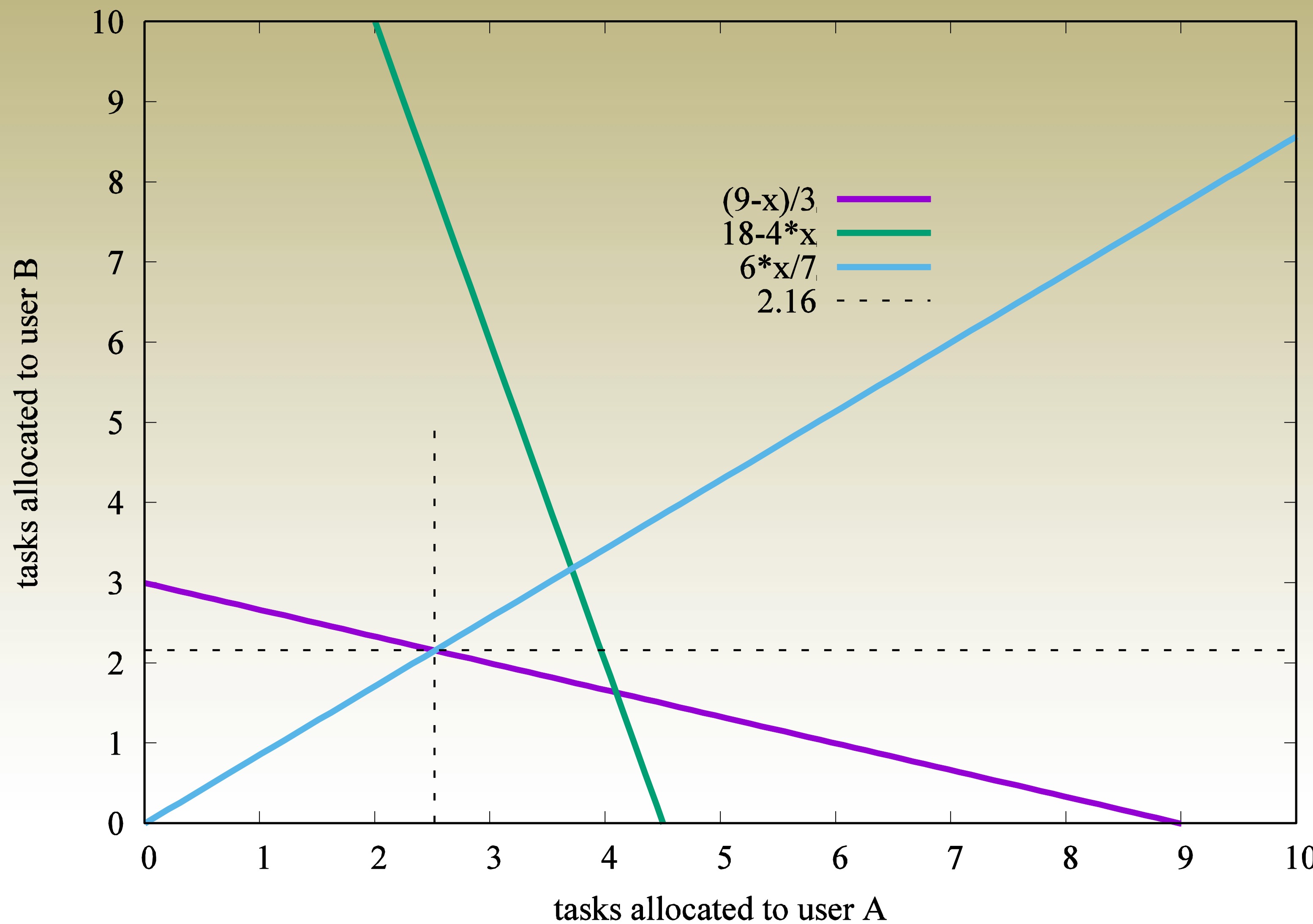
Geometrical Interpretation: Linear progr.



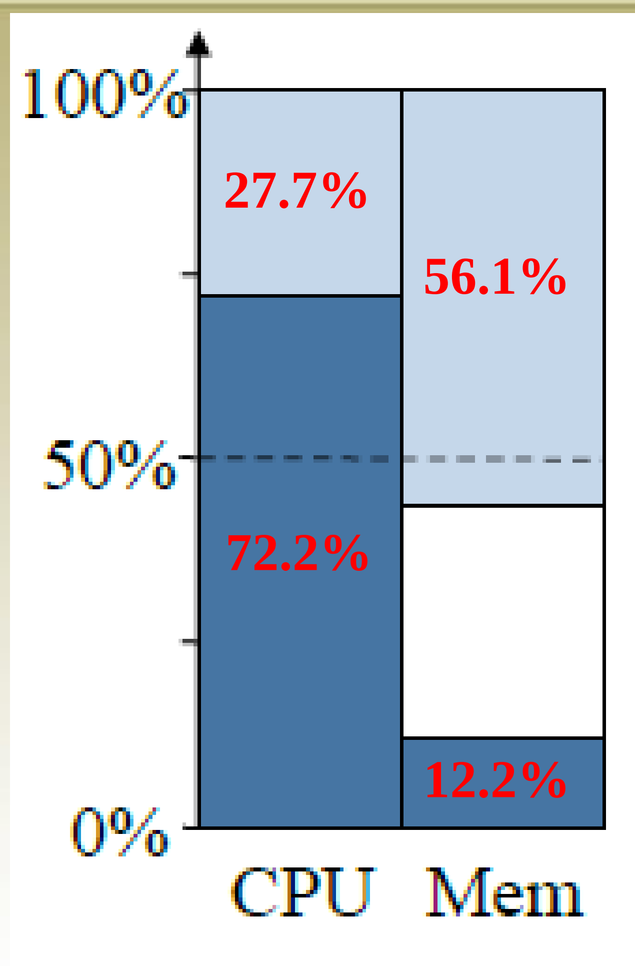
Geometrical Interpretation: Linear progr.



Geometrical Interpretation: Linear progr.



Asset Fairness allocation for the Example



Asset Fairness equalizes the total fraction of resources allocated to each user

Asset Fairness violates the sharing incentive property

- Two users in a system with <30, 30> total resources have demand vectors D1 = <1, 3>, and D2 = <1, 1>
- Asset fairness will allocate the first user 6 tasks and the second user 12 tasks
- The first user will receive <6, 18> resources, while the second will use <12, 12>
- While each user gets an equal aggregate share of 24/60, the second user gets less than half (15) of both resources
- This violates the sharing incentive property, as the second user would be better off to statically partition the cluster and own half of the nodes

Asset Fairness violates the bottleneck fairness property

- Consider a scenario with total resources <21, 21>
- Two users with demand vectors D1 = <3, 2> and D2 = <4, 1>
- Thus, resource 1 is the bottleneck resource

- Asset fairness will give each user 3 tasks, equalizing their aggregate usage to 15
- However, this only gives the first user 3/7 of resource 1 (the contended bottleneck resource), violating bottleneck fairness

Asset Fairness does not satisfy resource monotonicity

- Two users A and B with demands <4, 2> and <1, 1> and 77 units of two resources
- Asset fairness allocates A a total of <44, 22> and B <33, 33> equalizing their sum of shares to 66/77
- If resource two is doubled, both users' share of the second resource is halved, while the first resource is saturated

- Asset fairness now decreases A's allocation to <42, 21> and increases B's to <35, 35>, equalizing their shares to $42/77 + 21/154 = 35/77 + 35/154 = 105/154$
- Thus, resource monotonicity is violated

Competitive Equilibrium from Equal Incomes (CEEI)

- With CEEI, each user receives initially 1/n of every resource, and subsequently,
- each user trades her resources with other users in a perfectly competitive market
  - A *perfect market* satisfies the price-taking (i.e., no single user affects prices) and market-clearance (i.e., matching supply and demand via price adjustment) assumptions
- The outcome of CEEI is both: envy-free, & Pareto efficient
- CEEI allocation is given by the *Nash bargaining* solution:
  - The Nash bargaining solution picks the feasible allocation that maximizes $\Pi\{u_i(a_i)\}$, where $u_i(a_i)$ is the utility that user i gets from her allocation a_i. To simplify the comparison, we assume that the utility that a user gets from her allocation is simply her dominant share s_i

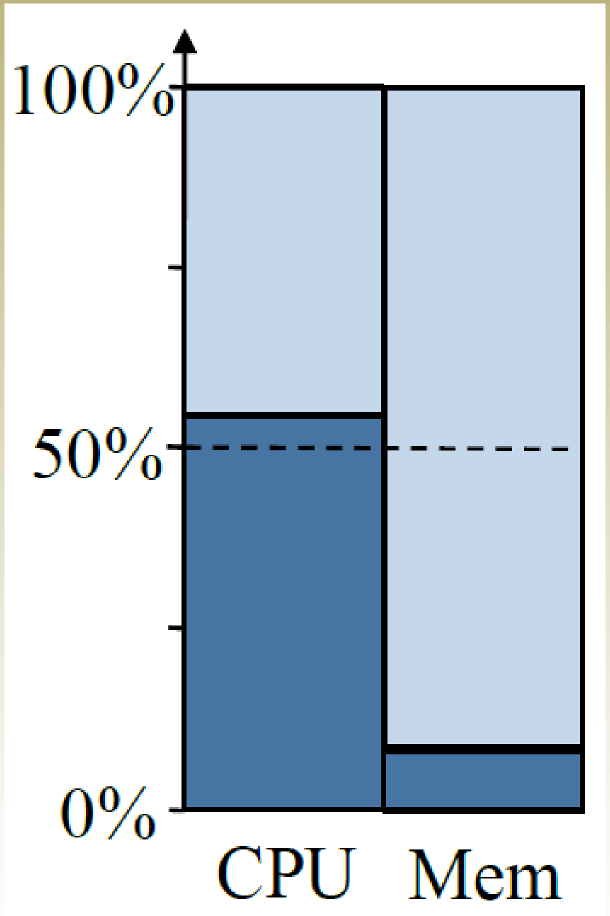
Competitive Equilibrium from Equal Incomes (CEEI)

- Let x and y be the number of tasks allocated by CEEI to users A and B from the previous example, where the dominant share of user A is 4x/18 =2x/9 while the dominant share of user B is 3y/9 = y/3,
- CEEI allocation can be computed mathematically:

$$
\begin{aligned}
&\max\,(x \cdot y) && \text{(maximize Nash product)} \\
&\quad \text{subject to} && \\
&\qquad x + 3y \;\le\; && 9 \text{ (CPU constraint)} \\
&\qquad 4x + y \;\le\; && 18 \text{ (Memory constraint)}
\end{aligned}
$$

- Solving the above yields: x = 45/11=4.1 and y = 18/11=1.6
- Thus, user A gets <4.1 CPUs, 16.4 GB>, while user B gets <4.9 CPUs, 1.6 GB>

CEEI allocation for the Example



CEEI assumes a perfectly competitive market, and thus strives to find a solution satisfying market clearance, where every resource has been allocated

CEEI is not strategy-proof

- Assume total resources <100, 100> and two users with demands <16, 1> and <1, 2>
- CEEI allocates 100/31 and 1500/31 tasks to each user respectively (approximately 3.2 and 48.8 tasks)
- If user 1 changes her demand vector to <16, 8>, asking for more of resource 2 than she actually needs, CEEI gives the users 25/6 and 100/3 tasks respectively (approximately 4:2 and 33:3 tasks)

- Thus, user 1 improves her number of tasks from 3.2 to 4.2 by lying about her demand vector
- User 2 suffers because of this, as her task allocation decreases

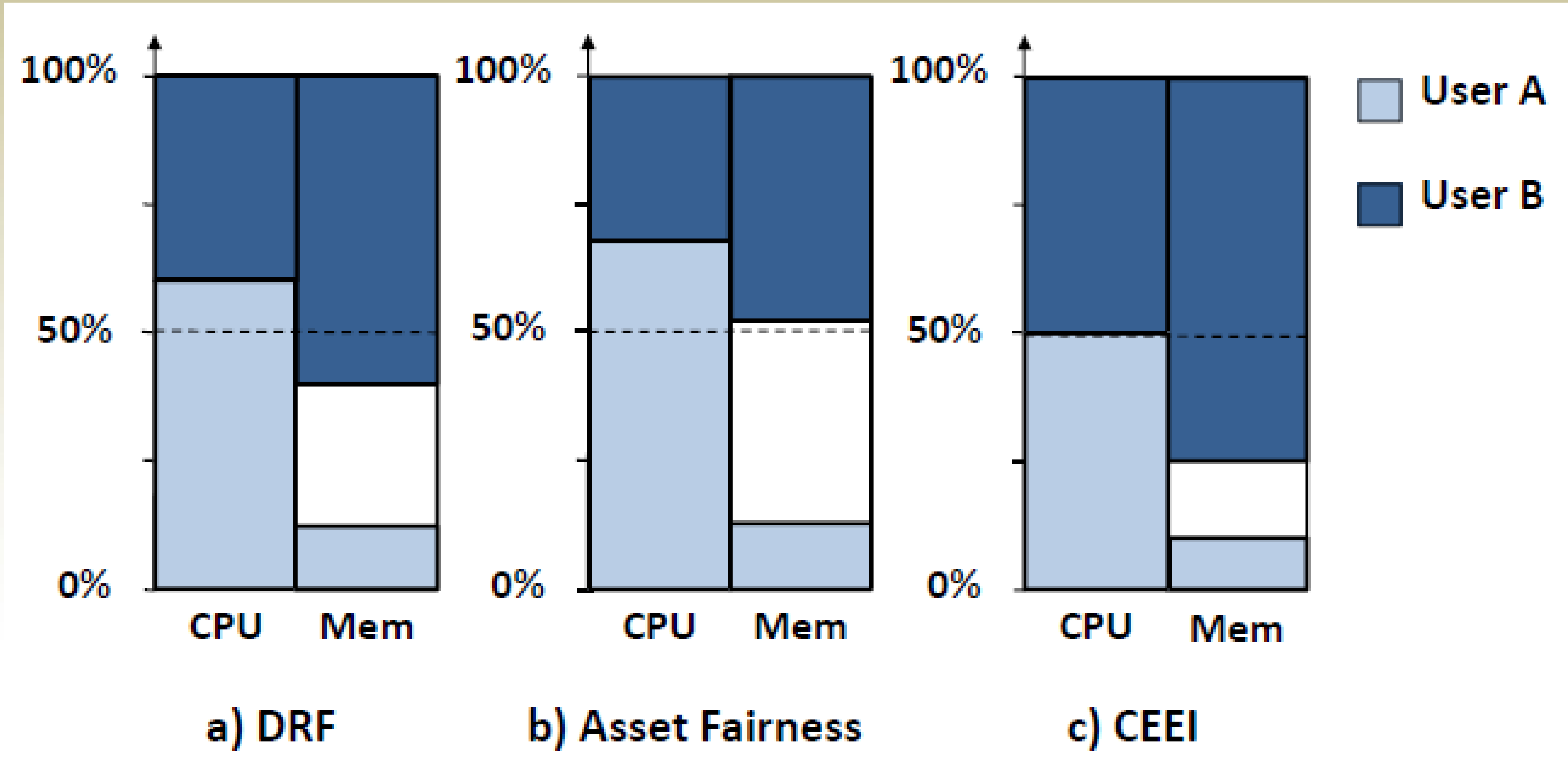
CEEI violates population monotonicity

- Consider the total resource vector <100, 100> and three users with the following demand vectors D_1=<4,1>, D_2=<1,16>, and D_3=<16,1>
- CEEI will yield the allocation A_1 = <11.3, 5.4, 3.1>

- If user 3 leaves the system and relinquishes her resource, CEEI gives the new allocation A_2 = <23.8, 4.8>, which has made user 2 worse off than in A_1

Example of DRF vs. Asset vs. CEEI

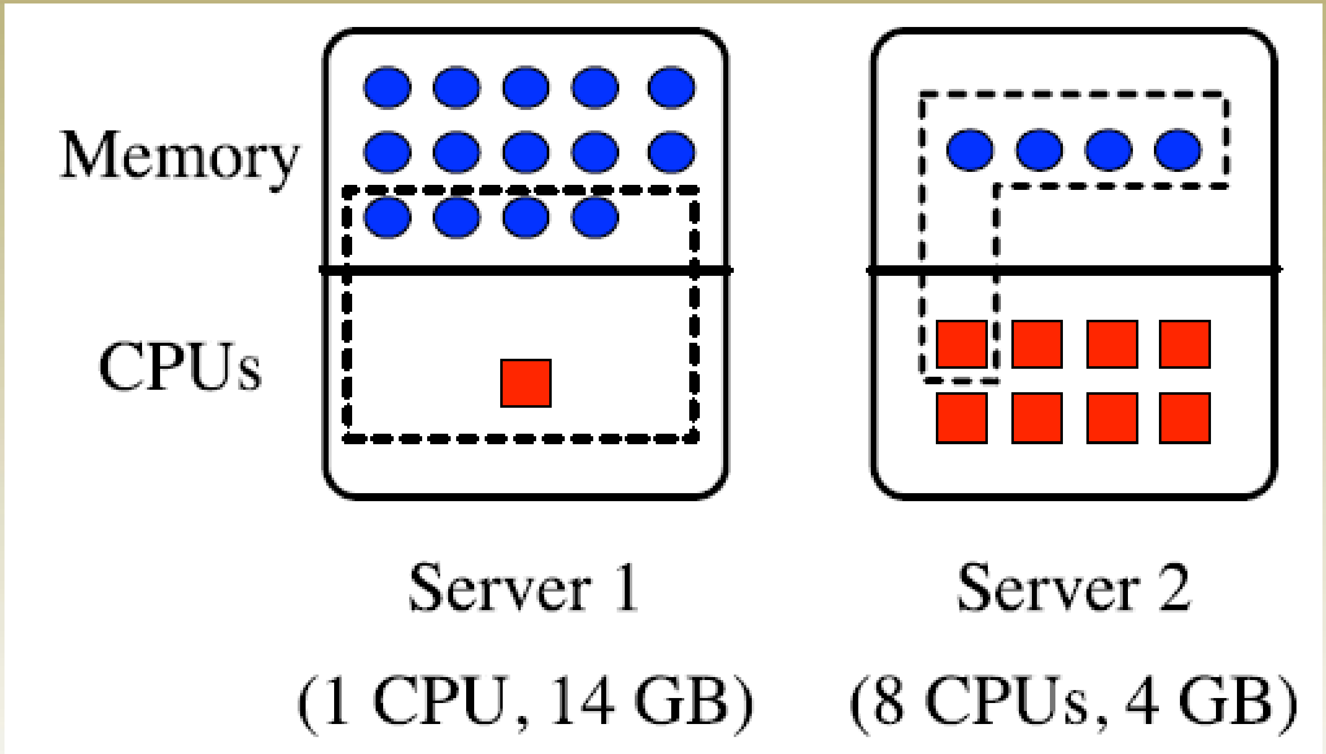
- Resources <1000 CPUs, 1000 GB>
- 2 users A: <2 CPU, 3 GB> and B: <5 CPU, 1 GB>



Properties of DRF, Asset Fairness, CEEI

Property	Asset	CEEI	DRF
Share guarantee		✔	✔
Strategy-proofness	✔		✔
Pareto efficiency	✔	✔	✔
Envy-freeness	✔	✔	✔
Single resource fairness	✔	✔	✔
Bottleneck res. fairness		✔	✔
Population monotonicity	✔		✔
Resource monotonicity			

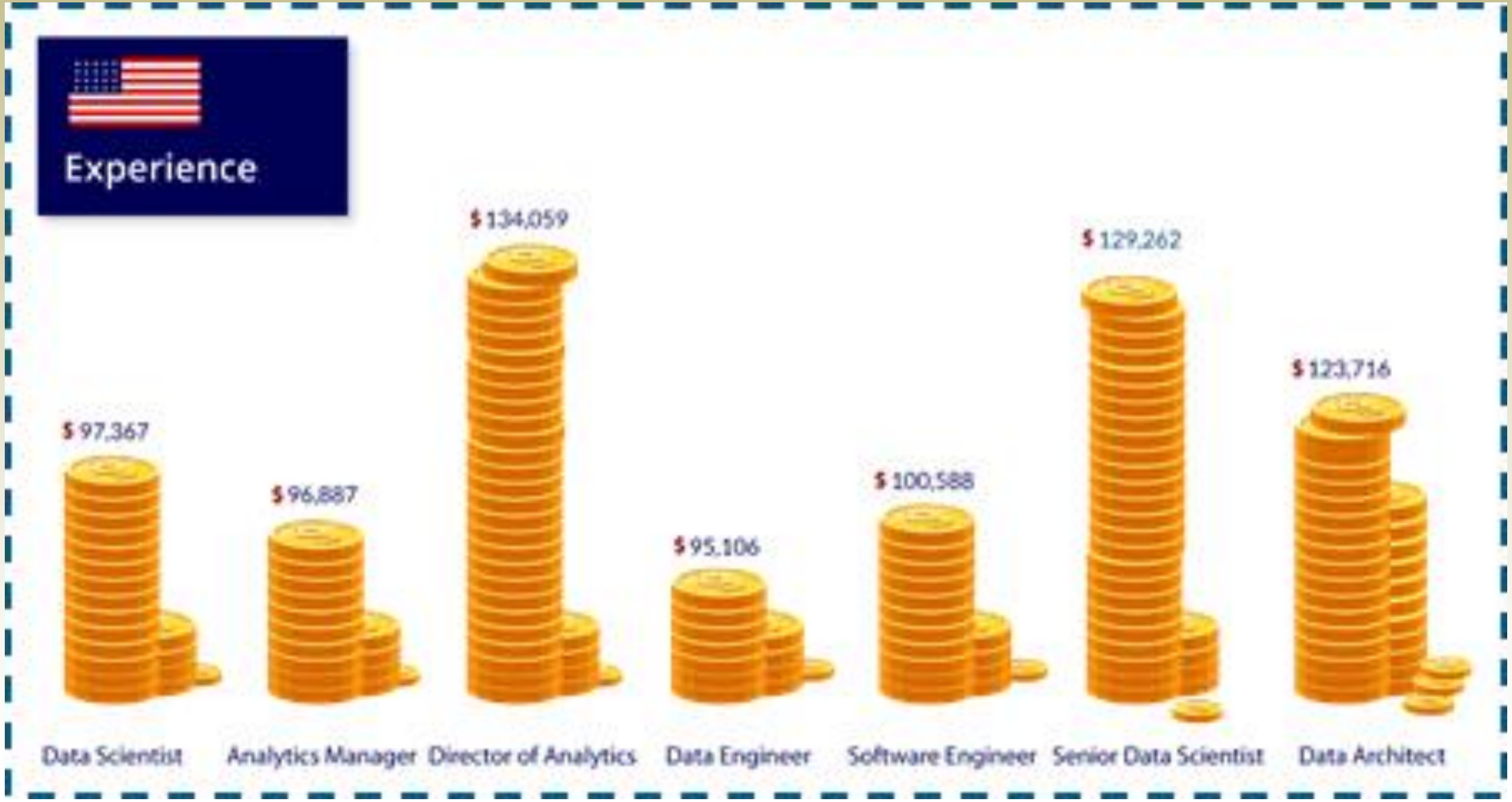
What if "all-in-one" is not valid?



- User A's tasks demand <1 CPU, 4 GB>, and user B's tasks demand <3 CPUs, 1 GB> each, as previous
- DRF allocates: 3 tasks in user A and 2 tasks in user B
- Here: **user A can get at most 1 task in either server!**

Average salary for employees with Hadoop skills

US: Position-wise salary distribution



US: Experience-wise salary distribution

US: Major companies hiring for Hadoop

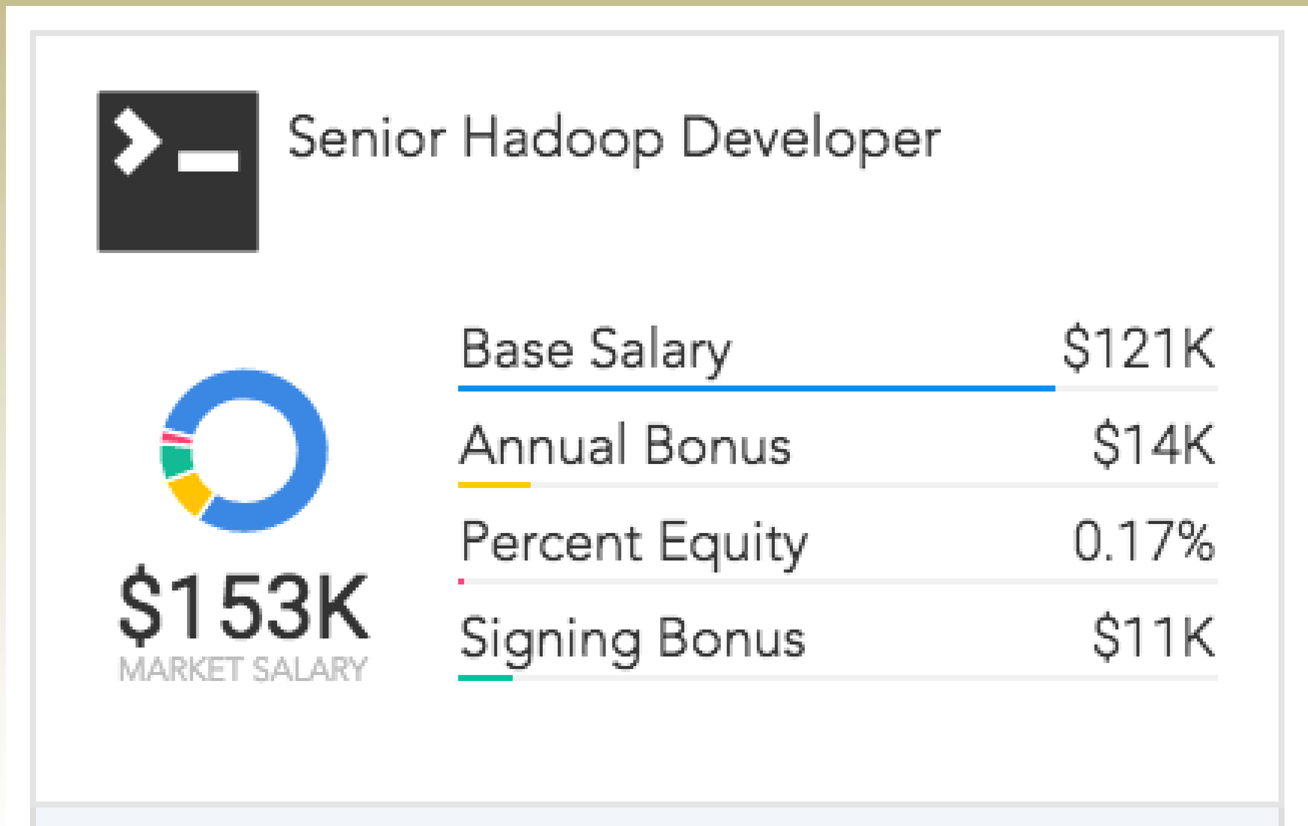
- Amazon Inc: $78,264 - $161,178
- International Business Machines (IBM) Corp.: $72,052 - $163,043
- Capital One Financial Corp: $90,200 - $183,994
- Microsoft Corp: $98,735 - $158,117
- Booz, Allen, and Hamilton: $54,248 - $172,310
- Facebook Inc: $92,110 - $199,332
- Deloitte Consulting LLP: $71,768 - $185,550
- General Electric Co (GE): $72,200 - $221,250
- Expedia, Inc.: $88,275 - $137,500
- UnitedHealth Group: $60,000 - $140,283
- Google, Inc.: $66,977 - $156,111
- Accenture: $78,906 - $183,125
- J.P. Morgan Chase & Co. (JPMCC): $92,371 - $182,322
- Cisco Systems Inc: $83,957 - $151,894
- Comcast Cable, Inc.: $73,899 - $157,812

US: Major companies hiring for Hadoop

- eBay Inc.: $110,738 - $213,679
- American Express Co. (AMEX): $85,569 - $140,482
- The Nielsen Company: $110,011
- Citibank: $94,259
- Deloitte: $92,500
- EY (Ernst & Young): $95,000
- Uber Technologies, Inc.: $111,910
- Paypal, Inc.: $142,482
- Humana, Inc.: $128,482
- Apple Computer, Inc: $132,635
- Verisk Analytics: $90,000
- Johnson & Johnson: $117,447
- Wells Fargo & Co.: $126,403
- Oracle Corp.: $143,415
- Aetna, Inc.: $85,059

US Avg salary: Hadoop developer

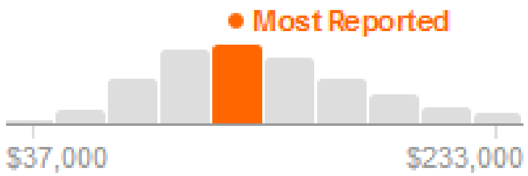
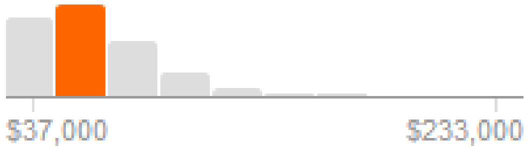
US Avg salary: senior Hadoop developer



Entry Level Hadoop Salaries in the United States

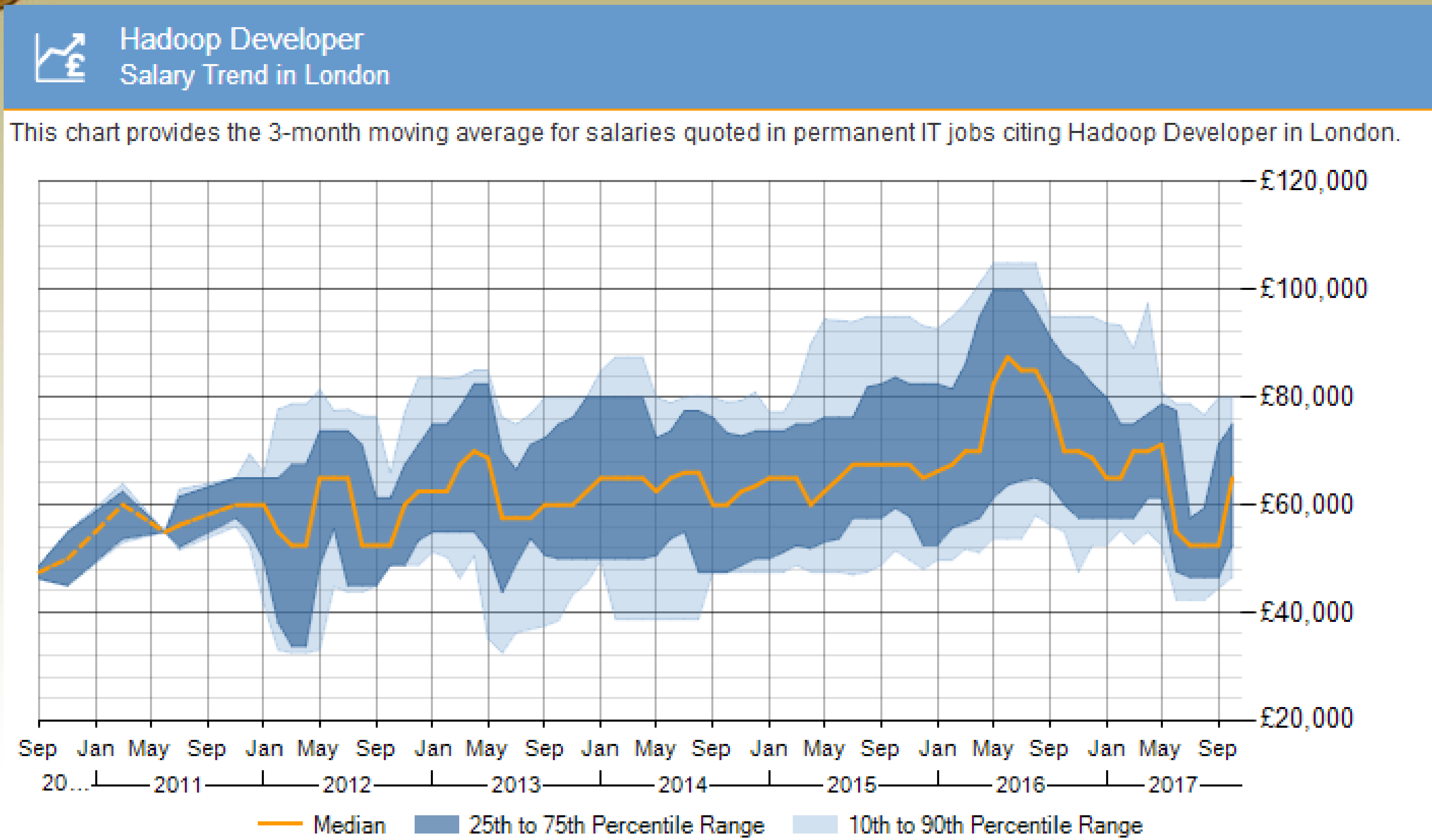
Location

United States

Popular Jobs	Average Salary	Salary Distribution
Data Scientist 40,323 salaries reported Data Scientist Jobs	**$130,529** per year	
Entry Level Software Engineer 3,778 salaries reported Entry Level Software Engineer Jobs	**$65,706** per year	
Research Scientist 8,784 salaries reported Research Scientist Jobs	**$76,171** per year	
Entry Level Technician 1,658 salaries reported Entry Level Technician Jobs	**$38,721** per year	

Hadoop developer: London Salary trend

3-month moving average for salaries quoted in permanent IT jobs citing Hadoop Developer in London



Hadoop Developer Skill Set

Top 30 Co-occurring IT Skills in London: six-months period up to October 2017

1	70 (100.00%)	Hadoop	10	20 (28.57%)	Telecoms
2	66 (94.29%)	Big Data	10	20 (28.57%)	Marketing
3	59 (84.29%)	Apache Spark	10	20 (28.57%)	Digital Marketing
4	49 (70.00%)	Java	11	18 (25.71%)	Elasticsearch
5	36 (51.43%)	Python	12	15 (21.43%)	OO
6	35 (50.00%)	Scala	13	14 (20.00%)	Akka
7	24 (34.29%)	Finance	13	14 (20.00%)	Greenfield Project
8	22 (31.43%)	Open Source	14	13 (18.57%)	Kafka
9	21 (30.00%)	Data Warehouse	15	12 (17.14%)	HBase
9	21 (30.00%)	ETL	16	11 (15.71%)	Docker
10	20 (28.57%)	Business Intelligence	17	10 (14.29%)	Amazon AWS
10	20 (28.57%)	SAP	17	10 (14.29%)	BDD
10	20 (28.57%)	Programme Management	17	10 (14.29%)	Google
10	20 (28.57%)	Manufacturing	17	10 (14.29%)	TDD
10	20 (28.57%)	Electronics	17	10 (14.29%)	Google Cloud Platform

Hadoop developer: Germany Salary trend