Dominant Resource Fairness: Fair Allocation of Multiple Resource Types

Ali Ghodsi,
Matei Zaharia,
Benjamin Hindman,
Andy Konwinski,
Scott Shenker,
Ion Stoica

University of California, Berkeley

Talk by: Yael Amsterdamer

Why Fairness?

- **Classic problem:** a common resource
  - machines, pastures…
- Every user has different needs
- How should the resources be split?
  - Not enough to satisfy everyone

First attempt

- Split according to the needs
- **Example:**
  - Alice needs 4GB for MATLAB, Bob needs 2GB for watching "The Dictator" in HD
  - The system has 4GB
  - Alice gets 2.6GB and Bob 1.4GB.
- **Downsides:**
  - Why should Alice get more?
  - Everybody has an incentive to lie

Second attempt

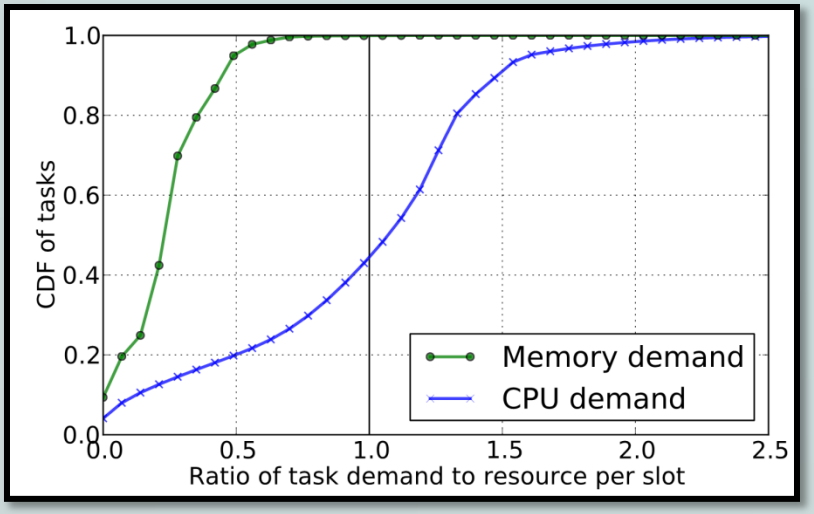
- Split evenly
- **In our example:**
  - Alice gets 2GB and Bob 2GB
- Max-Min fairness
  - Maximize the share of the worst participant, then the second, etc.
- **Upsides**
  - No lying
- **Downsides:**
  - Bob can probably do with less
  - Alice does need more resources…

Multiple Resources

- Sharing more than one type of resource
  - CPU, memory, hard drive, bandwidth
  - Pasture, water, trails
- We can split each resource separately
  - Evenly or according to needs
- A more sophisticated way of splitting?

Multiple Resources Examples

- **4 GB memory;** Alice needs 4 and Bob 2.
- **New resource: 4 CPUs;** Alice needs 2 and Bob 3.
- Splitting evenly does not make sense
  - Alice should get more memory, and Bob should get more CPUs.
  - An incentive to give up less needed resource for a more needed one
- **Example from the paper:**

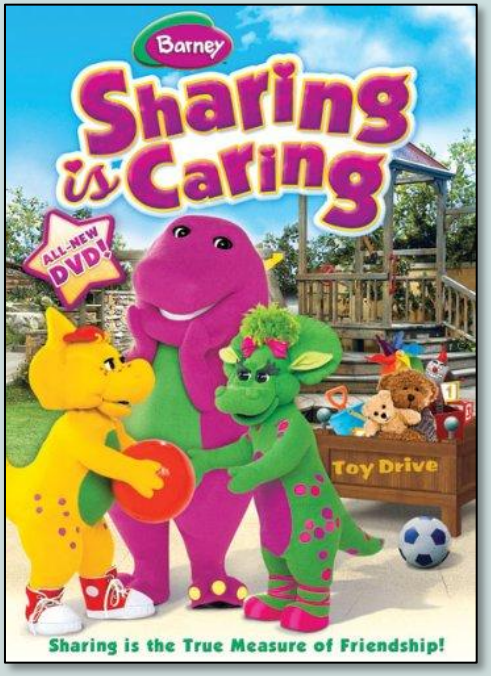


Properties of a good allocation

1. **Incentive Compatibility**
   - Participants cannot improve their situation by lying
2. **Pareto efficiency**
   - Some participant would lose from every change of allocation

Properties of a good allocation – cont'd

3. **Envy-freeness**
   - One participant would not prefer the allocation of another
   - Equal needs ➔ equivalent allocations
4. **Sharing incentive**
   - When each participant contributes equal resources to the pool
   - E.g., a node in a cluster; a specific region in a pasture
   - Sharing should be better than using one's own resources (1/n)
   - A minimum allocation guarantee



Additional Properties

5. **Single-resource fairness**
   - Only one resource ➔ Max-Min fairness
6. **Bottleneck fairness**
   - Equally demanded resource ➔ Max-Min fairness
7. **Monotonicity**
   - Removed user / added resource ➔ allocation does not decrease

Max-Min fairness – an exercise

- In a single resource case
- Is it incentive compatible?
  - Yes, demands are ignored
- Pareto efficient?
  - Yes, if someone gets more than 1/n, someone will get less
- Envy free?
  - Yes, same allocation for all
- Sharing incentive?
  - Weak yes, it is the same as not sharing

Dominant Resource Fairness - basics

- **The settings:** users have different types of tasks
- *n* types of resources with limited capacity

$$(r_1^{max}, r_2^{max}, \dots, r_n^{max})$$

- Every task type i has a demand vector

$$(r_1^i, r_2^i, \dots, r_n^i)$$

- **Definition:** the dominant resource of user i is the resource maximal relative demand

$$\arg\max_j \frac{r_j^i}{r_j^{\max}} \cdot a_i$$

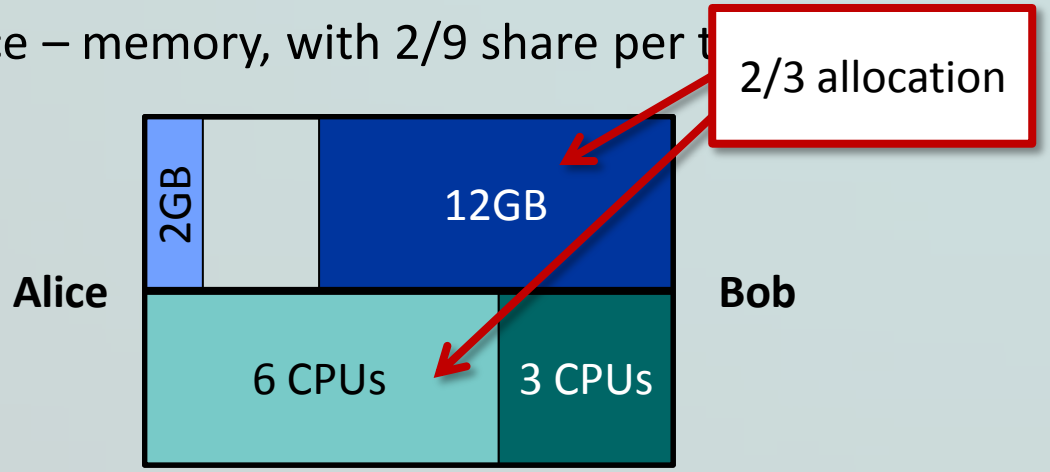
Dominant share

Dominant Resource Fairness

- **Policy Definition:**
  Maximize the minimal dominant share in the system (max-min-max!)
  - Not a trivial extension of Max-Min – allocation of one resource causes allocations of other resources

An example

- The system resources:
  (CPU, memory) = (9 CPUs, 18GB)
- Alice has MATLAB tasks with demand vector (3 CPUs, 2GB)
  - Her dominant resource – CPU, with 1/3 share per task
- Bob has movie tasks with demand vector (1CPUs, 4GB)
  - His dominant resource – memory, with 2/9 share per t
- The DRF allocation:



Problem formalization

- $x_1, x_2, \ldots, x_N$ – # of allocated tasks for each user.
- **Maximize** $x_1, x_2, \ldots, x_N$
  Subject to

$$\begin{cases} \sum_{i=1}^{N} r_k^i \cdot x_i \le r_k^{\max} \quad (k = 1, \ldots, n) \\ \max_k \frac{r_k^i}{r_k^{\max}} \cdot x_i = \max_l \frac{r_l^j}{r_l^{\max}} \cdot x_j \quad (i \ne j \in 1, \ldots, N) \end{cases}$$

Resource limitations

Dominant share balance

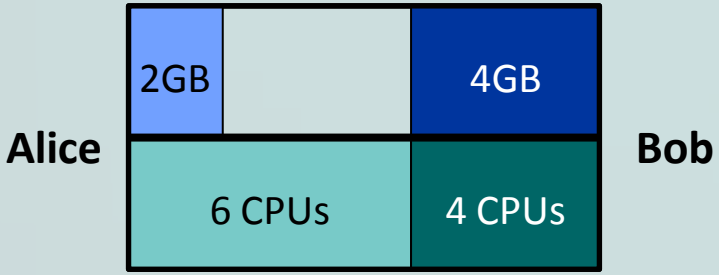
Greedy scheduling algorithm

- Iteratively:
  - Pick the user with minimal dominant share
  - If there are enough resources:
    - Launch their next task
    - Update the current resource allocation, resources per user, and user dominant shares

- **Advantages:**
  - Deals with different user task types
  - Deals with task completions
  - Deals with saturated resources

Asset Fairness

- Balance the total fraction of resources allocated for each user
- **Maximize** $x_1, x_2, \ldots, x_N$
  Subject to



$$\begin{cases} \sum_{i=1}^{N} r_k^i \cdot x_i \le r_k^{\max} \quad (k = 1, \ldots, n) \\ \sum_{k=1}^{n} \frac{r_k^i}{r_k^{\max}} \cdot x_i = \sum_{l=1}^{n} \frac{r_l^j}{r_l^{\max}} \cdot x_j \quad (i \ne j \in 1, \ldots, N) \end{cases}$$

- Violates sharing incentive property

Competitors

Competitive Equilibrium from Equal Incomes (CEEI)

- Resource division method from microeconomics
- Each user trades his resources in a perfect market
- Nash bargaining solution – maximizes the utility product
- **Maximize** $$\prod_{i=1}^{N} u_i(a_i) = \prod_{i=1}^{N} x_i$$

  Subject to

  $$\sum_{i=1}^{N} r_k^i \cdot x_i \leq r_k^{\max} \quad (k = 1, \dots, n)$$

Resources are fully utilized; Users that ask for unneeded resources will get more tasks

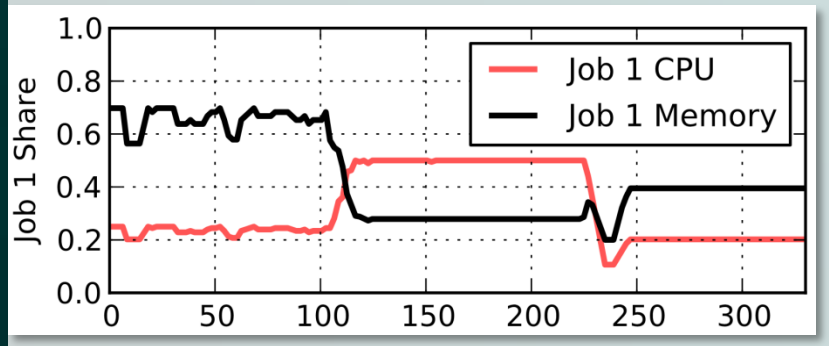
- Not incentive compatible

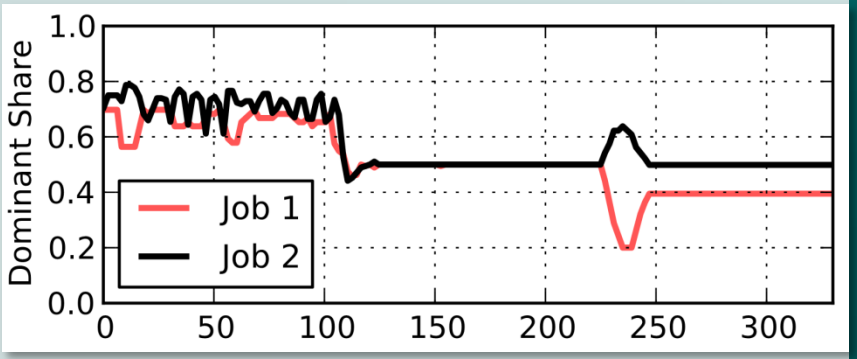
Comparison Table

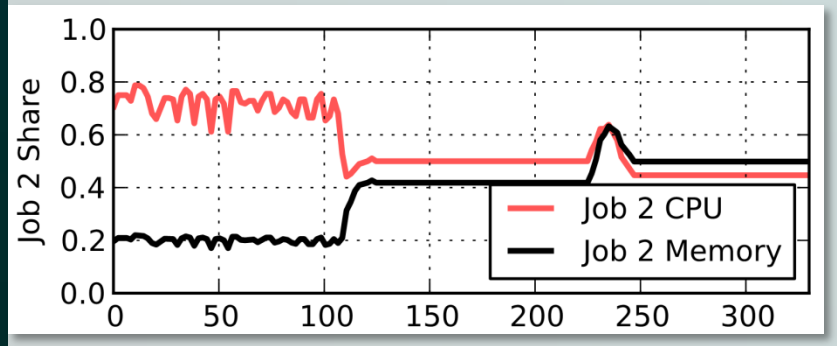
Property	Allocation Policy		
	Asset	CEEI	DRF
Sharing Incentive		✓	✓
Strategy-proofness	✓		✓
Envy-freeness	✓	✓	✓
Pareto efficiency	✓	✓	✓
Single Resource Fairness	✓	✓	✓
Bottleneck Fairness		✓	✓
Population Monotonicity	✓		✓
Resource Monotonicity			

- **Theorem:** sharing incentive and Pareto efficient policies cannot be resource monotone

DRF in practice







Summary

We have talked about

- Fairness has been studied mostly for a single resource type
- Users with different needs can benefit from sharing
- Properties of fair policy
  - **Incentive compatibility**
  - **Pareto efficiency**
  - **Envy-freeness**
  - **Sharing incentive**
- RDF maximizes minimal dominant resource
- Has better properties than competitors

Future work

- Practical aspects, more general investigation of the properties
- **Open problem:** Redefine fairness for tasks with different placement constraints
- Ideas:
  - Choose tasks by minimal dominant share AND unutilized type of machines
  - Relation to restricted job assignment
- Challenges:
  - Utilize system resources
  - Keep fairness properties (or prove impossibility)
    In particular, avoid prioritizing resource allocation at the expense of incentive compatibility

Thank You!

Questions?