THIS CLASS:

DYNAMIC FAIR DIVISION & ALLOCATING FOOD TO FOOD BANKS

PART I: JOHN DICKERSON
PART II: JACOB RASIEL & KEVIN BOCK

Thanks to: Nisarg Shah (NS)
TODAY’S PROBLEM

Like most lectures in this class:
• $m$ items (initially divisible, later indivisible)
• $k$ agents with private values for bundles of items
Either the agents, the items, or both arrive over time.

This class:
• **Start with fair allocation of multiple divisible resources in a dynamic setting** [Kash Procaccia Shah JAIR-2014]
• **Move to fair dynamic allocation of indivisible items via a restricted bidding language** [Aleksandrov et al. IJCAI-2015]
• **Wrap up with a richer bidding language based on funny money** [Prendergast w.p. 2015]
ALLOCATING OF DIVISIBLE RESOURCES WITHOUT MONEY

Allocating computational resources (CPU, RAM, HDD, etc)

• Organizational clusters (e.g., our new Horvitz cluster)
• Federated clouds
• NSF Supercomputing Centers

We’ll focus on fixed bundles (slots)

• Allocated using single resource abstraction

Highly inefficient when users have heterogeneous demands
DOMINANT RESOURCE FAIRNESS (DRF) MECHANISMS

Idea: Assume structure on user demands

Proportional demands (a.k.a. Leontief preferences)

\[ u(x_1, \ldots, x_m) = \min \left\{ \frac{x_1}{w_1}, \ldots, \frac{x_m}{w_m} \right\} \]

Example:

- User wishes to execute multiple instances of a job
- Each instance needs (1 unit RAM, 2 units CPU)
- Indifferent between (2, 4) and (2, 5)
- Happier with (2.1, 4.2)
STATIC DRF MECHANISM
Dominant Resource Fairness = equalize largest shares
(a.k.a. dominant shares)
PROBLEM WITH DRF
[Kash Procaccia Shah JAIR-14]

Assumes all agents are present from the beginning and all the job information is known upfront

Can relax this to dynamic setting:

• Agents arriving over time
• Job information of an agent only revealed upon arrival

This paper initiated the study of dynamic fair division

• Huge literature on fair division, but mostly static settings
• Still very little work on fair division in dynamic environments!
FORMAL DYNAMIC MODEL

Resources are known beforehand

Agents arrive at different times (steps), do not depart
- Total number of agents known in advance

Agents’ demands are proportional, revealed at arrival
- Each agent requires every resource

Simple dynamic allocation mechanism:
- At every step k
  - Input: k reported demands
  - Output: An allocation over the k present agents
- Terminate after final agent arrives

Irrevocability of resources!
# DESIDERATA

Properties of DRF, aims for a dynamic generalization

<table>
<thead>
<tr>
<th>Property</th>
<th>Static (DRF)</th>
<th>Dynamic (Desired)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Envy freeness</td>
<td>EF: No swaps.</td>
<td>EF: No swaps at any step.</td>
</tr>
<tr>
<td>Sharing incentives</td>
<td>SI: At least as good as equal split.</td>
<td>SI: At least as good as equal split to every present agent at all steps.</td>
</tr>
<tr>
<td>Strategyproofness</td>
<td>SP: No gains by misreporting.</td>
<td>SP: No gains at any step by misreporting.</td>
</tr>
<tr>
<td>Pareto optimality</td>
<td>PO: No “better” allocation.</td>
<td>DPO: At any step k, no “better” allocation using k/n share of each resource.</td>
</tr>
</tbody>
</table>
IMPOSSIBILITY RESULT

Envy freeness + Dynamic Pareto optimality = Impossible

- DPO requires allocating too much
- Later agents might envy earlier agents

Dropping either of them completely $\Rightarrow$ trivial mechanisms!

- Drop EF, trivial DPO mechanism ???????????
- Drop DPO, trivial EF mechanism ???????????

Relax one at a time ...
1) RELAXING ENVY FREENESS

Envy impossible to avoid if efficiency (DPO) required
• But unfair if an agent is allocated resources while being envied

Dynamic Envy Freeness (DEF)
• If agent \( i \) envies agent \( j \), then \( j \) must have arrived before \( i \) did, and must not have been allocated any resources since \( i \) arrived

Comparison to Forward EF [Walsh ADT-11]: An agent may only envy agents that arrived after her
• Forward EF is strictly weaker
• Trivial FEF mechanism ???????????????????
MECHANISM: DYNAMIC-DRF

1. Agent $k$ arrives
2. Start with (previous) allocation of step $k-1$
3. Keep allocating to all agents having the minimum “dominant” (largest) share at the same rate
   - Until a $k/n$ fraction of at least one resource is allocated

(A constrained “water-filling” algorithm.)

Dynamic-DRF satisfies relaxed envy freeness (DEF) along with the other properties (DPO, SI, SP).
DYNAMIC-DRF ILLUSTRATED

3 agents, 2 resources

Total

(1, ε)

(ε, 1)

(1, ε)
2) RELAXING DPO

Sometimes total fairness desired

Naïve approach: Wait for all the agents to arrive and then do a static envy free and Pareto optimal allocation

- Can we allocate more resources early?

Cautious Dynamic Pareto Optimality (CDPO)

- At every step, allocate as much as possible while ensuring EF can be achieved in the end irrespective of the future demands
- Cautious-LP: a constrained water-filling mechanism

Cautious-LP satisfies relaxed dynamic Pareto optimality (CDPO) along with the other properties (EF, SI, SP).
EXPERIMENTAL EVALUATION

Initial static DRF paper has had a big effect in industry.

Now: Dynamic-DRF and Cautious-LP under two objectives:

• Maximize the sum of dominant shares (utilitarian, maxsum)
• Maximize the minimum dominant share (egalitarian, maxmin)

Comparison with provable lower and upper bounds

Data: traces of real workloads on a Google compute cell

• 7-hour period in 2011, 2 resources (CPU and RAM)
• code.google.com/p/googleclusterdata/wiki/ClusterData2011_1
EXPERIMENTAL RESULTS

Maxsum objective
100 agents

Maxmin objective
100 agents
DISCUSSION

Relaxation: allowing zero demands

- Trivial mechanisms for SI+DPO+SP no longer work
- Open question: possibility of SI+DPO+SP in this case

Allowing agent departures and revocability of resources

- No re-arrivals → same mechanism (water-filling) for freed resources
- Departures with re-arrivals
  - Pareto optimality requires allocating resources freed on a departure
  - Need to revoke when the departed agent re-arrives
NEXT CLASS:
COMBINATORIAL ASSIGNMENT PROBLEMS