Matching with Happy Endings

Sublinear Metric Distortion for Matching

Joint with Moses Charikar (Stanford), Alireza Haqi (Stanford), Yichen Huang (Harvard), Prasanna Ramakrishnan (Harvard)






Jabari Hastings | *Pre-STOC 2027*

Matching is Fundamental

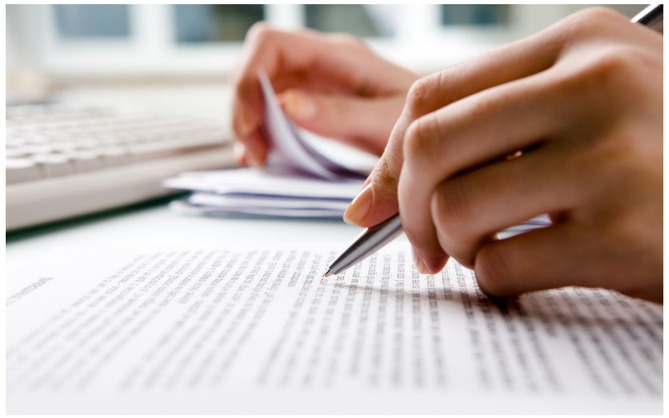
- Matching mechanisms are everywhere:



Students to Schools



Instructors to Courses



Reviewers to Papers

...

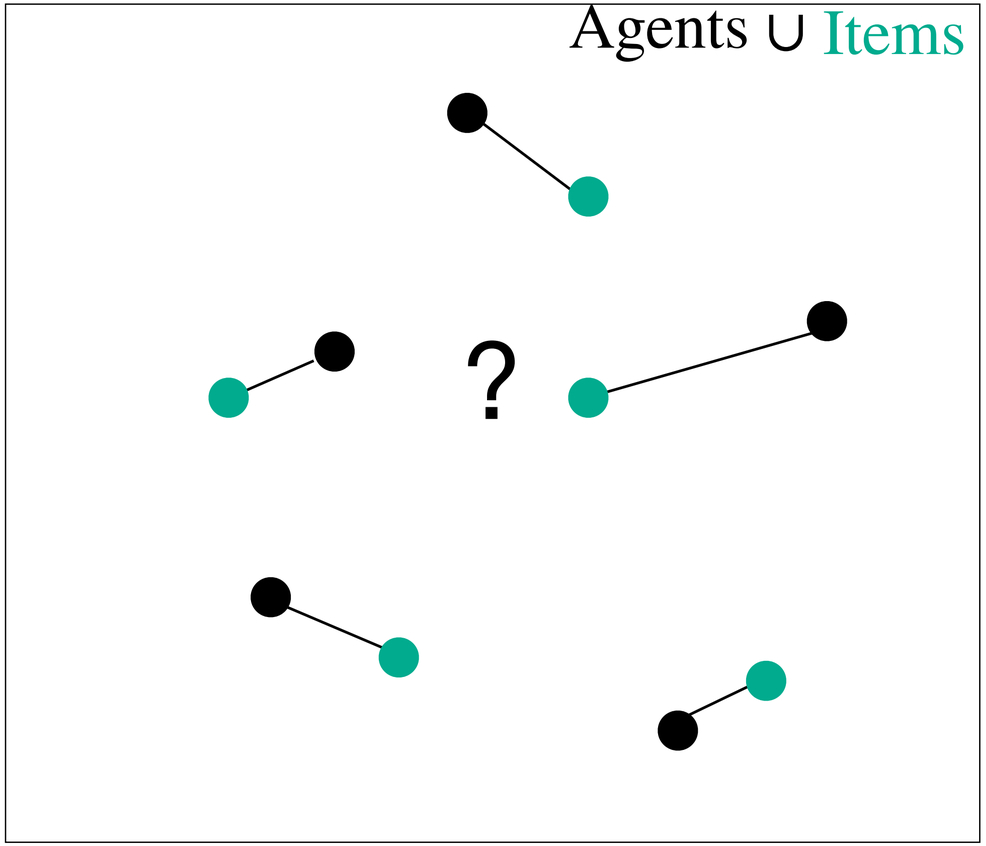
- They often rely on **partial information** about preferences (e.g., rankings).
- They often optimize outcomes based on latent **underlying values**.
- There is often a **spatial** way to reason about preferences.

Can we leverage this spatial reasoning?

Our Problem

Metric Matching

- There is some metric space d defined on **agents** $A = \{a_1, \ldots, a_n\}$ and **items** $B = \{b_1, \ldots, b_n\}$.
- The **cost** of the item b for agent a_i is given by the distance $d(a_i, b)$.
- **Goal:** We want a perfect matching $M : A \to B$ with low cost (i.e., the **sum** of assigned costs).
- **Caveat:** We do not know the metric! We are only given the agents' rankings of items based on distance.



Our Problem

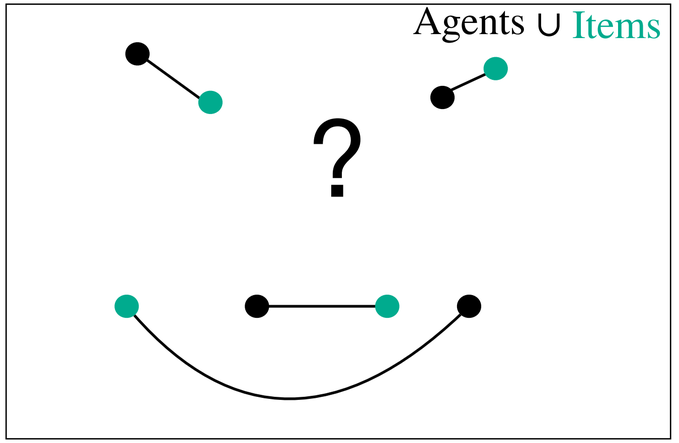
Metric Matching with Preferences

- **Agents** $A = \{a_1, \ldots, a_n\}$ and **Items** $B = \{b_1, \ldots, b_n\}$.
- We are given a **preference table** σ that consists of each agent's ranking over all the items.
- **Goal:** We want a matching M that approximates the optimal matching M^* in <u>every *consistent metric* d</u>:

$$\text{cost}(M, d) \leq \alpha \ \text{cost}(M^*, d) \qquad \forall d \text{ consistent}.$$

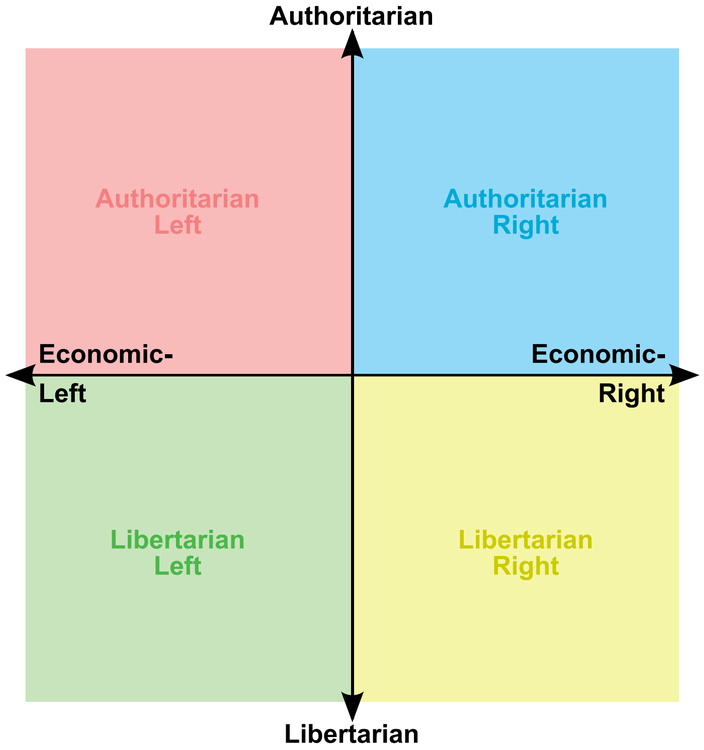
- The approximation factor α is called the **distortion**. *Smaller factors are more preferable.*

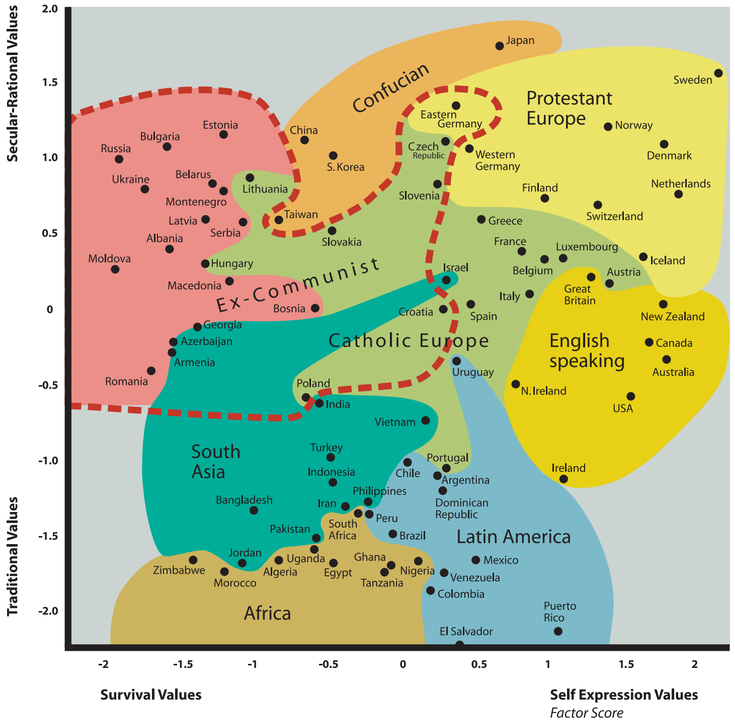
$$\begin{aligned} a_1 &: b_1 \succ b_2 \succ b_3 \succ b_4 \\ a_2 &: b_2 \succ b_1 \succ b_3 \succ b_4 \\ a_3 &: b_4 \succ b_3 \succ b_2 \succ b_1 \\ a_4 &: b_4 \succ b_3 \succ b_1 \succ b_2 \end{aligned}$$



Why the metric assumption…

- Preferences are often conveyed along some continuum (e.g., line, plane).





- Simultaneous approximation across all metric spaces indicates **robustness**.

By Koyos - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=5459884
By Original: AnonymousTranslation: Zemxer (Türkçe) and OmegaFallon - This file was derived from: Political Compass yLR wo text.svg:, Public Domain, https://commons.wikimedia.org/w/index.php?curid=79649927

[FGLRV ’25] give optimal **constant** distortion on the line.

See also [HR ’25] and [GKRSV ’26] for related settings.

Metric Distortion

Prior Work on Matching



$\Theta(2^n)$ **Serial Dictatorship (SD)** [CFFHT ’16, ’24]

$\Omega(2^{\sqrt{n}})$ **Boston Mechanism** [ACR ’23]

$O(n^2)$ **RepMatch Mechanism** [ACR ’23]

$O(n)$ **Random Serial Dictatorship (RSD)** [CFFHT ’16, ’24]

$\Omega(\log n)$ ***Any* Mechanism** [ACR ’23]

Fact. [HOG ’26]

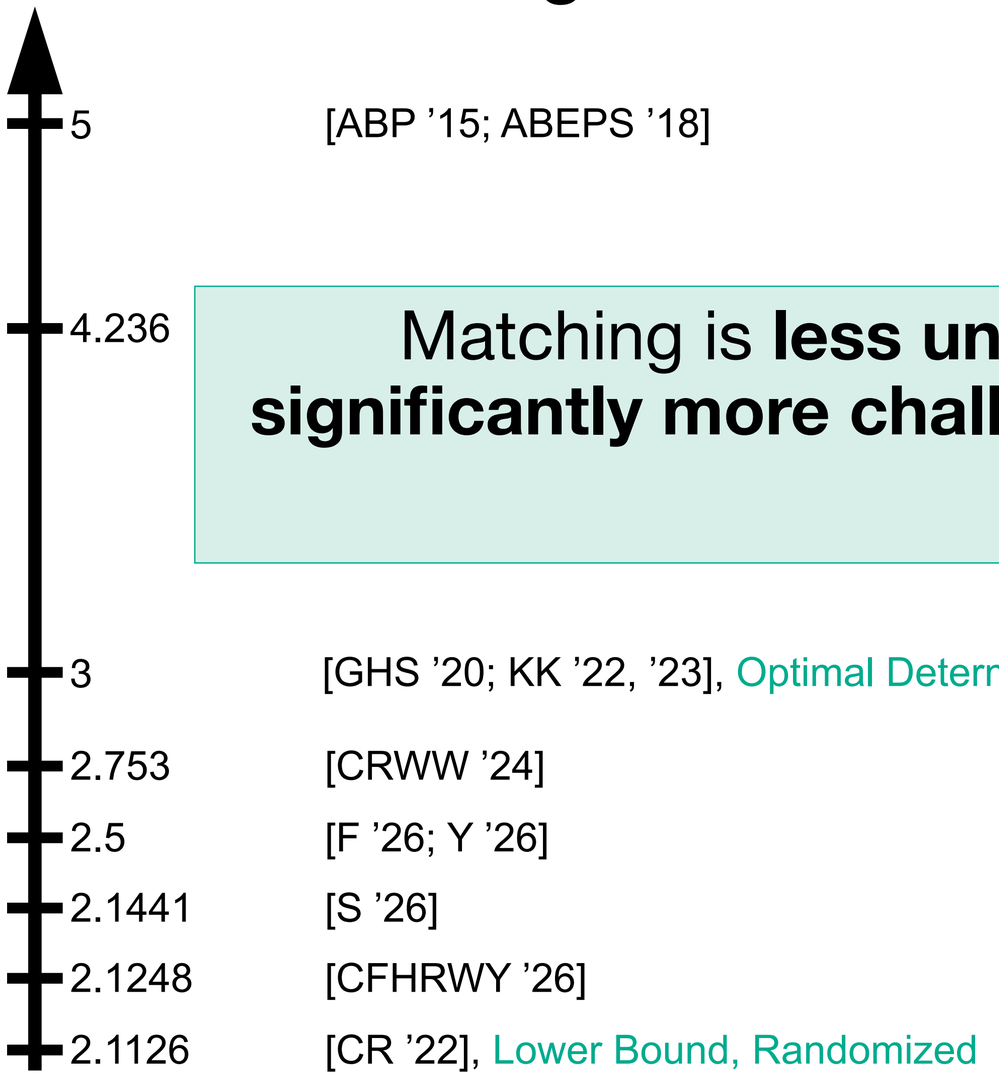
Any perfect fractional matching with distortion D can be **derandomized** to a perfect integral matching with distortion $D \cdot \operatorname{poly}\log(n)$.

Open Question.

Is there a matching mechanism with $n^{1-\varepsilon}$ distortion?

Over the same period…

Tremendous Progress in *Voting*



Alternative preference elicitation
[AAZ ’19; K ’20; AFP ’22; EHM ’24]
[AFJV ’24, ’25; AALS ’26; SZ ’26]
[FP ’26; CCW ’26]

[GAX ’17; FGMP ’19; FFM ’20; AFP ’22]

Deliberation
[FGMS ’17; GGM ’25; MYZ ’26]

Bounded randomness
[CGRW ’26; JP ’26]

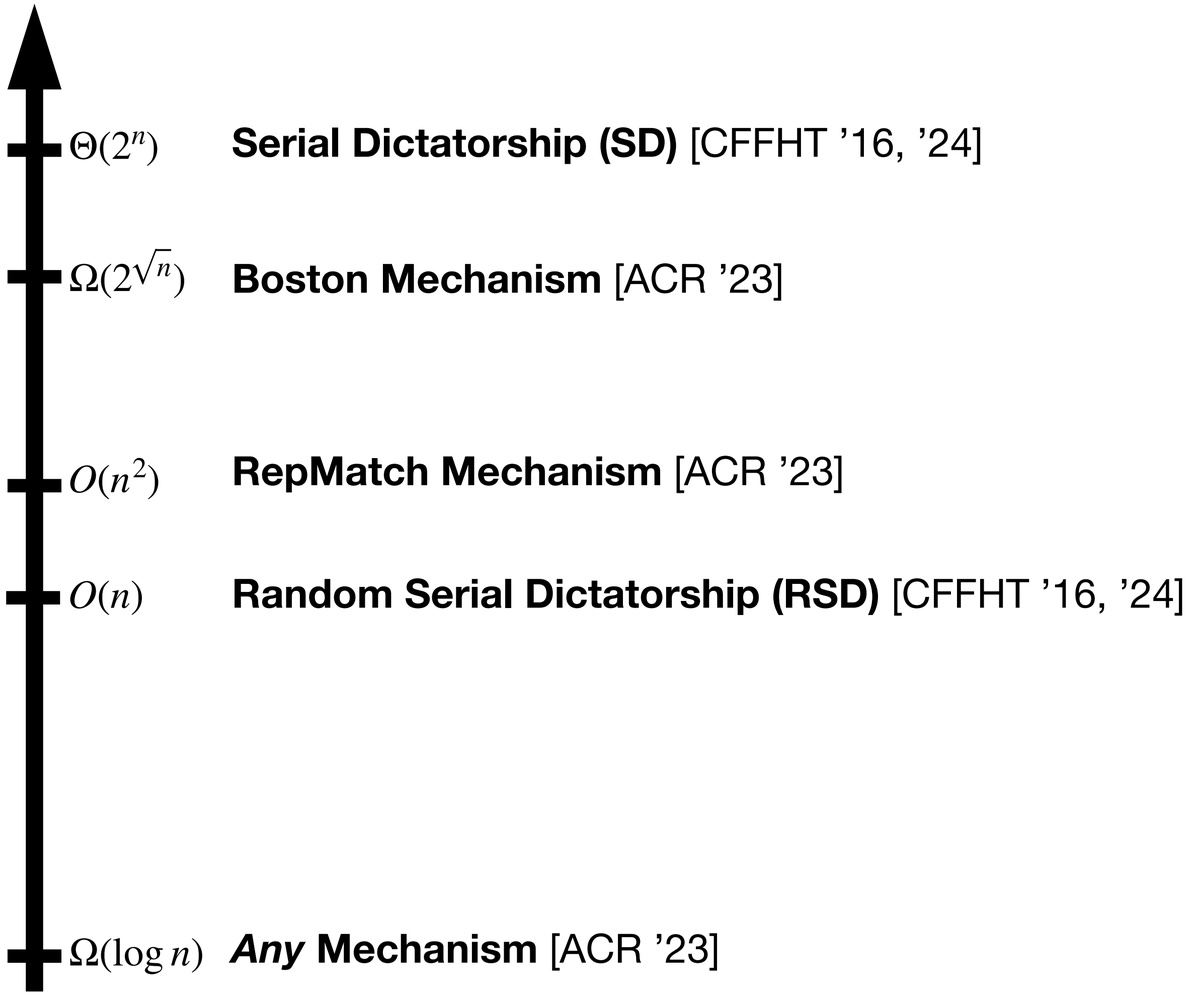
Matching is **less understood** and appears to be **significantly more challenging** in the distortion literature.

[FGLRV ’25] give optimal **constant** distortion on the line.

See also [HR ’25] and [GKRSV ’26] for related settings.

Distortion for Matching

Prior Work



Fact. [HOG ’26]

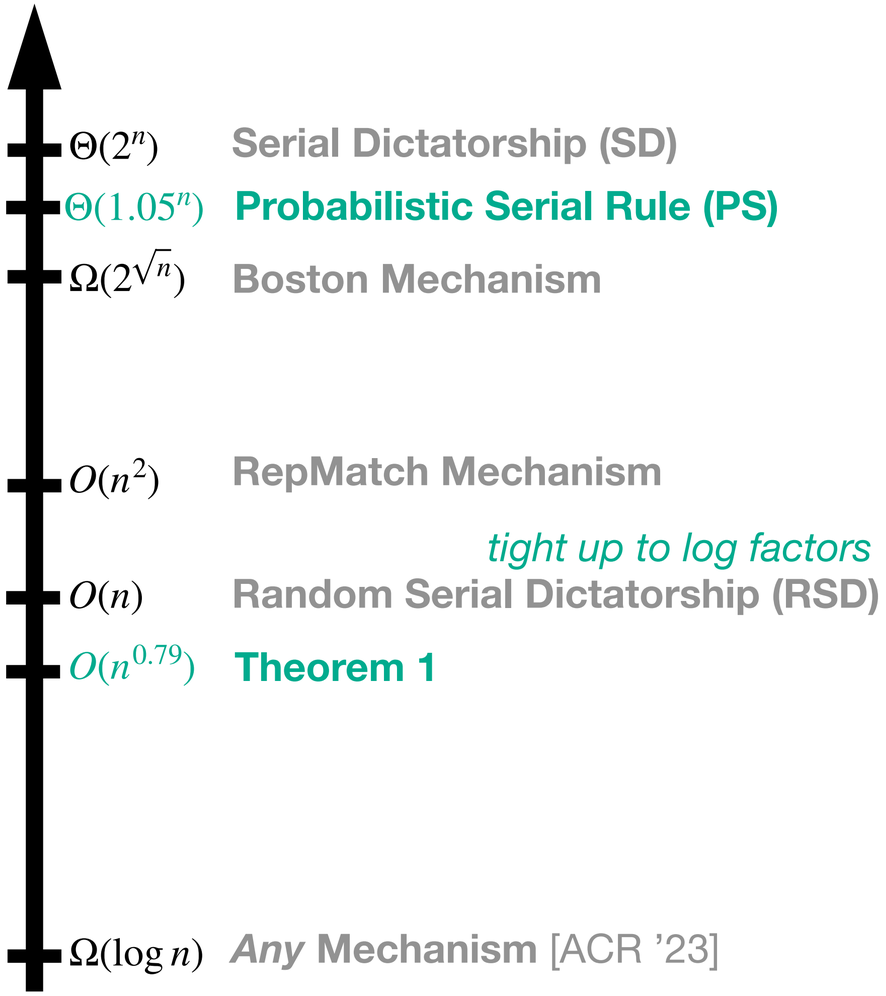
Any perfect fractional matching with distortion D can be **derandomized** to a perfect integral matching with distortion $D \cdot \operatorname{poly}\log(n)$.

Open Question.

Is there a matching mechanism with $n^{1-\varepsilon}$ distortion?

Distortion for Matching

Our Results



Theorem 1. (Sublinear Distortion)

There exists a randomized mechanism with distortion $O(n^{0.79})$.

Our approach can use either of two central matching mechanisms:

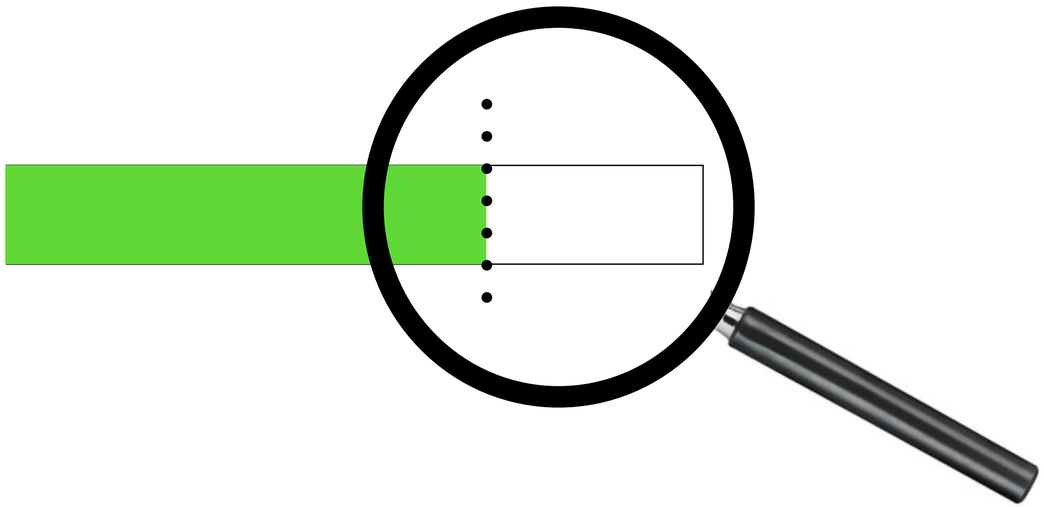
- Random Serial Dictatorship (RSD)
- Probabilistic Serial (PS) rule

Neither achieves the stated sublinear guarantee on its own.

Theorem 2. (Distortion of PS and RSD)

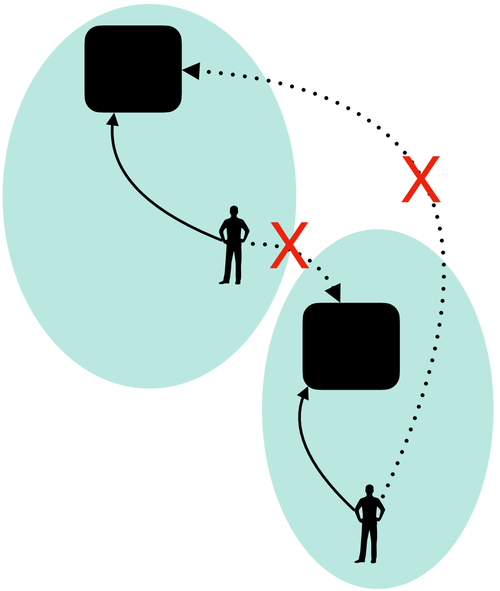
1. RSD can have distortion $\Omega(n/\sqrt{\log n})$.
2. PS can have distortion $\Theta(1.05^n)$.

Road Map



Part I: The Ending Matters

Mechanisms like RSD and PS make costly decisions in the very late stages.

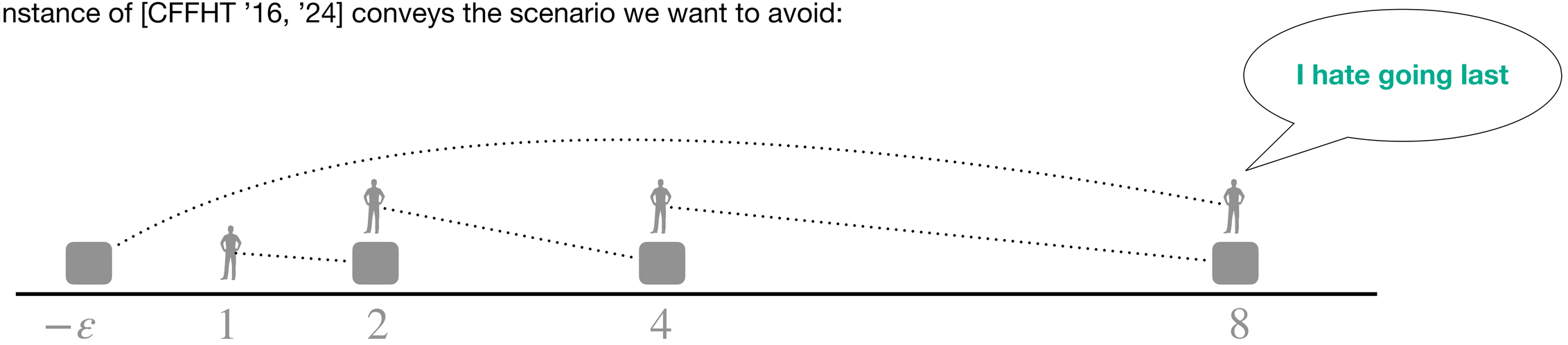


Part II: Finding A Happy Ending

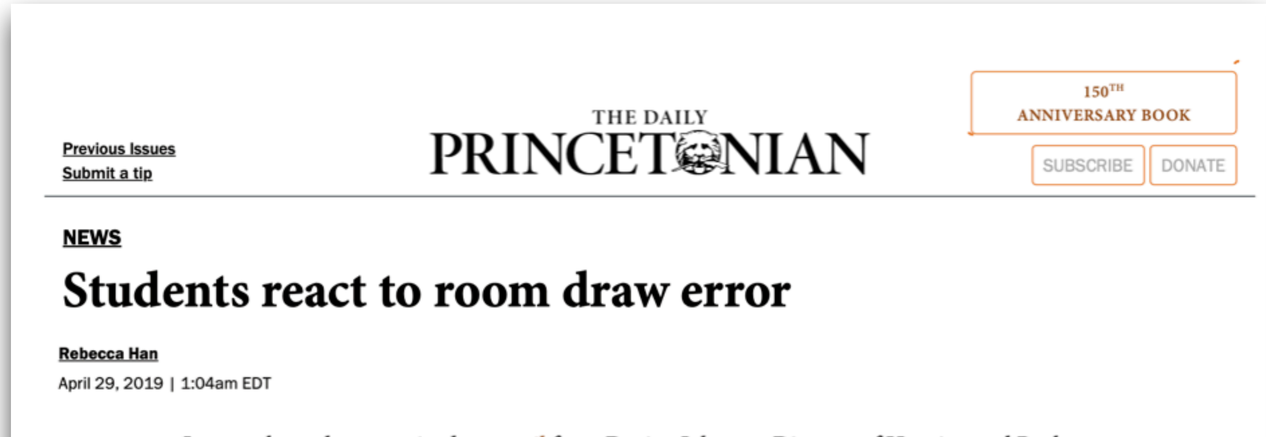
We **complete** the partial progress of RSD and PS in a more systematic way.

A Bad Ending

A line instance of [CFFHT ’16, ’24] conveys the scenario we want to avoid:



Going last is a nightmare scenario for on-campus housing assignments. **How badly can RSD and PS perform?**


Previous Issues
Submit a tip

THE DAILY PRINCETONIAN

150TH ANNIVERSARY BOOK

SUBSCRIBE DONATE

NEWS

Students react to room draw error

Rebecca Han

April 29, 2019 | 1:04am EDT

A Bad Ending for PS

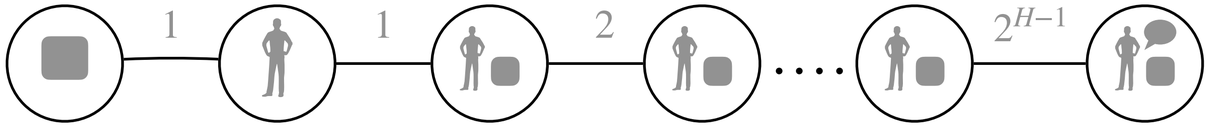
Algorithm (PS) [BM '01]

Each item has capacity 1.

In parallel, every agent eats her favorite available item at an equal rate until all items are exhausted.

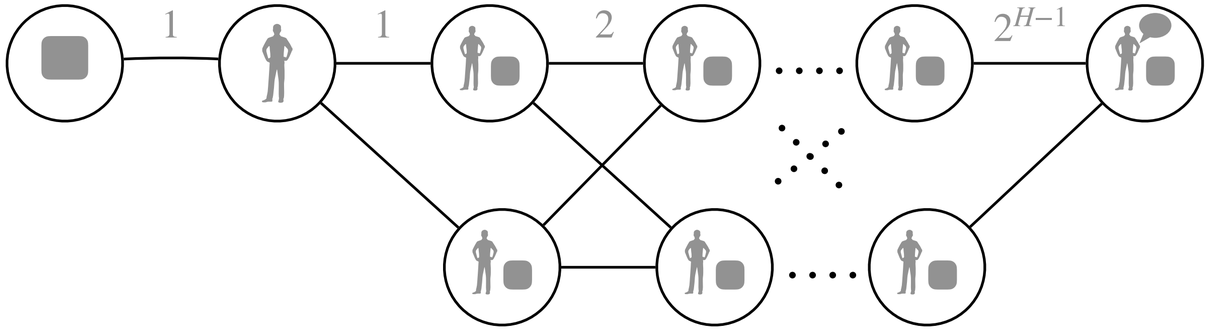
Match according to fractional shares.

Observation 1: A costly scenario already occurs on the line.



Each agent incurs cost at least $1/2$. Total is at least $n/2$

Observation 2: Branching can amplify the cost for the rightmost agent.



Agents favor items to the right and in the upper row.

Within same vertical layer, upper item is exhausted first.

Cost propagates more quickly to the rightmost agent:

- Rate is $5/9$ per layer, but distances double.
- Cost for rightmost agent scales by $10/9$ per layer.

A Bad Ending for PS

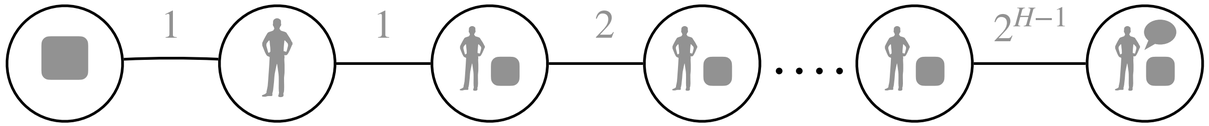
Algorithm (PS) [BM '01]

Each item has capacity 1.

In parallel, every agent eats her favorite available item at an equal rate until all items are exhausted.

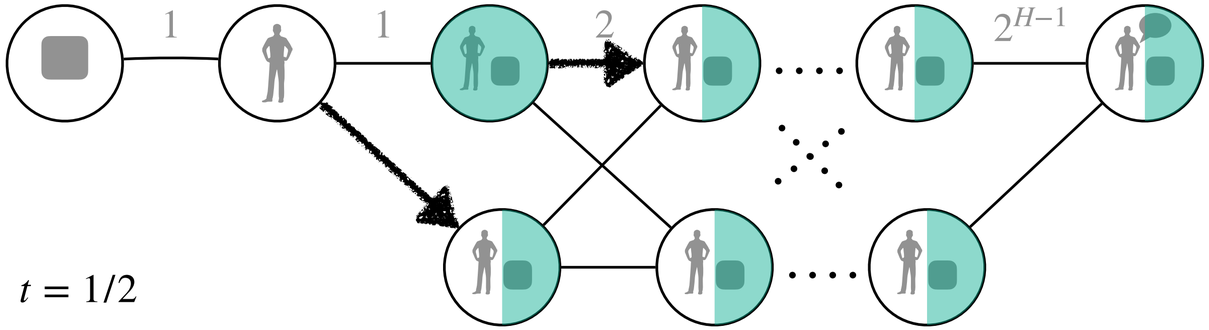
Match according to fractional shares.

Observation 1: A costly scenario already occurs on the line.



Each agent incurs cost at least $1/2$. Total is at least $n/2$

Observation 2: Branching can amplify the cost for the rightmost agent.



Within same vertical layer, upper item is exhausted first.

Cost propagates more quickly to the rightmost agent:

- Rate is $5/9$ per layer, but distances double.
- Cost for rightmost agent scales by $10/9$ per layer.

Agents favor items to the right and in the upper row.

A Bad Ending for PS

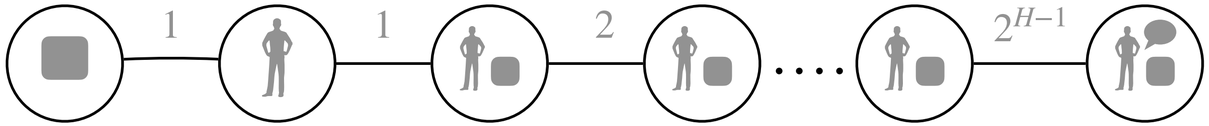
Algorithm (PS) [BM '01]

Each item has capacity 1.

In parallel, every agent eats her favorite available item at an equal rate until all items are exhausted.

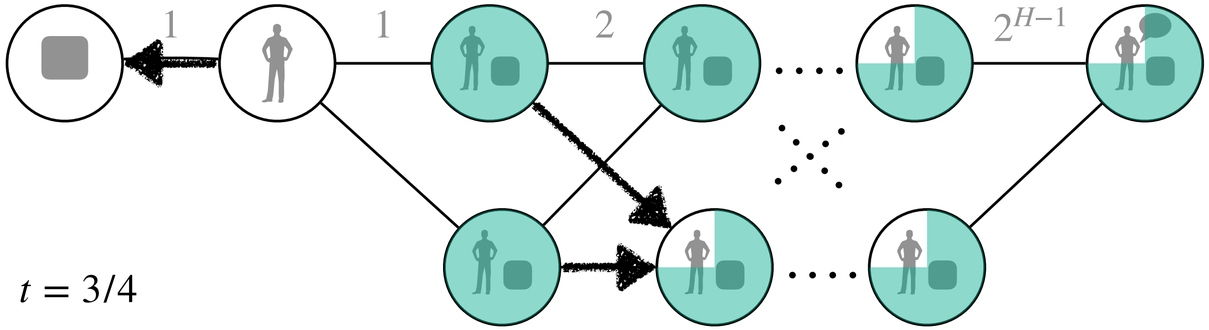
Match according to fractional shares.

Observation 1: A costly scenario already occurs on the line.



Each agent incurs cost at least $1/2$. Total is at least $n/2$

Observation 2: Branching can amplify the cost for the rightmost agent.



Within same vertical layer, upper item is exhausted first.

Cost propagates more quickly to the rightmost agent:

- Rate is $5/9$ per layer, but distances double.
- Cost for rightmost agent scales by $10/9$ per layer.

Agents favor items to the right and in the upper row.

A Bad Ending for PS

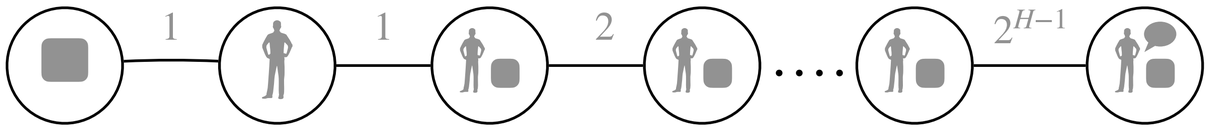
Algorithm (PS) [BM '01]

Each item has capacity 1.

In parallel, every agent eats her favorite available item at an equal rate until all items are exhausted.

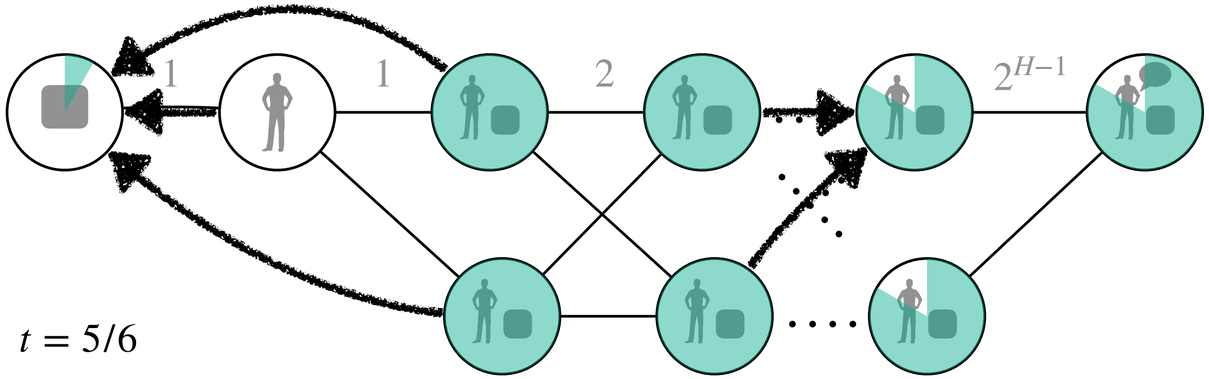
Match according to fractional shares.

Observation 1: A costly scenario already occurs on the line.



Each agent incurs cost at least $1/2$. Total is at least $n/2$

Observation 2: Branching can amplify the cost for the rightmost agent.



Within same vertical layer, upper item is exhausted first.

Cost propagates more quickly to the rightmost agent:

- Rate is $5/9$ per layer, but distances double.
- Cost for rightmost agent scales by $10/9$ per layer.

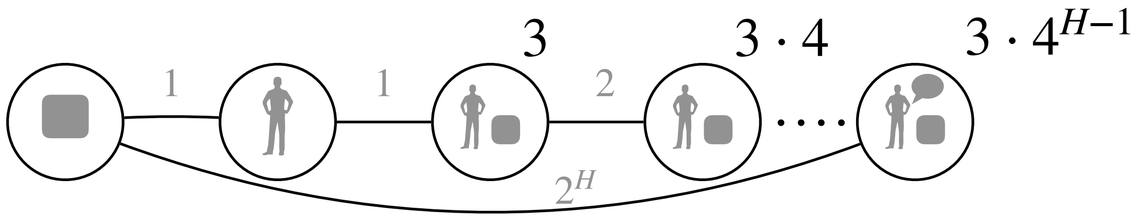
$$\text{cost(PS)} \geq (10/9)^{n/2} \approx 1.05^n$$

Agents favor items to the right and in the upper row.

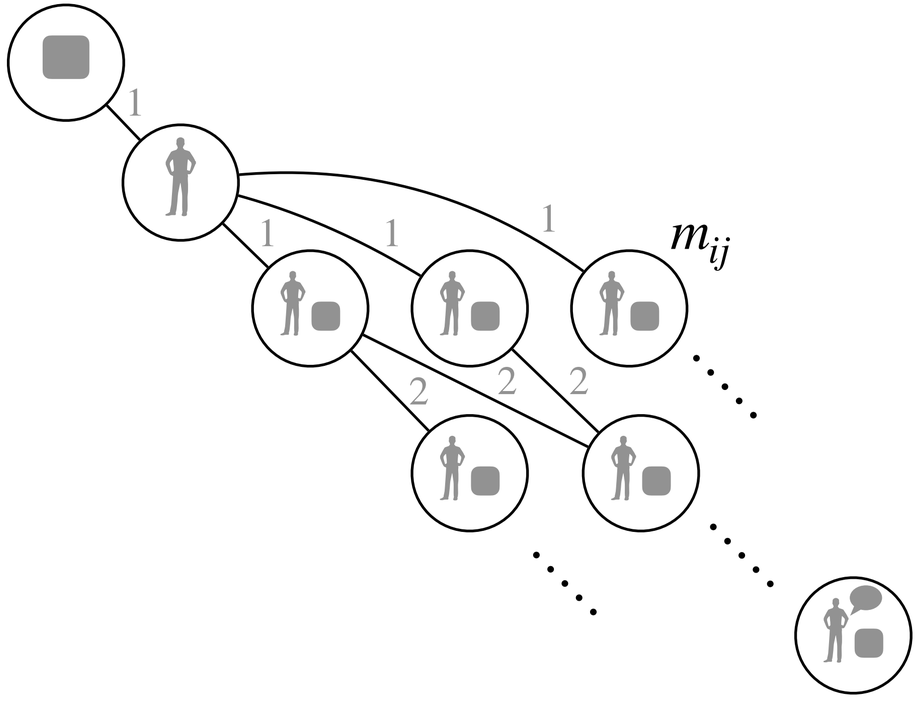
A Bad Ending for RSD

Algorithm (RSD)
Assign each agent a random priority in [0, 1].
Match each agent to her favorite available item on her turn.

Observation 1: Cloning can hurt RSD [CFFHT ’16, ’24]



Observation 2: Branching also hurts RSD.



The left steals from the right.

Expected number of “long chains” is at least $(3/4)^{H-1}$.

$$\text{cost(RSD)} \geq (3/4)^{H-1} \cdot 2^H = n^{\log_4(3/2)}$$

Parents steal from children.

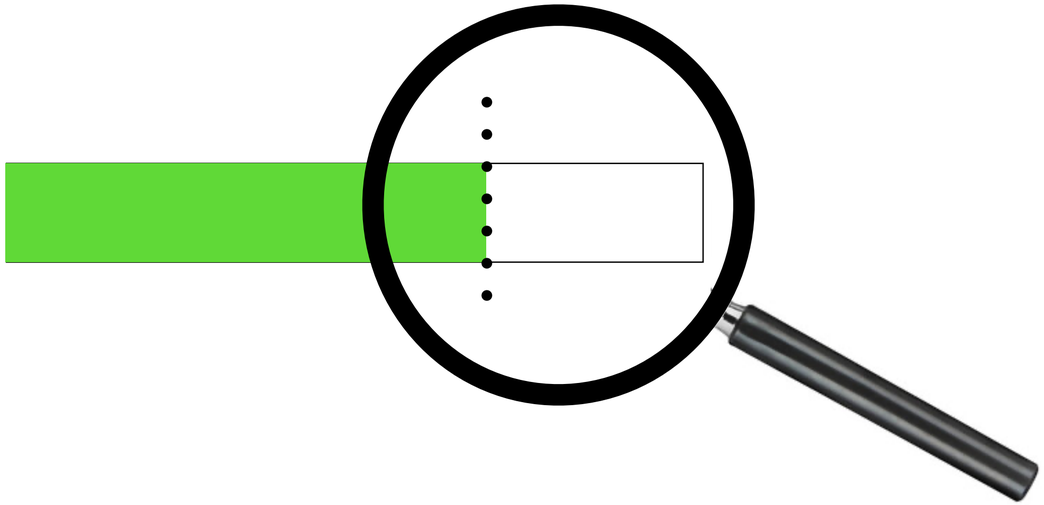
Cloning is significantly more involved.

- Emulate a *Poisson* branching process
- **Prune** paths that contribute little to deepest levels.
- **Prune** masses of agents/items to keep instance small.

For carefully tuned parameters:

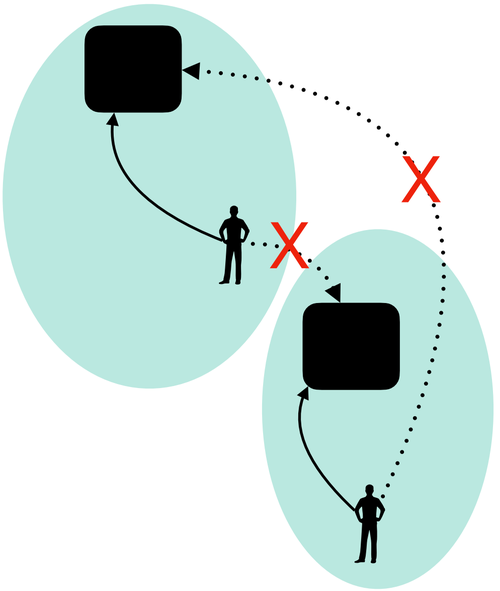
- **Expected number of “long chains” is at least 1.**
- Can show $\text{cost(RSD)}/\text{OPT} \geq \Omega(n/\sqrt{\log n})$.

Road Map



Part I: The Ending Matters

Mechanisms like RSD and PS make costly decisions in the very late stages.



Part II: Finding A Happy Ending

We **complete** the partial progress of RSD and PS in a more systematic way.

Finding a Happy Ending

Suppose we have k agents left to assign.

- It would be nice to avoid further cascades…
- It would be nice to have a set of fallback options for each agent.

RepMatch [ACR '23, HR '25] could provide these fallback options via its **balanced partitions** $\mathscr{P} = (A_j, B_j)_{i=1}^{m}$.

Algorithm (RepMatch). [ACR '23, HR '25]

Start with representatives $\{r\}_{r \in A}$, each claiming their favorite item.

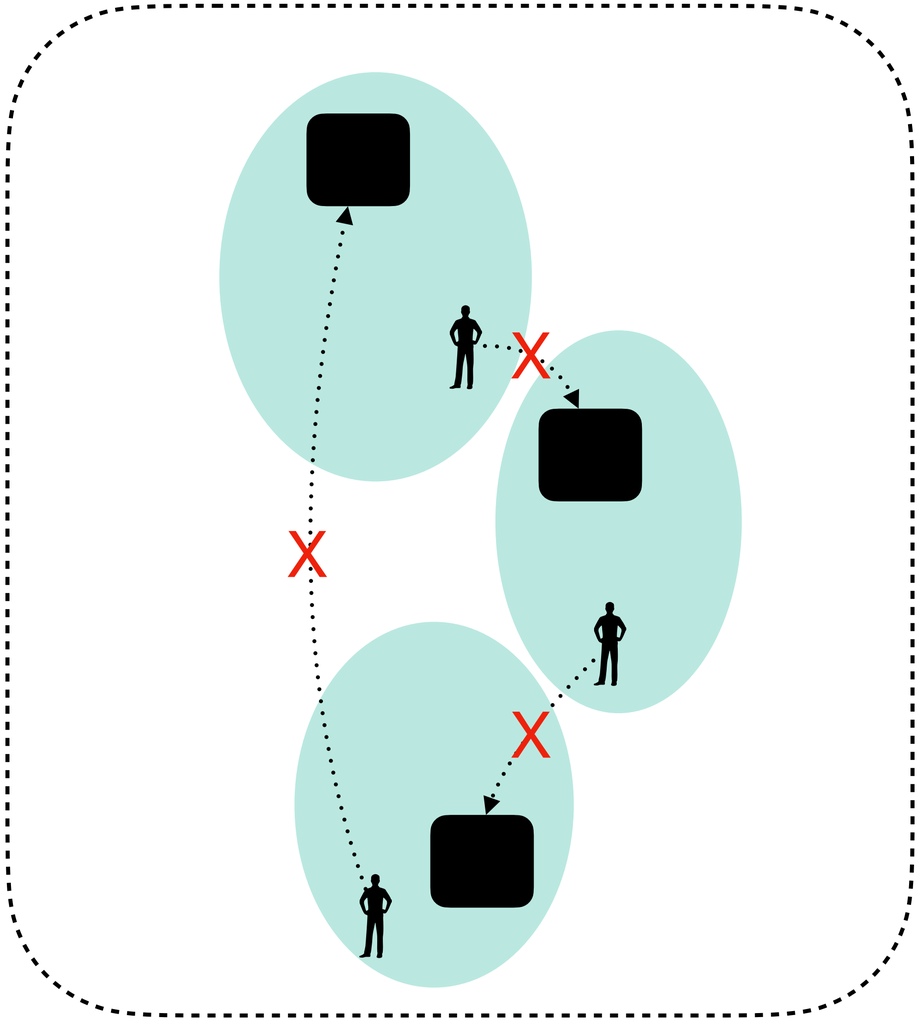
While there exist clusters S, T with overlapping claimed items:

- Merge S and T.
- The representative of $S \cup T$ is ~~FixedRule(r_S, r_T)~~ RandomRule(r_S, r_T).
- New representative claims $|S| + |T|$ favorite items.

Enumerate clusters $A_1, \ldots, A_m$. Set B_j to be r_{A_j}'s claimed items.

Return $\mathscr{P} = (A_j, B_j)_{i=1}^{m}$

Free agents could be assigned to free items in their partition!

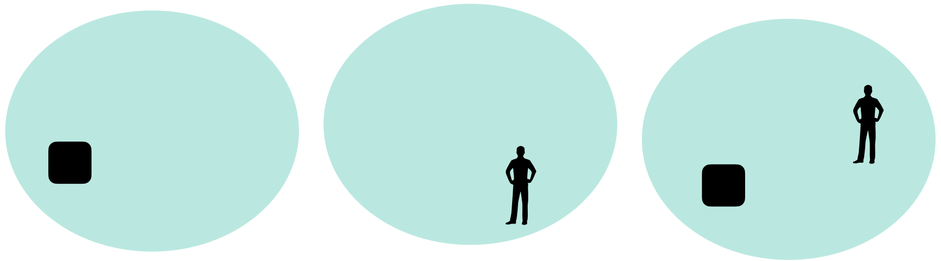


Metric Space d

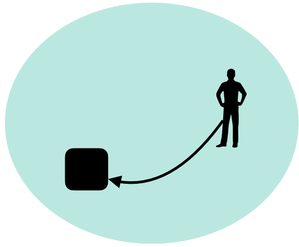
Finding a Happy Ending

Cluster Completion

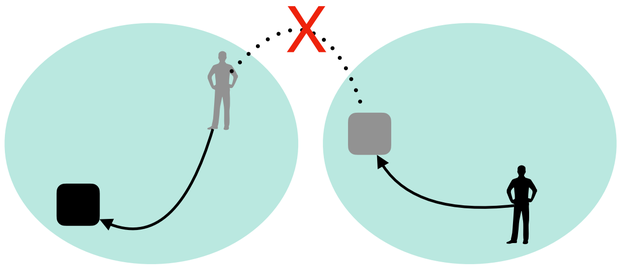
1. Identify $\mathscr{P} = (A_j, B_j)_{i=1}^{m}$ for the k free agents/items.



2. Assign every free agent to a free item in her partition, where applicable.



3. If a free agent has no free item in her partition, then create an **alternating path** by successively displacing an agent that does not belong to the partition.



<u>Bookkeeping</u>

Each of the k last assignments costs at most $\sum_{j=1}^{m} \max_{a \in A_j, b \in B_j} d(a,b)$.

Our variant of *RepMatch* gives a <u>very strong guarantee</u>:

$$W(\mathscr{P}, d) := \mathbb{E}\left[\sum_{j=1}^{m} \max_{a \in A_j, b \in B_j} d(a,b)\right] = O(n^{0.58})\ \text{OPT}$$

Hence, completion causes an increase of only $k \cdot W(\mathscr{P}, d)$.

Finding a Happy Ending

We derive the following general result.

Theorem.

Fix a partial integral matching $\widehat{M}$ leaving k free agents and a balanced partition $\mathscr{P}$. If $M \leftarrow \mathrm{ClusterCompletion}(\hat{M}, \mathscr{P})$, then $\mathrm{cost}(M, d) \leq \mathrm{cost}(\widehat{M}, d) + kW(\mathscr{P}, d)$

Building on [ACR '23] we can **find a <u>happy ending for RSD.</u>**

Theorem. [ACR '23]

$$\mathrm{cost}(\mathrm{TruncRSD}(\sigma, k)) \leq \frac{n-k}{k+1}\ \mathrm{OPT}\,.$$

Completing $\mathrm{TruncRSD}$ costs at most:

$$\frac{n-k}{k+1}\ \mathrm{OPT} + kW(\mathscr{P}, d)$$

We show:

$$W(\mathscr{P}, d) \leq n^{0.58}\ \mathrm{OPT}\,.$$

$k = \Theta(n^{0.21})$ implies distortion $O(n^{0.79})$.

Finding a Happy Ending

We derive an even general result using flows!

Theorem.

Fix a **fractional** matching $\widehat{M}$ leaving **a mass of** k free agents and a balanced partition $\mathscr{P}$. If $M \leftarrow \text{ClusterCompletion}(\hat{M}, \mathscr{P})$, then $\text{cost}(M, d) \leq \text{cost}(\widehat{M}, d) + kW(\mathscr{P}, d)$

We also prove a guarantee for a truncated version of PS.

Theorem.

$$\text{cost}(\text{TruncPS}(\sigma, k), d) \leq \frac{n-k}{k} \text{ OPT}.$$

Algorithm 2. $\text{TruncPS}(\sigma, k)$

- Run $\text{PS}(\sigma)$ for until time $1 - \frac{k}{n}$.
- Return the resulting assignment Y.

Completing TruncPS costs at most:

$$\frac{n-k}{k} \text{ OPT} + W(\mathscr{P}, d).$$

We show:

$$W(\mathscr{P}, d) \leq n^{0.58} \text{ OPT}.$$

$k = \Theta(n^{0.21})$ implies distortion $O(n^{0.79})$.

Our General Recipe

Theorem.

Fix a partial matching $\widehat{M}$ leaving k free agents and a balanced partition $\mathscr{P}$. Let $M \leftarrow \text{ClusterCompletion}(\widehat{M}, \mathscr{P})$.

Then $\text{cost}(M, d) \leq \text{cost}(\widehat{M}, d) + kW(\mathscr{P}, d)$.

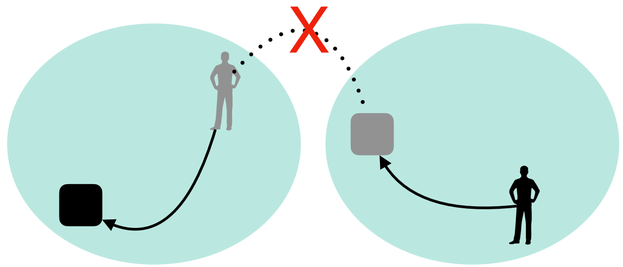


$$\text{cost}(\text{TruncALG}(k), d) \leq f(k)\,\text{cost}(M^*, d)$$

Truncate the mechanism, and give strong guarantees for partial assignment

[ACR '23] do precisely this for RSD.

We can also do this for PS!



Complete the partial assignment using preprocessed clusters

We can ensure $W(\mathscr{P}, d) \leq n^{0.58}$ with the randomized version of RepMatch

Conclusion

RSD and PS do not have distortion $n^{1-\varepsilon}$ on their own

Theorem. (Distortion of PS and RSD)

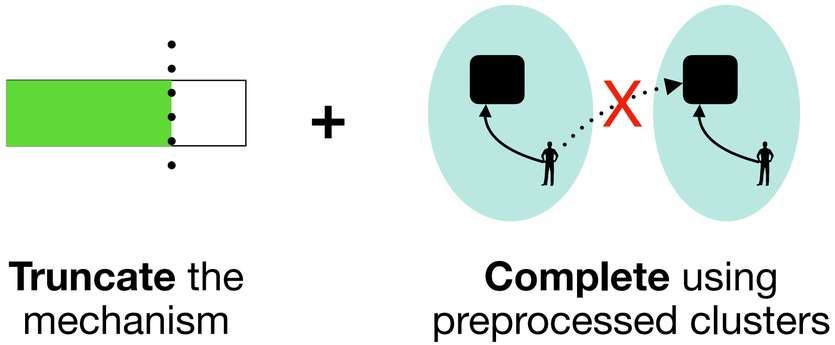
1. RSD can have distortion $\Omega(n/\sqrt{\log n})$.
2. PS has distortion $\Theta((10/9)^{n/2})$.

…but they can be repaired to get sublinear distortion.

Theorem. (Sublinear Distortion)

There exists a randomized mechanism with distortion $O(n^{0.79})$.

We provide a general recipe for mending mechanisms



Truncate the mechanism

Complete using preprocessed clusters

… and establish many more result in the full paper!

Other Theorems.

1. Coordinated merging (in the style of RepMatch) step can also get sublinear distortion!
2. An improved lower bound for ***truthful*** mechanisms, from $\Omega(\log n)$ to $2^{\Omega(\sqrt{\log n})}$.