Online Matching with High Probability

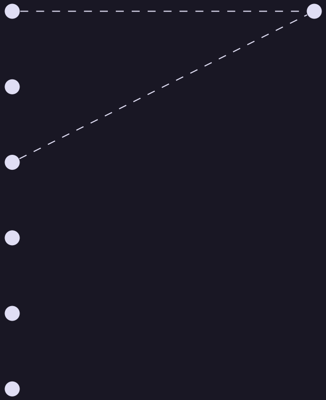
Milena Mihail, **Thorben Tröbst**

Symposium on Algorithmic Game Theory

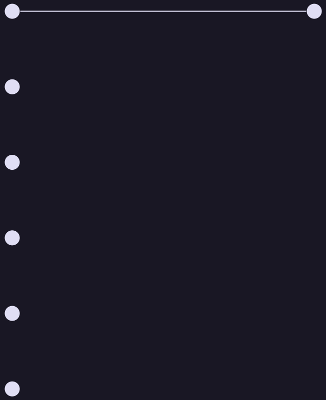
Online Bipartite Matching

Online Bipartite Matching

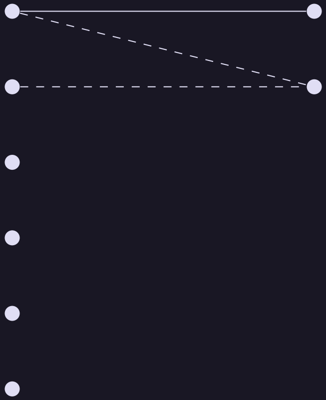
Online Bipartite Matching



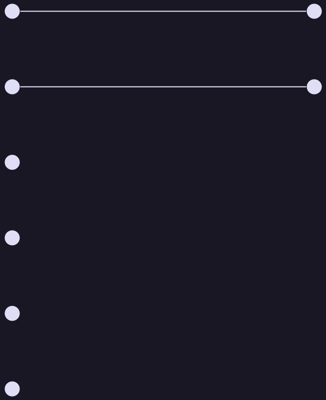
Online Bipartite Matching



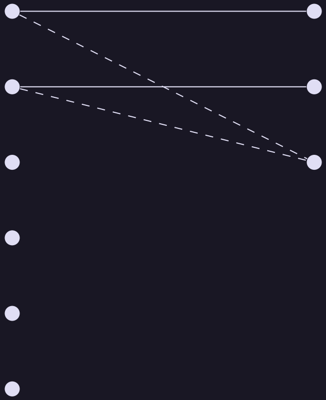
Online Bipartite Matching



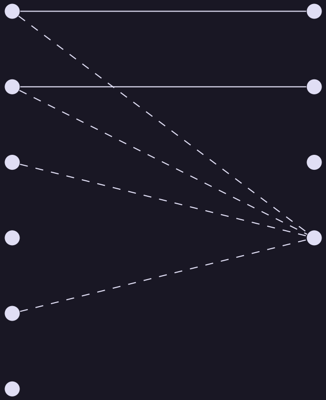
Online Bipartite Matching



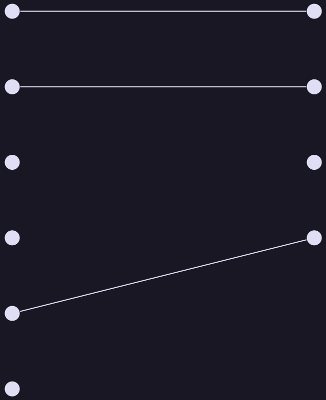
Online Bipartite Matching



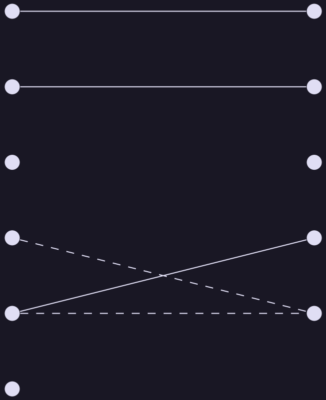
Online Bipartite Matching



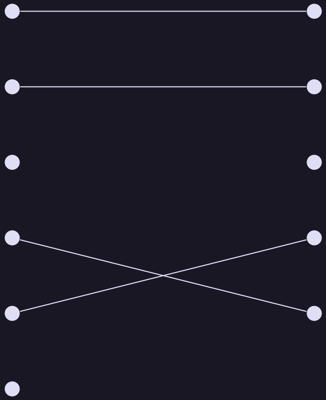
Online Bipartite Matching



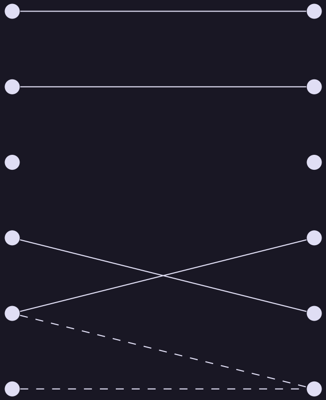
Online Bipartite Matching



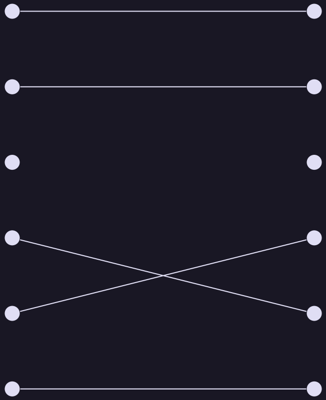
Online Bipartite Matching



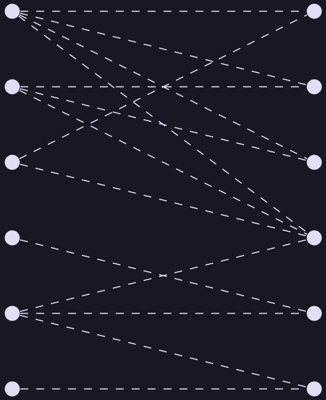
Online Bipartite Matching



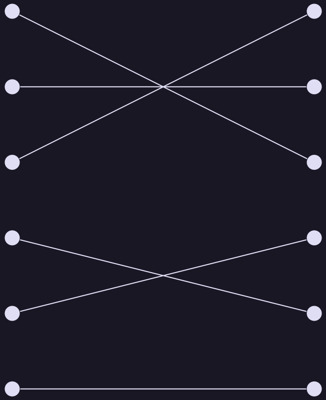
Online Bipartite Matching



Online Bipartite Matching



Online Bipartite Matching



- $G = (S, B, E)$ is a bipartite graph consisting of offline vertices S and online vertices B.

Online Bipartite Matching II

- $G = (S, B, E)$ is a bipartite graph consisting of offline vertices S and online vertices B.
- Online vertices arrive one by one in adversarial order.

Online Bipartite Matching II

- $G = (S, B, E)$ is a bipartite graph consisting of offline vertices S and online vertices B.
- Online vertices arrive one by one in adverserial order.
- The algorithm must irrevocably and immediately match revealed online vertices.

- $G = (S, B, E)$ is a bipartite graph consisting of offline vertices S and online vertices B.
- Online vertices arrive one by one in adverserial order.
- The algorithm must irrevocably and immediately match revealed online vertices.
- The goal is to maximize the competitive ratio, i.e.

$$\frac{|M_{\text{online}}|}{\text{OPT}_{\text{offline}}}.$$

Classic results for Online Bipartite Matching:

Classic results for Online Bipartite Matching:

- The Greedy algorithm (match whenever possible) is $1/2$-competitive.

Classic results for Online Bipartite Matching:

- The Greedy algorithm (match whenever possible) is $1/2$-competitive.
- $1/2$-competitive is best possible for deterministic algorithms.

Classic results for Online Bipartite Matching:

- The Greedy algorithm (match whenever possible) is $1/2$-competitive.
- $1/2$-competitive is best possible for deterministic algorithms.
- The randomized Ranking algorithm is $(1 - 1/e)$-competitive in expectation.

Classic results for Online Bipartite Matching:

- The Greedy algorithm (match whenever possible) is $1/2$-competitive.
- $1/2$-competitive is best possible for deterministic algorithms.
- The randomized Ranking algorithm is $(1 - 1/e)$-competitive in expectation.
- $(1 - 1/e)$-competitive in expectation is best possible for randomized algorithms.

Central Question

Question

Can we solve the Online Bipartite Matching Problem with high probability as opposed to just in expectation?

Randomization and Concentration Guarantees

The Power of Randomized Algorithms

Many problems have more natural, efficient, or better algorithms using randomization:

- Quicksort
- Miller-Rabin primality test
- Hashing
- Polynomial identity testing
- Perfect matching on parallel machines
- Many online algorithms!

Expectation vs. Concentration

Usually we analyze in expectation or showing non-zero probability of success.

Expectation vs. Concentration

Usually we analyze in expectation or showing non-zero probability of success.

Example: Let C be the total number of comparisons of Quicksort with random pivots.

Expectation vs. Concentration

Usually we analyze in expectation or showing non-zero probability of success.

Example: Let C be the total number of comparisons of Quicksort with random pivots.

- Most people have seen: $\mathbb{E}[C] = O(n \log n)$.

Expectation vs. Concentration

Usually we analyze in expectation or showing non-zero probability of success.

Example: Let C be the total number of comparisons of Quicksort with random pivots.

- Most people have seen: $\mathbb{E}[C] = O(n \log n)$.
- Fewer know: $\mathbb{P}[C > c_0 \cdot n \log n] < \frac{1}{n}$ for some c_0.

Usually we analyze in expectation or showing non-zero probability of success.

Example: Let C be the total number of comparisons of Quicksort with random pivots.

- Most people have seen: $\mathbb{E}[C] = O(n \log n)$.
- Fewer know: $\mathbb{P}[C > c_0 \cdot n \log n] < \frac{1}{n}$ for some c_0.
- But did you know:

$$\mathbb{P}[|C/\mathbb{E}[C] - 1| > \epsilon] < n^{-2\epsilon(\ln\ln n - \ln(1/\epsilon) + O(\ln\ln\ln n))}$$

Usefulness of Concentration Results

Concentration results are useful:

- Insight about typical behavior in practice.
- Confidence that bad behavior is extremely unlikely.

Concentration results are useful:

- Insight about typical behavior in practice.
- Confidence that bad behavior is extremely unlikely.

However, concentration results are relatively rare because we can simply run the algorithm $O(\log n)$ many times (boosting).

Concentration results are useful:

- Insight about typical behavior in practice.
- Confidence that bad behavior is extremely unlikely.

However, concentration results are relatively rare because we can simply run the algorithm $O(\log n)$ many times (boosting).

Problem

Online algorithms cannot be boosted!

Ranking

Ranking Algorithm

Ranking by Karp, Vazirani, Vazirani (1990):

1. First, pick a random permutation π on the offline vertices.
2. On arrival: match to (currently unmatched) offline vertex j that minimizes rank $\pi(j)$.

Ranking Algorithm

Ranking by Karp, Vazirani, Vazirani (1990):

1. First, pick a random permutation π on the offline vertices.
2. On arrival: match to (currently unmatched) offline vertex j that minimizes rank $\pi(j)$.

Theorem (Karp, Vazirani, Vazirani 1990)

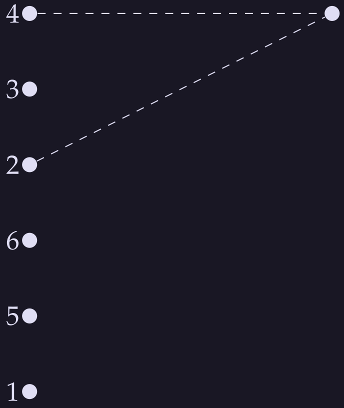
Let M be the matching generated by Ranking, then

$$\mathbb{E}[|M|] \geq \left(1 - \frac{1}{e}\right) \mathrm{OPT}.$$

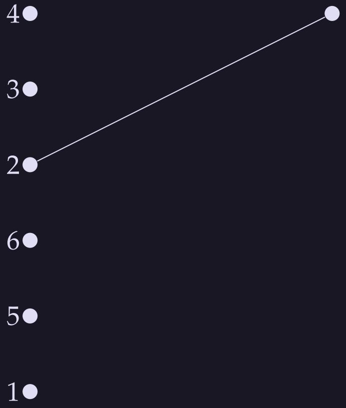
Ranking Example


4
3
2
6
5
1

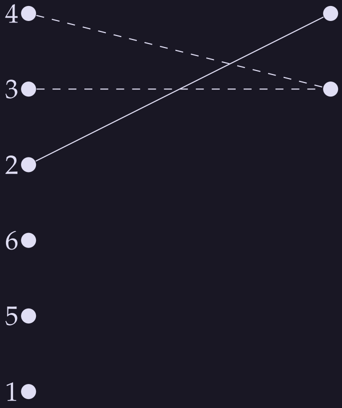
Ranking Example



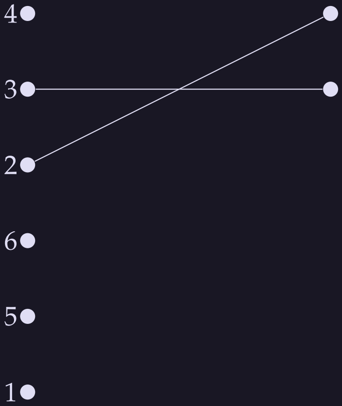
Ranking Example



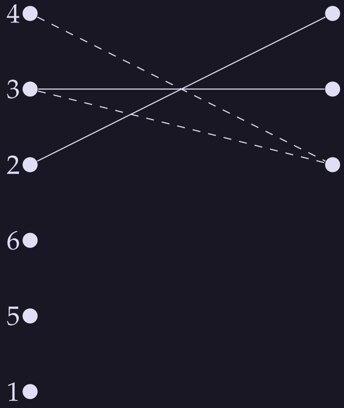
Ranking Example



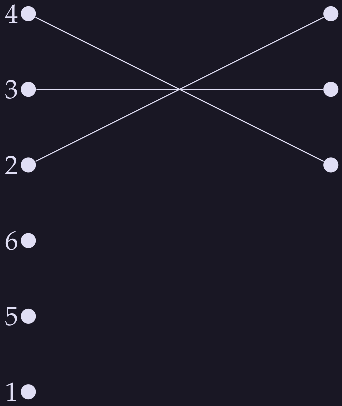
Ranking Example



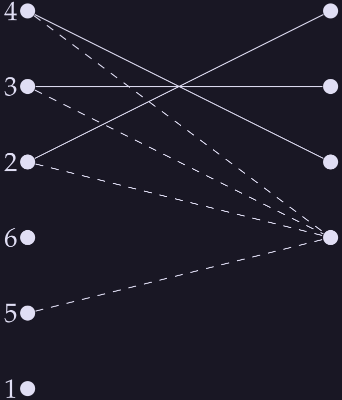
Ranking Example



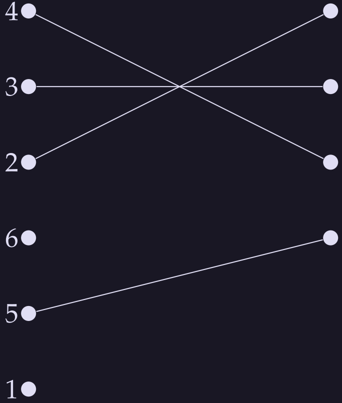
Ranking Example



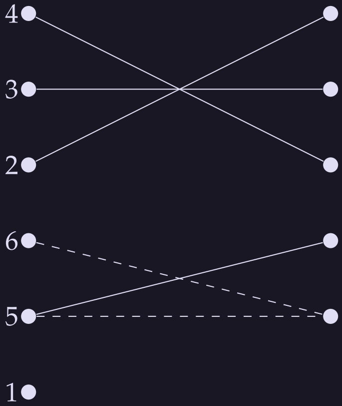
Ranking Example



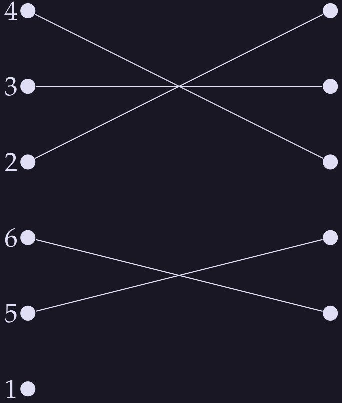
Ranking Example



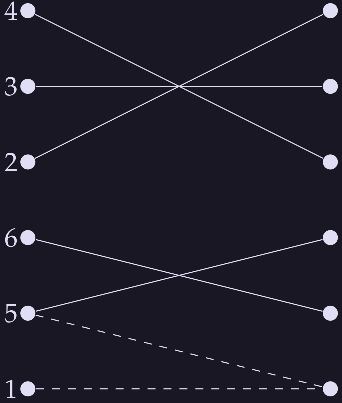
Ranking Example



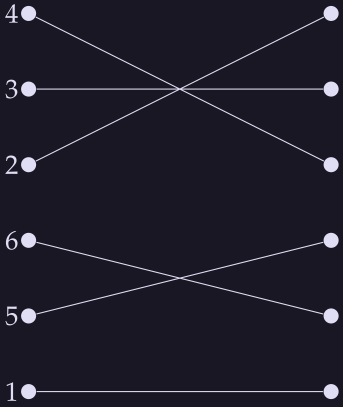
Ranking Example



Ranking Example



Ranking Example



Question

Does the competitive ratio of Ranking *hold with high probability or just in expectation?*

Question

Does the competitive ratio of Ranking hold with high probability or just in expectation?

Theorem

Let M be the matching generated by Ranking, then

$$\mathbb{P}\left[|M| < \left(1 - \frac{1}{e} - \alpha\right)\mathrm{OPT}\right] < e^{-2\alpha^2 \mathrm{OPT}}.$$

Concentration of Ranking

Theorem (McDiarmid 1989)

Let $x \in [0,1]^n$ be uniformly distributed.

Theorem (McDiarmid 1989)

Let $x \in [0,1]^n$ be uniformly distributed.

Let $f : [0,1]^n \to \mathbb{R}$ have bounded differences, i.e. there is some $c \in \mathbb{R}^n_{\geq 0}$ such that if $x, x' \in [0,1]$ disagree only on coordinate i, then $|f(x) - f(x')| \leq c_i$.

Theorem (McDiarmid 1989)

Let $x \in [0,1]^n$ be uniformly distributed.

Let $f : [0,1]^n \to \mathbb{R}$ have bounded differences, i.e. there is some $c \in \mathbb{R}^n_{\geq 0}$ such that if $x, x' \in [0,1]$ disagree only on coordinate i, then $|f(x) - f(x')| \leq c_i$.

Then:

$$\mathbb{P}\left[f(x) < \mathbb{E}[f(y)] - t\right] < e^{-\frac{2t^2}{\sum_{i=1}^n c_i^2}}.$$

Ranking can be cast in the McDiarmid framework:

Applying McDiarmid to Ranking

Ranking can be cast in the McDiarmid framework:

- Instead of picking a random permutation on the offline vertices, pick one $x_i \in [0, 1]$ for each.

Ranking can be cast in the McDiarmid framework:

- Instead of picking a random permutation on the offline vertices, pick one $x_i \in [0, 1]$ for each.
- With probability 1, all x_i are distinct and their order determines the ranks.

Ranking can be cast in the McDiarmid framework:

- Instead of picking a random permutation on the offline vertices, pick one $x_i \in [0, 1]$ for each.
- With probability 1, all x_i are distinct and their order determines the ranks.
- $f(x)$ is the size of the matching output by Ranking.

Lemma

f satisfies bounded differences with $c_i \equiv 1$.

Theorem

Assuming $\mathrm{OPT} = n$ (i.e. instance has a perfect matching):

$$\mathbb{P}\left[f(x) < \left(1 - \frac{1}{e} - \alpha\right)n\right] < e^{-2\alpha^2 n}.$$

Proof. Plug $\mathbb{E}[f(x)] \geq \left(1 - \frac{1}{e}\right)n$ and $c_i \equiv 1$ into McDiarmid. □

Proving Bounded Differences Property

Instead of changing ranks, consider removing a vertex:

Instead of changing ranks, consider removing a vertex:

Lemma

Assume all ranks are fixed and let j be some offline vertex.

Instead of changing ranks, consider removing a vertex:

Lemma

Assume all ranks are fixed and let j be some offline vertex.

Let M be the output of RANKING and let M_{-j} be the output of RANKING if j is removed from the instance.

Instead of changing ranks, consider removing a vertex:

Lemma

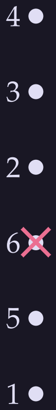
Assume all ranks are fixed and let j be some offline vertex.

Let M be the output of Ranking *and let M_{-j} be the output of* Ranking *if j is removed from the instance.*

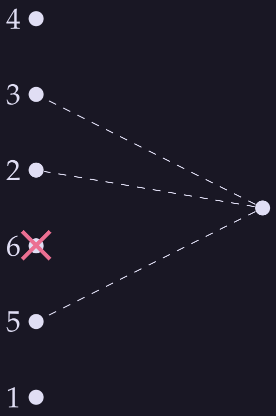
Then $|M_{-j}| \leq |M| \leq |M_{-j} + 1|$.

Proving Bounded Differences Property II

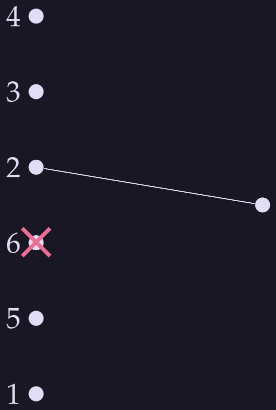
Proving Bounded Differences Property II



Proving Bounded Differences Property II

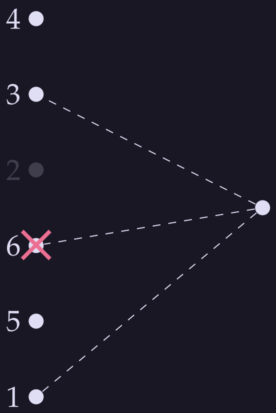


Proving Bounded Differences Property II

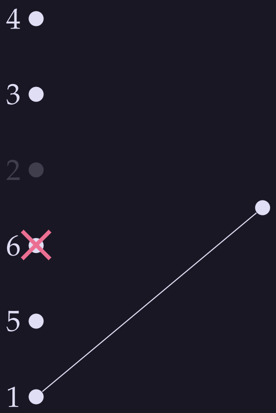


Proving Bounded Differences Property II

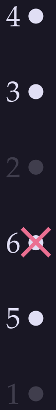
Proving Bounded Differences Property II



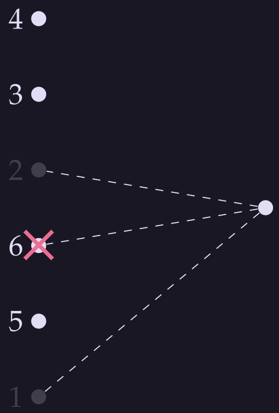
Proving Bounded Differences Property II



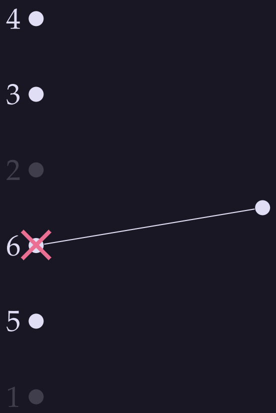
Proving Bounded Differences Property II



Proving Bounded Differences Property II

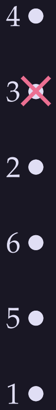


Proving Bounded Differences Property II

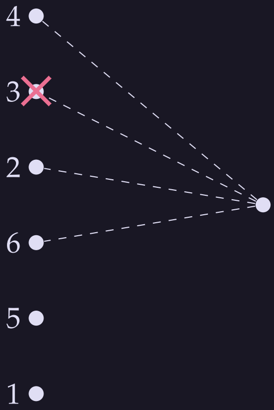


Proving Bounded Differences Property II

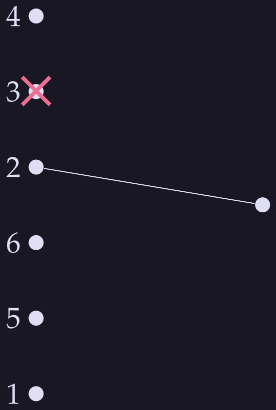
Proving Bounded Differences Property III



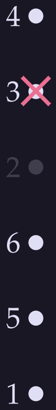
Proving Bounded Differences Property III



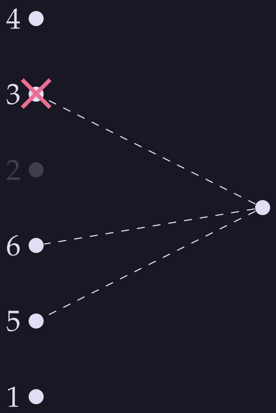
Proving Bounded Differences Property III



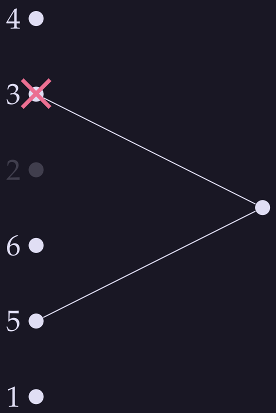
Proving Bounded Differences Property III



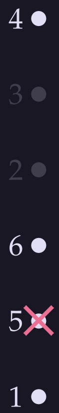
Proving Bounded Differences Property III



Proving Bounded Differences Property III



Proving Bounded Differences Property III



Proving Bounded Differences Property IV

Lemma

f satisfies the bounded differences property for $c_i \equiv 1$.

Proving Bounded Differences Property IV

Lemma

f satisfies the bounded differences property for $c_i \equiv 1$.

Proof. Consider $x, x' \in [0,1]^n$ that differ only on j. Then $|f(x) - f(x')| \leq 1$ since $x_{-j} = x'_{-j}$. $\square$

Generalizations

In Fully Online Matching:

Fully Online Matching

In Fully Online Matching:

- Can be non-bipartite

Fully Online Matching

In Fully Online Matching:

- Can be non-bipartite
- All vertices arrive and depart online

In Fully Online Matching:

- Can be non-bipartite
- All vertices arrive and depart online
- Vertices can only be matched if their [arrival, departure] overlap.

Fully Online Matching

In Fully Online Matching:

- Can be non-bipartite
- All vertices arrive and depart online
- Vertices can only be matched if their [arrival, departure] overlap.

Theorem

For the Fully Online Matching Problem, we have

$$\mathbb{E}[|M| < (\rho - \alpha)\mathrm{OPT}] < e^{-\alpha^2 \mathrm{OPT}}$$

where M is produced by Fully Online Ranking *and $\rho \approx 0.521$.*

Vertex-Weighted Online Bipartite Matching

In Vertex-Weighted Online Bipartite Matching:

In Vertex-Weighted Online Bipartite Matching:

- Each offline vertex j has a weight w_j.

Vertex-Weighted Online Bipartite Matching

In Vertex-Weighted Online Bipartite Matching:

- Each offline vertex j has a weight w_j.
- Goal is to maximize sum of weights of matched vertices.

In Vertex-Weighted Online Bipartite Matching:

- Each offline vertex j has a weight w_j.
- Goal is to maximize sum of weights of matched vertices.

Theorem

For each $\alpha > 0$, there exists an algorithm such that

$$\mathbb{P}\left[w(M) < \left(1 - \frac{1}{e} - \alpha\right)\mathrm{OPT}\right] < e^{-\frac{1}{50}\alpha^4 \frac{\mathrm{OPT}^2}{\|w\|_2^2}}.$$

Thank you!