Almost Tight Bounds for Online Hypergraph Matching

Thorben Tröbst (joint work with Rajan Udwani)

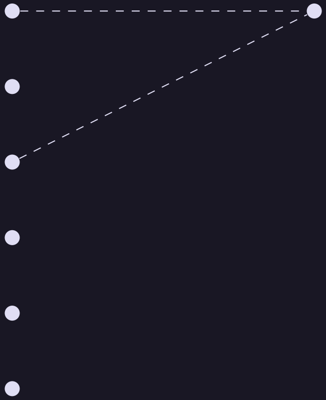
Theory Seminar, October 14, 2022

Department of Computer Science, University of California, Irvine

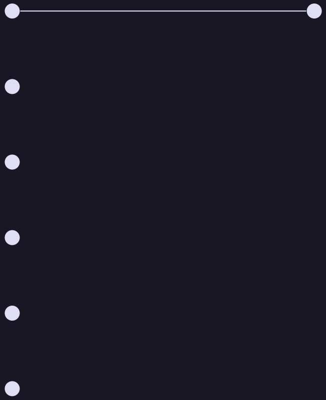
Online Bipartite Matching

Online Bipartite Matching

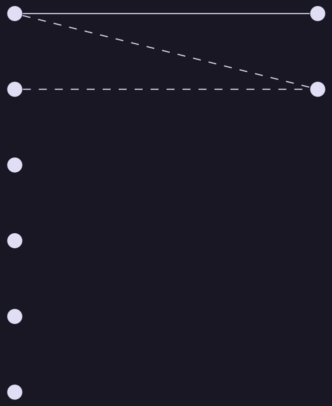
Online Bipartite Matching



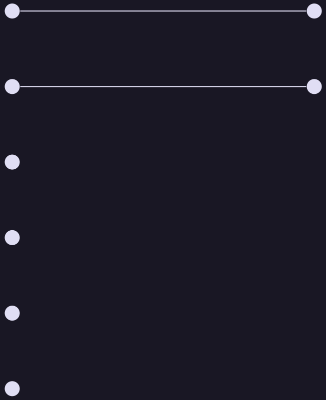
Online Bipartite Matching



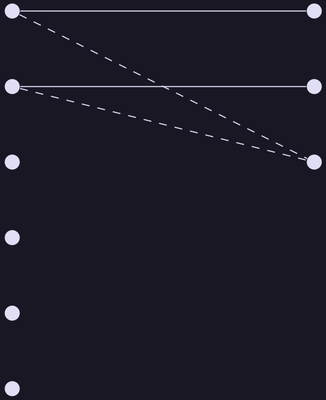
Online Bipartite Matching



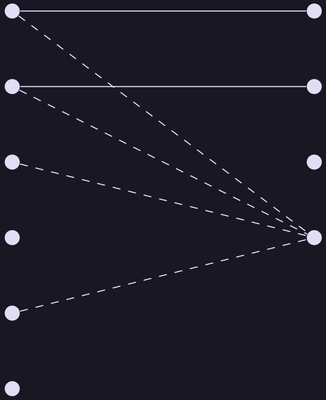
Online Bipartite Matching



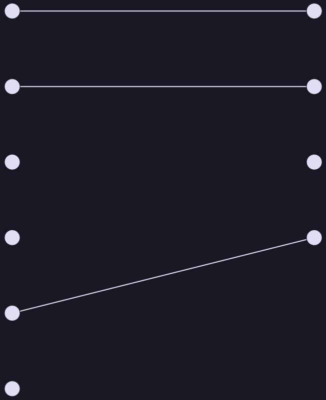
Online Bipartite Matching



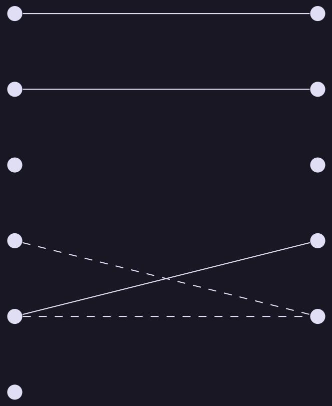
Online Bipartite Matching



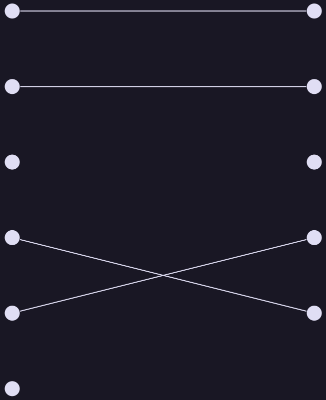
Online Bipartite Matching



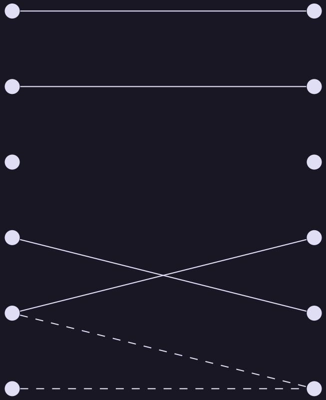
Online Bipartite Matching



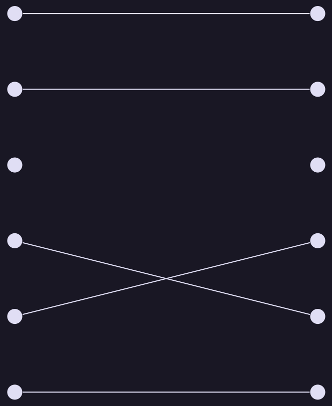
Online Bipartite Matching



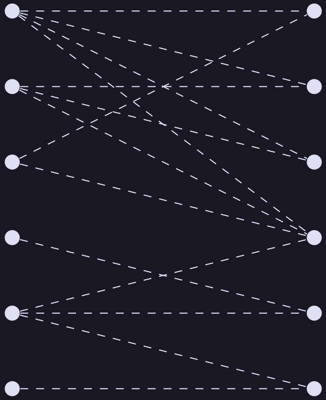
Online Bipartite Matching



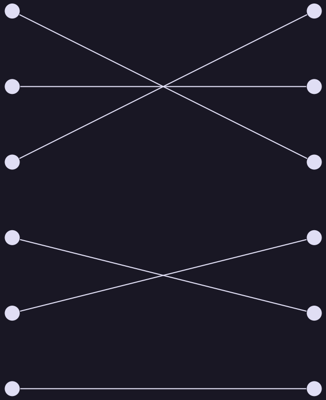
Online Bipartite Matching



Online Bipartite Matching



Online Bipartite Matching



Online Bipartite Matching II

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

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

Online vertices arrive one by one in adverserial order.

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 deterministic GREEDY algorithm (match whenever possible) is $1/2$-competitive (and this is best possible).

Classic results for Online Bipartite Matching:

- The deterministic Greedy algorithm (match whenever possible) is $1/2$-competitive (and this is best possible).
- The randomized Ranking algorithm is $(1-1/e)$-competitive in expectation and with high probability (and this is best possible).

Classic results for Online Bipartite Matching:

- The deterministic Greedy algorithm (match whenever possible) is $1/2$-competitive (and this is best possible).
- The randomized Ranking algorithm is $(1 - 1/e)$-competitive in expectation and with high probability (and this is best possible).
- The deterministic but fractional Waterfilling algorithm is $(1 - 1/e)$-competitive (and this is best possible).

Online Hypergraph Matching

k-Uniform Settings

Instead of graphs, consider k-uniform hypergraphs. We study the competitive ratio depending on k:

k-Uniform Settings

Instead of graphs, consider k-uniform hypergraphs. We study the competitive ratio depending on k:

- The immediate generalization would be k-partite with vertex arrivals.

k-Uniform Settings

Instead of graphs, consider k-uniform hypergraphs. We study the competitive ratio depending on k:

- The immediate generalization would be k-partite with vertex arrivals.
- Can also look at edge arrivals.

k-Uniform Settings

Instead of graphs, consider k-uniform hypergraphs. We study the competitive ratio depending on k:

- The immediate generalization would be k-partite with vertex arrivals.
- Can also look at edge arrivals.
- Small k and large k are different regimes.

k-Uniform Settings

Instead of graphs, consider k-uniform hypergraphs. We study the competitive ratio depending on k:

- The immediate generalization would be k-partite with vertex arrivals.
- Can also look at edge arrivals.
- Small k and large k are different regimes.
- k-partite does not seem to be too important for the large k regime.

Vertex Arrival / Edge Arrival

Vertex Arrival: when a vertex arrives, all of its hyperedges are revealed.

Edge Arrival: hyperedges arrive one by one.

Vertex Arrival / Edge Arrival

Vertex Arrival: when a vertex arrives, all of its hyperedges are revealed.

Edge Arrival: hyperedges arrive one by one.

Lemma

Edge arrival is at least as hard as vertex arrival.

Vertex Arrival: when a vertex arrives, all of its hyperedges are revealed.

Edge Arrival: hyperedges arrive one by one.

Lemma

Edge arrival is at least as hard as vertex arrival.

Lemma

Vertex arrival on a k-uniform instance is at least as hard as edge arrival on a $(k-1)$-uniform instance.

In this talk we will look at

- k-Uniform Online Hypergraph Matching with edge arrivals,

Overview

In this talk we will look at

- k-Uniform Online Hypergraph Matching with edge arrivals,
- for large k,

Overview

In this talk we will look at

- k-Uniform Online Hypergraph Matching with edge arrivals,
- for large k,
- with integral matchings and

Overview

In this talk we will look at

- k-Uniform Online Hypergraph Matching with edge arrivals,
- for large k,
- with integral matchings and
- fractional matchings.

Integral Setting

Theorem

The greedy algorithm is $1/k$*-competitive for* k*-Uniform Online Hypergraph Matching with Edge Arrivals.*

Theorem

The greedy algorithm is $1/k$-competitive for k-Uniform Online Hypergraph Matching with Edge Arrivals.

Proof.

Let $\mathrm{OPT} = m$.

Greedy Lower Bound

Theorem

The greedy algorithm is $1/k$-competitive for k-Uniform Online Hypergraph Matching with Edge Arrivals.

Proof.

Let $\mathrm{OPT} = m$. Every edge from the optimum solution must contain a vertex from Greedy.

Greedy Lower Bound

Theorem

The greedy algorithm is $1/k$-competitive for k-Uniform Online Hypergraph Matching with Edge Arrivals.

Proof.

Let $\mathrm{OPT} = m$. Every edge from the optimum solution must contain a vertex from Greedy. Thus Greedy covers at least m vertices which requires m/k edges.

Theorem

The greedy algorithm is $1/k$-competitive for k-Uniform Online Hypergraph Matching with Edge Arrivals.

Proof.

Let $\mathrm{OPT} = m$. Every edge from the optimum solution must contain a vertex from GREEDY. Thus GREEDY covers at least m vertices which requires m/k edges. Hence the competitive ratio is at least $1/k$. □

To get upper bounds on the competitive ratio, we need the following famous lemma:

Lemma

Let α be the best competitive ratio of any randomized algorithm. Let β be the competitive ratio of the best deterministic algorithm against some fixed distribution of instances. Then $\alpha \leq \beta$.

$4/k$ Upper Bound

Let us start with a warmup:

Theorem

If k is even, then there does not exist a $(\frac{4}{k} + \epsilon)$-competitive algorithm for the k-uniform online hypergraph matching problem for any $\epsilon > 0$.

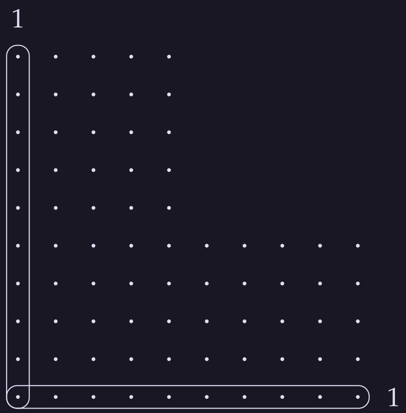
Let us start with a warmup:

Theorem

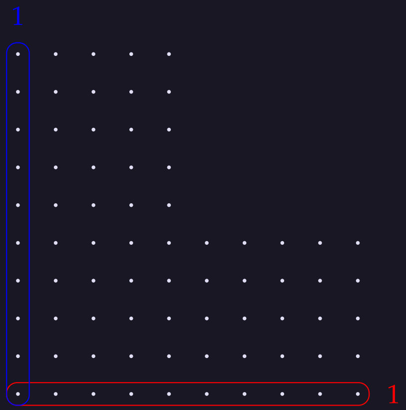
If k is even, then there does not exist a $(\frac{4}{k} + \epsilon)$-competitive algorithm for the k-uniform online hypergraph matching problem for any $\epsilon > 0$.

Idea: use Yao's principle and construct a distribution over instances with $\mathrm{OPT} = k/2$ but the best deterministic algorithm can only get a matching of size 2.

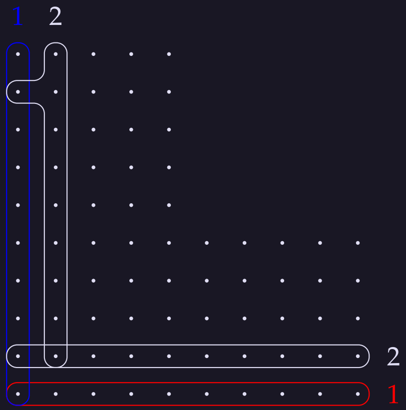
$4/k$ Upper Bound Construction



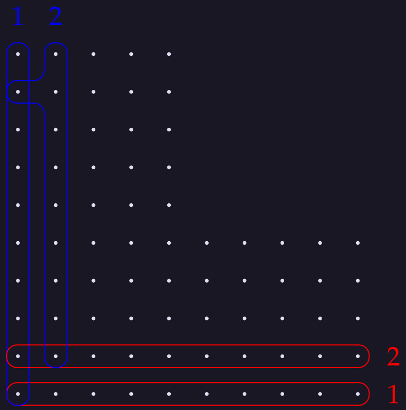
$4/k$ Upper Bound Construction

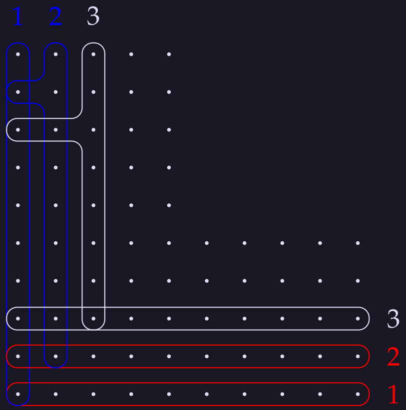


$4/k$ Upper Bound Construction

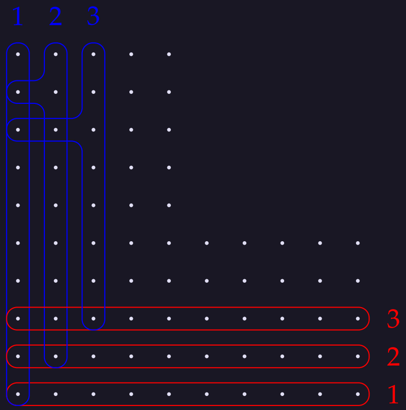


$4/k$ Upper Bound Construction

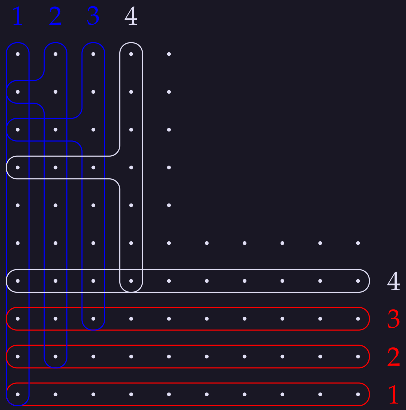



1
2
3
3
2
1

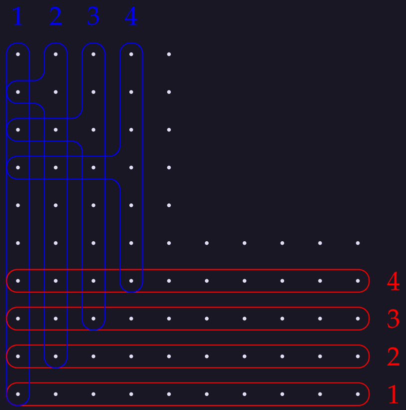
$4/k$ Upper Bound Construction



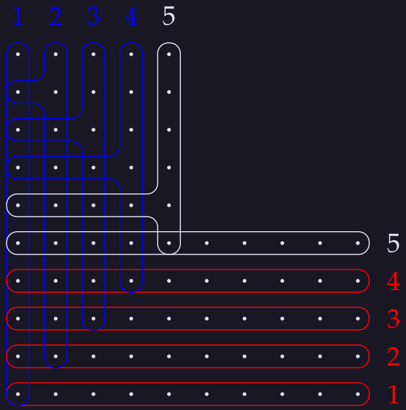
$4/k$ Upper Bound Construction



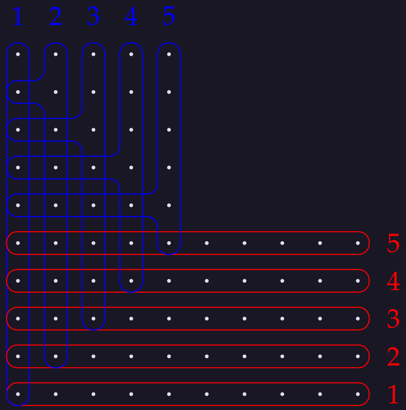
$4/k$ Upper Bound Construction



$4/k$ Upper Bound Construction



$4/k$ Upper Bound Construction



$4/k$ Upper Bound Proof

Proof.

Let α_i (β_i) be the probability that the red (blue) edge is matched in phase i.

Proof.

Let α_i (β_i) be the probability that the red (blue) edge is matched in phase i. Since the red and blue edges are determined independently and uniformly at random, we must have $\alpha_i = \beta_i$.

Proof.

Let α_i (β_i) be the probability that the red (blue) edge is matched in phase i. Since the red and blue edges are determined independently and uniformly at random, we must have $\alpha_i = \beta_i$. Moreover, since at most one blue edge can be picked, we know $\alpha_1 + \cdots + \alpha_{k/2} \leq 1$.

$4/k$ Upper Bound Proof

Proof.

Let α_i (β_i) be the probability that the red (blue) edge is matched in phase i. Since the red and blue edges are determined independently and uniformly at random, we must have $\alpha_i = \beta_i$. Moreover, since at most one blue edge can be picked, we know $\alpha_1 + \cdots + \alpha_{k/2} \leq 1$. Thus the expected size of the matching generated by the algorithm is at most

$$\alpha_1 + \cdots + \alpha_{k/2} + \beta_1 + \cdots + \beta_{k/2} \leq 2. \qquad \square$$

$2/\kappa$ Upper Bound

We can do better:

We can do better:

Theorem

If k is a power of two, then there does not exist a $(\frac{2}{k} + \epsilon)$-competitive algorithm for the online hypergraph matching problem for any $\epsilon > 0$.

2/k Upper Bound

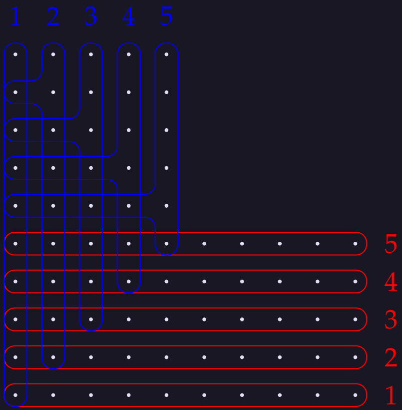
We can do better:

Theorem

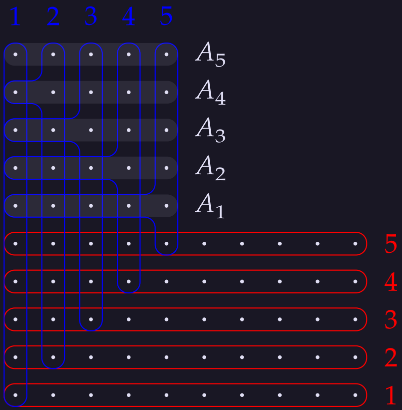
If k is a power of two, then there does not exist a $(\frac{2}{k} + \epsilon)$-competitive algorithm for the online hypergraph matching problem for any $\epsilon > 0$.

Idea: use the $4/k$ construction recursively.

The Gadget G_{10}

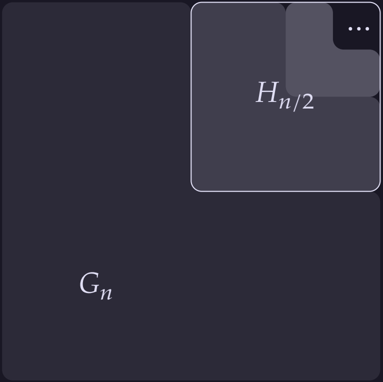


The Gadget G_{10}

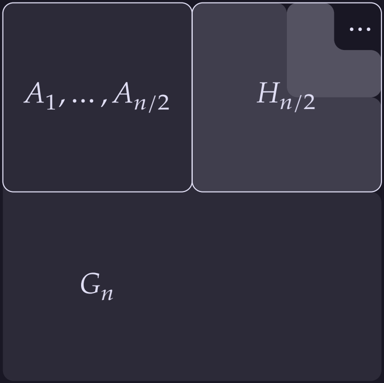



G_n

The Recursive Construction of H_n



The Recursive Construction of H_n



$2/k$ Upper Bound Proof

Proof.

The construction guarantees:

Proof.

The construction guarantees:

1. There are n red and n blue edges.

Proof.

The construction guarantees:

1. There are n red and n blue edges.
2. The edges appear in n phases, each of which consists of one red and one blue edge where the color is chosen uniformly and independently at random.

Proof.

The construction guarantees:

1. There are n red and n blue edges.
2. The edges appear in n phases, each of which consists of one red and one blue edge where the color is chosen uniformly and independently at random.
3. Every blue edge intersects all future edges.

Proof.

The construction guarantees:

1. There are n red and n blue edges.
2. The edges appear in n phases, each of which consists of one red and one blue edge where the color is chosen uniformly and independently at random.
3. Every blue edge intersects all future edges.
4. Every red edge is disjoint from all future edges.

Proof.

The construction guarantees:

1. There are n red and n blue edges.
2. The edges appear in n phases, each of which consists of one red and one blue edge where the color is chosen uniformly and independently at random.
3. Every blue edge intersects all future edges.
4. Every red edge is disjoint from all future edges.

Thus, the algorithm can still only get 2 whereas $\mathrm{OPT} = n$. □

Remarks and Future Work

Remarks and Future Work

- Note that the best known approximation algorithm (offline) achieves $3/k$.

Remarks and Future Work

- Note that the best known approximation algorithm (offline) achieves $3/k$.
- For $k = 2$ it is known, that $1/2$ is the best competitive ratio.

Remarks and Future Work

- Note that the best known approximation algorithm (offline) achieves $3/k$.
- For $k = 2$ it is known, that $1/2$ is the best competitive ratio.
- Open Problem: Is there any k where we can beat $1/k$ by any amount?

Remarks and Future Work

- Note that the best known approximation algorithm (offline) achieves $3/k$.
- For $k = 2$ it is known, that $1/2$ is the best competitive ratio.
- Open Problem: Is there any k where we can beat $1/k$ by any amount?
- Open Problem: Can we show that asymptotically, $1/k$ is the best possible?

Fractional Setting

Lower Bound: Water-Filling

Somewhat surprisingly, we can do much better for the fractional setting:

Somewhat surprisingly, we can do much better for the fractional setting:

Theorem

For the fractional k-uniform online hypergraph matching problem, there exists a $\frac{1-o(1)}{\ln k}$-competitive algorithm.

Water-Filling Algorithm

Algorithm 1: Hypergraph Water-Filling

For each $i \in V$, let $x_i = \sum_{e:i\in e} y_e$.
for *each edge* e *which arrives* **do**
  Match e continuously as long as $\sum_{i\in e}(k \ln(k))^{x_i - 1} \leq 1$.

Here y_e is the fill-level of edge e. The crucial part of the algorithm is the early stopping condition which depends on k.

Proof.

When Water-Filling is matching edge e in line 3, we interpret the quantity $\sum_{i \in e} (k \ln(k))^{x_i - 1}$ as the *price* of edge e.

Proof.

When Water-Filling is matching edge e in line 3, we interpret the quantity $\sum_{i\in e}(k\ln(k))^{x_i-1}$ as the *price* of edge e. Accordingly, if e is matched some by some infinitesimal amount $\mathrm{d}t$, then,

1. For every resource $i \in e$, we increase the *revenue* r_i by $(k\ln(k))^{x_i-1}\mathrm{d}t$.
2. We increase the *utility* u_e of e by $\left(1 - \sum_{i\in e}(k\ln(k))^{x_i-1}\right)\mathrm{d}t$.

Water-Filling Proof Sketch

Proof.

When Water-Filling is matching edge e in line 3, we interpret the quantity $\sum_{i \in e} (k \ln(k))^{x_i - 1}$ as the *price* of edge e. Accordingly, if e is matched some by some infinitesimal amount $\mathrm{d}t$, then,

1. For every resource $i \in e$, we increase the *revenue* r_i by $(k \ln(k))^{x_i - 1} \mathrm{d}t$.
2. We increase the *utility* u_e of e by $\left(1 - \sum_{i \in e} (k \ln(k))^{x_i - 1}\right) \mathrm{d}t$.

Note that this implies that the total sum of all revenues and utilities is equal to the total size of the matching.

Proof.

Now one can show for every $e \in E$:

$$u_e + \sum_{i \in e} r_i \geq \frac{1 - \frac{1}{\ln(k)}}{\ln(k) + \ln(\ln(k))} \geq \frac{1 - o(1)}{\ln(k)}.$$

Proof.

Now one can show for every $e \in E$:

$$u_e + \sum_{i \in e} r_i \geq \frac{1 - \frac{1}{\ln(k)}}{\ln(k) + \ln(\ln(k))} \geq \frac{1 - o(1)}{\ln(k)}.$$

The theorem then follows via weak duality since u_e and r_i can be scaled up to be a dual solution for the fractional hypergraph matching LP. □

In the fractional setting, we can give a matching upper bound (asymptotically):

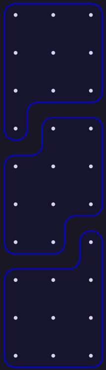
Upper Bound

In the fractional setting, we can give a matching upper bound (asymptotically):

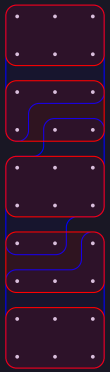
Theorem

There does not exist any online algorithm which is $\frac{1+\epsilon}{\ln(k)}$-competitive for the k-uniform online hypergraph matching problem as k tends to infinity.

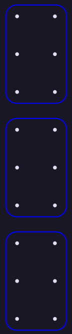
Upper Bound Construction ($k = 10, l = 3, \delta = 0.5$)



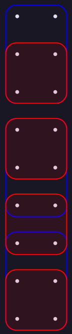
Upper Bound Construction ($k = 10$, $l = 3$, $\delta = 0.5$)



Upper Bound Construction ($k = 10, l = 3, \delta = 0.5$)

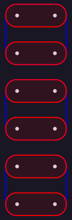


Upper Bound Construction ($k = 10$, $l = 3$, $\delta = 0.5$)

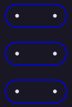


Upper Bound Construction ($k = 10, l = 3, \delta = 0.5$)

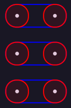
Upper Bound Construction ($k = 10, l = 3, \delta = 0.5$)



Upper Bound Construction ($k = 10$, $l = 3$, $\delta = 0.5$)



Upper Bound Construction ($k = 10$, $l = 3$, $\delta = 0.5$)



Upper Bound Construction ($k = 10, l = 3, \delta = 0.5$)

Upper Bound Proof Sketch

Proof.

Let α be the competitive ratio. There are $(1 - o(1)) \log_{1+\delta}(k)$ phases. In each, we cover l edges with at least $(1 + \delta)l - 1$ edges. Thus $\mathrm{ALG} \geq \alpha \mathrm{OPT} \geq \alpha(1 - o(1)) \log_{1+\delta}(k)(\delta l - 1)$.

Upper Bound Proof Sketch

Proof.

Let α be the competitive ratio. There are $(1 - o(1)) \log_{1+\delta}(k)$ phases. In each, we cover l edges with at least $(1+\delta)l - 1$ edges. Thus $\mathrm{ALG} \geq \alpha \mathrm{OPT} \geq \alpha(1 - o(1)) \log_{1+\delta}(k)(\delta l - 1)$.

Let $E^\star \subseteq E$ be the most used edges picked in each iteration and let y be the fractional matching constructed by the algorithm. Then because these edges are always the l most covered edges we know that

$$y(E^\star) \geq \min_{m \geq 1} \frac{l}{\left\lfloor \frac{lm}{\left\lfloor \frac{m}{1+\delta} \right\rfloor} \right\rfloor} \mathrm{ALG} \geq \frac{1}{1 + \delta - \frac{1}{l}} \mathrm{ALG}.$$

□

Upper Bound Proof Sketch II

Proof.

Lastly, we know that that all edges in $E^\star$ overlap in the final l vertices. Thus $y(E^\star) \leq l$ and by combining we get

$$\alpha \leq \frac{\left(1 + \delta - \frac{1}{l}\right) l}{(1 - o(1)) \log_{1+\delta}(k)(\delta l - 1)} = \frac{((1+\delta)l - 1)\ln(1+\delta)}{(1 - o(1))(\delta l - 1)} \cdot \frac{1}{\ln(k)}$$

□

- It is interesting that there is such a huge gap between integral and fractional here (unlike for other online matching problems).

Remarks and Future Work

- It is interesting that there is such a huge gap between integral and fractional here (unlike for other online matching problems).
- The result can be extended to edge weights under a free-disposal assumption.

Remarks and Future Work

- It is interesting that there is such a huge gap between integral and fractional here (unlike for other online matching problems).
- The result can be extended to edge weights under a free-disposal assumption.
- The only exact tight bounds are known for $k = 2$. For larger k we know better lower / upper bounds than shown here but they are not tight.

Thank your for listening!