Recent Advances in Online Matching: Edge-Weighted

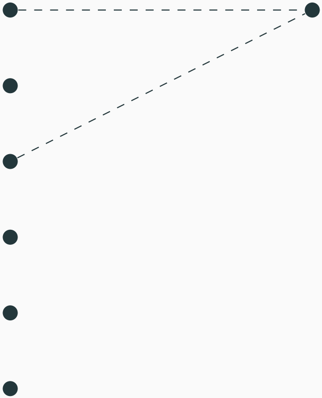
Thorben Tröbst

Theory Seminar, October 15, 2020

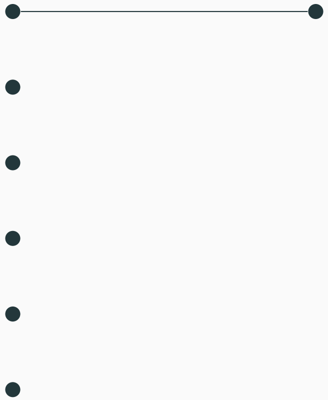
Department of Computer Science, University of California, Irvine

Online Bipartite Matching

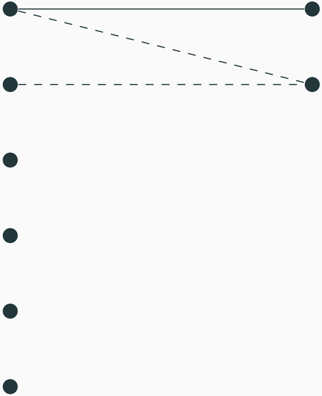
Online Bipartite Matching



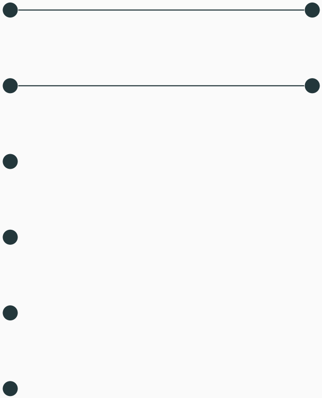
Online Bipartite Matching



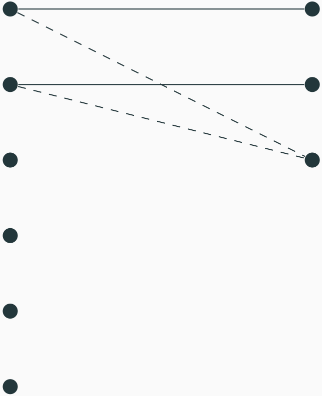
Online Bipartite Matching



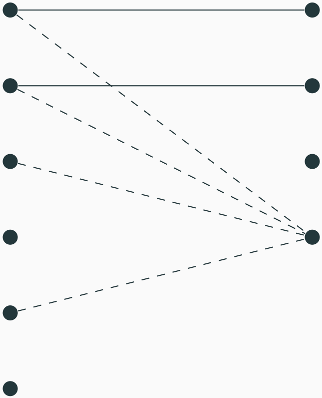
Online Bipartite Matching



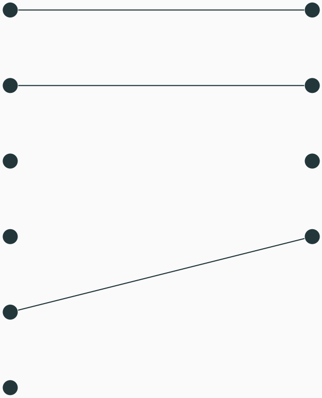
Online Bipartite Matching



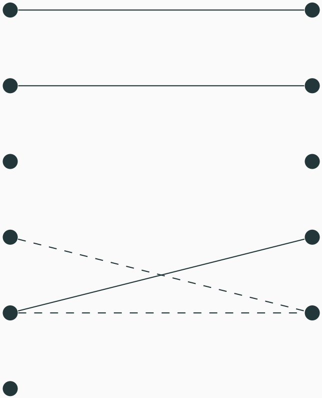
Online Bipartite Matching



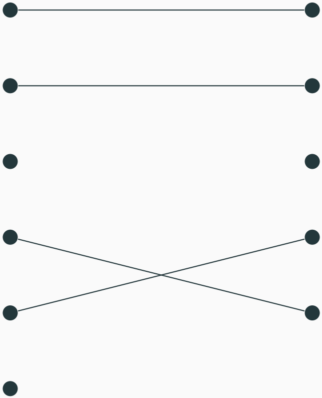
Online Bipartite Matching



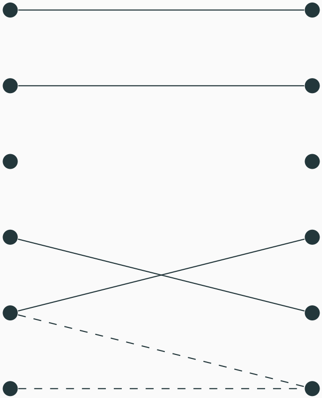
Online Bipartite Matching



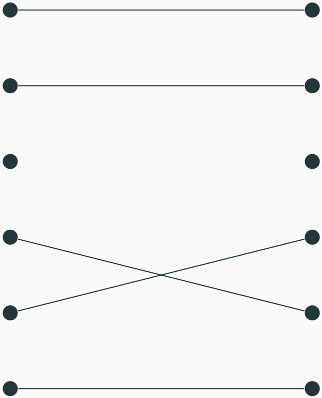
Online Bipartite Matching



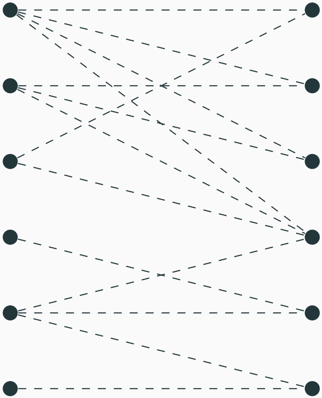
Online Bipartite Matching



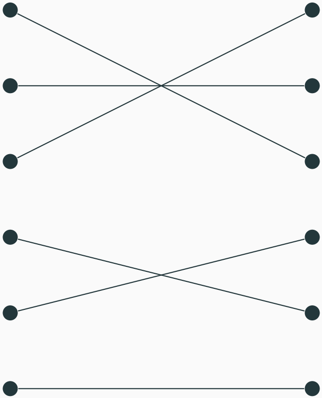
Online Bipartite Matching



Online Bipartite Matching



Online Bipartite Matching



Online Bipartite Matching II

$G = (S, B, E)$ 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 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.

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.

The goal is to maximize the competitive ratio, i.e.

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

There are two main algorithmic ideas for online matching problems:

Algorithms for Online Matching Problems

There are two main algorithmic ideas for online matching problems:

- Ranking

Algorithms for Online Matching Problems

There are two main algorithmic ideas for online matching problems:

- Ranking
  - Idea: randomly permute offline vertices and then match online vertices to the first available offline vertex.

Algorithms for Online Matching Problems

There are two main algorithmic ideas for online matching problems:

- Ranking
  - Idea: randomly permute offline vertices and then match online vertices to the first available offline vertex.
  - Provides integral solution in a randomized algorithm.

Algorithms for Online Matching Problems

There are two main algorithmic ideas for online matching problems:

- Ranking
  - Idea: randomly permute offline vertices and then match online vertices to the first available offline vertex.
  - Provides integral solution in a randomized algorithm.
- Water-Filling / Balancing

Algorithms for Online Matching Problems

There are two main algorithmic ideas for online matching problems:

- Ranking
  - Idea: randomly permute offline vertices and then match online vertices to the first available offline vertex.
  - Provides integral solution in a randomized algorithm.
- Water-Filling / Balancing
  - Idea: continuously allocate online vertices to the least-matched offline vertices.

Algorithms for Online Matching Problems

There are two main algorithmic ideas for online matching problems:

- RANKING
  - Idea: randomly permute offline vertices and then match online vertices to the first available offline vertex.
  - Provides integral solution in a randomized algorithm.
- WATER-FILLING / BALANCING
  - Idea: continuously allocate online vertices to the least-matched offline vertices.
  - Provides fractional solution in a deterministic algorithm.

In the Edge-Weighted Online Bipartite Matching Problem, every edge comes with a value v_{ji}.

Edge-Weighted Online Bipartite Matching

In the Edge-Weighted Online Bipartite Matching Problem, every edge comes with a value v_{ji}.

The goal is to maximize the value of matched edges.

Edge-Weighted Online Bipartite Matching

In the Edge-Weighted Online Bipartite Matching Problem, every edge comes with a value v_{ji}.

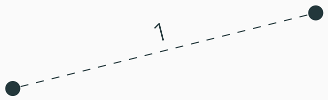
The goal is to maximize the value of matched edges.

Unfortunately, no algorithm has constant competitive ratio:

In the Edge-Weighted Online Bipartite Matching Problem, every edge comes with a value v_{ji}.

The goal is to maximize the value of matched edges.

Unfortunately, no algorithm has constant competitive ratio:

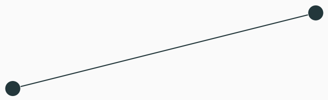


Edge-Weighted Online Bipartite Matching

In the Edge-Weighted Online Bipartite Matching Problem, every edge comes with a value v_{ji}.

The goal is to maximize the value of matched edges.

Unfortunately, no algorithm has constant competitive ratio:

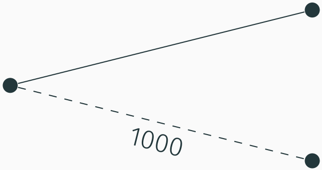


Edge-Weighted Online Bipartite Matching

In the Edge-Weighted Online Bipartite Matching Problem, every edge comes with a value v_{ji}.

The goal is to maximize the value of matched edges.

Unfortunately, no algorithm has constant competitive ratio:



Edge-Weighted Online Bipartite Matching

In the Edge-Weighted Online Bipartite Matching Problem, every edge comes with a value v_{ji}.

The goal is to maximize the value of matched edges.

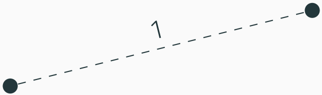
Unfortunately, no algorithm has constant competitive ratio:

Edge-Weighted Online Bipartite Matching

In the Edge-Weighted Online Bipartite Matching Problem, every edge comes with a value v_{ji}.

The goal is to maximize the value of matched edges.

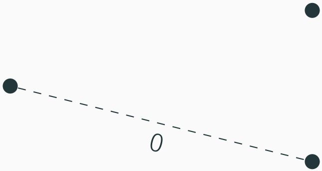
Unfortunately, no algorithm has constant competitive ratio:



In the Edge-Weighted Online Bipartite Matching Problem, every edge comes with a value v_{ji}.

The goal is to maximize the value of matched edges.

Unfortunately, no algorithm has constant competitive ratio:



Edge-Weighted Online Bipartite Matching

In the Edge-Weighted Online Bipartite Matching Problem, every edge comes with a value v_{ji}.

The goal is to maximize the value of matched edges.

Unfortunately, no algorithm has constant competitive ratio:

Free Disposal

In order to obtain a meaningful setting, we need an extra condition:

Definition

An online matching problem allows free disposal if the offline vertices are allowed drop previously matched online vertices.

Free Disposal

In order to obtain a meaningful setting, we need an extra condition:

Definition

An online matching problem allows free disposal if the offline vertices are allowed drop previously matched online vertices.

$\Rightarrow$ Greedy algorithm is now $\frac{1}{2}$-competitive!

Free Disposal

In order to obtain a meaningful setting, we need an extra condition:

Definition

An online matching problem allows free disposal if the offline vertices are allowed drop previously matched online vertices.

$\Rightarrow$ Greedy algorithm is now $\frac{1}{2}$-competitive!

$\Rightarrow$ This was best known until a recent breakthrough by Zadimoghaddam!

A New Algorithm for Unweighted

Rounding Balance

People tried for a long time to extend Ranking to no avail.

Rounding Balance

People tried for a long time to extend RANKING to no avail.

$\Rightarrow$ Lets find a new algorithm for the unweighted case and then try to extend that!

Rounding Balance

People tried for a long time to extend RANKING to no avail.

$\Rightarrow$ Lets find a new algorithm for the unweighted case and then try to extend that!

Consider the $\frac{1}{2}$-BALANCE algorithm:

Rounding Balance

People tried for a long time to extend RANKING to no avail.

$\Rightarrow$ Lets find a new algorithm for the unweighted case and then try to extend that!

Consider the $\frac{1}{2}$-BALANCE algorithm:

- Whenever an online vertex i arrives, let j_1, j_2 be the two neighbors which are currently least matched.

Rounding Balance

People tried for a long time to extend RANKING to no avail.

$\Rightarrow$ Lets find a new algorithm for the unweighted case and then try to extend that!

Consider the $\frac{1}{2}$-BALANCE algorithm:

- Whenever an online vertex i arrives, let j_1, j_2 be the two neighbors which are currently least matched.
- Fractionally match i to j_1 and j_2 with a value of $\frac{1}{2}$ each.

Rounding Balance

People tried for a long time to extend RANKING to no avail.

$\Rightarrow$ Lets find a new algorithm for the unweighted case and then try to extend that!

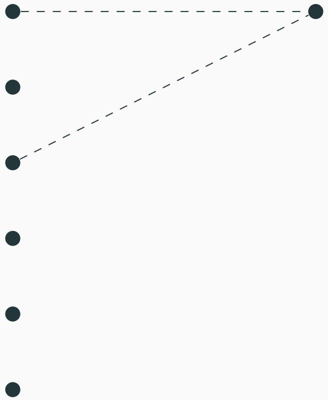
Consider the $\frac{1}{2}$-BALANCE algorithm:

- Whenever an online vertex i arrives, let j_1, j_2 be the two neighbors which are currently least matched.
- Fractionally match i to j_1 and j_2 with a value of $\frac{1}{2}$ each.

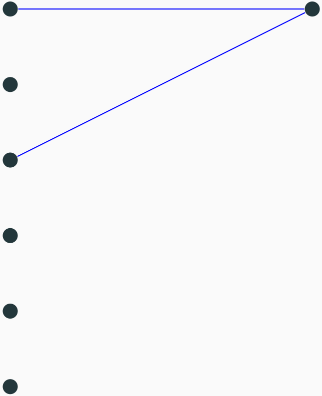
$\Rightarrow$ Yields $\frac{5}{9}$-competitive fractional matching!

$\frac{1}{2}$-Balance Example

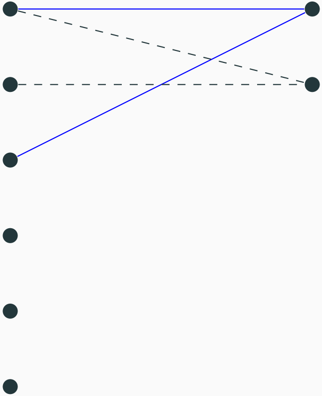
$\frac{1}{2}$-Balance Example



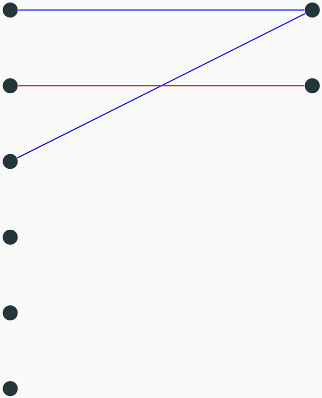
$\frac{1}{2}$-Balance Example



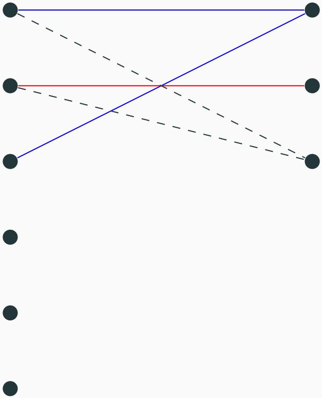
$\frac{1}{2}$-Balance Example



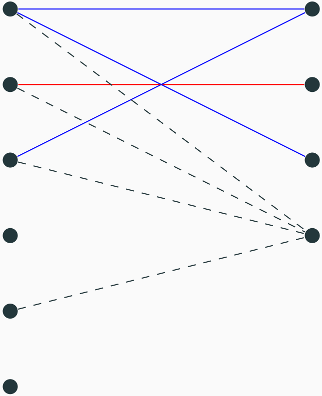
$\frac{1}{2}$-Balance Example



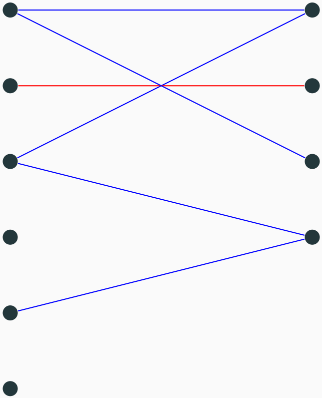
$\frac{1}{2}$-Balance Example



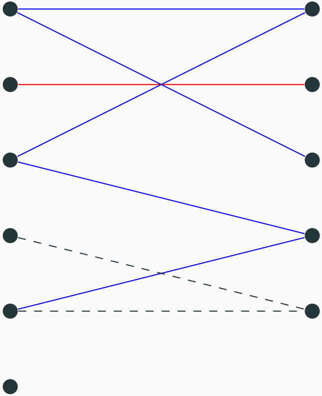
$\frac{1}{2}$-Balance Example



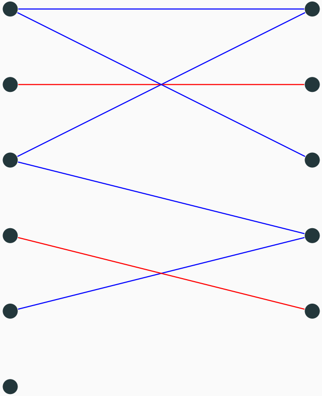
$\frac{1}{2}$-Balance Example



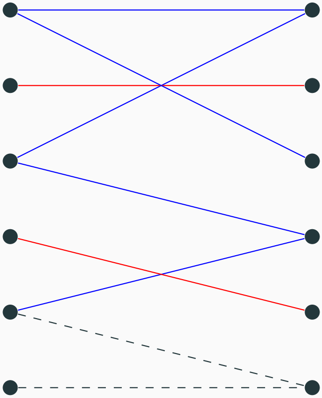
$\frac{1}{2}$-Balance Example



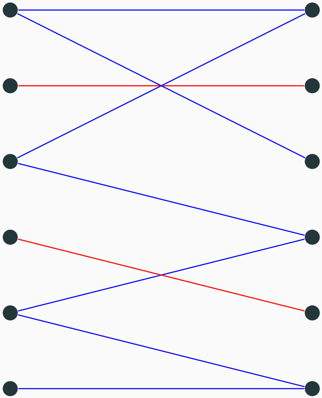
$\frac{1}{2}$-Balance Example



$\frac{1}{2}$-Balance Example



$\frac{1}{2}$-Balance Example



Rounding

We get a half-integral matching that is guaranteed to be $\frac{5}{9}$-competitive.

Rounding

We get a half-integral matching that is guaranteed to be $\frac{5}{9}$-competitive.

$\Rightarrow$ Can we round it somehow without too much loss?

Rounding

We get a half-integral matching that is guaranteed to be $\frac{5}{9}$-competitive.

$\Rightarrow$ Can we round it somehow without too much loss?

Note: Rounding after the fact is very easy with zero loss!

Rounding

We get a half-integral matching that is guaranteed to be $\frac{5}{9}$-competitive.

$\Rightarrow$ Can we round it somehow without too much loss?

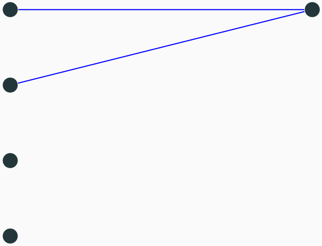
Note: Rounding after the fact is very easy with zero loss!

Core issue: How to round $\frac{1}{2}$-Balance online?

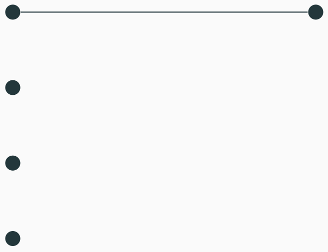
Rounding Problem

-
-
-
-

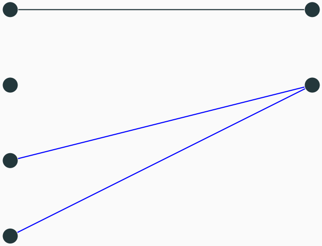
Rounding Problem



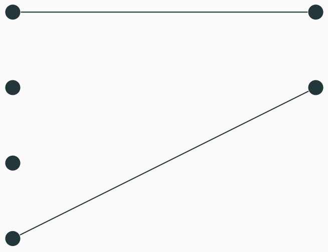
Rounding Problem



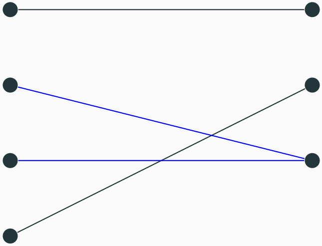
Rounding Problem



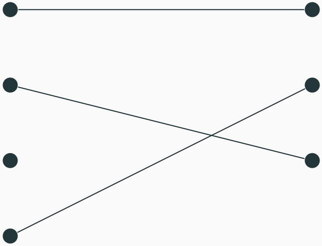
Rounding Problem



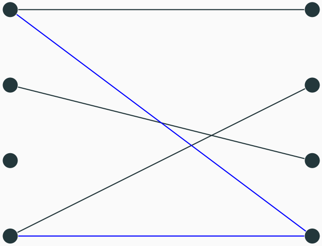
Rounding Problem



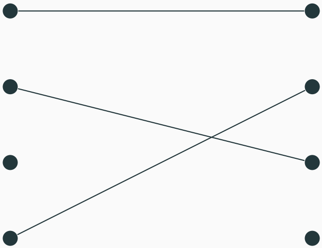
Rounding Problem



Rounding Problem



Rounding Problem



$\Rightarrow$ No rounding strategy can do better than 7/8-approx!

Rounding Strategy for Unweighted

Obvious rounding strategy: pick uniformly at random among the two choices.

Rounding Strategy for Unweighted

Obvious rounding strategy: pick uniformly at random among the two choices.

Problem: Does not beat $\frac{1}{2}$ due to collisions!

Rounding Strategy for Unweighted

Obvious rounding strategy: pick uniformly at random among the two choices.

Problem: Does not beat $\frac{1}{2}$ due to collisions!

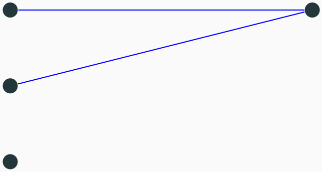
$\Rightarrow$ Pick uniformly if both neighbors are unpicked, otherwise try to avoid collisions.

Rounding Problem II

Just avoiding collisions is still not quite optimal:

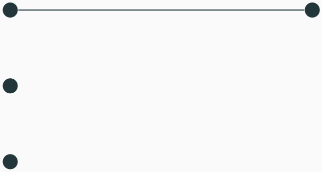
Rounding Problem II

Just avoiding collisions is still not quite optimal:



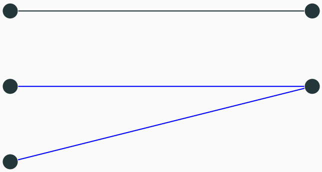
Rounding Problem II

Just avoiding collisions is still not quite optimal:



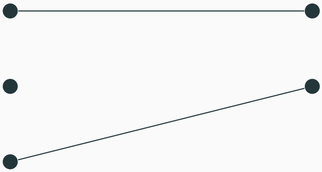
Rounding Problem II

Just avoiding collisions is still not quite optimal:



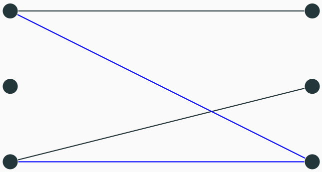
Rounding Problem II

Just avoiding collisions is still not quite optimal:



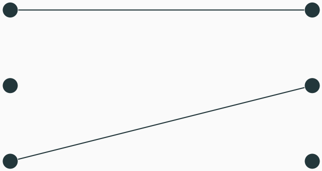
Rounding Problem II

Just avoiding collisions is still not quite optimal:



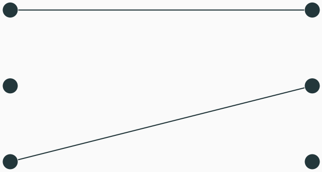
Rounding Problem II

Just avoiding collisions is still not quite optimal:



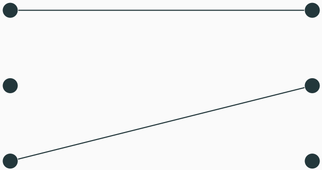
Rounding Problem II

Just avoiding collisions is still not quite optimal:



Rounding Problem II

Just avoiding collisions is still not quite optimal:



$\Rightarrow$ If a vertex was not chosen previously, choose it!

Unweighted Conclusion

Recall that we started with a $\frac{5}{9}$-competitive fractional matching...

Unweighted Conclusion

Recall that we started with a $\frac{5}{9}$-competitive fractional matching...then we rounded and (in the worst case) get $\frac{7}{8}$ of the fractional matching.

Unweighted Conclusion

Recall that we started with a $\frac{5}{9}$-competitive fractional matching...then we rounded and (in the worst case) get $\frac{7}{8}$ of the fractional matching.

But $\frac{5}{9} \times \frac{7}{8} = \frac{35}{72} < \frac{1}{2}$ which is not good enough!

Unweighted Conclusion

Recall that we started with a $\frac{5}{9}$-competitive fractional matching...then we rounded and (in the worst case) get $\frac{7}{8}$ of the fractional matching.

But $\frac{5}{9} \times \frac{7}{8} = \frac{35}{72} < \frac{1}{2}$ which is not good enough!

$\Rightarrow$ A fine-grained analysis can show that in order to get close to really only get a $\frac{5}{9}$-competitive fractional matching, we need many C_4 or C_6 in the support. But small cycles are easy to round!

Unweighted Conclusion

Recall that we started with a $\frac{5}{9}$-competitive fractional matching...then we rounded and (in the worst case) get $\frac{7}{8}$ of the fractional matching.

But $\frac{5}{9} \times \frac{7}{8} = \frac{35}{72} < \frac{1}{2}$ which is not good enough!

$\Rightarrow$ A fine-grained analysis can show that in order to get close to really only get a $\frac{5}{9}$-competitive fractional matching, we need many C_4 or C_6 in the support. But small cycles are easy to round!

So we can beat $\frac{1}{2}$ with rounded BALANCE!

The Crux of Weighted: Online Correlated Selection

Extending the Unweighted Algorithm

To extend the unweighted idea to weighted, two ingredients are needed:

Extending the Unweighted Algorithm

To extend the unweighted idea to weighted, two ingredients are needed:

- Weighted version of $\frac{1}{2}$-Balance

Extending the Unweighted Algorithm

To extend the unweighted idea to weighted, two ingredients are needed:

- Weighted version of $\frac{1}{2}$-Balance
- Weighted rounding

Extending the Unweighted Algorithm

To extend the unweighted idea to weighted, two ingredients are needed:

- Weighted version of $\frac{1}{2}$-Balance
- Weighted rounding

Good news: weighted $\frac{1}{2}$-Balance can still be done with factor $\frac{5}{9}$. See my talk last year.

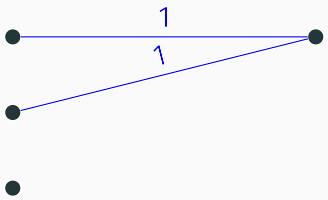
Weighted Rounding Problem

But how do we adapt the rounding strategy?

-
-
-

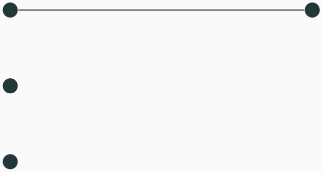
Weighted Rounding Problem

But how do we adapt the rounding strategy?



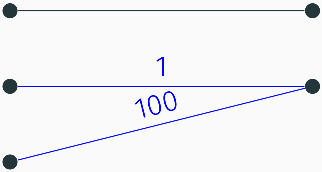
Weighted Rounding Problem

But how do we adapt the rounding strategy?



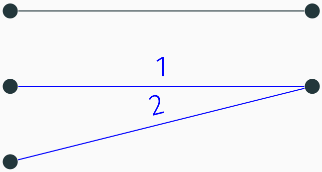
Weighted Rounding Problem

But how do we adapt the rounding strategy?

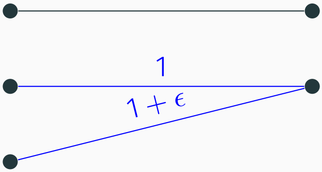


Weighted Rounding Problem

But how do we adapt the rounding strategy?



But how do we adapt the rounding strategy?



We can avoid weights entirely if we show a uniform bound:

We can avoid weights entirely if we show a uniform bound:

Problem

Give an online, randomized rounding strategy such that each edge is picked with probability $\frac{1}{2}$ and there is some amount of negative correlation among vertices.

We can avoid weights entirely if we show a uniform bound:

Problem

Give an online, randomized rounding strategy such that each edge is picked with probability $\frac{1}{2}$ and there is some amount of negative correlation among vertices.

This is the problem of Online Correlated Selection, the key ingredient of the weighted matching breakthrough!

The OCS Problem II

We are given a set A (known in advance) and pairs $\{a, b\} \subseteq A$ arrive online adversarially.

The OCS Problem II

We are given a set A (known in advance) and pairs $\{a, b\} \subseteq A$ arrive online adversarially.

Task: For each pair $P = \{a, b\}$ pick a winner $w \in P$ online and irrecovably such that

The OCS Problem II

We are given a set A (known in advance) and pairs $\{a, b\} \subseteq A$ arrive online adversarially.

Task: For each pair $P = \{a, b\}$ pick a winner $w \in P$ online and irrecovably such that

- $\mathbb{P}[w = a] = \mathbb{P}[w = b] = \frac{1}{2}$.

The OCS Problem II

We are given a set A (known in advance) and pairs $\{a, b\} \subseteq A$ arrive online adversarially.

Task: For each pair $P = \{a, b\}$ pick a winner $w \in P$ online and irrecovably such that

- $\mathbb{P}[w = a] = \mathbb{P}[w = b] = \frac{1}{2}$.
- For any fixed $c \in A$, the probability that c has not been matched after appearing in k pairs is $2^{-k}(1 - \gamma)^{k-1}$ for some $\gamma > 0$.

The Weighted Algorithm

The algorithm for weighted online matching is now clear:

The Weighted Algorithm

The algorithm for weighted online matching is now clear:

- As vertices arrive, use weighted $\frac{1}{2}$-Balance to determine (at most) two offline vertices j_1, j_2 to connect to.

The Weighted Algorithm

The algorithm for weighted online matching is now clear:

- As vertices arrive, use weighted $\frac{1}{2}$-Balance to determine (at most) two offline vertices j_1, j_2 to connect to.
- Use an OCS to decide the winner of $\{j_1, j_2\}$.

The algorithm for weighted online matching is now clear:

- As vertices arrive, use weighted $\frac{1}{2}$-Balance to determine (at most) two offline vertices j_1, j_2 to connect to.
- Use an OCS to decide the winner of $\{j_1, j_2\}$.
- If the negative correlation is large enough (e.g. $\gamma = \frac{1}{16}$), we beat $\frac{1}{2}$ just like the unweighted case!

Constructing an OCS

Let us construct a remarkably simple OCS for $\gamma = \frac{1}{16}$:

$$\{b, e\} \quad \{c, a\} \quad \{c, d\} \quad \{a, b\} \quad \{e, c\} \quad \{a, c\} \quad \{a, f\} \quad \{b, e\}$$

Constructing an OCS

Let us construct a remarkably simple OCS for $\gamma = \frac{1}{16}$:

$$\begin{tikzcd}[column sep=small]
\{b,e\} & \{c,a\} & \{c,d\} & \{a,b\} \arrow[ll, bend right] \arrow[lll, bend left] \arrow[rr, bend left] \arrow[rrrr, bend right] & \{e,c\} & \{a,c\} & \{a,f\} & \{b,e\}
\end{tikzcd}$$

Constructing an OCS

Let us construct a remarkably simple OCS for $\gamma = \frac{1}{16}$:

$$\{b,e\} \quad \{c,a\} \quad \{c,d\} \quad \{a,b\} \quad \{e,c\} \quad \{a,c\} \quad \{a,f\} \quad \{b,e\}$$

Constructing an OCS

Let us construct a remarkably simple OCS for $\gamma = \frac{1}{16}$:

$$\{b, e\} \quad \{c, a\} \quad \{c, d\} \quad \{a, b\} \quad \{e, c\} \quad \{a, c\} \quad \{a, f\} \quad \{b, e\}$$

Constructing an OCS

With this construction one can see:

Constructing an OCS

With this construction one can see:

- Winners are always picked with probability $\frac{1}{2}$.

Constructing an OCS

With this construction one can see:

- Winners are always picked with probability $\frac{1}{2}$.
- Probability that two pairs which contain consecutive occurences of some $c \in A$ are matched is (at least) $\frac{1}{16}$.

Constructing an OCS

With this construction one can see:

- Winners are always picked with probability $\frac{1}{2}$.
- Probability that two pairs which contain consecutive occurences of some $c \in A$ are matched is (at least) $\frac{1}{16}$.
- Whenever such a pair is matched, perfect negative correlation happens for c.

Constructing an OCS

With this construction one can see:

- Winners are always picked with probability $\frac{1}{2}$.
- Probability that two pairs which contain consecutive occurences of some $c \in A$ are matched is (at least) $\frac{1}{16}$.
- Whenever such a pair is matched, perfect negative correlation happens for c.
- Implies $2^{-k}(1 - \frac{1}{16})^{k-1}$ chance of not getting matched after k occurences!

Conclusion

Impact and Recent Works

- The OCS has already been improved by Charikar and Blanc by looking at $\frac{1}{k}$-Balance or even the continuous variant called Water-Filling.

Impact and Recent Works

- The OCS has already been improved by Charikar and Blanc by looking at $\frac{1}{k}$-Balance or even the continuous variant called Water-Filling.
- This leads to a 0.5368-competitive algorithm, much better than the 0.505 achieved viat he $\frac{1}{2}$-Balance approach!

- The OCS has already been improved by Charikar and Blanc by looking at $\frac{1}{k}$-Balance or even the continuous variant called Water-Filling.
- This leads to a 0.5368-competitive algorithm, much better than the 0.505 achieved viat he $\frac{1}{2}$-Balance approach!
- OCS has also been used to give an algorithm for the general AdWords problem that beats $\frac{1}{2}$.

Impact and Recent Works

- The OCS has already been improved by Charikar and Blanc by looking at $\frac{1}{k}$-BALANCE or even the continuous variant called WATER-FILLING.
- This leads to a 0.5368-competitive algorithm, much better than the 0.505 achieved viat he $\frac{1}{2}$-BALANCE approach!
- OCS has also been used to give an algorithm for the general AdWords problem that beats $\frac{1}{2}$.
- Several other online matching problems have edge weighted variants that could be tackled by this new tool.

Thank You!