Popular Matchings - Part II

Kavitha Telikepalli

(TIFR, Mumbai)

School on Computational Social Choice and Economics @ IIT Jodhpur

Maximum matchings

Applications where the size of the matching is more important than vertex preferences:

- matching medical students to hospitals for residency;
- matching doctors to hospitals in a pandemic;
- assigning sailors to billets (accommodation).

Maximum matchings

Applications where the size of the matching is more important than vertex preferences:

- matching medical students to hospitals for residency;
- matching doctors to hospitals in a pandemic;
- assigning sailors to billets (accommodation).

Here $\{$admissible solutions$\} = \{$maximum matchings$\}$.

Maximum matchings

Applications where the size of the matching is more important than vertex preferences:

- matching medical students to hospitals for residency;
- matching doctors to hospitals in a pandemic;
- assigning sailors to billets (accommodation).

Here $\{\text{admissible solutions}\} = \{\text{maximum matchings}\}$.

The goal is to find a *best* maximum matching as per vertex preferences.

Maximum matchings

Applications where the size of the matching is more important than vertex preferences:

- matching medical students to hospitals for residency;
- matching doctors to hospitals in a pandemic;
- assigning sailors to billets (accommodation).

Here $\{\text{admissible solutions}\} = \{\text{maximum matchings}\}$.

The goal is to find a *best* maximum matching as per vertex preferences.

- How about a maximum matching with the *minimum* number of blocking edges?
  - Finding such a matching is NP-hard (Biro, Manlove, and Mittal 2010).

Maximum matchings

Applications where the size of the matching is more important than vertex preferences:

- matching medical students to hospitals for residency;
- matching doctors to hospitals in a pandemic;
- assigning sailors to billets (accommodation).

Here {admissible solutions} = {maximum matchings}.

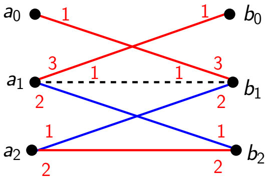
The goal is to find a *best* maximum matching as per vertex preferences.

- How about a maximum matching with the *minimum* number of blocking edges?
  - Finding such a matching is NP-hard (Biro, Manlove, and Mittal 2010).

- How about a maximum matching that is popular?

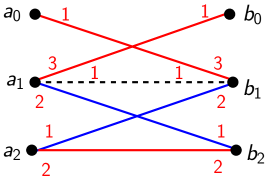
Maximum matchings and popularity

It can be the case that no maximum matching is popular.



Maximum matchings and popularity

It can be the case that no maximum matching is popular.

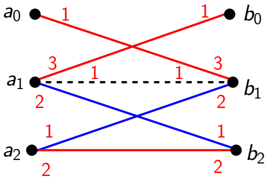


How about a maximum matching M that is popular *among* maximum matchings?

- So M is a maximum matching.
- Furthermore, $M \succeq M'$ for all maximum matchings M'.

Maximum matchings and popularity

It can be the case that no maximum matching is popular.



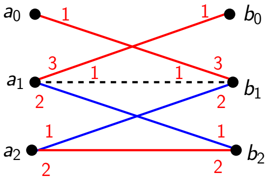
How about a maximum matching M that is popular *among* maximum matchings?

- So M is a maximum matching.
- Furthermore, $M \succeq M'$ for all maximum matchings M'.

Does such a "popular maximum matching" always exist in G?

Maximum matchings and popularity

It can be the case that no maximum matching is popular.



How about a maximum matching M that is popular *among* maximum matchings?

- So M is a maximum matching.
- Furthermore, $M \succeq M'$ for all maximum matchings M'.

Does such a "popular maximum matching" always exist in G?

- Furthermore, is it easy to find one?

Colourful graphs

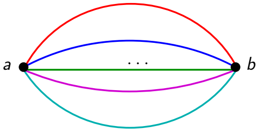
Recall our red/blue graph G' where every edge had a red copy and a blue copy.

Colourful graphs

Recall our red/blue graph G' where every edge had a red copy and a blue copy.

Suppose we use <u>n colours</u>, where $|A| = n$. Call the resulting graph G^*.

- Every edge ab in G has n parallel copies in G^*: $ab, ab, \ldots, ab, \ldots, ab, ab$.

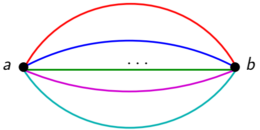


Colourful graphs

Recall our red/blue graph G' where every edge had a red copy and a blue copy.

Suppose we use $\underline{n \text{ colours}}$, where $|A| = n$. Call the resulting graph G^*.

- Every edge ab in G has n parallel copies in G^*: $ab, ab, \ldots, ab, \ldots, ab, ab$.



For any vertex a in A:

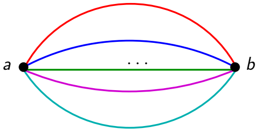
$$red \succ_a blue \succ_a \cdots \succ_a green \succ_a \cdots \succ_a magenta \succ_a cyan.$$

Colourful graphs

Recall our red/blue graph G' where every edge had a red copy and a blue copy.

Suppose we use <u>n colours</u>, where $|A| = n$. Call the resulting graph G^*.

- Every edge ab in G has n parallel copies in G^*: $ab, ab, \ldots, ab, \ldots, ab, ab$.



For any vertex a in A:

$$\textit{red} \succ_a \textit{blue} \succ_a \cdots \succ_a \textit{green} \succ_a \cdots \succ_a \textit{magenta} \succ_a \textit{cyan}.$$

For any vertex b in B:

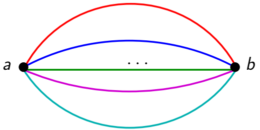
$$\textit{cyan} \succ_b \textit{magenta} \succ_b \cdots \succ_b \textit{green} \succ_b \cdots \succ_b \textit{blue} \succ_b \textit{red}.$$

Colourful graphs

Recall our red/blue graph G' where every edge had a red copy and a blue copy.

Suppose we use <u>n colours</u>, where $|A| = n$. Call the resulting graph G^*.

- Every edge ab in G has n parallel copies in G^*: $ab, ab, \ldots, ab, \ldots, ab, ab$.



For any vertex a in A:

$$red \succ_a blue \succ_a \cdots \succ_a green \succ_a \cdots \succ_a magenta \succ_a cyan.$$

For any vertex b in B:

$$cyan \succ_b magenta \succ_b \cdots \succ_b green \succ_b \cdots \succ_b blue \succ_b red.$$

Within any colour class, every vertex u maintains its original preference order $\succ_u$.

Popular maximum matchings

An extension of our earlier algorithm

- Construct the colourful graph $G^* = (A \cup B, E^*)$.
- Run Gale-Shapley algorithm in G^* to compute M^*.
- Return the corresponding matching M in G.

Popular maximum matchings

An extension of our earlier algorithm

- Construct the colourful graph $G^* = (A \cup B, E^*)$.
- Run Gale-Shapley algorithm in G^* to compute M^*.
- Return the corresponding matching M in G.

- CLAIM 1. M is a maximum matching in G.

Popular maximum matchings

An extension of our earlier algorithm

- Construct the colourful graph $G^* = (A \cup B, E^*)$.
- Run Gale-Shapley algorithm in G^* to compute M^*.
- Return the corresponding matching M in G.

- CLAIM 1. M is a maximum matching in G.
- CLAIM 2. $M \succeq M'$ for all maximum matchings M' in G.

Popular maximum matchings

An extension of our earlier algorithm

- Construct the colourful graph $G^* = (A \cup B, E^*)$.
- Run Gale-Shapley algorithm in G^* to compute M^*.
- Return the corresponding matching M in G.

- CLAIM 1. M is a maximum matching in G.
- CLAIM 2. $M \succeq M'$ for all maximum matchings M' in G.

Claims 1 and 2 $\Rightarrow$ M is a popular maximum matching.

Popular maximum matchings

An extension of our earlier algorithm

- Construct the colourful graph $G^* = (A \cup B, E^*)$.
- Run Gale-Shapley algorithm in G^* to compute M^*.
- Return the corresponding matching M in G.

- CLAIM 1. M is a maximum matching in G.
- CLAIM 2. $M \succeq M'$ for all maximum matchings M' in G.

Claims 1 and 2 $\Rightarrow$ M is a popular maximum matching.

- Moreover, such a matching can be computed easily.

The LP method

Recall the following edge weight function wt_M in G. For any edge ab:

$$\mathsf{wt}_M(ab) \;=\; \mathsf{vote}_a(b, M(a)) \;+\; \mathsf{vote}_b(a, M(b)).$$

$$\text{Here } \mathsf{vote}_v(u, u') \;=\; \begin{cases} 1 & \text{if } v \text{ prefers } u \text{ to } u' \\ -1 & \text{if } v \text{ prefers } u' \text{ to } u \\ 0 & \text{otherwise.} \end{cases}$$

So $\mathsf{wt}_M(e) \in \{0, \pm 2\}$ for any edge e.

The LP method

Recall the following edge weight function wt_M in G. For any edge ab:

$$\mathsf{wt}_M(ab) \;=\; \mathsf{vote}_a(b, M(a)) \;+\; \mathsf{vote}_b(a, M(b)).$$

$$\text{Here } \mathsf{vote}_v(u, u') \;=\; \begin{cases} 1 & \text{if } v \text{ prefers } u \text{ to } u' \\ -1 & \text{if } v \text{ prefers } u' \text{ to } u \\ 0 & \text{otherwise.} \end{cases}$$

So $\mathsf{wt}_M(e) \in \{0, \pm 2\}$ for any edge e.

- Let M be a maximum matching in G.
- OBSERVATION. $\mathsf{wt}_M(N) \leq 0$ for all maximum matchings N
  $\Rightarrow M$ is a popular maximum matching in G.

The LP method

LP for max-weight maximum matching in G:

$$\max \sum_e \mathrm{wt}_M(e) \cdot x_e$$

$$\sum_{e \in \delta(u)} x_e \quad \leq \quad 1 \quad \forall u \in A \cup B$$

$$\sum_{a \in A} \sum_{e \in \delta(a)} x_e \quad = \quad k \quad \text{and} \quad x_e \geq 0 \quad \forall e \in E.$$

Here k is the size of a maximum matching in G.

The LP method

LP for max-weight maximum matching in G:

$$\max \sum_{e} \mathrm{wt}_M(e) \cdot x_e$$

$$\sum_{e \in \delta(u)} x_e \quad \leq \quad 1 \;\; \forall u \in A \cup B$$

$$\sum_{a \in A} \sum_{e \in \delta(a)} x_e \quad = \quad k \quad \text{and} \quad x_e \geq 0 \;\; \forall e \in E.$$

Here k is the size of a maximum matching in G.

Optimal value of this LP is at most 0 $\Rightarrow \mathrm{wt}_M(N) \leq 0$ for all maximum matchings N

The LP method

LP for max-weight maximum matching in G:

$$\max \sum_{e} \text{wt}_M(e) \cdot x_e$$

$$\sum_{e \in \delta(u)} x_e \leq 1 \quad \forall u \in A \cup B$$

$$\sum_{a \in A} \sum_{e \in \delta(a)} x_e = k \quad \text{and} \quad x_e \geq 0 \ \forall e \in E.$$

Here k is the size of a maximum matching in G.

Optimal value of this LP is at most 0 $\Rightarrow \text{wt}_M(N) \leq 0$ for all maximum matchings N

$\Rightarrow M$ is a popular maximum matching in G.

The dual LP

$$\min \quad k \cdot z \;+\; \sum_{u} \alpha_u$$

$$\begin{aligned} \alpha_a + \alpha_b + z &\geq \mathrm{wt}_M(ab) \quad \forall\ ab \in E \\ \alpha_u &\geq 0 \qquad\qquad\ \ \forall\ u \in A \cup B. \end{aligned}$$

Our goal is to show that the optimal value of the dual LP is at most 0.

The dual LP

$$\min \quad k \cdot z \; + \; \sum_u \alpha_u$$

$$\begin{aligned} \alpha_a + \alpha_b + z \quad &\geq \quad \mathsf{wt}_M(ab) \quad \forall \; ab \in E \\ \alpha_u \quad &\geq \quad 0 \qquad\qquad\quad \forall \; u \in A \cup B. \end{aligned}$$

Our goal is to show that the optimal value of the dual LP is at most 0.

- Thus our goal is to show a dual feasible solution $(\vec{\alpha}, z)$ such that

$$k \cdot z + \sum_u \alpha_u = 0.$$

The dual LP

$$\min \quad k \cdot z \; + \; \sum_u \alpha_u$$

$$\begin{aligned} \alpha_a + \alpha_b + z \quad &\geq \quad \mathrm{wt}_M(ab) \quad \forall\; ab \in E \\ \alpha_u \quad &\geq \quad 0 \qquad\qquad\quad \forall\; u \in A \cup B. \end{aligned}$$

Our goal is to show that the optimal value of the dual LP is at most 0.

- Thus our goal is to show a dual feasible solution $(\vec{\alpha}, z)$ such that

$$k \cdot z + \sum_u \alpha_u = 0.$$

- Recall the colourful graph G^*:
  - let colour 0, colour 1, ..., colour $n-1$ denote the n colours.

A partition of the vertex set $A \cup B$

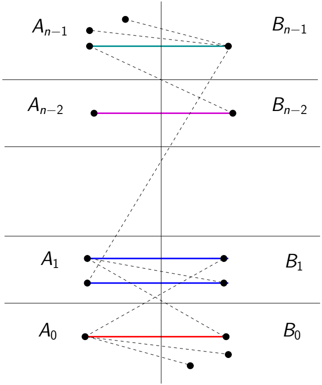
For $0 \leq i \leq n-1$, let $A_i = \{a \in A : a$ is matched along a colour i edge in $M^*\}$.

For $0 \leq i \leq n-1$, let $B_i = \{b \in B : b$ is matched along a colour i edge in $M^*\}$.

A partition of the vertex set $A \cup B$

For $0 \leq i \leq n-1$, let $A_i = \{a \in A : a \text{ is matched along a colour } i \text{ edge in } M^*\}$.

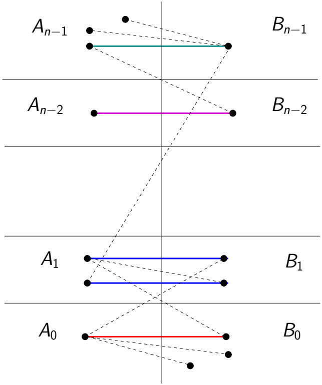
For $0 \leq i \leq n-1$, let $B_i = \{b \in B : b \text{ is matched along a colour } i \text{ edge in } M^*\}$.



A partition of the vertex set $A \cup B$

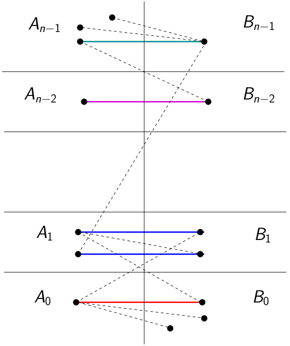
For $0 \leq i \leq n-1$, let $A_i = \{a \in A : a \text{ is matched along a colour } i \text{ edge in } M^*\}$.

For $0 \leq i \leq n-1$, let $B_i = \{b \in B : b \text{ is matched along a colour } i \text{ edge in } M^*\}$.



Unmatched vertices of A are in A_{n-1} and unmatched vertices of B are in B_0.

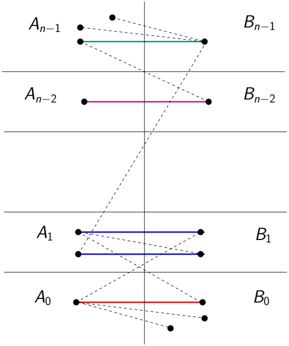
A partition of the vertex set $A \cup B$



The following properties hold due to the stability of M^* in G^*:

(1) For any i, the matching M restricted to $A_i \cup B_i$ is stable.

A partition of the vertex set $A \cup B$

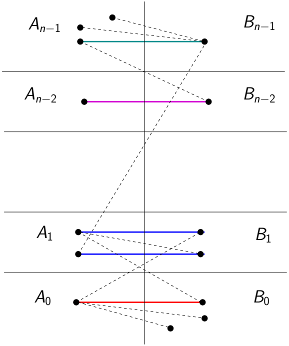


The following properties hold due to the stability of M^* in G^*:

(1) For any i, the matching M restricted to $A_i \cup B_i$ is stable.

(2) For any edge ab where $a \in A_{i+1}$ and $b \in B_i$: $\text{wt}_M(ab) = -2$.

A partition of the vertex set $A \cup B$



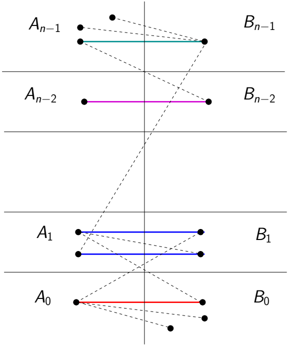
The following properties hold due to the stability of M^* in G^*:

(1) For any i, the matching M restricted to $A_i \cup B_i$ is stable.

(2) For any edge ab where $a \in A_{i+1}$ and $b \in B_i$: $\mathrm{wt}_M(ab) = -2$.

(3) G has no edge in $A_i \times B_j$ where $i \geq j + 2$.

A partition of the vertex set $A \cup B$



The following properties hold due to the stability of M^* in G^*:

(1) For any i, the matching M restricted to $A_i \cup B_i$ is stable.

(2) For any edge ab where $a \in A_{i+1}$ and $b \in B_i$: $\text{wt}_M(ab) = -2$.

(3) G has no edge in $A_i \times B_j$ where $i \geq j + 2$.

(4) There is no augmenting path with respect to M.

A dual certificate

Property (4) implies that M is a maximum matching in G.

A dual certificate

Property (4) implies that M is a maximum matching in G.

Let $z = -2(n-1)$. For $0 \leq i \leq n-1$:

- if $a \in A_i$ then set $\alpha_a = 2(n-1) - 2i$;
- if $b \in B_i$ then set $\alpha_b = 2i$.
  - So $\alpha_u = 0$ for any $u \in A_{n-1} \cup B_0$.

A dual certificate

Property (4) implies that M is a maximum matching in G.

Let $z = -2(n-1)$. For $0 \le i \le n-1$:

- if $a \in A_i$ then set $\alpha_a = 2(n-1) - 2i$;
- if $b \in B_i$ then set $\alpha_b = 2i$.
  - So $\alpha_u = 0$ for any $u \in A_{n-1} \cup B_0$.

Observe that $\alpha_a + \alpha_b + z \;=\; 2(n-1) - 2i + 2i - 2(n-1) \;=\; 0$ for each $ab \in M$.
($a \in A_i$ and $b \in B_i$ for some $i \in \{0, \ldots, n-1\}$)

A dual certificate

Property (4) implies that M is a maximum matching in G.

Let $z = -2(n-1)$. For $0 \le i \le n-1$:

- if $a \in A_i$ then set $\alpha_a = 2(n-1) - 2i$;
- if $b \in B_i$ then set $\alpha_b = 2i$.
  - So $\alpha_u = 0$ for any $u \in A_{n-1} \cup B_0$.

Observe that $\alpha_a + \alpha_b + z \;=\; 2(n-1) - 2i + 2i - 2(n-1) \;=\; 0$ for each $ab \in M$.
($a \in A_i$ and $b \in B_i$ for some $i \in \{0, \ldots, n-1\}$)

- Hence $k \cdot z + \sum_u \alpha_u \;=\; \sum_{ab \in M}(\alpha_a + \alpha_b + z) \;=\; 0$.

  (since $\alpha_u = 0$ for unmatched u)

A dual certificate

Property (4) implies that M is a maximum matching in G.

Let $z = -2(n-1)$. For $0 \le i \le n-1$:

- if $a \in A_i$ then set $\alpha_a = 2(n-1) - 2i$;
- if $b \in B_i$ then set $\alpha_b = 2i$.
  - So $\alpha_u = 0$ for any $u \in A_{n-1} \cup B_0$.

Observe that $\alpha_a + \alpha_b + z \;=\; 2(n-1) - 2i + 2i - 2(n-1) \;=\; 0$ for each $ab \in M$.
($a \in A_i$ and $b \in B_i$ for some $i \in \{0, \ldots, n-1\}$)

- Hence $k \cdot z + \sum_u \alpha_u \;=\; \sum_{ab \in M}(\alpha_a + \alpha_b + z) \;=\; 0$.

  (since $\alpha_u = 0$ for unmatched u)

So $\vec{\alpha}$ is dual-feasible $\Rightarrow$ the optimal value of the dual LP is at most 0.

Dual feasibility of $\vec{\alpha}$

Recall our assignment of α-values for $0 \leq i \leq n-1$:

- if $a \in A_i$ then set $\alpha_a = 2(n-1) - 2i$;
- if $b \in B_i$ then set $\alpha_b = 2i$;

Dual feasibility of $\vec{\alpha}$

Recall our assignment of α-values for $0 \leq i \leq n-1$:

- if $a \in A_i$ then set $\alpha_a = 2(n-1) - 2i$;
- if $b \in B_i$ then set $\alpha_b = 2i$;

Also $z = -2(n-1)$ and $\text{wt}_M(ab) \in \{0, \pm 2\}$.

Dual feasibility of $\vec{\alpha}$

Recall our assignment of α-values for $0 \le i \le n-1$:

- if $a \in A_i$ then set $\alpha_a = 2(n-1) - 2i$;
- if $b \in B_i$ then set $\alpha_b = 2i$;

Also $z = -2(n-1)$ and $\mathrm{wt}_M(ab) \in \{0, \pm 2\}$.

Recall properties (1)-(3):

(1) For any edge ab where $a \in A_i$ and $b \in B_i$: $\mathrm{wt}_M(ab) \le 0$.

(2) For any edge ab where $a \in A_{i+1}$ and $b \in B_i$: $\mathrm{wt}_M(ab) = -2$.

(3) G has no edge in $A_i \times B_j$ where $i \ge j + 2$.

Dual feasibility of $\vec{\alpha}$

Recall our assignment of α-values for $0 \leq i \leq n-1$:

- if $a \in A_i$ then set $\alpha_a = 2(n-1) - 2i$;
- if $b \in B_i$ then set $\alpha_b = 2i$;

Also $z = -2(n-1)$ and $\text{wt}_M(ab) \in \{0, \pm 2\}$.

Recall properties (1)-(3):

(1) For any edge ab where $a \in A_i$ and $b \in B_i$: $\text{wt}_M(ab) \leq 0$.

(2) For any edge ab where $a \in A_{i+1}$ and $b \in B_i$: $\text{wt}_M(ab) = -2$.

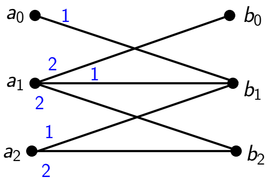
(3) G has no edge in $A_i \times B_j$ where $i \geq j + 2$.

These properties imply that $\alpha_a + \alpha_b + z \geq \text{wt}_M(ab)$ for all $ab \in E$.

- Thus $(\vec{\alpha}, z)$ is dual feasible.
- Hence M is a popular maximum matching.

The original model

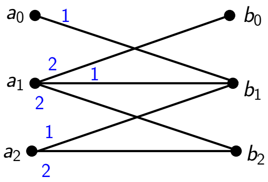
Popular matching algorithms were first studied in the model of *one-sided* preferences.



Vertices on the left are agents and those on the right are items.

The original model

Popular matching algorithms were first studied in the model of *one-sided* preferences.



Vertices on the left are agents and those on the right are items.

- Agents have preferences over their neighbours;
- items have no preferences.

This is also called a *house allocation* instance.

Popular matchings in this model

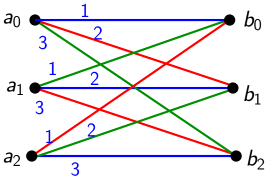
We say $M \succ N$, i.e., M is *more popular* than N, if

$$\underbrace{|\{a \in A : M \succ_a N\}|}_{\#\text{ of votes for } M} > \underbrace{|\{a \in A : N \succ_a M\}|}_{\#\text{ of votes for } N}.$$

Popular matchings in this model

We say $M \succ N$, i.e., M is *more popular* than N, if

$$\underbrace{|\{a \in A : M \succ_a N\}|}_{\#\text{ of votes for } M} > \underbrace{|\{a \in A : N \succ_a M\}|}_{\#\text{ of votes for } N}.$$



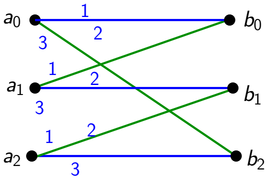
Let us hold elections between some pairs of matchings here.

- Say, between the green matching and the blue matching.

Popular matchings in this model

We say $M \succ N$, i.e., M is *more popular* than N, if

$$\underbrace{|\{a \in A : M \succ_a N\}|}_{\#\text{ of votes for } M} > \underbrace{|\{a \in A : N \succ_a M\}|}_{\#\text{ of votes for } N}.$$



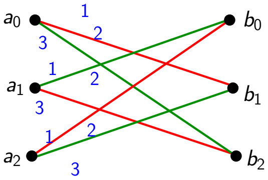
The green matching is more popular than the blue matching.

- In the green vs blue election: green gets 2 votes and blue gets only 1.

Popular matchings in this model

We say $M \succ N$, i.e., M is *more popular* than N, if

$$\underbrace{|\{a \in A : M \succ_a N\}|}_{\#\text{ of votes for } M} > \underbrace{|\{a \in A : N \succ_a M\}|}_{\#\text{ of votes for } N}.$$



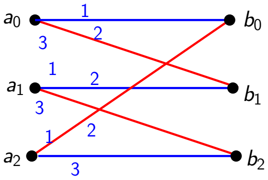
The red matching is more popular than the green matching.

- In the red vs green election: red gets 2 votes and green gets only 1.

Popular matchings in this model

We say $M \succ N$, i.e., M is *more popular* than N, if

$$\underbrace{|\{a \in A : M \succ_a N\}|}_{\#\text{ of votes for } M} > \underbrace{|\{a \in A : N \succ_a M\}|}_{\#\text{ of votes for } N}.$$



The blue matching is more popular than the red matching.

- In the blue vs red election: blue gets 2 votes and red gets only 1.

Popular matchings

So we have blue $\succ$ red $\succ$ green $\succ$ blue.

- For every matching here, there is a *more popular* matching.
- So this instance has no popular matching.

Popular matchings

So we have blue $\succ$ red $\succ$ green $\succ$ blue.

- For every matching here, there is a *more popular* matching.
- So this instance has no popular matching.

The popular matching problem

- *Given an instance* $G = (A \cup B, E)$, *does* G *admit a popular matching?*

Popular matchings

So we have blue $\succ$ red $\succ$ green $\succ$ blue.

- For every matching here, there is a *more popular* matching.
- So this instance has no popular matching.

The popular matching problem

- *Given an instance* $G = (A \cup B, E)$*, does* G *admit a popular matching?*

An efficient algorithm to solve the above problem is known [AIKM 2007].

Popular matchings

So we have blue $\succ$ red $\succ$ green $\succ$ blue.

- For every matching here, there is a *more popular* matching.
- So this instance has no popular matching.

The popular matching problem

- *Given an instance* $G = (A \cup B, E)$*, does* G *admit a popular matching?*

An efficient algorithm to solve the above problem is known [AIKM 2007].

- This algorithm works even when there are ties in preferences.

Popular matchings

So we have blue $\succ$ red $\succ$ green $\succ$ blue.

- For every matching here, there is a *more popular* matching.
- So this instance has no popular matching.

The popular matching problem

- *Given an instance $G = (A \cup B, E)$, does G admit a popular matching?*

An efficient algorithm to solve the above problem is known [AIKM 2007].

- This algorithm works even when there are ties in preferences.
- However it does not work for *partial order* preferences (which are more general).

Popular matchings

So we have blue $\succ$ red $\succ$ green $\succ$ blue.

- For every matching here, there is a *more popular* matching.
- So this instance has no popular matching.

The popular matching problem

- *Given an instance $G = (A \cup B, E)$, does G admit a popular matching?*

An efficient algorithm to solve the above problem is known [AIKM 2007].

- This algorithm works even when there are ties in preferences.
- However it does not work for *partial order* preferences (which are more general).
- Moreover, it does not extend to solve the popular maximum matching problem.

Popular assignments

OUR PROBLEM. Find a popular maximum matching in G, if one exists.

By adding appropriate dummy agents and artificial items to G:

- we can assume wlog that G has a perfect matching, i.e., an assignment.

Popular assignments

OUR PROBLEM. Find a popular maximum matching in G, if one exists.

By adding appropriate dummy agents and artificial items to G:

- we can assume wlog that G has a perfect matching, i.e., an assignment.

The popular assignment problem

- *Given an instance $G = (A \cup B, E)$, does G admit a popular assignment?*

Popular assignments

OUR PROBLEM. Find a popular maximum matching in G, if one exists.

By adding appropriate dummy agents and artificial items to G:

- we can assume wlog that G has a perfect matching, i.e., an assignment.

The popular assignment problem

- *Given an instance $G = (A \cup B, E)$, does G admit a popular assignment?*

- This is a generalization of the popular matching problem.
  - Moreover, agents have partial order preferences.

Popular assignments

Our Problem. Find a popular maximum matching in G, if one exists.

By adding appropriate dummy agents and artificial items to G:

- we can assume wlog that G has a perfect matching, i.e., an assignment.

The popular assignment problem

- *Given an instance $G = (A \cup B, E)$, does G admit a popular assignment?*

- This is a generalization of the popular matching problem.
  - Moreover, agents have partial order preferences.
- Our algorithm is combinatorial and it is based on the LP-method.

The LP-method

Given an assignment M, define edge weights in G as follows. For any edge ab:

$$\mathrm{wt}_M(ab) = \begin{cases} 1 & \text{if } a \text{ prefers } b \text{ to } M(a); \\ -1 & \text{if } a \text{ prefers } M(a) \text{ to } b; \\ 0 & \text{otherwise.} \end{cases}$$

The LP-method

Given an assignment M, define edge weights in G as follows. For any edge ab:

$$\mathsf{wt}_M(ab) = \begin{cases} 1 & \text{if } a \text{ prefers } b \text{ to } M(a); \\ -1 & \text{if } a \text{ prefers } M(a) \text{ to } b; \\ 0 & \text{otherwise.} \end{cases}$$

OBSERVATION. For any assignment N:

$$\mathsf{wt}_M(N) = \sum_{e \in N} \mathsf{wt}_M(e) = \#\text{ of votes for } N - \#\text{ of votes for } M.$$

The LP-method

Given an assignment M, define edge weights in G as follows. For any edge ab:

$$\mathrm{wt}_M(ab) = \begin{cases} 1 & \text{if } a \text{ prefers } b \text{ to } M(a); \\ -1 & \text{if } a \text{ prefers } M(a) \text{ to } b; \\ 0 & \text{otherwise.} \end{cases}$$

OBSERVATION. For any assignment N:

$$\mathrm{wt}_M(N) = \sum_{e \in N} \mathrm{wt}_M(e) = \#\text{ of votes for } N - \#\text{ of votes for } M.$$

- M is a popular assignment $\iff$ $\mathrm{wt}_M(N) \le 0$ for all assignments N in G.

The LP-method

LP for max-weight assignment:

$$\max \sum_{e \in E} \mathsf{wt}_M(e) \cdot x_e$$

$$\sum_{e \in \delta(v)} x_e \quad = \quad 1 \;\; \forall v \in A \cup B \quad \text{and} \quad x_e \;\geq\; 0 \;\; \forall e \in E.$$

M is popular $\iff$ the optimal value of this LP is at most 0.

The LP-method

LP for max-weight assignment:

$$\max \sum_{e \in E} \mathsf{wt}_M(e) \cdot x_e$$

$$\sum_{e \in \delta(v)} x_e \quad = \quad 1 \;\; \forall v \in A \cup B \quad \text{and} \quad x_e \;\geq\; 0 \;\; \forall e \in E.$$

M is popular $\iff$ the optimal value of this LP is at most 0.

The dual LP:

$$\min \sum_{u} \alpha_u$$

$$\alpha_a + \alpha_b \quad \geq \quad \mathsf{wt}_M(ab) \quad \forall ab \in E$$

M is popular $\iff$ the optimal value of the dual LP is at most 0.

The LP-method

The dual LP:

$$\min \sum_{u} \alpha_u$$

$$\alpha_a + \alpha_b \quad \geq \quad \mathsf{wt}_M(ab) \quad \forall ab \in E$$

M is popular $\iff$ the optimal value of the dual LP is at most 0.

The LP-method

The dual LP:

$$\min \sum_u \alpha_u$$

$$\alpha_a + \alpha_b \quad \geq \quad \text{wt}_M(ab) \quad \forall ab \in E$$

M is popular $\iff$ the optimal value of the dual LP is at most 0.

CLAIM. M is popular $\iff$ $\exists$ dual feasible $\vec{\alpha}$ such that $\sum_u \alpha_u = 0$ and

- $\alpha_a \in \{0, 1, 2, \ldots, n-1\}$ for all $a \in A$;
- $\alpha_b \in \{0, -1, -2, \ldots, -(n-1)\}$ for all $b \in B$.

Such an $\vec{\alpha}$ will certify that M is a popular assignment in G.

A colour function

Let $c : B \to \{0, 1, 2, \ldots, n-1\}$.

- We can define a subgraph $G_c = (A \cup B, E_c)$ of G as follows.
- For each $a \in A$: let $c^*(a) = \underbrace{\max\{c(b) : b \in \text{Nbr}(a)\}}_{\text{best colour among its neighbours}}$.

A colour function

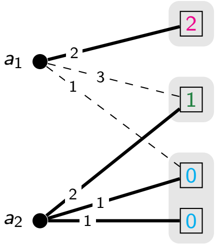
Let $c : B \to \{0, 1, 2, \ldots, n-1\}$.

- We can define a subgraph $G_c = (A \cup B, E_c)$ of G as follows.
- For each $a \in A$: let $c^*(a) = \underbrace{\max\{c(b) : b \in \text{Nbr}(a)\}}_{\text{best colour among its neighbours}}$.

The edge set E_c of G_c has those edges ab where:

- $c(b) = c^*(a)$ and b is a most preferred neighbour for a in colour $c^*(a)$.
- $c(b) = c^*(a) - 1$ and b is a most preferred neighbour for a in the "second best" colour $c^*(a) - 1$ and a prefers b to all neighbours in colour $c^*(a)$.

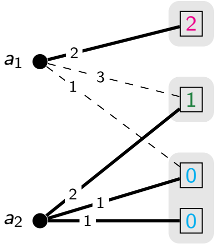
The subgraph G_c



Here $c^*(a_1) = 2$ and $c^*(a_2) = 1$.

- a_1 does not prefer its neighbour in colour 1 to its neighbour in colour 2;
- a_2 prefers its neighbours in colour 0 to its neighbour in colour 1;
- the bold edges are in E_c and the dashed edges are not.

The subgraph G_c

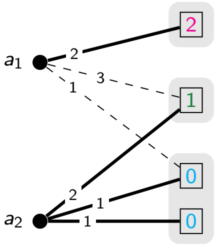


Here $c^*(a_1) = 2$ and $c^*(a_2) = 1$.

- a_1 does not prefer its neighbour in colour 1 to its neighbour in colour 2;
- a_2 prefers its neighbours in colour 0 to its neighbour in colour 1;
- the bold edges are in E_c and the dashed edges are not.

G_c has a perfect matching $M \Rightarrow M$ is a popular assignment in G.

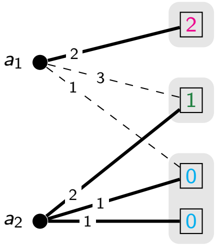
$\exists$ such a colouring $c \iff \exists$ a popular assignment in G



G has a popular assignment if and only if

- $\exists c : B \to \{0, 1, 2, \ldots, n-1\}$ s.t. G_c admits a perfect matching (call it M);
  - $\alpha_a = c(M(a))$ for $a \in A$ and $\alpha_b = -c(b)$ for $b \in B$ is M's dual certificate.

$\exists$ such a colouring $c \iff \exists$ a popular assignment in G



G has a popular assignment if and only if

- $\exists c : B \to \{0, 1, 2, \ldots, n-1\}$ s.t. G_c admits a perfect matching (call it M);
  - $\alpha_a = c(M(a))$ for $a \in A$ and $\alpha_b = -c(b)$ for $b \in B$ is M's dual certificate.

OUR PROBLEM: Find such a colouring $c : B \to \{0, 1, 2, \ldots, n-1\}$, if there exists one.

Finding a popular assignment/the right c [KKMSS 2022]

1. Initialize $c(b) = 0$ for every $b \in B$.
2. Compute a maximum matching M in the subgraph G_c.
3. If M is perfect then return M.

Finding a popular assignment/the right c [KKMSS 2022]

1. Initialize $c(b) = 0$ for every $b \in B$.
2. Compute a maximum matching M in the subgraph G_c.
3. If M is perfect then return M.
4. For every unmatched item b do: $c(b) = c(b) + 1$.
5. If $c(b) = n$ for any $b \in B$ then return "no"; else go back to Step 2.

Finding a popular assignment/the right c [KKMSS 2022]

1. Initialize $c(b) = 0$ for every $b \in B$.
2. Compute a maximum matching M in the subgraph G_c.
3. If M is perfect then return M.
4. For every unmatched item b do: $c(b) = c(b) + 1$.
5. If $c(b) = n$ for any $b \in B$ then return "no"; else go back to Step 2.

Eventually, either a perfect matching M in G_c is returned
or $c(b) = n$ for some $b \in B$.

Finding a popular assignment/the right c [KKMSS 2022]

1. Initialize $c(b) = 0$ for every $b \in B$.
2. Compute a maximum matching M in the subgraph G_c.
3. If M is perfect then return M.
4. For every unmatched item b do: $c(b) = c(b) + 1$.
5. If $c(b) = n$ for any $b \in B$ then return "no"; else go back to Step 2.

Eventually, either a perfect matching M in G_c is returned
or $c(b) = n$ for some $b \in B$.

- In the former case M is a popular assignment with $\vec{\alpha}$ as a dual certificate:

$$\alpha_a = c(M(a)) \text{ for } a \in A \text{ and } \alpha_b = -c(b) \text{ for } b \in B.$$

- In the latter case we show that G has *no* popular assignment.

Finding a popular assignment/the right c [KKMSS 2022]

1. Initialize $c(b) = 0$ for every $b \in B$.
2. Compute a maximum matching M in the subgraph G_c.
3. If M is perfect then return M.
4. For every unmatched item b do: $c(b) = c(b) + 1$.
5. If $c(b) = n$ for any $b \in B$ then return "no"; else go back to Step 2.

Eventually, either a perfect matching M in G_c is returned
or $c(b) = n$ for some $b \in B$.

- In the former case M is a popular assignment with $\vec{\alpha}$ as a dual certificate:

$$\alpha_a = c(M(a)) \text{ for } a \in A \text{ and } \alpha_b = -c(b) \text{ for } b \in B.$$

- In the latter case we show that G has *no* popular assignment.

For the popular matching problem, we will replace "$c(b) = n$" with "$c(b) = 2$".

When the algorithm says G has no popular assignment

This means $c(b) = n$ for some $b \in B$.

When the algorithm says G has no popular assignment

This means $c(b) = n$ for some $b \in B$.

Suppose G has a popular assignment M'; let $\vec{\alpha}'$ be a dual certificate for M'.

- Define $c'(b) = -\alpha'_b$ for all $b \in B$.
- CLAIM. $c'(b) \geq c(b)$ for all b.

When the algorithm says G has no popular assignment

This means $c(b) = n$ for some $b \in B$.

Suppose G has a popular assignment M'; let $\vec{\alpha}'$ be a dual certificate for M'.

- Define $c'(b) = -\alpha'_b$ for all $b \in B$.
- CLAIM. $c'(b) \geq c(b)$ for all b.
  - So $c(b) = n \Rightarrow c'(b) \geq n$, a contradiction.

  recall that $\alpha'_b \in \{0, -1, \ldots, -(n-1)\}$, so $0 \leq c'(b) \leq n - 1$.

When the algorithm says G has no popular assignment

This means $c(b) = n$ for some $b \in B$.

Suppose G has a popular assignment M'; let $\vec{\alpha}'$ be a dual certificate for M'.

- Define $c'(b) = -\alpha'_b$ for all $b \in B$.
- CLAIM. $c'(b) \geq c(b)$ for all b.
  - So $c(b) = n \Rightarrow c'(b) \geq n$, a contradiction.

    recall that $\alpha'_b \in \{0, -1, \ldots, -(n-1)\}$, so $0 \leq c'(b) \leq n-1$.

Let c_0 be the current c-function in our algorithm.

- Assume $c'(b) \geq c_0(b)$ for all b since $c(b) = 0$ is the initial c-function.

When the algorithm says G has no popular assignment

This means $c(b) = n$ for some $b \in B$.

Suppose G has a popular assignment M'; let $\vec{\alpha}'$ be a dual certificate for M'.

- Define $c'(b) = -\alpha'_b$ for all $b \in B$.
- CLAIM. $c'(b) \geq c(b)$ for all b.
  - So $c(b) = n \Rightarrow c'(b) \geq n$, a contradiction.

    recall that $\alpha'_b \in \{0, -1, \ldots, -(n-1)\}$, so $0 \leq c'(b) \leq n - 1$.

Let c_0 be the current c-function in our algorithm.

- Assume $c'(b) \geq c_0(b)$ for all b since $c(b) = 0$ is the initial c-function.

LEMMA. If b is unmatched in the matching M in G_{c_0} then $c'(b) > c_0(b)$.

When the algorithm says G has no popular assignment

We increase $c_0(b)$ only if b is unmatched in M.

- So for any vertex b unmatched in M: $c'(b) \geq c_0(b) + 1 = c_{\text{new}}(b)$.

  (assuming the lemma)

When the algorithm says G has no popular assignment

We increase $c_0(b)$ only if b is unmatched in M.

- So for any vertex b unmatched in M: $c'(b) \geq c_0(b) + 1 = c_{\text{new}}(b)$.
  (assuming the lemma)
- For any vertex b matched in M: $c'(b) \geq c_0(b) = c_{\text{new}}(b)$.

Thus $c'(b) \geq c_{\text{new}}(b)$ for all b.

When the algorithm says G has no popular assignment

We increase $c_0(b)$ only if b is unmatched in M.

- So for any vertex b unmatched in M: $c'(b) \geq c_0(b) + 1 = c_{\text{new}}(b)$.
  (assuming the lemma)
- For any vertex b matched in M: $c'(b) \geq c_0(b) = c_{\text{new}}(b)$.

Thus $c'(b) \geq c_{\text{new}}(b)$ for all b.

To show $c'(b) > c_0(b)$ for an unmatched b:

- Let M be our maximum matching in G_{c_0} and let b_0 be unmatched in M.

When the algorithm says G has no popular assignment

We increase $c_0(b)$ only if b is unmatched in M.

- So for any vertex b unmatched in M: $c'(b) \geq c_0(b) + 1 = c_{\text{new}}(b)$.
  (assuming the lemma)
- For any vertex b matched in M: $c'(b) \geq c_0(b) = c_{\text{new}}(b)$.

Thus $c'(b) \geq c_{\text{new}}(b)$ for all b.

To show $c'(b) > c_0(b)$ for an unmatched b:

- Let M be our maximum matching in G_{c_0} and let b_0 be unmatched in M.
  - $M \oplus M'$ contains an M-augmenting path P in G that starts at b_0.

When the algorithm says G has no popular assignment

We increase $c_0(b)$ only if b is unmatched in M.

- So for any vertex b unmatched in M: $c'(b) \geq c_0(b) + 1 = c_{\text{new}}(b)$.
  (assuming the lemma)
- For any vertex b matched in M: $c'(b) \geq c_0(b) = c_{\text{new}}(b)$.

Thus $c'(b) \geq c_{\text{new}}(b)$ for all b.

To show $c'(b) > c_0(b)$ for an unmatched b:

- Let M be our maximum matching in G_{c_0} and let b_0 be unmatched in M.
  - $M \oplus M'$ contains an M-augmenting path P in G that starts at b_0.
  - M is a maximum matching in G_{c_0} $\Rightarrow$ P has an edge not in E_{c_0}.

When the algorithm says G has no popular assignment

We increase $c_0(b)$ only if b is unmatched in M.

- So for any vertex b unmatched in M: $c'(b) \geq c_0(b) + 1 = c_{\text{new}}(b)$.
  (assuming the lemma)
- For any vertex b matched in M: $c'(b) \geq c_0(b) = c_{\text{new}}(b)$.

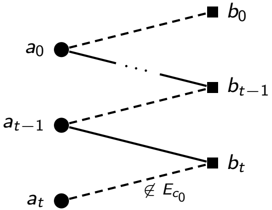
Thus $c'(b) \geq c_{\text{new}}(b)$ for all b.

To show $c'(b) > c_0(b)$ for an unmatched b:

- Let M be our maximum matching in G_{c_0} and let b_0 be unmatched in M.
  - $M \oplus M'$ contains an M-augmenting path P in G that starts at b_0.
  - M is a maximum matching in $G_{c_0} \Rightarrow P$ has an edge not in E_{c_0}.
  - Let $b_0 - a_0 - \ldots - b_t - a_t$ be any prefix of P such that $a_t b_t \notin E_{c_0}$.

A proof sketch

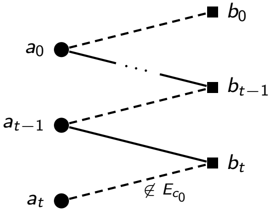
We have $a_i b_i \in M' \subseteq E_{c'}$ for all $i \in \{0, \ldots, t\}$.



And $a_i b_{i+1} \in M \subseteq E_{c_0}$ for all $i \in \{0, \ldots, t-1\}$.

A proof sketch

We have $a_i b_i \in M' \subseteq E_{c'}$ for all $i \in \{0, \ldots, t\}$.



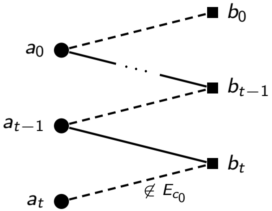
And $a_i b_{i+1} \in M \subseteq E_{c_0}$ for all $i \in \{0, \ldots, t-1\}$.

We will first show that $c'(b_t) > c_0(b_t)$.

- This will lead to the proof that $c'(b_0) > c_0(b_0)$.

To show $c'(b_t) > c_0(b_t)$

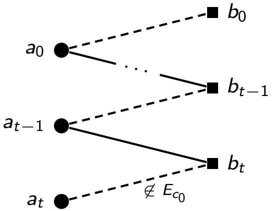
Since $a_t b_t \notin E_{c_0}$, one of cases 1-3 holds:



1. a_t has a neighbor with colour at least $c_0(b_t) + 2$.
2. a_t has a neighbor with colour $c_0(b_t) + 1$ that is at least as good as b_t.
3. a_t has a neighbor with colour $c_0(b_t)$ that is better than b_t.

To show $c'(b_t) > c_0(b_t)$

Since $a_t b_t \notin E_{c_0}$, one of cases 1-3 holds:



1. a_t has a neighbor with colour at least $c_0(b_t) + 2$.
2. a_t has a neighbor with colour $c_0(b_t) + 1$ that is at least as good as b_t.
3. a_t has a neighbor with colour $c_0(b_t)$ that is better than b_t.

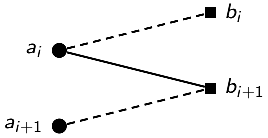
Suppose $c'(b_t) = c_0(b_t)$. Since $c'(b) \geq c_0(b)$ for all b:

- in each case we get $a_t b_t \notin E_{c'}$, a contradiction (since $a_t b_t \in M' \subseteq E_{c'}$).

Thus $c'(b_t) > c_0(b_t)$.

To show that $c'(b_0) > c_0(b_0)$

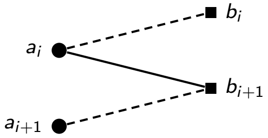
Suppose $\exists i \in \{0, \ldots, t-1\}$ with $c'(b_{i+1}) > c_0(b_{i+1})$ but $c'(b_i) = c_0(b_i)$.



Since $a_i b_{i+1} \in M \subseteq E_{c_0}$, we have the following possibilities:

To show that $c'(b_0) > c_0(b_0)$

Suppose $\exists i \in \{0, \ldots, t-1\}$ with $c'(b_{i+1}) > c_0(b_{i+1})$ but $c'(b_i) = c_0(b_i)$.



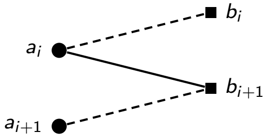
Since $a_i b_{i+1} \in M \subseteq E_{c_0}$, we have the following possibilities:

- $c_0(b_{i+1}) \geq c_0(b_i) + 1$: then $c'(b_{i+1}) \geq c'(b_i) + 2$.
- $c_0(b_{i+1}) = c_0(b_i)$: then $b_i \not\succ_{a_i} b_{i+1}$, but $c'(b_{i+1}) \geq c'(b_i) + 1$.
- $c_0(b_{i+1}) = c_0(b_i) - 1$: then $b_{i+1} \succ_{a_i} b_i$, but $c'(b_{i+1}) \geq c'(b_i)$.

In each case, we get $a_i b_i \notin E_{c'}$ by the definition of $G_{c'}$, a contradiction.

To show that $c'(b_0) > c_0(b_0)$

Suppose $\exists i \in \{0, \ldots, t-1\}$ with $c'(b_{i+1}) > c_0(b_{i+1})$ but $c'(b_i) = c_0(b_i)$.



Since $a_i b_{i+1} \in M \subseteq E_{c_0}$, we have the following possibilities:

- $c_0(b_{i+1}) \geq c_0(b_i) + 1$: then $c'(b_{i+1}) \geq c'(b_i) + 2$.
- $c_0(b_{i+1}) = c_0(b_i)$: then $b_i \not\succ_{a_i} b_{i+1}$, but $c'(b_{i+1}) \geq c'(b_i) + 1$.
- $c_0(b_{i+1}) = c_0(b_i) - 1$: then $b_{i+1} \succ_{a_i} b_i$, but $c'(b_{i+1}) \geq c'(b_i)$.

In each case, we get $a_i b_i \notin E_{c'}$ by the definition of $G_{c'}$, a contradiction.

- Thus $c'(b_i) > c_0(b_i)$ for all $i \in \{0, \ldots, t-1\}$, so $c'(b_0) > c_0(b_0)$. $\square$

Matroids

A matroid $M = (E, \mathcal{I})$:

- E is the ground set;
- $\mathcal{I} \subseteq 2^E$ is called the family of *independent sets*.

The collection of sets $\mathcal{I}$ must satisfy these two axioms:

- $X \subseteq Y$ and $Y \in \mathcal{I} \Rightarrow X \in \mathcal{I}$.
- $X, Y \in \mathcal{I}$ and $|Y| > |X| \Rightarrow \exists e \in Y \setminus X$ s.t. $X \cup \{e\} \in \mathcal{I}$.

Matroids

A matroid $M = (E, \mathcal{I})$:

- E is the ground set;
- $\mathcal{I} \subseteq 2^E$ is called the family of *independent sets*.

The collection of sets $\mathcal{I}$ must satisfy these two axioms:

- $X \subseteq Y$ and $Y \in \mathcal{I} \Rightarrow X \in \mathcal{I}$.
- $X, Y \in \mathcal{I}$ and $|Y| > |X| \Rightarrow \exists e \in Y \setminus X$ s.t. $X \cup \{e\} \in \mathcal{I}$.

An example: partition matroid.

- E is partitioned into disjoint sets $E_1, E_2, \ldots, E_n$;
- $\mathcal{I} = \{X \subseteq E : |X \cap E_i| \leq 1 \ \forall i = 1, \ldots, n\}$.

Matroids

A matroid $M = (E, \mathcal{I})$:

- E is the ground set;
- $\mathcal{I} \subseteq 2^E$ is called the family of *independent sets*.

The collection of sets $\mathcal{I}$ must satisfy these two axioms:

- $X \subseteq Y$ and $Y \in \mathcal{I} \Rightarrow X \in \mathcal{I}$.
- $X, Y \in \mathcal{I}$ and $|Y| > |X| \Rightarrow \exists e \in Y \setminus X$ s.t. $X \cup \{e\} \in \mathcal{I}$.

An example: partition matroid.

- E is partitioned into disjoint sets $E_1, E_2, \ldots, E_n$;
- $\mathcal{I} = \{X \subseteq E : |X \cap E_i| \leq 1 \ \forall i = 1, \ldots, n\}$.

Matroid intersection: 2 matroids $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$.

- Independent sets common to both matroids: sets in $\mathcal{I}_1 \cap \mathcal{I}_2$.

Matroid intersection

Matchings in bipartite graphs: Common independent sets in the intersection of two partition matroids.

Matroid intersection

Matchings in bipartite graphs: Common independent sets in the intersection of two partition matroids.

Maximum matchings in bipartite graphs: largest common independent sets in the intersection of two partition matroids.

- Largest common independent set: also called a common base.

Matroid intersection

Matchings in bipartite graphs: Common independent sets in the intersection of two partition matroids.

Maximum matchings in bipartite graphs: largest common independent sets in the intersection of two partition matroids.

- Largest common independent set: also called a common base.

In the intersection of two partition matroids:

(where one matroid is $E = \cup_a \delta(a)$ and each a has preferences over $\delta(a)$)

- a popular matching is a popular common independent set;
- a popular assignment is a popular common base.

Matroid intersection

Matchings in bipartite graphs: Common independent sets in the intersection of two partition matroids.

Maximum matchings in bipartite graphs: largest common independent sets in the intersection of two partition matroids.

- Largest common independent set: also called a common base.

In the intersection of two partition matroids:
(where one matroid is $E = \cup_a \delta(a)$ and each a has preferences over $\delta(a)$)

- a popular matching is a popular common independent set;
- a popular assignment is a popular common base.

When one matroid is a partition matroid and the other is any arbitrary matroid $(E, \mathcal{I})$:

- the popular assignment algorithm can be generalized to solve:
  - the popular common base/independent set problems [KMSY24];
  - thus several other "popular problems" can be solved efficiently.

References

1. D. J. Abraham, R. W. Irving, T. Kavitha, and K. Mehlhorn.
   Popular matchings. SIAM Journal on Computing, 2007.

2. T. Kavitha.
   A size-popularity tradeoff in the stable marriage problem. SIAM Journal on Computing, 2014.

3. T. Kavitha, T. Király, J. Matuschke, I. Schlotter, U. Schmidt-Kraepelin. *The popular assignment problem: when cardinality is more important than popularity.* In SODA 2022.

4. T. Kavitha, K. Makino, I. Schlotter, Y. Yokoi. *Arborescences, colorful forests, and popularity.* In SODA 2024 (to appear).

——————————— *Thank you!* ———————————