Popular Matchings - Part I

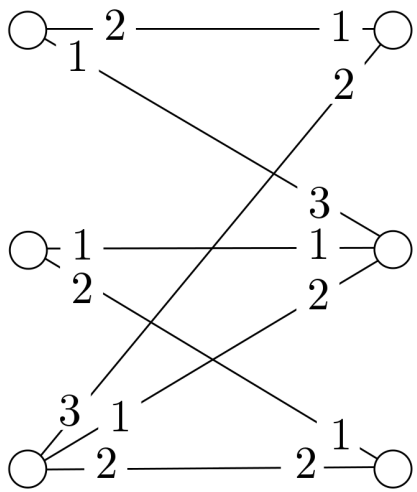
Kavitha Telikepalli

(TIFR, Mumbai)

School on Computational Social Choice
and Economics @ IIT Jodhpur

The input

A bipartite graph G where every vertex v has a strict ranking $\succ_v$ of its neighbors.



A well-studied model used in many two-sided markets:

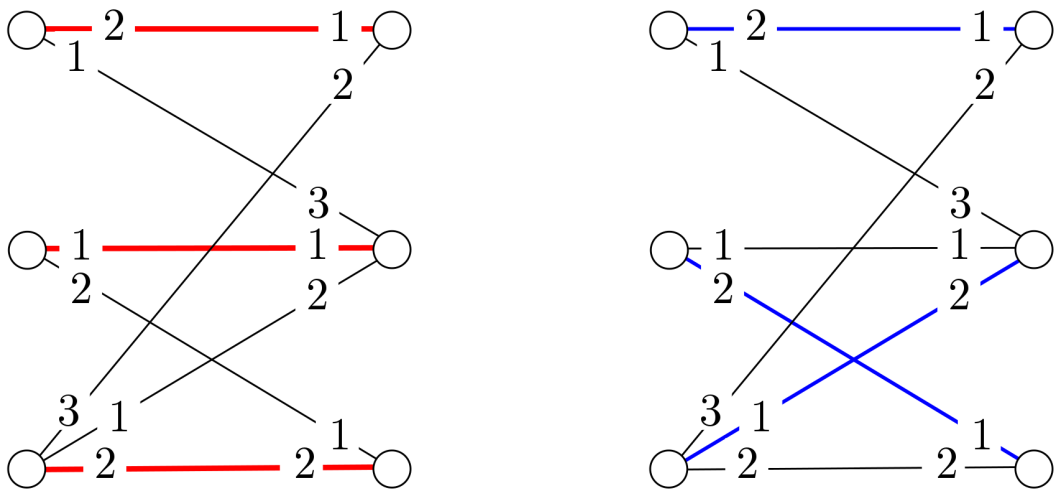
- students to schools;
- medical residents to hospitals;
- agents to jobs.

Stability

A matching M is stable if there is no edge ab such that:

$$b \succ_a M(a) \qquad \text{and} \qquad a \succ_b M(b)$$

(i.e., a and b prefer each other to their respective assignments in M)



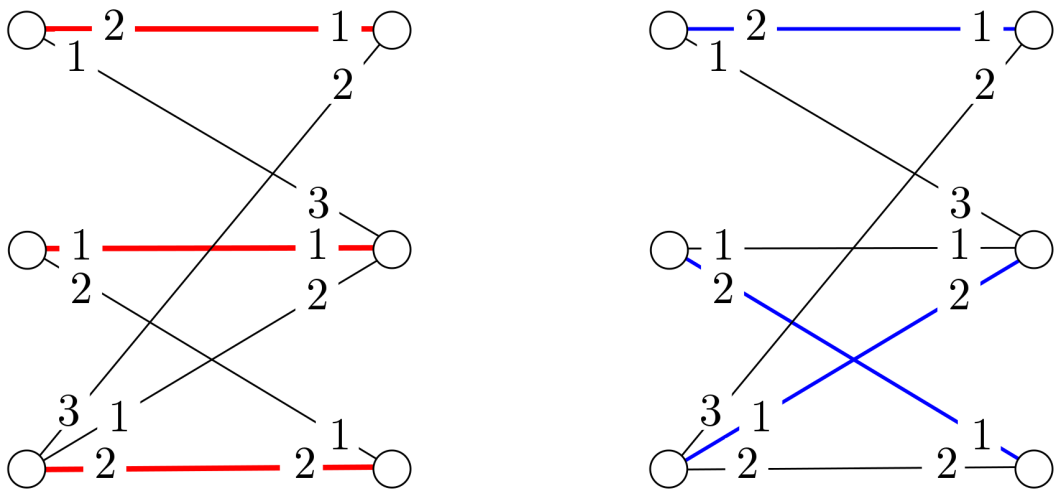
Such an edge ab is said to *block* M.

Stability

A matching M is stable if there is no edge ab such that:

$$b \succ_a M(a) \qquad \text{and} \qquad a \succ_b M(b)$$

(i.e., a and b prefer each other to their respective assignments in M)



Such an edge ab is said to *block* M.

- The red matching is stable but the blue one is not.

Stable matchings

Do stable matchings always exist? Can we find one efficiently?

- Yes (Gale and Shapley, 1962).

Stable matchings

Do stable matchings always exist? Can we find one efficiently?

- Yes (Gale and Shapley, 1962).

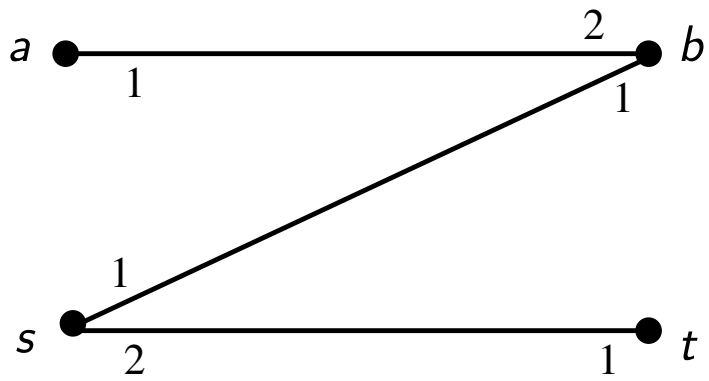
The Gale-Shapley algorithm: agents propose and jobs dispose — this is a very simple and clean algorithm.

Stable matchings

Do stable matchings always exist? Can we find one efficiently?

- Yes (Gale and Shapley, 1962).

The Gale-Shapley algorithm: agents propose and jobs dispose — this is a very simple and clean algorithm.



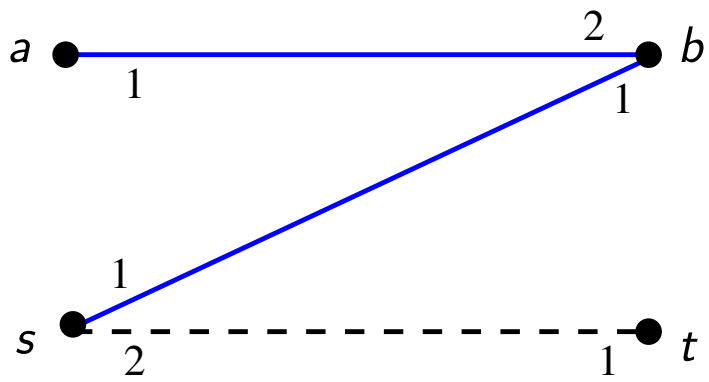
Let us run Gale-Shapley algorithm on this instance.

Stable matchings

Do stable matchings always exist? Can we find one efficiently?

- Yes (Gale and Shapley, 1962).

The Gale-Shapley algorithm: agents propose and jobs dispose — this is a very simple and clean algorithm.



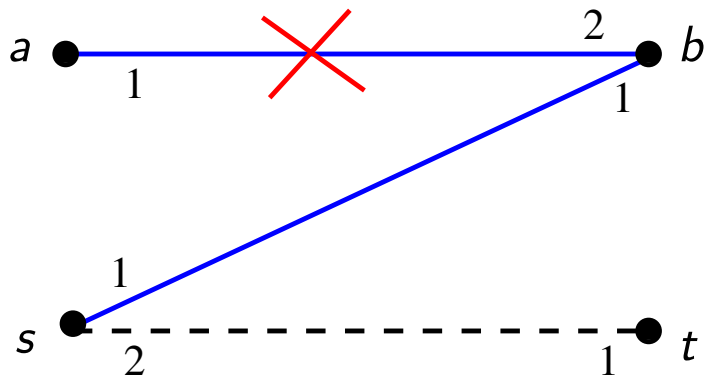
Initially both a and s propose to their top neighbour b.

Stable matchings

Do stable matchings always exist? Can we find one efficiently?

- Yes (Gale and Shapley, 1962).

The Gale-Shapley algorithm: agents propose and jobs dispose — this is a very simple and clean algorithm.



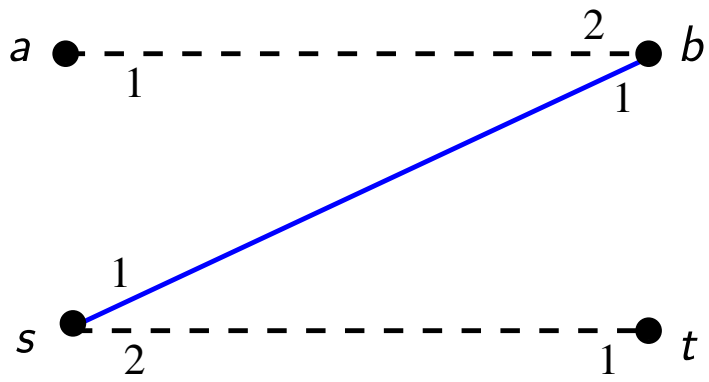
b rejects a's proposal and (tentatively) accepts s's proposal.

Stable matchings

Do stable matchings always exist? Can we find one efficiently?

- Yes (Gale and Shapley, 1962).

The Gale-Shapley algorithm: agents propose and jobs dispose — this is a very simple and clean algorithm.



a has no other neighbour to propose to; we get the matching $\{sb\}$.

Applications of stable matchings

Stable matchings are used in several problems in economics, computer science, and operations research.

Applications of stable matchings

Stable matchings are used in several problems in economics, computer science, and operations research.

To match students to schools in New York:

- How Game Theory Helped Improve New York City's High School Application Process, New York Times, December 5, 2014.

To match students to colleges in France:

- Stable Matching in Practice, Claire Mathieu. ESA 2018, Keynote talk.

To match students to engineering colleges in India:

- Centralized admissions for engineering colleges in India, S. Baswana, P. P. Chakrabarti, S. Chandran, Y. Kanoria, and U. Patange. INFORMS Journal on Applied Analytics, 2018.

Applications of stable matchings

To match medical residents to hospitals:

- National Resident Matching Program `http://www.nrmp.org/whythematch.pdf`
- Canadian Resident Matching Service
  `https://www.carms.ca/the-match/how-it-works/`

Applications of stable matchings

To match medical residents to hospitals:

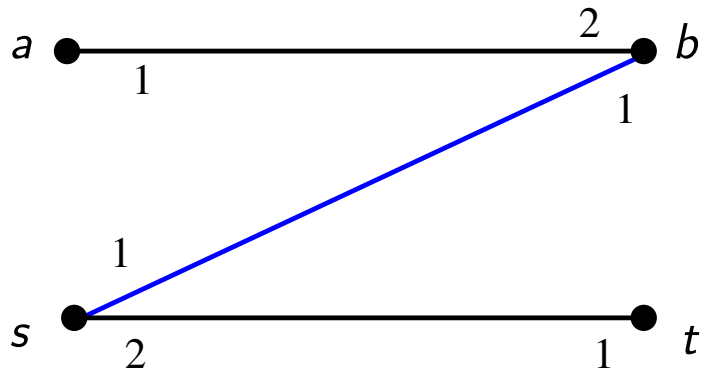
- National Resident Matching Program `http://www.nrmp.org/whythematch.pdf`
- Canadian Resident Matching Service `https://www.carms.ca/the-match/how-it-works/`

In all these applications, we want a large matching.

All stable matchings match the same subset of vertices (Rural Hospitals Theorem).

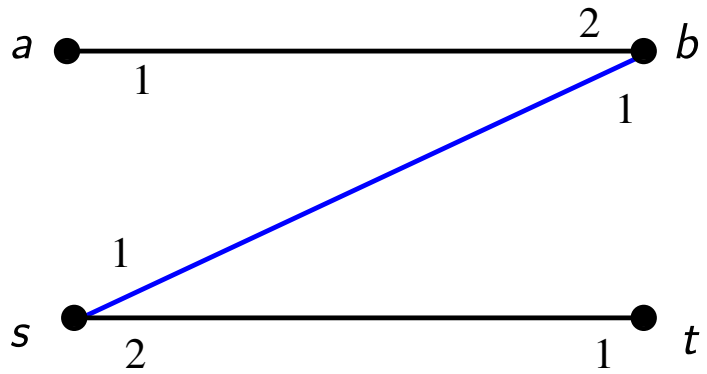
Size versus Stability

The size of a stable matching could be as low as $|M_{\max}|/2$.



Size versus Stability

The size of a stable matching could be as low as $|M_{\max}|/2$.

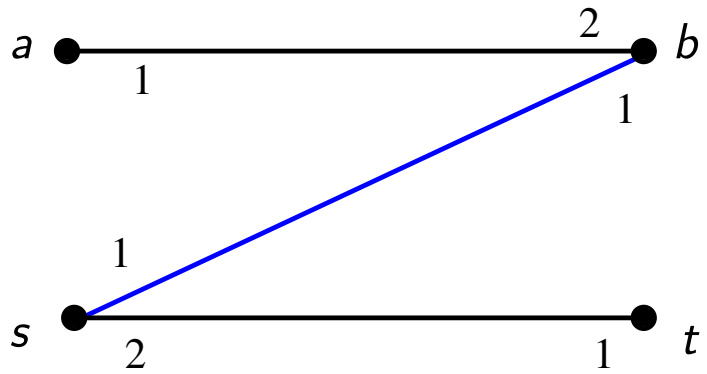


Recall that the max-size matching $\{ab, st\}$ is unstable.

- Forbidding blocking edges constrains the size of the matching.

Size versus Stability

The size of a stable matching could be as low as $|M_{\max}|/2$.



Recall that the max-size matching $\{ab, st\}$ is unstable.

- Forbidding blocking edges constrains the size of the matching.

Drawbacks of stability:

- Size can be half the size of a max-size matching;
- Models a situation where every edge has a "veto power".

Beyond stability

Can we relax stability so as to cope with these issues? We want a set that:

- contains stability as a special case;
- shifts the focus from "veto power" to "collective decision";
- allows for matchings of size larger than stable matchings.

Beyond stability

Can we relax stability so as to cope with these issues? We want a set that:

- contains stability as a special case;
- shifts the focus from "veto power" to "collective decision";
- allows for matchings of size larger than stable matchings.

$\Rightarrow$ Popular matchings

Beyond stability

Can we relax stability so as to cope with these issues? We want a set that:

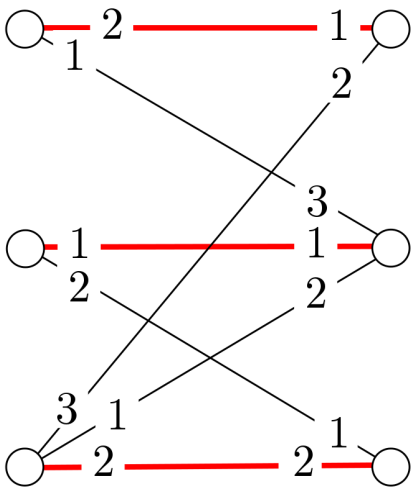
- contains stability as a special case;
- shifts the focus from "veto power" to "collective decision";
- allows for matchings of size larger than stable matchings.

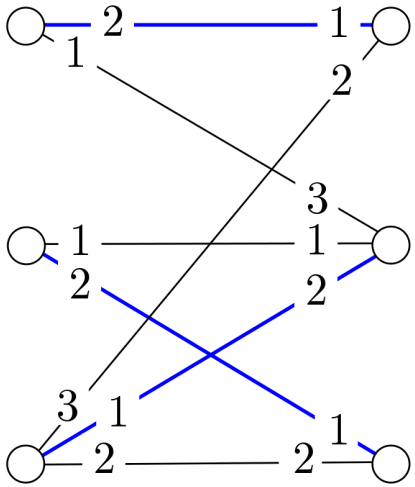
$\Rightarrow$ Popular matchings

- Any pair of matchings M and N can be compared via an M-vs-N election.
  - Every vertex votes for the matching in $\{M, N\}$ that it prefers.

Elections between pairs of matchings

Consider the election between the red and blue matchings.

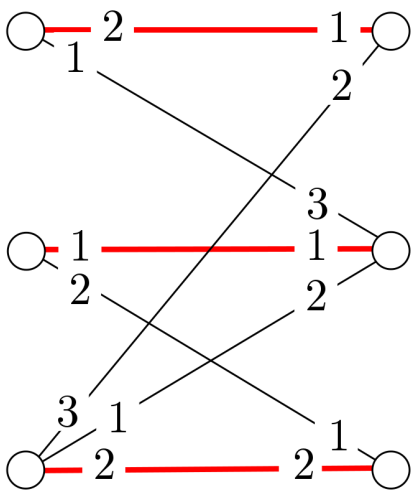


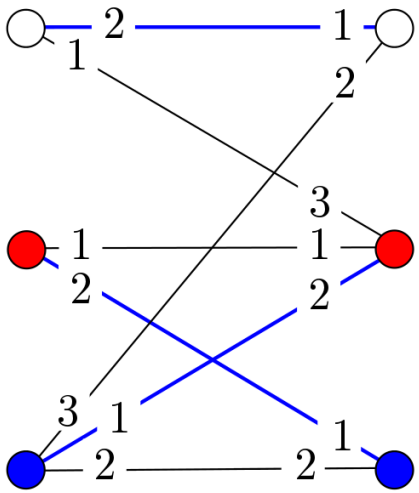


Elections between pairs of matchings

Consider the election between the red and blue matchings.

- this election is a tie.

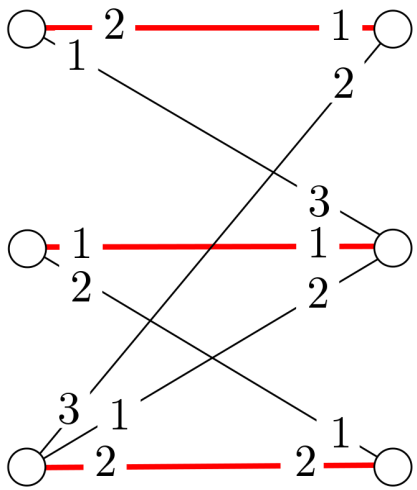


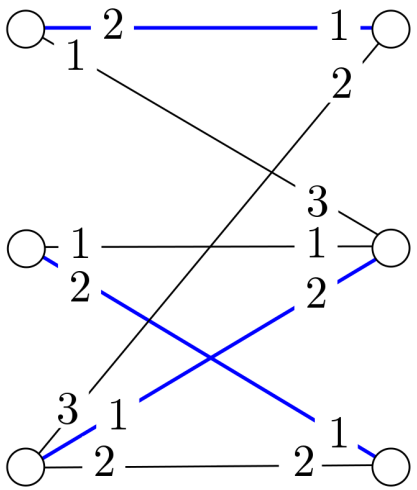


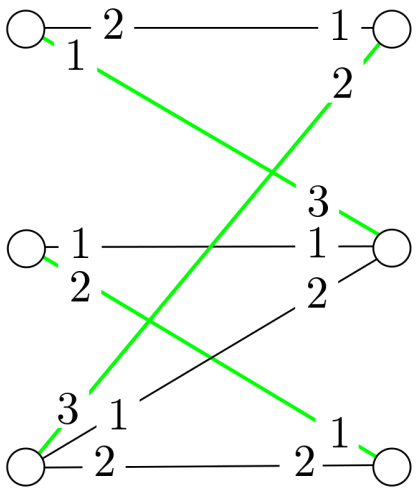
Elections between pairs of matchings

Consider the election between the red and blue matchings.

- this election is a tie.





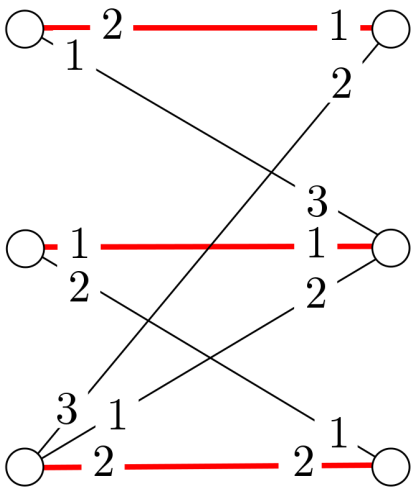


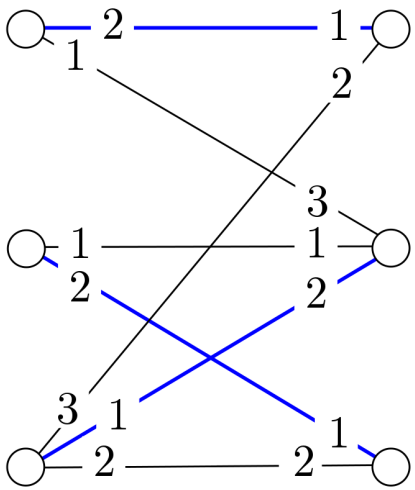
Consider the election between the red and green matchings.

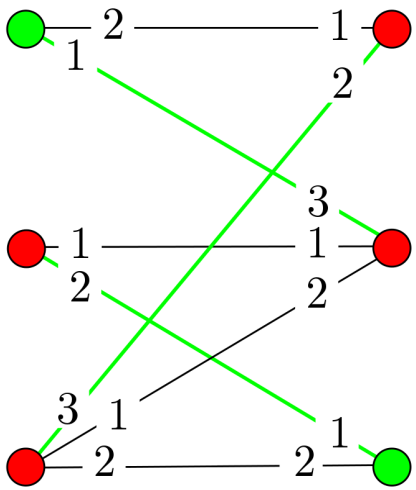
Elections between pairs of matchings

Consider the election between the red and blue matchings.

- this election is a tie.







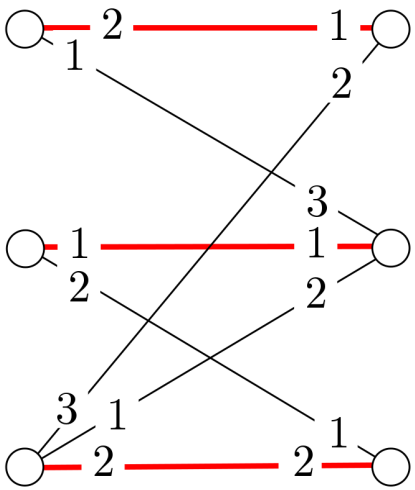
Consider the election between the red and green matchings.

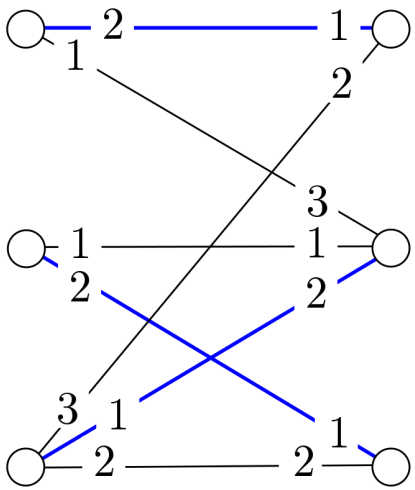
- the green matching loses this election.

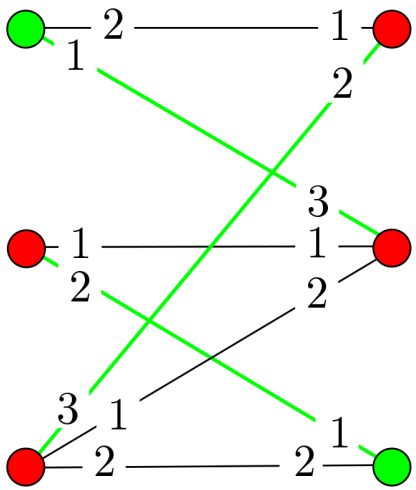
Elections between pairs of matchings

Consider the election between the red and blue matchings.

- this election is a tie.







Consider the election between the red and green matchings.

- the green matching loses this election.

A popular matching is one that does not lose any election.

Condorcet winner

Condorcet winner: A candidate who defeats every other candidate in their head-to-head election.

	30%	30%	40%
1	a	b	c
2	b	a	a
3	c	c	b

- Here a is the Condorcet winner.
- $a \succ b$ and $a \succ c$. (a defeats b and a defeats c)

Condorcet winner

Condorcet winner: A candidate who defeats every other candidate in their head-to-head election.

	30%	30%	40%
1	a	b	c
2	b	a	a
3	c	c	b

- Here a is the Condorcet winner.
- $a \succ b$ and $a \succ c$. (a defeats b and a defeats c)

Condorcet winner

Condorcet winner: A candidate who defeats every other candidate in their head-to-head election.

	30%	30%	40%
1	a	b	c
2	b	a	a
3	c	c	b

- Here a is the Condorcet winner.
- $a \succ b$ and $a \succ c$. (a defeats b and a defeats c)

Weak Condorcet winner

A weak Condorcet winner is one that is never defeated.

- x is a weak Condorcet winner $\implies x \succeq y$ for all candidates y.

Weak Condorcet winner

A weak Condorcet winner is one that is never defeated.

- x is a weak Condorcet winner $\implies x \succeq y$ for all candidates y.

However a (weak) Condorcet winner need not always exist.

	33.3%	33.3%	33.3%
1	a	b	c
2	b	c	a
3	c	a	b

- Here we have: $a \succ b \succ c \succ a$.

Weak Condorcet winner

A weak Condorcet winner is one that is never defeated.

- x is a weak Condorcet winner $\implies x \succeq y$ for all candidates y.

However a (weak) Condorcet winner need not always exist.

	33.3%	33.3%	33.3%
1	a	b	c
2	b	c	a
3	c	a	b

- Here we have: $a \succ b \succ c \succ a$.

Weak Condorcet winner

A weak Condorcet winner is one that is never defeated.

- x is a weak Condorcet winner $\implies x \succeq y$ for all candidates y.

However a (weak) Condorcet winner need not always exist.

	33.3%	33.3%	33.3%
1	a	b	c
2	b	c	a
3	c	a	b

- Here we have: $a \succ b \succ c \succ a$.

Weak Condorcet winner

A weak Condorcet winner is one that is never defeated.

- x is a weak Condorcet winner $\implies x \succeq y$ for all candidates y.

However a (weak) Condorcet winner need not always exist.

	33.3%	33.3%	33.3%
1	a	b	c
2	b	c	a
3	c	a	b

- Here we have: $a \succ b \succ c \succ a$.

Weak Condorcet winner in our setting

Matching M is a weak Condorcet winner $\iff$ $M \succeq N$ for all matchings N.

- So M does not lose a head-to-head election against any matching.
- Such a matching M is popular.

Weak Condorcet winner in our setting

Matching M is a weak Condorcet winner $\iff$ $M \succeq N$ for all matchings N.

- So M does not lose a head-to-head election against any matching.
- Such a matching M is popular.

Every stable matching is popular (Gärdenfors, 1975).

Weak Condorcet winner in our setting

Matching M is a weak Condorcet winner $\iff$ $M \succeq N$ for all matchings N.

- So M does not lose a head-to-head election against any matching.
- Such a matching M is popular.

Every stable matching is popular (Gärdenfors, 1975).

Comparing a stable matching S with any matching N:

- u prefers N to S $\implies$ $N(u)$ has to prefer S to N;

  (otherwise the edge between u and $N(u)$ blocks S)
- so # of votes for $N \leq$ # of votes for S.

Popular matchings

Properties of popular matchings:

- contains stability as a special case;
- shifts the focus from "veto power" to "collective decision"; ✓
- allows for matchings of size larger than stable matchings.

Popular matchings

Properties of popular matchings:

- contains stability as a special case; ✓
- shifts the focus from "veto power" to "collective decision"; ✓
- allows for matchings of size larger than stable matchings.

Every stable matching is popular (Gärdenfors, 1975).

Popular matchings

Properties of popular matchings:

- contains stability as a special case; ✓
- shifts the focus from "veto power" to "collective decision"; ✓
- allows for matchings of size larger than stable matchings. ✓

Every stable matching is popular (Gärdenfors, 1975).

- Stable matchings are min-size popular matchings.

Popular matchings

Properties of popular matchings:

- contains stability as a special case; ✓
- shifts the focus from "veto power" to "collective decision"; ✓
- allows for matchings of size larger than stable matchings. ✓

Every stable matching is popular (Gärdenfors, 1975).

- Stable matchings are min-size popular matchings.

Is there an efficient algorithm to find a max-size popular matching?

A first attempt to find a max-size popular matching

Start with M = any stable matching in G.

While M is not a max-size popular matching do:

A first attempt to find a max-size popular matching

Start with M = any stable matching in G.

While M is not a max-size popular matching do:

- find a suitable augmenting path p with respect to M;
- augment M along p, i.e., $M = M \oplus p$.

A first attempt to find a max-size popular matching

Start with M = any stable matching in G.

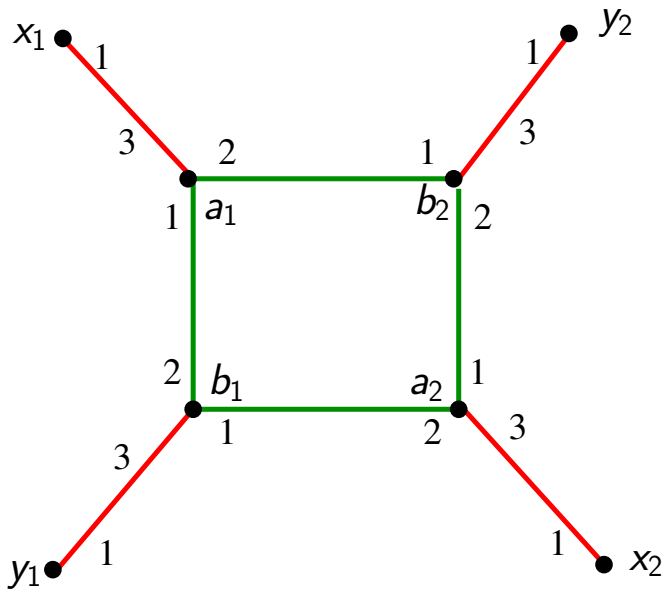
While M is not a max-size popular matching do:

- find a suitable augmenting path p with respect to M;
- augment M along p, i.e., $M = M \oplus p$.

The invariant should be that M is a popular matching in G.

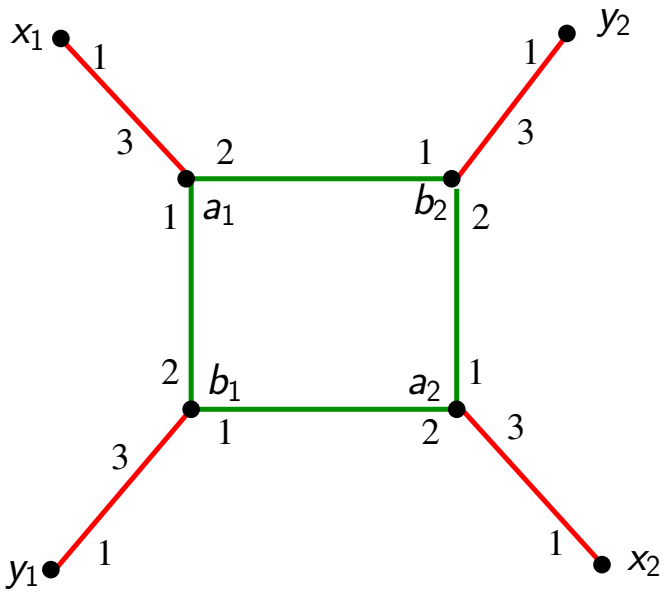
An interesting example

A stable matching has size 2 here while a max-size popular matching has size 4.



An interesting example

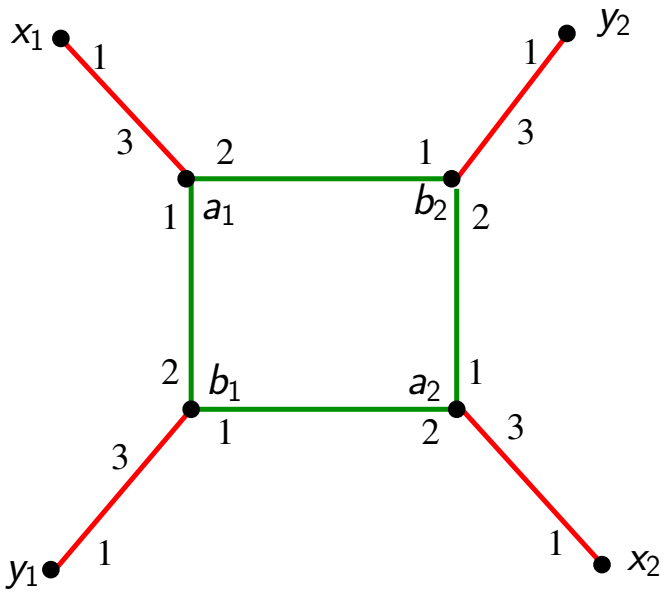
A stable matching has size 2 here while a max-size popular matching has size 4.



- There is no popular matching of size 3 here.
- So an "augmenting path" based approach would not work.

An interesting example

A stable matching has size 2 here while a max-size popular matching has size 4.



- There is no popular matching of size 3 here.
- So an "augmenting path" based approach would not work.

To find a max-size popular matching, can we adapt the Gale-Shapley algorithm?

To find a max-size popular matching

Stability is easy to check: no edge blocks a stable matching.

Popularity requires comparing our matching with all the matchings in G.

- Does a popular matching admit a simple *certificate* of its popularity?
- How do we show that *no* larger matching is popular?

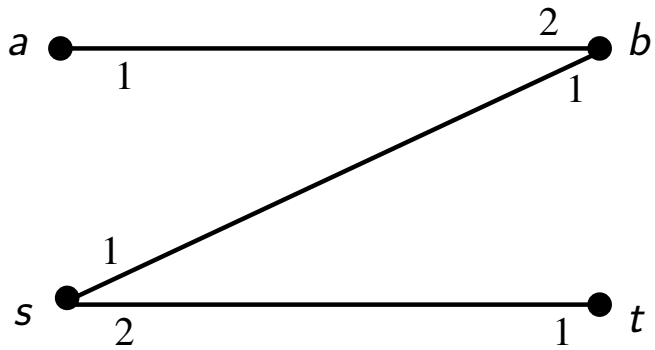
To find a max-size popular matching

Stability is easy to check: no edge blocks a stable matching.

Popularity requires comparing our matching with all the matchings in G.

- Does a popular matching admit a simple *certificate* of its popularity?
- How do we show that *no* larger matching is popular?

Let G be the graph below:



We want to find the matching $\{ab, st\}$.

- This is a max-size popular matching in this instance.

A new instance G'

A new graph G' such that $\{ab, st\}$ is the stable matching in G'?

A new instance G'

A new graph G' such that $\{ab, st\}$ is the stable matching in G'?

Suppose we replace every edge uv in G by the pair of edges uv and uv in G':

- that is, by two parallel edges: one red and the other blue.

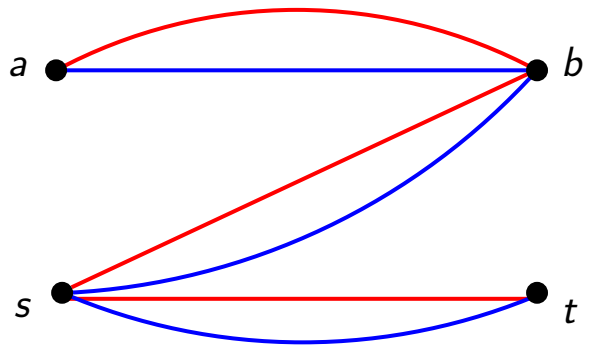
A new instance G'

A new graph G' such that $\{ab, st\}$ is the stable matching in G'?

Suppose we replace every edge uv in G by the pair of edges uv and uv in G':

- that is, by two parallel edges: one red and the other blue.

The corresponding graph G' is:



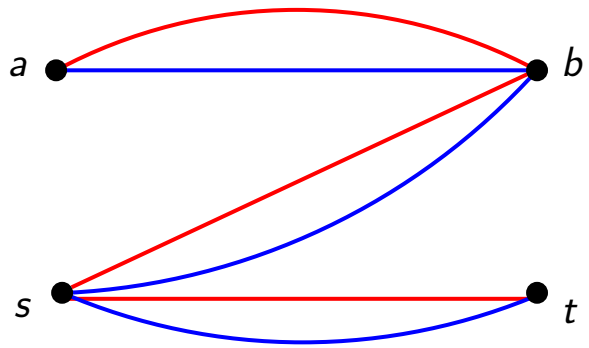
A new instance G'

A new graph G' such that $\{ab, st\}$ is the stable matching in G'?

Suppose we replace every edge uv in G by the pair of edges uv and uv in G':

- that is, by two parallel edges: one red and the other blue.

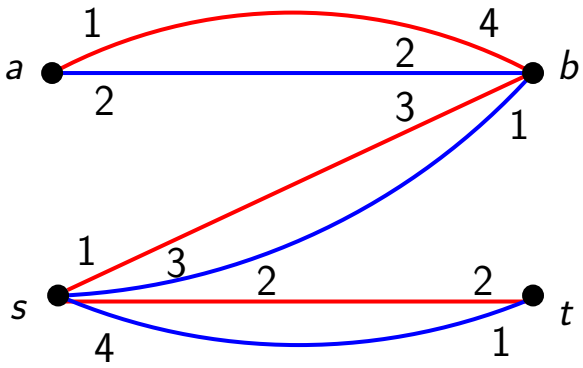
The corresponding graph G' is:



- Every vertex in A prefers any red edge to any blue edge.
- Every vertex in B prefers any blue edge to any red edge.

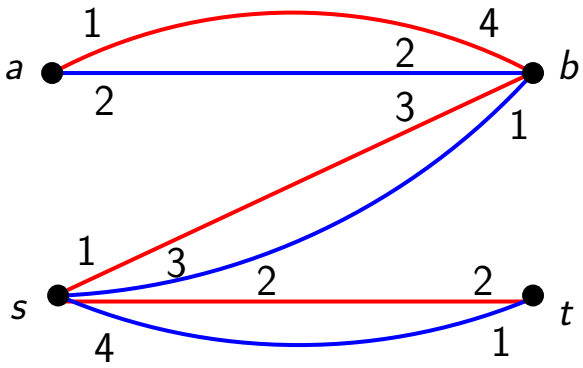
A new instance G'

So the graph G' with preferences is:



A new instance G'

So the graph G' with preferences is:

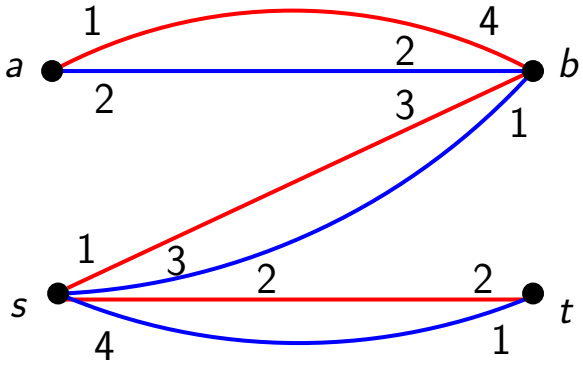


- The preference order of s in G is $sb \succ_s st$. Its preference order in G' is:

$$sb \succ_s st \succ_s sb \succ_s st.$$

A new instance G'

So the graph G' with preferences is:



- The preference order of s in G is $sb \succ_s st$. Its preference order in G' is:

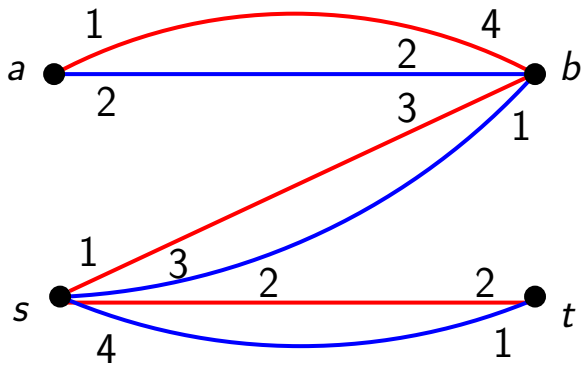
$$sb \succ_s st \succ_s sb \succ_s st.$$

- The preference order of b in G is $sb \succ_b ab$. Its preference order in G' is:

$$sb \succ_b ab \succ_b sb \succ_b ab.$$

A new instance G'

The graph G' with preferences is:

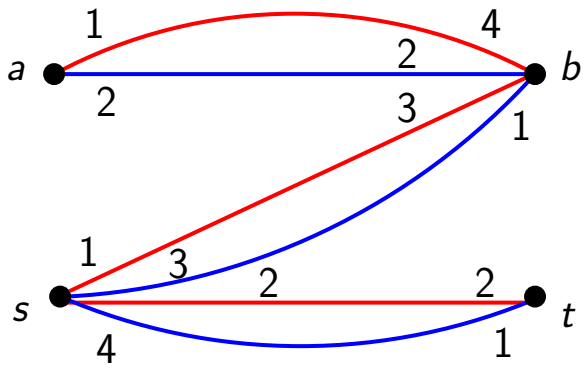


Recall the stable matching $\{sb\}$ in G.

- In the graph G', neither $\{sb\}$ nor $\{sb\}$ is stable.

A new instance G'

The graph G' with preferences is:

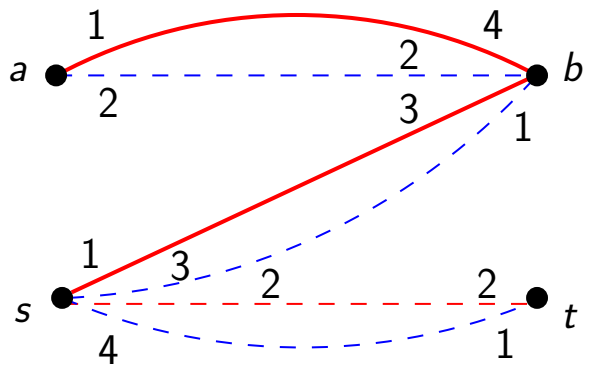


Recall the stable matching $\{sb\}$ in G.

- In the graph G', neither $\{sb\}$ nor $\{sb\}$ is stable.
  - The edge ab blocks the matching $\{sb\}$.
  - The edge st blocks the matching $\{sb\}$.

Computing a stable matching in G'

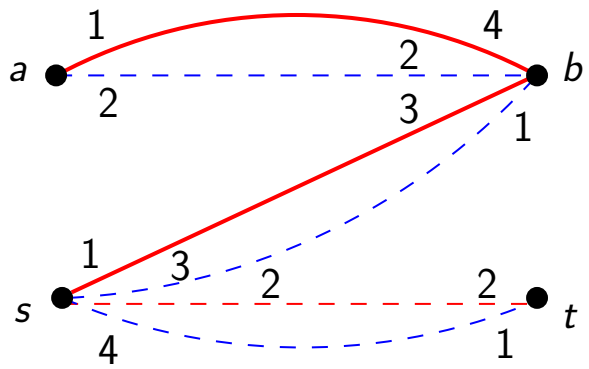
Let us run Gale-Shapley algorithm in G'.



- Both a and s propose to b along their red edges.

Computing a stable matching in G'

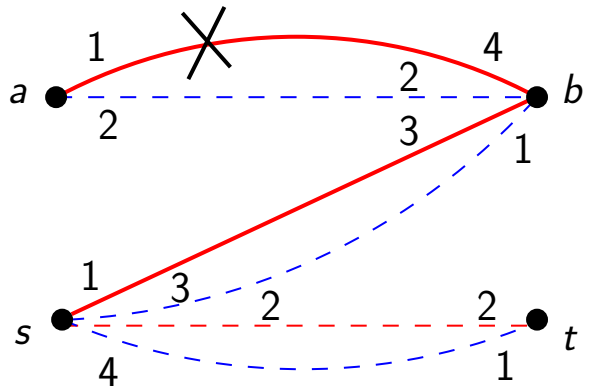
Let us run Gale-Shapley algorithm in G'.



- Both a and s propose to b along their red edges.
- b prefers s's proposal to a's proposal.

Computing a stable matching in G'

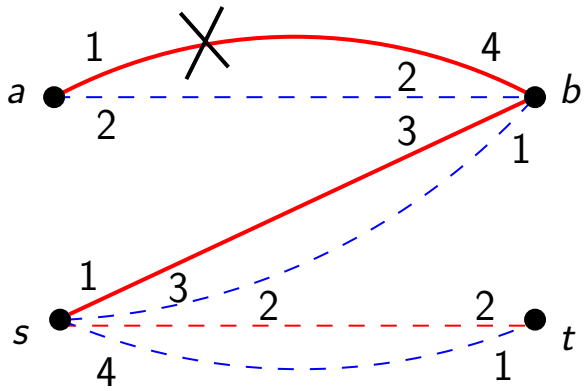
Let us run Gale-Shapley algorithm in G'.



- So b rejects a's proposal and (tentatively) accepts s's proposal.

Computing a stable matching in G'

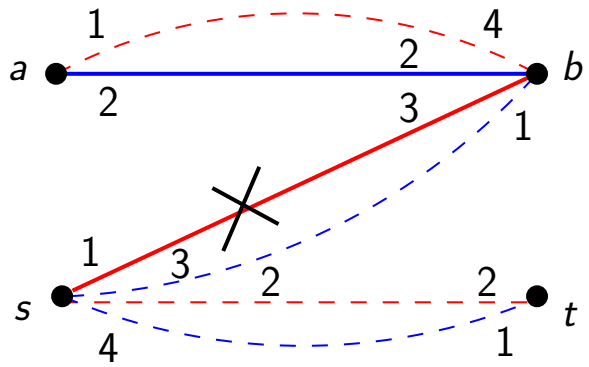
Let us run Gale-Shapley algorithm in G'.



- So b rejects a's proposal and (tentatively) accepts s's proposal.
- Then a proposes along its next favorite edge: this is ab.

Computing a stable matching in G'

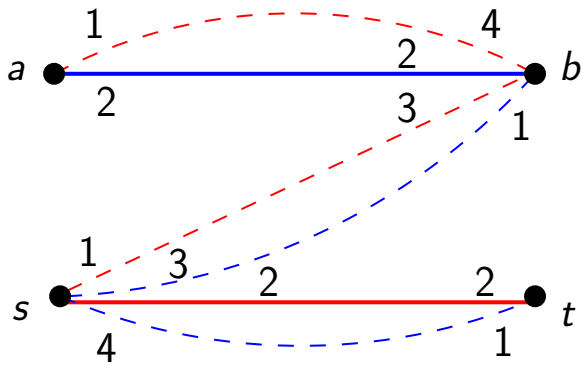
Let us run Gale-Shapley algorithm in G'.



- Observe that now b prefers a's proposal to s's proposal.
- So b rejects s's proposal and (tentatively) accepts a's proposal.

Computing a stable matching in G'

Let us run Gale-Shapley algorithm in G'.



- Then s proposes along its next most favorite edge st.
- t (tentatively) accepts s's proposal. This is the end of the algorithm.

Computing a stable matching in G'

All tentative proposals become definite.

So we get the stable matching $\{ab, st\}$ in G'.

- Ignoring colours, this is the desired matching $M = \{ab, st\}$ in G.

Computing a stable matching in G'

All tentative proposals become definite.

So we get the stable matching $\{ab, st\}$ in G'.

- Ignoring colours, this is the desired matching $M = \{ab, st\}$ in G.

Our algorithm in $G = (A \cup B, E)$

- Construct the red/blue graph $G' = (A \cup B, E')$.
- Run Gale-Shapley algorithm in G' to compute M'.
- Return the corresponding matching M in G.

Computing a stable matching in G'

All tentative proposals become definite.

So we get the stable matching $\{ab, st\}$ in G'.

- Ignoring colours, this is the desired matching $M = \{ab, st\}$ in G.

Our algorithm in $G = (A \cup B, E)$

- Construct the red/blue graph $G' = (A \cup B, E')$.
- Run Gale-Shapley algorithm in G' to compute M'.
- Return the corresponding matching M in G.

CLAIM. M is a max-size popular matching in G.

Analyzing our algorithm

Every popular matching admits a simple certificate of its popularity.

- The certificate for M is given by red/blue edge colours in the matching M'.

Analyzing our algorithm

Every popular matching admits a simple certificate of its popularity.

- The certificate for M is given by red/blue edge colours in the matching M'.

Let us define an edge weight function in G. For any edge ab:

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

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

Analyzing our algorithm

Every popular matching admits a simple certificate of its popularity.

- The certificate for M is given by red/blue edge colours in the matching M'.

Let us define an edge weight function in G. For any edge ab:

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

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

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

- Observation. For any edge e, $\mathrm{wt}_M(e) = 2 \iff e$ is a blocking edge to M.

An appropriate edge weight function

Let us augment G with self-loops:

- any matching $\rightsquigarrow$ a perfect matching via self-loops.

An appropriate edge weight function

Let us augment G with self-loops:

- any matching $\rightsquigarrow$ a perfect matching via self-loops.

For any self-loop uu:

$$\text{let } \text{wt}_M(uu) \;=\; \text{vote}_u(u, M(u)) \;=\; \begin{cases} 0 & \text{if } M(u) = u \\ -1 & \text{otherwise.} \end{cases}$$

An appropriate edge weight function

Let us augment G with self-loops:

- any matching $\rightsquigarrow$ a perfect matching via self-loops.

For any self-loop uu:

$$\text{let } \text{wt}_M(uu) \;=\; \text{vote}_u(u, M(u)) \;=\; \begin{cases} 0 & \text{if } M(u) = u \\ -1 & \text{otherwise.} \end{cases}$$

OBSERVATION. For any perfect matching N:

$$\text{wt}_M(N) \;=\; \#\text{ of votes for } N \;-\; \#\text{ of votes for } M.$$

An appropriate edge weight function

Let us augment G with self-loops:

- any matching $\rightsquigarrow$ a perfect matching via self-loops.

For any self-loop uu:

$$\text{let } \text{wt}_M(uu) \;=\; \text{vote}_u(u, M(u)) \;=\; \begin{cases} 0 & \text{if } M(u) = u \\ -1 & \text{otherwise.} \end{cases}$$

OBSERVATION. For any perfect matching N:

$$\text{wt}_M(N) \;=\; \#\text{ of votes for } N \;-\; \#\text{ of votes for } M.$$

- M is popular $\iff$ $\text{wt}_M(N) \leq 0$ for any perfect matching N.

An appropriate edge weight function

Let us augment G with self-loops:

- any matching $\rightsquigarrow$ a perfect matching via self-loops.

For any self-loop uu:

$$\text{let } \text{wt}_M(uu) \;=\; \text{vote}_u(u, M(u)) \;=\; \begin{cases} 0 & \text{if } M(u) = u \\ -1 & \text{otherwise.} \end{cases}$$

OBSERVATION. For any perfect matching N:

$$\text{wt}_M(N) \;=\; \#\text{ of votes for } N \;-\; \#\text{ of votes for } M.$$

- M is popular $\iff$ $\text{wt}_M(N) \leq 0$ for any perfect matching N.

  $\iff$ any perfect matching in G with edge weights given by wt_M has weight at most 0.

LP for max-weight perfect matching

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

$$\begin{aligned} \sum_{e \in \delta(u) \cup \{uu\}} x_e &= 1 \quad \forall u \in A \cup B \\ x_e &\geq 0 \quad \forall e \in E \cup \{\text{self-loops}\}. \end{aligned}$$

LP for max-weight perfect matching

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

$$\begin{aligned} \sum_{e \in \delta(u) \cup \{uu\}} x_e &= 1 \quad \forall u \in A \cup B \\ x_e &\geq 0 \quad \forall e \in E \cup \{\text{self-loops}\}. \end{aligned}$$

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

LP for max-weight perfect matching

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

$$\begin{aligned} \sum_{e \in \delta(u) \cup \{uu\}} x_e &= 1 \quad \forall u \in A \cup B \\ x_e &\geq 0 \quad \forall e \in E \cup \{\text{self-loops}\}. \end{aligned}$$

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

Dual LP

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

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

LP for max-weight perfect matching

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

$$\begin{aligned} \sum_{e \in \delta(u) \cup \{uu\}} x_e &= 1 \quad \forall u \in A \cup B \\ x_e &\geq 0 \quad \forall e \in E \cup \{\text{self-loops}\}. \end{aligned}$$

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

Dual LP

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

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

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

Dual certificate

Every stable matching S has a simple dual certificate: $\vec{\alpha} = \vec{0}$.

- This is because $\text{wt}_S(e) \leq 0$ for all edges e.

Does M computed by our algorithm have an easy-to-describe dual certificate?

Dual certificate

Every stable matching S has a simple dual certificate: $\vec{\alpha} = \vec{0}$.

- This is because $\mathrm{wt}_S(e) \leq 0$ for all edges e.

Does M computed by our algorithm have an easy-to-describe dual certificate?

For each vertex $a \in A$:

- a is matched along a red edge in M': set $\alpha_a = 1$.
- a is matched along a blue edge in M': set $\alpha_a = -1$.
- a is unmatched in M': set $\alpha_a = 0$.

Dual certificate

Every stable matching S has a simple dual certificate: $\vec{\alpha} = \vec{0}$.

- This is because $\mathrm{wt}_S(e) \leq 0$ for all edges e.

Does M computed by our algorithm have an easy-to-describe dual certificate?

For each vertex $a \in A$:

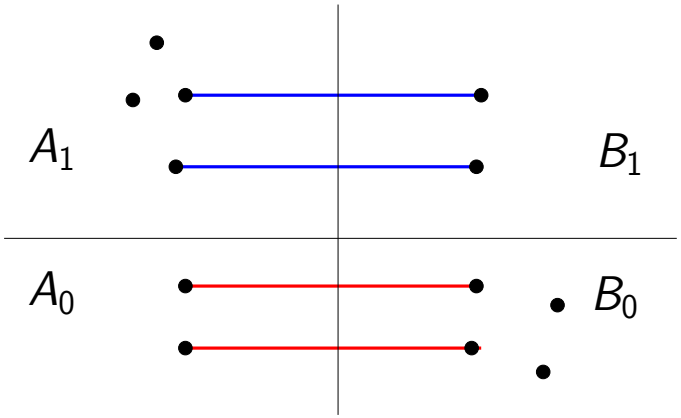
- a is matched along a red edge in M': set $\alpha_a = 1$.
- a is matched along a blue edge in M': set $\alpha_a = -1$.
- a is unmatched in M': set $\alpha_a = 0$.

For each vertex $b \in B$:

- b is matched along a red edge in M': set $\alpha_b = -1$.
- b is matched along a blue edge in M': set $\alpha_b = 1$.
- b is unmatched in M': set $\alpha_b = 0$.

Dual certificate

A useful picture:

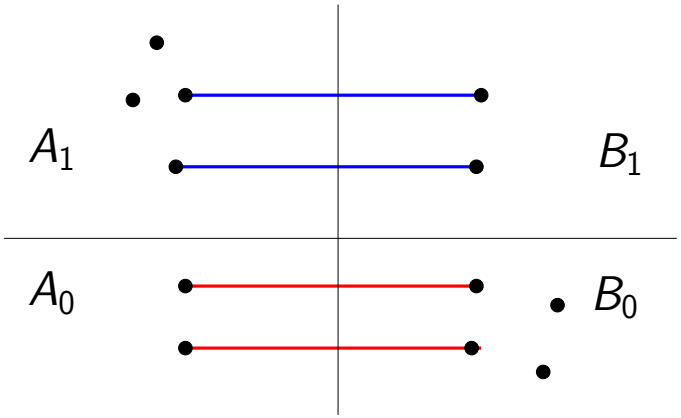


So vertices matched along red edges are in $A_0 \cup B_0$.

And vertices matched along blue edges are in $A_1 \cup B_1$.

Dual certificate

A useful picture:



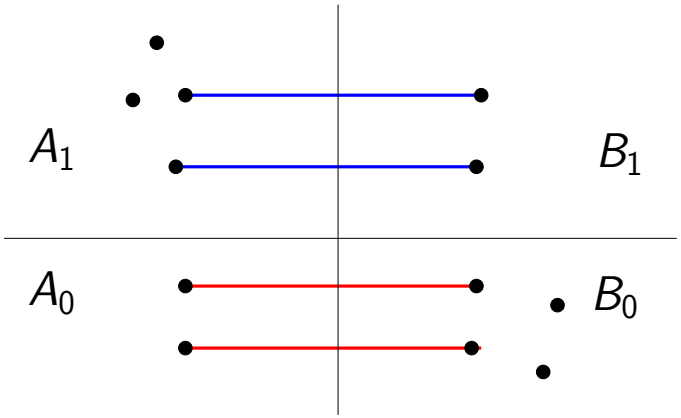
So vertices matched along red edges are in $A_0 \cup B_0$.

And vertices matched along blue edges are in $A_1 \cup B_1$.

- Unmatched vertices of A (resp., B) are in A_1 (resp., B_0).

Dual certificate

A useful picture:



So vertices matched along red edges are in $A_0 \cup B_0$.

And vertices matched along blue edges are in $A_1 \cup B_1$.

- Unmatched vertices of A (resp., B) are in A_1 (resp., B_0).

α-values were assigned as follows:

- $\alpha_u = 1$ for all $u \in A_0 \cup B_1$;
- $\alpha_u = -1$ for all matched $u \in A_1 \cup B_0$;
- $\alpha_u = 0$ for all unmatched u.

Dual feasibility of $\vec{\alpha}$

We need to show this vector $\vec{\alpha}$ is a feasible solution to the dual LP.

Dual LP

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

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

Dual feasibility of $\vec{\alpha}$

We need to show this vector $\vec{\alpha}$ is a feasible solution to the dual LP.

Dual LP

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

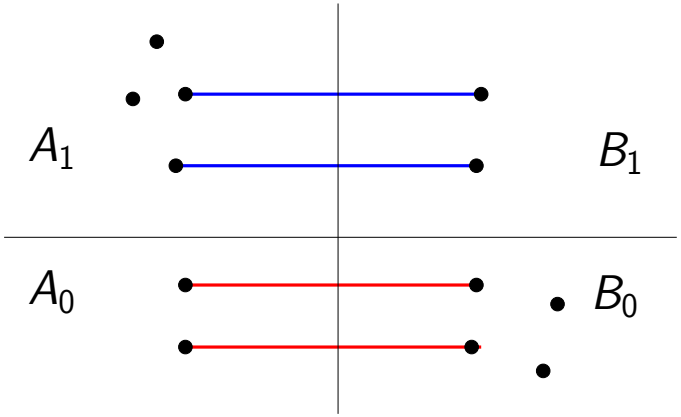
$$\begin{aligned} \alpha_a + \alpha_b &\geq \mathrm{wt}_M(ab) \quad \forall\ ab \in E \\ \alpha_u &\geq \mathrm{wt}_M(uu) \quad \forall\ u \in A \cup B. \end{aligned}$$

We will also show that $\sum_{u \in A \cup B} \alpha_u = 0$.

- This will mean the dual optimal solution is at most 0.
- This will prove M is a popular matching.

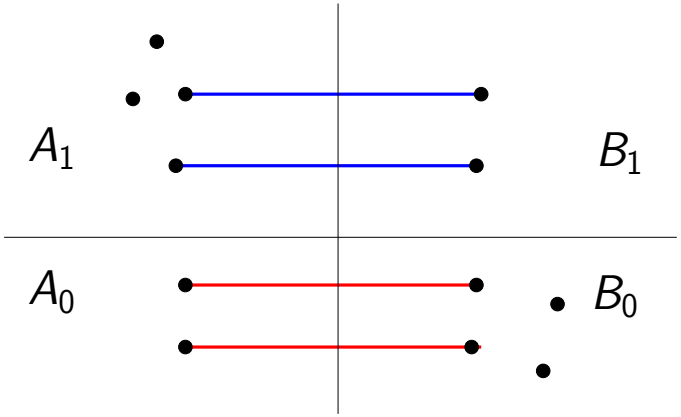
Dual feasibility of $\vec{\alpha}$

Recall that $\alpha_u \in \{0, \pm 1\}$:



Dual feasibility of $\vec{\alpha}$

Recall that $\alpha_u \in \{0, \pm 1\}$:

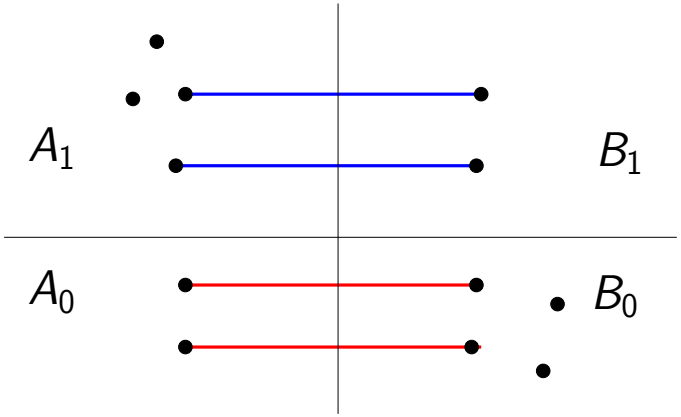


OBSERVATION. *The constraint* $\alpha_u \geq \mathrm{wt}_M(uu)$ *holds for all vertices* u.

- For a matched vertex u, we have $\alpha_u \geq -1 = \mathrm{wt}_M(uu)$.
- For an unmatched vertex u, we have $\alpha_u = 0 = \mathrm{wt}_M(uu)$.

Dual feasibility of $\vec{\alpha}$

Recall that $\alpha_u \in \{0, \pm 1\}$:



OBSERVATION. *The constraint* $\alpha_u \geq \mathrm{wt}_M(uu)$ *holds for all vertices* u.

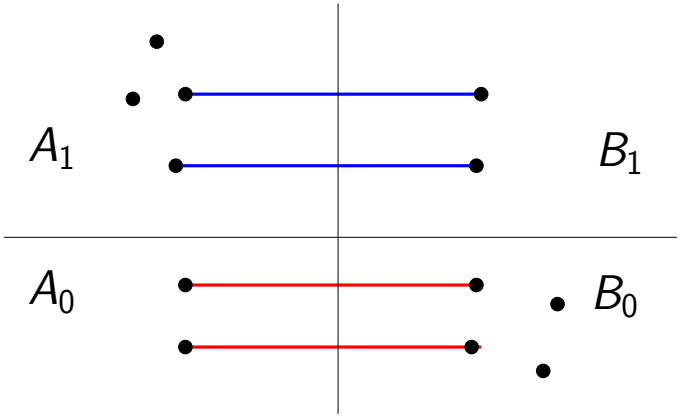
- For a matched vertex u, we have $\alpha_u \geq -1 = \mathrm{wt}_M(uu)$.
- For an unmatched vertex u, we have $\alpha_u = 0 = \mathrm{wt}_M(uu)$.

LEMMA. *The constraint* $\alpha_a + \alpha_b \geq \mathrm{wt}_M(ab)$ *holds for all* $ab \in E$.

- We will use the stability of M' in the instance G' to prove the lemma.

Dual feasibility of $\vec{\alpha}$

Recall that $\alpha_u \in \{0, \pm 1\}$:



Observation. *The constraint $\alpha_u \geq \mathrm{wt}_M(uu)$ holds for all vertices u.*

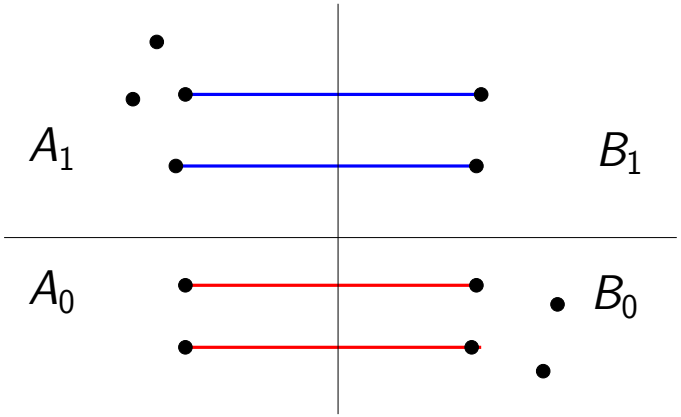
- For a matched vertex u, we have $\alpha_u \geq -1 = \mathrm{wt}_M(uu)$.
- For an unmatched vertex u, we have $\alpha_u = 0 = \mathrm{wt}_M(uu)$.

Lemma. *The constraint $\alpha_a + \alpha_b \geq \mathrm{wt}_M(ab)$ holds for all $ab \in E$.*

- We will use the stability of M' in the instance G' to prove the lemma.

Conclusion. So $\vec{\alpha}$ is dual-feasible.

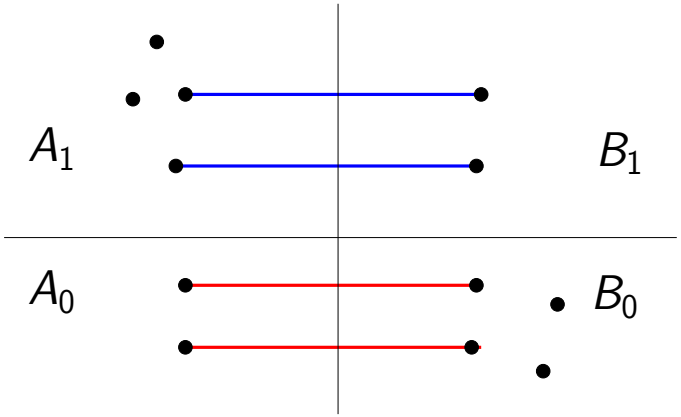
Optimal value of the dual LP



Every edge in M' is a red edge or a blue edge.

- So $\alpha_a + \alpha_b = 0$ for all $ab \in M$.
- Since $\alpha_u = 0$ for all unmatched vertices, $\sum_{u \in A \cup B} \alpha_u = 0$.

Optimal value of the dual LP



Every edge in M' is a red edge or a blue edge.

- So $\alpha_a + \alpha_b = 0$ for all $ab \in M$.
- Since $\alpha_u = 0$ for all unmatched vertices, $\sum_{u \in A \cup B} \alpha_u = 0$.

Thus the optimal value of the dual LP is at most 0.

- Hence M is a popular matching.

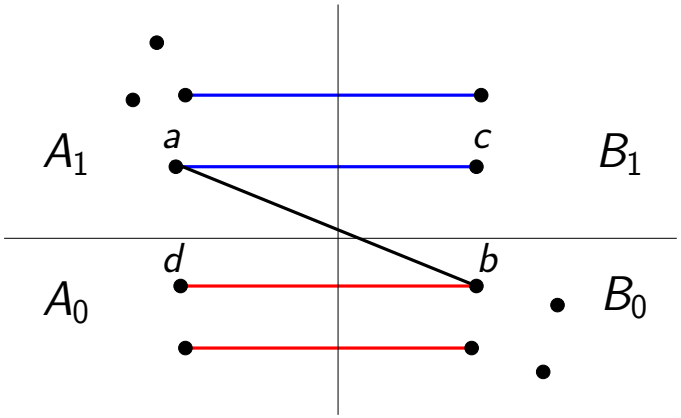
Proof of the lemma

To show $\alpha_a + \alpha_b \geq \mathrm{wt}_M(ab)$ holds for all $ab \in E$.

Proof of the lemma

To show $\alpha_a + \alpha_b \geq \text{wt}_M(ab)$ holds for all $ab \in E$.

Case 1. Suppose $\alpha_a = \alpha_b = -1$.

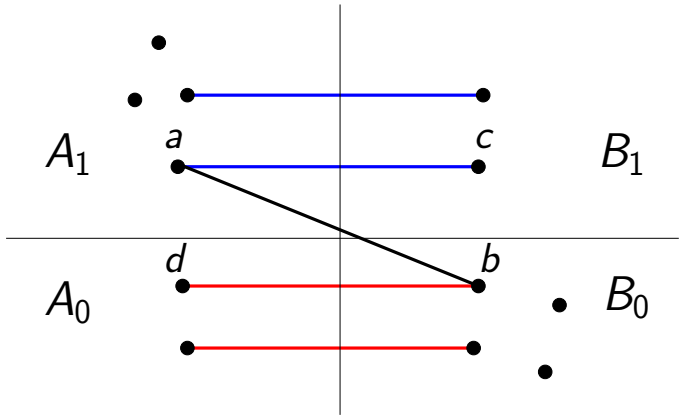


- So $ac \in M'$ and $bd \in M'$ for some neighbors c and d of a and b, respectively.

Proof of the lemma

To show $\alpha_a + \alpha_b \geq \text{wt}_M(ab)$ holds for all $ab \in E$.

Case 1. Suppose $\alpha_a = \alpha_b = -1$.

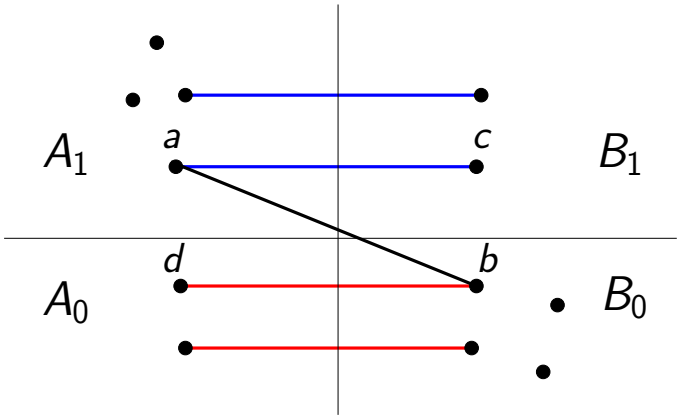


- So $ac \in M'$ and $bd \in M'$ for some neighbors c and d of a and b, respectively.
- Observe that (i) a prefers c to b and (ii) b prefers d to a.
  - This is because a never proposed along ab.
  - Furthermore, b rejected a's proposal along ab.

Proof of the lemma

To show $\alpha_a + \alpha_b \geq \text{wt}_M(ab)$ holds for all $ab \in E$.

Case 1. Suppose $\alpha_a = \alpha_b = -1$.

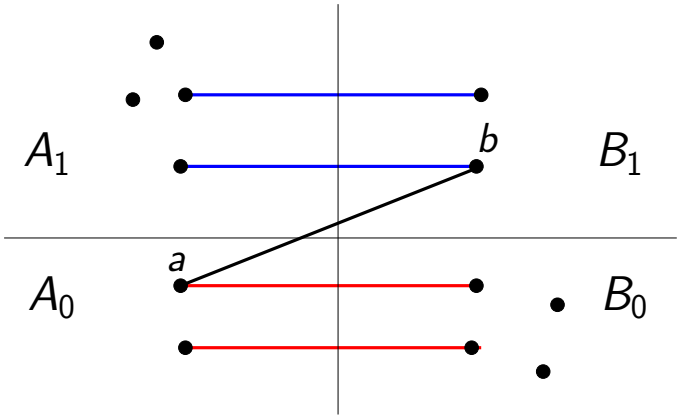


- So $ac \in M'$ and $bd \in M'$ for some neighbors c and d of a and b, respectively.
- Observe that (i) a prefers c to b and (ii) b prefers d to a.
  - This is because a never proposed along ab.
  - Furthermore, b rejected a's proposal along ab.

Thus $\text{wt}_M(ab) = -2$, hence $\alpha_a + \alpha_b = -2 = \text{wt}_M(ab)$.

Proof of the lemma

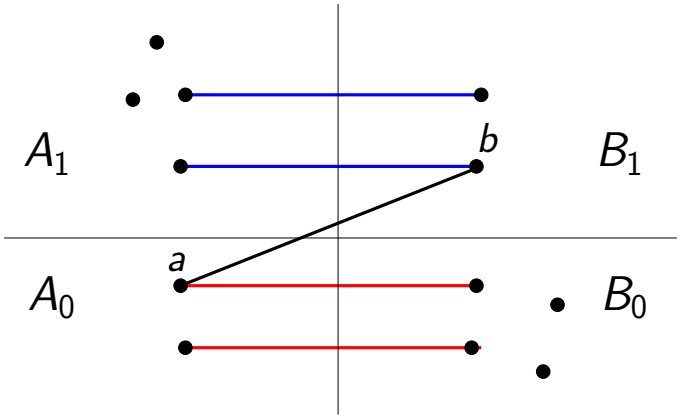
Case 2. Suppose $\alpha_a = \alpha_b = 1$.



- Since $\text{wt}_M(ab) \in \{0, \pm 2\}$, we have $\alpha_a + \alpha_b = 2 \geq \text{wt}_M(ab)$.

Proof of the lemma

Case 2. Suppose $\alpha_a = \alpha_b = 1$.



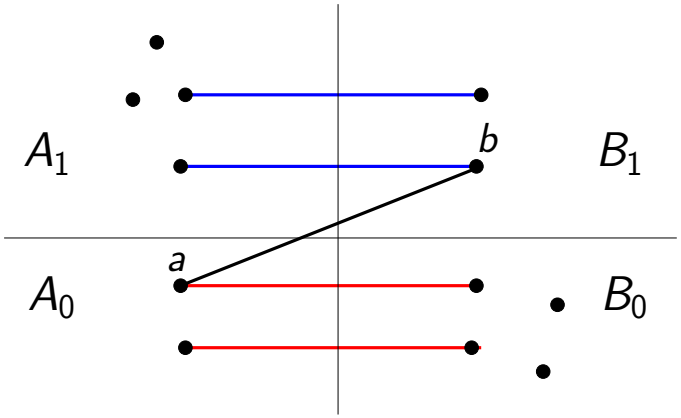
- Since $\text{wt}_M(ab) \in \{0, \pm 2\}$, we have $\alpha_a + \alpha_b = 2 \geq \text{wt}_M(ab)$.

Case 3. Suppose $\alpha_a = 1$ and $\alpha_b = -1$.

- This means ac and bd are in M' for some neighbors c and d.
- M' is stable in $G' \Rightarrow ab$ does not block M'.

Proof of the lemma

Case 2. Suppose $\alpha_a = \alpha_b = 1$.



- Since $\text{wt}_M(ab) \in \{0, \pm 2\}$, we have $\alpha_a + \alpha_b = 2 \geq \text{wt}_M(ab)$.

Case 3. Suppose $\alpha_a = 1$ and $\alpha_b = -1$.

- This means ac and bd are in M' for some neighbors c and d.
- M' is stable in $G' \Rightarrow ab$ does not block M'.

Thus $\text{wt}_M(ab) \leq 0$, hence $\alpha_a + \alpha_b = 0 \geq \text{wt}_M(ab)$.

Proof of the lemma

Case 4. Suppose $\alpha_a = -1$ and $\alpha_b = 1$.

- This means ac and bd are in M' for some neighbors c and d.
- M' is stable in $G' \Rightarrow ab$ does not block M'.

Thus $\text{wt}_M(ab) \leq 0$, hence $\alpha_a + \alpha_b = 0 \geq \text{wt}_M(ab)$.

Proof of the lemma

Case 4. Suppose $\alpha_a = -1$ and $\alpha_b = 1$.

- This means ac and bd are in M' for some neighbors c and d.
- M' is stable in $G' \Rightarrow ab$ does not block M'.

Thus $\text{wt}_M(ab) \leq 0$, hence $\alpha_a + \alpha_b = 0 \geq \text{wt}_M(ab)$.

Case 5. Suppose $\alpha_a = 0$.

Since M' is stable in G', ab does not block M'.

- This means $bd \in M'$ for some neighbor d that b prefers to a.

Proof of the lemma

Case 4. Suppose $\alpha_a = -1$ and $\alpha_b = 1$.

- This means ac and bd are in M' for some neighbors c and d.
- M' is stable in $G' \Rightarrow ab$ does not block M'.

Thus $\text{wt}_M(ab) \leq 0$, hence $\alpha_a + \alpha_b = 0 \geq \text{wt}_M(ab)$.

Case 5. Suppose $\alpha_a = 0$.

Since M' is stable in G', ab does not block M'.

- This means $bd \in M'$ for some neighbor d that b prefers to a.

Thus $\alpha_b = 1$, hence $\alpha_a + \alpha_b = 1 \geq 0 = \text{wt}_M(ab)$.

Proof of the lemma

Case 4. Suppose $\alpha_a = -1$ and $\alpha_b = 1$.

- This means ac and bd are in M' for some neighbors c and d.
- M' is stable in $G' \Rightarrow ab$ does not block M'.

Thus $\text{wt}_M(ab) \leq 0$, hence $\alpha_a + \alpha_b = 0 \geq \text{wt}_M(ab)$.

Case 5. Suppose $\alpha_a = 0$.

Since M' is stable in G', ab does not block M'.

- This means $bd \in M'$ for some neighbor d that b prefers to a.

Thus $\alpha_b = 1$, hence $\alpha_a + \alpha_b = 1 \geq 0 = \text{wt}_M(ab)$.

An analogous analysis holds when $\alpha_b = 0$.

Slack edges

For any edge ab incident to an unmatched vertex a:

- Recall that $\alpha_a = 0$ while $\alpha_b = 1$;
- so $\alpha_a + \alpha_b = 1 > 0 = \text{wt}_M(ab)$, thus the edge ab is *slack*.

Slack edges

For any edge ab incident to an unmatched vertex a:

- Recall that $\alpha_a = 0$ while $\alpha_b = 1$;
- so $\alpha_a + \alpha_b = 1 > 0 = \text{wt}_M(ab)$, thus the edge ab is *slack*.

Complementary Slackness

Any matching N with a slack edge is not an optimal solution to the primal LP;

- in other words, $\text{wt}_M(N) < 0$ (equivalently, M defeats N).

Slack edges

For any edge ab incident to an unmatched vertex a:

- Recall that $\alpha_a = 0$ while $\alpha_b = 1$;
- so $\alpha_a + \alpha_b = 1 > 0 = \text{wt}_M(ab)$, thus the edge ab is *slack*.

COMPLEMENTARY SLACKNESS

Any matching N with a slack edge is not an optimal solution to the primal LP;

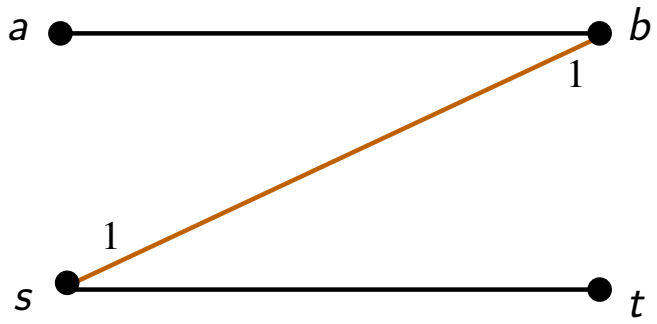
- in other words, $\text{wt}_M(N) < 0$ (equivalently, M defeats N).

Thus any matching larger than M is unpopular.

- So M is a max-size popular matching. □

Lower bound on $|M|$

Claim. There is no length 3 augmenting path wrt M in G.

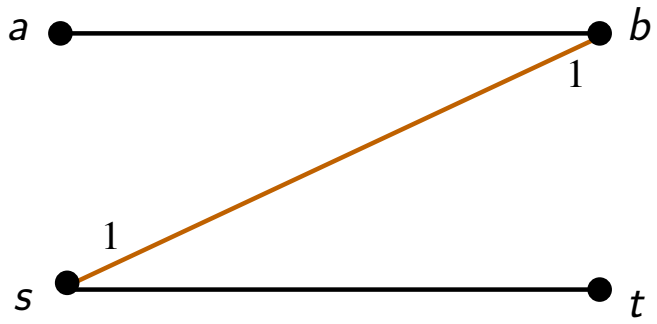


- $a - b - s - t$ is an augmenting path wrt $M \implies$ either ab or st blocks M'

  (a contradiction to M''s stability in G')

Lower bound on $|M|$

CLAIM. There is no length 3 augmenting path wrt M in G.



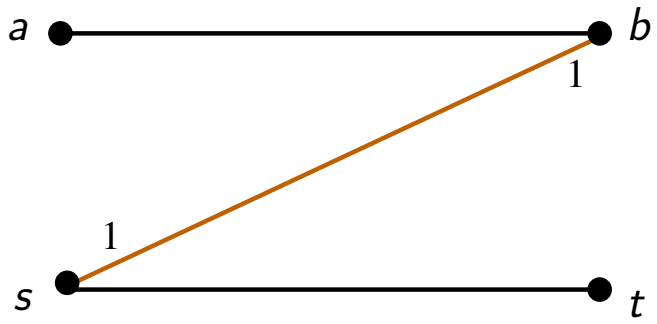
- $a - b - s - t$ is an augmenting path wrt $M \implies$ either ab or st blocks M'
  (a contradiction to M''s stability in G')

Hence any augmenting path in $M \oplus M_{\max}$ has length ≥ 5.

- Thus $|M| \geq \frac{2}{3} \cdot |M_{\max}|$.

Lower bound on $|M|$

Claim. There is no length 3 augmenting path wrt M in G.



- $a - b - s - t$ is an augmenting path wrt $M \implies$ either ab or st blocks M'
  (a contradiction to M''s stability in G')

Hence any augmenting path in $M \oplus M_{\max}$ has length ≥ 5.

- Thus $|M| \geq \frac{2}{3} \cdot |M_{\max}|$.

Thank you!