Popular Matchings under Lower Quotas and their relationship to Stability under Classifications

Ankita Sarkar

Chennai Mathematical Institute

2020

Contents

Stable Marriage Problem

Input :

- A bipartite graph, $G = (\mathcal{M} \uplus \mathcal{W}, E)$, representing men and women, with edges between men and women who are potential partners in marriage.

Stable Marriage Problem

Input :

- A bipartite graph, $G = (\mathcal{M} \uplus \mathcal{W}, E)$, representing men and women, with edges between men and women who are potential partners in marriage.
- For each vertex v, a preference list $(E(v), \leq_v)$, i.e. an ordering of preference over potential partners.

Stable Marriage Problem

Input :

- A bipartite graph, $G = (\mathcal{M} \uplus \mathcal{W}, E)$, representing men and women, with edges between men and women who are potential partners in marriage.
- For each vertex v, a preference list $(E(v), \leq_v)$, i.e. an ordering of preference over potential partners.

Task : Pick a matching, representing couples, such that everyone is "happy".

Stable Marriage Problem

Input :

- A bipartite graph, $G = (\mathcal{M} \uplus \mathcal{W}, E)$, representing men and women, with edges between men and women who are potential partners in marriage.
- For each vertex v, a preference list $(E(v), \leq_v)$, i.e. an ordering of preference over potential partners.

Task : Pick a matching, representing couples, such that everyone is "happy".

Assumption : Strict preference lists, i.e. a vertex v has preference list of the form $(E(v), <_v)$.

Defining “Happiness”

Defining "Happiness"

- no pair has incentive to deviate from the assigned matching.

Blocking Pairs and Stability

Definition

(m, w) is a blocking pair with respect to a matching $M \subseteq E$ if the following conditions hold :

- $(m, w) \in E \setminus M$
- either m is unmatched or $w >_m M(m)$
- either w is unmatched or $m >_w M(w)$

Blocking Pairs and Stability

Definition

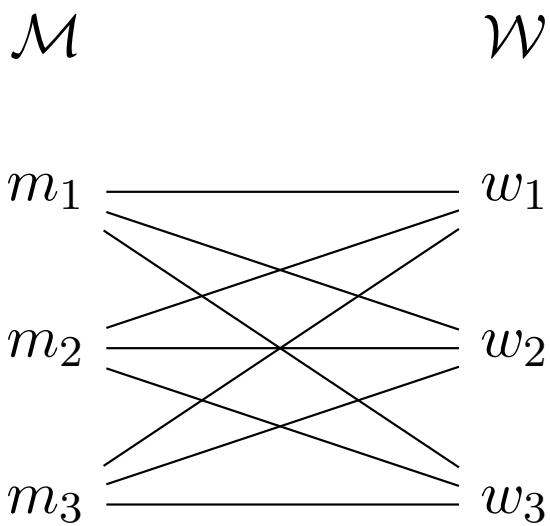
(m, w) is a blocking pair with respect to a matching $M \subseteq E$ if the following conditions hold :

- $(m, w) \in E \setminus M$
- either m is unmatched or $w >_m M(m)$
- either w is unmatched or $m >_w M(w)$

Definition

A matching M is said to be stable if there are no blocking pairs with respect to M.

Example



$m_1 : w_1\ w_2\ w_3$ $\quad$ $w_1 : m_2\ m_3\ m_1$

$m_2 : w_2\ w_3\ w_1$ $\quad$ $w_2 : m_3\ m_1\ m_2$

$m_3 : w_3\ w_1\ w_2$ $\quad$ $w_3 : m_1\ m_2\ m_3$

Example

$$m_1 : w_1\ w_2\ w_3 \qquad w_1 : m_2\ m_3\ m_1$$
$$m_2 : w_2\ w_3\ w_1 \qquad w_2 : m_3\ m_1\ m_2$$
$$m_3 : w_3\ w_1\ w_2 \qquad w_3 : m_1\ m_2\ m_3$$

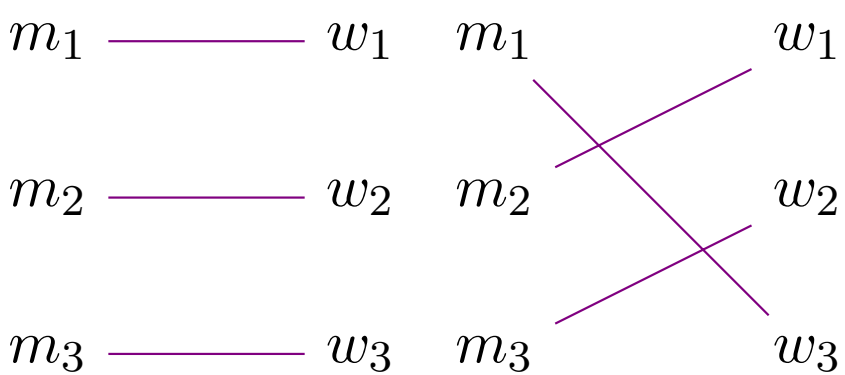
m_1 ——— w_1

m_2 ——— w_2

m_3 ——— w_3

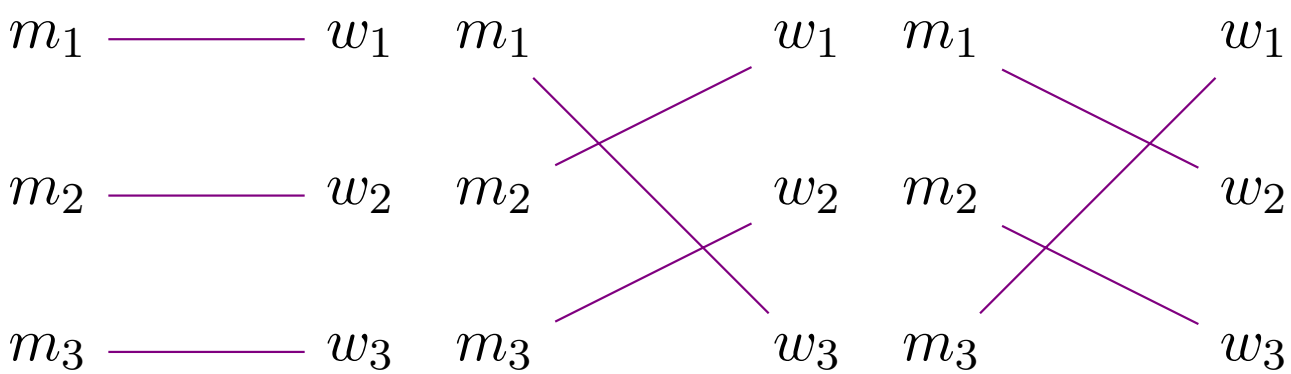
Example

$$m_1 : w_1\ w_2\ w_3 \qquad w_1 : m_2\ m_3\ m_1$$
$$m_2 : w_2\ w_3\ w_1 \qquad w_2 : m_3\ m_1\ m_2$$
$$m_3 : w_3\ w_1\ w_2 \qquad w_3 : m_1\ m_2\ m_3$$



Example

$$m_1 : w_1\ w_2\ w_3 \qquad w_1 : m_2\ m_3\ m_1$$
$$m_2 : w_2\ w_3\ w_1 \qquad w_2 : m_3\ m_1\ m_2$$
$$m_3 : w_3\ w_1\ w_2 \qquad w_3 : m_1\ m_2\ m_3$$



Finding Stable Matchings

[Gale and Shapley, 1962] : showed existence of stable matchings via an algorithm to find them.

Gale-Shapley Algorithm

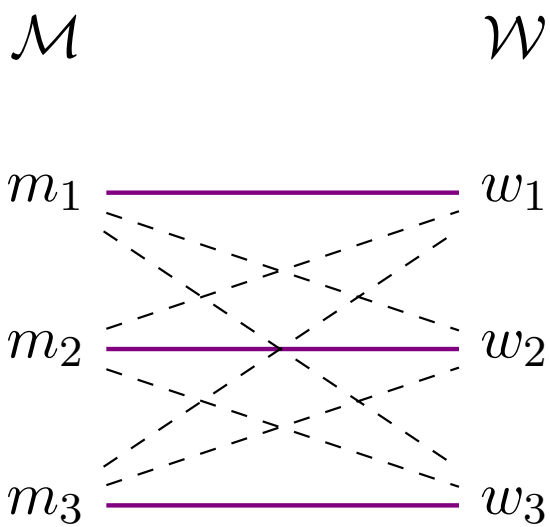
Gale-Shapley Algorithm

- Each man proposes to women, in the order most to least preferred, until he finds a match or exhausts his preference list.

Gale-Shapley Algorithm

- Each man proposes to women, in the order most to least preferred, until he finds a match or exhausts his preference list.
- Each woman provisionally accepts the first proposal she receives. Thereafter, she compares incoming proposals with her existing partner, and rejects the less preferred man.

Examples

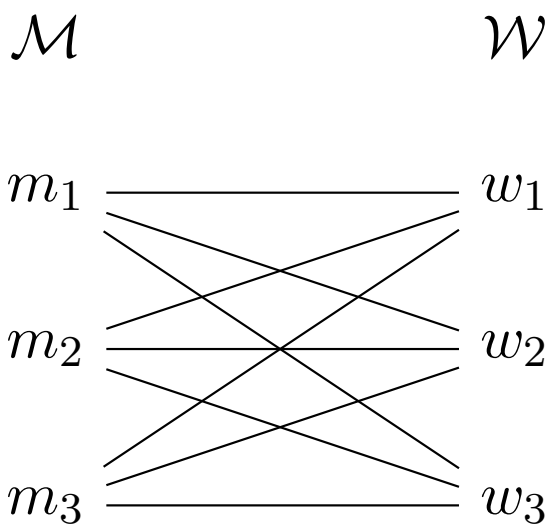


$m_1 : w_1\ w_2\ w_3$ $\quad$ $w_1 : m_2\ m_3\ m_1$

$m_2 : w_2\ w_3\ w_1$ $\quad$ $w_2 : m_3\ m_1\ m_2$

$m_3 : w_3\ w_1\ w_2$ $\quad$ $w_3 : m_1\ m_2\ m_3$

Examples

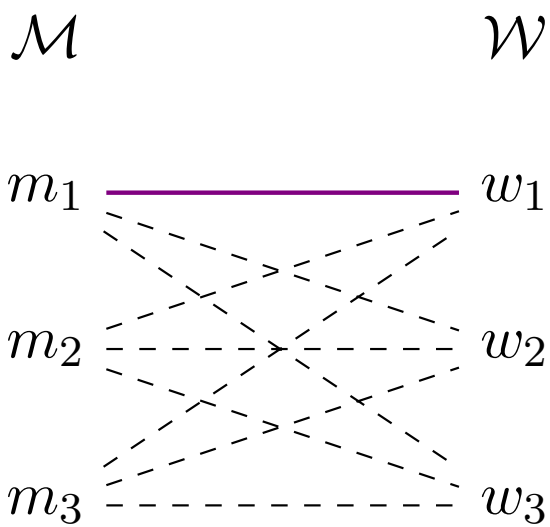


$m_1 : w_1\ w_2\ w_3$ $\quad$ $w_1 : m_2\ m_3\ m_1$

$m_2 : w_1\ w_2\ w_3$ $\quad$ $w_2 : m_3\ m_1\ m_2$

$m_3 : w_1\ w_2\ w_3$ $\quad$ $w_3 : m_1\ m_2\ m_3$

Examples

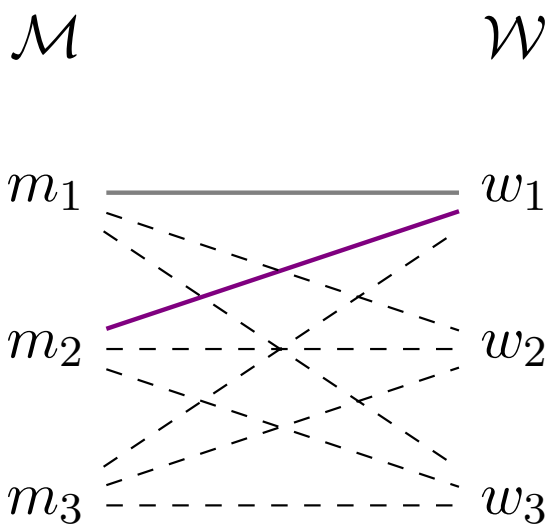


$m_1 : w_1\ w_2\ w_3$ $\quad$ $w_1 : m_2\ m_3\ m_1$

$m_2 : w_1\ w_2\ w_3$ $\quad$ $w_2 : m_3\ m_1\ m_2$

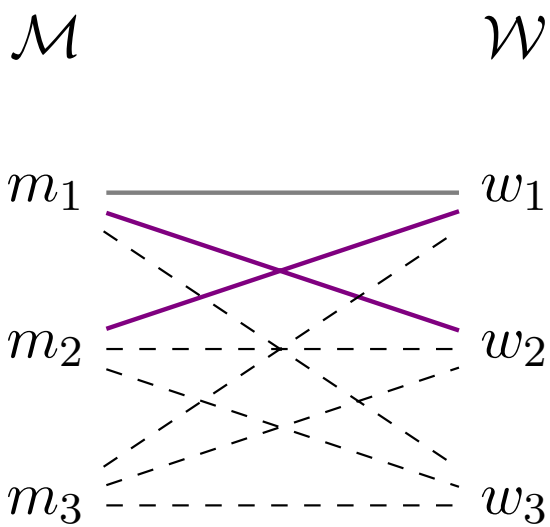
$m_3 : w_1\ w_2\ w_3$ $\quad$ $w_3 : m_1\ m_2\ m_3$

Examples



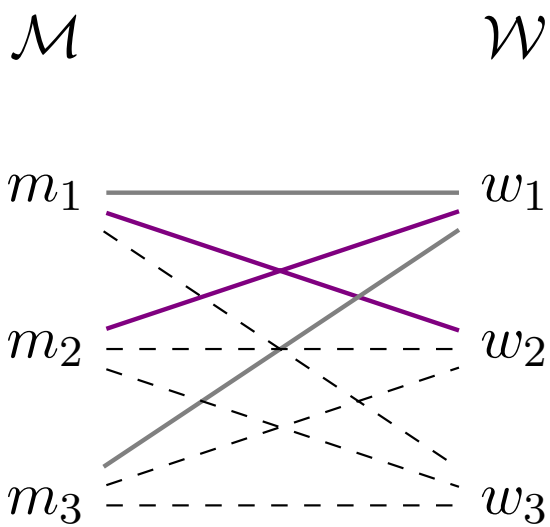
$$m_1 : w_1\ w_2\ w_3 \qquad w_1 : m_2\ m_3\ m_1$$
$$m_2 : w_1\ w_2\ w_3 \qquad w_2 : m_3\ m_1\ m_2$$
$$m_3 : w_1\ w_2\ w_3 \qquad w_3 : m_1\ m_2\ m_3$$

Examples



$$m_1 : w_1\ w_2\ w_3 \qquad w_1 : m_2\ m_3\ m_1$$
$$m_2 : w_1\ w_2\ w_3 \qquad w_2 : m_3\ m_1\ m_2$$
$$m_3 : w_1\ w_2\ w_3 \qquad w_3 : m_1\ m_2\ m_3$$

Examples

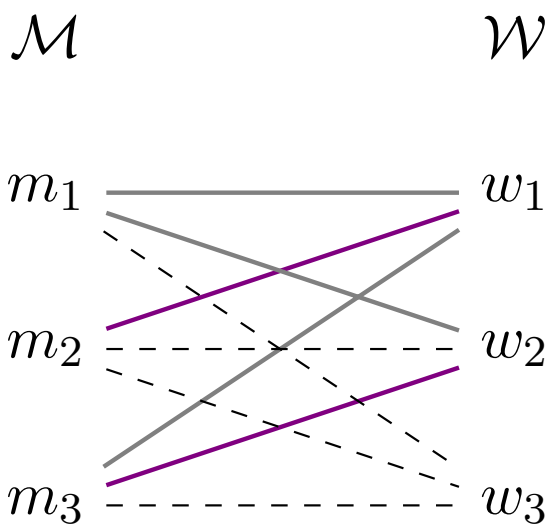


$m_1 : w_1\ w_2\ w_3$ $\quad$ $w_1 : m_2\ m_3\ m_1$

$m_2 : w_1\ w_2\ w_3$ $\quad$ $w_2 : m_3\ m_1\ m_2$

$m_3 : w_1\ w_2\ w_3$ $\quad$ $w_3 : m_1\ m_2\ m_3$

Examples

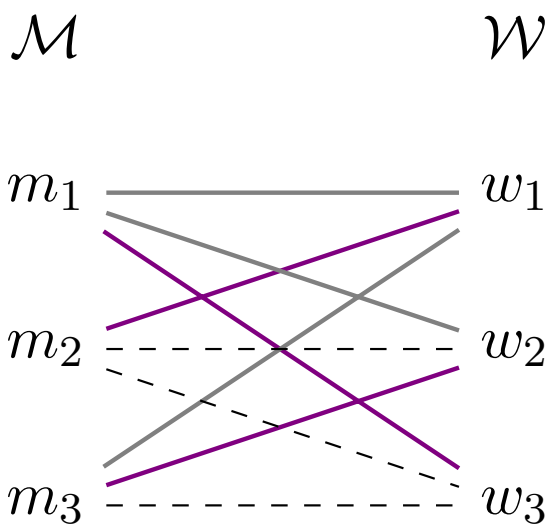


$m_1 : w_1\ w_2\ w_3$ $\quad$ $w_1 : m_2\ m_3\ m_1$

$m_2 : w_1\ w_2\ w_3$ $\quad$ $w_2 : m_3\ m_1\ m_2$

$m_3 : w_1\ w_2\ w_3$ $\quad$ $w_3 : m_1\ m_2\ m_3$

Examples



$m_1 : w_1\ w_2\ w_3$ $\quad$ $w_1 : m_2\ m_3\ m_1$

$m_2 : w_1\ w_2\ w_3$ $\quad$ $w_2 : m_3\ m_1\ m_2$

$m_3 : w_1\ w_2\ w_3$ $\quad$ $w_3 : m_1\ m_2\ m_3$

Contents

Hospital-Residents Problem

Input :

- A bipartite graph, $G = (\mathcal{H} \uplus \mathcal{R}, E)$, representing hospitals and residents, with an edge between hospital-resident pairs that are acceptable to each other.

Hospital-Residents Problem

Input :

- A bipartite graph, $G = (\mathcal{H} \uplus \mathcal{R}, E)$, representing hospitals and residents, with an edge between hospital-resident pairs that are acceptable to each other.
- For each vertex v, a preference list $(E(v), <_v)$, i.e. an ordering of preference over potential partners.

Hospital-Residents Problem

Input :

- A bipartite graph, $G = (\mathcal{H} \uplus \mathcal{R}, E)$, representing hospitals and residents, with an edge between hospital-resident pairs that are acceptable to each other.
- For each vertex v, a preference list $(E(v), <_v)$, i.e. an ordering of preference over potential partners.
- $q^+ : \mathcal{H} \to \mathbb{N}$, upper quotas representing the maximum number of residents a hospital wants to hire.

Hospital-Residents Problem

Input :

- A bipartite graph, $G = (\mathcal{H} \uplus \mathcal{R}, E)$, representing hospitals and residents, with an edge between hospital-resident pairs that are acceptable to each other.
- For each vertex v, a preference list $(E(v), <_v)$, i.e. an ordering of preference over potential partners.
- $q^+ : \mathcal{H} \to \mathbb{N}$, upper quotas representing the maximum number of residents a hospital wants to hire.

Definition

A matching is $M \subseteq E$ such that $\forall h \in \mathcal{H}, |M(h)| \leq q^+(h)$ and $\forall r \in \mathcal{R}, |M(r)| = 1$.

Blocking Pairs and Stability

Definition

$(h, r) \in \mathcal{H} \times \mathcal{R}$ is a blocking pair with respect to a matching $M \subseteq E$ if the following conditions hold :

- $(h, r) \in E \setminus M$
- either h is undersubscribed, i.e. $|M(h)| < q^+(h)$; or $\exists r' \in M(h), r >_h r'$
- either r is unmatched or $h >_r M(r)$

Blocking Pairs and Stability

Definition

$(h, r) \in \mathcal{H} \times \mathcal{R}$ is a blocking pair with respect to a matching $M \subseteq E$ if the following conditions hold :

- $(h, r) \in E \setminus M$
- either h is undersubscribed, i.e. $|M(h)| < q^+(h)$; or $\exists r' \in M(h), r >_h r'$
- either r is unmatched or $h >_r M(r)$

Definition

A matching M is said to be stable if there are no blocking pairs with respect to M.

Gale-Shapley Algorithm

- Hospitals propose down their preference list as long as they have space, or until the list is exhausted.

Gale-Shapley Algorithm

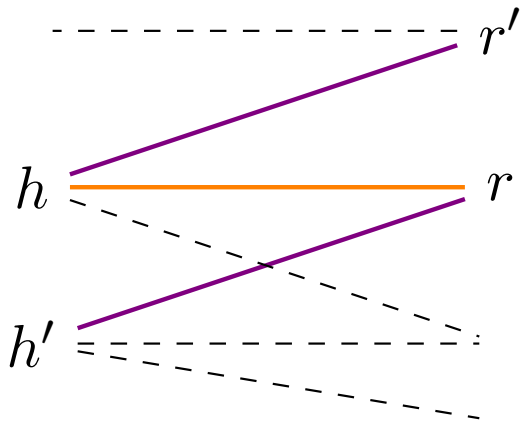
- Hospitals propose down their preference list as long as they have space, or until the list is exhausted.
- Residents accept the first proposal, and compare subsequent proposals with the existing partner, rejecting the less preferred.

Correctness

Proof Sketch : Let the matching obtained via Gale-Shapley algorithm be M.

Correctness

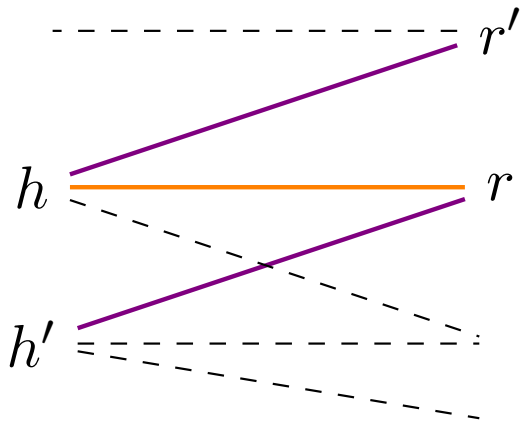
Proof Sketch : Let the matching obtained via Gale-Shapley algorithm be M.



- Suppose (h, r) blocks M. Pick $r' \in M(h), r' <_h r$.

Correctness

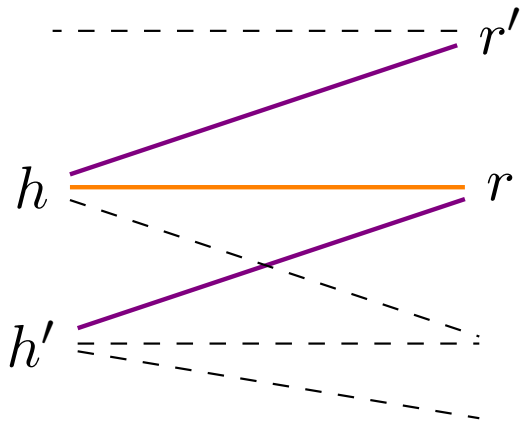
Proof Sketch : Let the matching obtained via Gale-Shapley algorithm be M.



- Suppose (h, r) blocks M. Pick $r' \in M(h), r' <_h r$.
- h must have proposed to r before r'. Thus h was rejected in favour of some $h'', h'' >_r h$.

Correctness

Proof Sketch : Let the matching obtained via Gale-Shapley algorithm be M.



- Suppose (h, r) blocks M. Pick $r' \in M(h), r' <_h r$.
- h must have proposed to r before r'. Thus h was rejected in favour of some $h'', h'' >_r h$.
- But partners of residents only improve. So $h' \geq_r h'' >_r h$, which is a contradiction.

Complexity

Each hospital proposes at most once to each of its neighbour residents. Hence, the Gale-Shapley algorithm runs in $\mathcal{O}(|E|)$ time.

Contents

Popular Matchings

Motivation : Stable matchings are too small in size.

Popular Matchings

Motivation : Stable matchings are too small in size.

Theorem (Rural Hospitals Theorem, [Roth, 1986])

- *All stable matchings have the same size.*
- *Undersubscribed hospitals are matched to the same set of residents in every stable matching.*

Popular Matchings

Motivation : Stable matchings are too small in size.

Theorem (Rural Hospitals Theorem, [Roth, 1986])

- *All stable matchings have the same size.*
- *Undersubscribed hospitals are matched to the same set of residents in every stable matching.*

This means that if the Gale-Shapley output is small, all stable matchings are small. They can in fact be as small as $1/2$ the size of a maximum matching.

Example : Small stable matching

$m_1 : w_1 \ w_2$ $\qquad$ $w_1 : m_1 \ m_2$

$m_2 : w_1$ $\qquad$ $w_2 : m_1$

The Gale-Shapley output is $\{(m_1, w_1)\}$ while the maximum matching is of size 2.

Defining “Happiness”

Defining "Happiness"

- **Stability :** no pair has incentive to deviate from the assigned matching.

Defining "Happiness"

- **Stability :** no pair has incentive to deviate from the assigned matching.
- **Popularity :** in a comparison with any other matching, a majority of vertices prefer this matching.

Voting between Matchings

Definition

For a vertex v, its vote is defined as follows :
$vote_v : E(v) \times E(v) \to \{-1, 1\}$, where

$$vote_v(u, w) = \begin{cases} 1 & u >_v w \\ -1 & u <_v w \\ 0 & u = w \end{cases}$$

Voting between Matchings

We want to have an election between matchings where each vertex votes. But how do hospitals compare their multiple neighbours?

Voting between Matchings

We want to have an election between matchings where each vertex votes. But how do hospitals compare their multiple neighbours?

Definition

For a vertex v, wlog, $|M(v) \setminus M'(v)| \leq |M'(v) \setminus M(v)|$. Fix an arbitrary bijection between $M(v) \setminus M'(v)$ and an arbitrary subset of the same size in $M'(v) \setminus M(v)$. This bijection is $\mathbf{corr}_v|_{M(v) \setminus M'(v)}$ and for any remaining vertex u in $M'(v) \setminus M(v)$, $\mathbf{corr}_v(u) = \perp$.

Voting between Matchings

Once we fix an arbitrary correspondence as above for each hospital, we can calculate each vertex's vote between two matchings :

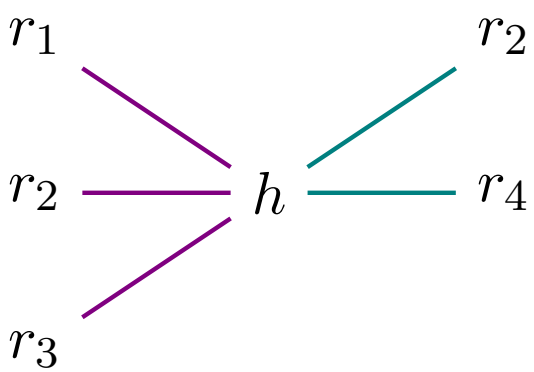
Definition

$$vote_v(M, M') = \sum_{u \in M(v) \setminus M'(v)} vote_v(u, corr_v(u))$$

Here we assume that $vote_v(u, \perp)$ is always 1 for any u.

Example

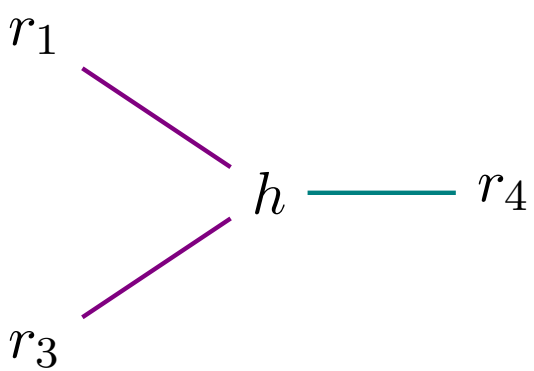
$M_1(h)$ $M_2(h)$



$h : r_1\ r_2\ r_4\ r_3$

Example

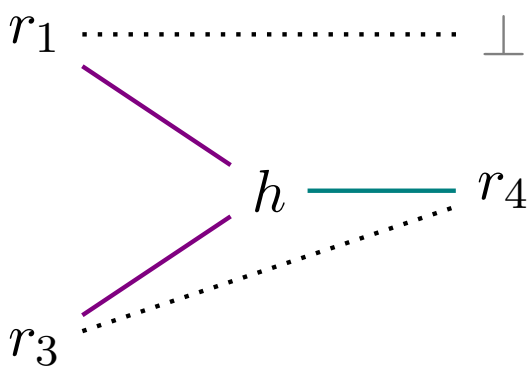
$M_1(h) \setminus M_2(h)$ $\quad$ $M_2(h) \setminus M_1(h)$



$h : r_1\ r_2\ r_4\ r_3$

Example

$M_1(h) \setminus M_2(h)$ $\quad$ $M_2(h) \setminus M_1(h)$

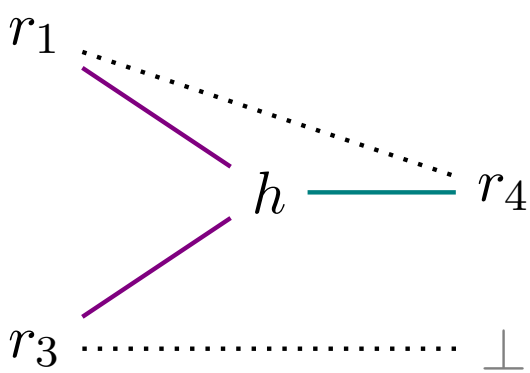


$h : r_1\ r_2\ r_4\ r_3$

$$vote_h(M_1, M_2) = 1 - 1 = 0$$

Example

$M_1(h) \setminus M_2(h)$ $\quad$ $M_2(h) \setminus M_1(h)$



$h : r_1\ r_2\ r_4\ r_3$

$$vote_h(M_1, M_2) = 1 + 1 = 2$$

Popularity

Now we can formalise popularity :

Definition

M' is more popular than M (denoted $M' \succ M$) if

$$\sum_{v \in \mathcal{H} \uplus \mathcal{R}} vote_v(M', M) > 0$$

M is popular among a set of matchings $\mathcal{M} \subseteq 2^E$ if no matching in $\mathcal{M}$ is more popular than M.

Finding Popular Matchings

Finding Popular Matchings

- Stable matchings are popular.

Finding Popular Matchings

- Stable matchings are popular.
- Recall : motivation for popularity is that stable matchings are small.

Finding Popular Matchings

- Stable matchings are popular.
- Recall : motivation for popularity is that stable matchings are small.
- **Task :** Find maximum cardinality popular matchings.

Modifying Gale-Shapley

Algorithm in [Huang and Kavitha, 2011].

Modifying Gale-Shapley

Algorithm in [Huang and Kavitha, 2011].

Idea :

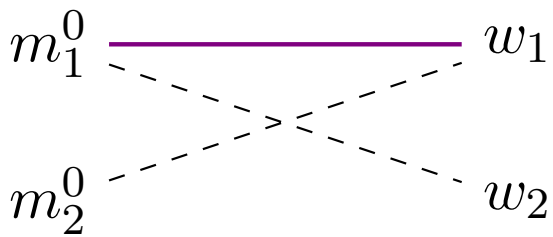
- Usual round of proposals as per Gale-Shapley.
- Undersubscribed hospitals propose again with higher priority, where residents always prefer higher priority hospitals.

Example

In the earlier example:

$m_1 : w_1\ w_2$ $\qquad$ $w_1 : m_1\ m_2$

$m_2 : w_1$ $\qquad$ $w_2 : m_1$



Example

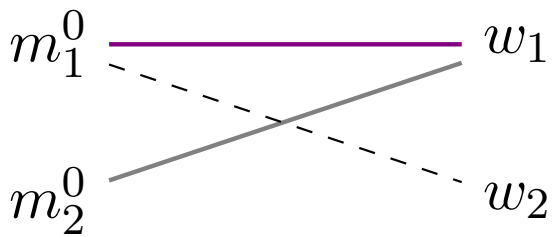
In the earlier example:

$m_1 : w_1\ w_2$

$m_2 : w_1$

$w_1 : m_1\ m_2$

$w_2 : m_1$



Example

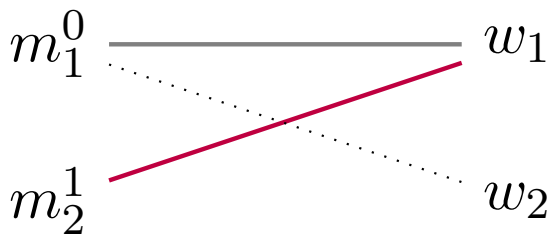
In the earlier example:

$m_1 : w_1\ w_2$

$m_2 : w_1$

$w_1 : m_1\ m_2$

$w_2 : m_1$



Example

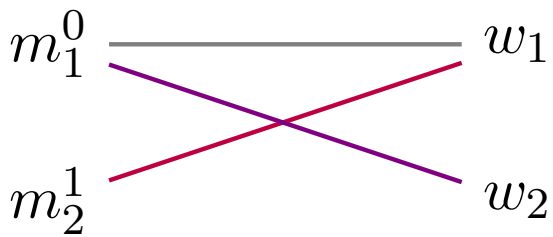
In the earlier example:

$m_1 : w_1\ w_2$

$m_2 : w_1$

$w_1 : m_1\ m_2$

$w_2 : m_1$



Correctness

Intuition : Stable matching is already popular. To increase size without losing popularity, we balance losing and gaining votes in second round.

Correctness

Intuition : Stable matching is already popular. To increase size without losing popularity, we balance losing and gaining votes in second round.

Observation :

- Votes lost : men matched at priority 0, women matched with priority-1 men.

Correctness

Intuition : Stable matching is already popular. To increase size without losing popularity, we balance losing and gaining votes in second round.

Observation :

- Votes lost : men matched at priority 0, women matched with priority-1 men.
- Votes gained : men matched at priority 1, women matched with priority-0 men.

Correctness

Proof strategy :

- Our matching be M, other matching be N. Consider alternating paths and cycles (according to $\mathbf{corr}$) in $M \oplus N$.

Correctness

Proof strategy :

- Our matching be M, other matching be N. Consider alternating paths and cycles (according to $\mathbf{corr}$) in $M \oplus N$.
- Label N-edges in the above paths and cycles as per the vote of each vertex v in $vote_v(N, M)$.

Correctness

Proof strategy :

- Our matching be M, other matching be N. Consider alternating paths and cycles (according to $\mathbf{corr}$) in $M \oplus N$.
- Label N-edges in the above paths and cycles as per the vote of each vertex v in $vote_v(N, M)$.
- Opposing votes along an edge cancel; we show that there are at most as many $(+1, +1)$ edges as $(-1, -1)$ edges.

Correctness

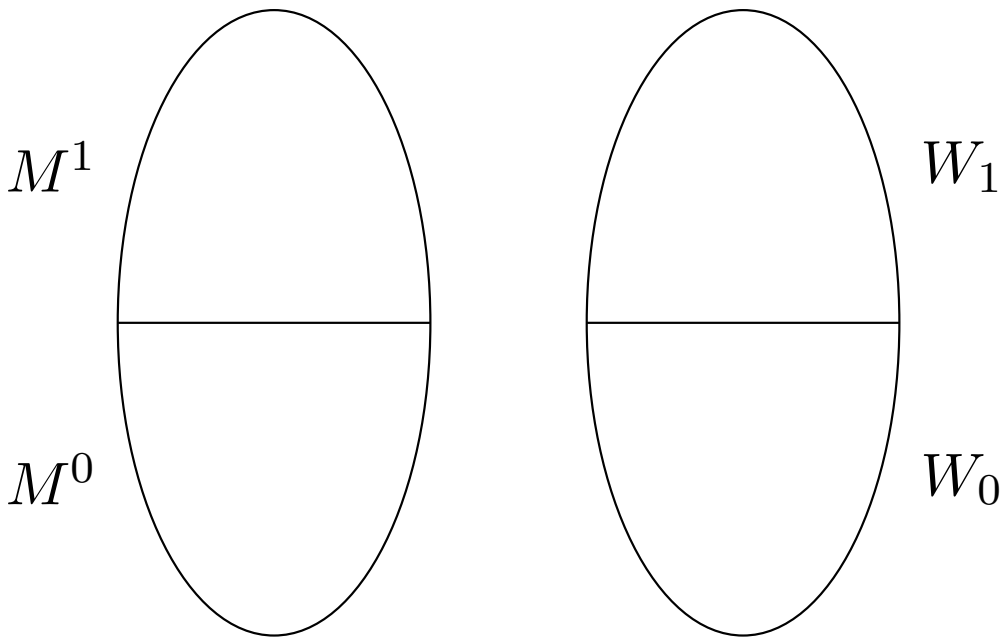
Proof sketch :

- M^0 : Men matched at priority 0; M^1 : Men matched at priority 1
- W_0 : Unmatched women and women matched to priority-0 men; W_1 : women matched to priority-1 men.

Correctness

Proof sketch :

- M^0 : Men matched at priority 0; M^1 : Men matched at priority 1
- W_0 : Unmatched women and women matched to priority-0 men; W_1 : women matched to priority-1 men.



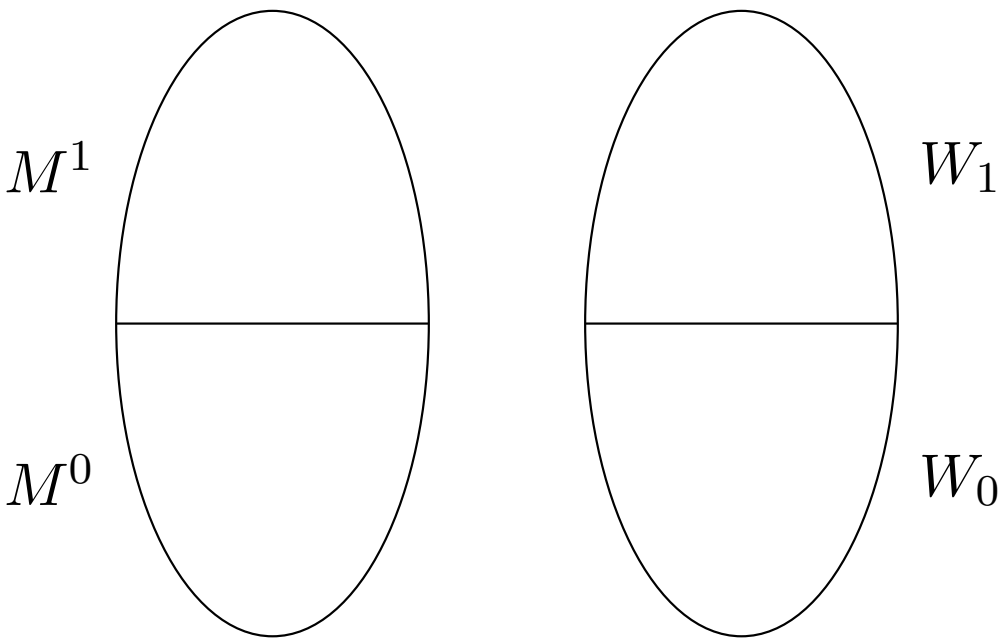
Correctness

Proof sketch :

- M^0 : Men matched at priority 0; M^1 : Men matched at priority 1
- W_0 : Unmatched women and women matched to priority-0 men; W_1 : women matched to priority-1 men.

M edge

N edge



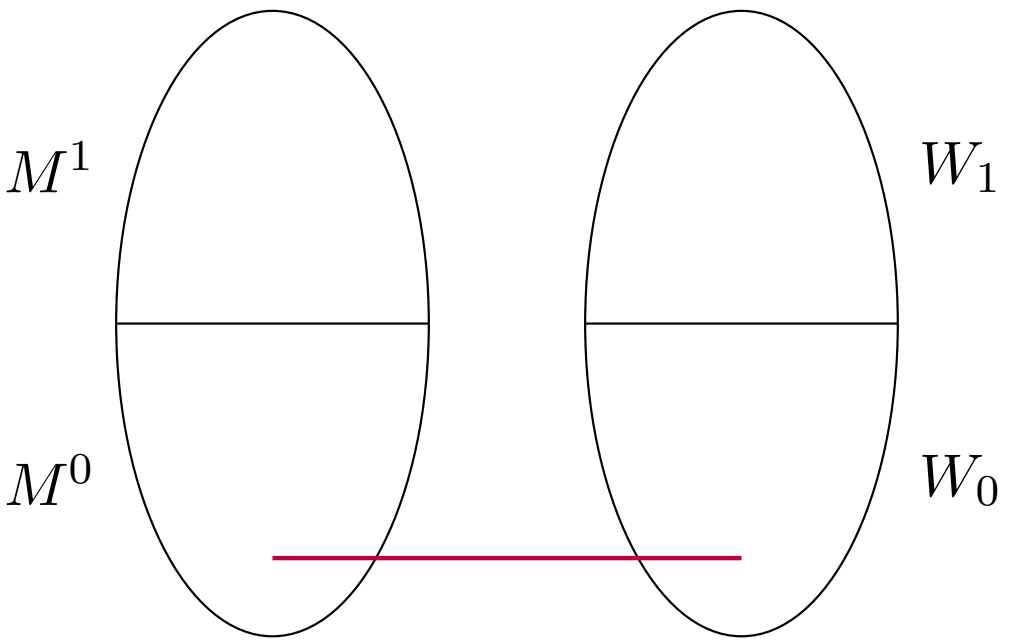
Correctness

Proof sketch :

- M^0 : Men matched at priority 0; M^1 : Men matched at priority 1
- W_0 : Unmatched women and women matched to priority-0 men; W_1 : women matched to priority-1 men.

M edge

N edge



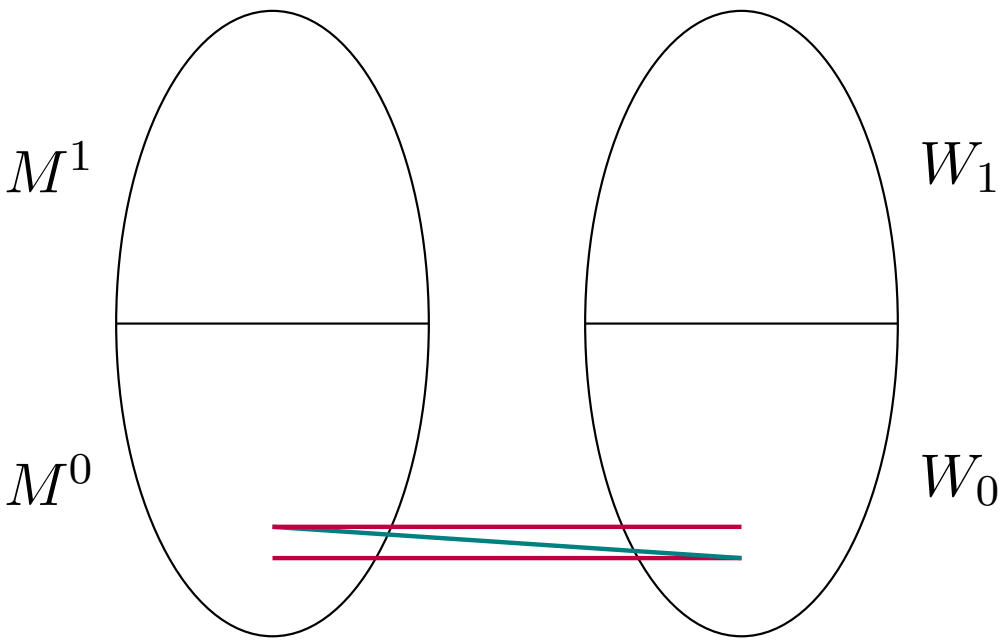
Correctness

Proof sketch :

- M^0 : Men matched at priority 0; M^1 : Men matched at priority 1
- W_0 : Unmatched women and women matched to priority-0 men; W_1 : women matched to priority-1 men.

M edge

N edge



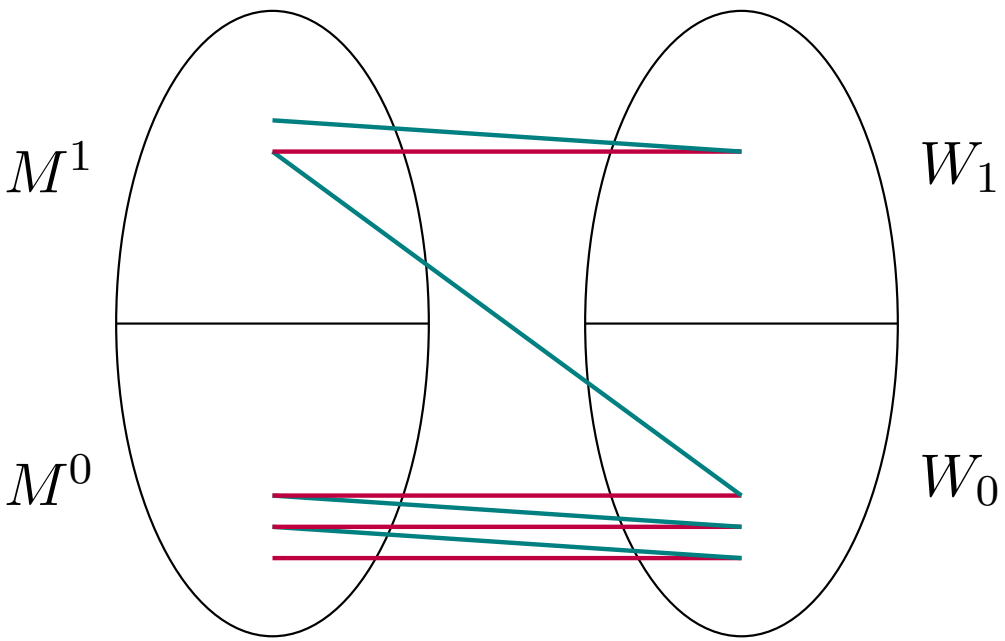
Correctness

Proof sketch :

- M^0 : Men matched at priority 0; M^1 : Men matched at priority 1
- W_0 : Unmatched women and women matched to priority-0 men; W_1 : women matched to priority-1 men.

M edge

N edge



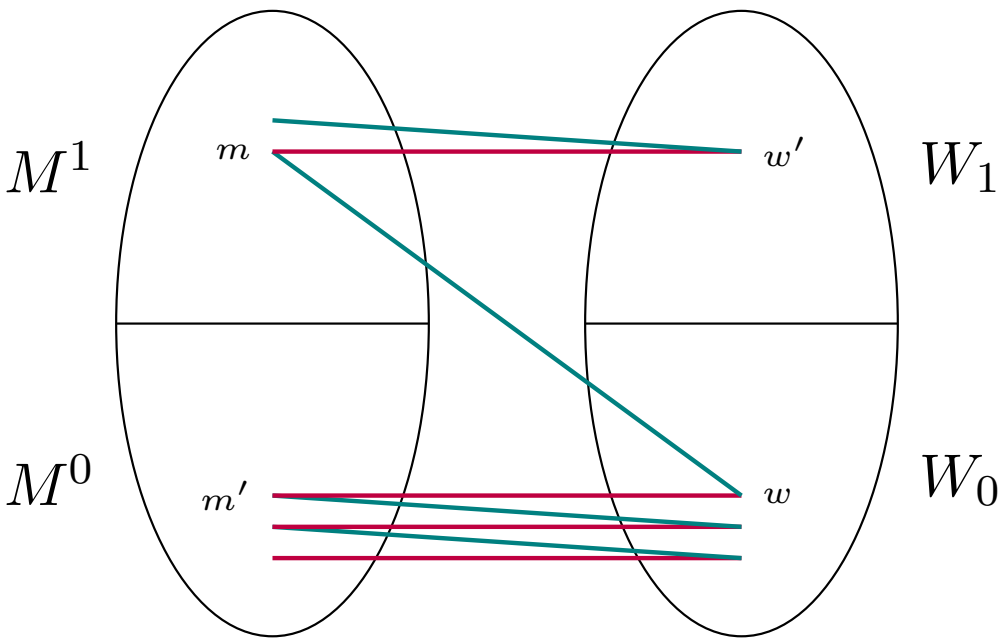
Correctness

Proof sketch :

- M^0 : Men matched at priority 0; M^1 : Men matched at priority 1
- W_0 : Unmatched women and women matched to priority-0 men; W_1 : women matched to priority-1 men.

M edge

N edge



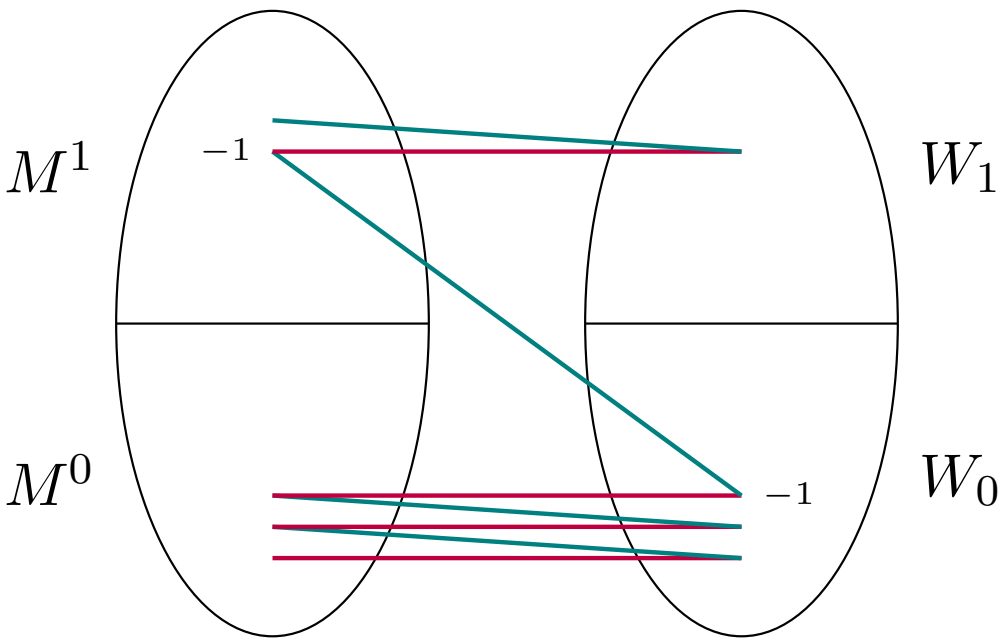
Correctness

Proof sketch :

- M^0 : Men matched at priority 0; M^1 : Men matched at priority 1
- W_0 : Unmatched women and women matched to priority-0 men; W_1 : women matched to priority-1 men.

M edge

N edge



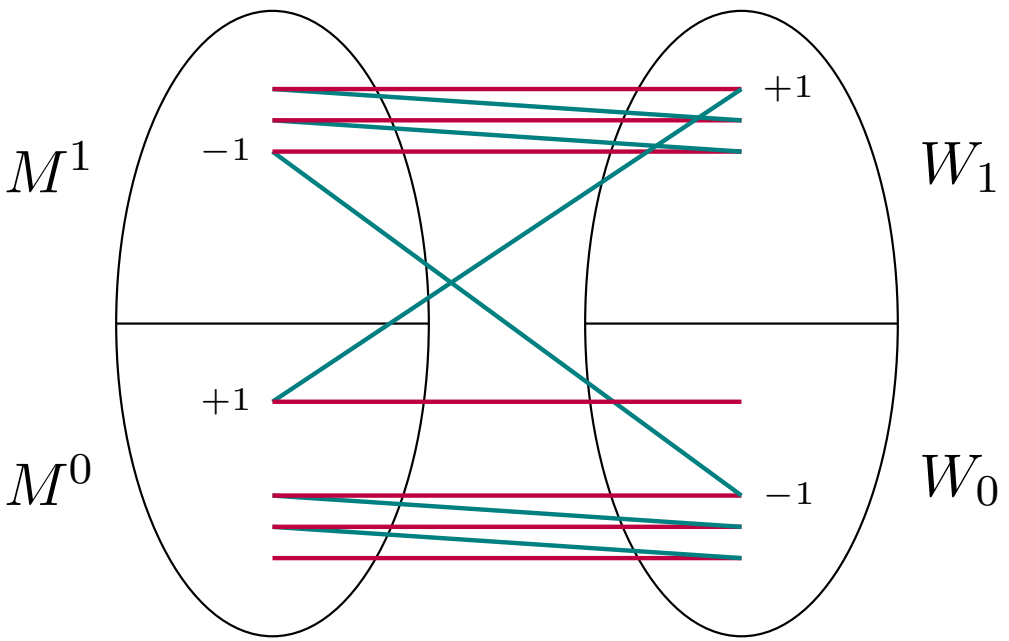
Correctness

Proof sketch :

- M^0 : Men matched at priority 0; M^1 : Men matched at priority 1
- W_0 : Unmatched women and women matched to priority-0 men; W_1 : women matched to priority-1 men.

M edge

N edge



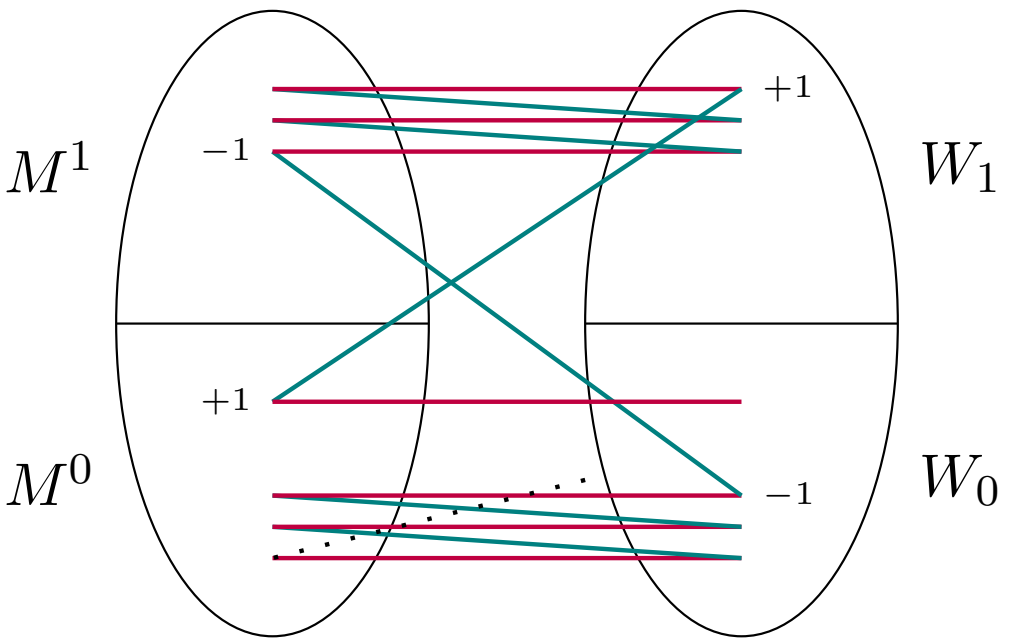
Correctness

Proof sketch :

- M^0 : Men matched at priority 0; M^1 : Men matched at priority 1
- W_0 : Unmatched women and women matched to priority-0 men; W_1 : women matched to priority-1 men.

M edge

N edge



Reduction to Stability

Construction : we make two copies of each hospital h, and add dummy residents $d_h^1, \ldots, d_h^{q^+(h)}$.

Original instance	New instance
$h : r_1 \ \ldots \ r_k$	$h^0 : r_1 \ \ldots, r_k \ d_h^1 \ \ldots \ d_h^{q^+(h)}$ $h^1 : d_h^1 \ \ldots \ d_h^{q^+(h)} \ r_1 \ \ldots, r_k$
$r : h_1 \ \ldots \ h_k$	$r : h_1^1 \ \ldots \ h_k^1 \ h_1^0 \ \ldots \ h_k^0$ $d_h^i : h^0 \ h^1$

Example

Transformed instance :

$$m_1^0 : w_1\ w_2\ d_{m_1}^1 \qquad w_1 : m_1^1\ m_2^1\ m_1^0\ m_2^0$$
$$m_1^1 : d_{m_1}^1\ w_1\ w_2 \qquad w_2 : m_1^1\ m_1^0$$
$$m_2^0 : w_1\ d_{m_2}^1 \qquad d_{m_1}^1 : m_1^0\ m_1^1$$
$$m_2^1 : d_{m_2}^1\ w_1 \qquad d_{m_2}^1 : m_2^0\ m_2^1$$

Example

Transformed instance :

$$
\begin{array}{ll}
m_1^0 : w_1\ w_2\ d_{m_1}^1 & w_1 : m_1^1\ m_2^1\ m_1^0\ m_2^0 \\
m_1^1 : d_{m_1}^1\ w_1\ w_2 & w_2 : m_1^1\ m_1^0 \\
m_2^0 : w_1\ d_{m_2}^1 & d_{m_1}^1 : m_1^0\ m_1^1 \\
m_2^1 : d_{m_2}^1\ w_1 & d_{m_2}^1 : m_2^0\ m_2^1
\end{array}
$$

Gale-Shapley output is $\{(m_1^0, w_2), (m_1^1, d_{m_1}^1), (m_2^0, d_{m_2}^1), (m_2^1, w_1)\}$.
This gives the final matching $\{(m_1, w_2), (m_2, w_1)\}$.

Contents

Hospital-Residents under Lower Quotas

Input :

- Usual hospital-residents instance : $G = (\mathcal{H} \uplus \mathcal{R}, E)$; $\forall v, (E(v), <_v)$; $q^+ : \mathcal{H} \to \mathbb{N}$.
- $q^- : \mathcal{H} \to \mathbb{N}$, representing the minimum number of residents it requires.

Hospital-Residents under Lower Quotas

Input :

- Usual hospital-residents instance : $G = (\mathcal{H} \uplus \mathcal{R}, E)$; $\forall v, (E(v), <_v)$; $q^+ : \mathcal{H} \to \mathbb{N}$.
- $q^- : \mathcal{H} \to \mathbb{N}$, representing the minimum number of residents it requires.

Task : Find a matching that respects lower quotas (is feasible), and also maximises "happiness".

Hospital-Residents under Lower Quotas

Input :

- Usual hospital-residents instance : $G = (\mathcal{H} \uplus \mathcal{R}, E)$; $\forall v, (E(v), <_v)$; $q^+ : \mathcal{H} \to \mathbb{N}$.
- $q^- : \mathcal{H} \to \mathbb{N}$, representing the minimum number of residents it requires.

Task : Find a matching that respects lower quotas (is feasible), and also maximises "happiness".
Question : Can we find a feasible stable matching?

Hospital-Residents under Lower Quotas

Input :

- Usual hospital-residents instance : $G = (\mathcal{H} \uplus \mathcal{R}, E)$; $\forall v, (E(v), <_v)$; $q^+ : \mathcal{H} \to \mathbb{N}$.
- $q^- : \mathcal{H} \to \mathbb{N}$, representing the minimum number of residents it requires.

Task : Find a matching that respects lower quotas (is feasible), and also maximises "happiness".
Question : Can we find a feasible stable matching?
Answer : Not always.

Example

$$h_1 : r_1 \ r_2 \ r_3 \qquad r_1 : h_1$$
$$h_2 : r_2 \ r_3 \qquad r_2 : h_1 \ h_2$$
$$h_3 : r_3 \qquad r_3 : h_1 \ h_2 \ h_3$$

Hospitals h_1, h_2, h_3 have upper quotas $3, 2, 1$ respectively. All lower quotas are 1.

Example

$$h_1 : r_1\ r_2\ r_3 \qquad r_1 : h_1$$
$$h_2 : r_2\ r_3 \qquad r_2 : h_1\ h_2$$
$$h_3 : r_3 \qquad r_3 : h_1\ h_2\ h_3$$

Hospitals h_1, h_2, h_3 have upper quotas $3, 2, 1$ respectively. All lower quotas are 1.
Gale-Shapley matches all three residents with h_1. This is infeasible. By Rural Hospitals Theorem, then all stable matchings are infeasible.

Popularity among Feasible Matchings

Algorithm by [Nasre and Nimbhorkar, 2017] :

Popularity among Feasible Matchings

Algorithm by [Nasre and Nimbhorkar, 2017] :

- Usual Gale-Shapley proposals (priority 0).
- Undersubscribed hospitals propose again with priority 1.

Popularity among Feasible Matchings

Algorithm by [Nasre and Nimbhorkar, 2017] :

- Usual Gale-Shapley proposals (priority 0).
- Undersubscribed hospitals propose again with priority 1.
- Deficient hospitals, i.e. those with lower quota unfilled, propose sufficiently more times, but with upper quotas capped to lower quotas (priority 2 and above).

Popularity among Feasible Matchings

Algorithm by [Nasre and Nimbhorkar, 2017] :

- Usual Gale-Shapley proposals (priority 0).
- Undersubscribed hospitals propose again with priority 1.
- Deficient hospitals, i.e. those with lower quota unfilled, propose sufficiently more times, but with upper quotas capped to lower quotas (priority 2 and above).

Priorities : $0, \ldots, \mu - 1$, where $\mu = \sum_{h \in \mathcal{H}} q^-(h) + 2$.

Popularity among Feasible Matchings

Algorithm by [Nasre and Nimbhorkar, 2017] :

- Usual Gale-Shapley proposals (priority 0).
- Undersubscribed hospitals propose again with priority 1.
- Deficient hospitals, i.e. those with lower quota unfilled, propose sufficiently more times, but with upper quotas capped to lower quotas (priority 2 and above).

Priorities : $0, \ldots, \mu - 1$, where $\mu = \sum_{h \in \mathcal{H}} q^-(h) + 2$.
Reduction to Stability : similar.

Correctness

If it is feasible, then it is maximum cardinality popular among feasible matchings – similar alternating path argument.

Correctness

If it is feasible, then it is maximum cardinality popular among feasible matchings – similar alternating path argument.
Question : Is M feasible?

Correctness

If it is feasible, then it is maximum cardinality popular among feasible matchings – similar alternating path argument.
Question : Is M feasible?
Answer : If there exists a feasible matching, then M is feasible.

Correctness

Proof sketch :
If there is a deficient hospital, we track an alternating path starting from it, counting hospitals in that path.

Correctness

Proof sketch :
If there is a deficient hospital, we track an alternating path starting from it, counting hospitals in that path.
We find that the length of this path depends on μ, but also hospitals can't appear in this more than $\sum_{h \in \mathcal{H}} q^{-}(h)$ times.

Correctness

Proof sketch :
If there is a deficient hospital, we track an alternating path starting from it, counting hospitals in that path.
We find that the length of this path depends on μ, but also hospitals can't appear in this more than $\sum_{h \in \mathcal{H}} q^{-}(h)$ times.
Thus our choice of μ will give a contradiction.

Complexity

Observation : Feasible instance $\implies \mu - 2 \leq |\mathcal{R}|$. We can assume that trivially infeasible instances that violate this are ruled out before using the algorithm.

Complexity

Observation : Feasible instance $\implies \mu - 2 \leq |\mathcal{R}|$. We can assume that trivially infeasible instances that violate this are ruled out before using the algorithm.
The new edge set is of size $\mu\,|E| = \mathcal{O}(|\mathcal{R}|\,|E|)$.

Example

$$h_1 : r_1\ r_2\ r_3 \qquad\qquad r_1 : h_1$$
$$h_2 : r_2\ r_3 \qquad\qquad r_2 : h_1\ h_2$$
$$h_3 : r_3 \qquad\qquad r_3 : h_1\ h_2\ h_3$$

Hospitals h_1, h_2, h_3 have upper quotas $3, 2, 1$ respectively. All lower quotas are 1.

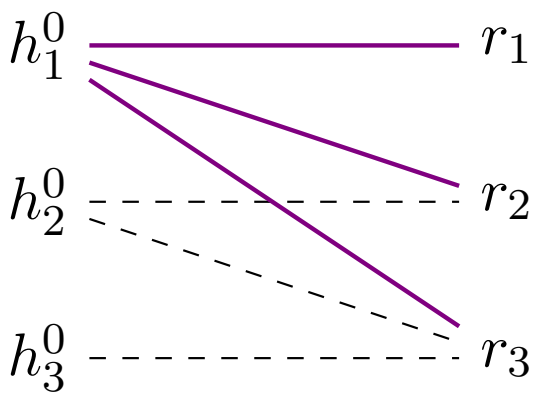
Example

$h_1 : r_1\ r_2\ r_3$ $\qquad$ $r_1 : h_1$

$h_2 : r_2\ r_3$ $\qquad$ $r_2 : h_1\ h_2$

$h_3 : r_3$ $\qquad$ $r_3 : h_1\ h_2\ h_3$

Hospitals h_1, h_2, h_3 have upper quotas $3, 2, 1$ respectively. All lower quotas are 1.



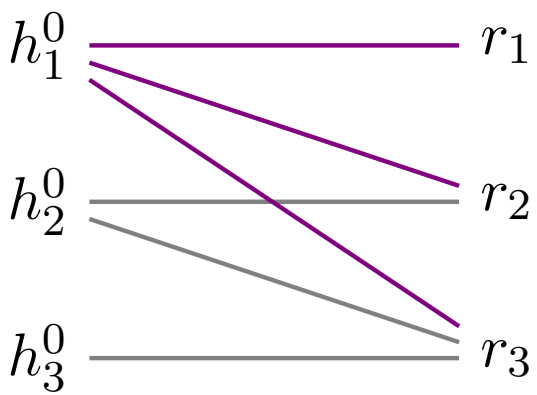
Example

$h_1 : r_1\ r_2\ r_3$ $\qquad$ $r_1 : h_1$

$h_2 : r_2\ r_3$ $\qquad$ $r_2 : h_1\ h_2$

$h_3 : r_3$ $\qquad$ $r_3 : h_1\ h_2\ h_3$

Hospitals h_1, h_2, h_3 have upper quotas $3, 2, 1$ respectively. All lower quotas are 1.



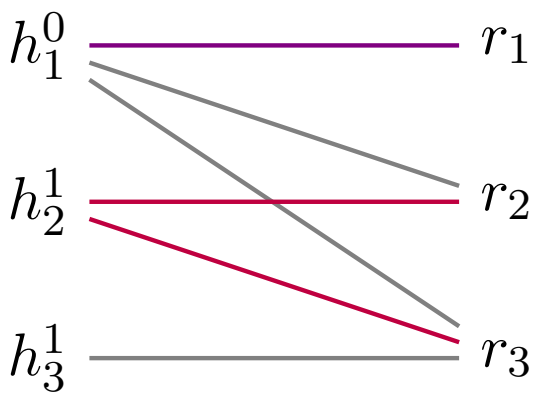
Example

$h_1 : r_1\ r_2\ r_3$ $\qquad$ $r_1 : h_1$

$h_2 : r_2\ r_3$ $\qquad$ $r_2 : h_1\ h_2$

$h_3 : r_3$ $\qquad$ $r_3 : h_1\ h_2\ h_3$

Hospitals h_1, h_2, h_3 have upper quotas $3, 2, 1$ respectively. All lower quotas are 1.



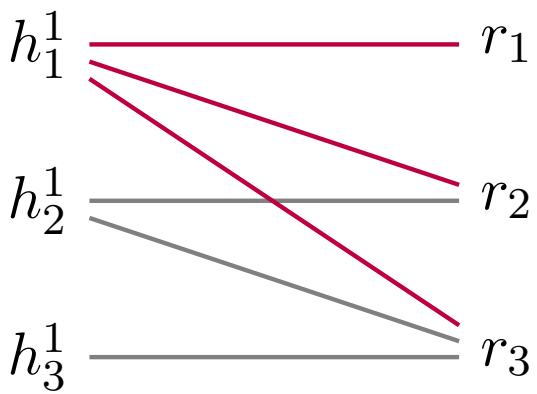
Example

$h_1 : r_1\ r_2\ r_3$ $\qquad$ $r_1 : h_1$

$h_2 : r_2\ r_3$ $\qquad$ $r_2 : h_1\ h_2$

$h_3 : r_3$ $\qquad$ $r_3 : h_1\ h_2\ h_3$

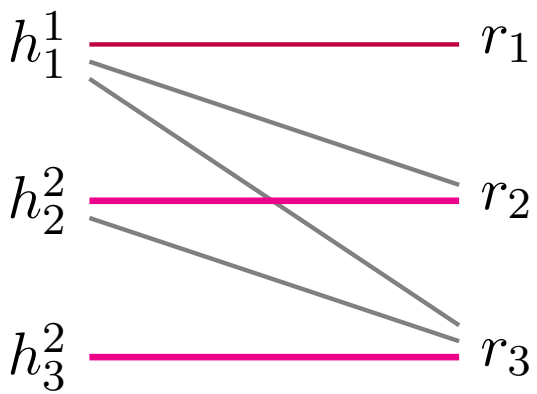
Hospitals h_1, h_2, h_3 have upper quotas $3, 2, 1$ respectively. All lower quotas are 1.



Example

$$h_1 : r_1\ r_2\ r_3 \qquad r_1 : h_1$$
$$h_2 : r_2\ r_3 \qquad r_2 : h_1\ h_2$$
$$h_3 : r_3 \qquad r_3 : h_1\ h_2\ h_3$$

Hospitals h_1, h_2, h_3 have upper quotas $3, 2, 1$ respectively. All lower quotas are 1.



Contents

Both-sided Lower Quotas

- In addition to earlier constraints, some residents must be matched.

Both-sided Lower Quotas

- In addition to earlier constraints, some residents must be matched.
- Formally, we can view residents as having $q^+ = 1$.

Both-sided Lower Quotas

- In addition to earlier constraints, some residents must be matched.
- Formally, we can view residents as having $q^+ = 1$. Now they also have $q^- \in \{0, 1\}$.

Popularity among Feasible Matchings

Strategy :

- Allow repeated proposals as before.

Popularity among Feasible Matchings

Strategy :

- Allow repeated proposals as before.
- Reverse direction of proposals – deficient residents can propose with higher priority.

Popularity among Feasible Matchings

Strategy :

- Allow repeated proposals as before.
- Reverse direction of proposals – deficient residents can propose with higher priority. These proposals do not violate lower quotas of hospitals.

Example

$$m_1 : w_1 \ w_2 \ w_3 \qquad\qquad w_1 : m_1 \ m_2 \ m_3$$
$$m_2 : w_1 \qquad\qquad w_2 : m_1$$
$$m_3 : w_1 \qquad\qquad w_3 : m_1$$

m_3 and w_3 must be matched.

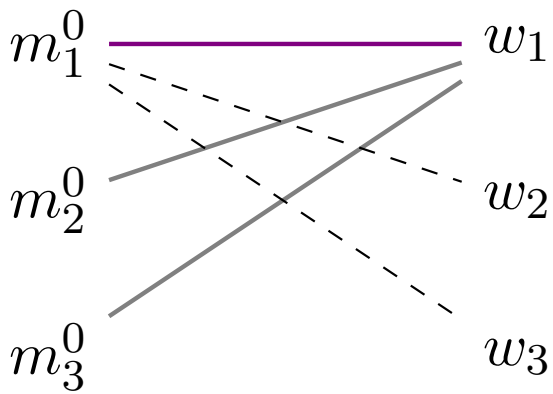
Example

$m_1 : w_1\ w_2\ w_3$ $\qquad$ $w_1 : m_1\ m_2\ m_3$

$m_2 : w_1$ $\qquad$ $w_2 : m_1$

$m_3 : w_1$ $\qquad$ $w_3 : m_1$

m_3 and w_3 must be matched.



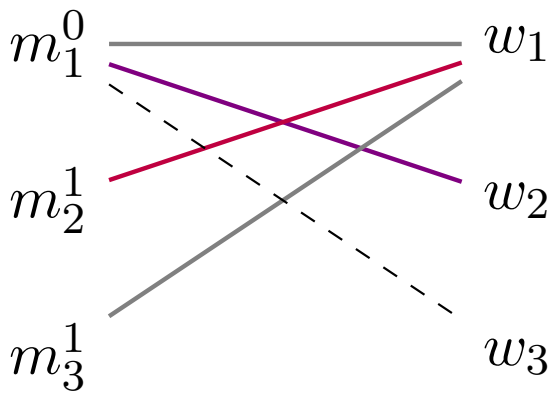
Example

$m_1 : w_1\ w_2\ w_3$ $\qquad$ $w_1 : m_1\ m_2\ m_3$

$m_2 : w_1$ $\qquad$ $w_2 : m_1$

$m_3 : w_1$ $\qquad$ $w_3 : m_1$

m_3 and w_3 must be matched.



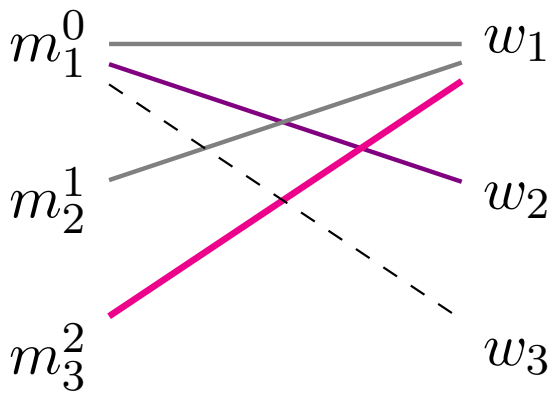
Example

$m_1 : w_1\ w_2\ w_3$ $\qquad$ $w_1 : m_1\ m_2\ m_3$

$m_2 : w_1$ $\qquad$ $w_2 : m_1$

$m_3 : w_1$ $\qquad$ $w_3 : m_1$

m_3 and w_3 must be matched.



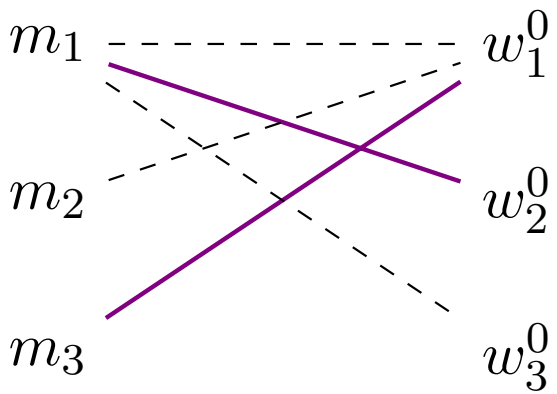
Example

$m_1 : w_1\ w_2\ w_3$ $\qquad$ $w_1 : m_1\ m_2\ m_3$

$m_2 : w_1$ $\qquad$ $w_2 : m_1$

$m_3 : w_1$ $\qquad$ $w_3 : m_1$

m_3 and w_3 must be matched.



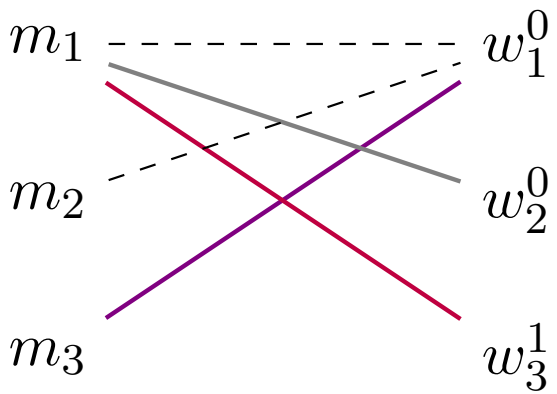
Example

$m_1 : w_1\ w_2\ w_3$ $\qquad$ $w_1 : m_1\ m_2\ m_3$

$m_2 : w_1$ $\qquad$ $w_2 : m_1$

$m_3 : w_1$ $\qquad$ $w_3 : m_1$

m_3 and w_3 must be matched.



Reduction to Stability

- Make copies and add dummies to both sides. Dummy hospitals have upper quota 1.

Reduction to Stability

- Make copies and add dummies to both sides. Dummy hospitals have upper quota 1.
- Hospitals quotas are capped for priorities 2 and above, resident quotas are capped for priorities 1 and above.
- We choose $\mu_H = \sum_{h \in \mathcal{H}} q^-(h) + 2$, $\mu_R = \sum_{r \in \mathcal{R}} q^-(r) + 2$.

Reduction to Stability

- Make copies and add dummies to both sides. Dummy hospitals have upper quota 1.
- Hospitals quotas are capped for priorities 2 and above, resident quotas are capped for priorities 1 and above.
- We choose $\mu_H = \sum_{h \in \mathcal{H}} q^-(h) + 2$, $\mu_R = \sum_{r \in \mathcal{R}} q^-(r) + 2$.

If a vertex v had preference list $\mathbf{pref}(v)$, we give all its copies the concatenation of $\mathbf{pref}(v)^{\mu-1} \ldots \mathbf{pref}(v)^0$ padded with appropriate number of dummies, where $\mu = \mu_H$ if $v \in \mathcal{R}$ and vice-versa.

Correctness

Similar arguments as for one-sided lower quotas almost work.

Correctness

Similar arguments as for one-sided lower quotas almost work.
Hurdle : What if the same hospital-resident pair is matched at different priorities?

Correctness

Similar arguments as for one-sided lower quotas almost work.
Hurdle : What if the same hospital-resident pair is matched at different priorities? Then the new instance overcounts places filled, hence potentially gives infeasible solution.

Correctness

Similar arguments as for one-sided lower quotas almost work.
Hurdle : What if the same hospital-resident pair is matched at different priorities? Then the new instance overcounts places filled, hence potentially gives infeasible solution.
Solution : This does not happen with one-sided lower quotas because a resident matches to a hospital exactly once.

Correctness

Similar arguments as for one-sided lower quotas almost work.
Hurdle : What if the same hospital-resident pair is matched at different priorities? Then the new instance overcounts places filled, hence potentially gives infeasible solution.
Solution : This does not happen with one-sided lower quotas because a resident matches to a hospital exactly once. But here, residents also have copies.

Correctness

Similar arguments as for one-sided lower quotas almost work.
Hurdle : What if the same hospital-resident pair is matched at different priorities? Then the new instance overcounts places filled, hence potentially gives infeasible solution.
Solution : This does not happen with one-sided lower quotas because a resident matches to a hospital exactly once. But here, residents also have copies. But we notice that the matching M we obtain respects upper quotas.

Complexity

The new edge set is of size

$$\begin{aligned}\mu_H \mu_R \left|E\right| &\leq (\left|\mathcal{R}\right| + 2)(\sum_{h \in \mathcal{H}} q^+(h) + 2) \left|E\right| \\ &\leq (\left|\mathcal{R}\right| + 2)^2 \left|E\right| \\ &= \mathcal{O}(\left|\mathcal{R}\right|^2 \left|E\right|)\end{aligned}$$

Contents

The Students-Courses Problem

Many-to-many version of Hospital-Residents.

The Students-Courses Problem

Many-to-many version of Hospital-Residents.

Input :

- $G = (\mathcal{S} \uplus \mathcal{C}, E)$ representing students and courses, with preference lists $(E(v), <_v)$ for each vertex v.
- Upper quotas $q^+ : \mathcal{S} \uplus \mathcal{C} \to \mathbb{N}$

The Students-Courses Problem

Many-to-many version of Hospital-Residents.

Input :

- $G = (\mathcal{S} \uplus \mathcal{C}, E)$ representing students and courses, with preference lists $(E(v), <_v)$ for each vertex v.
- Upper quotas $q^+ : \mathcal{S} \uplus \mathcal{C} \to \mathbb{N}$

Definition

A matching $M \subseteq E$ is such that for each vertex $v \in \mathcal{S} \uplus \mathcal{C}$, $|M(v)| \leq q^+(v)$.

The Students-Courses Problem

Many-to-many version of Hospital-Residents.

Input :

- $G = (\mathcal{S} \uplus \mathcal{C}, E)$ representing students and courses, with preference lists $(E(v), <_v)$ for each vertex v.
- Upper quotas $q^+ : \mathcal{S} \uplus \mathcal{C} \to \mathbb{N}$

Definition

A matching $M \subseteq E$ is such that for each vertex $v \in \mathcal{S} \uplus \mathcal{C}$, $|M(v)| \leq q^+(v)$.

Definition

An edge $(s, c) \in E \setminus M$ blocks M when

- $|M(s)| < q^+(s)$ or $\exists c' \in M(s), c' <_s c$, and
- $|M(c)| < q^+(c)$ or $\exists s' \in M(c), s' <_c s$

Stability and Popularity

- Without lower quotas, Gale-Shapley gives a stable matching in $\mathcal{O}(|E|)$ time.

Stability and Popularity

- Without lower quotas, Gale-Shapley gives a stable matching in $\mathcal{O}(|E|)$ time.
- [Brandl and Kavitha, 2017] give an algorithm for maximum-cardinality popularity, similar to the algorithm in [Huang and Kavitha, 2011].

Students-Courses under Lower Quotas

One side, say students, have lower quotas $q^- : \mathcal{S} \to \mathbb{N}$

- Priorities for students, as done for hospitals in [Nasre and Nimbhorkar, 2017].

Students-Courses under Lower Quotas

One side, say students, have lower quotas $q^- : \mathcal{S} \to \mathbb{N}$

- Priorities for students, as done for hospitals in [Nasre and Nimbhorkar, 2017].
- Same overcounting hurdle arises. Here, we cannot using upper quotas since they are arbitrary on both sides.

Students-Courses under Lower Quotas

One side, say students, have lower quotas $q^- : \mathcal{S} \to \mathbb{N}$

- Priorities for students, as done for hospitals in [Nasre and Nimbhorkar, 2017].
- Same overcounting hurdle arises. Here, we cannot using upper quotas since they are arbitrary on both sides.

Solution : Using classifications.

Students-Courses under Lower Quotas

One side, say students, have lower quotas $q^- : \mathcal{S} \to \mathbb{N}$

- Priorities for students, as done for hospitals in [Nasre and Nimbhorkar, 2017].
- Same overcounting hurdle arises. Here, we cannot using upper quotas since they are arbitrary on both sides.

Solution : Using classifications.
We put all copies of a student vertex in a class, and each course imposes a class upper quota of one on each.

Students-Courses under Lower Quotas

One side, say students, have lower quotas $q^- : \mathcal{S} \to \mathbb{N}$

- Priorities for students, as done for hospitals in [Nasre and Nimbhorkar, 2017].
- Same overcounting hurdle arises. Here, we cannot using upper quotas since they are arbitrary on both sides.

Solution : Using classifications.
We put all copies of a student vertex in a class, and each course imposes a class upper quota of one on each. We seek a stable matching respecting these constraints.

Correctness and Complexity using Classifications

For these simple classifications – just a partition – we can find the required matching in time linear in edges.

Correctness and Complexity using Classifications

For these simple classifications – just a partition – we can find the required matching in time linear in edges.
The remaining proof is similar to earlier.

Correctness and Complexity using Classifications

For these simple classifications – just a partition – we can find the required matching in time linear in edges.
The remaining proof is similar to earlier.
This runs in time

$$\begin{aligned}\mu\,|E| &= (\sum_{s\in\mathcal{S}} q^-(s) + 2)\,|E| \\ &\leq (\sum_{c\in\mathcal{C}} q^+(c) + 2)\,|E| \\ &\leq (\sum_{c\in\mathcal{C}} |E(c)| + 2)\,|E| \\ &= \mathcal{O}(|E|^2)\end{aligned}$$

Contents

Classifications in Hospital-Residents

Input :

- A hospital-residents instance with lower quotas.

Classifications in Hospital-Residents

Input :

- A hospital-residents instance with lower quotas.
- Each hospital h has classes over its neighbour set, $\mathscr{C}_h \subseteq 2^{E(h)}$, and class quotas $q_h^+, q_h^- : \mathscr{C}_h \to \mathbb{N}$

Classifications in Hospital-Residents

Input :

- A hospital-residents instance with lower quotas.
- Each hospital h has classes over its neighbour set, $\mathscr{C}_h \subseteq 2^{E(h)}$, and class quotas $q_h^+, q_h^- : \mathscr{C}_h \to \mathbb{N}$

This problem is studied in [Huang, 2010].

Classifications in Hospital-Residents

Input :

- A hospital-residents instance with lower quotas.
- Each hospital h has classes over its neighbour set, $\mathscr{C}_h \subseteq 2^{E(h)}$, and class quotas $q_h^+, q_h^- : \mathscr{C}_h \to \mathbb{N}$

This problem is studied in [Huang, 2010]. What we require for our earlier algorithm is a many-to-many version, but for a very special case :

Classifications in Hospital-Residents

Input :

- A hospital-residents instance with lower quotas.
- Each hospital h has classes over its neighbour set, $\mathscr{C}_h \subseteq 2^{E(h)}$, and class quotas $q_h^+, q_h^- : \mathscr{C}_h \to \mathbb{N}$

This problem is studied in [Huang, 2010]. What we require for our earlier algorithm is a many-to-many version, but for a very special case :

- There are no lower quotas, because we apply this to the transformed instance.

Classifications in Hospital-Residents

Input :

- A hospital-residents instance with lower quotas.
- Each hospital h has classes over its neighbour set, $\mathscr{C}_h \subseteq 2^{E(h)}$, and class quotas $q_h^+, q_h^- : \mathscr{C}_h \to \mathbb{N}$

This problem is studied in [Huang, 2010]. What we require for our earlier algorithm is a many-to-many version, but for a very special case :

- There are no lower quotas, because we apply this to the transformed instance.
- In [Huang, 2010], laminar classes are studied, which is more general than just partitions.

Classifications in Hospital-Residents

Input :

- A hospital-residents instance with lower quotas.
- Each hospital h has classes over its neighbour set, $\mathscr{C}_h \subseteq 2^{E(h)}$, and class quotas $q_h^+, q_h^- : \mathscr{C}_h \to \mathbb{N}$

This problem is studied in [Huang, 2010]. What we require for our earlier algorithm is a many-to-many version, but for a very special case :

- There are no lower quotas, because we apply this to the transformed instance.
- In [Huang, 2010], laminar classes are studied, which is more general than just partitions.
- All our class upper quotas are 1.

Classifications in Hospital-Residents

Input :

- A hospital-residents instance with lower quotas.
- Each hospital h has classes over its neighbour set, $\mathscr{C}_h \subseteq 2^{E(h)}$, and class quotas $q_h^+, q_h^- : \mathscr{C}_h \to \mathbb{N}$

This problem is studied in [Huang, 2010]. What we require for our earlier algorithm is a many-to-many version, but for a very special case :

- There are no lower quotas, because we apply this to the transformed instance.
- In [Huang, 2010], laminar classes are studied, which is more general than just partitions.
- All our class upper quotas are 1.
- All courses impose the same classes.

We utilise the first two properties to simplify Huang's algorithm to our purpose.

Stability under Classifications

Under classifications, “blocking” needs to be redefined to respect class quotas.

Stability under Classifications

Under classifications, "blocking" needs to be redefined to respect class quotas. Huang defines a blocking group, which in the absence of lower quotas coincides with a new kind of blocking pair.

Stability under Classifications

Under classifications, "blocking" needs to be redefined to respect class quotas. Huang defines a blocking group, which in the absence of lower quotas coincides with a new kind of blocking pair. We call this a feasible blocking pair.

Stability under Classifications

Under classifications, "blocking" needs to be redefined to respect class quotas. Huang defines a blocking group, which in the absence of lower quotas coincides with a new kind of blocking pair. We call this a feasible blocking pair.
A feasible blocking pair w.r.t. M is an unmatched edge that would deviate from M without violating class quotas.

Stability under Classifications

Under classifications, "blocking" needs to be redefined to respect class quotas. Huang defines a blocking group, which in the absence of lower quotas coincides with a new kind of blocking pair. We call this a feasible blocking pair.
A feasible blocking pair w.r.t. M is an unmatched edge that would deviate from M without violating class quotas.

Definition

$C_h^r \in \mathscr{C}_h$ be the class in which h puts r. A feasible blocking pair is a blocking pair (h, r) such that either $|M(h) \cap C_h^r| < q_h^+(C_h^r)$, or $\exists r' \in M(h) \cap C_h^r, r' <_h r$.

Stability under Classifications

Under classifications, "blocking" needs to be redefined to respect class quotas. Huang defines a blocking group, which in the absence of lower quotas coincides with a new kind of blocking pair. We call this a feasible blocking pair.
A feasible blocking pair w.r.t. M is an unmatched edge that would deviate from M without violating class quotas.

Definition

$C_h^r \in \mathscr{C}_h$ be the class in which h puts r. A feasible blocking pair is a blocking pair (h, r) such that either $|M(h) \cap C_h^r| < q_h^+(C_h^r)$, or $\exists r' \in M(h) \cap C_h^r, r' <_h r$.

Definition

M is class-stable if it has no feasible blocking pairs.

Application to Students-Courses with Lower Quotas

- In the transformed instance with copies and dummies, each course considers all copies of a student to be a class with upper quota 1.

Application to Students-Courses with Lower Quotas

- In the transformed instance with copies and dummies, each course considers all copies of a student to be a class with upper quota 1.
- With a few minor observations, our earlier proofs go through for a class-stable (rather than stable) matching in the transformed instance.

Application to Students-Courses with Lower Quotas

- In the transformed instance with copies and dummies, each course considers all copies of a student to be a class with upper quota 1.
- With a few minor observations, our earlier proofs go through for a class-stable (rather than stable) matching in the transformed instance.
- The many-to-many version can be reduced to the case in Huang's algorithm. However it takes $\mathcal{O}(|E|^2)$ time, which due to successive reductions gives us $\mathcal{O}(|E|^8)$ in our application.

Application to Students-Courses with Lower Quotas

- In the transformed instance with copies and dummies, each course considers all copies of a student to be a class with upper quota 1.
- With a few minor observations, our earlier proofs go through for a class-stable (rather than stable) matching in the transformed instance.
- The many-to-many version can be reduced to the case in Huang's algorithm. However it takes $\mathcal{O}(|E|^2)$ time, which due to successive reductions gives us $\mathcal{O}(|E|^8)$ in our application.
- We adapt Huang's algorithm to a linear-time version specific to our requirement.

Application to Students-Courses with Lower Quotas

Our scenario : Students-Courses, courses impose partition-classes on students, with class upper quotas.

Application to Students-Courses with Lower Quotas

Our scenario : Students-Courses, courses impose partition-classes on students, with class upper quotas.
Huang's algorithm is resident-proposing, i.e. the side that imposes classes receives proposals from the side that is classified. So our algorithm is student-proposing. The algorithm is as follows :

Application to Students-Courses with Lower Quotas

Our scenario : Students-Courses, courses impose partition-classes on students, with class upper quotas.
Huang's algorithm is resident-proposing, i.e. the side that imposes classes receives proposals from the side that is classified. So our algorithm is student-proposing. The algorithm is as follows :

- Students propose to courses
- A course c, on receiving a proposal from a student s :

Application to Students-Courses with Lower Quotas

Our scenario : Students-Courses, courses impose partition-classes on students, with class upper quotas.
Huang's algorithm is resident-proposing, i.e. the side that imposes classes receives proposals from the side that is classified. So our algorithm is student-proposing. The algorithm is as follows :

- Students propose to courses
- A course c, on receiving a proposal from a student s :
  - If c is undersubscribed and C^s is undersubscribed, it accepts.

Application to Students-Courses with Lower Quotas

Our scenario : Students-Courses, courses impose partition-classes on students, with class upper quotas.
Huang's algorithm is resident-proposing, i.e. the side that imposes classes receives proposals from the side that is classified. So our algorithm is student-proposing. The algorithm is as follows :

- Students propose to courses
- A course c, on receiving a proposal from a student s :
  - If c is undersubscribed and C^s is undersubscribed, it accepts.
  - If C^s is full, it compares s with the least preferred existing student in C^s, and keeps the more preferred one.

Application to Students-Courses with Lower Quotas

Our scenario : Students-Courses, courses impose partition-classes on students, with class upper quotas.
Huang's algorithm is resident-proposing, i.e. the side that imposes classes receives proposals from the side that is classified. So our algorithm is student-proposing. The algorithm is as follows :

- Students propose to courses
- A course c, on receiving a proposal from a student s :
  - If c is undersubscribed and C^s is undersubscribed, it accepts.
  - If C^s is full, it compares s with the least preferred existing student in C^s, and keeps the more preferred one.
  - If C^s has space but c is overall full, then it instead compares s with the overall least-preferred student.

Application to Students-Courses with Lower Quotas

Our scenario : Students-Courses, courses impose partition-classes on students, with class upper quotas.
Huang's algorithm is resident-proposing, i.e. the side that imposes classes receives proposals from the side that is classified. So our algorithm is student-proposing. The algorithm is as follows :

- Students propose to courses
- A course c, on receiving a proposal from a student s :
  - If c is undersubscribed and C^s is undersubscribed, it accepts.
  - If C^s is full, it compares s with the least preferred existing student in C^s, and keeps the more preferred one.
  - If C^s has space but c is overall full, then it instead compares s with the overall least-preferred student.

Correctness : similar to Gale-Shapley.

Contents

Students-Courses with both-sided Lower Quotas

- If we give priorities to both sides, it appears we need classes on both sides.

Students-Courses with both-sided Lower Quotas

- If we give priorities to both sides, it appears we need classes on both sides. Stability under both-sided classifications is studied in [Fleiner and Kamiyama, 2016].

Students-Courses with both-sided Lower Quotas

- If we give priorities to both sides, it appears we need classes on both sides. Stability under both-sided classifications is studied in [Fleiner and Kamiyama, 2016].
- But both-sided classifications are not enough – $(s^0, c^2), (s^1, c^3)$ might both be matched in the transformed instance.

Students-Courses with both-sided Lower Quotas

- If we give priorities to both sides, it appears we need classes on both sides. Stability under both-sided classifications is studied in [Fleiner and Kamiyama, 2016].
- But both-sided classifications are not enough – $(s^0, c^2), (s^1, c^3)$ might both be matched in the transformed instance. Fact : this only occurs with one side at priorities 0 and 1.

Students-Courses with both-sided Lower Quotas

- If we give priorities to both sides, it appears we need classes on both sides. Stability under both-sided classifications is studied in [Fleiner and Kamiyama, 2016].
- But both-sided classifications are not enough – $(s^0, c^2), (s^1, c^3)$ might both be matched in the transformed instance. Fact : this only occurs with one side at priorities 0 and 1.
- We need more sophisticated constraints,

Students-Courses with both-sided Lower Quotas

- If we give priorities to both sides, it appears we need classes on both sides. Stability under both-sided classifications is studied in [Fleiner and Kamiyama, 2016].
- But both-sided classifications are not enough – $(s^0, c^2), (s^1, c^3)$ might both be matched in the transformed instance. Fact : this only occurs with one side at priorities 0 and 1.
- We need more sophisticated constraints, like restricted edges or matroid constraints.

Questions

Thank you

References I


Brandl, F. and Kavitha, T. (2017).
Popular matchings with multiple partners.
In Lokam, S. V. and Ramanujam, R., editors, *37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2017, December 11-15, 2017, Kanpur, India*, volume 93 of *LIPIcs*, pages 19:1–19:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.


Fleiner, T. and Kamiyama, N. (2016).
A matroid approach to stable matchings with lower quotas.
Mathematics of Operations Research, 41(2):734–744.


Gale, D. and Shapley, L. S. (1962).
College admissions and the stability of marriage.
The American Mathematical Monthly, 69(1):9–15.

References II

Huang, C. and Kavitha, T. (2011).
Popular matchings in the stable marriage problem.
In Aceto, L., Henzinger, M., and Sgall, J., editors, *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part I*, volume 6755 of *Lecture Notes in Computer Science*, pages 666–677. Springer.

Huang, C.-C. (2010).
Classified stable matching.
In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, page 1235–1253, USA. Society for Industrial and Applied Mathematics.

References III


Nasre, M. and Nimbhorkar, P. (2017).
Popular matchings with lower quotas.
In Lokam, S. V. and Ramanujam, R., editors, *37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2017, December 11-15, 2017, Kanpur, India*, volume 93 of *LIPIcs*, pages 44:1–44:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.


Roth, A. E. (1986).
On the allocation of residents to rural hospitals: A general property of two-sided matching markets.
Econometrica, 54(2):425–427.