CMSC5706 Topics in Theoretical Computer Science

Week 7: Stable Matching

Instructor: Shengyu Zhang
Bipartite graph

- (Undirected) Bipartite graph:
  - $G = (V, E)$ for which $V$ can be partitioned into two parts
    - $V = M \cup W$ with $M \cap W = \emptyset$,
  - And all edges $e = (m, w)$ have $m \in M$ and $w \in W$. 

Diagram showing a bipartite graph with two distinct sets $M$ and $W$.
Matching, maximum matching

- **Matching**: a collection of vertex-disjoint edges
  - a subset $E' \subseteq E$ s.t. no two edges $e, e' \in E'$ are incident.

- $|E'|$: size of matching.

- **Maximum matching**: a matching with the maximum size.

- This lecture: matching in a bipartite graph
Perfect matching

- There may be some vertices not incident to any edge.

- **Perfect matching**: a matching with no such isolated vertex.
  - needs at least: $|M| = |W|

- We’ll assume $|M| = |W|$ in the rest of the lecture.
Men’s Preference

- Suppose a man sees these women.
- He has a preference among them.
  - What’s your preference list?
- Different men may have different lists.
Women’s preference

- Women also have their preference lists.

- Assume no tie.
  - The general case can be handled similarly.
Setting

- **n men, n women**
- Each **man** has a preference list of all **women**
- Each **woman** has a preference list of all **men**
- We want to match them.

\[
\begin{align*}
    w_1 &> w_2 > w_3 > w_4 & m_1 \\
    w_1 &> w_2 > w_3 > w_4 & m_2 \\
    w_2 &> w_1 > w_3 > w_4 & m_3 \\
    w_3 &> w_2 > w_4 > w_1 & m_4
\end{align*}
\]
Setting

- Consider this matching.
- And this pair \((m_1, w_1)\).
  - \(m_1\) is matched to \(w_2\), but he likes \(w_1\) more.
  - \(w_1\) is matched to \(m_2\), but she likes \(w_1\) more.
- What if \(m_1\) and \(w_1\) meet one day?

\[
\begin{align*}
w_1 & > w_2 > w_3 > w_4 & m_1 & < w_1 & m_3 & > m_1 > m_2 > m_4 \\
w_1 & > w_2 > w_3 > w_4 & m_2 & < w_2 & m_3 & > m_4 > m_1 > m_2 \\
w_2 & > w_1 > w_3 > w_4 & m_3 & < w_3 & m_1 & > m_4 > m_2 > m_3 \\
w_3 & > w_2 > w_4 > w_1 & m_4 & < w_4 & m_4 & > m_1 > m_3 > m_2
\end{align*}
\]
A stability property

- Suppose there are two couples with these preferences.

  - The marriage is unstable, because $m_1$ and $w_1$ like each other more than their currently assigned ones!

  \[
  w_1 > w_2 \quad m_1 > m_2 \quad w_1 > w_2 \quad m_1 > m_2
  \]
Stability

- Such a pair is called a blocking pair.

$w_1 > w_2$  $m_1 \rightarrow w_1$  $m_1 > m_2$

$w_1 > w_2$  $m_2 \rightarrow w_2$  $m_1 > m_2$

- **Question**: Can we have a matching without any blocking pair?
  - Such a matching is then called a stable matching.
Real applications

- If you think marriage is a bit artificial since there is no centralized arranger, here is a real application.

- Medical students work as interns at hospitals.
  - In the US more than 20,000 medical students and 4,000 hospitals are matched through a clearinghouse, called NRMP (National Resident Matching Program).
Real applications

- Students and hospitals submit preference rankings to the clearinghouse, who uses a specified rule to decide who works where.

- Question: What is a good way to match students and hospitals?
More than one question

- **Question:** Does a stable matching always exist?

- **Question:** If yes, how to find one?

- **Question:** What mathematical / economic properties it has?
Good news: Stable matchings always exist.

- **Theorem** (Gale-Shapley) For any given preference lists, there always exists a stable matching.

- They actually gave an algorithm, which bears some resemblance to real marriages.
Consider a simple dynamics

- ∀ matching \( f \), ∀ blocking pair \((m, w)\),
  - Remove the old pairing \((m, f(m))\) and \((w, f(w))\)
    - \( f(m) \): the woman matched to \( m \) in \( f \). (\( f(w) \): similar.)
  - Match \( m \) and \( w \)
  - Match \( f(m) \) and \( f(w) \)

- Question: Would repeating this finally lead to a stable matching?

\[
\begin{align*}
\text{w}_1 & > \text{w}_2 \\
\text{m}_1 & > \text{m}_2 \\
\text{w}_1 & > \text{w}_2 \\
\text{m}_2 & > \text{m}_2
\end{align*}
\]
Example

- Can you find an counterexample?
- Next we’ll give an algorithm that actually works.
- Let’s first run the algorithm on an example.
Algorithm by an example

\[ w_1 > w_2 > w_3 > w_4 \]

\[ m_1 \quad m_3 > m_1 > m_2 > m_4 \]

\[ w_1 \quad m_3 > m_4 > m_1 > m_2 \]

\[ w_2 \quad m_1 > m_4 > m_2 > m_3 \]

\[ w_3 \quad m_4 > m_1 > m_3 > m_2 \]

\[ w_4 \]
Gale-Shapley (Deferred-Acceptance) Algorithm

- Initially all men and women are free
- while there is a man $m$ who is free and hasn’t proposed to every woman
  - choose such a man $m$ arbitrarily
  - let $w$ be the highest ranked woman in $m$’s preference list to whom $m$ hasn’t proposed yet
  - // next: $m$ proposes to $w$
    - if $w$ is free, then $(m, w)$ become engaged
    - else, suppose $w$ is currently engaged to $m’$
      - if $w$ prefers $m’$ to $m$, then $m$ remains free
      - if $w$ prefers $m$ to $m’$, then $(m, w)$ becomes engaged and $m’$ becomes free
- Return the set of engaged pairs as a matching
Analysis of the algorithm

- We will show the following:

1. The algorithm always terminates…

2. … in $O(n^2)$ steps, // $n$ men and $n$ women.

3. and generates a stable matching.
Some observations

- In each iteration, one man $m$ proposes to a new woman $w$.
- For any man: The women he proposes to get worse and worse
  - according to his preference list
  - Because he proposes to a new woman only when the previous one dumps him
    - forcing him to try next (worse!) ones.
Time bound

- Each **man** proposes at most $n$ steps.
  - since his proposed **women** are worse and worse
- There are $n$ **men**.
- Therefore: at most $n^2$ proposals.
- Since each iteration has exactly one proposal, there are at most $n^2$ iterations.
- **Theorem.** Gale-Shapley algorithm terminates after at most $n^2$ iterations.
Correctness

Suppose the algorithm returns a matching $f$ with a blocking pair $(m, w)$,

- i.e. $m$ prefers $w$ to $w'$ and $w$ prefers $m$ to $m'$, where $w'$ and $m'$ are their current partner.

Note: $m$’s last proposal was to $w'$; see the algorithm.

$m$ has proposed to $w$ before to $w'$.
- Since $m$ proposes from best to worst.

But at the end of the day, $w$ chose $m'$

So $m'$ also proposed to $w$ at some point.
Correctness

Suppose the algorithm returns a matching $f$ with a blocking pair $(m, w)$,

- i.e. $m$ prefers $w$ to $w'$ and $w$ prefers $m$ to $m'$, where $w'$ and $m'$ are their current partner.

So both $m$ and $m'$ proposed to $w$.

And $w$ finally married $m'$ instead of $m$.

No matter who, $m$ or $m'$, proposed first, $w$ prefers $m'$ to $m$.

A contradiction to our assumption.
Some observations

- For any **man**: His fiancé gets worse and worse (according to his preference list)
  - because he changes fiancé only when the previous one dumps him, forcing him to try next (worse!) ones.

- For any **woman**: Her fiancé gets better and better (according to her preference list)
  - because she changes fiancé only when a better man proposes to her.
Women propose?

- What if women propose?

\[ w_1 > w_2 > w_3 \quad m_1 > m_2 > m_3 \]

\[ w_1 > w_2 > w_3 \quad m_1 > m_3 > m_2 \]

\[ w_1 > w_3 > w_2 \quad m_1 > m_2 > m_3 \]
Which stable matching is better?

- As a man, which matching you prefer?
  - What if you are $m_1$? What if you are $m_2$?

- As a woman, which matching you prefer?
  - What if you are $w_1$? What if you are $w_2$?

**GS algorithm: men propose**

$w_1 > w_2$  $m_1$

$w_2 > w_1$  $m_2$

**GS algorithm: women propose**

$w_1$  $m_2 > m_1$

$w_2$  $m_1 > m_2$
Stable Matching by G-S, men propose

- For any man $m$, his set of valid partners is $vp(m) = \{w: f(m) = w \text{ for some stable matching } f\}$

- $\text{best}(m)$: the best $w \in vp(m)$.
  - “best”: according to $m$’s preference.

- **Theorem.** Gale-Shapley algorithm matches all men $m$ to $\text{best}(m)$.

- **Implications:**
  - different orders of free men picked do not matter
  - for any men $m_1 \neq m_2$, $\text{best}(m_1) \neq \text{best}(m_2)$
Proof

- For contradiction, assume that some $m^*$ is matched to worse than $w^* = \text{best}(m^*)$.
- Since $m^*$ proposes in the decreasing order, $m^*$ must be rejected by $w^*$ in the course of the GS algorithm.
- Note that $w^* \in \nu p(m^*)$. So there exists a man rejected by his valid partner.
Proof

- Consider the first such moment \( t \) that some \( m \) is rejected by some \( w \in vp(m) \).
- Since \( m \) proposes in the decreasing order, \( w = best(m) \).
- What triggers the rejection?
  - Either \( m \) proposed but was turned down (\( w \) prefers her current partner),
  - or \( w \) broke her engagement to \( m \) in favor of a better proposal.
- In either case, at moment \( t \), \( w \) is engaged to a man \( m' \) whom she prefers to \( m \), i.e., \( m' >_w m \).
Proof

- By def of \( \text{best}(m) \), \( \exists \) a stable matching \( f \) assigning \( m \) to \( w \).
- Assume that \( m' \) is matched to \( w' \neq w \) in \( f \).
- At moment \( t \), \( m \) is first man rejected by someone in \( \text{vp}(m) \).
- So no one in \( \text{vp}(m') \), including \( w' \), rejected \( m' \) by now.
  - \( w' \in \text{vp}(m') \) since \( w' \) and \( m' \) are paired up in the stable matching \( f \).
- If \( w <_{m'} w' \), \( m' \) should have proposed to \( w' \). But now \( m' \) is with \( w \), so \( m' \) has been dumped by \( w' \). Impossible.
- Hence \( w >_{m'} w' \). Contradiction to fact that \( f \) is stable. \( \square \)
How about women?

- Recall: $\text{best}(m)$ is the best woman matched to $m$ in all possible stable matchings.
- GS algorithm matches all men $m$ to $\text{best}(m)$.
- $\text{worst}(w)$ is the worst man matched to $w$ in all possible stable matchings.

- **Theorem.** GS algorithm matches all women $w$ to $\text{worst}(w)$. 
Proof

- By the last theorem, each $m$ is matched to $w = best(m)$ when GS(men propose) gives $f$.
- We’ll show that $m = worst(w)$.
- Suppose there is a stable matching $f'$ in which $w$ is matched to an even worse $m' <_w m$.
- Consider $m$’s partner in $f'$; call her $w'$.
- $w >_m w'$, because $w = f(m) = best(m)$.
- Then $(m, w)$ is a blocking pair in $f'$. Contradiction!
Who should propose?

- Thus if men propose, then
- in each man’s eyes:
  - His engaged women get worse and worse.
  - But finally he gets the best possible. (The best that avoids a later divorce.)
- in each woman’s eyes:
  - Her engaged men get better and better.
  - But finally she gets the worst possible. (The worst that avoids a later divorce.)
Recall: Gale-Shapley algorithm runs in time $O(n^2)$ in the worst case.

Question: Can we improve this?

Note: An input has $O(n^2 \log n)$ bits, so even reading the input needs this much time.

So the above question should be asked in certain random access model.
Query

- For example, such queries
  - What’s woman $w$’s ranking of man $m$?
  - Which man does woman $w$ rank at place $k$?
  - Who does woman $w$ prefer, $m$ or $m'$?
  - ...

- The above examples are on women’s preferences. Similarly we can have queries on men’s preferences.

- Some queries need $\log n$ bits to answer, some need only 1 bit.
  - The latter is called Boolean queries.
Observation: *Communication* can simulate all these queries.
Recall: Communication complexity

- Two parties, Alice and Bob, jointly compute a function $f$ on input $(x, y)$.
  - $x$ known only to Alice and $y$ only to Bob.
- **Communication complexity**: how many bits are needed to be exchanged?
communication setting

Suppose that Alice has all women’s preference lists,
and Bob has all men’s preference lists.
Then any aforementioned query can be simulated by communication.
Algorithm to protocol

- **Fact.** Any algorithm using $k$ queries of $b$-bit answer can be made into a communication protocol using $kb$ communication bits.

- **Method:** Both Alice and Bob run the algorithm. Whenever they need to make a query, the one who has the answer tells the other.
Algorithm to protocol

- E.g. consider query “What’s woman $w$’s ranking of man $m$?”
- Alice has the answer
  - since she owns all women’s preference lists
- So Alice sends the answer to Bob, who then also knows the answer to continue the algorithm.
Lower bounds

- **Theorem.** Any protocol to find a stable matching needs $\Omega(n^2)$ communication bits.

- **Theorem.** Any protocol verifying whether a given matching is stable needs $\Omega(n^2)$ communication bits.

Together with the query-communication relation, we know that it takes $\Omega(n^2/t)$ queries if each query has a $t$-bit answer.

- In particular, both tasks need $\Omega(n^2)$ Boolean queries.
Lower bounds

- **Theorem.** Any protocol to find a stable matching needs $\Omega(n^2)$ communication bits.

- **Theorem.** Any protocol verifying whether a given matching is stable needs $\Omega(n^2)$ communication bits.

- **Method:** Reduce the problem to a well-known problem called Disjointness.
Recall: Communication complexity

\[ x \in \{0,1\}^N \]

\[ y \in \{0,1\}^N \]

\[ \text{Disj}_N(x, y) = \begin{cases} 
0 & \text{if } \exists i \text{ s.t. } x_i = y_i = 1 \\
1 & \text{otherwise} 
\end{cases} \]

- **Theorem.** Any protocol solving Disj$_N$ problem needs $\Omega(N)$ communication bits.
  - even for randomized protocols.
Reduction to verification

- For two strings $x$ and $y$ both of $n(n - 1)$ bits,
  - as input of $\text{Disj}_N$, where $N = n(n - 1)$
- we map them to instance of Stable Matching
- For $w_i$: $(m_j: x_{ij} = 1)m_i(m_j: x_{ij} = 0)$
- For $m_j$: $(w_i: y_{ij} = 1)w_j(w_i: y_{ij} = 0)$
- Matching $\mu_{id} = \{(1,1), \ldots, (n,n)\}$.
- $\mu_{id}$ is unstable $\iff \exists (i, j), x_{ij} = 1$ and $y_{ij} = 1$ $\iff \text{Disj}_N(x, y) = 0$
Finding

- The lower bound for finding a stable matching is similar, but a bit more technically involved.

- Omitted here.
Summary for Stable Matching

- A bipartite matching is stable if no block pair exists.

- Gale-Shapley algorithm finds a stable matching by at most $n^2$ iterations.
  - This $\Omega(n^2)$ complexity is necessary.

- Whichever side proposes finally get their best possible.