EEE360C: Algorithms

Stable Matching Problem

Summer 2019

Department of Electrical and Computer Engineering
University of Texas at Austin

The Problem

The Informal Problem

Consider the problem of optimally matching a set of applicants to a set of open positions.

The Informal Problem

Consider the problem of optimally matching a set of applicants to a set of open positions.

- Applicants to summer internships

The Informal Problem

Consider the problem of optimally matching a set of applicants to a set of open positions.

- Applicants to summer internships
- Applicants to graduate school

The Informal Problem

Consider the problem of optimally matching a set of applicants to a set of open positions.

- Applicants to summer internships
- Applicants to graduate school
- Medical school graduate applicants to residency programs

The Informal Problem

Consider the problem of optimally matching a set of applicants to a set of open positions.

- Applicants to summer internships
- Applicants to graduate school
- Medical school graduate applicants to residency programs
- Eligible males wanting to marry eligible females

The Informal Problem

Consider the problem of optimally matching a set of applicants to a set of open positions.

- Applicants to summer internships
- Applicants to graduate school
- Medical school graduate applicants to residency programs
- Eligible males wanting to marry eligible females

Actually, it seems like it should be easy. Why is it a hard problem, practically?

The Informal Problem

Consider the problem of optimally matching a set of applicants to a set of open positions.

- Applicants to summer internships
- Applicants to graduate school
- Medical school graduate applicants to residency programs
- Eligible males wanting to marry eligible females

Actually, it seems like it should be easy. Why is it a hard problem, practically?

Let's think about how to solve the problem if we have perfect information...

Stablity and Instability

Given a set of preferences among hospitals and medical students, define a self-reinforcing admissions process.

Stablity and Instability

Given a set of preferences among hospitals and medical students, define a self-reinforcing admissions process.

Unstable Pair

Applicant x and hospital y are unstable if:

- x prefers y to its assigned hospital
- y prefers x to one of its admitted students

Stablity and Instability

Given a set of preferences among hospitals and medical students, define a self-reinforcing admissions process.

Unstable Pair

Applicant x and hospital y are unstable if:

- x prefers y to its assigned hospital
- y prefers x to one of its admitted students

Stable Assignment

A stable assignment is one with no unstable pairs.

- This is a natural and desirable condition.
- Individual self-interest will prevent any applicant/hospital deal being made.

Formulating the Problem

The Problem, Formally

Consider a set $M = \{m_1, \ldots, m_n\}$ of n men and a set $W = \{w_1, \ldots, w_n\}$ of n women.

Formulating the Problem

The Problem, Formally

Consider a set $M = \{m_1, \ldots, m_n\}$ of n men and a set $W = \{w_1, \ldots, w_n\}$ of n women.

- A matching S is a set of ordered pairs, each from $M \times W$, s.t. each member of M and each member of W appears in at most one pair in S.

The Problem, Formally

Consider a set $M = \{m_1, \ldots, m_n\}$ of n men and a set $W = \{w_1, \ldots, w_n\}$ of n women.

- A matching S is a set of ordered pairs, each from $M \times W$, s.t. each member of M and each member of W appears in at most one pair in S.
- A perfect matching S' is a matching s.t. each member of M and each member of W appears in exactly one pair in S'.

The Problem, Formally

Consider a set $M = \{m_1, \ldots, m_n\}$ of n men and a set $W = \{w_1, \ldots, w_n\}$ of n women.

- A matching S is a set of ordered pairs, each from $M \times W$, s.t. each member of M and each member of W appears in at most one pair in S.
- A perfect matching S' is a matching s.t. each member of M and each member of W appears in exactly one pair in S'.
- Each man $m \in M$ ranks all of the women; m prefers w to w' if m ranks w higher than w'. We refer to the ordered ranking of m as his preference list.

Formulating the Problem

The Problem, Formally

Consider a set $M = \{m_1, \ldots, m_n\}$ of n men and a set $W = \{w_1, \ldots, w_n\}$ of n women.

- A matching S is a set of ordered pairs, each from $M \times W$, s.t. each member of M and each member of W appears in at most one pair in S.
- A perfect matching S' is a matching s.t. each member of M and each member of W appears in exactly one pair in S'.
- Each man $m \in M$ ranks all of the women; m prefers w to w' if m ranks w higher than w'. We refer to the ordered ranking of m as his preference list.
- Each woman ranks all of the men in the same way.

Formulating the Problem

The Problem, Formally

Consider a set $M = \{m_1, \ldots, m_n\}$ of n men and a set $W = \{w_1, \ldots, w_n\}$ of n women.

- A matching S is a set of ordered pairs, each from $M \times W$, s.t. each member of M and each member of W appears in at most one pair in S.
- A perfect matching S' is a matching s.t. each member of M and each member of W appears in exactly one pair in S'.
- Each man $m \in M$ ranks all of the women; m prefers w to w' if m ranks w higher than w'. We refer to the ordered ranking of m as his preference list.
- Each woman ranks all of the men in the same way.
- An instability results when a perfect matching S contains two pairs (m, w) and (m', w') s.t. m prefers w' to w and w' prefers m to m'.

Formulating the Problem

The Problem, Formally

Consider a set $M = \{m_1, \ldots, m_n\}$ of n men and a set $W = \{w_1, \ldots, w_n\}$ of n women.

- A matching S is a set of ordered pairs, each from $M \times W$, s.t. each member of M and each member of W appears in at most one pair in S.
- A perfect matching S' is a matching s.t. each member of M and each member of W appears in exactly one pair in S'.
- Each man $m \in M$ ranks all of the women; m prefers w to w' if m ranks w higher than w'. We refer to the ordered ranking of m as his preference list.
- Each woman ranks all of the men in the same way.
- An instability results when a perfect matching S contains two pairs (m, w) and (m', w') s.t. m prefers w' to w and w' prefers m to m'.

GOAL: A perfect set of marriages with no instabilities.

An Example

Consider the following pairs exist in matching set S,

- (m, w)
- (sm', w')

An Example

Consider the following pairs exist in matching set S,

- (m, w)
- (sm', w')

when the preference list is

- m prefers w' to w
- w' prefers m' to m

An Example

Consider the following pairs exist in matching set S,

- (m, w)
- (sm', w')

when the preference list is

- m prefers w' to w
- w' prefers m' to m

(m, w') is an instability with respect to S

Questions About Stable Marriage

Key Questions

1. Does there exist a stable matching for every set of preference lists?

Questions About Stable Marriage

Key Questions

1. Does there exist a stable matching for every set of preference lists?
2. Given a set of preference lists, can we efficiently construct a stable matching if there is one?

The Solution

The Gale-Shapley Algorithm

GALE-SHAPLEY()

The Gale-Shapley Algorithm

GALE-SHAPLEY()

The Gale-Shapley Algorithm

GALE-SHAPLEY()

```
Initially all m ∈ M and w ∈ W are free
```

The Gale-Shapley Algorithm

```
GALE-SHAPLEY()
Initially all m ∈ M and w ∈ W are free
while ∃m who is free and hasn't proposed to every w ∈ W
    do Choose such a man m
```

The Gale-Shapley Algorithm

Gale-Shapley()

```
Initially all m ∈ M and w ∈ W are free
while ∃m who is free and hasn't proposed to every w ∈ W
    do Choose such a man m
       Let w be the highest ranked in m's preference list
         to whom m has not yet proposed
```

The Gale-Shapley Algorithm

GALE-SHAPLEY()

Initially all $m \in M$ and $w \in W$ are free
while $\exists m$ who is free and hasn't proposed to every $w \in W$
    do Choose such a man m
        Let w be the highest ranked in m's preference list
          to whom m has not yet proposed
        if w is free
            then (m, w) become engaged

The Gale-Shapley Algorithm

GALE-SHAPLEY()

Initially all $m \in M$ and $w \in W$ are free
while $\exists m$ who is free and hasn't proposed to every $w \in W$
    do Choose such a man m
        Let w be the highest ranked in m's preference list
          to whom m has not yet proposed
        if w is free
          then (m, w) become engaged
          else w is currently engaged to m'

The Gale-Shapley Algorithm

GALE-SHAPLEY()

Initially all $m \in M$ and $w \in W$ are free
while $\exists m$ who is free and hasn't proposed to every $w \in W$
    do Choose such a man m
        Let w be the highest ranked in m's preference list
          to whom m has not yet proposed
        if w is free
          then (m, w) become engaged
          else w is currently engaged to m'
              if w prefers m' to m
                then m remains free

The Gale-Shapley Algorithm

GALE-SHAPLEY()

Initially all $m \in M$ and $w \in W$ are free
while $\exists m$ who is free and hasn't proposed to every $w \in W$
    do Choose such a man m
        Let w be the highest ranked in m's preference list
          to whom m has not yet proposed
        if w is free
          then (m, w) become engaged
          else w is currently engaged to m'
                if w prefers m' to m
                  then m remains free
                  else w prefers m to m'
                        (m, w) become engaged
                        m' becomes free

The Gale-Shapley Algorithm

GALE-SHAPLEY()

Initially all $m \in M$ and $w \in W$ are free
while $\exists m$ who is free and hasn't proposed to every $w \in W$
    do Choose such a man m
        Let w be the highest ranked in m's preference list
          to whom m has not yet proposed
        if w is free
          then (m, w) become engaged
          else w is currently engaged to m'
              if w prefers m' to m
                then m remains free
                else w prefers m to m'
                    (m, w) become engaged
                    m' becomes free
return the set S of engaged pairs

The Proofs

But Does it Work?

It’s never a bad idea to convince yourself informally on a simple example...

But Does it Work?

It's never a bad idea to convince yourself informally on a simple example...

Some Axioms

But Does it Work?

It's never a bad idea to convince yourself informally on a simple example...

Some Axioms

- w remains engaged from the point at which she receives her first proposal

But Does it Work?

It's never a bad idea to convince yourself informally on a simple example...

Some Axioms

- *w* remains engaged from the point at which she receives her first proposal
- the sequence of partners with which *w* is engaged gets increasingly better (in terms of her preference list)

But Does it Work?

It's never a bad idea to convince yourself informally on a simple example...

Some Axioms

- w remains engaged from the point at which she receives her first proposal
- the sequence of partners with which w is engaged gets increasingly better (in terms of her preference list)
- the sequence of women to whom m proposes get increasingly worse (in terms of his preference list)

Observations

Men propose to women in decreasing order of preference (they're "optimistic").

Observations

Men propose to women in decreasing order of preference (they're "optimistic").

Once a woman is matched, she never becomes unmatched (she only "trades up").

Termination

Theorem

The G-S algorithm terminates after at most n^2 iterations of the while loop.

Termination

Theorem

The G-S algorithm terminates after at most n^2 iterations of the while loop.

What is a good measure of progress?

Termination

Theorem

The G-S algorithm terminates after at most n^2 iterations of the while loop.

What is a good measure of progress?

- the number of free men?

Termination

Theorem

The G-S algorithm terminates after at most n^2 iterations of the while loop.

What is a good measure of progress?

- the number of free men?
- the number of engaged couples?

Termination

Theorem

The G-S algorithm terminates after at most n^2 iterations of the while loop.

What is a good measure of progress?

- the number of free men?
- the number of engaged couples?
- the number of proposals made?

Termination

Theorem

The G-S algorithm terminates after at most n^2 iterations of the while loop.

What is a good measure of progress?

- the number of free men?
- the number of engaged couples?
- the number of proposals made?

Proof

Each iteration consists of one man proposing to a woman he has never proposed to before. So we count the number of proposals. After each iteration of the while loop, the number of proposals increases by one; the total number of proposals is upper bounded by n^2.

Towards a Perfect Matching

Theorem

If m is free at some point in the execution of the algorithm, then there is a woman to whom he has not yet proposed.

Towards a Perfect Matching

Theorem

If m is free at some point in the execution of the algorithm, then there is a woman to whom he has not yet proposed.

Proof

If, at some point, m is free but has already proposed to every woman. Then every woman must be engaged (because once engaged, they stay engaged, and they would have said yes to m if they weren't engaged when he proposed). Since all n women are engaged there must be n engaged men. This contradicts the claim that m is free.

Towards a Perfect Matching (cont.)

Theorem

The set S returned at termination is a perfect matching.

Towards a Perfect Matching (cont.)

Theorem

The set S returned at termination is a perfect matching.

Proof

Suppose the algorithm terminates with a free man m. Then m must have proposed to every woman (otherwise the while loop would still be active, and we wouldn't be at termination). But this contradicts the previous theorem, which stated that there cannot be a free man that has proposed to every woman.

Correctness

Theorem

Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.

Correctness

Theorem

Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.

Proof

- Assume there is an instability.

Correctness

Theorem

Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.

Proof

- Assume there is an instability. Then there exist two pairs (m, w) and (m', w') in S s.t.

$$m \text{ prefers } w' \text{ to } w \tag{1}$$

$$w' \text{ prefers } m \text{ to } m' \tag{2}$$

Correctness

Theorem

Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.

Proof

- Assume there is an instability. Then there exist two pairs (m, w) and (m', w') in S s.t.

$$m \text{ prefers } w' \text{ to } w \tag{1}$$

$$w' \text{ prefers } m \text{ to } m' \tag{2}$$

- Since m is matched with w. During execution, m's last proposal must have been to w. Had m proposed to w' at some earlier time?

Correctness

Theorem

Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.

Proof

- Assume there is an instability. Then there exist two pairs (m, w) and (m', w') in S s.t.

$$m \text{ prefers } w' \text{ to } w \tag{1}$$

$$w' \text{ prefers } m \text{ to } m' \tag{2}$$

- Since m is matched with w. During execution, m's last proposal must have been to w. Had m proposed to w' at some earlier time?
- If no, w must be higher than w' on m's preference list else m would have proposed to w'. This is a contradition to eq. 1.

Correctness

Theorem

Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.

Proof

- Assume there is an instability. Then there exist two pairs (m, w) and (m', w') in S s.t.
$$m \text{ prefers } w' \text{ to } w \tag{1}$$
$$w' \text{ prefers } m \text{ to } m' \tag{2}$$
- Since m is matched with w. During execution, m's last proposal must have been to w. Had m proposed to w' at some earlier time?
- If no, w must be higher than w' on m's preference list else m would have proposed to w'. This is a contradition to eq. 1.
- If yes, then he was rejected by w' in favor of some other guy m''.

Correctness

Theorem

Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.

Proof

- Assume there is an instability. Then there exist two pairs (m, w) and (m', w') in S s.t.

$$m \text{ prefers } w' \text{ to } w \tag{1}$$

$$w' \text{ prefers } m \text{ to } m' \tag{2}$$

- Since m is matched with w. During execution, m's last proposal must have been to w. Had m proposed to w' at some earlier time?
- If no, w must be higher than w' on m's preference list else m would have proposed to w'. This is a contradition to eq. 1.
- If yes, then he was rejected by w' in favor of some other guy m''.
- Either $m'' = m'$ or w' prefers m' to m'' (since the quality of her match only goes up). Either way, this is a contradicts eq. 2

Correctness

Theorem

Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.

Proof

- Assume there is an instability. Then there exist two pairs (m, w) and (m', w') in S s.t.

$$m \text{ prefers } w' \text{ to } w \tag{1}$$

$$w' \text{ prefers } m \text{ to } m' \tag{2}$$

- Since m is matched with w. During execution, m's last proposal must have been to w. Had m proposed to w' at some earlier time?
- If no, w must be higher than w' on m's preference list else m would have proposed to w'. This is a contradition to eq. 1.
- If yes, then he was rejected by w' in favor of some other guy m''.
- Either $m'' = m'$ or w' prefers m' to m'' (since the quality of her match only goes up). Either way, this is a contradicts eq. 2
- Therefore, S is a stable matching.

In Class Exercise

Exercise

Prove that if all men have the same list of preferences and all women have the same list of preferences, then only one stable matching exists.

In Class Exercise

Exercise

Prove that if all men have the same list of preferences and all women have the same list of preferences, then only one stable matching exists.

Solution - Proof by Contradiction

The matching is exactly the rank order (i.e., the top ranked man marries the top ranked woman; the second ranked man marries the second ranked woman, and so on). Suppose this was not the case. Then woman at rank i in the mens' order is married to some man at rank j in the womens' order. Say $j > i$. There must also exist another "mismatched" pair such that the woman at rank i' is married to the man at rank j', $j' < i'$. i and j' are ranked higher on each others preference lists, and this represents an instability in which each would leave their partners and remarry.

Implementation

Summary

The Gale-Shapley algorithm guarantees to find a stable matching for any problem instance.

Summary

The Gale-Shapley algorithm guarantees to find a stable matching for any problem instance.

- How do we implement the Gale-Shapley algorithm efficiently?
- If there are multiple stable matchings, which one does the algorithm find?

Implementation

We can describe a $O(n^2)$ implementation.

Implementation

We can describe a $O(n^2)$ implementation.

Representing Men and Women

- Assume men are named $1 \ldots n$ and women are named $1' \ldots n'$

Implementation

We can describe a $O(n^2)$ implementation.

Representing Men and Women

- Assume men are named $1 \dots n$ and women are named $1' \dots n'$

Engagements

- maintain a list of free men in a queue
- maintain two arrays of length n, $\mathit{wife}[m]$ and $\mathit{husband}[w]$
  - set entry to 0 if unmatched
  - if m matched to w, then $\mathit{wife}[m] = w$ and $\mathit{husband}[w] = m$

Implementation

We can describe a $O(n^2)$ implementation.

Representing Men and Women

- Assume men are named $1 \ldots n$ and women are named $1' \ldots n'$

Engagements

- maintain a list of free men in a queue
- maintain two arrays of length n, $wife[m]$ and $husband[w]$
  - set entry to 0 if unmatched
  - if m matched to w, then $wife[m] = w$ and $husband[w] = m$

Proposals

- For each man, maintain a list of women, ordered by preference
- Maintain an array $count[m]$, the number of proposals made by m

Implementation (cont.)

Women Rejecting/Accepting

- For each woman, create inverse of preference list.
- Allows constant time queries:
  - A woman prefers m to m' if $\mathit{inverse}[m] < \mathit{inverse}[m']$

Implementation (cont.)

Women Rejecting/Accepting

- For each woman, create inverse of preference list.
- Allows constant time queries:
  - A woman prefers m to m' if $inverse[m] < inverse[m']$

Proposal Process

- The first free man, m, in the queue proposes the woman at the front of his preference list, w
- He increments $count[m]$ and removes w from his preference list
- w accepts the proposal if she is unengaged or prefers m to her current match
- if w accepts, her former match goes back on the queue of men; otherwise m proposes to his next favorite

Understanding the Solution

For a given problem instance, there may be several stable matchings. Do all executions of Gale-Shapley yield the same stable matching? If so, which one?

Understanding the Solution (cont.)—An Example

Given the following preference list:

$$m \text{ prefers } w \text{ to } w'$$
$$m' \text{ prefers } w' \text{ to } w$$
$$w \text{ prefers } m' \text{ to } m$$
$$w' \text{ prefers } m \text{ to } m'$$

Understanding the Solution (cont.)—An Example

Given the following preference list:

$$m \text{ prefers } w \text{ to } w'$$
$$m' \text{ prefers } w' \text{ to } w$$
$$w \text{ prefers } m' \text{ to } m$$
$$w' \text{ prefers } m \text{ to } m'$$

In any execution of G-S algorithm,

$$m \text{ becomes engaged to } w$$
$$m' \text{ becomes engaged to } w'$$

Understanding the Solution (cont.)—An Example

Given the following preference list:

$$m \text{ prefers } w \text{ to } w'$$
$$m' \text{ prefers } w' \text{ to } w$$
$$w \text{ prefers } m' \text{ to } m$$
$$w' \text{ prefers } m \text{ to } m'$$

In any execution of G-S algorithm,

$$m \text{ becomes engaged to } w$$
$$m' \text{ becomes engaged to } w'$$

But, there is another possible stable matching (m', w) and (m, w').

Understanding the Solution (cont.)—An Example

Given the following preference list:

$$m \text{ prefers } w \text{ to } w'$$
$$m' \text{ prefers } w' \text{ to } w$$
$$w \text{ prefers } m' \text{ to } m$$
$$w' \text{ prefers } m \text{ to } m'$$

In any execution of G-S algorithm,

$$m \text{ becomes engaged to } w$$
$$m' \text{ becomes engaged to } w'$$

But, there is another possible stable matching (m', w) and (m, w').
However, this possibility is not attainable in the version of G-S algorithm where men propose.

Understanding the Solution (cont.)

- w is a *valid partner* for a man m if there is a stable matching that contains the pair (m, w).

Understanding the Solution (cont.)

- w is a *valid partner* for a man m if there is a stable matching that contains the pair (m, w).
- w is the *best valid partner* of a man m if w is a valid partner of m and no woman whom m ranks higher than w is a valid partner of his.

Understanding the Solution (cont.)

- w is a *valid partner* for a man m if there is a stable matching that contains the pair (m, w).
- w is the *best valid partner* of a man m if w is a valid partner of m and no woman whom m ranks higher than w is a valid partner of his.
- A man-optimal assignment is one in which every man receives the best valid partner

Understanding the Solution (cont.)

- *w* is a *valid partner* for a man *m* if there is a stable matching that contains the pair (m, w).
- *w* is the *best valid partner* of a man *m* if *w* is a valid partner of *m* and no woman whom m ranks higher than *w* is a valid partner of his.
- A man-optimal assignment is one in which every man receives the best valid partner

Claims

Understanding the Solution (cont.)

- w is a *valid partner* for a man m if there is a stable matching that contains the pair (m, w).
- w is the *best valid partner* of a man m if w is a valid partner of m and no woman whom m ranks higher than w is a valid partner of his.
- A man-optimal assignment is one in which every man receives the best valid partner

Claims

- Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

Understanding the Solution (cont.)

- w is a *valid partner* for a man m if there is a stable matching that contains the pair (m, w).
- w is the *best valid partner* of a man m if w is a valid partner of m and no woman whom m ranks higher than w is a valid partner of his.
- A man-optimal assignment is one in which every man receives the best valid partner

Claims

- Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.
- Claim 2: All executions of GS yield woman-pessimal assignment, which is a stable matching (i.e., each woman receives the worst possible valid partner).

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner.
  (Note: Men propose in decreasing order of preference.)

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner. *(Note: Men propose in decreasing order of preference.)*
- $\implies$ Some man is rejected by a valid partner during G-S algorithm.

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner. *(Note: Men propose in decreasing order of preference.)*
- $\implies$ Some man is rejected by a valid partner during G-S algorithm.
- Let *m* be such a man and let *w* be the first valid women that rejects him.

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner. *(Note: Men propose in decreasing order of preference.)*
- $\implies$ Some man is rejected by a valid partner during G-S algorithm.
- Let m be such a man and let w be the first valid women that rejects him.
- Let S be a stable matching where (m, w) are matched. *(Note: Such a set exists because w is a valid partner of m.)*

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner. *(Note: Men propose in decreasing order of preference.)*
- $\implies$ Some man is rejected by a valid partner during G-S algorithm.
- Let m be such a man and let w be the first valid women that rejects him.
- Let S be a stable matching where (m, w) are matched. *(Note: Such a set exists because w is a valid partner of m.)*
- When m is rejected by w in GS, w forms or continues engagement with m'.

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner. *(Note: Men propose in decreasing order of preference.)*
- $\implies$ Some man is rejected by a valid partner during G-S algorithm.
- Let m be such a man and let w be the first valid women that rejects him.
- Let S be a stable matching where (m, w) are matched. *(Note: Such a set exists because w is a valid partner of m.)*
- When m is rejected by w in GS, w forms or continues engagement with m'.
- $\implies$ w prefers m' to m.

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner. *(Note: Men propose in decreasing order of preference.)*
- $\implies$ Some man is rejected by a valid partner during G-S algorithm.
- Let m be such a man and let w be the first valid women that rejects him.
- Let S be a stable matching where (m, w) are matched. *(Note: Such a set exists because w is a valid partner of m.)*
- When m is rejected by w in GS, w forms or continues engagement with m'.
- $\implies$ w prefers m' to m.
- Let w' be a partner of m' in S.

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner. *(Note: Men propose in decreasing order of preference.)*
- $\implies$ Some man is rejected by a valid partner during G-S algorithm.
- Let m be such a man and let w be the first valid women that rejects him.
- Let S be a stable matching where (m, w) are matched. *(Note: Such a set exists because w is a valid partner of m.)*
- When m is rejected by w in GS, w forms or continues engagement with m'.
- $\implies$ w prefers m' to m.
- Let w' be a partner of m' in S.
- m' has not been rejected by any valid partner at the point when m is rejected by w. *Note: This is because this is a first reject by a valid partner.*

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner. *(Note: Men propose in decreasing order of preference.)*
- $\implies$ Some man is rejected by a valid partner during G-S algorithm.
- Let m be such a man and let w be the first valid women that rejects him.
- Let S be a stable matching where (m, w) are matched. *(Note: Such a set exists because w is a valid partner of m.)*
- When m is rejected by w in GS, w forms or continues engagement with m'.
- $\implies$ w prefers m' to m.
- Let w' be a partner of m' in S.
- m' has not been rejected by any valid partner at the point when m is rejected by w. *Note: This is because this is a first reject by a valid partner.*
- Thus m' has not yet proposed to w' when he proposes to w. *Note: This is because proposals are done in decreasing order of preference.*

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner. *(Note: Men propose in decreasing order of preference.)*
- $\implies$ Some man is rejected by a valid partner during G-S algorithm.
- Let m be such a man and let w be the first valid women that rejects him.
- Let S be a stable matching where (m, w) are matched. *(Note: Such a set exists because w is a valid partner of m.)*
- When m is rejected by w in GS, w forms or continues engagement with m'.
- $\implies$ w prefers m' to m.
- Let w' be a partner of m' in S.
- m' has not been rejected by any valid partner at the point when m is rejected by w. *Note: This is because this is a first reject by a valid partner.*
- Thus m' has not yet proposed to w' when he proposes to w. *Note: This is because proposals are done in decreasing order of preference.*
- $\implies$ m' prefers w to w'.

Proof of Claim 1:

Claim 1: All executions of GS yield man-optimal assignment, which is a stable matching.

- Suppose a man is matched with someone other than his best valid partner. *(Note: Men propose in decreasing order of preference.)*
- $\implies$ Some man is rejected by a valid partner during G-S algorithm.
- Let m be such a man and let w be the first valid women that rejects him.
- Let S be a stable matching where (m, w) are matched. *(Note: Such a set exists because w is a valid partner of m.)*
- When m is rejected by w in GS, w forms or continues engagement with m'.
- $\implies$ w prefers m' to m.
- Let w' be a partner of m' in S.
- m' has not been rejected by any valid partner at the point when m is rejected by w. *Note: This is because this is a first reject by a valid partner.*
- Thus m' has not yet proposed to w' when he proposes to w. *Note: This is because proposals are done in decreasing order of preference.*
- $\implies$ m' prefers w to w'.
- Thus (m', w) is unstable in S, a contradiction.

Variations

What About Similar Problems?

Consider the stable roommate problem. $2n$ people each rank the others from 1 to $2n - 1$. The goal is to assign roommate pairs so that none are unstable.

	1st	2nd	3rd
Adam	B	C	D
Bob	C	A	D
Chris	A	B	D
Doofus	A	B	C

A-B, C-D $\Rightarrow$ B-C unstable
A-C, B-D $\Rightarrow$ A-B unstable
A-D, B-C $\Rightarrow$ A-C unstable

Observation: a stable matching for the stable roommate problem doesn't always exist.

Final Thoughts

Steps in Algorithm Design

- Formulate the problem precisely.
- Design an algorithm for the problem.
- Prove the algorithm correct.
- Give a bound on the algorithm's running time.

Design Techniques

We'll explore algorithm design by enumerating a set of design techniques. And learning to recognize problems that likely belong to one class or another.

Questions

?