CSE 421: Introduction to Algorithms

Stable Matching

Shayan Oveis Gharan

Propose-And-Reject Algorithm [Gale-Shapley'62]

```
Initialize each side to be free.
while (some company is free and hasn't proposed to every
applicant) {
    Choose such a c
    a = 1st woman on c's list to whom c has not yet proposed
    if (a is free)
        assign c and a
    else if (a prefers c to her current c')
        assign c and a, and c' to be free
    else
        a rejects c
}
```

First step: Properties of Algorithm

Observation 1: Companies propose to Applicants in decreasing order of preference.

Observation 2: Each company proposes to each applicant at most once

Observation 3: Once an applicant is matched, she never becomes unmatched; she only "trades up."

1) Termination

Claim. Algorithm terminates after $\leq \boldsymbol{n^2}$ iterations of while loop.

Proof. Observation 2: Each company proposes to each applicant at most once.

Each company makes at most n proposals

So, there are only n^2 possible proposals. ▪

	1st	2nd	3rd	4th	5th
Vmware	A	B	C	D	E
Walmart	B	C	D	A	E
Xfinity	C	D	A	B	E
Yamaha	D	A	B	C	E
Zoom	A	B	C	D	E

	1st	2nd	3rd	4th	5th
Amy	W	X	Y	Z	V
Brenda	X	Y	Z	V	W
Claire	Y	Z	V	W	X
Diane	Z	V	W	X	Y
Erika	V	W	X	Y	Z

n(n-1) + 1 proposals required

2) Correctness: Output is Perfect matching

Claim. All Companies and Applicants get matched.

Proof. (by contradiction)

Suppose, for sake of contradiction, that c_1 is not matched upon termination of algorithm.

Then some applicant, say a_1, is not matched upon termination.

By Observation 3 (only trading up, never becoming unmatched), a_1 was never proposed to.

But, c_1 proposes to everyone, since it ends up unmatched.

■

2) Correctness: Stability

Claim. No unstable pairs.

Proof. (by contradiction)

Suppose c, a is an unstable pair: each prefers each other to the partner in Gale-Shapley matching **S***.

Obs1: companies propose in decreasing order of preference

Case 1: c never proposed to a.

$\Rightarrow$ c prefers its **S*** partner to a.

$\Rightarrow$ c, a is stable.

Case 2: c proposed to a.

$\Rightarrow$ a rejected c (right away or later)

$\Rightarrow$ a prefers her **S*** partner to c.

$\Rightarrow$ c, a is stable.

Obs3: applicants only trade up

In either case c, a is stable, a contradiction. ■

Summary

Stable matching problem: Given n companies and n applicants, and their preferences, find a stable matching if one exists.

- Gale-Shapley algorithm: Guarantees to find a stable matching for any problem instance.

- Q: How to implement GS algorithm efficiently?

- Q: If there are multiple stable matchings, which one does GS find?

- Q: How many stable matchings are there?

Efficient Implementation

We describe $O(n^2)$ time implementation. This is linear in input size.

Representing company and applicant:

Assume companies are named 1, …, n.
Assume applicants are named n+1, …, 2n.

Data Structure:

Maintain a list of free company, e.g., in a queue.
Maintain two arrays **applicant[c]**, and **company[a]**.

- set entry to **0** if unmatched
- if **c** matched to **a** then **applicant[c]=a** and **company[a]=c**

Companies proposing:

For each company, maintain a list of applicants, ordered by preference.
Maintain an array **count[c]** that counts the number of proposals made by company **c**.

Efficient Implementation

Applicants rejecting/accepting.

Does applicant **a** prefer **c** to **c'**?

For each applicant, create inverse of preference list of companies.

Constant time access for each query after **O(n)** preprocessing per appliacant. **O(n²)** total reprocessing cost.

a_i	1st	2nd	3rd	4th	5th	6th	7th	8th
Pref	8	3	7	1	4	5	6	2

a_i	1	2	3	4	5	6	7	8
Inverse	4th	8th	2nd	5th	6th	7th	3rd	1st

```
for i = 1 to n
   for j = 1 to n
      inverse[i][pref[i][j]] = j
```

a_i prefers company **3** to **6**

since `inverse[i][3]`=2 < 7=`inverse[i][6]`

Summary

Stable matching problem: Given n companies and n applicants, and their preferences, find a stable matching if one exists.

- Gale-Shapley algorithm: Guarantees to find a stable matching for any problem instance.
- Q: How to implement GS algorithm efficiently?
- Q: If there are multiple stable matchings, which one does GS find?
- Q: How many stable matchings are there?

Understanding the Solution

Q. 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?

An instance with two stable matchings:

- $(c_1, a_1), (c_2, a_2)$.
- $(c_1, a_2), (c_2, a_1)$.

	1st	2nd
c_1	a_1	a_2
c_2	a_2	a_1

	1st	2nd
a_1	c_2	c_1
a_2	c_1	c_2

Company Optimal Assignments

Definition: Company c is a valid partner of applicant a if there exists some stable matching in which they are matched.

Company-optimal matching: Each company receives the best valid partner (according to his preferences).

- Not that each company receives its most favorite applicant.

Example

Here
Valid-partner(c_1) $= \{a_1, a_2\}$
Valid-partner(c_2) $= \{a_1, a_2\}$
Valid-partner(c_3) $= \{a_3\}$.

Company-optimal matching $\{c_1, a_1\}, \{c_2, a_2\}, \{c_3, a_3\}$

	favorite ↓ 1st	2nd	least favorite ↓ 3rd
c_1	a_1	a_2	a_3
c_2	a_2	a_1	a_3
c_3	a_1	a_2	a_3

	favorite ↓ 1st	2nd	least favorite ↓ 3rd
a_1	c_2	c_1	c_3
a_2	c_1	c_2	c_3
a_3	c_1	c_2	c_3

Company Optimal Assignments

Definition: Company c is a valid partner of applicant a if there exists some stable matching in which they are matched.

Company-optimal matching: Each company receives the best valid partner (according to its preferences).

- Not that each company receives its most favorite applicant.

Claim: All executions of GS yield a company-optimal matching, which is a stable matching!

- So, output of GS is unique!!
- No reason a priori to believe that company-optimal matching is perfect, let alone stable.

Company Optimality

S
(c, a)
(c', a')
$\ldots$

Claim: GS matching **S*** is company-optimal.

Proof: (by contradiction)

Suppose some company is paired with someone other than its best partner. Companies propose in decreasing order of preference $\Rightarrow$ some company is rejected by a valid partner.

Let c be the first such rejection, and let a be its best valid partner.

Let **S** be a stable matching where c and a are matched.

In building **S***, when c is rejected, a is assigned to a company, say c' whom she prefers to c.

Let c' be a' partner in **S**.

In building **S***, c' is not rejected by any valid partner at the point when c is rejected by a. Thus, c' prefers a to a'.

But a prefers c' to c.

Thus (c', a) is unstable in **S**. ■

since this is the first rejection by a valid partner

Man Optimality Summary

Company-optimality: In version of GS where companies propose, each comapny receives the best valid partner.

a is a valid partner of c if there exist some stable matching where c and a are paired

Q: Does company-optimality come at the expense of the applicants?

Applicant Pessimality

Applicant-pessimal assignment: Each applicant receives the worst valid partner.

Claim. GS finds applicant-pessimal stable matching **S***.

Proof.

Suppose (c, a) matched in **S***, but c is not the worst valid partner for a.

There exists stable matching **S** in which a is paired with a company, say c', whom she likes less than c.

Let a' be c partner in **S**.

c prefers a to a'. ← company-optimality of **S***

Thus, (c, a) is an unstable in **S**.

■

Summary

- Stable matching problem: Given **n** men and **n** women, and their preferences, find a stable matching if one exists.

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

- GS algorithm finds a stable matching in $O(n^2)$ time.
- GS algorithm finds man-optimal woman pessimal matching
- Q: How many stable matching are there?

Lessons Learned

- Powerful ideas learned in course.
  - Isolate underlying structure of problem.
  - Create useful and efficient algorithms.

- Potentially deep social ramifications. [legal disclaimer]
  - Always try to propose first!

How many stable Matchings?

We already show every instance has at least 1 stable matchings.

There are instances with about 2.24^n stable matchings for

[Karlin-O-Weber'17]: Every instance has at most 131072^n stable matchings

[Palmer-Palvolgyi'20]: Every instance has at most 4.47^n stable matchings

[Research-Question]:

Is there an "efficient" algorithm that chooses a uniformly random stable matching of a given instance.

Extensions: Matching Residents to Hospitals

Comapnies ≈ hospitals, Applicants ≈ med school residents.

- Variant 1: Some participants declare others as unacceptable.
- Variant 2: Unequal number of companies and applicants.
- Variant 3: A hospital wants to hire multiple residents

e.g. A resident not interested in Cleveland

An analogous version of GS algorithm works!