CSE 421: Introduction to Algorithms

Stable Matching

Shayan Oveis Gharan

Administrativia Stuffs

Lectures: M/W/F 1:30-2:20
Location: Gates G20

Office hours: M 12:30-1:20 W 2:30-3:20, Allen center 636

Discussion Board: Use edstem https://edstem.org

Practice website: https://usaco.training

CSE 421: Introduction to Algorithms

Winter, 2018

Shayan Oveis Gharan

MWF 2:30-3:20, MGH 389
Office hours in CSE 636
M/W/F 3:30-4:20

Textbook:

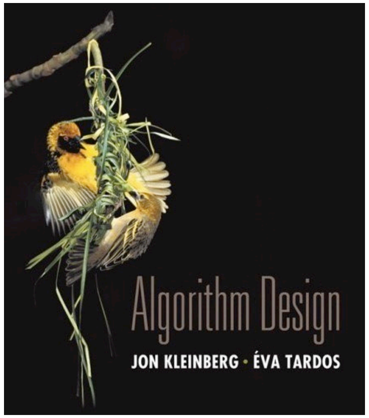
Algorithm Design by Jon Kleinberg and Eva Tardos, Addison-Wesley, 2006. We will cover almost all of chapters 1-8 of the Kleinberg/Tardos text plus some additional material from later chapters. In addition, I recommend reading chapter 5 of Introduction to Algorithms: A Creative Approach, by Udi Manber, Addison-Wesley 1989. This book has a unique point of view on algorithm design.

Another handy reference is Steven Skiena's Stonybrook Algorithm Repository

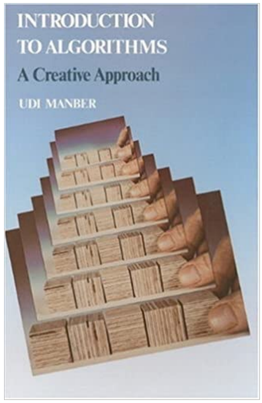
Grading Scheme (Roughly):

Homework 50%
Midterm 15-20%
Final Exam 30-35%

cs.washington.edu/421



Course textbook



Supplementary text

TAs

Xiyang Liu	Mon 10:30-11:20 AM	
Marian Dietz	Mon 3:30-4:20 PM	
Raymond Patrick Guo	Mon 4:30-5:30	
Robert Stevens	Tue 11:30-12:20	
Airei Fukuzawa	Tue 12:30-1:20 PM	Gates (CSE II) 150
Aman Thukral	Tue 2:30-3:30	Gates (CSE II) 121
Tom Tian	Tue 3:30-4:30	Gates (CSE II) 150
Dorna Abdolazimi	Tue 4:30-5:30	
Sophie Lin Robertson	Wed 10:30-11:30	
Sela Navot	Wed 11:30-12:30	Gates 131
Albert Weng	Wed 3:30-4:30	Gates 150

Sections

- Sections participation is mandatory.
- HW problems in this course are hard. TAs will solve related problems from previous offering of the curse
- It is a very good opportunity to improve your problem solving skills

Grading

- Weekly HWs, First HW due April 3rd
- Submit to Gradescope
- Midterm (04/29/2024), Final (06/04/2024)
  - Exams are open book, open note, no internet access
  - Midterm 50 minutes, Final 110 minutes.
- HW 50%, Midterm 15-20%, Final 30-35%
- Extra Credit problems can boost your final GPA by 0.1

Daily Quizzes

- One quiz before every lecture
- 1-2 questions about the materials of the previous lecture
- Typically yes/no or multiple choice

- Login to canvas (assignment tab) to access the quiz

- Will release questions in the morning before class, you have around 3-4 minutes to answer

- Daily Quizes can boost up your final GPA by 0.1
- If you don’t answer any of them you can still get 4.0!

Structure of the course

- First 2-3 lectures overview of proof techniques
  - Proof by Contradiction
  - Induction
  - Take a look at CSE 311 Lectures/assignments for preparation
- Graph Algorithms
- Greedy Algorithms
- Divid & Conquor

Midterm

- Dynamic Programming,
- Network Flow
- Approximation Algorithms and Linear Programming
- Np Completeness

Final

Stable Matching Problem

Given n companies $c_1, \ldots, c_n$,
and n applicants, $a_1, \ldots, a_n$
find a "stable matching".

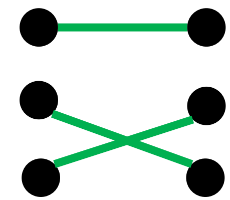
- Participants rate members of opposite group.
- Each company lists applicants in order of preference.
- Each applicant lists companies in order of preference.

	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

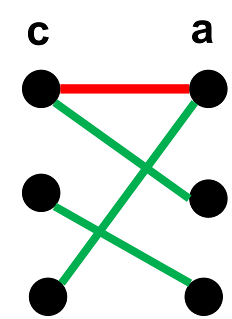
Stable Matching

Perfect matching:
- Each company gets exactly one applicant.
- Each applicant gets exactly one company.



Stability: no incentive for some pair of participants to undermine assignment by joint action.

In a matching M, an unmatched pair a-c is unstable if a and c prefer each other to current partners.



Stable matching: perfect matching with no unstable pairs.

Stable matching problem: Given the preference lists of n companies and n applicants, find a stable matching if one exists.

Example

Question. Is assignment $(c_1, a_3), (c_2, a_2), (c_3, a_1)$ stable?

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

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

Example

Question. Is assignment $(c_1, a_3), (c_2, a_2), (c_3, a_1)$ stable?

Answer. No. a_2, c_1 will hook up.

favorite ↓ … least favorite ↓

	1st	2nd	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 ↓ … least favorite ↓

	1st	2nd	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

Example

Question: Is assignment $(c_1, a_1), (c_2, a_2), (c_3, a_3)$ stable?

Answer: Yes.

favorite ↓ least favorite ↓

	1st	2nd	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 ↓ least favorite ↓

	1st	2nd	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

Existence of Stable Matchings

Question. Do stable matchings always exist?
Answer. Yes, but not obvious a priori.

Stable roommate problem:

2n people; each person ranks others from **1** to **2n-1**.
Assign roommate pairs so that no unstable pairs.

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

A-B, C-D ⇒ B-C unstable
A-C, B-D ⇒ A-B unstable
A-D, B-C ⇒ A-C unstable

So, Stable matchings do not always exist for stable roommate problem.

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."

What do we need to prove?

1) The algorithm ends in a "small" number of steps.
   - How many steps does it take?

2) The algorithm is correct [usually the harder part]
   - It outputs a perfect matching
   - The output matching is stable

1) Termination / Runtime

Claim. Algorithm terminates after $\leq n^2$ 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: they prefer 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?