CSE 421 Winter 2025
Lecture 1: Intro, Stable Matching

Nathan Brunelle

http://www.cs.uw.edu/421

Course Goals

Two Goals:

1. Learn specific noteworthy algorithms
2. Hone insights on how to design algorithms for novel problems

What is an algorithm?

- a finite sequence of **mathematically rigorous** **instructions**, typically used to **solve** a class of specific problems or to perform a **computation**. [January 2025]
- a finite sequence of **well-defined** **instructions**, typically used to **solve** a class of specific problems or to perform a **computation**. [November 2021]
- a set of **instructions**, typically to **solve** a class of problems or perform a **computation**. [August 2019]
- an **unambiguous** specification of how to **solve** a class of problems. [September 2018]
- How it will sometimes feel

Course Goals

~~Two Goals:~~ Three Goals

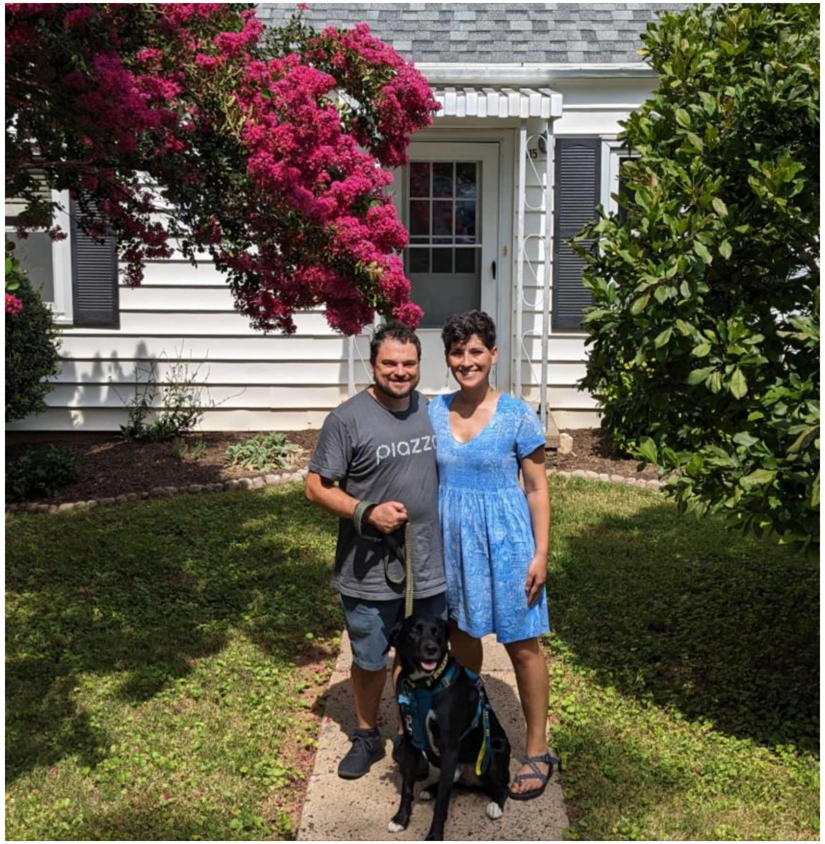
1. Learn specific noteworthy algorithms
2. Hone insights on how to design algorithms for novel problems
3. Have Fun!

Hopefully not you…

I Quit!

Nathan Brunelle

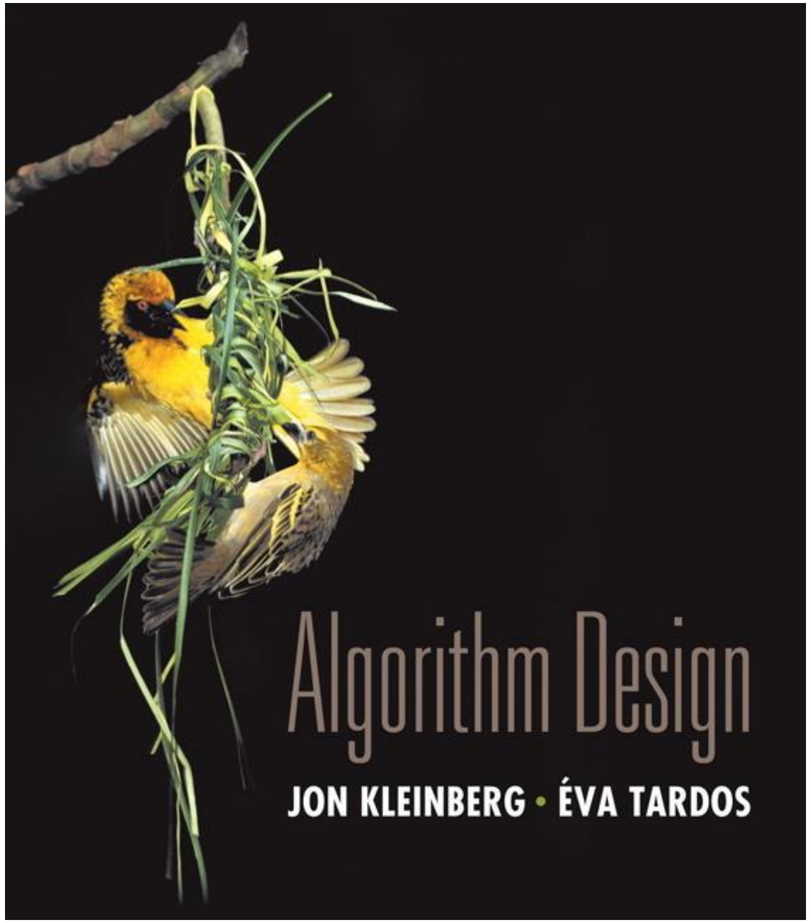
- Born: Virginia Beach, VA
- Ugrad: Math and CS at University of Virginia
- Grad: CS at University of Virginia
- Taught at UVA for 6 years
  - Intro to programming (e.g. 121)
  - Discrete Math (e.g. 311)
  - Algorithms (e.g. 412)
  - Theory of Computation (e.g. 431)



Our Amazing TAs!

Course Info

- Text:
  - Kleinberg and Tardos Algorithm Design
    - Recommended but not required
  - Other supplements, which we'll make available
- Course Page:
  - http://www.cs.uw.edu/421



Communication

- EdStem Discussion board
    - Your first stop for questions about course content & assignments
    - Course announcements will be made there
        - Announcements will be forwarded to your email as well

Course Meetings

- Lecture
  - Materials posted (slides before class, inked slides after)
  - Recorded using Panopto
  - Ask questions, focus on key ideas (rarely coding details)

- Section
  - Practice problems!
  - Answer content/homework questions

- Office hours
  - Use them: *please visit us!*

Tasks

- 8 Weekly individual homework exercises (60%)
  - Mechanical Problems (1 per each)
    - Apply knowledge of a specific algorithm
  - Long-Form Problems (3 per each)
    - Design and analyze a new algorithm
- Midterm and final exam (40%)
  - Midterm: 15%
    - Wednesday Feb 19, 6:00pm-7:30pm
  - Final: 25%
    - Monday March 17, 2:30pm-5:20pm
    - cumulative

Grading

- Homework:
  - Each mechanical problem is worth 10 points
  - Each long-form problem is worth 25 points
  - We count your 7 best mechanical problems (1 dropped)
  - We count your 20 best long-form problems (4 dropped)
  - Score is therefore out of 570 points
- Late Work:
  - Unless you talk to Nathan, nothing is accepted 48 hours after the deadline
  - You have 10 late problem-days. After those have been used, late problems will receive a 50% grade penalty (multiplicative)

Collaboration

- Try it yourself first
- Collaborate with classmates (no external interactive help on assignments permitted)
  - Collaboration is “whiteboard only”
  - Looking for a collaborator?
- Cite your sources!

Getting Started (Your TODO List)

- Make sure you are on Ed (a.k.a. EdStem)!
- Attend your first Quiz Section Thursday!
- Homework 1 will be out Wednesday
  - You will have enough to start on it after section Thursday
  - Start thinking about it right away after that
- Sign up for CSE 490D
- Attend lecture and participate
  - Students who participate do better on average

Matching Medical Residents to Hospitals

Goal: Given a set of preferences among hospitals and medical school residents (graduating medical students), design a *self-reinforcing* admissions process.

Unstable pair: applicant $\boldsymbol{x}$ and hospital $\boldsymbol{y}$ are *unstable* if:

- $\boldsymbol{x}$ prefers $\boldsymbol{y}$ to their assigned hospital.
- $\boldsymbol{y}$ prefers $\boldsymbol{x}$ to one of its admitted residents.

Stable assignment. Assignment with no unstable pairs.

- Natural and desirable condition.
- Individual self-interest will prevent any applicant/hospital side deal from being made.

Simplification: Stable Matching Problem

Goal: Given two groups of n people each, find a "suitable" matching.

- Participants rate members from opposite group.
- Each person lists members from the other group in order of preference from best to worst.

favorite ↓ least favorite ↓

	1st	2nd	3rd
X	A	B	C
Y	B	A	C
Z	A	B	C

*Group **P** Preference Profile*

favorite ↓ least favorite ↓

	1st	2nd	3rd
A	Y	X	Z
B	X	Y	Z
C	X	Y	Z

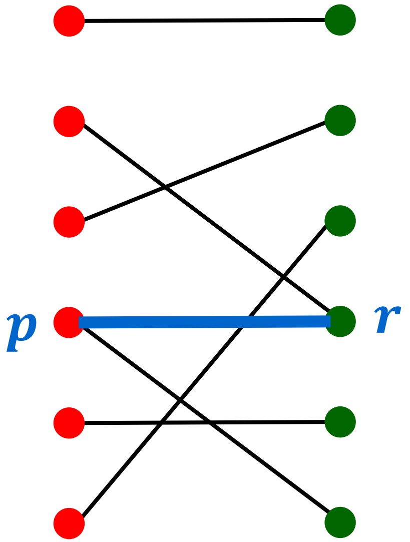
*Group **R** Preference Profile*

Stable Matching Problem

Perfect matching: everyone is matched to precisely one person from the other group

Stability: self-reinforcing, i.e. no pair has incentive to defect from their assignment.

- For a matching $\boldsymbol{M}$, an unmatched pair $\boldsymbol{p}$-$\boldsymbol{r}$ from different groups is *unstable* if $\boldsymbol{p}$ and $\boldsymbol{r}$ prefer each other to current partners.
- Unstable pair $\boldsymbol{p}$-$\boldsymbol{r}$ could each improve by ignoring the assignment.



Stable matching: perfect matching with no unstable pairs.

Stable matching problem: Given the preference lists of $\boldsymbol{n}$ people from each of two groups, find a stable matching between the two groups if one exists.

Stable Matching Problem

Q: Is matching (X,C), (Y,B), (Z,A) stable?

	favorite ↓		least favorite ↓
	1st	2nd	3rd
X	A	B	C
Y	B	A	C
Z	A	B	C

Group ***P*** *Preference Profile*

	favorite ↓		least favorite ↓
	1st	2nd	3rd
A	Y	X	Z
B	X	Y	Z
C	X	Y	Z

Group ***R*** *Preference Profile*

Stable Matching Problem

Q: Is matching (X,C), (Y,B), (Z,A) stable?

A: No. B and X prefer each other.

favorite ↓ least favorite ↓

	1st	2nd	3rd
X	A	B	C
Y	B	A	C
Z	A	B	C

*Group **P** Preference Profile*

favorite ↓ least favorite ↓

	1st	2nd	3rd
A	Y	X	Z
B	X	Y	Z
C	X	Y	Z

*Group **R** Preference Profile*

Stable Matching Problem

Q: Is matching (X,A), (Y,B), (Z,C) stable?

favorite ↓ least favorite ↓

	1st	2nd	3rd
X	A	B	C
Y	B	A	C
Z	A	B	C

*Group **P** Preference Profile*

favorite ↓ least favorite ↓

	1st	2nd	3rd
A	Y	X	Z
B	X	Y	Z
C	X	Y	Z

*Group **R** Preference Profile*

Stable Matching Problem

Q: Is matching (X,A), (Y,B), (Z,C) stable?

A: Yes

favorite ↓ least favorite ↓

	1st	2nd	3rd
X	A	B	C
Y	B	A	C
Z	A	B	C

*Group **P** Preference Profile*

favorite ↓ least favorite ↓

	1st	2nd	3rd
A	Y	X	Z
B	X	Y	Z
C	X	Y	Z

*Group **R** Preference Profile*

Variant: “Stable Roommate” Problem (one set rather than 2)

Q. Do stable matchings always exist?

A. Not exactly obvious…

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
A	B	C	D
B	C	A	D
C	A	B	D
D	A	B	C

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

Observation: Stable matchings do not always exist for stable roommate problem.

Propose-And-Reject Algorithm

Propose-and-reject algorithm: [Gale-Shapley 1962]

Intuitive method that guarantees to find a stable matching.

- Members of one group ***P*** make *proposals*, the other group ***R*** *receives* proposals

```
Initialize each person to be free
while (some p in P is free) {
    Choose some free p in P
    r = 1st person on p's preference list to whom p has not yet proposed
    if (r is free)
        tentatively match (p,r)   //p and r both engaged, no longer free
    else if (r prefers p to current tentative match p′)
        replace (p′,r) by (p,r)   //p now engaged, p′ now free
    else
        r rejects p
}
```

Propose and Reject Algorithm Example

```
Initialize each person to be free.
while (some p in P is free) {
    Choose some free p in P
    r = 1st person on p's preference list to whom p has not yet proposed
    if (r is free)
        tentatively match (p,r)   //p and r both engaged, no longer free
    else if (r prefers p to current tentative match p′)
        replace (p′,r) by (p,r)   //p now engaged, p′ now free
    else
        r rejects p
}
```

Tentative Matches:

X	
Y	
Z	

favorite ↓ least favorite ↓

	1st	2nd	3rd
X	A	B	C
Y	B	A	C
Z	A	B	C

*Group **P** Preference Profile*

favorite ↓ least favorite ↓

	1st	2nd	3rd
A	Y	X	Z
B	X	Y	Z
C	X	Y	Z

*Group **R** Preference Profile*

Why Does This Work?

- What do we need to know before we’re convinced that this algorithm is “correct”?

Why Does This Work?

- What do we need to know before we’re convinced that this algorithm is “correct”?
  - That is terminates (no infinite loop)
  - That it produces a stable matching
    - It’s perfect (everyone gets paired with exactly one partner)
    - It’s stable (no unmatched pair mutually prefer each other)

Proof of Correctness: Termination (not obvious from the code)

Observation 1: Members of P propose in decreasing order of preference.

Claim: The Gale-Shapley Algorithm terminates after at most n^2 iterations.

Proof: Proposals are never repeated (by Observation 1) and there are only n^2 possible proposals. ■

It could be nearly that bad…

General form of this example will take $n(n-1)+1$ proposals.

	1st	2nd	3rd	4th	5th
V	A	B	C	D	E
W	B	C	D	A	E
X	C	D	A	B	E
Y	D	A	B	C	E
Z	A	B	C	D	E

Preference Profile for P

	1st	2nd	3rd	4th	5th
A	W	X	Y	Z	V
B	X	Y	Z	V	W
C	Y	Z	V	W	X
D	Z	V	W	X	Y
E	V	W	X	Y	Z

Preference Profile for R

Proof of Correctness: Perfection

Observation 2: Once a member of $\boldsymbol{R}$ is matched, they never become free; they only "trade up.“

Claim: Everyone gets matched.

Proof:

- If no proposer is free then everyone is matched.
- After some $\boldsymbol{p}$ proposes to the last person on their list, all the $\boldsymbol{r}$ in $\boldsymbol{R}$ have been proposed to by someone (by $\boldsymbol{p}$ at least).
- By Observation 2, every $\boldsymbol{r}$ in $\boldsymbol{R}$ is matched at that point.
- Since $|\boldsymbol{P}| = |\boldsymbol{R}|$ every $\boldsymbol{p}$ in $\boldsymbol{P}$ is also matched. ■

Proof of Correctness: Stability

Claim: No unstable pairs in the final Gale-Shapley matching $\boldsymbol{M}$

Proof: Consider a pair $\boldsymbol{p}$-$\boldsymbol{r}$ not matched by $\boldsymbol{M}$

Case 1: $\boldsymbol{p}$ never proposed to $\boldsymbol{r}$.

$\Rightarrow$ $\boldsymbol{p}$ prefers $\boldsymbol{M}$-partner to $\boldsymbol{r}$.

$\Rightarrow$ $\boldsymbol{p}$-$\boldsymbol{r}$ is not unstable for $\boldsymbol{M}$.

Case 2: $\boldsymbol{p}$ proposed to $\boldsymbol{r}$.

$\Rightarrow$ $\boldsymbol{r}$ rejected $\boldsymbol{p}$ (right away or later when trading up)

$\Rightarrow$ $\boldsymbol{r}$ prefers $\boldsymbol{M}$-partner to $\boldsymbol{p}$.

$\Rightarrow$ $\boldsymbol{p}$-$\boldsymbol{r}$ is not unstable for $\boldsymbol{M}$. ■

Summary

Stable matching problem: Given n people in each of two groups, and their preferences, find a stable matching if one exists.

Stable: No pair of people both prefer to be with each other rather than with their assigned partner

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

⇒ Stable matching always exists!