

# CSE 421 Section 1

## Stable Matching

# **Administrivia & Introductions**



# Your Section TAs

- TA 1
  - Anything you want to say about yourself
- TA 2
  - content

# Homework

- Submissions
  - LaTeX
  - Word Editor that supports mathematical equations
- All homeworks will be turned in via Gradescope
- Homeworks typically due on Wednesdays at 11:59pm
  
- Late day policy – 5 late days. Maximum 2 days late per assignment. Otherwise 25% per day.

# Announcements & Reminders

- Section Materials
  - Handouts will be provided in each section
  - Worksheets and sample solutions will be available on the course calendar later this evening
- HW1
  - Due Wednesday 1/10 @ 11:59pm

# Stable Matching



# Stable Matching

Given  $2n$  people, in two groups, **M** and **W**, of  $n$  people, with each person having a preference list for members of the other group, how can we find a stable matching between them?

## Perfect Matching:

- Each person **m** in **M** is paired with exactly one person **w** in **W**
- Each person **w** in **W** is paired with exactly one person **m** in **M**

**Stability:** No ability to exchange partners

**Unstable:** An unmatched pair **m-w** is unstable if they both prefer each other to current matches

**Stable Matching:** perfect matching with no unstable pairs

# Gale-Shapley Algorithm

Algorithm to find a stable matching:

Initially all  $m$  in  $M$  and  $w$  in  $W$  are free  
while there is a free  $m$

    Let  $w$  be highest on  $m$ 's list that  $m$  has not proposed to  
    if  $w$  is free

        match  $(m, w)$

    else //  $w$  is not free

        Let  $m'$  be the current match of  $w$

        if  $w$  prefers  $m$  to  $m'$

            unmatch  $(m', w)$

            match  $(m, w)$



# Problem 1 – Gale-Shapley

Consider the following stable matching instance:

**m**<sub>1</sub>: w<sub>3</sub>, w<sub>1</sub>, w<sub>2</sub>, w<sub>4</sub>

**m**<sub>2</sub>: w<sub>2</sub>, w<sub>1</sub>, w<sub>4</sub>, w<sub>3</sub>

**m**<sub>3</sub>: w<sub>2</sub>, w<sub>3</sub>, w<sub>1</sub>, w<sub>4</sub>

**m**<sub>4</sub>: w<sub>3</sub>, w<sub>4</sub>, w<sub>1</sub>, w<sub>2</sub>

**w**<sub>1</sub>: m<sub>4</sub>, m<sub>1</sub>, m<sub>3</sub>, m<sub>2</sub>

**w**<sub>2</sub>: m<sub>1</sub>, m<sub>3</sub>, m<sub>2</sub>, m<sub>4</sub>

**w**<sub>3</sub>: m<sub>1</sub>, m<sub>3</sub>, m<sub>4</sub>, m<sub>2</sub>

**w**<sub>4</sub>: m<sub>3</sub>, m<sub>1</sub>, m<sub>2</sub>, m<sub>4</sub>

- a) Run the Gale-Shapley Algorithm on the instance above. When choosing which free  $m$  in  $\mathbf{M}$  to propose next, always choose the one with the smallest index (e.g., if  $m_1$  and  $m_2$  are both free, always choose  $m_1$ ).

# Problem 1 – Gale-Shapley

- b) Run the Gale-Shapley Algorithm on the instance above. When choosing which free  $m$  in  $\mathbf{M}$  to propose next, always choose the one with the *largest* index (e.g., if  $m_1$  and  $m_2$  are both free, always choose  $m_2$ ). Do you get the same result?

$\mathbf{m}_1$ :  $w_3, w_1, w_2, w_4$

$\mathbf{m}_2$ :  $w_2, w_1, w_4, w_3$

$\mathbf{m}_3$ :  $w_2, w_3, w_1, w_4$

$\mathbf{m}_4$ :  $w_3, w_4, w_1, w_2$

$\mathbf{w}_1$ :  $m_4, m_1, m_3, m_2$

$\mathbf{w}_2$ :  $m_1, m_3, m_2, m_4$

$\mathbf{w}_3$ :  $m_1, m_3, m_4, m_2$

$\mathbf{w}_4$ :  $m_3, m_1, m_2, m_4$

# Problem 1 – Gale-Shapley

- c) Now run the algorithm with the people in  $\mathbf{W}$  proposing, breaking ties by taking the free  $w_i$  with the smallest index. Do you get the same result?

$\mathbf{m}_1$ :  $w_3, w_1, w_2, w_4$

$\mathbf{m}_2$ :  $w_2, w_1, w_4, w_3$

$\mathbf{m}_3$ :  $w_2, w_3, w_1, w_4$

$\mathbf{m}_4$ :  $w_3, w_4, w_1, w_2$

$\mathbf{w}_1$ :  $m_4, m_1, m_3, m_2$

$\mathbf{w}_2$ :  $m_1, m_3, m_2, m_4$

$\mathbf{w}_3$ :  $m_1, m_3, m_4, m_2$

$\mathbf{w}_4$ :  $m_3, m_1, m_2, m_4$

## Problem 2 – True/False

Determine if the following statements are true or false:

- a) True or false? In every instance of the stable matching problem, there is a stable matching containing a pair  $(m,w)$  such that  $m$  is ranked first on the preference list of  $w$  and  $w$  is ranked first on the preference list of  $m$ .

Work on this problem with the people around you, and then we'll go over it together!

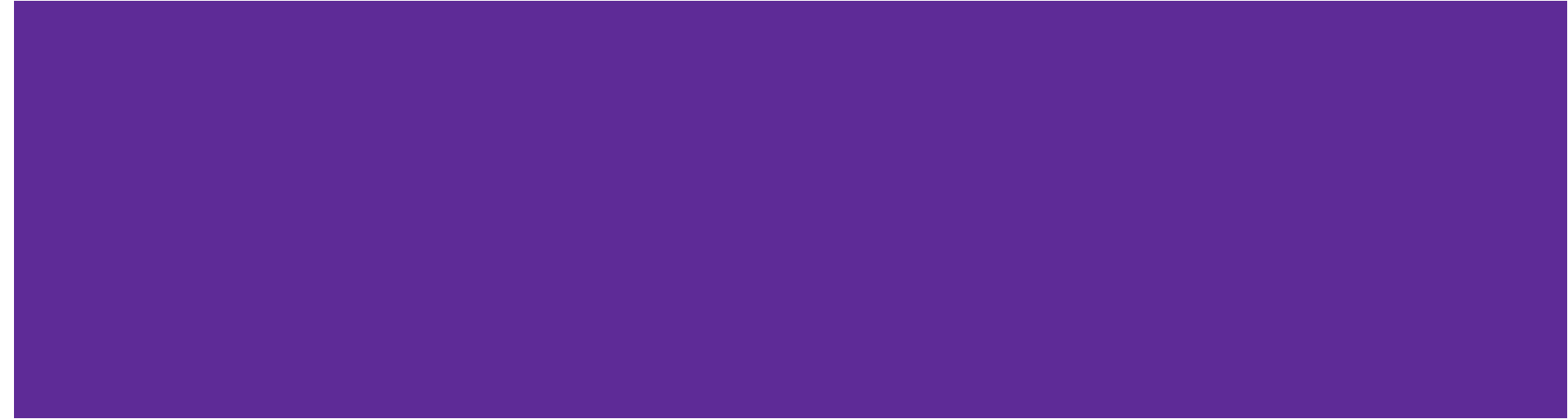
## Problem 2 – True/False

Determine if the following statements are true or false:

- b) Consider an instance of the stable matching problem in which there exists a man  $m$  and a woman  $w$  such that  $m$  is ranked first on the preference list of  $w$  and  $w$  is ranked first on the preference list of  $m$ . Then in every stable matching  $S$  for this instance, the pair  $(m,w)$  belong to  $S$ .

Work on this problem with the people around you, and then we'll go over it together!

# Induction



# Induction

- You will be writing induction proofs in this class.
- The style requirements for proofs in this class are less stringent than the style requirements from 311.

# Induction Template

Let  $P(n)$  be “(whatever you’re trying to prove)”.

We show  $P(n)$  holds for all  $n$  by induction on  $n$ .

Base Case: Show  $P(b)$  is true.

Inductive Hypothesis: Suppose  $P(k)$  holds for an arbitrary  $k \geq b$

Inductive Step: Show  $P(k + 1)$  (i.e. get  $P(k) \rightarrow P(k + 1)$ )

Conclusion: Therefore,  $P(n)$  holds for all  $n$  by the principle of induction.



## Problem 3 – Induction Review

Consider the following claim:

Let  $P(n)$  be “Every tree with  $n$  nodes has  $n - 1$  edges.”

- a) What is the correct “skeleton” of the inductive step (i.e., the right things to assume and the right target)?
  
- b) Prove the claim by induction.

## Problem 3 – Induction Review

Consider the following claim:

Let  $P(n)$  be “Every tree with  $n$  nodes has  $n - 1$  edges.”

- a) What is the correct “skeleton” of the inductive step (i.e., the right things to assume and the right target)?

Work on this problem with the people around you, and then we'll go over it together!

# Problem 3 – Induction Review

Consider the following claim:

Let  $P(n)$  be “Every tree with  $n$  nodes has  $n - 1$  edges.”

- a) What is the correct “skeleton” of the inductive step (i.e., the right things to assume and the right target)?

# KEY Induction Concept

It might be really tempting to structure the inductive step of this problem as something like, “start with an arbitrary tree  $T$  of size  $k$  nodes, and then add a node to it, making tree  $T'$  with  $k + 1$  nodes.”

This is a **BAD** idea! Then we'd have to cover every possible way to add on a node (and prove that we had actually dealt with every possible case), making the overall proof way more complicated and unwieldy.

Instead, we **ALWAYS** want to start with the bigger thing (in this case, with the arbitrary tree  $T'$  of size  $k + 1$ ) and find the smaller thing inside of it.

## Problem 3 – Induction Review

Consider the following claim:

Let  $P(n)$  be “Every tree with  $n$  nodes has  $n - 1$  edges.”

Prove the claim by induction.

Work on this problem with the people around you, and then we'll go over it together!

**Proof or Counterexample?**



# Prove or Disprove?

Often, you will be given a statement, and then asked to either prove or disprove it. This can be stressful! How do you know which you should start with?

The best way to begin, especially when you don't know if the claim is even true, is to try to understand it better by producing some examples. This has two main benefits that will help, whether you end up proving or disproving the claim:

- 1) You get a better understanding of the statement so now you have a clear method of approach, OR
- 2) You find a counterexample, which allows you to easily write a quick proof that the statement is false!

## Problem 2 – A Quick Proof

Is it possible to have a stable matching instance with more than 2 stable matchings? If so, give an instance and at least 3 stable matchings. If not, prove that every instance has at most 2 stable matchings.

Work on this problem with the people around you, and then we'll go over it together!



## Problem 2 – A Quick Proof

Is it possible to have a stable matching instance with more than 2 stable matchings? If so, give an instance and at least 3 stable matchings. If not, prove that every instance has at most 2 stable matchings.

# **That's All, Folks!**

**Thanks for coming to section this week!**  
**Any questions?**