Platforms & Tools

D2L (DEN): Syllabus
Link to office hours calendar
Pre-Lecture Notes
Lecture Notes
Lecture Videos
HW Assignments
Any other reference material

Piazza : Discussion Board
Announcements

Gradescope : HW Submissions
Exams
Grading issues

Roles & Responsibilities

- Instructor — lectures, Discussions, OHs
- TAs — Grading, OHs
- CS Dept. Advisors — D_Clearances, etc.
- DEN Support — D2L issues

Textbook

Algorithm Design, by John Kleinberg & Eva Tardos

Supplemental Textbook

Introduction to Algorithms, 3rd edition, by Cormen et al.

Your Responsibilities

- Attending lectures and Discussions
- Completing reading assignments
- Completing HW assignments
- Doing as many other practice problems as possible
- Taking exams

Your Grade

- HW assignments	12%
- Midterm Exam 1	30%
- Midterm Exam 2	30%
- Midterm Exam 3	23%
- Final Project	5%

Grading Scale

90_100	A	60-64.99	C^+
86_89.99	A^-	55_59.99	C
80_85.99	B^+	50_54.99	C^-
70_79.99	B	45_49.99	D
65_69.99	B^-	Below 44.99	F

Instructor reserves the right to lower some or all grade boundaries at the end of the semester.

Prerequisites

- Discrete Math
  - Proof Methods: Mathematical Induction, Proof by Contradiction, etc.
  - Sorting Methods
  - Graph Theory Basics: Directed vs Undirected, Trees, Paths, Cycles, Connectivity, Adjacency List/Matrix, DAG, Topological Ordering, etc.
  - Graph Search Algorithms: BFS, DFS
  - Asymptotic Notation
- Data Structures
  - Basic data structures: Arrays, Linked Lists, Stacks, Queues

High Level Syllabus

- Introduction
- Review of some prerequisite topics
- Major algorithmic techniques
  - Greedy
  - Divide & Conquer
  - Dynamic Programming ← Exam 1

- Network Flow
- Computational Complexity Theory ← Exam 2
- Approximation Algorithms
- Linear Programming ← Exam 3

Corrections to TEDx Talk

1_ "An Algorithm is a set of instructions in machine language."

Algorithms have existed long before digital computers came about. The root of the word "algorithm" goes back to Kharazmi -- An Iranian mathematician (~780-850).

2_ "... As algorithmic science advanced on Wall Street..."

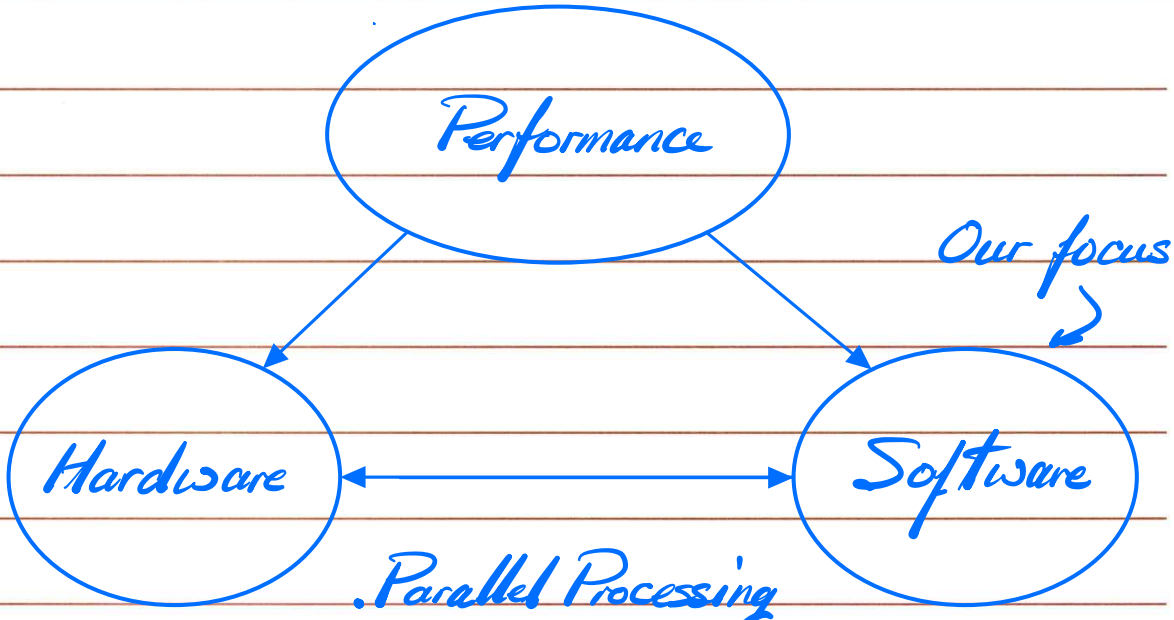
Algorithmic science as it relates to digital computing first advanced in engineering, mathematics and physics fields.

3_ "... invites 6 million algorithms for a listen..."

???

Key attributes of an algorithm:

- Correctness
- Performance



When studying a problem, we go through the following steps:

1- Come up with a concise problem statement

2- Present a solution

3- Prove the correctness of your solution

4- Analyse its performance (complexity analysis)

Stable Matching

Problem: How do we match n men with n women so they stay together ever after?

Step 1: Come up with a concise problem statement.

We have a set of n men,

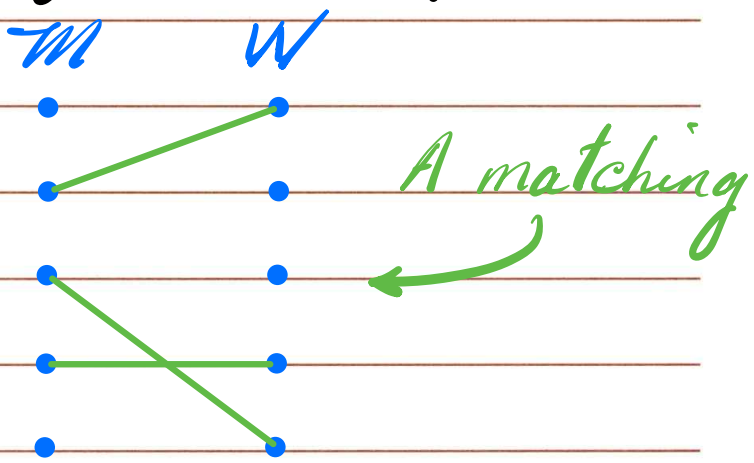
$$M = \{m_1, m_2, \ldots, m_n\}$$

We have a set of n women,

$$W = \{w_1, w_2, \ldots, w_n\}$$

Step 1 (cont'd):

Def. A matching S is a set of ordered pairs.



Def. A perfect matching S' is a matching with the property that each member of M and each member of W appear in exactly one pair in S'.

Step 1 (cont'd): Add notion of preferences

Each man $m \in M$ ranks all women.

- m prefers w to w' if m ranks w higher than w'.
- Ordered ranking of m is his preference list.

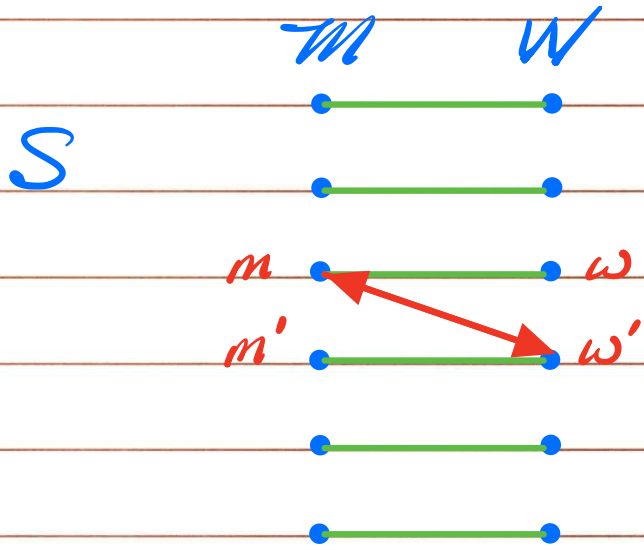
$$P_{m_i} = \{w_{i_1}, w_{i_2}, \ldots, w_{i_n}\}$$

Similarly, each woman w ranks all men.

$$P_{w_j} = \{m_{j_1}, m_{j_2}, \ldots, m_{j_n}\}$$

Step 1 (cont'd): Define instability

Consider below matching S.



If (m, w) and (m', w') are ordered pairs in S, and

- m prefers w' over w
- w' prefers m over m'

Then the pair (m, w') is called an instability WRT S.

Step 1 (cont'd): Define stable matching

Def. Matching S is stable if

1. It is a perfect matching.
2. There are no instabilities WRT S.

Step 1 (cont'd): Define input and expected output

Input: Preference lists for a set of n men and n women.

Output: A set of n marriages (ordered pairs) with no instabilities.

Step 2: Present a solution.

(Gale-Shapley algorithm)

Initially all $m \in M$ and $w \in W$ are free
While there is a free man
  Choose a free man
  Let w be m's highest-ranked woman
  to whom he has not yet proposed
  If w is free then
    (m, w) become engaged

  Else, say w is engaged to m'
    If w prefers m' to m then
      m remains free
    Else
      (m, w) become engaged
      m' becomes free
    End if
  End if
End while
Return the set of engaged pairs

Step 3 : Prove the correctness of this solution.

1- Prove that the while loop terminates

2- Prove that at termination we have a perfect matching

3- Prove that there are no instabilities in our perfect matching

1- We will prove that the while loop terminates in at most n^2 iterations

Observations:

a- Each iteration of the while loop consists of a free man proposing to a woman.

b- After her first engagement, a woman will always stay engaged.

Claim: It is impossible for a man to be free after having proposed to all n women.

Proof:
Assume a man has become free after having proposed to all n women. Based on observation b, all n women must be engaged at that time. But since there are only $n-1$ other men, it is impossible for all n women to be engaged.

So, based on observation a, each man can contribute to at most n iterations of the while loop. And since there are only n men, then the maximum number of iterations in the while loop will be n^2.

2- Prove that at termination we have a perfect matching

While loop terminates when there are no free men, in other words, every man is engaged to a woman. And since there are an equal number of men and women, this will be a perfect matching.

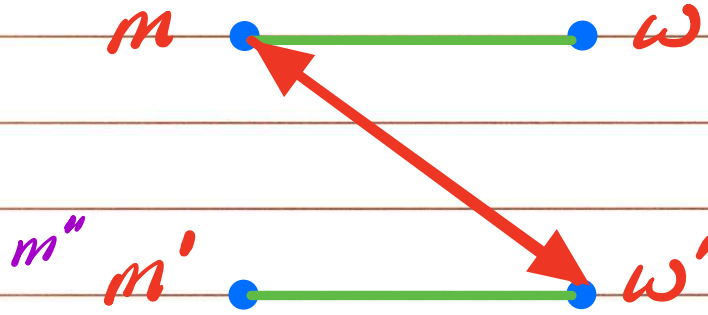
3. We will prove (by contradiction) that there are no instabilities in our perfect matching

First, some observations:

1. From the woman's perspective, she starts single, and once she gets engaged, she can only get into better engagements.

2. From the man's perspective, he starts single, and once he gets engaged, he might get rejected (repeatedly), only to settle for a lower ranking woman each time.

Proof: We will assume that an instability exists in our the perfect matching produced by the Gale-Shapley algorithm, involving two pairs (m,w) and (m',w').

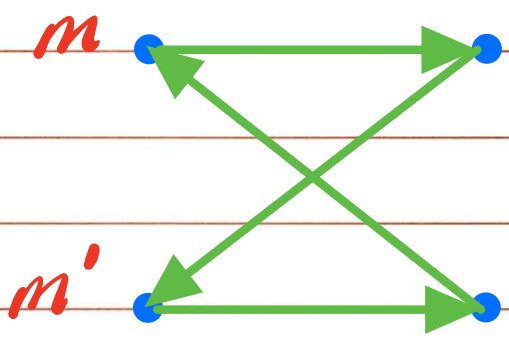


Q: Did m propose to w' at some point in the execution?

If no, then w must be higher than w' on his list → contradiction!

If yes, he must have been rejected in favor of m'' and due to observation 1, either $m''=m'$ or m' is better than m'' → contradiction!

Are stable matchings unique?



arrows indicate preferences

Women proposing: (m, w'), (m', w)

Men proposing: (m, w), (m', w')

In general, there can be many stable matchings given a set of preference lists for men and women. Gale-Shapley can find at most two stable matchings, one with men proposing (man preferred) and one with women proposing (woman preferred).

Step 4: Complexity Analysis

Operations involved in each iteration:

1- Identify a free man

2- For a man m, identify the highest ranked woman to whom he has not yet proposed.

3- For a woman w, decide if w is engaged, and if so to whom.

4- For a woman w and two men m and m', decide which man is preferred by w.

5- Place a man back in the list of free men.

1- Identify a free man

	Get	Put
Stack implementation	O(1)	O(1)
Queue "	O(1)	O(1)
Array "	O(1)	O(1)
Linked list "	O(1)	O(1)

2- Identify the highest ranked woman to whom m has not yet proposed

Keep an array Next[1..n] where Next[m] points to the position of the next woman on m's preference list that m will be proposing to

Given the men's preference lists: ManPref[1..n, 1..n], where ManPref[m,i] denotes the i^{th} woman on man m's preference list, and the Next array, we can find the next woman w to whom m will be proposing to by:

w = ManPref[m, Next[m]]

takes O(1)

3_ Determine woman w's status

Keep an array called Current[1..n] where Current[w] is Null if w is single, and set to m if w is engaged to m.

Checking Current[w] takes O(1)

4- Determine which man is preferred by w.

WomanPref[i]

3	8	4	31	1	...	← men's ID's
1	2	3	4	5	← Rankings	

to speed up this operation will need to construct a WomanRanking array:

WomanRanking[i]

5		1	3	...	← Rankings
1	2	3	4	5	← men's ID's

Preparation (before entering GS iterations)

Create a WomanRanking array where WomanRanking[w, m] contains the rank of man m based on w's preference.

Preparation takes $O(n^2)$ Time
GS iterations take $O(n^2)$ Time

Overall time complexity $O(n^2)$ Time

Def. Woman w is a valid partner of man m if there exists a stable matching that contains the pair (m, w)

Consider the preference list for man m_i

$$P_{m_i} = \{ w_{i_1}, w_{i_2}, \ldots, w_{i_k}, \ldots, w_{i_m}, \ldots, w_{i_\ell}, \ldots, w_{i_n} \}$$

if w_{i_k}, w_{i_m}, and w_{i_ℓ} are m_i's only valid partners, then

- w_{i_k} is m_i's best valid partner
- w_{i_ℓ} is m_i's worst valid partner

Discussion 1

1. Prove that every execution of the G-S algorithm (when men are proposing) results in the same stable matching regardless of the order in which men propose.

2. Prove that when we run the G-S algorithm with men proposing, women end up with their worst valid partners.

3. True or False:
   In every stable matching that Gale–Shapley algorithm may end up with when men propose, there is a man who is matched to his highest-ranked woman.

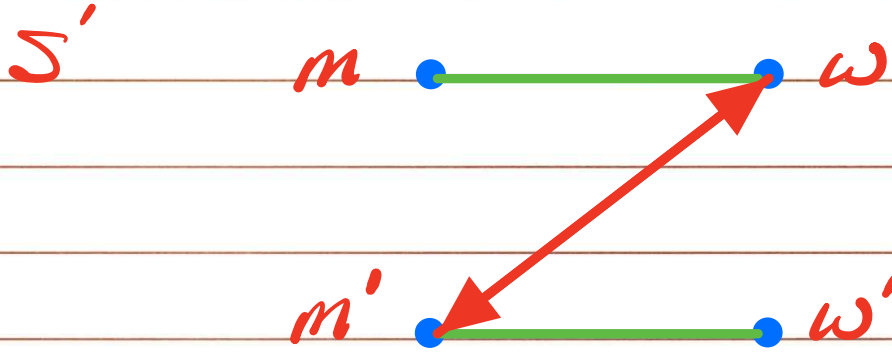
4. In a connected bipartite graph, is the bipartition unique? Justify your answer.

1. Prove that every execution of the G-S algorithm (when men are proposing) results in the same stable matching regardless of the order in which men propose.

Plan: To prove this, we will show that when men propose, they always end up with their best valid partner.

Proof (by contradiction):

Say m is the first man rejected by a valid partner w in favor of m'. Then consider a stable matching S' where (m, w) is a pair.



(m', w) will be an instability WRT S' since

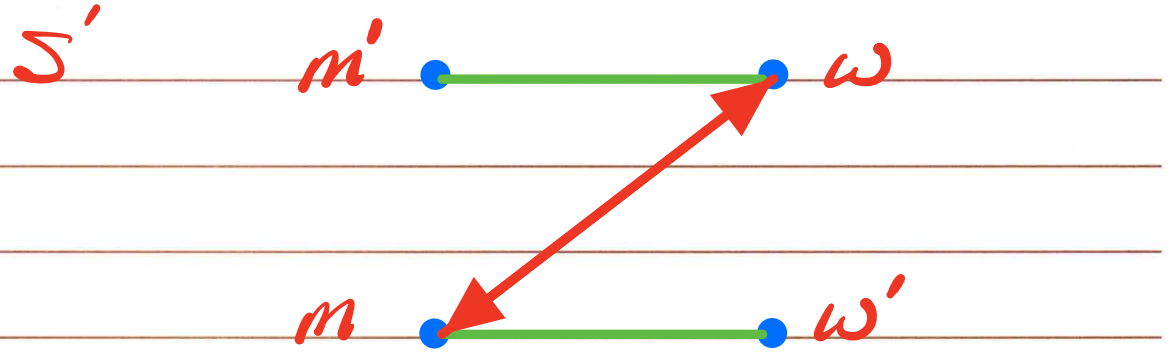
- w prefers m' over m
- m' prefers w over w' (since he first proposed to w before w')

For more details refer to textbook.

2. Prove that when we run the G-S algorithm with men proposing, women end up with their worst valid partners.

Proof (by contradiction):

Suppose we end up with a matching S where for a pair (m, w) in S, m is not w's worst valid partner.
So, there must be another matching S', where w is paired with a man m' whom she likes even less.



(m, w) will be an instability WRT S' since
- w prefers m over m'
- m prefers w over w' (since w is his best valid partner)

3. True or False:
In every stable matching that Gale–Shapley algorithm may end up with when men propose, there is a man who is matched to his highest-ranked woman.

Here is a counterexample:

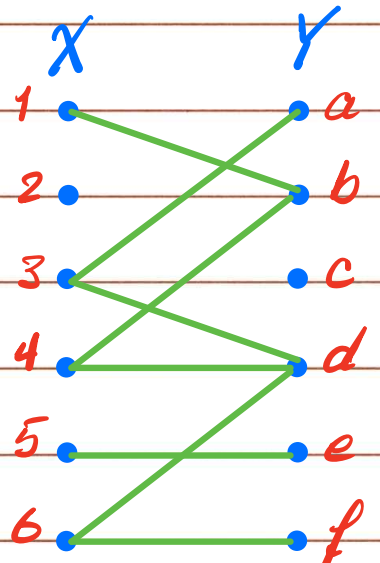
m_1	m_2	m_3	w_1	w_2	w_3
w_1	w_1	w_2	m_3	m_2	m_1
w_2	w_2	w_1	m_2	m_1	m_2
w_3	w_3	w_3	m_1	m_3	m_3

With men proposing, we end up with

(m_1, w_3), (m_2, w_2), and (m_3, w_1)

All men are rejected by their highest-ranked woman.

4. In a connected bipartite graph, is the bipartition unique? Justify your answer.



Bipartite but Not a connected! So, it is possible to find more than one bipartition in this graph: $X' = \{1,2,c,3,4,e,6\}$, $Y' = \{a,b,d,5,f\}$

However, for a connected bipartite graph:

This is True. We can prove this by contradiction. Let's say we have already found the bipartition X,Y where all edges in the graph go between X and Y. Now let's assume that we have another bipartition X', Y' different from X, Y. If X', Y' is different from X, Y, then there must be at least one node *i* that used to be in X and now is not in X' but has moved into Y'. We can run a BFS starting from node *i*. Say *i* is at level 1 in the BFS tree. All nodes at level 2 that must have belonged to Y should now be in X', and all nodes at level 3 which must have belonged to X should now be in Y', and so on. In other words, all node that used be in X are now in Y' and all nodes that were in Y are now in X'. So X', Y' is not different from X, Y.