Homework 1

● Graded

Student

Abhishek Soundalgekar

Total Points

50 / 50 pts

Question 1

Q1 **5** / 5 pts

+ 0 pts Incorrect

✔ **+ 2 pts** Correctly state n - 1

✔ **+ 3 pts** ****Provides a correct example.

Question 2

Q2 **5** / 5 pts

+ 0 pts Incorrect

✔ **+ 2 pts** Correctly identify the statement as true.

✔ **+ 3 pts** Provide a correct explanation or example.

+ 2 pts Correct explanation but wrong final answer

nice

Question 3

Q3 **5** / 5 pts

+ 0 pts Incorrect

✔ **+ 1 pt** Correct claim that there are configurations of shows without stable matching

✔ **+ 4 pts** Provides a correct counterexample

Question 4

Q4 **15** / 15 pts

+ 0 pts Incorrect

✔ **+ 8 pts** Algorithm
1pt: Loop condition (line 1)
2 pts: hospitals offer the next highest-ranked student (line 2)
2pts: case that s is free (lines 3-4)
3pts: cases if s is at another h'

✔ **+ 1 pt** Proof
1 pt: Algorithm terminates in finite steps (optional to mention in O(mn) steps)

✔ **+ 2 pts** Proof
2pts: All positions get filled

✔ **+ 2 pts** Proof
2pts: Explain why no instability of the first type

✔ **+ 2 pts** Proof
2pts: Explain why no instability of the second type

Question 5

Q5 **10** / 10 pts

+ 0 pts Incorrect

✔ **+ 4 pts** Proof that stable matching exists

✔ **+ 6 pts** Explain that this stable matching is unique

Question 6

Q6 **10** / 10 pts

+ 0 pts Incorrect

✔ **+ 4 pts** Correct '=>' proof
(a) 1pt: Correctly state forward claim
(b) 3pt: Correctly justify forward claim

✔ **+ 6 pts** Correct '<=' proof
(a) 1pt: Correctly state backwards claim
(b) 5pts: Correctly justify backwards claim

+ 10 pts Correct True/False

+ 0 pts Click here to replace this description.

Question assigned to the following page: 1

Homework 1
Analysis of Algorithm

Name: Abhishek Sowndalgekar
USCID: 2089011000

Question 1

Answer 1

Given: G-S algorithm for n men and n women.
To Find: Maximum number of times a man may be rejected as a function of n.
Give an example where this happens.

The Gale-Shapley algorithm is used to find a stable matching between two groups, typically men and women in the classic setup. Each man proposes to women in order of his preference, and each woman either accepts a proposal if she's single or compares the proposing man to her current fiance, choosing the one she prefers more.

In G-S algorithm for n men and n women, to determine the maximum number of times a man may be rejected we consider the worst case - scenario. Each man proposes to women in order of his preference, and each

Question assigned to the following page: 1

woman either accepts or rejects a proposal based on her current engagement and preference list

[A man can't be rejected by all n women]

→ worst case scenario:

The maximum number of rejections a man can face is $n-1$, as he will be accepted by the n^{th} women when rejected by the first $n-1$.

[Maximum number of rejects = $n-1$]

Example: I take 5 men (M_1, M_2, M_3, M_4, M_5) and 5 women (W_1, W_2, W_3, W_4, W_5)

Preference

M_1	M_2	M_3	M_4	M_5
W_1	W_1	W_1	W_1	W_1
W_2	W_2	W_2	W_2	W_2
W_3	W_3	W_3	W_3	W_3
W_4	W_4	W_4	W_4	W_4
W_5	W_5	W_5	W_5	W_5

All men rank women identically

W_1	W_2	W_3	W_4	W_5
M_5	M_5	M_5	M_5	M_5
M_4	M_4	M_4	M_4	M_4
M_3	M_3	M_3	M_3	M_3
M_2	M_2	M_2	M_2	M_2
M_1	M_1	M_1	M_1	M_1

All women rank men in reverse order

Question assigned to the following page: 1

→ G S Algorithm Executions :

1] Round 1 :
All men propose to W_1.
W_1 accepts M_5 (top choice of W_1) and rejects M_1, M_2, M_3, M_4.
Men Rejected by W_1 : M_1, M_2, M_3, M_4

2] Round 2 :
Rejected men (M_1, M_2, M_3, M_4) propose to W_2.
W_2 accepts M_4 (her top remaining choice) and rejects M_1, M_2, M_3.
Men Rejected by W_2 : M_1, M_2, M_3

3] Round 3 :
Rejected men (M_1, M_2, M_3) propose to W_3.
W_3 accepts M_3 and rejects M_1, M_2.
Men Rejected by W_3 : M_1, M_2

4] Round 4 :
Rejected men (M_1, M_2) propose to W_4.
W_4 accepts M_2 and rejects M_1.
Men rejected by W_4 : M_1

5] Round 5 : M_1 proposes to W_5.
W_5 has no other proposals and accepts M_1.

Question assigned to the following page: 1

The final matching

$M_5 \leftrightarrow W_1$, $M_4 \leftrightarrow W_2$, $M_3 \leftrightarrow W_3$, $M_2 \leftrightarrow W_4$, $M_1 \leftrightarrow W_5$

In this example we can see that M_1 is rejected by all women except W_5 which is a total of 4 rejections.

This happened because women's preference favoured higher-indexed men ($M_5 > M_4 > \dots > M_1$) causing lower-ranked men to cascade down the preference list.

The algorithm terminates in 5 rounds with M_1 being the last to be matched.

Max rejections M_1 faced $\Rightarrow 4 = n - 1$.

Question assigned to the following page: 2

Question 2

Answer 2

The given statement is True.
The statement would result in true when the most preferred woman for each man is different.
An explation using example:
Lets consider $n = 3$.

Men's preferences	Women's preferences
$M_1 : W_1 > W_2 > W_3$	$W_1 : M_1 > M_2 > M_3$
$M_2 : W_2 > W_1 > W_3$	$W_2 : M_2 > M_1 > M_3$
$M_3 : W_3 > W_1 > W_2$	$W_3 : M_3 > M_1 > M_2$

All men propose to their first-choice women ($M_1 \to W_1$, $M_2 \to W_2$, $M_3 \to W_3$). And each woman accepts her first choice man.

The resulting stable matching is (M_1, W_1) (M_2, W_2) (M_3, W_3), where every man is matched with his most preferred woman.
This example ensures that no conflicts arise and the matching is stable. Hence for all $n \geq 2$, such a preference set exists.

Hence the given statement is True.

Question assigned to the following page: 3

Question 3

Answer 3

We resolve this question by solving the 'b' part. Giving an example of a set of TV shows and associated ratings for which there is no stable pair of schedules.

Let's take the example with $n = 6$

A's ratings : [12, 10, 8, 6, 4, 2]

B's ratings : [11, 9, 7, 5, 3, 1]

[A network wins a slot if higher rating. A stable pair would mean neither network can unilaterally rearrange schedule to win]

1. First schedule (both in descending order) full season.

S-Net. A : [12, 10, 8, 6, 4, 2]

T-Net. B : [11, 9, 7, 5, 3, 1]

Slotting :

Slot 1 : A wins (As $12 > 11$)

Slot 2 : A wins ($10 > 9$)

Slot 3 : A wins ($8 > 7$)

Slot 4 : A wins ($6 > 5$)

Slot 5 : A wins ($4 > 3$)

Slot 6 : A wins ($2 > 1$)

Question assigned to the following page: 3

2. Instability arises when B reorders T schedule. Let's reorder T in ascending order.

S - Network A : [12, 10, 8, 6, 4, 2] (no change)
T - Network B : [1, 3, 5, 7, 9, 11] (ascending)

slotting :

slot 1 : A wins (12 > 1)
slot 2 : A wins (10 > 3)
slot 3 : A wins (8 > 5)
slot 4 : B wins (6 < 7)
slot 5 : B wins (4 < 9)
slot 6 : B wins (2 < 11)

A wins 3 slots & B wins 3 slots.

3. Network A can retaliate by reorderering its schedule to win more slots

S - Net. A : [2, 4, 6, 8, 10, 12] (reorder ascending)
T - Net B : [1, 3, 5, 7, 9, 11] (no change)
slotting : slot 1 : A wins
slot 2 : A wins
slot 3 : A wins
slot 4 : A wins
slot 5 : A wins
slot 6 : A wins

Question assigned to the following page: 3

A wins all 6 slots.
To this Network B will modify T schedule back to its original descending order, restarting the cycle and causing instability.

Thus this example shows there is no stable pair of schedules.

Question assigned to the following page: 4

Question 4

Answer 4

We can consider hospitals in the place of men and students in the place of women like in the G.S. algorithm and the solution given by this is a stable matching can be shown.

As stated in the problem statement each hospital offers a position to a student at most once, and in each iteration hospital offers a position to some student so this algorithm terminates in $O(mn)$ steps where m is the number of hospitals and n is the number of students.

Let there be 'V' vacancies available at hospital H. The algorithm would terminate with all the vacancies filled because any hospital that did not fill all of its vacancies must give to every student. Student who rejected must be committed to some other hospital H'.
If H still has vacancies it would mean all vacancies is greater than n (no. of students). This contradicts the assumption given, meaning that all the vacancies get filled. Thus the algo. would terminate giving a matching M for any preference list.

Question assigned to the following page: 4

ALGORITHM:

```
1  for each hospital h:
2      track quota Qh (no. of positions to fill)
3      track pointer Ph (index of next student in h's list)

4  Loop until No Proposals Possible:
5      flag exist = False
6      for each hospital h in fixed order:
7          if Qh > 0 and Ph ≤ length: (h's preference list)
8          Let s be the student at Ph in h's list
9          Increment Ph by 1 (move to next student)
10         if s is unassigned: Assign s to h
11         decrement Qh by 1
12         set exist = True
13         else let h' be hospital where s is assigned
14             if s prefers h over h':
15                 unassign s from h'
16                 Assign s to h, decrement Qh by 1
17                 Increment Qh' by 1 (free up spot at h')
18                 Set exist = True
19         else: skip h (no more student or quota filled)
20     if exist = False:
21         Break the loop (terminate)
```

Question assigned to the following page: 4

The condition for termination would be:
→ All hospitals have either filled their quotas or exhausted their preference lists
→ Unassigned students remain unmatched.

Stability:
→ Each hospital proposes to each student at most once. Terminates in $O(mn)$ total steps.
→ All positions are filled as hospitals continue proposing until quotas are filled or preference lists are exhausted. Surplus of students ensures no hospital with unfilled quota remains
→ No instabilities:
First type: If h prefers unassigned s' over assigned s, h would have proposed to s' first. First type instability (a student s' was preferenced over s by h but not admitted), then h must have offered s before s' who wasn't offered, this is a contradiction because h prefers s' to s. Thus not first type instability.
Second type: If s' prefers h over current h', h would have 'poached' s' during proposal.
Second type instability where s & s' currently at h & h' have a swap beneficial to s' & h, then s' should not be admitted by h when it considered it before s, meant s' prefers h' to h leading to a contradiction.
Instability not of second type.
Thus the matching is stable & at least one S.M. exists.

Question assigned to the following page: 5

Question 5

Answer 5

Let's prove the unique stable matching using the matching S where m_i is matched to w_i for all $i = 1, 2 \ldots N$. To denote matching S, a is matched to b we will use the notation $S(a) = b$. S is a matching because every man is paired with exactly one woman and vice-versa.

If m_j prefers w_k to w_j where $k < j$, then such a higher ranked w_k prefers encounter m_j. Thus no instabilities & S is a stable matching.

Stability of S:

Assume a blocking pair (m_k, w_e) exists in S
→ m_k prefers w_e over w_k, implying $l < k$
(since m_k prefers w_k over all w_j with $j > k$)
→ w_e prefers m_k over m_e, implying $k < l$
(since w_e prefers m_e over all m_j with $j > e$)
This creates a contradiction: $l < k$ & $k < l$ can't both hold. Thus S is stable

Question assigned to the following page: 5

Uniqueness:

To prove that the stable matching is unique we use contradiction, let's assume another stable matching S' which is different from S.

In S', there must be at least one pair (m_k, w_l) with $k \neq l$.

Case 1: $l > k$

→ w_l prefers m_l over m_k (by her order)

→ m_k prefers w_k over w_l (by his order)

→ If w_k is not matched to m_k in S', (m_k, w_k) forms a blocking pair.

Case 2: $l < k$

→ m_k prefers w_k over w_l

→ w_l prefers m_l over m_k

→ If m_l is not matched to w_l in S', (m_l, w_l) forms a blocking pair

In both cases S' can't be stable. Hence S is the only stable matching.

Question assigned to the following page: 6

Question 6

Answer 6

→ The given statement is True

→ If both algorithms yield the same matching:
- The male-optimal (male-proposing) and female-optimal (female-proposing) matchings coincide.
- This implies no other stable matching exists, as optimality for both genders can only occur if there is a single stable matching.

→ If there is a unique stable matching
- Both algorithms must produce it, as there are no alternatives
- The male-proposing & female-proposing versions can't diverge because no other stable pairs exist.

Question assigned to the following page: 6

→ Contrapositive :

If multiple stable matchings exist, the male-proposing and female-proposing algorithms will produce different results (male-optimal v/s female optimal).

Thus the statement holds true

A unique stable matching exists if and only if both versions of the male-shapley algorithms produce the same matching

———— / ———— / ————