Question 1.

Let f(n)= n^4 log n+ $2n^6$ $+10^{10}$. Bound the growth of function f(n) asymptotically. Can you come up with a tight bound? You should clearly justify your answer here (use formal proof).

Answer:

Given, f(n)= n^4 log n+ $2n^6$ $+10^{10}$

Let, g(n) = n^6 .

f(n) = Theta(g(n)) if there exists C1, C2, n0 where C1 * g(n) <= f(n) <= C2 * g(n) for all n >= n0

Let's assume, f(n) = Theta(g(n)) = Theta(n^6)

Then, C1 * n^6 <= f(n) <= C2 * n^6 for all n >= n0. We need to find C1, C2, n0.

To find C2, lets replace all values in f(n) with multiples of n^6 .

i.e; n^6 + $2n^6$ $+10^{10}$ * n^6 .

We can say, f(n) <= n^6 + $2n^6$ $+10^{10}$ * n^6 since,

n^4 log n <= n^6 for all n >= 1

$2n^6$ <= $2n^6$ for all n >= 1

10^{10} <= 10^{10} * n^6 for all n >= 1

If we add all the above 3, we get n^4 log n+ $2n^6$ $+10^{10}$ <= n^6 + $2n^6$ $+10^{10}$ * n^6 for all n >= 1

=== f(n) <= (3 + 10^{10}) * n^6 for all n >= 1

So, C2 = (3 + 10^{10})

To find C1, lets just consider only n^6 .

We can say f(n) >= n^6 for all n >= 1 since,

n^6 <= 2 * n^6 for all n >= 1

So, n^6 <= 2 * n^6 + n^4 log n for all n >= 1

And, n^6 <= 2 * n^6 + n^4 log n $+10^{10}$ for all n >= 1

So, n^6 <= 2 * n^6 + n^4 log n $+10^{10}$ for all n >= 1

=== n^6 <= f(n) for all n >= 1

So, C1 = 1

Therefore, 1 * n^6 <= f(n) <= (3 + 10^{10}) * n^6 for n >= 1

According to Tight Bound Theorem, f(n) = Theta(g(n)) if we have C1, C2, n0 where C1 * g(n) <= f(n) <= C2 * g(n) for all n >= n0

Therefore, f(n) = Theta(n^6) for all n >= 1 where C1 = 1 and C2 = (3 + 10^{10})

Question 2. Consider an instance of the stable matching problem with 4 men and 4 women. Provide their preference lists as you would like (show them using two tables as in the slides).

- a) Then provide an assignment that is *unstable*.
- b) Now provide an assignment that is *stable*.

For each case above, explain in detail why the assignment is stable/unstable.

Answer:

Let A, B, C, D are men and W, X, Y, Z are women.

The preference list of Men is

	I	II	III	IV

A	W	X	Y	Z
B	X	Y	Z	W
C	Y	Z	W	X
D	Z	W	Y	X

The preference list of Women is

	I	II	III	IV
W	A	B	C	D
X	B	C	D	A
Y	D	A	B	C
Z	C	B	D	A

a) One Unstable pair is A - X, B - W, C - Y, D - Z

Explanation:

In the given matching we have a pair A - W. A is paired with X but he prefers W to X. Also, W is paired with B but she prefers A to B. We can say that a matching is stable if we have a perfect matching with no unstable pairs. Since we have an unstable pair A - W, the given pair is an Unstable Pair.

b) One Stable pair is A - W, B - X, C - Y, D - Z

Explanation:

To prove that this pair is stable, we should have perfect matching with no unstable pairs. Since, each men is paired with only one women and each women is paired with only one men, this assignment is a perfect matching.

To prove we have no unstable pairs let's consider all the possible pairs other than the given matching and prove that it is not unstable.

A - X: It is not unstable, since A doesn't prefer X to W and also X doesn't prefer A to B.
A - Y: It is not unstable, since A doesn't prefer Y to W even though Y prefer A over C.
A - Z: It is not unstable, since A doesn't prefer Z to W and also Z doesn't prefer A to D.
B - W: It is not unstable, since B doesn't prefer W to X and also W doesn't prefer B to A.
B - Y: It is not unstable, since B doesn't prefer Y to X even though Y prefer B over C.
B - Z: It is not unstable, since B doesn't prefer Z to X even though Z prefer B over D.
C - W: It is not unstable, since C doesn't prefer W to Y and W doesn't prefer C to A.
C - X: It is not unstable, since C doesn't prefer X to Y and X doesn't prefer C to B.
C - Z: It is not unstable, since C doesn't prefer Z to Y even though Z prefer C to D.
D - W: It is not unstable, since D doesn't prefer W to Z and W doesn't prefer D to A.
D - X: It is not unstable, since D doesn't prefer X to Z and X doesn't prefer D to B.
D - Y: It is not unstable, since D doesn't prefer Y to Z even though Y prefer D to C.

Since there is no unstable pair in the given matching pair, the given matching pair is a stable pair.

Question 3.

The interest, naturally, was in finding a way of assigning each student to at most one hospital, in such a way that all available positions in all hospitals were filled. (Since we are assuming a surplus of students, there would be some students who do not get assigned to any hospital.)

We say that an assignment of students to hospitals is *stable* if neither of the following situations arises.

- First type of instability: There are students s and s', and a hospital h, so that
  - s is assigned to h, and
  - s' is assigned to no hospital, and
  - h prefers s' to s.
- Second type of instability: There are students s and s', and hospitals h and h', so that
  - s is assigned to h, and
  - s' is assigned to h', and
  - h prefers s' to s, and
  - s' prefers h to h'.

So we basically have the Stable Matching Problem, except that (i) hospitals generally want more than one resident, and (ii) there is a surplus of medical students.

Show that there is always a stable assignment of students to hospitals, and give an algorithm to find one.

Answer:

The given problem is similar to Stable Matching Problem for N men and N women with some additional constraints.

The differences are:

1) There are some N hospitals and M students.
2) Each hospital require some number of students which is >= 1.
3) There are surplus number of students so that at the end each hospital can fulfil their admission count, but some students will be left without admission at the end.

The Similarities are:

1) Each hospital has preference list of students.
2) Each student has preference list of hospitals.

If we compare this problem to Stable Matching Problem of Men & Women, then

Hospital == Men
Student == Women

An Assignment is Stable if all the following conditions are false:

1) A Hospital wasn't able to admit the required number of students they are looking for.
2) First Type of Instability: There are students s and s` and a hospital h such that
   - s is assigned to h
   - s` is assigned to no hospital
   - h prefers s` to s
3) Second Type of Instability: There are students s and s` and hospitals h and h` such that
   - s is assigned to h
   - s` is assigned to h`
   - h prefers s` to s
   - s` prefers h to h`

There is always a Stable Matching Between Hospitals and Students. We can come up with the algorithm with little tweaks to original GS Algorithm.

Algorithm for this problem is:

```
Initialize each student to be free and each hospital admit count to 0.
while (some hospital hasn't reached their required admit count and hasn't asked all students) {
	Choose such a hospital h
```

s = 1^{st} student on h's list to whom h has not yet asked
if (s is free)
admit s to their hospital h. Increase admit count of that hospital h by 1
else if (s prefers h to his/her admit h`)
s changes admit from hospital h` to h. Increase admit count of h by 1 and decrease admit count of h` by 1
else
s rejects admit from hospital h
}

Similar to Stable Matching Algorithm for Men & Women, this algorithm is also Hospital-Optimal Assignment: Each hospital gets best valid students.

Observation 1: Hospitals asks students in decreasing order of preference

Observation 2: Once a student is admitted, he/she never becomes not admitted; they only "trades up."

Observation 3: Algorithm terminates after all hospital get their required admission count of asked all students.

Proof that algorithm always finds a stable matching:
Claim 1: Hospitals got all the admits they are looking for.
Proof by Contradiction:

- Suppose, a hospital h haven't got all the admits they are looking for upon termination of algorithm.
- Algorithm terminates after all hospital get their required admission count of asked all students according to Observation 3.
- Since there are surplus of students, By Observation 2 a student s who hasn't got admission is not admitted to hospital h only if hospital h hasn't asked student s.
- So, according to Observation 2 and 3, hospital h will definitely admit s and fulfil their admit count.

Claim 2: First Type of Instability hasn't occurred.
First Type of Instability: There are students s and s` and a hospital h such that

- s is assigned to h
- s` is assigned to no hospital
- h prefers s` to s

Proof by Contradiction:

- Suppose, a hospital h admitted s. s` hasn't got admission i.e; student is free and h prefers s` to s.
- s` won't get any admit only if no hospital asked him. A hospital won't ask s` if they got all the admits they want before they reached s` in their preference list since hospitals ask students in decreasing order of preference.
- So, if h prefers s` to s and s` is unmatched then undoubtedly h asks s` before s for admit. So first type of instability will never occur.

Claim 3: Second Type of Instability hasn't occurred.
Second Type of Instability: There are students s and s` and hospitals h and h` such that

- s is assigned to h
- s` is assigned to h`

- h prefers s` to s
- s` prefers h to h`

Proof by Contradiction:

- Suppose, a hospital h and student s prefer each other to s` and h`.
- Then, since hospitals ask student in decrease order of preference, h asks s before s`.
- If s is admitted to h` that means s rejected h's admit either right away or later.
- So, s prefers h` to h since students once assigned only trades-up but won't trades- down. So second type of instability will never occur.

Since An Assignment is Stable if all the following conditions are false:

1) A Hospital wasn't able to admit the required number of students they are looking for.
2) First Type of Instability: There are students s and s` and a hospital h such that
   - s is assigned to h
   - s` is assigned to no hospital
   - h prefers s` to s
3) Second Type of Instability: There are students s and s` and hospitals h and h` such that
   - s is assigned to h
   - s` is assigned to h`
   - h prefers s` to s
   - s` prefers h to h`

And we proved all the above statements are false in above claims 1, 2, and 3. Therefore we can say that algorithm given above always finds a stable matching for a given preference lists of hospitals and students.

Question 4. Let f(n)= 3 log n^5+ n^2 +14, then f(n) is O(?). Justify.

Answer:

f(n) = 3 log n^5+ n^2 +14.

= 15logn + n^2 + 14 [Since, $\log a^b$ = b * loga]

According to Upper Bound Definition, f(n) = O(g(n)), if there exists a positive constants C, n0 such that f(n) <= C * g(n) for all n >= n0. We need to find those C, n0 where f(n) becomes <= C * g(n).

Let g(n) = $30n^2$

We can write it as g(n) = $15n^2$ + n^2 + $14n^2$

And f(n) = 15logn + n^2 + 14

If we compare f(n) and g(n),

a) 15logn in f(n) will always be < $15n^2$ in g(n) since logn is always < n for all n >= 1

b) n^2 in f(n) will always be equal to n^2 in f(n) for all n >= 1 since they are the same

c) 14 in f(n) will always be <= 14 n^2 in g(n) since is n^2 >= 1 for all n >= 1.

If we add all the three, we can clearly say that 15logn + n^2 + 14 <= $15n^2$ + n^2 + $14n^2$ for all n >= 1

Which means f(n) <= 30 * n^2 for all n >= 1.

Therefore, according to Upper Bound Definition, f(n) = O(n^2) where C = 30 and n0 = 1.