Homework #1 Solutions

(Due 11:59 PM Oct. 11th)

1. Exercise 3, Page 22 (20pts)

- No (10pts)
- Proof by given a counter example (10pts):
    - For example. Assume A has TV shows $\{a_1, a_2\}$ that have rating of $\{30, 50\}$, and B has TV shows $\{b_1, b_2\}$ that have rating of $\{20, 40\}$.
    - If A shows schedule $\{a_1, a_2\}$, B shows $\{b_1, b_2\}$. Then B can win more by changing its schedule to be $\{b_2, b_1\}$.
    - Then, if A shows schedule $\{a_1, a_2\}$, B shows $\{b_2, b_1\}$. Then A can win more by changing its schedule to be $\{a_2, a_1\}$.
    - Then, if A shows schedule $\{a_2, a_1\}$, B shows $\{b_2, b_1\}$. Then B can win more by changing its schedule to be $\{b_1, b_2\}$.
    - Then, if A shows schedule $\{a_2, a_1\}$, B shows $\{b_1, b_2\}$. Then A can win more by changing its schedule to be $\{b_2, b_1\}$.
    - …… In this case, there is no stable pair of schedules, one always wants to change.

2. Exercise 4, on Page 22 (20pts)

- Algorithm (12pts):

```
- While ∈ hospital $h_i$ has available positions
    - $h_i$ offers a position to the next student $s_j$ on its preference list
    - if $s_j$ is free, has not committed to any hospitals
        - $s_j$ accepts the offer
    - else ($s_j$ is already committed to a hospital $h_k$)
        - if $s_j$ prefers $h_k$ than $h_i$:
            - $s_j$ remains with $h_k$, reject $h_i$
        - else ($s_j$ prefers $h_i$ than $h_k$):
            - $s_j$ becomes committed to $h_i$, leave $h_k$.
            - number of available positions at $h_k$ increase by one
            - number of available positions at $h_i$ decrease by one
```

- Proof the assignment is stable (8pts)
    - Prove the first instability cannot exist by proof by contradiction (4pts)
        - Suppose there are student s and s', and a hospital h. If h prefers s' than s, then h would have offered a position to s' prior to s. After that, s' should keep having a position at some hospital h', but not being free (like during the classes, woman will keep pairing with a man but not being along). Contradiction.
    - Prove the second instability cannot exist by proof by contradiction (4pts)
        - Suppose (h, s), (h', s'), and (h, s') is a pair that causes instability. h prefers s' to s means h has proposed to s' prior to s. However, h is not matched with s' anymore, means there was a h'' that is better than h in the preference list of s' $(h'' > h)$. Finally, h' matched with s' (we know when the algorithm end, (h', s') is a pair), and this means $h' > h''$ since the matched hospital of a student can only get better (like the observation during the class that woman's partner always gets better). So, we have $h' > h'' > h$. However, since (h, s') is a pair that causes instability, we should have $(h > h')$. Contradiction.

3. Exercise 6 on page 25 (20pts)

- We set up a stable matching problem. Each ship and port will have its own preference list. Each ship ranks each port in chronological order of its visits to them. Each port ranks each ship in reverse chronological order of their visits to it. Then we just need to show "the stable matching between ships and ports defines an acceptable assignment of stopping ports." (12pts)

- Proof: (8pts)

    - We assume the assignment is not acceptable. By definition, this means there are at least two ships that are in the same port on the same day. That is, there exists a ship s' passes through port p after ship s has already stopped there.

    - We assume after running the GS algorithm, (s, p) and (s', p') are paired. The constructions of the preference lists are mentioned in the first paragraph. (convert the problem into GS algorithm, 4pts)

    - Then, "*A ship s' passes through port p after ship s has already stopped there*" means in the preference list of s', we have $p > p'$ (since we generate the preference list of ships based on the chronological order of their visit)

    - "*A ship s' passes through port p after ship s has already stopped there*" also means in the preference list of p, we have $s' > s$ (since we generate the preference list of ports based on the reversed chronological order of the ships' visits.

    - If we see the pair (s', p) as an unstable edge, then this is exactly the same problem as when we want to prove the GS algorithm will always output stable matching. We want to prove this kind of unstable edge, (s', p), does not exist. This is what we have done during the class and discussion. (Describe the two ships conflict as unstable edge clearly,4pts)

    - Given the stable matching problem defined above, GS algorithm will always give out stable matching, with no unstable edge (proved on the class). (We allow students to directly call "GS algorithm always output stable matching" ONLY IF the students convert the problem clearly from the ship-port scheduling to the format of stable matching problem we taught on the class)

4. Exercise 4 on page 67 (10pts)

$$g_1 < g_3 < g_4 < g_5 < g_2 < g_7 < g_6$$

There are 1+2+3+4+5+6 =21 pairs of comparison. 0.5pt for each wrong pair of comparison. Maximum -10pts. For example, if the result is $g_5 < g_3 < g_4 < g_1 < g_2 < g_7 < g_6$, in which the pairs in incorrect orders are: (g_5, g_3), (g_5, g_4), (g_5, g_1), (g_1, g_3), (g_1, g_4). Then in total, there will be -2.5 pts.

- Compare $g1$ *and* $g3$: First taking logarithm, and then variable conversion: $x = logn$
- Compare $g3$ *and* $g4$: Variable conversion: $x = logn$, then its exponential vs polynomial
- Compare $g4$ *and* $g5$: Obviously $logn > 4/3$ when n is large
- Compare $g5$ *and* $g2$: First taking logarithm, and then variable conversion: $x = logn$
- Compare $g2$ *and* $g7$: Obviously $n^2 > n$ when n is large
- Compare $g7$ *and* $g6$: Obviously $2^n > n^2$ when n is large

5. a). Prove (by induction) that sum of the first n integers (1+2+....+n) is n(n+1)/2 (10pts)
b). What is 1^3 + 2^3 + 3^3 + ... + n^3 = ? Prove your answer by induction. (10pts)

a) Proof by induction. (10pts)

- When $n = 1$: 1 = 1(1+1)/2. Satisfied

- Assume when $n = k$, $(1 + 2 + \cdots + k) = k(k + 1)/2$

- When $n = k + 1$:

  - $1 + 2 + \cdots + k + 1 = (1 + 2 + \cdots + k) + k + 1 = \frac{k(k+1)}{2} + k + 1 = \frac{(k+1)(k+2)}{2}$

- The results match our assumption, proved.

b) Proof by induction. (10pts)

- Our hypothesis:$1^3 + 2^3 + \cdots + n^3 = \frac{n^2(n+1)^2}{4}$

- When $n = 1$: $1 = \frac{1^2(1+1)^2}{4}$. Satisfied

- Assume when $n = k, 1^3 + 2^3 + \cdots + k^3 = \frac{k^2(k+1)^2}{4}$

- When $n = k + 1$:

  - $1^3 + 2^3 + \cdots + (k + 1)^3 = \frac{k^2(k+1)^2}{4} + (k^3 + 3k^2 + 3k + 1)$

$$= \frac{k^4 + 2k^3 + k^2 + 4k^3 + 12k^2 + 12k + 4}{4}$$

$$= \frac{(k+1)^2(k+2)^2}{4}$$

- The results match our assumption, proved.

6. Given an array A of size N. The elements of the array consist of positive integers. You have to find the largest element with minimum frequency. (10pts)

If the students give out solution based on sorting (O(nlogn)), then 5pts.

If the students give out a working algorithm with O(n), then 10pts.

- The given problem can be solved by storing the frequency of the array in a Hashmap. Key is the current value, Val is the corresponding frequency.

- Store the frequency of each value in Hashmap B, meanwhile keep min_f as the min frequency among all values in the array A during the storing process. – O(n)

- Define max_v as the maximum value with the min frequency. We linear traverse the Hashmap B to see which Key with frequency of min_f is the largest by comparing max_v and the value of current Key. If larger, max_v = the value of current Key – O(n)

-After linear traversing, return max_v .