CS 180 HOMEWORK #1
DUE Oct 11, 2018

WANG, ZHENG (404855295)

Exercise 1. *Exercise 2, Page 22*

Proof. **True**

Suppose towards contardiction that there exists some stable matching S such that $(m, w) \notin S$.

Since all of the man and women must be mathched, then there exists pair $(m, w_2) \in S$, where w_2 is some woman lower in m's list than w (as w ranks first in m's list).

Also, there exists pair $(m_2, w) \in S$, where m_2 is some man lower in w's list than m (as m ranks the first in w's list).

Thus, take the pairs (m, w') and (m', w). Since it is known that m prefers w to w' and w prefers m to m', this is an instance of instalbe match, a contradiction to the assumption that the matching is stable. □

Exercise 2. *Exercise 3, Page 22*

Proof. **There exist a case where there is no stable pair**

An example is when there is 3 slots.
Suppose A network has program a_1, a_2, a_3, each with rating $1, 3, 5$; B network has program b_1, b_2, b_3, each with rating $2, 4, 6$.

This is summarized in the following table: *Claim*: For all schedules T used by A, There exists a schedule S for B such that B can win 2 out of 3 slots.

Proof: Let b_3 compete against a_1, b_2 compete agiainst a_2, and b_1 compete against a_3.

Network	**A**			**B**		
Program	a_1	a_2	a_3	b_1	b_2	b_3
Rate	1	3	5	2	4	6

Claim: For all schedules S' used by B, There exists a schedule T' for A such that A can win 2 out of 3 slots.

Proof: Let a_3 compete against b_2, a_2 compete agianst b_1, and a_1 compete against b_3.

Since both of the network has a potential to win 2 out of 3 slots, which ever network wins less than 2 slots can shift the scheduel to win 2 slots, and make the other network winning only 1 slot. But this then make the other network become capable of shifting its scheduel and win two slots. This cycle can continue forever

Thus there is no stable matching. □

Exercise 3. *Exercise 8, Page 27*

Proof. There exists a set of preferenc list such that a switch would imporve the partner of a women who switched preferencs:
Suppose the list of all men is: (m_1, m_2, m_3) and algorithm choose man in this order, the list of all women is : (w_1, w_2, w_3), and w_2 is the women who shifts perference.
Suppose the perference list for man is:

$$\begin{cases} m_1 : (w_2, w_1, w_3) \\ m_2 : (w_2, w_3, w_1) \\ m_3 : (w_3, w_2, w_1) \end{cases}$$

and for women, the **true** preference list is :

$$\begin{cases} w_1 : (m_1, m_2, m_3) \\ w_2 : (m_3, m_2, m_1) \\ w_3 : (m_2, m_3, m_1) \end{cases}$$

Then before w shift her preference, the algorithm does the following:

In the first round, m_1 propose and get engaged to w_2; then m_2 propose to w_2 and get engaged, setting m_1 free; then m_3 propose to w_3 and get engaged.

In the second round, m_1 propose to w_1 and get engaged.

This result in the pairing:

$$m_1 \longleftrightarrow w_1$$
$$m_2 \longleftrightarrow w_2$$
$$m_3 \longleftrightarrow w_3$$

After w_2 shift preference list to (m_3, m_1, m_2), the algorithm does the following:

In the first round, m_1 propose to w_2 and get engaged; then m_2 propose to w_2 and get reject, so m_2 remains free; then m_3 propose to w_3 and get engaged.

In the second round, m_2 propose to w_3 and get engaged, setting m_3 free.

In the third round, m_3 propose to w_2 and get engaged, setting m_1 free.

In the fourth round, m_1 propose to w_1 and get engaged.

The resulting pairing is:

$$m_1 \longleftrightarrow w_1$$
$$m_2 \longleftrightarrow w_3$$
$$m_3 \longleftrightarrow w_2$$

Then, w_2 ends up with an improved partner. □

Exercise 4. *Exercise 4, Page 67*

The list in ascending order is:

$$g_1(n) = 2^{\sqrt{\log n}}$$
$$g_3(n) = n(\log n)^3$$
$$g_4(n) = n^{4/3}$$
$$g_5(n) = n^{\log n}$$
$$g_2(n) = 2^n$$
$$g_7(n) = 2^{n^2}$$
$$g_6(n) = 2^{2^n}$$

Exercise 5(a)

Proof.
Let $P(n)$ be the statment such that:

$$P(n) : \text{“}1 + 2 + \cdots + n = \frac{n(n+1)}{2}\text{”}$$

$P(1)$ says that $1 = ((1+1) \cdot 1)/2$, which is true.
Aussme that $P(n)$ is true, then we have

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

Thus, $P(n+1)$ holds. By induction, $P(n)$ holds for all $n \in \mathbb{N}$. □

Exercise 5(b)

Proof. Let $P(n)$ be the statement such that:

$$P(n): \text{“}1 \times 2 + 2 \times 3 + 3 \times 4 + \cdots + n(n+1) = \frac{n(n+1)(n+2)}{3}\text{”}$$

$P(1)$ says that $1 \times 2 = \frac{1\cdot(1+1)\cdot(1+2)}{3}$, which is true.
Assume that $P(n)$ holds, then we have

$$\begin{aligned} 1 \times 2 + 2 \times 3 + \cdots + n(n+1) + (n+1)(n+2) =& \frac{n(n+1)(n+2)}{3} + \frac{3(n+1)(n+2)}{3} \\ =& \frac{(n+1)(n+2)(n+3)}{3} \end{aligned}$$

Thus, $P(n+1)$ holds. By induction, $P(n)$ holds for all $n \in \mathbb{N}$. □

Exercise 6
Using the mega-small step used in class, we take m-step as a mega-step.

In the class, we discussed and algorithm that set $m = \sqrt{n}$. However, it is obvious that as we try more mega-steps, The total number of tries increases. This is because we the number of tries after the first egg breaks at a perticualr mega-step remains constant.

Thus the lower bound for this algorithm is m. In the worst case, the algorithm will run at about $2m$ time.

However, we could see that if we go m step in the first mega-step try, but then go up by only $m-1$ after the egg passes the first mega-step try, and $m-2$ after passing the second mega-step try...(In general we go up by $m-k$ after passing the k^{th} mega-step try), then we would always endup with $(m-k)+k = m$ tries whenever the first egg breaks. Thus, this should improve the algorithm discussed in the class.

So, when there are 200 steps, using this idea we end up getting that

$$m + (m-1) + (m-2) + \cdots + 1 = \frac{m(m+1)}{2} = 200$$

So we have to take 20 tries (since the excat solution is 19.5) to solve this problem. Comparing to the algorithm that takes $m = \sqrt{n}$, which should takes about 28 tries, this is an improvment.

Thus for n steps, we generalize the equation for 200-step case and get this equation:

$$\frac{m(m+1)}{2} = n$$

It has solution $\frac{\sqrt{1+8n}-1}{2}$. Therefore, in general, we will have to take $\lceil\frac{\sqrt{1+8n}-1}{2}\rceil$ steps to find out the answer. With this general formula, we can now proof that our algorithm is better than the one we discussed in class:

$$\lim_{n\to\infty}\frac{T_{class}(n)}{T_{new}(n)} = \frac{2\sqrt{n}}{\frac{\sqrt{1+8n}-1}{2}} = \sqrt{2}$$

Since this limit is larger than 1. The algorithm discussed in the class will run longer than our new algorithm at large n, so this new algorithm will have better performance.