Sample Solution to Problem Set 1

1. (4 points) Instability of a local improvement algorithm

As we discussed in class, there may be several approaches to finding a stable matching. One reasonable approach is the following. Suppose we start with an arbitrary matching, and then repeat the following step until there are no unstable pairs.

- If there exists unstable pairs (m, w) and (m', w') such that m prefers w' over w and w' prefers m over m', then replace the pairs by (m, w') and (m', w).

The above local improvement algorithm is fairly natural. But it does not work! Cycles may occur, especially if you choose the "wrong" unstable pairs to swap, causing the algorithm to loop forever. Consider the following preference lists with 3 women A B, C, and 3 men U, V, and W.

A	B	C	U	V	W
U	W	U	B	A	A
W	U	V	A	B	B
V	V	W	C	C	C

For the above preference list, show that there exists a 4-step cycle in the local improvement algorithm, starting with the matching $\{(A, U), (B, V), (C, W)\}$.

Answer:

$$\begin{aligned} &(A,U)(B,V)(C,W)\\ &\hfill B \leftrightarrow U\\ &(A,V)(B,U)(C,W)\\ &\hfill B \leftrightarrow W\\ &(A,V)(B,W)(C,U)\\ &\hfill A \leftrightarrow W\\ &(A,W)(B,V)(C,U)\\ &\hfill B \leftrightarrow U\\ &(A,U)(B,V)(C,W) \end{aligned}$$

2. (4 points) Stable Matching with ties in preference lists

The Stable Matching Problem, as we discussed in class, assumes that all resource providers and resource seekers (say, hospitals and students) have a fully ordered list of preferences. Here, we will consider a version of the problem in which one of the parties, say the students, can have equal preference for certain options.

As before we have a set H of n hospitals, and a set S of n students. Assume each hospital and each student ranks the members of the other set. The hospitals have a strict preference ordering of the students; given students s_1 and s_2, hospital h either prefers s_1 to s_2, or vice versa. On the student side, we allow for ties. A student s may prefer hospital h_i over hospital h_j, have equal preference for h_j and h_k, and prefer h_j and h_k over h_l.

With equal preferences allowed in the ranking of hospitals by students, we define the notion of instability as follows:

An *instability* in a perfect matching S consists of a hospital h and a student s, such that each of h and s has a *strictly higher* preference for each other than their partner in S.

In this variation of the problem, we ask the following.

True or **False**: Does there always exist a perfect matching with no instability?

If your answer is true, then give an algorithm that is guaranteed to find a perfect matching with no instability, and prove the correctness of your algorithm.

If your answer is false, then give an instance of a set of hospitals and students with preference lists for which you argue that every perfect matching has an instability.

Answer: True. There always exists a stable matching. The algorithm is adapted from the Gale-Shapley algorithm we reviewed in class, where hospitals propose and the proposal either results in a match, or is rejected by the student proposed to. A rejection occurs when the student has a current partner who ranks higher than or is in a tie with the proposing hospital.

```
MODIFIEDGALESHAPELY
Input: Hospitals, Students, Preference lists
Initialize M to empty matching.
while some hospital h is unmatched and has not proposed to every student:
    let s be a student most preferred among ones not yet proposed to by h
    if s is unmatched
      add (h, s) to M
    else
      if s prefers h to current partner h'
        replace (h', s) with (h, s) in M
      else s rejects h
return M
```

The following observations from the Gale-Shapley algorithm hold for the modified G-S algorithm:

Observation 1: Hospitals propose to students in decreasing order of preference.

Observation 2: Once a student is matched, the student never becomes unmatched; only *trades up*.

Proof of termination: The modified G-S algorithm terminates after at most n^2 iterations of the while loop. To see this, note that each time through the while loop, a hospital proposes to a new student. There are n hospitals and at most n students that each hospital will propose to. Thus, there are at most n^2 possible iterations through the while loop.

Proof of Correctness: The modified G-S algorithm always returns a stable matching. We show it in three steps.

Lemma 1: The modified G-S algorithm returns matching.

Proof: (1.1) Each hospital proposes only when free (unmatched). There is at most one match for each hospital. (1.2) Every student appears in at most one match. Once matched, they only trades up.

It follows that the modified G-S algorithm outputs a matching.

Lemma 2: The modified G-S algorithm returns a perfect matching.

Proof: The while loop terminates only when (a) all hospitals are matched, or (b) a free hospital has run out of students to propose to on its preference list. If all hospitals are matched, from (1.1) it follows that the modified G-S algorithm results in a perfect matching.

If at the end of the while loop there is a free hospital that has run out of students to propose to, it implies that all the students are matched. From (1.2), it follows that all the hospitals are matched, resulting in a contradiction. This proves that the modified G-S algorithm outputs a perfect matching.

Lemma 3: The modified G-S algorithm outputs a stable matching.

Proof: Lets assume that the modified G-S algorithm returns a perfect matching M and that there is an instability with respect to M; i.e., there are pairs (h, s) and (h', s') in M such that h prefers s' over s , and s' prefers h over h'. Since hospitals propose in order of preference, h must have proposed to s' prior to proposing to s. There are 2 possible scenarios:

1. s' rejected h preferring her current partner, say h; i.e., h^* ranks strictly higher than h in the preference list of s'. Since students only trade up, it follows that the final partner of s' in M, h', ranks higher than h.

2. s' accepted the proposal from h, but broke it later preferring a trade up. Once again, it follows that the final partner of s' in M, h', ranks higher than h. This contradicts our assumption that s' prefers h over h'. Therefore, there is no instability with respect to the output of the modified G-S algorithm. The modified G-S algorithm returns a stable matching.

3. (4 points) Fibonacci numbers

The *Fibonacci numbers* are defined by the following recurrence:

$$\begin{aligned} F_0 &= 0 \\ F_1 &= 1, \text{ and} \\ F_i &= F_{i-1} + F_{i-2} \text{ for } i \geq 2. \end{aligned}$$

Prove by induction that for any $n \geq 2$, F_n equals $A_{1,1}^{n-1}$, where A^n is the nth power of the following matrix:

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}.$$

Answer: We show by induction on $n \geq 1$ that:

$$A^n = \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix}.$$

Note that once we have shown the above fact, the desired claim of the problem follows immediately.

The induction base ($n = 1$) follows directly from the definition of A and the fact that $F_0 = 0$ and $F_1 = F_2 = 1$.

We now establish the induction step for $n = m \geq 2$ assuming that the claim holds for all $n < m$. We write A^m as $A^{m-1}A$ and obtain:

$$A^m = \begin{pmatrix} F_m & F_{m-1} \\ F_{m-1} & F_{m-2} \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} F_m + F_{m-1} & F_m \\ F_{m-1} + F_{m-2} & F_{m-1} \end{pmatrix} = \begin{pmatrix} F_{m+1} & F_m \\ F_m & F_{m-1} \end{pmatrix}.$$

4. (4 points) Ordering functions

Arrange the following functions in order from the slowest growing function to the fastest growing function. Briefly justify your answers. (*Hint:* It may help to plot the functions and obtain an estimate of their relative growth rates. In some cases, it may also help to express the functions as a power of 2 and then compare.)

$$\sqrt{n} \quad n\sqrt{\lg n} \quad 2^{\sqrt{\lg n}} \quad (\lg n)^2$$

Answer: The order from slowest growing to the fastest growing function is: $(\lg n)^2$, $2^{\sqrt{\lg n}}$, $\sqrt{n}$, $n\sqrt{\lg n}$. The informal reasoning behind this is as follows. The function $(\lg n)^2$ clearly grows slowly than $n\sqrt{\lg n}$ since the former is a poly-logarithmic function and the latter is polynomial. Where do we place $2^{\sqrt{\lg n}}$ and $\sqrt{n}$? The function $\sqrt{n}$ is a polynomial and grows faster than $(\lg n)^2$ and clearly slower than $n\sqrt{\lg n}$ since the exponent on n is $1/2 < 1$.

That leaves $2^{\sqrt{\lg n}}$. The function $\lg n$ can be written as $2^{\lg \lg n}$ and $\sqrt{n}$ as $2^{(\lg n)/2}$. Comparing the three functions $2^{\sqrt{\lg n}}$, $\lg n$, and $\sqrt{n}$ is equivalent to comparing $\sqrt{\lg n}$, $\lg \lg n$ and $(\lg n)/2$. Since $\sqrt{\lg n}$ is greater than $\lg \lg n$ and smaller than $(\lg n)/2$, we can place $2^{\sqrt{\lg n}}$ between the other two functions.

We now present a formal proof using three steps: (i) $(\lg n)^2 = o(2^{\sqrt{\lg n}})$, (ii) $2^{\sqrt{\lg n}} = o(\sqrt{n})$, and (iii) $\sqrt{n} = o(n\sqrt{\lg n})$.

(i) We use the connection between asymptotic notation and limits.

$$\begin{aligned} \lim_{n\to\infty} \frac{(\lg n)^2}{2^{\sqrt{\lg n}}} &= \lim_{m\to\infty} \frac{m^2}{2^{\sqrt{m}}} \\ &= \lim_{m\to\infty} \frac{2m}{2^{\sqrt{m}} \ln 2/(2\sqrt{m})} \\ &= \lim_{m\to\infty} \frac{4m^{3/2}}{2^{\sqrt{m}}} \end{aligned}$$

$$
\begin{aligned}
&= \lim_{m\to\infty} \frac{6\sqrt{m})}{2^{\sqrt{m}}(\ln 2)^2/(2\sqrt{m})} \\
&= \lim_{m\to\infty} \frac{12m}{2^{\sqrt{m}}(\ln 2)^2} \\
&= \lim_{m\to\infty} \frac{12}{2^{\sqrt{m}}(\ln 2)^3/(2\sqrt{m})} \\
&= \lim_{m\to\infty} \frac{24\sqrt{m}}{2^{\sqrt{m}}(\ln 2)^3} \\
&= \lim_{m\to\infty} \frac{24/(2\sqrt{m})}{2^{\sqrt{m}}(\ln 2)^4/(2\sqrt{m})} \\
&= \lim_{m\to\infty} \frac{24}{2^{\sqrt{m}}(\ln 2)^4} \\
&= 0,
\end{aligned}
$$

where we replace $\lg n$ by m in the first step (since $\lg n$ is monotonically increasing in n) and use L'Hopital's rule multiple times. The final step holds since $2^{\sqrt{m}}$ goes to infinity as m goes to infinity. Therefore, $(\lg n)^2 = o(2^{\sqrt{\lg n}})$.

(ii) We again use the connection between asymptotic notation and limits.

$$
\begin{aligned}
\lim_{n\to\infty} \frac{2^{\sqrt{\lg n}}}{\sqrt{n}} &= \lim_{n\to\infty} \frac{2^{\sqrt{\lg n}}}{2^{(\lg n)/2}} \\
&= \lim_{n\to\infty} \frac{1}{2^{(\lg n)/2-\sqrt{\lg n}}} \\
&= 0,
\end{aligned}
$$

since $(\lg n)/2 - \sqrt{\lg n}$ goes to ∞ as n goes to ∞. Therefore, $2^{\sqrt{\lg n}} = o(\sqrt{n})$.

(iii) We again use the connection between asymptotic notation and limits.

$$
\begin{aligned}
\lim_{n\to\infty} \frac{\sqrt{n}}{n\sqrt{\lg n}} &= \lim_{n\to\infty} \frac{1}{\sqrt{n}(\lg n)} \\
&= 0,
\end{aligned}
$$

Therefore, $\sqrt{n} = o(n\sqrt{\lg n})$.

5. (2 × 2 = 4 points) Properties of asymptotic notation

Let $f(n)$, $g(n)$, and $h(n)$ be asymptotically positive and monotonically increasing functions. For each of the following statements, decide whether you think it is true or false and give a proof or a counterexample.

(a) $f(n) + g(n) = \Theta(\max\{f(n), g(n)\})$.

Answer: True. Since $f(n) + g(n) \geq \max\{f(n), g(n)\}$ for all n, we have $f(n) + g(n) = \Omega(\max\{f(n), g(n)\})$ ($c = 1, n_0 = 1$). Since $f(n)+g(n) \leq 2\max\{f(n), g(n)\}$ for all n, we have $f(n) + g(n) = O(\max\{f(n), g(n)\})$ ($c = 2, n_0 = 1$).

(b) If $f(n) = O(g(n))$, then $2^{f(n)}$ is $O(2^{g(n)})$.

Answer: False. Take $f(n) = n$ and $g(n) = n/2$. Clearly, $f(n) = O(g(n))$ since $f(n) \leq 2g(n)$ for all $n \geq 0$. But $2^{f(n)} = 2^n$ while $2^{g(n)} = 2^{n/2}$, implying that $2^{f(n)}/2^{g(n)}$ equals $2^{n/2}$ which tends to ∞ as n tends to ∞. Hence, $2^{f(n)}$ is not $O(2^{g(n)})$.