CENG 567 Design and Analysis of Algorithms
Homework 1

Ardan Yılmaz 2172195

October 23, 2022

1 Stable Matching

Men	Pref. List
X	C,D,A,B
Y	D,A,B,C
Z	A,B,C,D
W	D,C,A,B

Women	Pref. List
A	X,Y,Z,W
B	X,Y,Z,W
C	Y,Z,X,W
D	Z,X,Y,W

(a) Use the Gale and Shapley algorithm to find a man-optimal stable matching.

Man-optimal matching is yielded when men propose in decreasing order in their preference lists. The following are the steps taken:

1. X proposes to C. C, without a match, accepts X. X-C matched.
2. Y proposes to D. D, without a match, accepts Y. Y-D matched.
3. Z proposes to A. A, without a match, accepts Z. Z-A matched.
4. W proposes to D. D, with a better partner (Y) than W, rejects W.
5. W proposes to C. C, with a better partner (X) than W, rejects W.
6. W proposes to A. A, with a better partner (Z) than W, rejects W.
7. W proposes to B. B, without a match, accepts W. W-B matched.

No free men or women left. Algorithm stops.

(X-C, Y-D, Z-A, W-B) is the stable matching obtained using man-optimal GS algorithm.

(b) Find a woman-optimal stable matching in this instance.

Women-optimal matching is yielded when women propose in decreasing order in their preference lists. And now, men accept when they are first offered, however trade up when proposed to by a better option. The following are the steps taken:

1. A proposes to X. X, without a current match, accepts X. A-X matched.
2. B proposes to X. X, with a better partner (A), rejects B.
3. B proposes to Y. Y, without a current match, accepts Y. B-Y matched.
4. C proposes to Y, Y, with a better partner (B), rejects C.
5. C proposes to Z. Z, without a current match, accepts C. C-Z matched.
6. D proposes to Z. Z, with a better partner (C), rejects D.
7. D proposes to X. X likes D better than his current partner, A, so X trades up to D. Hence, A is free and D-X matched.
8. A proposes to Y. Y likes A better than his current partner, B, so Y trades up to A. Hence, B is free and A-Y matched.
9. B proposes to Z. Z likes B better than his current partner, C, so Z trades up to B. Hence, C is free and B-Z matched.
10. C proposes to X. X likes C better than his current partner, D, so X trades up to C. Hence, D is free and C-X matched.
11. D proposes to Y. Y likes D better than his current partner, A, so Y trades up to D. Hence, A is free and D-Y matched.
12. A proposes to Z. Z likes A better than his current partner, B, so Z trades up to A. Hence, B is free and A-Z matched.
13. B proposes to W. W, without a partner, accepts B. B and W settle down for each other. B-W matched.

No free men or women left. Algorithm stops.

(C-X, D-Y, A-Z, B-W) is the stable matching obtained using woman-optimal GS algorithm.

(c) Suppose we are given n men and n women where each man has a preference list of all women and every woman has preference list of all men. On a stable matching instance, prove that if we run the Gale-Shapley algorithm twice, once with men proposing and once with women proposing and we obtain the same matching, then the instance has a unique stable solution.

The GS algorithm is known to produce proposer-optimal mathcings, ie, each person from the proposer group gets their best valid partner.

Claim: The following are members of stable matchings, S and S', such that $(m1, w1), (m2, w2) \in S$ and $(m1, w2), (m2, w1) \in S'$, another stable match.

Knowing woman and man-optimal matchings yield the same matching, S with $(m1, w1), (m2, w2) \in S$. That is, m1 and w1 are the best valid partners for each other, ie, there is no one valid on their preference lists whom they like better. The same also applies to m2 and w2, ie, they are the best valid partners for each other.

For S' to be a stable match, one of the following should hold: (1) m1 likes w2 better than w1 and w2 is a valid partner for him or (2) w2 likes m1 better than m2 and w1 is a valid partner for him. Knowing couples, m1 and w1 and m2 and w2, are the best valid partners for each other, the claim is not possible.

Thus, S is the only stable match there exists.

(d) Show that the Gale-Shapley algorithm terminates within $(n-1)^2+1$ iterations for n boys and n girls and this bound is tight.

Men, except for the last, proposes to n-1 women in the worst case after they are accepted by the last woman in their preference lists. This makes $(n-1)(n-1)$ iterations for the first $n-1$ men. As every guy settled down with the woman at the bottom of their lists, the last guy gets his first offer. This, in total, makes $(n-1)^2+1$ iterations.

2 Alternative Matchings

Men	Pref. List
X	B,A,D,C
Y	C,B,A,D
Z	D,A,C,B
W	C,B,D,A

Women	Pref. List
A	W,Y,X,Z
B	Z,Y,W,X
C	X,Z,Y,W
D	X,W,Z,Y

2.a Is $S = \{(XA), (YB), (ZC), (WD)\}$ a stable matching?

No one pair, matched with others, favors one another better, hence this is a stable matching.

To illustrate,

1. Check X-A match X matches with A whereas he favors B better. However, B is with Y, whom she likes better than X. A settled down for X, whereas, she liked W and Y better. However, both W and Y are down with someone whom they like better than X, namely, D and B. Hence, there is no problem with this match.

2. Check Y-B match Y matches with B, whereas, he likes C better. However, C is already down with Z, whom she favors better than Y. And, B likes Z better than Y. However, Z is already down with C, whom she likes better. Hence, there is no problem with this match either.

3. Check Z-C match Z matches with C, whereas, he favors D or A better, both of whom are already down with people they like better, namely, W and X. And, though C favors X better, he is with A, whom he likes better than C.

Hence, there is no problem with this match either.

4. Check W-D match Though W favors C or B rather than D, they are both down with people they like better than W, namely, Z and Y. And, D likes X better than W, who has a better match, D, than A. Hence, there is no problem with this match either.

That is, this is a stable matching.

2.b. Find all stable matchings of n men and n women. Is there a polynomial time algorithm for this?

The only algorithm I can come up with is the naive approach, where all possible matches are listed, an $O(n!)$ operation. Then check stability of each match. The first part of the algorithm is clearly not polynomial.

2.c. Can a woman end up better off by lying about her preferences?

A woman can end up better off by lying, and below is an example preference list taken from [**1**], where a woman gets a better option by lying.

Preference list of men:
D: A,B,C
E: B,A,C
F: A,B,C

Preference list of women (A lies that she prefers F over D (her lie is indicated in bold in the above list)):
A: E,D,F —— **E,F,D**
B: D,E,F
C: D,E,F

Running the GS algorithm on the true preferences, the final matches are: $D-A, E-B, F-C$. However, if A lies, the final match is: $D-B, E-A, F-C$. That is, being honest, A ended up with her last preference; whereas, she can get her first choice by lying.

3 Order of Growth

a. Prove $n! \in \Omega(2^{n-1})$ **and** $n! \in \Omega(2^{n-1})$

a.1. Prove $n! \in O(n^n)$

if $\lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$, $f(n) \in O(g(n))$

$\lim_{n\to\infty} \frac{n!}{n^n} = \lim_{n\to\infty} \frac{n(n-1)(n-2)...1}{n.n.n.n...n} =$

$(\lim_{n\to\infty} \frac{n}{n}) * (\lim_{n\to\infty} \frac{n-1}{n}) * ...(\lim_{n\to\infty} \frac{1}{n})$

Careful analysis yields that the denominator grows faster, hence this limit converges.

Applying L'Hopital's Rule to the first multiplicative terms, they are all 1. The last term, $\lim_{n\to\infty} \frac{1}{n} = 0$, so $\lim_{n\to\infty} \frac{n!}{n^n} = 0$. Thus, $n! \in O(n^n)$.

a.2. Prove $n! \in \Omega(2^{n-1})$
if $\lim_{n\to\infty} \frac{f(n)}{g(n)} > 0$, $f(n) \in \Omega(g(n))$

$\lim_{n\to\infty} \frac{n!}{2^{n-1}} = \lim_{n\to\infty} \frac{n(n-1)(n-2)...1}{2.2.2....2} = \infty$

The limit is clearly divergent, ie, $n!$ grows much faster than 2^{n-1}. Hence, $n! \in \Omega(2^{n-1})$

(b) Prove for any fixed k, if $n! \in \Omega(k^{n-k})$

By the definition of Ω, for $n! \in \Omega(k^{n-k})$, the following should hold:

$\exists c > 0, n_0 \geq 0$ such that $n! \geq c.k^{n-k} \geq 0$ for $\forall n > n_0$

There are two cases for k, ie, $k \geq n$ or $k < n$

Case 1: if $k \geq n$, then $n - k \leq 0$. That is $k^{n-k} < 1$ and hence the following clearly holds $k^{n-k} < n!$

Case 2: if $k < n$, then $n - k < n$. As proven in **a.2**, $k^{n-k} < n!$ for $\forall k < n$

The requirement for $n! \in \Omega(k^{n-k})$ is satisfied.

(c) Prove for any fixed k, if $n! \in \Omega(k^n)$

By the definition of Ω, for $n! \in \Omega(k^n)$, the following should hold:

$\exists c > 0, n_0 > 0$ such that $n! >= c.k^n >= 0$ for $\forall n > n_0$

This equation does not hold for $k > n$. Hence, $n! \notin \Omega(2^{n-1})$

4 Asymptotics

```
for i = 1, 2, 3..., n do
    for j = i+1, ... ,n do
        compute the maximum of the entries A[i],A[i + 1], · · · ,A[j].
        store the maximum value in B[i, j].
    end for
end for
```

4.a. Find f(n) s.t the run time of the algorithm is $O(f(n))$

The outer-most loop executes for n times. The inner loops execute for at most n times, ie, for each value of i and j, the number of times they execute differ in a range from 1 to n. There are 3 loops nested in each other executing at most n times, hence, the run-time of the algorithm is upper bounded by n^3, which makes $f(n) = O(n^3)$

4.b. Prove the run time of the algorithm is also lower bounded by $f(n) \in \Omega(n^3)$.

The total number of times the algorithm is executed can be found by: $\sum_{i=1}^{n}\sum_{j=i}^{n}\sum_{k=i}^{j} k$, ie, it is a function of degree 3, ie, $T(n) = a_0n^3 + a_1n^2 + a_2n + a_3$. That's why it is also lower bounded by n^3.

4.c. Design a faster algorithm.

Computing the maximum of entries, an $O(n)$ operation, in the inner-most loop is seems redundant. We can have a dynamic-programming approach instead. That is, each entry in the upper triangle of matrix B, is the maximum among those entries that are in the same up to the index of that entry and the one in the same index in matrix A. In other words, $B[i, j] = max\{B[i, j - 1], A[j]\}$, as $B[j - 1]$ is the largest among the items in that row and A up to index, $j - 1$.

As we have multiple rows, one can calculate the values of the first entries of the upper triangle of the matrix B, ie, $B[i, i+1]$, as a preprocess step with $O(n)$ complexity. And then, $B[i, j] = max\{B[i, j-1], A[j]\}$ operation can be applied at every step. The whole algorithm with $O(n^2)$ complexity is as follows:

Algorithm 1 A more efficient algorithm

for $i = 1, 2, 3..., n-1$ **do** //Preprocess step
    $B[i, i+1] = max\{A[i], A[i+1]\}$
end for
for $i = 1, 2, 3..., n$ **do**
    for j = i+2, ... ,n **do**
        $B[i, j] = max\{B[i, j-1], A[j]\}$
    end for
end for

The preprocess step of the algorithm is $O(n)$, and the rest is $O(n^2)$. The overall complexity is $O(n) + O(n^2) = O(n^2)$

5 Big O and Ω

(5.a) $f(n) = 1/2n^2$

Claim 5.a.1 $f(n) = 1/2n^2 \in O(n^2)$

Pf. By big O definition, $f(n) \in O(g(n))$ if $\exists c > 0, n_0 \geq n$, the following holds $0 \leq f(n) \leq c.g(n)$ for $\forall n > n_0$
$f(n) = 1/2n^2 \leq n^2$ for $\forall n > 0$, where $c = 1$ and $n_0 = 1$
Hence, $f(n) = 1/2n^2 \in O(n^2)$

Claim 5.a.2 $f(n) = 1/2n^2 \in \Omega(n^2)$

Pf. By big Ω definition, $f(n) \in \Omega(g(n))$ if $\exists c > 0, n_0 \geq n$, the following holds $f(n) \geq c.g(n)0 \geq 0$ for $\forall n > n_0$
Choose c = 1/4, and $n_0 = 1$, then $1/2n^2 \geq 1/4n^2 \geq 0$ for $\forall n > n_0$
Hence, $f(n) = 1/2n^2 \in \Omega(n^2)$

As $f(n) \in O(n^2)$ and $f(n) \in \Omega(n^2)$, $f(n) \in \theta(n^2)$, that is, H = L = 2

(5.b) $f(n) = n(log(n))^3$

5.b.1

Claim 1: $f(n) = nlog^3(n) \in O(n^2)$

Pf. 1

Let $g(n) = n^2$. Dividing both $g(n)$ and $f(n)$ by n: $g(n)/n = n$ and $f(n)/3 = log^3(n)$

Taking cubic root of both sides, $(g(n)/n)^{1/3} = n^{1/3}$ and $(f(n)/n)^{1/3} = log(n)$
Knowing any polynomial function grows faster than logarithmic ones, $n^{1/3} > log(n)$. Hence, $f(n) = nlog^3(n) \in O(g(n^2))$.

Claim 2: H=2 is the smallest integer constant satisfying the following: $f(n) \in O(n^H)$.

Pf. 2: Knowing $f(n) = nlog^3(n) \in O(g(n^2))$, check if $f(n) = nlog^3(n) \in O(g(n))$.

Check $\lim_{n\to\infty} \frac{n}{nlog^3(n)}$, applying L'Hopital's Rule
$\lim_{n\to\infty} \frac{n}{nlog^3(n)} = \lim_{n\to\infty} \frac{1}{log^3(n)} = 0$.

That is, $f(n) = nlog^3(n) \notin O(g(n))$. This concludes the proof that H=2 is the smallest integer satisfying.

5.b.2

Claim 1: $f(n) = nlog^3(n) \in \Omega(n)$

Pf. 1: $\lim_{n\to\infty} \frac{nlog^3(n)}{n} = \lim_{n\to\infty} \frac{log^3 n + n * 3log^2 n}{1}$ (By the L'Hopital's Rule).

$\lim_{n\to\infty} \frac{log^3 n + n * 3log^2 n}{1} > 0$. Hence, $f(n) = nlog^3(n) \in \Omega(n)$.

Claim 2: L=1 is the largest integer satisfying $f(n) = nlog^3(n) \in \Omega(n^L)$

Pf. 2: Assume L=2 also satisfies the above equation, ie, $f(n) = nlog^3(n) \in \Omega(n^2)$

For $f(n) \in \Omega(g(n)$, $\lim_{n\to\infty} \frac{f(n)}{g(n)} > 0$.

$\lim_{n\to\infty} \frac{nlog^3 n}{n^2} = \lim_{n\to\infty} \frac{log^3 n}{n} = 0$, as n grows faster than any logarithmic function. That is, we arrive at a contradiction, ie, $f(n) = nlog^3(n) \notin \Omega(n^2)$

L=1 is the largest integer satisfying $f(n) = nlog^3(n) \in \Omega(n^L)$

5.c

$\sum_{n=0}^{\lceil logn \rceil} \frac{n}{2^i} = n \sum_{n=0}^{\lceil logn \rceil} (1/2)^i = n * \frac{1 - 1/2^{logn+1}}{1 - 1/2} = 2n(1 - \frac{1}{2n}) = 2n - 1 \in O(n)$ and $\Omega(n)$. Hence, H=L=1 are the largest and smallest values.

5.d

$\sum_{i=1}^{n} i^3 = (\frac{n(n+1)}{2})^2$, which is clearly $O(n^4)$ and $\Omega(n^4)$. That is, L=4 is the largest, and the H=4 is the smallest integer values possible.

5.e

$2^{(logn)^2}$ is clearly an exponential function, that is, it cannot be upper bounded by a polynomial. By the same argument, the lower bound of it approaches to infinity, ie, not a real number. Thus, H and L values do not exist.

6 REFERENCES

[1] Gale Shapley algorithm and stable matching - medium. (n.d.). Retrieved October 22, 2022, from https://medium.com/@yunhanh/gale-shapley-algorithm-and-stable-matching-dbf1bf748541