CMSC 27200 - Problem Set 1

Sohini Banerjee

January 12, 2024

1a

- a_1 makes an offer to b_3
  - accepted since b_3 unmatched
  - partial matching: $M = \{(a_1, b_3)\}$
- a_2 makes an offer to b_1
  - accepted since b_1 unmatched
  - partial matching: $M = \{(a_1, b_3), (a_2, b_1)\}$
- a_3 makes an offer to b_4
  - accepted since b_4 unmatched
  - partial matching: $M = \{(a_1, b_3), (a_2, b_1), (a_3, b_4)\}$
- a_4 makes an offer to b_1
  - accepted since b_1 prefers a_4 to a_2
  - partial matching: $M = \{(a_1, b_3), (a_3, b_4), (a_4, b_1)\}$
- a_2 makes an offer to b_4
  - rejected since b_4 prefers a_3 to a_2
  - partial matching: $M = \{(a_1, b_3), (a_3, b_4), (a_4, b_1)\}$
- a_2 makes an offer to b_3
  - accepted since b_3 prefers a_2 to a_1
  - partial matching: $M = \{(a_2, b_3), (a_3, b_4), (a_4, b_1)\}$
- a_1 makes an offer to b_2
  - accepted since b_2 unmatched
  - partial matching: $M = \{(a_1, b_2), (a_2, b_3), (a_3, b_4), (a_4, b_1)\}$

final matching: $M = \{(a_1, b_2), (a_2, b_3), (a_3, b_4), (a_4, b_1)\}$

1b

- b_1 makes an offer to a_1
  - accepted since a_1 unmatched
  - partial matching: $M = \{(a_1, b_1)\}$
- b_2 makes an offer to a_3
  - accepted since a_3 unmatched
  - partial matching: $M = \{(a_1, b_1), (a_3, b_2)\}$
- b_3 makes an offer to a_3
  - accepted since a_3 prefers b_3 to b_2
  - partial matching: $M = \{(a_1, b_1), (a_3, b_3)\}$
- b_4 makes an offer to a_1
  - rejected since a_1 prefers b_1 to b_4
  - partial matching: $M = \{(a_1, b_1), (a_3, b_3)\}$
- b_2 makes an offer to a_4
  - accepted since a_4 unmatched
  - partial matching: $M = \{(a_1, b_1), (a_3, b_3), (a_4, b_2)\}$
- b_4 makes an offer to a_4
  - accepted since a_4 prefers b_4 to b_2
  - partial matching: $M = \{(a_1, b_1), (a_3, b_3), (a_4, b_4)\}$
- b_2 makes an offer to a_2
  - accepted since a_2 unmatched
  - partial matching: $M = \{(a_1, b_1), (a_2, b_2), (a_3, b_3), (a_4, b_4)\}$

final matching: $M = \{(a_1, b_1), (a_2, b_2), (a_3, b_3), (a_4, b_4)\}$

- a_1 prefers b_2 to b_1, so a_1 is unhappier.
- a_2 prefers b_3 to b_2, so a_2 is unhappier.
- a_3 prefers b_4 to b_3, so a_3 is unhappier.
- a_4 prefers b_1 to b_4, so a_4 is unhappier.
- b_1 prefers a_1 to a_4, so b_1 is happier.
- b_2 prefers a_2 to a_1, so b_2 is happier.
- b_3 prefers a_3 to a_2, so b_3 is happier.
- b_4 prefers a_4 to a_3, so b_4 is happier.

Therefore, b_1, b_2, b_3, b_4 are happier in the new stable matching. This makes sense because when group B makes offers, everyone in group A gets their worst possible partner in a stable matching, while everyone in group B gets their best possible partner in a stable matching.

1c

When group A makes offers, we get the best possible stable matching for group A and worst possible stable matching for group B. When group B makes offers, we get the worst possible stable matching for group A and best possible stable matching for group B. Knowing this, we can form a range of what possible partners group A can have, ranging from their best possible matching to worst possible stable matching. The range for group A is all the members of B that are between A's best and worst option. Thus, the options for A are:

- $a_1 : b_2, b_1$
- $a_2 : b_3, b_2$
- $a_3 : b_4, b_1, b_3$
- $a_4 : b_1, b_4$

From here, we can form all the possible stable matchings:

- $M = \{(a_1, b_2), (a_2, b_3), (a_3, b_4), (a_4, b_1)\}$
- $M = \{(a_1, b_2), (a_2, b_3), (a_3, b_1), (a_4, b_4)\}$
- $M = \{(a_1, b_1), (a_2, b_2), (a_3, b_3), (a_4, b_4)\}$

2a

- $3n + 2n^2 - 15 : O(n^2)$
- $4n \log n + 3n + 20 : O(n \log n)$
- $2n^2 - n + \frac{1}{6}n^3 : O(n^3)$
- $\frac{1}{2}\sqrt{n} + 7(\log n)^{10} : O(\sqrt{n})$
  - Taking the logarithm of both terms, we get $\log \sqrt{n} = \frac{1}{2} \log n$ and $\log((\log n)^{10}) = 10 \log \log n$. $\log n$ grows faster than $\log \log n$, so we get $O(\sqrt{n})$.
- $5n \log n + \frac{n}{\log n} + 47n^{\log \log n} : O(n^{\log \log n})$
  - We know that $n \log n$ grows faster than $\frac{n}{\log n}$. Taking the logarithm of $n \log n$ and $n^{\log \log n}$, we get $\log(n \log n) = \log n + \log \log n$ and $\log(n^{\log \log n}) = \log \log n \cdot \log n$. $\log \log n \cdot \log n$ grows faster than $\log n + \log \log n$, so we get $O(n^{\log \log n})$.

2b

The following is the functions in order of increasing asymptotic growth. The ones on the same line indicate same asymptotic growth:

- $2^{\log \log n}, \log n$
- $2^{\sqrt{\log n}}$
- n
- $n \log n$
- $\frac{n^2}{\log n}$

- $n^2, 2^{2\log n}$
- $n^{\log n}$
- 2^n
- $n2^n$
- 2^{2n}

3a

Assume for contradiction that everyone in group A is matched with their least preferred partner. Consider the last proposition where a_i makes an offer to b_j, its least preferred partner. b_j must accept because Gale Shapley results in a stable matching, and a_i does not have anyone else left to ask after b_j. Given that b_j must accept, there can be two scenarios:

- b_j is unmatched.
- b_j is matched and leaves its current partner for a_i. This is not possible because b_j is its current partner's nth choice, meaning if b_j leaves its current partner for a_i, its current partner will have nobody else in group B to ask.

Therefore, the latter is not possible, and b_j must be unmatched. Since everyone in group A who is currently matched is matched to their least preferred partner (who is not b_j since b_j is unmatched), they must have already proposed to b_j at some point since b_j must be preferred to their current partner. This means b_j already received $n-1$ offers, so it must be matched because it has received an offer. Thus, we get a contradiction because we claimed that b_j must be unmatched.

Thus, everyone in group A cannot be matched with their least preferred partner.

3b

Everyone in group B can be paired with their least preferred partner when group A makes the offers. This is possible because we can have everyone in group A propose to a different person on their first offer. For each a_i proposing to a b_j, the b_j has that a_i as their last choice. We demonstrate as follows:

We propose the following preference lists for group A:

$$\begin{array}{rccc} a_1 : & b_1 & ... & ... \\ a_2 : & b_2 & ... & ... \\ ... & ... & ... & ... \\ a_{n-1} : & b_{n-1} & ... & ... \\ a_n : & b_n & ... & ... \end{array}$$

We propose the following preference lists for group B:

$$\begin{array}{rccc} b_1 : & ... & ... & a_1 \\ b_2 : & ... & ... & a_2 \\ ... & ... & ... & ... \\ b_{n-1} : & ... & ... & a_{n-1} \\ b_n : & ... & ... & a_n \end{array}$$

If we simulate the offers, we get the following steps:

- a_1 offers b_1, which accepts because b_1 unmatched.

- a_2 offers b_2, which accepts because b_2 unmatched.
- Continuing this pattern, all a_i offer their 1st choice, which accepts because they are all unique and thus unmatched.

In this match, everyone in group B is matched to their least preferred partner in group A.

3c

The Gale Shapley algorithm terminates when all members of group B are assigned a partner. Thus, to maximize the number of offers made, we must postpone the last member of group B getting an offer. The maximum number of offers a member of group B can receive is n. So, excluding the final member of B, we can get a maximum of n offers for the remaining $n-1$ members of B. Then, the unpaired final member of B receives an offer and accepts it. Here, the algorithm terminates with $(n-1)(n)+1 = n^2-n+1$ offers.

We propose the following preference lists for group A:

$$\begin{array}{rcccccc}
a_1: & b_1 & b_2 & \dots & b_{n-1} & b_n \\
a_2: & b_2 & b_3 & \dots & b_1 & b_n \\
a_3: & b_3 & b_4 & \dots & b_2 & b_n \\
\dots & \dots & \dots & \dots & \dots & \dots \\
a_{n-1}: & b_{n-1} & b_1 & \dots & b_{n-2} & b_n \\
a_n: & b_1 & b_2 & \dots & b_{n-1} & b_n
\end{array}$$

We propose the following preference lists for group B:

$$\begin{array}{rcccccc}
b_1: & a_2 & a_3 & \dots & a_n & a_1 \\
b_2: & a_3 & a_4 & \dots & a_1 & a_2 \\
b_3: & a_4 & a_5 & \dots & a_2 & a_3 \\
\dots & \dots & \dots & \dots & \dots & \dots \\
b_{n-1}: & a_n & a_1 & \dots & a_{n-2} & a_{n-1} \\
b_n: & a_1 & a_2 & \dots & a_{n-1} & a_n
\end{array}$$

These preference lists allow $n^2 - n + 1$ offers to be made. Everyone in group A except a_1 gets their $(n-1)$th preferred partner, and a_1 gets its nth preferred partner. If we simulate the offers, we get the following steps:

- First, a_1 through a_{n-1} ask their 1st choice partner and are paired. Then, a_n asks b_1 and b_1 leaves a_1 for a_n.
- a_1 is currently the only unpaired member of group A. a_1 asks b_2 and b_2 leaves a_2 for b_1. This makes b_2 the only unpaired member of group A. With this pattern, a_1 through a_{n-1} ask their 2nd choice partner. Here, when the a_ith person asks their 2nd choice partner, the a_{i+1}th person loses their 1st choice partner. Finally, a_n asks b_2 and b_2 leaves a_1 for a_n.
- We repeat this pattern, exhausting all the offers in the first $n-1$ columns. The last step in this procedure is when a_n asks b_{n-1} and b_{n-1} leaves a_1 for a_n.
- a_1 is the only unmatched member of group A and asks b_n, which accepts because b_n is everyone's last choice partner so it has not received any offers and is unmatched.
- The final matching is a_1 with its nth choice and everyone else in group A with its $(n-1)$th choice. In this case, the algorithm terminates when b_n receives an offer. Before, that all members of B except b_n (of which there are $n-1$) receive n offers.

4a

If we consider all the offers made by group A, we have n^2 offers. This is because there are n members of each group, and each member of group A can only make one offer to each member of group B. Once an offer has been made, the same offer cannot be made again. Similarly, if we consider all the offers made by group B, we have n^2 offers. Thus, the maximum number of offers is $n^2 + n^2 = 2n^2$, which is $O(n^2)$.

4b

We are able to produce a set of preferences for which an unstable match results. We demonstrate as follows:

Let the following be the preferences for group A:

- $a_1 : b_1, b_2$
- $a_2 : b_1, b_2$

Let the following be the preferences for group B:

- $b_1 : a_2, a_1$
- $b_2 : a_2, a_1$

The following offers take place with group A and group B alternating in making the offers, where group A starts at $t = 1$.

- a_1 makes an offer to b_1
  - accepted since b_1 unmatched
  - partial matching: $M = \{(a_1, b_1)\}$
- b_2 makes an offer to a_2
  - accepted since a_2 unmatched
  - partial matching: $M = \{(a_1, b_1), (a_2, b_2)\}$

The pairing (a_2, b_1) is unstable with respect to this matching. a_2 would prefer b_1 to its current partner b_2. Similarly, b_1 would prefer a_2 to its current partner a_1.

4c

Using $n = 6$ people in each group, we can construct a set of preferences and offers where we do not get a matching.

We propose the following preference lists for group A:

$$
\begin{array}{lcccccc}
a_1 : & b_4 & b_1 & b_2 & b_3 & b_5 & b_6 \\
a_2 : & b_2 & b_1 & b_3 & b_4 & b_5 & b_6 \\
a_3 : & b_3 & b_1 & b_2 & b_4 & b_5 & b_6 \\
a_4 : & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 \\
a_5 : & b_6 & b_5 & b_1 & b_2 & b_3 & b_4 \\
a_6 : & b_1 & b_6 & b_5 & b_2 & b_3 & b_4
\end{array}
$$

We propose the following preference lists for group B:

$$
\begin{array}{lllllll}
b_1: & a_4 & a_1 & a_2 & a_3 & a_5 & a_6 \\
b_2: & a_2 & a_1 & a_3 & a_4 & a_5 & a_6 \\
b_3: & a_3 & a_1 & a_2 & a_4 & a_5 & a_6 \\
b_4: & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 \\
b_5: & a_1 & a_5 & a_6 & a_2 & a_3 & a_4 \\
b_6: & a_6 & a_1 & a_5 & a_2 & a_3 & a_4
\end{array}
$$

Assume group A starts giving offers at $t = 1$.

- a_6 asks b_1
  - accepted since b_1 unmatched
  - partial matching: $M = \{(a_6, b_1)\}$
- b_6 asks a_6
  - rejected since a_6 prefers b_1 to b_6
  - partial matching: $M = \{(a_6, b_1)\}$
- a_2 asks b_2
  - accepted since b_2 unmatched
  - partial matching: $M = \{(a_2, b_2), (a_6, b_1)\}$
- b_6 asks a_1
  - accepted since a_1 unmatched
  - partial matching: $M = \{(a_1, b_6), (a_2, b_2), (a_6, b_1)\}$
- a_5 asks b_6
  - rejected since b_6 prefers a_1 to a_5
  - partial matching: $M = \{(a_1, b_6), (a_2, b_2), (a_6, b_1)\}$
- b_5 asks a_1
  - accepted since a_1 prefers b_5 to b_6
  - partial matching: $M = \{(a_1, b_5), (a_2, b_2), (a_6, b_1)\}$
- a_5 asks b_5
  - rejected since b_5 prefers a_1 to a_5
  - partial matching: $M = \{(a_1, b_5), (a_2, b_2), (a_6, b_1)\}$
- b_6 asks a_5
  - accepted since a_5 unmatched
  - partial matching: $M = \{(a_1, b_5), (a_2, b_2), (a_5, b_6), (a_6, b_1)\}$
- a_3 asks b_3
  - accepted since b_3 unmatched
  - partial matching: $M = \{(a_1, b_5), (a_2, b_2), (a_3, b_3), (a_5, b_6), (a_6, b_1)\}$
- b_4 asks a_1
  - accepted since a_1 prefers b_4 to b_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_5, b_6), (a_6, b_1)\}$

- a_4 asks b_1
  - accepted since b_1 prefers a_4 to a_6
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_5, b_6)\}$
- b_5 asks a_5
  - rejected since a_5 prefers b_6 to b_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_5, b_6)\}$
- a_6 asks b_6
  - accepted since b_6 prefers a_6 to a_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_6, b_6)\}$
- b_5 asks a_6
  - rejected since a_6 prefers b_6 to b_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_6, b_6)\}$
- a_5 asks b_1
  - rejected since b_1 prefers a_4 to a_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_6, b_6)\}$
- b_5 asks a_2
  - rejected since a_2 prefers b_2 to b_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_6, b_6)\}$
- a_5 asks b_2
  - rejected since b_2 prefers a_2 to a_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_6, b_6)\}$
- b_5 asks a_3
  - rejected since a_3 prefers b_3 to b_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_6, b_6)\}$
- a_5 asks b_3
  - rejected since b_3 prefers a_3 to a_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_6, b_6)\}$
- b_5 asks a_4
  - rejected since a_4 prefers b_1 to b_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_6, b_6)\}$
- a_5 asks b_4
  - rejected since b_4 prefers a_1 to a_5
  - partial matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_6, b_6)\}$

final matching: $M = \{(a_1, b_4), (a_2, b_2), (a_3, b_3), (a_4, b_1), (a_6, b_6)\}$

At this point, a_5 and b_5 have asked all members of group B and group A, respectively. They are the only members of each group left without a partner but have nobody left to ask that has not already rejected them. In this case, we are unable to find a partner for a_5 and b_5 with the above steps of this algorithm. Thus, this algorithm does not always produce a matching.