CS 70 Discrete Mathematics and Probability Theory
Spring 2020 DIS 1B

1 Stable Matching

Consider the set of jobs $J = \{1, 2, 3\}$ and the set of candidates $C = \{A, B, C\}$ with the following preferences.

Jobs	Candidates
1	A > B > C
2	B > A > C
3	A > B > C

Candidates	Jobs
A	2 > 1 > 3
B	1 > 3 > 2
C	1 > 2 > 3

Run the traditional propose-and-reject algorithm on this example. How many days does it take and what is the resulting pairing? (Show your work.)

Solution:

The algorithm takes 5 days to produce a matching. The resulting pairing is as follows. The circles indicate the job that a candidate picked on a given day (and rejected the rest).

$$\{(A,2),(B,1),(C,3)\}.$$

Candidate	Day 1	Day 2	Day 3	Day 4	Day 5
A	(1),3	(1)	1,(2)	(2)	(2)
B	(2)	2,(3)	(3)	(1),3	(1)
C					(3)

2 Propose-and-Reject Proofs

Prove the following statements about the traditional propose-and-reject algorithm.

(a) In any execution of the algorithm, if a candidate receives a proposal on day i, then she receives some proposal on every day thereafter until termination.

(b) In any execution of the algorithm, if a candidate receives no proposal on day i, then she receives no proposal on any previous day j, $1 \leq j < i$.

(c) In any execution of the algorithm, there is at least one candidate who only receives a single proposal. (Hint: use the parts above!)

Solution:

(a) The idea is to induct on the number of days passed so far.
Base case: Candidate C receives a proposal on day i
Inductive Step: Assume C receives a proposal on day $j \geq i$ from job J. We want to show she will also get a proposal on day $j+1$. There are two cases: C prefers J to all other offers, or C prefers some job J' to J. In the first case J proposes to C on day $j+1$ and in the second J' proposes to C on day $j+1$ so C receives at least one proposal on day $j+1$.

(b) One way is to use a proof by contradiction. Assume that a candidate receives no proposal on day i but did receive a proposal on some previous day j, $1 \leq j < i$. By the previous part, since the candidate received a proposal on day j, she must receive at least one proposal on every day after j. But $i > j$, so the candidate must have received a proposal on day i, contradicting our original assumption that she did not.

(c) Let's say the algorithm takes k days. This means that every candidate must have received a proposal on day k. However, this also means that there is at least one candidate C who does not receive a proposal on day $k-1$–if this were not the case, the algorithm would have already terminated on day $k-1$. Then from part (b), since C did not receive a proposal on day $k-1$, she didn't receive a proposal on any day before k. Furthermore, we know she got exactly one proposal on day k, since the algorithm terminated on that day. Thus, we have that C receives exactly one proposal throughout the entire run of the algorithm.

3 Be a Judge

For each of the following statements about the traditional stable matching algorithm with jobs proposing, indicate whether the statement is True or False and justify your answer with a short 2-3 line explanation:

(a) There is a set of preferences for n jobs and n candidates for $n > 1$, such that in a stable matching algorithm execution every job ends up with its least preferred candidate.

(b) In a stable matching instance, if job J and candidate C each put each other at the top of their respective preference lists, then J must be paired with C in every stable pairing.

(c) In a stable matching instance with at least two jobs and two candidates, if job J and candidate C each put each other at the bottom of their respective preference lists, then J cannot be paired with C in any stable pairing.

(d) For every $n > 1$, there is a stable matching instance for n jobs and n candidates which has an unstable pairing in which every unmatched job-candidate pair is a rogue couple.

Solution:

(a) **False**: If this were to occur it would mean that at the end of the algorithm, every job would have proposed to every candidate on its list and has been rejected $n-1$ times. This would also

require every candidate to reject $n-1$ jobs. We know this is impossible though, as we learned above that at least one candidate receives a single proposal. Thus, there must be at least one candidate who is not proposed to until the very last day.

(b) **True**: We give a simple proof by contradiction. Assume that J and C can put each other at the top of their respective preference lists, but J and C are not paired with each other in some stable pairing. Then we have a stable pairing which includes the pairings (J, C'), (J', C), for some job J' and candidate C'. However, J prefers C over its partner in this pairing, since C is at the top of his preference list. Similarly C prefers J over her current job. Thus (J,C) form a rogue couple, so the pairing is not stable. We have arrived at a contradiction.

Therefore if job J and candidate C put each other at the top of their respective preference lists, then J must be paired with C in a stable pairing.

(c) **False**: The key here is to realize that this is possible if job J and candidate C are at the bottom of everybody else's preference list as well. Consider the following example with the jobs :m and J and the candidates w and C. Suppose that their preference lists are as follows:

$$\begin{aligned} j &: c, C \\ J &: c, C \\ c &: j, J \\ C &: j, J \end{aligned}$$

It is clear that J and C are at the bottom of each other's preference lists; however, it is also true that (j,c) and (J,C) is a stable pairing (indeed, it is the only stable pairing). So, we have a contradiction to the statement: here is a stable marriage instance with at least two jobs :and two candidates, and job J and candidate C put each other at the bottom of their respective preference lists, but yet J and C are paired together in a stable pairing. A more concrete example is laid out in question 1.

(d) **True**: The key idea to this solution is that we want a set of preferences for which J_i and C_i like each other the least and to put J_i and C_i together. In this matching each unmatched couple is a rogue couple. More formally, suppose $n > 1$ and we have jobs :$J_1, ..., J_n$ and candidates $C_1, ..., C_n$. Further, assume that for $1 \leq i \leq n$, preference lists are as follows for every job J_i and candidate C_i:

$$\begin{aligned} & & \textit{highest} \Longrightarrow \textit{lowest} \\ J_i: & \quad C_i \;\; C_{i+1} \;\; C_{i+2} \;\; \ldots \;\; C_{i-1} \\ & & \textit{highest} \Longrightarrow \textit{lowest} \\ C_i: & \quad J_i \;\; J_{i-1} \;\; J_{i-2} \;\; \ldots \;\; J_{i+1} \end{aligned}$$

Note that the indices are taken modulo n, so if i refers to $n+1$ in the preference lists above, it is really referring to 1. The idea in this construction is that there is a fixed ordering of jobs into a cycle, and a fixed ordering of candidates into another cycle. Every job's preference list complies to the ordering of candidates into the cycle, with the only difference between different jobs' preferences being where in the ordering the preference list begins. The analogous situation holds for candidates's preference lists.

Now consider the unstable pairing in which each job J_i, $1 \leq i \leq n$ is paired as (J_i, C_{i-1}). (J_1 is paired to C_n.) We claim every unmatched job-candidate pair is a rogue couple.

In this pairing, every job J_i is paired with candidate C_{i-1} at the bottom of its preference list, and every candidate C_i is paired with job J_{i+1} at the bottom of her preference list. Thus every job prefers any candidate it has not been matched to over its partner, and likewise for candidates. So any unmatched pair (J, C) is a rogue couple.

4 Universal Preference

Suppose that preferences in a stable matching instance are universal: all n jobs share the preferences $C_1 > C_2 > \cdots > C_n$ and all candidates share the preferences $J_1 > J_2 > \cdots > J_n$.

(a) What pairing do we get from running the algorithm with jobs proposing? Can you prove this happens for all n?

(b) What pairing do we get from running the algorithm with candidates proposing?

(c) What does this tell us about the number of stable pairings?

Solution:

(a) The pairing results in (C_i, J_i) for each $i \in \{1, 2, ..., n\}$.

This result can be proved by induction:

Our base case is when $n = 1$, so the only pairing is (C_1, J_1), and thus the base case is trivially true.
Now assume this is true for some $n \in \mathbb{N}$.
On the first day with $n+1$ jobs and $n+1$ candidates, all $n+1$ jobs will propose to C_1. C_1 prefers J_1 the most, and the rest of the jobs will be rejected. This leaves a set of n unpaired jobs and n unpaired candidates who all have the same preferences (after the pairing of (C_1, J_1)). By the process of induction, this means that every i^{th} preferred candidate will be paired with the i^{th} preferred job.

(b) The pairings will again result in (J_i, C_i) for each $i \in \{1, 2, ..., n\}$. This can be proved by induction in the same as above, but replacing "job" with "candidate" and vice-versa.

(c) We know that job-proposing produces a candidate-pessimal stable pairing. We also know that candidate-proposing produces a candidate-optimal stable pairing. We found that candidate-optimal and candidate-pessimal pairings are the same. This means that there is only one stable pairing, since both the best and worst pairings (for candidates) are the same pairings.