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.)

Answer:

Days	Pairs	Rejected/Left
1	$1 \to A$, $2 \to B$, $3 \nrightarrow A$	3,C
2	$1 \to A$, $3 \to B$, $2 \nrightarrow B$	2,C
3	$2 \to A$, $3 \to B$, $1 \nrightarrow A$	1,C
4	$1 \to B$, $2 \to A$, $3 \nrightarrow B$	3,C
5	$1 \to B$, $2 \to A$, $3 \to C$	None

Final pairs: {(1,B),(2,A),(3,C)}

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.**

Proof. We proceed by induction on day j.

Base case($k = i$). It's obviously true.

Induction Hypothesis. We assume that for some day k after i and before termination, the claim is true.

Induction Step. We prove for k+1. Since the offer can't be withdrawn and the candidate has recieved an offer from some J, it will only have the following two cases.

Case 1. He receives another offers which are superior to J on his list. Then he will reject the current J and accept the another one.

Case 2. He receives another offers which are inferior to J on his list or he doesn't receive any other offers. In that case, he still keep his current offer choice, i.e. J.

Thus, we conclude the claim is true for k+1 and accordingly, for any day j after i and before termination, he receives some proposal. □

(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, $1 \leqslant j < i$.**

Proof. We proceed by contradiction. We claim that if a candidate receives no proposal on day i, he receives some proposal on some day j, $1 \leqslant j < i$. We assume the very day is day t. According to 2.(a), we know that the candidate will receive some proposals on every day thereafter until termination, which means he will receive a proposal on day i as well. That's contradictory to the fact. So the hypothesis is false and the former claim is true. □

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

We assume the algorithm goes k days, which means all the candidate receives a proposal at day k. That also means on day k-1 there must exist at least one candidate C who don't receive a proposal otherwise the algorithm ends on the day k-1. According to 2.(b), we know that C doesn't receive a proposal before k, i.e. through the entire process C only receives a proposal on day k, which has proved there is at least one candidate who only receives a single proposal. □

3 Be a Judge

By stable matching instance, we mean a set of jobs and candidates and their preference lists. For each of the following statements, indicate whether the statement is True or False and justify your answer with a short 2-3 line explanation:

(a) ***There is a stable matching instance for n jobs and n candidates for n>1, such that in a stable matching algorithm with jobs proposing, every job ends up with its least preferred candidate.**

False.

Explanation. If this is true, that means in the algorithm every job has been rejected n-1 times, i.e. every candidate reject n-1 jobs. However, according to 2.(c), we know that there must have a candidate who only receives a proposal on the last day. So the claim is false.

(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.**

True.

Proof. We proceed by contradiction. We assume the claim is false, which means there exists a stable matching where J and C aren't paired, i.e. J pairs with C^* and C pairs with J^*. Nevertheless, J prefers C to C^* and C prefers J to J^* so (J,C) becomes a rogue pair which is contradictory to the stable matching. □

(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.**

False.

Explanation. If all the other candidates put J at the bottom of their preference lists and all the jobs put C at the bottom of their preference lists, and we assume except for J and C there don't exist rogue pairs, it isn't possible that J or C can find a candidate or job to form the rogue pair.

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

True.

Proof. We make a proof by construction. The construction is as follows:

J_1	$\cdots > C_1$
J_2	$\cdots > C_2$
$\vdots$	$\vdots$
J_i	$\cdots > C_i$
$\vdots$	$\vdots$
J_n	$\cdots > C_n$

C_1	$\cdots > J_1$
C_2	$\cdots > J_2$
$\vdots$	$\vdots$
C_i	$\cdots > J_i$
$\vdots$	$\vdots$
C_n	$\cdots > J_n$

And we pair the job and candidate according to the index. Then for every pair $(J_i, C_j)(i \neq j)$, it's easy to find it's a rogue pair according to our preference table.

4 Pairing Up

*** Prove that for every even $n \geqslant 2$, there exists an instance of the stable matching problem with n jobs and n candidates such that the instance has at least $2^{\frac{n}{2}}$ distinct stable matchings.**

Proof. We proceed by construction of such instance.

Noting that n is even, we can put job 2k-1 and 2k into a pair and candidate 2k-1 and 2k into a pair, where $k = \frac{n}{2}$. Then we can construct as follows:

J_1	$C_1 > C_2 > \cdots$
J_2	$C_2 > C_1 > \cdots$
J_3	$C_3 > C_4 > \cdots$
J_4	$C_4 > C_3 > \cdots$
$\vdots$	$\vdots$
J_{2k-1}	$C_{2k-1} > C_{2k} > \cdots$
J_{2k}	$C_{2k} > C_{2k-1} > \cdots$

C_1	$J_2 > J_1 > \cdots$
C_2	$J_1 > J_2 > \cdots$
C_3	$J_4 > J_3 > \cdots$
C_4	$J_3 > J_4 > \cdots$
$\vdots$	$\vdots$
C_{2k-1}	$J_{2k} > J_{2k-1} > \cdots$
C_{2k}	$J_{2k-1} > J_{2k} > \cdots$

Now we consider the Job 2i-1, Job 2i and Candidate 2i-1, Candidate 2i $(1 \leqslant i \leqslant k)$. They can be paired in two ways:

Case 1: $\cdots (J_{2i-1}, C_{2i-1})(J_{2i}, C_{2i}) \cdots$

Case 2: $\cdots (J_{2i-1}, C_{2i})(J_{2i}, C_{2i-1}) \cdots$

According to the preference list, it's easy to know that it's impossible to form rogue pair between other candidate with J_{2i-1}, J_{2i} and so do candidates C_{2i-1}, C_{2i}. Now we consider the two pairs. Because C_{2i-1} is on the top of J_{2i-1}'s preference list and C_{2i} is on the top of J_{2i}'s preference list, in case 1 there isn't any rogue pair. Because J_{2i-1} is on the top of C_{2i}'s preference list and J_{2i} is on the top of C_{2i-1}'s preference list, there isn't any rogue pair in case 2. That means every such pair: job 2i-1 and job 2i, candidate 2i-1 and candidate 2i, there exists two stable matching choices. So for all the k pairs, there exists 2^k distinct stable matchings. □