Stable Matching

$J: \{j_1, j_2, \ldots, j_n\}$ $C: \{c_1, c_2, \ldots, c_n\}$

Rogue Couple: Some pair (j_x, c_y) that prefer each other over their matched partner.

Propose & Reject

Morning: Job proposes to top Candidate on their list who hasn't rejected them.

Midday: Each candidate rejects all offers they recieve, save for their most preferred (which they "leave on a string")

Night: All jobs rejected cross off the Candidate they proposed to in the morning (same Candidate who rejected them during the day).

Stable: No rogue couples in output matching of PAR.

Improvement Lemma: Each candidate's job on string only gets more and more preferred.

Proofs Advice: Contradiction, Induction, $\underbrace{\text{WOP}}_{\text{any nonempty subset of } \mathbb{N} \text{ has a smallest element.}}$

CS 70 Discrete Mathematics and Probability Theory
Spring 2021 **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.)

Candidates

Day	A	B	C
1:	1, ~~3~~	2	
2:	1	~~2~~, 3	
3:	~~1~~, 2	3	
4:	2	1, ~~3~~	
5:	2	1	3

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

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.

Base Case: receive proposal on day 1 ✓

I.H.: For some $k \geq i$, receive on day k.

I.S.: Want to prove receive some proposal on day k+1.

By WOP, on day k we select favorite, J.

day k+1: J proposes again → nonempty set of proposals.

Select favorite, J*

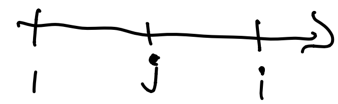
J* ≥ J

Improvement Lemma

(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, $1 \leq j < i$.

Does receive proposal on some previous day j, $1 \leq j < i$

⇒ will receive proposal on day 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!)

d days

(d-1)th day: if everyone has an offer

⇒ by def., algorithm is done.

⇒ some cand. didn't have an offer

⇒ that person didn't have offer on any day j from $1 \leq j < d-1$

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.

False

J_1: $C_{i_1} > \cdots > C_{j_1}$

J_2: $C_{i_2} > \cdots > C_{j_2}$

J_3: $C_{i_3} > \cdots > C_{j_3}$

$\vdots$

J_n: $C_{i_n} > \cdots > C_{j_n}$

each job rejected by n-1 candidates.

↓

each candidate rejected n-1 jobs over the course of the algorithm.

(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

Contradiction: Assume

$J - C^*$

$J^* - C$

not stable

↓

(J, C) is rogue couple

C: $J > J^*$ b/c J is my #1

J: $C > C^*$ b/c C is my #1

(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

Jobs:

$1: A > B$

$2: A > B$

Candidates:

$A: 1 > 2$

$B: 1 > 2$

$\{(A, 1), (B, 2)\}$ is stable

(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.

True

$1: \ldots > C_1$

$2: \ldots > C_2$

$\vdots$

$n: \ldots > C_n$

$C_1: \ldots > 1$

$C_2: \ldots > 2$

$\vdots$

$C_n: \ldots > n$

$(C_1, 1), (C_2, 2), \ldots, (C_n, n)$