CS 70 Discrete Mathematics and Probability Theory
Spring 2022 Koushik Sen and Satish Rao

HW 02

1 Pairing Up

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

Solution:

To prove that there exists such a stable matching instance for any even $n \geq 2$, we just need to show how to construct such an instance.

The idea here is that we can create pairs of jobs and pairs of candidates: pair up job $2k-1$ and $2k$ into a pair and candidate $2k-1$ and $2k$ into a pair, for $1 \leq k \leq n/2$ (you might come to this idea since we are asked to prove this for *even* n).

For n, we have $n/2$ pairs. Choose the preference lists such that the kth pair of jobs rank the kth pair of candidates just higher than the $(k+1)$th pair of candidates (the pairs wrap around from the last pair to the first pair), and the kth pair of candidates rank the kth pair of jobs just higher than the $(k+1)$th pair of jobs. Within each pair of pairs (j, j') and (c, c'), let j prefer c, let j' prefer c', let c prefer j', and let c' prefer j.

Our construction thus results in preference lists like follows:

J_1	$C_1 > C_2 > \ldots$
J_2	$C_2 > C_1 > \ldots$
$\vdots$	$\vdots$
J_{2k-1}	$C_{2k-1} > C_{2k} > \ldots$
J_{2k}	$C_{2k} > C_{2k-1} > \ldots$
$\vdots$	$\vdots$
J_{n-1}	$C_{n-1} > C_n > \ldots$
J_n	$C_n > C_{n-1} > \ldots$

C_1	$J_2 > J_1 > \ldots$
C_2	$J_1 > J_2 > \ldots$
$\vdots$	$\vdots$
C_{2k-1}	$J_{2k} > J_{2k-1} > \ldots$
C_{2k}	$J_{2k-1} > J_{2k} > \ldots$
$\vdots$	$\vdots$
C_{n-1}	$J_n > J_{n-1} > \ldots$
C_n	$J_{n-1} > J_n > \ldots$

Each match will have jobs in the kth pair paired to candidates in the kth pair for $1 \leq k \leq n/2$.

A job j in pair k will never form a rogue couple with any candidate c in pair $m \neq k$. If $m > k$, then c prefers her current partner in the kth pair to j. If $m < k$, then j prefers its current partner in the kth pair to c. Then a rogue couple could only exist in the same pair - but this is impossible since exactly one of either j or c must be matched to their preferred choice in the pair.

Since each job in pair k can be stably matched to either candidate in pair k, and there are $n/2$ total pairs, the number of stable matchings is $2^{n/2}$.

2 Nothing Can Be Better Than Something

In the stable matching problem, suppose that some jobs and candidates have hard requirements and might not be able to just settle for anything. In other words, each job/candidate prefers being unmatched rather than be matched with those below a certain point in their preference list. The term entity to refer to a candidate/job. A matching could ultimately have to be partial, i.e., some entities would and should remain unmatched.

Consequently, the notion of stability here should be adjusted a little bit to capture the autonomy of both jobs to unilaterally fire employees and/or employees to just walk away. A matching is stable if

- there is no matched entity who prefers being unmatched over being with their current partner;
- there is no matched/filled job and unmatched candidate that would both prefer to be matched with each other over their current status;
- there is no matched job and matched candidate that would both prefer to be matched with each other over their current partners; and
- similarly, there is no unmatched job and matched candidate that would both prefer to be matched with each other over their current status;
- there is no unmatched job and unmatched candidate that would both prefer to be with each other over being unmatched.

(a) Prove that a stable pairing still exists in the case where we allow unmatched entities.

(HINT: You can approach this by introducing imaginary/virtual entities that jobs/candidates "match" if they are unmatched. How should you adjust the preference lists of jobs/candidates, including those of the newly introduced imaginary ones for this to work?)

(b) As you saw in the lecture, we may have different stable matchings. But interestingly, if an entity remains unmatched in one stable matching, it/she must remain unmatched in any other stable matching as well. Prove this fact by contradiction.

Solution:

(a) We form an instance of the standard stable matching problem as follows. Following, the hint, we introduce an imaginary mate, r_e, (let's call it a robot) for each entity. Note that we introduce one robot for each entity, i.e. there are as many robots as there are candidates+jobs. For simplicity let us say each robot, r_e, is owned by the entity, e, we introduce it for.

Each robot, r_e, prefers its owner e, i.e. it puts its owner at the top of its preference list. The rest of its preference list is arbitrary and includes all other owners and the robots of other types of entities. An entity e of a robot, r_e puts it in their preference list exactly after the last entity they are willing to match with. i.e. owners like their robots more than entities they are not willing to match, but less than entities they like to match. The other robots can be placed in arbitrary order in e's list.

For this instance of the stable matching problem which we refer to as the robot instance, I, the propose and reject algorithm will give us a stable matching, M.

To extract the desired partial matching, we simply remove all pairs in M with at least one robot (two robots can match each other). We refer to each entity which is not matched as single. We observe, an entity, e, will never be matched with another entity's robot,r_e, because then e and its robot, r_e, would form a rogue couple (r_e prefers e to other owners, and e prefers r_e more than other robots). It remains to show this is a stable matching as specified in the problem as follows:

- there is no matched entity,e, who prefers being unmatched over being with their current partner since the e and r_e would form a rogue couple in M;
- there is no matched/filled job, j, and unmatched candidate, c, that would both prefer to be matched with each other since otherwise j and c would form a rogue couple in M;
- there is no matched job j and matched candidate c that would both prefer to be matched with each other over their current partners since then j and c would form a rogue couple in M;
- similarly, there is no unmatched job, j, and matched candidate, c, that would both prefer to be matched with each other over their current status since j and c would be a rogue couple in M;
- there is no unmatched job, j, and unmatched candidate,c, that would both prefer to be with each other over being unmatched since j and c would be a rogue couple in M.

Thus, the resulting partial matching is stable.

(b) We will perform proof by contradiction. Assume that there exists some job j_1 who is paired with a candidate c_1 in stable pairing S and unpaired in stable pairing T. The stable pairing S where j_1 and c_1 are paired means j_1 and c_1 both prefer to be with each other over being single. Since T is a stable pairing and j_1 is unpaired, c_1 must be paired in T with a job j_2 whom she prefers over j_1. (If c_1 were unpaired or paired with a job she does not prefer over j_1, then (j_1, c_1) would be a rogue couple in T, which is a contradiction.)

Since j_2 is paired with c_1 in T, it must be paired in S with some candidate c_2 whom j_2 prefers over c_1. This process continues (c_2 must be paired with some j_3 in T, j_3 must be paired with some c_3 in S, etc.) with the pattern that (c_i, j_i) is in S and (j_i, c_{i-1}) is in T. At some time i, one must encounter j_1 in this process as there are a finite number of jobs. A this point $(j_1 = j_i, c_{i-1})$ is in T, which implies that j_1 is matched in T.

A similar argument can be used for candidates.

This contradicts the assumption that j_1 is matched in T. Since no job or candidate can be paired in one stable pairing and unpaired in another, every job or candidate must be either paired in all stable pairings or unpaired in all stable pairings.

3 A Better Stable Pairing

In this problem we examine a simple way to *merge* two different solutions to a stable matching problem. Let R, R' be two distinct stable pairings. Define the new pairing $R \wedge R'$ as follows:

> For every job j, j's partner in $R \wedge R'$ is whichever is better (according to j's preference list) of their partners in R and R'.

Also, we will say that a job/candidate *prefers* a pairing R to a pairing R' if they prefers their partner in R to their partner in R'. We will use the following example:

jobs	preferences	candidates	preferences
A	1>2>3>4	1	D>C>B>A
B	2>1>4>3	2	C>D>A>B
C	3>4>1>2	3	B>A>D>C
D	4>3>2>1	4	A>B>D>C

(a) $R = \{(A,4),(B,3),(C,1),(D,2)\}$ and $R' = \{(A,3),(B,4),(C,2),(D,1)\}$ are stable pairings for the example given above. Calculate $R \wedge R'$ and show that it is also stable.

(b) Prove that, for any pairings R, R', no job prefers R or R' to $R \wedge R'$.

(c) Prove that, for any stable pairings R, R' where j and c are partners in R but not in R', one of the following holds:

- j prefers R to R' and c prefers R' to R; or
- j prefers R' to R and c prefers R to R'.

[*Hint*: Let J and C denote the sets of jobs and candidates respectively that prefer R to R', and J' and C' the sets of jobs and candidates that prefer R' to R. Note that $|J| + |J'| = |C| + |C'|$. (Why is this?) Show that $|J| \leq |C'|$ and that $|J'| \leq |C|$. Deduce that $|J'| = |C|$ and $|J| = |C'|$. The claim should now follow quite easily.]

(You may assume this result in the next part even if you don't prove it here.)

(d) Prove an interesting result: for any stable pairings R, R', (i) $R \wedge R'$ is a pairing [*Hint*: use the results from (c)], and (ii) it is also stable.

Solution:

(a) $R \wedge R' = \{(A,3),(B,4),(C,1),(D,2)\}$. This pairing can be seen to be stable by considering the different combinations of jobs and candidates. For instance, A prefers 2 to their current partner 3. However, 2 prefers her current partner D to A. Similarly, A prefers 1 the most, but 1 prefers her current partner C to A. We can prove the stability of this pairing by considering the remaining pairs like this.

(b) Let j be a job, and let their partners in R and R' be c and c' respectively, and without loss of generality, let $c > c'$ in j's list. Then their partner in $R \wedge R'$ is c, whom they prefer over c'. However, for j to prefer R or R' over $R \wedge R'$, j must prefer c or c' over c, which is not possible (since $c > c'$ in their list).

(c) Let J and C denote the sets of jobs and candidates respectively that prefer R to R', and J' and C' the sets of jobs and candidates that prefer R' to R. Note that $|J| + |J'| = |C| + |C'|$, since the left-hand side is the number of jobs who have different partners in the two pairings, and the right-hand side is the number of candidates who have different partners.

Now, in R there cannot be a pair (j,c) such that $j \in J$ and $c \in C$, since this will be a rogue couple in R'. Hence the partner in R of every jobs in J must lie in C', and hence $|J| \leq |C'|$. A similar argument shows that every jobs in J' must have a partner in R' who lies in C, and hence $|J'| \leq |C|$.

Since $|J| + |J'| = |C| + |C'|$, both these inequalities must actually be tight, and hence we have $|J'| = |C|$ and $|J| = |C'|$. The result is now immediate: if the jobs j is partners with the candidate c in one but not both pairings, then

- either $j \in J$ and $c \in C'$, i.e., j prefers R to R' and c prefers R' to R,
- or $j \in J'$ and $c \in C$, i.e., j prefers R' to R and c prefers R to R'.

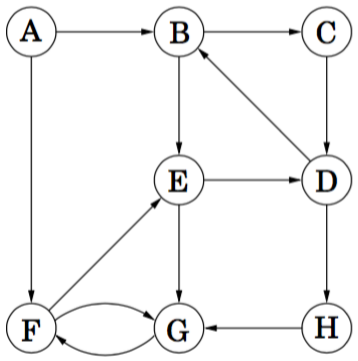
(d) (i) If $R \wedge R'$ is not a pairing, then it is because two jobs get the same candidate, or two candidates get the same job. Without loss of generality, assume it is the former case, with $(j,c) \in R$ and $(j',c) \in R'$ causing the problem. Hence j prefers R to R', and j' prefers R' to R. Using the results of the previous part would imply that c would prefer R' over R, and R over R' respectively, which is a contradiction.

(ii) Now suppose $R \wedge R'$ has a rogue couple (j,c). Then j strictly prefers c to their partners in both R and R'. Further, c prefers j to her partner in $R \wedge R'$. Let c's partners in R and R' be j_1 and j_2. If she is finally matched to j_1, then (j,c) is a rogue couple in R; on the other hand, if she is matched to j_2, then (j,c) is a rogue couple in R'. Since these are the only two choices for c's partner, we have a contradiction in either case.

4 Graph Basics

In the first few parts, you will be answering questions on the following graph G.

(a) What are the vertex and edge sets V and E for graph G?



(b) Which vertex has the highest in-degree? Which vertex has the lowest in-degree? Which vertices have the same in-degree and out-degree?

(c) What are the paths from vertex B to F, assuming no vertex is visited twice? Which one is the shortest path?

(d) Which of the following are cycles in G?

i. $(B,C),(C,D),(D,B)$
ii. $(F,G),(G,F)$
iii. $(A,B),(B,C),(C,D),(D,B)$
iv. $(B,C),(C,D),(D,H),(H,G),(G,F),(F,E),(E,D),(D,B)$

(e) Which of the following are walks in G?

i. (E,G)
ii. $(E,G),(G,F)$
iii. $(F,G),(G,F)$
iv. $(A,B),(B,C),(C,D),(H,G)$
v. $(E,G),(G,F),(F,G),(G,C)$
vi. $(E,D),(D,B),(B,E),(E,D),(D,H),(H,G),(G,F)$

(f) Which of the following are tours in G?

i. (E,G)
ii. $(E,G),(G,F)$
iii. $(F,G),(G,F)$
iv. $(E,D),(D,B),(B,E),(E,D),(D,H),(H,G),(G,F)$
v. $(B,C),(C,D),(D,H),(H,G),(G,F),(F,E),(E,D),(D,B)$

In the following three parts, let's consider a general undirected graph G with n vertices ($n \geq 3$). If true, provide a short proof. If false, show a counterexample.

(g) True/False: If each vertex of G has degree at most 1, then G does not have a cycle.

(h) True/False: If each vertex of G has degree at least 2, then G has a cycle.

(i) True/False: If each vertex of G has degree at most 2, then G is not connected.

Solution:

(a) A graph is specified as an ordered pair $G = (V, E)$, where V is the vertex set and E is the edge set.

$$V = \{A, B, C, D, E, F, G, H\},$$
$$E = \{(A,B), (A,F), (B,C), (B,E), (C,D), (D,B), (D,H), (E,D), (E,G), (F,E), (F,G), (G,F), (H,G)\}.$$

(b) G has the highest in-degree (3). A has the lowest in-degree (0).

$\{B, C, D, E, F, H\}$ all have the same in-degree and out-degree. H and C has in-degree (out-degree) equal to 1 and the other four have in-degree (out-degree) equal to 2.

(c) There are three paths:
$(B,C), (C,D), (D,H), (H,G), (G,F)$
$(B,E), (E,D), (D,H), (H,G), (G,F)$
$(B,E), (E,G), (G,F)$
The first two have length 5, while the last one has length 3, so the last one is the shortest path.

(d) A cycle is a path that starts and ends at the same point. This means that (iii) is not a cycle, since it starts at A but ends at B. In addition, all the vertices $\{v_1, \ldots, v_n\}$ in the cycle $(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n)$ should be distinct, so (iv) is not a cycle. The correct answers are (i) and (ii).

(e) A walk consists of any sequence of edges such that the endpoint of each edge is the same as the starting vertex of the next edge in the sequence. Example (iv) does not fit this definition–even though it uses only valid edges, the endpoint of the second to last edge in D, while the start point of the next edge is H. Example (v) also is not a walk, since it tries to walk from G to C as its last step, but there is no such edge. All the rest are walks.

(f) A tour is a walk that has the same start and end vertex; examples (iii) and (v) satisfy this definition. Note in part (d), we already said that (iii) was a cycle–and indeed, all cycles are also tours.

(g) True. In order for there to be a cycle in G starting and ending at some vertex v, we would need at least two edges incident to v: one to leave v at the start of the cycle, and one to return to v at the end. If every vertex has degree at most 1, no vertex has two or more edges incident on it, so no vertex is capable of acting as the start and end point of a cycle.

(h) True. Consider starting a walk at some vertex v_0, and at each step, walking along a previously untraversed edge, stopping when we first visit some vertex w for the second time. If this

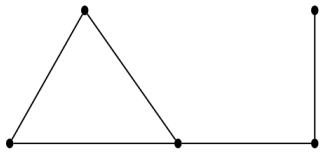
process terminates, the part of our walk from the first time we visited w until the second time is a cycle. Thus, it remains only to argue this process always terminates.

Each time we take a step from some vertex v, since we are not stopping, we must have visited that vertex exactly once and not yet left. It follows that we have used at most one edge incident with v (either we started at v, or we took an edge into v). Since v has degree at least 2, there must be another edge leaving v for us to take.

(i) False. For example, a 3-cycle (triangle) is connected and every vertex has degree 2.

5 Degree Sequences

The *degree sequence* of a graph is the sequence of the degrees of the vertices, arranged in descending order, with repetitions as needed. For example, the degree sequence of the following graph is $(3,2,2,2,1)$.



For each of the parts below, determine if there exists a simple undirected graph G (i.e. a graph without self-loops and multiple-edges) having the given degree sequence. Justify your claim.

(a) $(3,3,2,2)$

(b) $(3,2,2,2,2,1,1)$

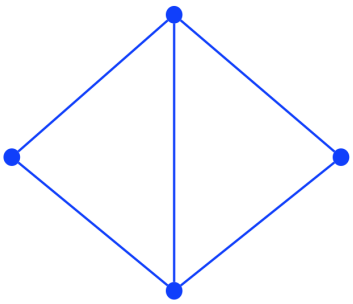
(c) $(6,2,2,2)$

(d) $(4,4,3,2,1)$

Solution:

(a) **Yes**

The following graph has degree sequence $(3,3,2,2)$.



(b) **No**

For any graph G, the number of vertices that have odd degree is even (since the sum of degrees is twice the number of edges). The given degree sequence has 3 odd degree vertices.

(c) **No**

The total number of vertices is 4. Hence there cannot be a vertex with degree 6.

(d) **No**

The total number of vertices is 5. Hence, any degree 4 vertex must have an edge with every other vertex. Since there are two degree 4 vertices, there cannot be a vertex with degree 1.

6 Proofs in Graphs

Please prove or disprove the following claims.

(a) On the axis from San Francisco traffic habits to Los Angeles traffic habits, Old California is more towards San Francisco: that is, civilized. In Old California, all roads were one way streets. Suppose Old California had n cities ($n \geq 2$) such that for every pair of cities X and Y, either X had a road to Y or Y had a road to X. Prove or disprove that there existed a city which was reachable from every other city by traveling through at most 2 roads.

[*Hint:* Induction]

(b) In lecture, we have shown that a connected undirected graph has an Eulerian tour if and only if every vertex has even degree.

Consider a connected graph G with n vertices which has exactly $2m$ vertices of odd degree, where $m > 0$. Prove or disprove that there are m walks that *together* cover all the edges of G (i.e., each edge of G occurs in exactly one of the m walks, and each of the walks should not contain any particular edge more than once).

Solution:

(a) We prove this by induction on the number of cities n.

Base case For $n = 2$, there's always a road from one city to the other.

Inductive Hypothesis When there are k cities, there exists a city c that is reachable from every other city by traveling through at most 2 roads.

Inductive Step Consider the case where there are $k+1$ cities. Remove one of the cities d and all of the roads to and from d. Now there are k cities, and by our inductive hypothesis, there exists some city c which is reachable from every other city by traveling through at most 2 roads. Let A be the set of cities with a road to c, and B be the set of cities two roads away from c. The inductive hypothesis states that the set S of the k cities consists of $S = \{c\} \cup A \cup B$.

Now add back d and all roads to and from d. Between d and every city in S, there must be a road from one to the other. If there is at least one road from d to $\{c\} \cup A$, c would still be reachable from d with at most 2 road traversals. Otherwise, if all roads from $\{c\} \cup A$ point to d, d will be reachable from every city in B with at most 2 road traversals, because every city in B can take one road to go to a city in A, then take one more road to go to d. In either case there exists a city in the new set of $k+1$ cities that is reachable from every other city by traveling at most 2 roads.

(b) We split the $2m$ odd-degree vertices into m pairs, and join each pair with an edge, adding m more edges in total. (Here, we allow for the possibility of multi-edges, that is, pairs of vertices with more than one edge between them.) Notice that now all vertices in this graph are of even degree. Now by Euler's theorem the resulting graph has an Eulerian tour. Removing the m added edges breaks the tour into m walks covering all the edges in the original graph, with each edge belonging to exactly one walk.

7 Bipartite Graphs

An undirected graph is bipartite if its vertices can be partitioned into two disjoint sets L, R such that each edge connects a vertex in L to a vertex in R (so there does not exist an edge that connects two vertices in L or two vertices in R).

(a) Suppose that a graph G is bipartite, with L and R being a bipartite partition of the vertices. Prove that $\sum_{v \in L} \deg(v) = \sum_{v \in R} \deg(v)$.

(b) Suppose that a graph G is bipartite, with L and R being a bipartite partition of the vertices. Let s and t denote the average degree of vertices in L and R respectively. Prove that $s/t = |R|/|L|$.

(c) Prove that a graph is bipartite if and only if it can be 2-colored. (A graph can be 2-colored if every vertex can be assigned one of two colors such that no two adjacent vertices have the same color).

Solution:

(a) Since G is bipartite, each edge connects one vertex in L with a vertex in R. Since each edge contributes equally to $\sum_{v \in L} \deg(v)$ and $\sum_{v \in R} \deg(v)$, we see that these two values must be equal.

(b) By part (a), we know that $\sum_{v \in L} \deg(v) = \sum_{v \in R} \deg(v)$. Thus $|L| \cdot s = |R| \cdot t$. A little algebra gives us the desired result.

(c) Given a bipartite graph, color all of the vertices in L one color, and all of the vertices in R the other color. Conversely, given a 2-colored graph (call the colors red and blue), there are no edges between red vertices and red vertices, and there are no edges between blue vertices and blue vertices. Hence, take L to be the set of red vertices and R to be the set of blue vertices. We see that the graph is bipartite.