CS 70 Spring 2019 — Discrete Mathematics and Probability Theory — Satish Rao and Babak Ayazifar — HW 2

1 Induction on Reals

Induction is always done over objects like natural numbers, but in some cases we can leverage induction to prove things about real numbers (with the appropriate mapping). We will attempt to prove the following by leveraging induction and finding an appropriate mapping.

Bob the Bug is on a window extending to the ground, trying to escape Sally the Spider. Sally has built her web from the ground to 2 inches up the window. Every second, Bob jumps 1 inch vertically up the window, then loses grip and falls to half his vertical height.

Prove that no matter how high Bob starts up the window, he will always fall into Sally's net in a finite number of seconds.

Solution: The basic idea is: First, prove directly that Bob will be netted if he starts ≤ 3 inches above the ground. Then, prove the inductive step: Given he dies if he starts $\leq n$ inches, show he also dies if starts $\leq n+1$ inches above the ground.
When working with natural numbers, we use induction to prove certain claims for discrete values (natural numbers), $n, n+1, \ldots$ and so on. In this case, since we want our claim to hold true for a subset of the real numbers, we have assumed our claim to hold true for all real numbers $\leq n$ - thereby creating our mapping. Therefore, if we can prove that for every natural number n, falling from height $\leq n$ means Bob will fall into the net, we will prove our claim for the entire subset of real numbers we want.
Let Bob's height up the window at time i be $x_i \in \mathbb{R}$. He starts at height $x_0 > 0$, and we have $x_{i+1} = (x_i + 1)/2$.

Proof: We prove this by induction on n.
Base Case: If he starts at height $x_0 \leq 3$, $x_1 = (x_0+1)/2 \leq 2$. So he will fall into the net in finite time (within the next second, in fact).

Inductive Hypothesis: Assume he falls into the net in finite time if he starts $x_0 \leq n$ inches above the ground.
Note: We are proceeding to do strong induction here as we are assuming our inductive hypothesis to hold true for all real numbers less than or equal to n which includes all the natural numbers less than or equal to n.

Inductive Step: We want to show: If he starts $x_0 \leq n+1$ inches above the ground, then he will also be netted in finite time.
If he starts at $x_0 \leq 3$ inches, the base case directly applies. Otherwise we have (for $x_0 > 3$):

$$\Delta = x_0 - x_1 = x_0 - (x_0+1)/2 = x_0/2 - 1/2 > 1$$

In other words, for all $x_0 > 3$, Bob falls more than 1 inch total in the first second. Therefore if $x_0 \le n+1$, we have $x_1 = x_0 - \Delta \le n+1-\Delta \le n$. So Bob will be $x_1 \le n$ inches high after 1 second after which point we know (by the inductive hypothesis) he dies in finite time.

2 Grid Induction

Pacman is walking on an infinite 2D grid. He starts at some location $(i, j) \in \mathbb{N}^2$ in the first quadrant, and is constrained to stay in the first quadrant (say, by walls along the x and y axes). Every second he does one of the following (if possible):

(i) Walk one step down, to $(i, j-1)$.

(ii) Walk one step left, to $(i-1, j)$.

For example, if he is at $(5,0)$, his only option is to walk left to $(4,0)$; if Pacman is instead at $(3,2)$, he could walk either to $(2,2)$ or $(3,1)$.

Prove by induction that no matter how he walks, he will always reach $(0,0)$ in finite time. (*Hint*: Try starting Pacman at a few small points like $(2,1)$ and looking all the different paths he could take to reach $(0,0)$. Do you notice a pattern?)

Solution:

Following the hint, we notice that it seems as though Pacman takes $i+j$ seconds to reach $(0,0)$ if he starts in position (i, j), regardless of what path he takes. This would imply that he reaches $(0,0)$ in a finite amount of time since $i+j$ is a finite number. Thus, if we can prove this stronger statement, we'll also have proved that Pacman reaches $(0,0)$ in finite time. In order to simplify the induction, we will induct on the quantity $i+j$ rather than inducting on i and j separately.

Base Case: If $i+j=0$, we know that $i=j=0$, since i and j must be non-negative. Hence, we have that Pacman is already at position $(0,0)$ and so will take $0=i+j$ steps to get there.

Inductive Hypothesis: Suppose that if Pacman starts at position (i, j) such that $i+j=n$, he will reach $(0,0)$ in n seconds regardless of his path.

Inductive Step: Now suppose Pacman starts at position (i, j) such that $i+j=n+1$. If Pacman's first move is to position $(i-1, j)$, the sum of his x and y positions will be $i-1+j=(i+j)-1=n$. Thus, our inductive hypothesis tells us that it will take him n further seconds to get to $(0,0)$ no matter what path he takes. If Pacman's first move isn't to $(i-1, j)$, then it must be to $(i, j-1)$. Again in this case, the inductive hypothesis will tell us that Pacman will use n more moves to get to $(0,0)$ no matter what path he takes. Thus, in either case, we have that Pacman will take a total of $n+1$ seconds (one for the first move and n for the remainder) in order to reach $(0,0)$, proving the claim for $n+1$.

One can also prove this statement without strengthening the inductive hypothesis. The proof isn't quite as elegant, but is included here anyways for reference. We first prove by induction on i that if Pacman starts from position $(i,0)$, he will reach $(0,0)$ in finite time.

Base Case: If $i = 0$, Pacman starts at position $(0,0)$, so he doesn't need any more steps. Thus, it takes Pacman 0 steps to reach the origin, where 0 is a finite number.

Inductive Hypothesis: Suppose that if $i = n$ (that is, if Pacman starts at position $(n,0)$), he will reach $(0,0)$ in finite time.

Inductive Step: Now say Pacman starts at position $(n+1,0)$. Since he is on the x-axis, he has only one move: he has to move to $(n,0)$. From the inductive hypothesis, we know he will only take finite time to get to $(0,0)$ once he's gotten to $(n,0)$, so he'll only take a finite amount of time plus one second to get there from $(n+1,0)$. A finite amount of time plus one second is still a finite amount of time, so we've proved the claim for $i = n+1$.

We can now use this statement as the base case to prove our original claim by induction on j.

Base Case: If $j = 0$, Pacman starts at position $(i,0)$ for some $i \in \mathbb{N}$. We proved above that Pacman must reach $(0,0)$ in finite time starting from here.

Inductive Hypothesis: Suppose that if Pacman starts in position (i,n), he'll reach $(0,0)$ in finite time no matter what i is.

Inductive Step: We now consider what happens if Pacman starts from position $(i,n+1)$, where i can be any natural number. If Pacman starts by moving down, we can immediately apply the inductive hypothesis, since Pacman will be in position (i,n). However, if Pacman moves to the left, he'll be in position $(i-1,n+1)$, so we can't yet apply the inductive hypothesis. But note that Pacman can't keep moving left forever: after i such moves, he'll hit the wall on the y-axis and be forced to move down. Thus, Pacman must make a vertical move after only finitely many horizontal moves–and once he makes that vertical move, he'll be in position (k,n) for some $0 \leq k \leq i$, so the inductive hypothesis tells us that it will only take him a finite amount of time to reach $(0,0)$ from there. This means that Pacman can only take a finite amount of time moving to the left, one second making his first move down, then a finite amount of additional time after his first vertical move. Since a finite number plus one plus another finite number is still finite, this gives us our desired claim: Pacman must reach $(0,0)$ in finite time if he starts from position $(i,n+1)$ for any $i \in \mathbb{N}$.

3 Stable Marriage

Consider a set of four men and four women with the following preferences:

men	preferences	women	preferences
A	1>2>3>4	1	D>A>B>C
B	1>3>2>4	2	A>B>C>D
C	1>3>2>4	3	A>B>C>D
D	3>1>2>4	4	A>B>D>C

(a) Run on this instance the stable matching algorithm presented in class. Show each day of the algorithm, and give the resulting matching, expressed as $\{(M,W),\ldots\}$.

(b) Suppose we relax the rules for the men, so that each unpaired man proposes to the next woman on his list at a time of his choice (some men might procrastinate for several days, while others might propose and get rejected several times in a single day). Prove that this modification will not change what pairing the algorithm outputs.

Solution:

(a) The situations on the successive days are:
Day 1 Proposals: $A \to 1$, $B \to 1$, $C \to 1$, $D \to 3$; B and C are rejected.
Day 2 Proposals: $A \to 1$, $B \to 3$, $C \to 3$, $D \to 3$; C and D are rejected.
Day 3: Proposals: $A \to 1$, $B \to 3$, $C \to 2$, $D \to 1$; A is rejected.
Day 4: Proposals: $A \to 2$, $B \to 3$, $C \to 2$, $D \to 1$; C is rejected.
Day 5: Proposals: $A \to 2$, $B \to 3$, $C \to 4$, $D \to 1$; no one is rejected.
Final matching: $(A,2),(B,3),(C,4),(D,1)$.

(b) Let us first establish the Improvement Lemma for our modified algorithm. If a woman W receives a proposal, then she will always keep that man on a string until a better man comes along. This is true even if the men procrastinate, since the man is not allowed to go to another woman until the W gets a better man.

Once we have established the Improvement Lemma, we can argue that the pairing is stable. Suppose the output contained a rogue couple (M^*,W^*). This means M^* must have proposed to W^* and gotten rejected eventually, which means that W^* left M^* for a better man. And by the Improvement Lemma for our modified algorithm, W^* will have to end up with a man that is better than M^* at the termination of the algorithm. Hence, (M^*,W^*) is not a rogue couple; W^* likes her current partner more than M^*.

Let P' be the pairing that results from our relaxed algorithm and P be the pairing that results from the regular algorithm. Assume that P' and P are not the same. Thus, there exists a man M who is not paired with W, his partner in P. Let's call the woman that M ends up with in P', W'. Since P is male optimal (it is the traditional stable marriage algorithm), and since P' is stable, we know that $W' < W$ on M's preference list.

If M somehow ends up with W' in P' though, that means that he must have proposed to W and ended up getting rejected at some point in time. Define a rejection to be *special* if it is a man getting rejected by his partner in P. Since in the modified algorithm, people could be proposing at any time, we're going to timestamp each *special* rejection. Since there is at least one special rejection but a finite number of special rejections, there is a special rejection with the smallest timestamp.

Call M_0 and W_0 the two people that participated in the special rejection, and let the timestamp of this be t. In order for W_0 to reject M_0 at time t, some other man, M', must have come in at time t to propose to W_0. Since W_0 rejects M_0, we know that $M' > M_0$ in W_0's preference list.

We can therefore conclude, from the Improvement Lemma of the traditional algorithm, that M' must never propose to W_0 in the traditional algorithm.

If M' never proposes to W_0 in the traditional algorithm, that means he ended up with a better woman, W', in P. Thus, in P', W' ended up rejecting M'. But this must happen before time t, and we have contradicted the minimality of t.

Therefore, t must not exist, and $P = P'$.

4 The Better Stable Matching

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

> For every man m, m's date in $R \wedge R'$ is whichever is better (according to m's preference list) of his dates in R and R'.

Also, we will say that a man/woman *prefers* a matching R to a matching R' if he/she prefers his/her date in R to his/her date in R'. We will use the following example:

men	preferences	women	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 matchings for the example given above. Calculate $R \wedge R'$ and show that it is also stable.

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

(c) Prove that, for any stable matchings R, R' where m and w are dates in R but not in R', one of the following holds:

- m prefers R to R' and w prefers R' to R; or
- m prefers R' to R and w prefers R to R'.

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

(You may assume this result in subsequent parts even if you don't prove it here.)

(d) Prove an interesting result: for any stable matchings R, R', (i) $R \wedge R'$ is a matching [*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 men and women. For instance, A prefers 2 to his 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 m be a man, and let his dates in R and R' be w and w' respectively, and without loss of generality, let $w > w'$ in m's list. Then his date in $R \wedge R'$ is w, whom he prefers over w'. However, for m to prefer R or R' over $R \wedge R'$, he must prefer w or w' over w, which is not possible (since $w > w'$ in his list).

(c) Let M and W denote the sets of men and women respectively that prefer R to R', and M' and W' the sets of men and women that prefer R' to R. Note that $|M|+|M'| = |W|+|W'|$, since the left-hand side is the number of men who have different partners in the two matchings, and the right-hand side is the number of women who have different partners.

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

Since $|M|+|M'| = |W|+|W'|$, both these inequalities must actually be tight, and hence we have $|M'| = |W|$ and $|M| = |W'|$. The result is now immediate: if the man m does not date the woman w in one but not both matchings, then

- either $m \in M$ and $w \in W'$, i.e., m prefers R to R' and w prefers R' to R,
- or $m \in M'$ and $w \in W$, i.e., m prefers R' to R and w prefers R to R'.

(d) (i) If $R \wedge R'$ is not a matching, then it is because two men get the same woman, or two women get the same man. Without loss of generality, assume it is the former case, with $(m,w) \in R$ and $(m',w) \in R'$ causing the problem. Hence m prefers R to R', and m' prefers R' to R. Using the results of the previous part would imply that w would prefer R' over R, and R over R' respectively, which is a contradiction.

(ii) Now suppose $R \wedge R'$ has a rogue couple (m,w). Then m strictly prefers w to his partners in both R and R'. Further, w prefers m to her partner in $R \wedge R'$. Let w's partners in R and R' be m_1 and m_2. If she is finally matched to m_1, then (m,w) is a rogue couple in R; on the other hand, if she is matched to m_2, then (m,w) is a rogue couple in R'. Since these are the only two choices for w's partner, we have a contradiction in either case.

5 Examples or It's Impossible

Determine if each of the situations below is possible with the traditional propose-and-reject algorithm. If so, give an example with at least 3 men and 3 women. Otherwise, give a brief proof as to why it's impossible.

(a) Every man gets his first choice.

(b) Every woman gets her first choice, even though her first choice does not prefer her the most.

(c) Every woman gets her last choice.

(d) Every man gets his last choice.

(e) A man who is second on every woman's list gets his last choice.

Solution:

(a) One way to construct an example is to have each man have a distinct first choice. For example,

Men	Preferences
1	$A > C > B$
2	$B > C > A$
3	$C > A > B$

Women	Preferences
A	$3 > 2 > 1$
B	$1 > 3 > 2$
C	$2 > 1 > 3$

(b) An example where every woman gets her first choice, and every man his second choice:

Men	Preferences
1	$C > A > B$
2	$A > B > C$
3	$A > C > B$

Women	Preferences
A	$1 > 2 > 3$
B	$2 > 3 > 1$
C	$3 > 1 > 2$

(c) One method for constructing an example of this is to have each woman have a unique last choice, and have each man most prefer the woman who likes him the least. As an example,

Men	Preferences
1	$A > C > B$
2	$B > C > A$
3	$C > B > A$

Women	Preferences
A	$2 > 3 > 1$
B	$1 > 3 > 2$
C	$1 > 2 > 3$

(d) Impossible. By contradiction: On the last day, every man proposes to his unique least-favorite woman. So prior to the last day, every man has been rejected by all $n-1$ other women, meaning every woman must have rejected all $n-1$ other men. But we can prove that there must exist at least one woman who only ever receives one proposal (we defer this proof to the next paragraph). A woman who receives only one proposal cannot reject any men, so it is impossible for every woman to have rejected $n-1$ men.

To complete this proof, we now just have to show that there always exists a woman who gets only one proposal throughout the entire course of the algorithm. For each woman w, let d_w be the first day in the algorithm that w received a proposal, and let D be the maximum d_w for any woman w. By the improvement lemma, we know that if a woman receives a proposal on day i, she must also receive a proposal on day j for all $j \geq i$. Since $D \geq d_w$ for all women w, this means that all women must get a proposal on day D, so the algorithm must terminate on day

D. But there must be some woman w^* for which $d_{w^*} = D$ (as D is the maximum of the d_ws), meaning that the first day w^* got a proposal was also the last day of the algorithm. Hence, w^* only received one proposal: the one she got on day D.

(e) One method for constructing an example is to have m_i and w_i both prefer each other the most for $1 \leq i < n$. We can then put m_n in the second position on every woman's list and w_n at the last position on m_n's list. An explicit example of this with $n = 3$ is shown below.

Men	Preferences
1	$A > C > B$
2	$B > C > A$
3	$A > B > C$

Women	Preferences
A	$1 > 3 > 2$
B	$2 > 3 > 1$
C	$1 > 3 > 2$

6 Short Answer: Graphs

(a) Bob removed a degree 3 node in an n-vertex tree, how many connected components are in the resulting graph? (An expression that may contain n.)

(b) Given an n-vertex tree, Bob added 10 edges to it, then Alice removed 5 edges and the resulting graph has 3 connected components. How many edges must be removed to remove all cycles in the resulting graph? (An expression that may contain n.)

(c) True or False: For all $n \geq 3$, the complete graph on n vertices, K_n has more edges than the n-dimensional hypercube. Justify your answer.

(d) A complete graph with n vertices where n is an odd prime can have all its edges covered with x edge-disjoint Hamiltonian cycles (a Hamiltonian cycle is a cycle where each vertex appears exactly once). What is the number, x, of such cycles required to cover the a complete graph? (Answer should be an expression that depends on n.)

(e) Give a set of edge-disjoint Hamiltonian cycles that covers the edges of K_5, the complete graph on 5 vertices. (Each path should be a sequence (or list) of edges in K_5, where an edge is written as a pair of vertices from the set $\{0,1,2,3,4\}$ - e.g: $(0,1),(1,2)$.)

Solution:

(a) **3**.
Each neighbor must be in a different connected component. This follows from a tree having a unique path between each neighbor in the tree as it is acyclic. The removed vertex broke that path, so each neighbor is in a separate component. Moreover, every other node is connected to one of the neighbors as every other vertex has a path to the removed node which must go through a neighbor.

(b) **7**

The problem is asking you to make each component into a tree. The components should have $n_1 - 1$, $n_2 - 1$ and $n_3 - 1$ edges each or a total of $n-3$ edges. The total number of edges after Bob and Alice did their work was $n-1+10-5 = n+4$, thus one needs to remove 7 edges to ensure there are no cycles.

(c) **False**

This is just an exercise in definitions. The complete graph has $n(n-1)/2$ edges where the hypercube has $n2^{n-1}$ edges. For $n \geq 3$, $2^{n-1} \geq (n-1)/2$.

(d) $(n-1)/2$.

Each cycle removes degree 2 from each vertex. As the degree of each vertex is $n-1$, we require a total of $\frac{n-1}{2}$ disjoint cycles. This is also sufficient. For a construction in the case of n being an odd prime, see explanations below.

(e) $(0,1),(1,2),(2,3),(3,4),(4,0)$
$(0,2),(2,4),(4,1),(1,3),(3,0)$

The following details a procedure for generating the paths using ideas from modular arithmetic. Note that modular arithmetic is not necessary for the solution, but it provides a clean solution.

The idea is that we can generate disjoint Hamiltonian cycles by repeatedly adding an element a to the current node. This produces the sequence of edges $(0,a),(a,2a),\ldots,((p-1)a,0)$ which are disjoint for different a, as long as $a \neq -a \pmod{p}$, as that would simply be subtracting a everytime. (In other words, there exists no integer k such that $-a+pk = a$.)

We use primality to say that inside a sequence the edges are disjoint since the elements $\{0a,\ldots,(p-1)a\}$ are distinct $\pmod{p}$.