EECS 70 Discrete Mathematics and Probability Theory
Fall 2014 Anant Sahai

Homework 2

This homework is due September 15, 2014, at 12:00 noon.

1. Warm-up Virtual Lab

Last week, you were asked to setup your Virtual Machine for the labs. Now it's time to actually get your feet wet! Please download the IPython starter code from Piazza or the course webpage, and answer the following questions:

(a) In Python, you can represent propositions as follows:
$x = a \vee \neg b$
$y = \neg a \wedge b$

Listing 1: Propositions

```
def x(a, b):
    return a or not b

def y(a, b):
    return not a and b
```

Implement the functions f, g, and h, each takes in three parameters, based on the following propositions. Comment on the relationship between each pair.
$f = (a \wedge \neg b) \vee c$
$g = (\neg a \vee b) \wedge \neg c$
$h = (a \vee c) \wedge (\neg b \vee c)$

(b) Implement the `forall` and `exists` functions, where each function takes in a non-empty list and a predicate (a function that takes in one parameter and returns either True or False). For example, if we apply the function `is_positive` to each element in `lst`, we get False and True, respectively.

Listing 2: Predicate

```
lst = [-1, 1]

def is_positive(x):
    return x > 0
```

`forall` should return True if all elements in the list satisfy the predicate and False otherwise. Similarly, `exists` should return True if there exists an element in the list that satisfies the predicate and False otherwise. Do not use Python's built-in `any` and `all` functions.

(c) Implement the `forall` function again, this time using the `exists` function only. Similarly, implement the `exists` function again using only `forall`.
Hint: you may find higher-order functions and anonymous functions from EECS 61A helpful for this question. For an example, take a look at the IPython Notebook.

Reminder: When you finish, don't forget to convert the notebook to pdf and merge it with your written homework. Please also zip the `ipynb` file and submit it as `hw2.zip`.

2. Inductive Charging

There are n cars on a circular track. Among all of them, they have exactly enough fuel (in total) for one car to circle the track. Two cars at the same location may transfer fuel between them.

(a) Prove, using whatever method you want, that there exists at least one car that has enough fuel to reach the next car along the track.

By contradiction: If not, then they wouldn't have enough fuel in total for one car to complete the track.

Formally: Order cars clockwise around the track, starting at some arbitrary car. Let the fuel in car i be f_i liters, where 1 liter of gas corresponds to 1 kilometer of travel. Let the track be D kilometers around, and let the distance between car i and car $i+1$ (in the modular sense) be d_i kilometers. Assume, for the sake of contradiction, that $f_i < d_i$ for all i (that is, no car can reach the next car). Then $\sum_{i=1}^{n} f_i < \sum_{i=1}^{n} d_i = D$. Contradiction (the cars must have enough fuel in total for one car to circle the track).

(b) Prove that there exists one car that can circle the track, by gathering fuel from other cars along the way. (That is, one car moving and all others stopped). Hint: Use the previous part. [1]

Since the initial orientation of cars is not fixed, we may orient all cars to point in the clockwise direction. Then, we will prove a stronger statement: For n cars oriented clockwise, which in total have enough fuel for one car to circle the track, there always exists one car that can circle the track, traveling clockwise.

The key idea is: Let Car A be a car that can reach the next clockwise car (Car B). Notice that since Car A can reach Car B, we can consider Car A to have the "effective fuel" of cars A+B, and remove Car B altogether (because Car A can travel to Car B, pick up Car B's fuel, and continue). This modified setup has $n-1$ cars oriented clockwise, with enough fuel in total to complete the track (total fuel was unchanged in the transform).

Then by inductive hypothesis, there is one car (Car X) here that can complete the track, traveling clockwise. Simply use this car's clockwise path for the original setup (n cars), making sure to unwrap "effective Car A" into "Car A, then Car B". Notice that strengthening the hypothesis to prove a **clockwise** path exists was necessary, to ensure there exists a path for Car X that reaches Car A before Car B (otherwise, our "unwrapping" step would fail).

The formal proof that follows illustrates the concept of a **reduction**, which is used in many areas of EECS (both theory and practice). Reductions are like calling subroutines in programming: if you have a new problem, you first convert it to a problem you've already solved, call the subroutine for that, then convert the solution back.

Base Case: $n = 1$: If one car oriented clockwise has exactly enough fuel to circle the track... it has enough fuel to circle the track, clockwise.

Inductive Hypothesis: Assume the statement is true for k cars: If k cars oriented clockwise have exactly enough total fuel for one car to circle the track, then there exists one car that can circle the

[1] To think about, after you complete this problem: Is your proof constructive or non-constructive? (That is, does it actually point to the exact car that can complete the track, or does it just prove that one such car must exist?) If it's non-constructive, then how do we actually find this car? (Can we write a program to do this, faster than actually trying every car?)

The proof is non-constructive, but can be converted into an algorithm as follows: First, orient all cars clockwise. Every time some car at point A can reach some car at point B, collapse them into one car at A (oriented clockwise), with the effective fuel of both cars (at A and B). Keep collapsing in this way until we have only car at some point P (why is this collapsing always possible?). The original car at P can complete the track.

track, traveling clockwise (by gathering fuel from the other stopped cars).

Inductive Step: We want to prove the statement for $k+1$ cars. Let a particular set of $k+1$ cars be $C = \{c_1, c_2, \ldots, c_{k+1}\}$, ordered clockwise. We know from part (a) that there exists at least one car which can reach the next clockwise car along the track. Without loss of generality [2], let c_1 be such a car (which can reach c_2). Let us define a new car $\widetilde{c}_1$ with the same initial location and orientation as c_1, but with the fuel of c_1 and c_2 combined. Then, construct a new set of only k cars: $\widetilde{C} = \{\widetilde{c}_1, c_3, \ldots c_{k+1}\}$ (that is, taking the original set C, removing c_2, and replacing c_1 with $\widetilde{c}_1$). Let us use $P(C)$ to denote "the problem setup with cars in C on the track". Then $P(\widetilde{C})$ is the new problem setup.

By construction, the total fuel of cars in $\widetilde{C}$ is the exact same as the total fuel of cars in C. In particular, since C has exactly enough total fuel for one car to circle the track, so does $\widetilde{C}$. Further, cars in $\widetilde{C}$ are also oriented clockwise, by construction. Therefore we may use the inductive hypothesis to conclude that of the k cars in $P(\widetilde{C})$, one of them can circle the track, traveling clockwise.

We are not done yet! We must show that if one of the cars in $P(\widetilde{C})$ can complete the track clockwise, then one of the cars in the original problem $P(C)$ can circle the track clockwise as well.

Say car c_i in $P(\widetilde{C})$ can complete the track, traveling clockwise. Consider its path, as it visits subsequent cars, picks up their fuel, and continues:

$$\widetilde{T} = c_i \to c_{i+1} \to c_{i+2} \to \ldots \to \widetilde{c}_1 \to c_3 \to \ldots$$

At some point, it must visit $\widetilde{c}_1$ (the modified car), pick up its fuel, and continue to c_3 (since it travels clockwise). To construct a valid path for the original problem, we unwrap this step, turning $\widetilde{c}_1 \to c_3$ into $c_1 \to c_2 \to c_3$:

$$T = c_i \to c_{i+1} \to c_{i+2} \to \ldots \to c_1 \to c_2 \to c_3 \to \ldots$$

Notice that the same car c_i in $P(C)$ can certainly execute path T up to reaching car c_1, since all steps are the same as in path $\widetilde{T}$. (Actually, since c_i and $\widetilde{c}_1$ are not necessarily different, we may have $c_i = \widetilde{c}_1$. In this case, by "the same car in $P(C)$", we mean c_1, not $\widetilde{c}_1$.)

Then, executing the steps $\widetilde{c}_1 \to c_3$ in $P(\widetilde{C})$ and the steps $c_1 \to c_2 \to c_3$ in $P(C)$ have exactly the same effect on the car's final location and fuel, and is feasible, by construction. Since the steps of T and $\widetilde{T}$ are the same after c_3, it is possible for one car in $P(C)$ to circle the track traveling clockwise, by executing path T.

Now we have shown how to transform a solution of the smaller, modified problem into a solution of the original problem, and our inductive step is complete.

3. Algorithm Correctness

This problem is a gentle introduction to rigorously proving correctness of algorithms.

Algorithm 1: Recursive Sum

```
sum_of_list(X):
    if X is empty:
        return 0
    else:
        return X[0] + sum_of_list(X[1:])
```

[2] Meaning: In this case, we assumed that c_1 was a car that could reach the next car, but the following steps would equivalently hold if, for example, c_5 was such a car. We could simply re-label cars clockwise, starting at c_5 instead of c_1. In general, the phrase "without loss of generality" means that we appear to be considering a specific case, but some symmetry of the problem (in this case, the modular nature of clockwise labeling) allows us to generalize beyond this specific case.

Algorithm 2: Iterative Sum

```
sum_of_list(X):
    total = 0
    for i = 0 to len(X)-1:
        total = total + X[i]
    return total
```

Algorithm 3: Find a maximal value from an array

```
find_max(X):
    front  = 0;
    end = length(X) - 1;
    while(front != end ) {
        if X[front] <= X[end]:
            front = front +1;
        else:
            end = end - 1;
    }
    return X[front];
```

(a) Prove that Algorithm 1 returns the sum of the list X. That is, prove that for all lists X (of length n):

$$\text{sum\_of\_list}(X) = \sum_{i=0}^{n-1} X[i]$$

We use induction on the length of the list, n.
Base case: when $n = 1$, Algorithm 1 returns $\text{sum\_of\_list}(X) = X[0]$;
Inductive hypothesis: assume when $n = m$, $\text{sum\_of\_list}(X) = \sum_{i=0}^{m-1} X[i]$.
Inductive step: when $n = m+1$, algorithm 1 returns

$$X[0] + \text{sum\_of\_list}(X[1:]) = X[0] + \sum_{i=1}^{m} X[i] \quad \text{(By inductive hypothesis)}$$
$$= \sum_{i=0}^{m} X[i]$$

(b) Prove that Algorithm 2 returns the sum of the list X. Notice, this algorithm is not recursive, so the previous method of proof won't work. Hint: Try proving that the loop iteration performs as expected.

It is not a recursive function. Consequently, we will use induction on the loop variable i and need to come up with the right claim to prove.
Claim: $\forall 0 \le i \le \text{length(X)} - 1$, `total` $= \sum_{k=0}^{i} X[k]$ after completing an iteration in the for-loop.
Proof:
Base case: when $i = 0$, `total` $= 0 + X[0] = \sum_{k=0}^{0} X[k]$
Inductive hypothesis: assume when $i = m$, `total` $= \sum_{k=0}^{m} X[k]$
Inductive step: consider $i = m+1$. At the end of the m-th iteration, `total` $= \sum_{k=0}^{m} X[k]$ based on the inductive hypothesis. `total` retains its value at the beginning of the $(m+1)$-th iteration, and then "`total` = `total` $+ X[m+1]$" is executed. Therefore, `total` $= \sum_{k=0}^{m} X[k] + X[m+1] = \sum_{k=0}^{m+1} X[k]$

Finally, the for-loop ends after $i = \text{length(X)} - 1$, so `total`=$\sum_{k=0}^{\text{length(X)}-1} X[k]$ after the for-loop.

(c) Prove that Algorithm 3 returns the maximum value of array X.

Let n be the number of iterations run in the while-loop, $\texttt{front}_n$ and $\texttt{end}_n$ be the values of `front` and `end` after completing the n-th iteration in the while-loop.
Claim: $\forall n \in \mathbb{N}$, the maximum value is always among $\texttt{X}[\texttt{front}_n ... \texttt{end}_n]$ after completing the n-th iteration in the while-loop.
Proof:
Base Case: when $n = 0$, $\texttt{front}_0 = 0$ and $\texttt{end}_0 = length(X) - 1$, the maximum must be among $\texttt{X}[\texttt{front}_0 ... \texttt{end}_0]$, which is the whole array.
Inductive hypothesis: assume when $n = m$, the maximum value is always among $\texttt{X}[\texttt{front}_m ... \texttt{end}_m]$ after completing the m-th iteration in the while-loop.
Inductive step: for $n = m+1$, when while-loop starts, there will be two cases:

Case 1. $\texttt{X}[\texttt{front}_m] \leq \texttt{X}[\texttt{end}_m]$.
Based on the inductive hypothesis, the maximum is among $\texttt{X}[\texttt{front}_m ... \texttt{end}_m]$ after completing the previous iteration, as well as at the beginning of this iteration.
In this case, we will get $\texttt{front}_{m+1} = \texttt{front}_m + 1$ and $\texttt{end}_{m+1} = \texttt{end}_m$ according to the code.
Since $\texttt{X}[\texttt{front}_m] \leq \texttt{X}[\texttt{end}_m]$, we have $\max\{\texttt{X}[\texttt{front}_{m+1} ... \texttt{end}_{m+1}]\} = \max\{\texttt{X}[\texttt{front}_m ... \texttt{end}_m]\}$.
Hence, the maximum is still among $\texttt{X}[\texttt{front}_{m+1} ... \texttt{end}_{m+1}]$ after completing this iteration.

Case 2. $\texttt{X}[\texttt{front}_m] > \texttt{X}[\texttt{end}_m]$.
Based on the inductive hypothesis, the maximum is among $\texttt{X}[\texttt{front}_m ... \texttt{end}_m]$ after completing the previous iteration, as well as at the beginning of this iteration.
In this case, we will get $\texttt{end}_{m+1} = \texttt{end}_m - 1$ and $\texttt{front}_{m+1} = \texttt{front}_m$ according to the code.
Since $\texttt{X}[\texttt{front}_m] > \texttt{X}[\texttt{end}_m]$, we have $\max\{\texttt{X}[\texttt{front}_{m+1} ... \texttt{end}_{m+1}]\} = \max\{\texttt{X}[\texttt{front}_m ... \texttt{end}_m]\}$.
Hence, the maximum is still among $\texttt{X}[\texttt{front}_{m+1} ... \texttt{end}_{m+1}]$ after completing this iteration.

Since either `front` will increase by 1 or `end` will decrease by 1 in every iteration, `front` will be equal to `end` after (`length(X)-1`) iterations. The while-loop will terminate once `front` = `end`. Based on the above induction, we can conclude that when the while-loop terminates, maximum is `X[front]` (because `front` = `end`).

4. Losing Marbles

Two EECS70 GSIs are playing a game, where there is an urn that contains some number of red marbles (R), green marbles (G), and blue marbles (B). There is also an infinite supply of marbles outside the urn. When it is a player's turn, the player may either:

(i) Remove one red marble from the urn, and add 3 green marbles.

(ii) Remove two green marbles from the urn, and add 7 blue marbles.

(iii) Remove one blue marble from the urn.

These are the only legal moves. The last player that can make a legal move wins. We play optimally, of course – meaning we always play one of the best possible legal moves.

(a) Prove by induction that, if the urn initially contains a finite number of marbles at the start of the game, then the game will end after a finite number of moves.

This is very similar to discussion 2M's "Grid Induction" problem (with the appropriate analogy in 3D).
Just as in "Grid Induction", we could prove this by first proving it for an urn containing only blue marbles (by simple induction), then extending to any number of green and blue marbles (by strong

induction, noticing that at some point we must reduce to only blue marbles and one or zero green marbles), then again extending to include any red marbles (similarly by strong induction).

In "Grid Induction", we were able to simplify the argument by considering an appropriate potential function. We can do that here as well: Imagine placing 1kg marbles on 3 pedestals: Red marbles on a 100 meter pedestal, Green marbles on a 10m one, and Blue marbles on a 1m one. Then every move corresponds to a downward flow of marbles (for example, move (i) converts 1 marble at 100m to 3 marbles at 10m). If we consider the potential energy $\Phi(R,G,B) = 100R + 10G + B$, we find Φ decreases by at least 1 every turn.

Lemma: The value of Φ will decrease by at least 1 after every legal move.

Proof: By cases:

- Move (i): the value of Φ will change from $(100R+10G+B)$ to $(100(R-1)+10(G+3)+B) = (100R+10G+B)-70$, decreasing by 70.
- Move (ii): the value of Φ will change from $(100R+10G+B)$ to $(100R+10(G-2)+(B+7))$, decreasing by 13.
- Move (iii): the value of Φ will change from $(100R+10G+B)$ to $(100R+10G+(B-1))$, decreasing by 1.

□

Since Φ is never negative, we can't play this game forever. This argument is logically correct, but we must formalize it through induction.

Claim: Let Φ_0 be the value of the initial configuration and Φ_n be the value of the configuration after n moves where n is a natural number. Then $\Phi_n \leq \Phi_0 - n$.

Proof: Proof by simple induction over n, the number of legal moves made.

Base case: Let $n = 0$, then $\Phi_0 - n$ becomes Φ_0 and Φ_n becomes Φ_0. Trivially we see that $\Phi_0 \leq \Phi_0$.

Inductive hypothesis: Assume that for some $k \in \mathbb{N}$, $\Phi_k \leq \Phi_0 - k$.

Inductive step: Now consider the $k+1$-th turn. By the lemma, we know that the step will decrease the value of Φ_{k+1} by at least 1, so we conclude that $\Phi_{k+1} \leq \Phi_k - 1$. Hence $\Phi_{k+1} \leq \Phi_k - 1 \leq (\Phi_0 - k) - 1 = \Phi_0 - (k+1)$, where in the second step we used the inductive hypothesis. The claim follows by induction. □

Since $\Phi_n \leq \Phi_0 - n$, we know $n \leq \Phi_0 - \Phi_n$. Since $\Phi_n \geq 0$ by construction (the value can never be negative since it is the sum of natural numbers multiplied by other natural numbers), we can conclude $n \leq \Phi_0$. Since Φ_0 is clearly finite, this means that the game can have only a finite number of legal moves played and must therefore end.

In the solution above we assign weights 100, 10, and 1 for red, green, and blue marbles, respectively. However, there are other weight assignments that make this proof idea work. Another workable assignment would be 13, 4, and 1 for red, green, and blue. *i.e.*, $\Phi = 13R + 4G + B$ since that also ensures that each legal move reduces the value of Φ by at least 1. On the other hand, weighing all marbles equally would not work, because some moves would increase the value of Φ, violating the intended invariant.

What you should get out of this problem: The main idea is to associate each state of the game with a number in a way that lets us establish a useful invariant. This is a common strategy in solving these kinds of problems, but it need not be the only way. Notice also that when saying what you're inducting over, you should provide more detail than just saying "We induct over k" or "Proof by induction over n". Unless it is very clear what n means, specify what quantity n represents.

(b) If the urn contains 2 green marbles and B blue marbles initially, then who will win the game? Prove it. In this case, does it matter what strategy the players use?

First, consider a smaller case: If we started with 0 green marbles, then clearly P1 ("Player 1", the first player) wins if and only if B is odd. (Because each player must remove one blue marble every turn, the total number of turns is the number of initial blue marbles. And P1 wins if and only if the total number of turns is odd).

With 2 green marbles, the idea is essentially: At some point, someone will convert the 2 greens to 7 blues. Now we have only blues, so the next player wins if and only if the number of blue marbles **at this point** is odd. Following this through:

Claim: With 2 green marbles and B blue marbles initially, P1 wins if and only if B is odd (regardless of strategy).

Proof: Since we know from part (a) that the game ends, at some point someone must convert the 2 greens to 7 blues (otherwise, there would still be a legal move available). Let the number of moves **before** this conversion occurs be x. Let the number of moves **after** conversion, until the game ends, be y. Each move before or after conversion must have removed a blue marble, so in total the game lasted $x+y+1$ moves, and removed a total of $x+y$ blue marbles. Since the game started with B blue marbles, ends with 0 blue marbles, and the conversion introduced 7 new blue marbles, we must have $B+7 = x+y$ (blue marbles initially + blue marbles added = total blue marbles removed). Since the game lasted $(x+y)+1 = (B+7)+1 = B+8$ moves, P1 wins if and only if this total number of moves is odd, or equivalently if and only if B is odd. □

(c) If the urn contains (R,G,B) red, green, and blue marbles initially, then who will win the game? Prove it. In this case, does it matter what strategy the players use?

We can try generalizing the previous logic, first by considering arbitrary numbers of green and blue marbles, then also allowing red marbles.

However, we can simplify the argument by using a potential method. This will let us prove both the strategy-invariance and the winner in one sweep.

From the proof of part (b), we notice that the total number of moves in the game was useful. Let us try to find a function $M(R,G,B)$ for this. The following are plausible steps we may take in coming up with such a function. The formal proof follows.

If $R = G = 0$, then trivially $M(0,0,B) = B$ since the only legal move is to remove a blue marble. Similarly, $M(0,1,B) = B$. From part (b), we know $M(0,2,B) = 8+B$, essentially because the greens must be converted to blues (1 move), then the additional blues must be removed (7 more moves). We may postulate that for even G, $M(0,G,B) = 4G+B$ for similar reasons (and in fact, could prove this by induction if we wish). Patching M slightly to deal with odd G, we may try: $M(0,G,B) = 8\lfloor \frac{G}{2} \rfloor + B$

To deal with reds, we notice that at some point every red marble is converted to 3 green marbles. If all these R conversions happen at the beginning, we would have

$$M(R,G,B) = R + M(0,G+3R,B) = R + 8\lfloor \frac{G+3R}{2} \rfloor + B.$$

Now, let us prove that this formula, which we arrived at by a series of intuitive steps and loose assumptions, actually works.

Claim: The total number of legal moves to complete a game, regardless of strategies, is always:

$$M(R,G,B) = R + 8\lfloor \frac{G+3R}{2} \rfloor + B.$$

Where R,G,B is the initial number of red, green, and blue marbles, respectively.

Proof: First, prove by cases (as in part (a)) that M decreases by **exactly one** with every legal move. Further, M has the additional property:

Lemma: $M(R,G,B)$ is 0 if and only if there are no legal moves remaining.
Proof: If there are no legal moves, then there must be: 0 reds, 0 blues, and either 0 or 1 greens (any other case would allow a legal move). And $M(0,0,0) = M(0,1,0) = 0$.
If $0 = M(R,G,B) = R + 8\lfloor \frac{G+3R}{2} \rfloor + B$, then each term in the sum on the right-hand side must be 0 (since they are all non-negative). Therefore: $R = B = 0$ and $8\lfloor \frac{G+3R}{2} \rfloor = 0 \implies \lfloor \frac{G}{2} \rfloor = 0 \implies G = 0$ or 1. □

Now we have a function $M(R,G,B)$ which decreases by exactly one with every legal move, and is zero if and only if the game is over.

Let s_0 be the initial state of the game (number of R,G,B marbles), and $s_1, s_2, \ldots, s_q$ be the states of the game after $1,2,\ldots,q$ moves, for some sequence of moves that ends the game. Since each move decreases M by exactly one, we must have $M(s_q) = M(s_0) - q$. But since the game ends at state s_q, we have $M(s_q) = 0$. Therefore, all sequences of moves that end the game in q moves have $q = M(s_0)$. □

Therefore P1 wins if and only if $M(R,G,B)$ is odd.

5. Induction for Real

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). For example:

Bob the Bug is on a window, 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.

The basic idea is: First, prove directly that he 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 he starts $\leq n+1$ inches above ground.

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:
Base Case: If he starts at height $x_0 \leq 3$, then $x_1 = (x_0+1)/2 \leq 2$. 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.

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, then 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 $x_0 \leq n+1$, we have $x_1 = x_0 - \Delta \leq n+1-\Delta \leq n$. So Bob will be $x_1 \leq n$ inches high after 1 second, after which point we know (by the inductive hypothesis) he dies in finite time. □

6. Hit or Miss?

State which of the proofs below is correct or incorrect. For the incorrect ones, please explain clearly where the logical error in the proof lies. Simply saying that the claim or the induction hypothesis is false is *not* a

valid explanation of what is wrong with the proof. You do not need to elaborate if you think the proof is correct.

(a) **Claim:** For all positive numbers $n \in \mathbb{R}$, $n^2 \geq n$.
Proof: The proof will be by induction on n.
Base Case: $1^2 \geq 1$. It is true for $n = 1$.
Inductive Hypothesis: Assume that $n^2 \geq n$.
Inductive Step: We must prove that $(n+1)^2 \geq n+1$. Starting from the left hand side,

$$\begin{aligned}(n+1)^2 &= n^2 + 2n + 1 \\ &\geq n+1.\end{aligned}$$

Therefore, the statement is true. □
Note that n is a real number. The proof is incorrect because it does not consider $0 < n < 1$, for which the claim is false. Also, by the way it is set up, it can only cover integers for $n \geq 1$.

(b) **Claim:** For all negative integers n, $-1-3-\ldots+(2n+1) = -n^2$.
Proof: The proof will be by induction on n.
Base Case: $-1 = -(-1)^2$. It is true for $n = -1$.
Inductive Hypothesis: Assume that $-1-3-\ldots+(2n+1) = -n^2$.
Inductive Step: We need to prove that the statement is also true for $n-1$ if it is true for n, that is, $-1-3-\ldots+(2(n-1)+1) = -(n-1)^2$. Starting from the left hand side,

$$\begin{aligned}-1-3-\ldots+(2(n-1)+1) &= (-1-3-\ldots(2n+1))+(2(n-1)+1) \\ &= -n^2+(2(n-1)+1) \quad \text{(Inductive Hypothesis)} \\ &= -n^2+2n-1 \\ &= -(n-1)^2\end{aligned}$$

Therefore, the statement is true. □
The proof is correct. The base case starts from the correct, identifiable end point, then the inductive step successfully proves that the statement continues to be true towards $-\infty$.

(c) **Claim:** For all positive integers n, $\sum_{i=0}^{n} 2^{-i} \leq 2$.

Proof: We will prove a stronger statement, that is, $\sum_{i=0}^{n} 2^{-i} = 2 - 2^{-n}$, by induction on n.
Base Case: $n = 1 \leq 2-1$. It is true for $n = 1$.
Inductive Hypothesis: Assume that $\sum_{i=0}^{n} 2^{-i} = 2 - 2^{-n}$.
Inductive Step: We must show that $\sum_{i=0}^{n+1} 2^{-i} = 2 - 2^{-(n+1)}$. Starting from the left hand side,

$$\begin{aligned}\sum_{i=0}^{n+1} 2^{-i} &= \sum_{i=0}^{n} 2^{-i} + 2^{-(n+1)} \\ &= (2-2^{-n}) + 2^{-(n+1)} \quad \text{(Inductive Hypothesis)} \\ &= 2 - 2^{-(n+1)}.\end{aligned}$$

Since $\sum_{i=0}^{n} 2^{-i} = 2 - 2^{-n} \leq 2$, the claim is true. □
The proof only has one minor error. It uses the wrong value for the base case $n = 1$. It should have shown that $2^0 + 2^{-1} = 2 - 2^{-1}$, instead of just using the value of n.

(d) **Claim:** For all nonnegative integers n, $2n = 0$.

Proof: We will prove by strong induction on n.
Base Case: $2 \times 0 = 0$. It is true for $n = 0$.
Inductive Hypothesis: Assume that $2k = 0$ for all $0 \leq k \leq n$.
Inductive Step: We must show that $2(n+1) = 0$. Write $n+1 = a+b$ where $0 < a, b \leq n$. From the inductive hypothesis, we know $2a = 0$ and $2b = 0$, therefore,

$$2(n+1) = 2(a+b) = 2a+2b = 0+0 = 0.$$

The statement is true. □

The proof is incorrect. When $n = 0$, we cannot write $n+1 = 1 = a+b$ where $0 < a, b \leq n = 0$.

(e) **Claim:** Every positive integer $n \geq 2$ has a unique prime factorization.
In other words, let $2 \leq p_1, p_2, \ldots, p_i \leq n$ be all prime numbers that divide n, there is only one unique way to write n as a product of primes,

$$n = p_1^{d_1} \cdot p_2^{d_2} \cdot \ldots \cdot p_i^{d_i},$$

where $d_1, d_2, \ldots, d_i \in \mathbb{N}$.

Proof: We will prove by strong induction on n.
Base Case: 2 is a prime itself. It is true for $n = 2$.
Inductive Hypothesis: Assume that the statement is true for all $2 \leq k \leq n$.
Inductive Step: We must prove that the statement is true for $n+1$. If $n+1$ is prime, then it itself is a unique prime factorization. Otherwise, $n+1$ can be written as $x \times y$ where $2 \leq x, y \leq n$. From the inductive hypothesis, both x and y have unique prime factorizations. The product of unique prime factorizations is unique, therefore, $n+1$ has a unique prime factorization. □

The proof is incomplete because it fails to mention that more than one pair of x and y can satisfy $n+1 = x \times y$. For example, let $n+1 = 32$. (x, y) can be either $(2^2, 2^3)$ or $(2^1, 2^4)$. The proof should have addressed that there can be multiple pairs of (x, y), and shown that they will all lead to the same unique prime factorization (2^5 in case of 32).

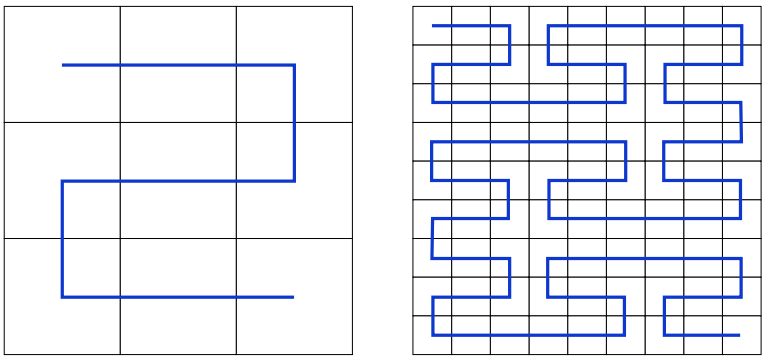
Neglecting possible cases can lead to wrong result. It just so happens that this claim is true and the uncovered case does not change the result.

7. Magic Lawn Mower

There is a magic lawn mower which can remove a perfect square foot of lawn underneath it in the blink of an eye. To make lawn-mowing less boring, the owner decides to cut each 3×3 square feet of his lawn in a '2' or '5' pattern. Can you prove that he can cut through any $3^n \times 3^n$ square feet of lawn ($n \in \mathbb{Z}^+$) with one continuous walk without cutting any same square foot twice? Figure 1 shows his walking patterns for 3×3 and $3^2 \times 3^2$ square feet.

Proof: For convenience, let G_n denote a $3^n \times 3^n$ grid. We will prove by induction on n.

- *Base Case:* He can use the '2' pattern itself to walk through G_1. (See Figure 1a.) It is true for $n = 1$.
- *Inductive Hypothesis:* Assume he can walk through G_n using only '2' and '5' patterns.
- *Inductive Step:* We must prove that he can also walk through G_{n+1} with '2' and '5' patterns. With a working walking pattern for G_n, we can try to get to G_{n+1} by following the path in G_n and replacing each cell with a 3×3 '2' or '5' pattern that still connects each connected cells in G_n together. (See Figure 2.) The question is, how do we choose between the patterns to still make them connected?



(a) 3×3 ft^2 (b) 9×9 ft^2

Figure 1: Example walking patterns

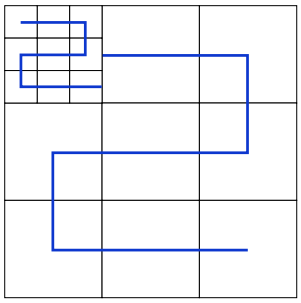
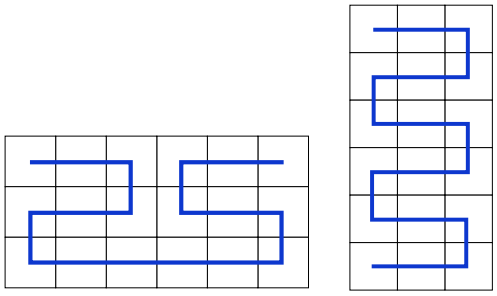


Figure 2: Filling in a cell of G_n with G_1 ($n = 1$ in this case).



(a) Horizontal (b) Vertical

Figure 3: '2' and '5' are horizontal and vertical reflections of each other

Note that '2' and '5' are horizontal and vertical reflections of each other (Figure 3), which means alternating between '2' and '5' will give us a connected line when moving in either horizontal or vertical direction.

Therefore, the owner can always generate a path for G_{n+1} by following the path for G_n and alternating between replacing each cell in G_n with the 3×3 '2' and '5' patterns. Figure 4 illustrates more steps of the method.

□

8. Series Induction

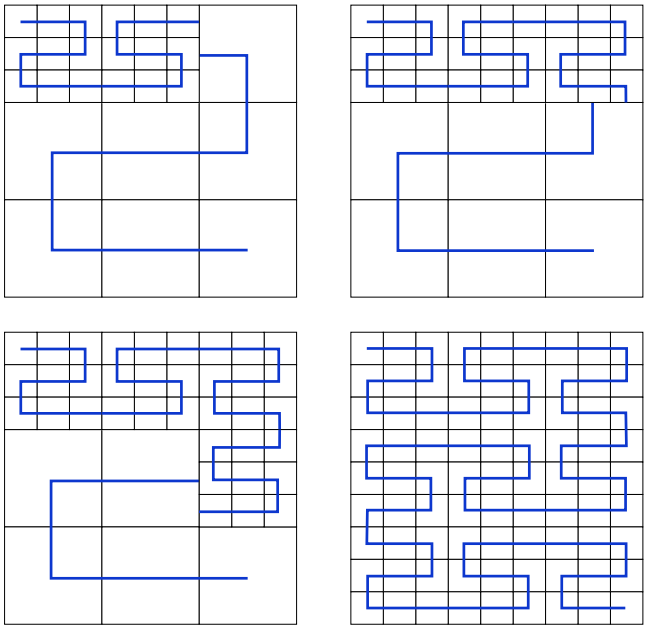


Figure 4: Filling in more cells of G_n with G_1 ($n = 1$ in this case).

For all $n \in \mathbb{N}$, let a_n be the number of subsets of $\{1, 2, \cdots, n\}$ that do not contain any two consecutive numbers (including the empty set).

(a) Show that a_n is the (n+2)-th Fibonacci number.

We use F_n to denote the n-th Fibonacci number, and call subsets of $\{1, 2, \cdots, n\}$ that do not contain any two consecutive numbers valid subsets.

Base case: For $n = 0$, $a_0 = 1 = F_2$ because the only subset of an empty set $\emptyset$ is the empty set. For $n = 1$, we have $a_1 = 2 = F_3$, since the two subsets, $\emptyset$ and $\{1\}$, are the only valid subsets.
Inductive hypothesis: Assume when $0 \leq n \leq m$, $a_n = F_{n+2}$.
Inductive step: when $n = m + 1 > 1$, we can divide all valid subsets of $\{1, 2, \cdots, m, (m+1)\}$ into two groups: one is the group of subsets that contain element $(m+1)$ and another is the group of subsets that do not contain element $(m+1)$. In the first group of subsets, there should not be element m in any of them, since $m+1$ will be included in every subset. Thus, the number of subsets in the first group is a_{m-1} because any valid subset of $\{1, 2, \cdots, (m-1)\}$ unioned with $\{m+1\}$ will become a valid subset of $\{1, 2, \cdots, m, (m+1)\}$. The number of subsets in the second group is a_m because valid subsets of $\{1, 2, \cdots, m\}$ are also valid subsets of $\{1, 2, \cdots, m, (m+1)\}$. Therefore, $a_{m+1} = a_m + a_{m-1} = F_{m+2} + F_{m+1} = F_{m+3}$.

(b) Prove that $a_n = \frac{\varphi^{n+2} - \bar{\varphi}^{n+2}}{\sqrt{5}}$, where $\varphi = \frac{1+\sqrt{5}}{2}$ is the golden ratio and $\bar{\varphi} = \frac{1-\sqrt{5}}{2}$ is the conjugate of φ.

Base case: When $n = 0$,

$$\frac{\varphi^2 - \bar{\varphi}^2}{\sqrt{5}} = \frac{1}{\sqrt{5}} \left(\left(\frac{1+\sqrt{5}}{2} \right)^2 - \left(\frac{1-\sqrt{5}}{2} \right)^2 \right) = \frac{1}{\sqrt{5}} \left(\frac{6+2\sqrt{5}}{4} - \frac{6-2\sqrt{5}}{4} \right) = 1 = a_0.$$

When $n = 1$,

$$\frac{\varphi^3-\bar{\varphi}^3}{\sqrt{5}} = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^3 - \left(\frac{1-\sqrt{5}}{2}\right)^3\right) = \frac{1}{\sqrt{5}}\left(\frac{1+3\sqrt{5}+3\times 5+5\sqrt{5}}{8} - \frac{1-3\sqrt{5}+3\times 5-5\sqrt{5}}{8}\right)$$
$$= \frac{1}{\sqrt{5}}\left(\frac{16\sqrt{5}}{8}\right) = 2 = a_1, \text{ as demonstrated in part (a).}$$

Inductive hypothesis: Assume when $0 \le n \le m$, $a_n = \frac{\varphi^{n+2}-\bar{\varphi}^{n+2}}{\sqrt{5}}$.
Inductive step: When $n = m+1 > 1$,

$$a_{m+1} = a_m + a_{m-1} = \frac{\varphi^{m+2}-\bar{\varphi}^{m+2}}{\sqrt{5}} + \frac{\varphi^{m+1}-\bar{\varphi}^{m+1}}{\sqrt{5}} = \frac{\varphi^{m+1}(\varphi+1)-\bar{\varphi}^{m+1}(\bar{\varphi}+1)}{\sqrt{5}}.$$

Since $\varphi^2 = \left(\frac{1+\sqrt{5}}{2}\right)^2 = \frac{6+2\sqrt{5}}{4} = \frac{3+\sqrt{5}}{2} = \varphi+1$ and $\bar{\varphi}^2 = \left(\frac{1-\sqrt{5}}{2}\right)^2 = \frac{6-2\sqrt{5}}{4} = \frac{3-\sqrt{5}}{2} = \bar{\varphi}+1$, we will get

$$a_{m+1} = \frac{\varphi^{m+1}\varphi^2-\bar{\varphi}^{m+1}\bar{\varphi}^2}{\sqrt{5}} = \frac{\varphi^{m+3}-\bar{\varphi}^{m+3}}{\sqrt{5}}.$$

9. Write Your Own Problem

Write your own problem related to this week's material and solve it. You may still work in groups to brainstorm problems, but each student should submit a unique problem. What is the problem? How to formulate it? How to solve it? What is the solution?