CSE373, MAT373: Analysis of Algorithms, Fall 2022 **Date:** 12/5/22

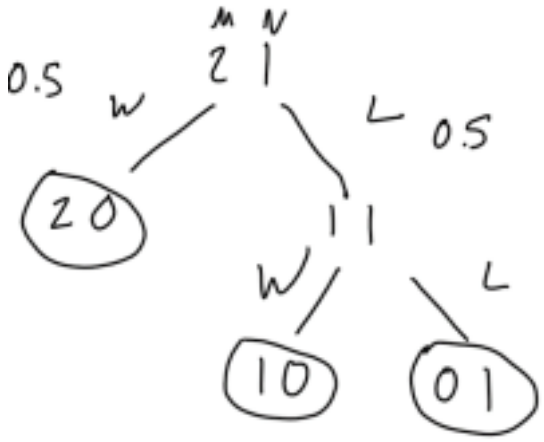
Homework #4

(Due: 12/5/22 **)**

GROUP NUMBER: 6

Group Members		
Name	SBU ID	% Contribution
Michael Lee	112424954	50
Christopher Lobato	114661869	50

1a.



We need to map out all the possibilities in this case there are 3 cases where the game ends 2 of which where Michael wins the game eventually. If we follow the scenarios in which Michael won we can see that the probability for one win was ½ and the probability of the other win was (½) * (½). Since both of these outcomes are possible, we just add the probabilities of both scenarios happening so probability that Michael wins is ¾.

1b.

In the case that M = 2 and N = 1 the scenarios will stay the same the only difference is that the probability of winning changes so in general using our drawing above we can deduce that the probability of Michael winning is $p+(1-p)(p) = p+(p-p^2)$. The first term is the scenario in which he wins on the first turn and the second term takes into account the scenario in which he loses the first game but then wins on his second turn.

1c.

GamblersRuin(M,N,p)

1. If M == 0
2. Return 0
3. ProbabilityOfWin = 0
4. FindProbability(M-1,N,(1-p),p)
5. FindProbability(M,N-1,p,p)
6. Return ProbabilityOfWin

FindProbability(M,N,p,pi)

1. If M == 0
2. //Do nothing
3. Else if N == 0
4. ProbabilityOfWin = ProbabilityOfWin + pi
5. Else
6. FindProbability(M-1, N, p, pi*(1-p))
7. FindProbability(M, N-1, p, pi*p)

This function GamblersRuin initializes a variable to help keep track of the probability of Michael

winning and calls the function FindProbability. The function FindProbability then recursively calls itself for scenarios in which Michael loses a round and scenarios when Rezaul loses a round. At each scenario we multiple the appropriate probability of the event happening(either p or 1-p) and eventually in scenarios where Rezaul ran out of dollars it will trigger the base case and add the probability of that winning scenario to our total ProbabilityOfWin.

1d.

It is unlikely that the algorithm will work since it is possible that the game will go on infinitely if neither player hits a losing streak in which they go completely broke. The above algorithm relies on the fact that there will eventually be a base case where a player will lose.

1e.

$$a_{n+1} = p\,a_n + 1-p\;\; a_{n-1}$$

$$x^2 - px - (1-p)$$

$$x^2 - px - 1 + p$$

$$x^2 - px + p - 1$$

$$x^2 - p(x-1) - 1$$

$$(x+1)(x-1) - p(x-1)$$

$$((x+1) - p)(x-1)$$

$$a_0 = C_1 1^n + C_2 (x+1-p)^n$$

As we begin to solve for the recurrence relation it becomes clear that it is not possible to know for certain the probability that Michael will win as one of the roots is dependent on the value of the p.

2. When the first passenger takes the wrong seat there are two sceneries:

- The passenger takes the last passenger's seat with probability $1/n$
- The passenger takes the seat of passenger m with probability $1/n$ for each m

a) As such the algorithm can be modeled using

$$P(n) = 1/n + 1/n \sum_{m=2}^{n-1} P(n-m+1)$$

As such we can see that

$$P(n) = \begin{cases} 2 & n = 2, \\ 1/n + 1/n \sum_{m=2}^{n-1} P(n-m+1) & n > 2 \end{cases}$$

which will have a $O(n^2)$ time complexity

b) When solving for $P(n)$, we can see that

$$nP(n) - (n-1)P(n-1) = P(n-1),$$

Therefore $P(n) = P(n-1)$ and as such $P(n) = P(2)$ for any $n \geq 2$. This will result in a $O(1)$ time complexity as any n will result in a constant, $1/2$.

3. In order to find out if it is possible to choose n questions from the question bank, the question chosen from the question bank cannot exceed the total number of categories the questions will be divided into.

from $1 \leq i \leq N$, i cannot be greater than m else $m_i \notin [1, m]$

For example, if $n = 4$ and $m = 4$, when the question is chosen from the question bank, if i-th question is 5th question it will be unable to place the question under a valid category.

However, $i \leq 4$ in question bank is possible where question $1 \leq i \leq 4$ is chosen, totalling 4 questions.

Then it satisfies the constraint of n questions being chosen with given parameters.