University of New Mexico
Department of Computer Science

Midterm Examination

CS 362 Data Structures and Algorithms
Spring, 2024

Name:
Email:

Directions:

- This exam lasts 75 minutes. It is closed book and notes, and no electronic devices are permitted. However, you are allowed to use 2 pages of handwritten "cheat sheets"
- *Show your work!* You will not get full credit, if we cannot figure out how you arrived at your answer.
- Write your solution in the space provided for the corresponding problem.

Question	Points	Score	Grader
1	20		
2	20		
3	20		
4	20		
5	20		
Total	100		

1. **Short Answer (4 points each)**

   Answer the following using *simplest possible* Θ *notation.*

   (a) Expected number of items at the $\log n$ level of a skip list containing n items?

   (b) Time to insert n items into a count-min sketch with k hash functions and m counters?

   (c) Solution to the recurrence: $T(n) = 4T(n/2) + n^2$

   (d) Solution to the recurrence: $T(n) = 8T(n/2) + n^2$

   (e) Solution to the recurrence: $f(n) = 5f(n-1) - 6f(n-2) + 3^n$ (answer in big-O)

2. **Induction (20 points)**

   An *elegant* string is defined recursively as follows:

   - The empty string is elegant
   - If S_1 and S_2 are elegant strings, then the string aS_1bS_2a is elegant

   So, for example, the strings aba, $aababa$ and $aabababaa$ are all elegant.

   Prove by induction that any elegant string of length n has exactly $(2/3)n$ a's in it. Don't forget the BC, IH and IS.

3. **Los Pollos Hermanos**

Gus plans to open franchises of his restaurant, *Los Pollos Hermanos*, along Central Avenue. There are n possible locations for franchises, where location i is at mile i on Central, so neighboring locations are distance 1 mile apart. There are two rules.

- At each location, there can be at most one restaurant, and the profit of a restaurant at location is i is v_i.
- Any two restaurants must be at least 2 miles apart.

(a) (3 points) Jesse proposes the following greedy algorithm for this problem. First, sort the locations by decreasing v_i values, then greedily choose the next possible location, provided that it doesn't conflict with previously chosen locations. Show that Jesse's algorithm doesn't always give maximum profit.

(b) (7 points) Walt suggests dynamic programming. For $i \geq 0$, let $m(i)$ be the maximum profit achievable by using locations 1 through i. Write a recurrence relation for $m(i)$. Don't forget the base case(s).

(c) (3 points) In 1-2 sentences, describe how you would create a dynamic program using the previous recurrence. What is the run time of your algorithm?

(d) (7 points) Now Gus wants to solve a generalization of the problem. There are two changes. First, for $1 < i \leq n$, location i is now distance d_i from location $i-1$. Second, any two restaurants must now be distance k apart for some parameter k. Write a new recurrence relation for this problem. Don't forget the base case(s).

4. **Probability** There is a grid with n by n nodes, n^2 **nodes total**. Each node is independently colored green with probability p and red otherwise.

   (a) (4 points) What is the expected number of green nodes?

   (b) (4 points) Use Markov's inequality to get an upper bound on the probability that at least one node is green.

   (c) (4 points) Now use a union bound to bound the probability that no node is green.

   (d) (8 points) A node is an *interior* node if it has 4 neighbors. So, our grid has $(n-2)^2$ interior nodes. Now, let $p = 1/2$. What is the expected number of interior nodes where the node and all 4 of its neighbors have the same color?

5. **Dance, Dance Revolution**

Dance, Dance Revolution (DDR) is played on a platform with 4 squares. You are given an input sequence σ over the symbols A, B, C and D, representing the four squares. In round $i \geq 1$, one of your feet must be on the square $\sigma[i]$. Your feet must always be in different squares, and you can move at most one foot at the start of each round to any new square. Your left foot starts in square A and right foot in square B.

Your goal is to maximize your *score*: the number of rounds in which neither foot moves. Below is an example game play.

σ	A	C	A	D	C	D	B
Feet position	(A,B)	(A,C)	(A,C)	(D,C)	(D,C)	(D,C)	(B,C)
Point?	0	0	1	0	1	1	0

You scored 3 points since there are 3 rounds where neither foot moved.

(a) (15 points) Write a recurrence relation for the value $m(i, \ell, r)$ which gives the maximum score possible on the first i symbols of σ if your left foot ends in square ℓ and your right foot ends in square r.

(b) (5 points) Describe a dynamic program to return the max score for any input σ of length n based on your recurrence. What are the dimensions of your table? How do you fill it in? What is the final value returned? What is the runtime of your algorithm?