University of New Mexico
Department of Computer Science

Midterm Examination

CS 561 Data Structures and Algorithms
Fall, 2020

Directions:

- This exam lasts 3 hours. It is open book, notes and Internet, but you are not allowed to discuss problems with any person.
- PLEASE: Start each main problem (i.e. Problems 1-5) at the top of a new page!
- Give yourself extra time to properly upload your solutions. Late exams may be penalized.
- *Show your work!* You will not get full credit if we cannot figure out how you arrived at your answer.

1. **Short Answer (2 points each)**

   Answer the following questions using the simplest possible Θ notation unless otherwise stated. Assume in recurrences that $f(n)$ is $\theta(1)$ for constant values of n.

   (a) What is $\sum_{i=1}^{n} \log i$?

   (b) What is $\sum_{i=1}^{n} 5^i$?

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

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

   (e) Solution to the following recurrence relation: $f(n) = 4f(n-1) - 4f(n-2) + 2^n$. For this problem, give your answer in big-O notation.

(f) Solution to the following recurrence relation: $T(n) = 8T(\sqrt{n}) + \log^2 n$

(g) Worst case runtime of randomized QuickSort on a list with n elements?

(h) Expected time to do a search in a skip list containing n items?

(i) Consider a certain operation *OP*. Say that over n calls to OP, the i-th call takes $\Theta(i)$ time. What is the amortized cost of OP?

(j) You flip n fair coins, and add each coin, one after the other, to the end of a row after flipping. Call a coin *satisfied* if it has the same value as its right neighbor in the row (the last coin in the row is never satisfied). What is the expected number of satisfied coins? For this problem, Give an exact answer (i.e. not just within Θ)

2. **SillyStrings**

For $n \geq 0$, the n-th SillyString, S_n is defined recursively as follows, where $x \bullet y$ denotes concatenation of string x and string y:

- $S_1 = a$
- $S_2 = bb$
- $S_n = S_{n-1} \bullet S_{n-2}$

Then, we have that $S_3 = bba$ and $S_4 = bbabb$.

(a) (2 points) What is S_6?

(b) (6 points) Prove by induction that every SillyString except S_1 starts with an b. Don't forget the BC,IH and IS.

(c) (12 points) Now, prove by induction that no SillyString contains the substring aa.

3. **Probability and Expectation**

In this problem, there is a n by n grid of lightbulbs, where each bulb is independently on with probability p, and off with probability $1 - p$.

(a) (2 points) What is the expected number of bulbs that are on.

(b) (3 points) A row or column is said to be *fully lit* if all bulbs in that row or column are on. Consider a fixed row or column, what is the probability that it is fully lit?

(c) (3 points) What is the expected number of fully lit rows and fully lit columns?

(d) (3 points) Use Markov's inequality to get an upper bound on the probability that at least 1 row or column is fully lit.

(e) (3 points) Now use a Union bound to get a (possibly different) upper bound on the probability that any of the $2n$ row or columns are fully lit. Hint: A single "bad" event is that a fixed row or column if fully lit.

(f) (6 points) Call a path *Up&Right* if it starts at the bottom-left bulb of the grid always moves to a bulb immediately above or immediately to the right of the current bulb, and ends up at the top-right bulb in the grid. Call such a path *fully lit* if all bulbs traversed by the path are on. Give an upper bound on the probability that any Up&Right path is fully lit. What does this bound give when $p = 1/4$? Hint: The number of bulbs in such a path is always $2n - 1$, and the number of Up&Right paths is no more than 2^{2n}, since there are $2n$ (up or right?) choices made in each path, and each choice has two outcomes.

4. **Danish Boxing**

In the Danish boxing problem, there are 2 sizes of boxes, one box that fits up to 4 Danishes, and one box that fits up to 7 danishes. The first box costs 2 cents, and the second box costs 3 cents. Your goal is to box n danishes with minimum cost.

(a) (4 points) Your co-worker proposes two greedy algorithms for packing Danishes. Greedy 1: Fill up as many size 7 boxes as possible, and then put any remaining Danishes in size 4 boxes. Greedy 2: Use only size 4 boxes. Give two counterexamples: one showing that Greedy 1 can fail to be optimal, and one showing that Greedy 2 can fail to be optimal.

(b) (4 points) Let $c(n)$ be the minimum cost way to box n Danishes. Write a recurrence relation for $c(n)$.

(c) (2 points) Briefly describe how you would design a dynamic program based on this recurrence to compute $c(n)$. What is the runtime of your algorithm?

(d) (10 points) You have received a shipment of defective boxes. Now each size 4 box fails, and drops all of its Danishes with probability p_1 and each size 7 box fails, and drops all of its danishes with probability p_2. The failure events are all independent. Your goal is to minimize the expected number of danishes dropped. Let $f(n)$ be the minimum expected dropped danishes over all boxings. Write a recurrence relation for $f(n)$.

5. **Chomper**

The game "Chomper" starts with a list of n distinct numbers. You and your opponent take turns removing either the leftmost or rightmost number until the list is empty. On your turn, if there are two or more numbers, you decide which of the two numbers to take. The winner is the player with the highest sum of numbers at the end of the game. Assume that you go first.

For parts (a) and (b) below, also assume that your opponent is *greedy*. That is they always take the highest number on their turn.

(a) (2 points) Prove that you should not also use the greedy strategy. That is, show that there is a game that you can win, but only if you do not follow the same greedy strategy as your opponent. Assume you go first.

(b) (6 points) Describe and analyze an algorithm to determine, given the initial list, the maximum number of points that you can collect playing against a greedy opponent. Hint: Let v_i be the value of the i-th number in the initial list; and $m(i, j)$ be the maximum number of points possible when it is your turn and the remaining list consists of the i-th through the j-th numbers.

(c) (4 points) Now assume that your opponent is random: they always choose the left endpoint with probability 1/2 and the right endpoint with probability 1/2. Describe and analyze an algorithm that returns your maximum expected score against such an opponent.

(d) (8 points) Now assume your opponent plays perfectly: they always make choices to optimize their total winnings against an opponent that is also playing optimally. Describe and analyze an algorithm that gives the maximum number of points you can obtain against such an opponent.