CS 473 ✧ Fall 2024
“Homework” 11

Practice only — Solutions will be released on Tuesday, December 10, 2024

---

1. Alex and Bo are playing another game with even more complex rules. Each player independently chooses an integer between 0 and n, then both players simultaneously reveal their choices, and finally they get points based on those choices.

   Chris and Dylan are watching the game, but they don't really understand the scoring rules, so instead, they decide to place bets on the *sum* of Alex and Bo's choices. They both somehow know the probabilities that Alex and Bo use, and they want to figure out the probability of each possible sum.

   Suppose Chris and Dylan are given a pair of arrays $A[0..n]$ and $B[0..n]$, where $A[i]$ is the probability that Alex chooses i, and $B[j]$ is the probability that Bo chooses j. Describe and analyze an algorithm that computes an array $P[0..2n]$, where $P[k]$ is the probability that the sum of Alex and Bo's choices is equal to k.

2. Suppose you are given an arbitrary directed graph $G = (V, E)$ with arbitrary edge weights $\ell : E \to \mathbb{R}$. Each edge in G is colored either red, white, or blue to indicate how you are permitted to modify its weight:

   - You may increase, but not decrease, the length of any red edge.
   - You may decrease, but not increase, the length of any blue edge.
   - You may not change the length of any black edge.

   The *cycle nullification* problem asks whether it is possible to modify the edge weights—subject to these color constraints—so that *every cycle in G has length* 0. Both the given weights and the new weights of the individual edges can be positive, negative, or zero. To keep the following problems simple, assume that G is strongly connected.

   (a) Describe a linear program that is feasible if and only if it is possible to make every cycle in G have length 0. *[Hint: Pick an arbitrary vertex s, and let* dist(v) *denote the length of every walk from s to v.]*

   (b) Construct the dual of the linear program from part (a). *[Hint: Choose a convenient objective function for your primal LP.]*

   (c) Give a self-contained description of the combinatorial problem encoded by the dual linear program from part (b). Do not use the words “linear”, “program”, or “dual”. Yes, you have seen this problem before.

   (d) Describe and analyze a self-contained algorithm to determine *in $O(EV)$ time* whether it is possible to make every cycle in G have length 0, using your dual formulation from part (c). Do not use the words “linear”, “program”, or “dual”.

3. Your eight-year-old cousin Elmo decides to teach his favorite new card game to his baby sister Daisy. At the beginning of the game, n cards are dealt face up in a long row. Each card is worth some number of points, which may be positive, negative, or zero. Then Elmo and Daisy take turns removing either the leftmost or rightmost card from the row, until all the cards are gone. At each turn, each player can decide which of the two cards to take. When the game ends, the player that has collected the most points wins.

   Daisy isn't old enough to get this whole “strategy” thing; she's just happy to play with her big brother. When it's her turn, she takes the either leftmost card or the rightmost card, each with probability $1/2$.

   Elmo, on the other hand, *really* wants to win. Having never taken an algorithms class, he follows the obvious greedy strategy—when it's his turn, Elmo *always* takes the card with the higher point value.

   Describe and analyze an algorithm to determine Elmo's expected score, given the initial sequence of n cards as input. Assume Elmo moves first, and that no two cards have the same value.

   For example, suppose the initial cards have values $1, 4, 8, 2$. Elmo takes the 2, because it's larger than 1. Then Daisy takes either 1 or 8 with equal probability. If Daisy takes the 1, then Elmo takes the 8; if Daisy takes the 8, then Elmo takes the 4. Thus, Elmo's expected score is $2 + (8 + 4)/2 = 8$.

4. You are attending a gala on the planet Krypton which k people are attending. There are n days in a Kryptonian year. Assume that the birthday of each person is on a uniformly random day in the year (and birthdays of different people are independent). We say that a triple-collision occurs whenever *three* people have the same birthday.

   Find a threshold value k^* for triple collisions. In other words, (i) if $k = o(k^*)$m the probability of having any triple collision is $o(1)$, and (ii) if $k = \omega(k^*)$, the probability of having a triple collision is $1 - o(1)$.[1]

   (a) Guess a threshold value k^*. You will prove its validity in parts (b) and (c) below.

   (b) Prove that probability of having any triple collision is $o(1)$ when $k = o(k^*)$. That is, if $k = ck^*$, the probability approaches 0 as c approaches 0.

   (c) Show that if X is an arbitrary non-negative integer random variable, then

   $$\Pr[X = 0] \le \frac{\mathrm{E}[(X - \mathrm{E}[X])^2]}{\mathrm{E}[X^2]}.$$

   (d) Using part (c), prove that probability of having any triple collision is $1 - o(1)$ when $k = \omega(k^*)$. That is, if $k = ck^*$, the probability approaches 1 as c gets arbitrarily large.

[1] Recall that $f(n) = o(g(n))$ means that $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$ and $f(n) = \omega(g(n))$ means that $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

5. A synchronous optical network (SONET) ring is an undirected cycle with n nodes, numbered consecutively from 0 to $n-1$; let e_i denote the edge between node i and node $(i+1) \bmod n$. Suppose we are given a (multi-)set C of m ordered pairs representing *calls*, where each ordered pair (i, j) represents a call originating at node i and destined for node j. Each call can be routed either clockwise or counterclockwise around the cycle. The *SONET ring loading* problem is to route the calls so as to minimize the maximum load on the network. For each index i, let L_i denote the number of calls that use edge e_i in either direction; our goal is to minimize $\max_i L_i$.

   (a) Write an LP relaxation for this problem, and use it to give a 2-approximation algorithm that *deterministically* rounds the LP solution.

   (b) Now suppose we are also given a positive real *capacity* $c(e_i)$ for each edge e_i in the cycle and a positive real *demand* $d(i, j)$ for each call $(i, j) \in C$. A natural generalization of the SONET ring loading problem refines the load L_i to be the sum of the demands of all calls that use edge e_i divided by the capacity $c(e_i)$; the objective is still to minimize the maximum load $\max_i L_i$. (In the original problem, all capacities and demands are equal to 1.) Describe and analyze a deterministic 2-approximation algorithm for this more general problem.

6. Suppose you are given a directed acyclic graph G with a single source vertex s. Describe an algorithm to determine whether G contains a ***spanning binary tree***. Your algorithm is looking for a spanning tree T of G, such that every vertex in G has at most two outgoing edges in T and every vertex of G *except* s has exactly one incoming edge in T.

   For example, given the dag on the left below as input, your algorithm should FALSE, because the largest binary subtree excludes one of the vertices.