

CS 561: Homework 6

Prof. Jared Saia, University of New Mexico

Remember: You are encouraged to work on the homework in groups, but please observe the “Star Trek” rule from the syllabus.

1. **Pizza Coins.** You are buying pizza for n friends. Each friend gives you a distinct one-dollar coin marked with that friend’s name. When you reach the pizza place, it is closed, so you return the n coins uniformly at random, giving one coin to each friend.
 - (a) For a fixed friend, what is the probability that the friend receives their own coin?
 - (b) What is the expected number of friends who receive their own coins?
 - (c) Use Markov’s inequality to upper-bound the probability that at least two friends receive their own coins.
 - (d) What is the expected number of unordered pairs of distinct friends $\{i, j\}$ for which friend i receives friend j ’s coin and friend j receives friend i ’s coin?
 - (e) Assume that n is even. On the way back, you lose a uniformly random subset of $n/2$ coins. Independently, you select a uniformly random subset of $n/2$ friends and distribute the remaining coins uniformly at random, one to each selected friend. Use a union bound to upper-bound the probability that at least one friend receives their own coin.
2. **Rooted Subtree Budget.** Let T be a rooted binary tree with n vertices. Each vertex v has a nonnegative weight $w(v)$, and an integer k satisfies $1 \leq k \leq n$. A set S of vertices is *rooted* if it contains the root and, whenever $v \in S$ is not the root, the parent of v also belongs to S . Your goal is to choose a rooted set of exactly k vertices having maximum total weight.
 - (a) Consider the greedy strategy that starts with the root and repeatedly adds a maximum-weight vertex whose parent has already been selected. Give a counterexample showing that this strategy need not be optimal.
 - (b) For a vertex v , let T_v denote the subtree of T rooted at v . For an integer $j \geq 1$, let $F(v, j)$ be the maximum weight of a set $S_v \subseteq T_v$ of exactly j vertices such that $v \in S_v$ and, whenever $u \in S_v \setminus \{v\}$, the parent of u also belongs to S_v . Also define $F(v, 0) = 0$, representing the choice of no vertices from T_v . Give a recurrence for $F(v, j)$ in terms of the two children of v . Include conventions for a missing child and for infeasible states.
 - (c) Describe a bottom-up dynamic program based on your recurrence. State the order in which entries are computed, the value returned, and the running time and space usage as functions of n and k .
 - (d) Explain how to augment the table so that the algorithm returns an optimal set of vertices, not just its total weight.

3. **Dance, Dance Revolution.** A dance platform has four squares $Q = \{A, B, C, D\}$. The input is a sequence $\sigma[1], \dots, \sigma[n]$ over Q . At the beginning of each round, you may move at most one foot to any square. During round i , at least one foot must be on square $\sigma[i]$, and your two feet must be on different squares. Before round 1, your left foot is on A and your right foot is on B .

You earn one point in a round if neither foot moves, and zero points if one foot moves. Your goal is to maximize your total score.

For $i \in \{0, 1, \dots, n\}$ and distinct $\ell, r \in Q$, let $M(i, \ell, r)$ be the maximum score over the first i rounds among plays that end round i with the left foot on ℓ and the right foot on r .

- (a) Give all base cases for $M(0, \ell, r)$, using $-\infty$ for an unreachable state.
 - (b) Give a recurrence for $M(i, \ell, r)$. Your recurrence should enforce the target-square requirement and should consider exactly those predecessor states from which at most one foot changes position.
 - (c) Describe a dynamic program based on your recurrence. State the table dimensions, the order in which it is filled, the value returned, and the running time and space usage.
4. **Burst Balloons.** There are n balloons in a row, with positive integer values $x_1, x_2, \dots, x_n$. If balloon i is burst while the closest unburst balloons to its left and right have values x_ℓ and x_r , respectively, then you earn $x_\ell x_i x_r$ coins. Treat either missing boundary balloon as having value 1. Your goal is to maximize the total number of coins earned by bursting all balloons.
- (a) Consider the greedy strategy that bursts a balloon giving the most coins in the current step, breaking ties by choosing the leftmost balloon. Give an input on which this strategy is not optimal.
 - (b) Add imaginary boundary balloons $x_0 = x_{n+1} = 1$. For $1 \leq i \leq j \leq n$, define in words an interval subproblem $B(i, j)$ for which it is useful to consider the *last* balloon burst in the interval $i, \dots, j$.
 - (c) Give a recurrence for $B(i, j)$, including the base case for an empty interval. Briefly justify why the two resulting subproblems are independent once the last balloon in an interval has been chosen.
 - (d) Describe a dynamic program based on the recurrence. State the order in which intervals are processed, the value returned, and the running time and space usage.