February USACO Review

Daniel Wisdom

March 8 2019

1 Painting the Barn (silver)

1.1 Problem

USACO Silver, February 2019 Farmer John is not good at multitasking. He gets distracted often, making it hard to complete long projects. Currently, he is trying to paint one side of his barn, but he keeps painting small rectangular areas and then getting sidetracked by the needs of tending to his cows, leaving some parts of the barn painted with more coats of paint than others. We can describe the side of the barn as a 2D x-y plane, on which Farmer John paints N,($1 \leq N \leq 10^5$) rectangles, each with sides parallel to the coordinate axes, each described by the coordinates of its lower-left and upper-right corner points. All coordinates are in the range $0 \leq x, y \leq 1000$.

Farmer John wants to apply several coats of paint to the barn so it doesn't need to be repainted again in the immediate future. However, he doesn't want to waste time applying an excessive number of coats of paint. It turns out that K,($1 \leq K \leq N$) coats of paint is the optimal amount. Please help him determine how much area of the barn is covered with exactly K coats of paint after he paints all his rectangles.

1.2 2D Prefix Sums

Normal 1D prefix sums store the sum (or minimum, xor, etc) from 0 to x. The recurrence is $pre[x] = A[x]+pre[x-1]$, where A is the array we are taking the prefix sums of.

2D prefix sums will store the sum (or min, xor, etc) over a rectangle from (0,0) to (x,y). The recurrence is:

$$pre[x][y] = A[x][y] + pre[x-1][y] + pre[x][y-1] - pre[x-1][y-1]$$

Visualize this as overlapping boxes. $pre[x-1][y]$ and $pre[x][y-1]$ both include the box (0,0),(x-1,y-1). By subtracting out that box once, we get the correct sum. if x=0, omit the contributions from $pre[x-1][y]$ and $pre[x-1][y-1]$. Likewise if y=0 omit those contributions.

When finding pre-box minimums, we do not need to cancel the duplicated area. This is because double counting something while finding the minimum doesn't affect the overall minimum.

$$pre[x][y] = min(A[x][y], pre[x-1][y], pre[x][y-1])$$

In other to calculate pre-box sums we need to calculate $pre[x-1][y]$ and $pre[x][y-1]$ before $pre[x][y]$. Two nested for loops with increasing x and y work.

Note: You can also do 3D or higher prefix sums if you want. The cancelling terms follow a similar pattern. $pre[x][y][z] = A[x][y][z] + pre[x-1][y][z] + pre[x][y-1][z] + pre[x][y][z-1] - pre[x-1][y-1][z] - pre[x-1][y][z-1] - pre[x][y-1][z-1] + pre[x-1][y-1][z-1]$. There are 2^{dim} terms, so 20 dimensional prefix sums are a poor idea.

1.3 Solution

There are only 10^6 tiles, so we will try to find the number of coats of paint on each tile. We can then count the number of tiles with exactly K coats.

We have a rectangle $(x_1, y_1), (x_2, y_2)$, where $x_1 < x_2$ and $y_1 < y_2$. The goal is to place values into an array, such that the pre-box sums of that array are the number of coats of paint on each tile.

Clearly all tiles upper-right of (x_1, y_1) get one more coat of paint. We will put a +1 at $A[x_1][y_1]$. But above y_2 that +1 shouldn't apply. So we put a -1 at $A[x_1][y_2]$. We also put a -1 at $A[x_2][y_1]$. But now, upper right of

(x_2, y_2) there is a net -1. So we put a +1 at $A[x_2][y_2]$. Now, inside the box the pre-box sum will be +1. Outside, the numbers will either cancel or be outside the pre-box.

We place the ±1s for each painted rectangle FJ paints and then run 2D prefix sums. The answer is the number of tiles at exactly K in the 2D prefix array.

Placing the ±1s is $O(N)$. The prefix sums are $O(L^2)$, where L is length of the area we are summing over, in this case 1000. The overall runtime is $O(N + L^2)$.

2 Cow Dating (Platinum)

2.1 Problem Statement

Not impressed by the lackluster dating websites currently available to cows (e.g., eHarmoony, Moosk, Plenty of Cows), Farmer John decides to launch a new cow dating site based on a fancy proprietary matching algorithm that matches cows and bulls according to a wide range of their mutual interests.

Bessie, in searching for a partner to the Valentine's Day Barn Dance, has decided to try out this site. After making her account, FJ's algorithm has given her a list of N possible matches ($1 \leq N \leq 10^6$). Going through the list, Bessie concludes that each bull has probability $p_i (0 < p_i < 1)$ of accepting an invitation from her for the dance.

Bessie decides to send an invitation to each bull in a contiguous interval of the list. Virtuous as always, she wants exactly one partner. Please help Bessie find the maximum probability of receiving exactly one accepted invitation, if she chooses the right interval.

2.2 Solution

Let's first consider the probability that Bessie gets exactly one acceptance. The probability is:

$$\sum_{l \leq i \leq r} (p_i \prod_{l \leq j \leq r, i \neq j} (1 - p_j))$$

Let's simplify this expression by setting $P = \prod_{l \leq i \leq r} (1 - p_i)$. Then, the expression simplifies to: $P \sum_{l \leq i \leq r} \frac{p_i}{1-p_i}$. Now, let's consider adding one more probability p' to the existing list of probabilities. Accounting for this new probability, the expression becomes $P(1 - p')(\frac{p'}{1-p'} + \sum_{l \leq i \leq r} \frac{p_i}{1-p_i})$. Obviously, if we want to include this new probability in our invitation, we want this to be greater than the old probability. In other words, we want $P(1-p')(\frac{p'}{1-p'} + \sum_{l \leq i \leq r} \frac{p_i}{1-p_i}) > P \sum_{l \leq i \leq r} \frac{p_i}{1-p_i}$. We can reduce this neatly to $1 > \sum_{l \leq i \leq r} \frac{p_i}{1-p_i}$. This means that as long as the running sum of $\frac{p_i}{1-p_i}$ is less than 1, it is optimal to extend our segment. Obviously, because $\frac{p}{1-p}$ is positive, our running total will increase. Note that Bessie wants to send an invitation to a contiguous interval. So, once our constraint is broken, we simply advance the left pointer to denote the new starting point. When the constraint has been broken, we will simply divide our current product by $(1 - p_l)$ and subtract our current sum by $\frac{p_l}{1-p_l}$. By doing so, we eliminate the need to recalculate everything over again. So, every time we pick a new starting point, we consider the maximal interval. This solution can be implemented in $O(N)$ with two pointer approach. It is $O(N)$ because our solution is simply just a scanning the array once.

3 Dishwashing

USACO Gold, February 2019 Bessie and Elsie are helping Farmer John wash the dishes, a more complicated process than one might think due to their lack of opposable thumbs. The two cows decide that Bessie will apply soap, and Elsie will rinse. Bessie is given a dirty stack of plates labeled 1 through N ($1 \leq N \leq 10^5$) Elsie has an empty stack, where clean plates will go. There is a counter in between Bessie and Elsie for soapy stacks.

At each step, either:

- Bessie takes a plate from the top of the dirty stack, applies soap, and then places it on the counter. When placing a soapy plate on the counter, Bessie must either (i) place the plate on top of an existing non-empty soapy stack or (ii) create a new soapy stack to the right of all existing soapy stacks.

- Elsie takes a plate from the top of the leftmost soapy stack. Elsie rinses the plate, then places it on top of the clean stack.

The goal is for the clean stack to have all plates in order, with the smallest label on the bottom and the largest label on the top. It may not be possible for the cows to achieve this goal for the entire stack of plates, so please determine the length of the largest prefix of the input ordering for which the goal is achievable.