Fall 23 Div I Week 3 - Solution Sketches

(taken from editorials of original problems)

Problem A

(Source: 2021-2022 ICPC, NERC, Northern Eurasia Onsite (Unrated, Online Mirror, ICPC Rules, Teams Preferred) Problem D)

This problem can be solved in a straightforward way. The key observation is that the order in which the letters are called out does not matter in this game. We only need to know how many times each letter is called out in order to go from the initial word s to the final word t.

So first, let us compute the number of occurrences of each letter from 'A' to 'Z' in both words s and t. Let's call them s_a and t_a for each letter a. Using these numbers, we can calculate how many times each letter shall be called in order to get a chance of getting to t. That is $s_a - t_a$ times for each letter a.

If $s_a - t_a < 0$ for any letter a, then the answer is "NO".

Otherwise, there is a chance for a positive answer. However, we also need to verify that the order of the letters in t is correct. The easy way to verify it is to simulate the game, dropping the first $s_a - t_a$ occurrences of each letter a, and then compare the result with t.

Problem B

(Source: COMPFEST 14 - Preliminary Online Mirror (Unrated, ICPC Rules, Teams Preferred) Problem B)

For a team of c players with a biggest power of b, the total power of the team is $b \times c$. So for a team with a biggest power of b to win, it needs to have at least $\lceil \frac{D+1}{b} \rceil$ players.

For each player i, we can calculate a value $f(i)$ which means a team that has player i as its biggest power needs to have at least $f(i)$ players to win. We can see that the bigger the value of P_i, the smaller the value of $f(i)$.

We can also see that if a formed team is said to have a fixed biggest power and a fixed number of players, the powers of the players that are less than the biggest power do not affect the total power of the team. So those players can be anyone.

Using the information above, we can form the teams using a greedy method. We iterate through each candidate player starting from the biggest P_i and form new teams with each next biggest candidate player power as each team's biggest power. We do that while maintaining the total number of extra players required to make all formed teams win. We stop once the number of remaining players is not enough for the total number of extra players required.

Time complexity: $O(N \log N)$

Problem C

(Source: The 2018 ICPC Asia Qingdao Regional Programming Contest (The 1st Universal Cup, Stage 9: Qingdao) Problem C)

Tutorial

First, find out in which intervals s and t are different.

We can only flip at most two intervals with two operations. If there are more than two intervals which are different, there is no solution and the answer is 0.

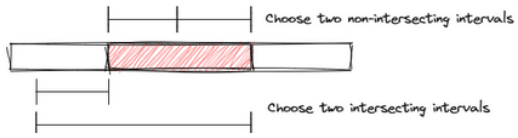
Next, let's discuss the cases where there are 0, 1 or 2 different intervals.

No Different Interval

If s and t are equal, both operations must flip the same interval to keep s unchanged. So the answer is the number of non-empty intervals, that is, $(\frac{n(n-1)}{2}+n)$.

One Different Interval

If there is exactly one interval, we have two types of ways to perform the operations. Let the length of the interval be l.



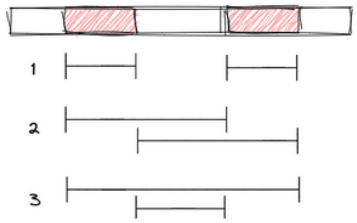
Type One: Choose two non-intersecting intervals such that their union is our target interval. There are $2(l-1)$ ways of this type.

Type Two: Choose two intersecting intervals such that their difference is our target interval. There are $2(n-l)$ ways of this type.

So the answer is $2(l-1)+2(n-l)=2(n-1)$。

Two Different Intervals

If there are two intervals, we have three types of ways to perform the operations.



By changing the order of the two operations we get two ways from each type. So the answer is $2\times 3=6$.

The time complexity is $\mathcal{O}(n)$.

Problem D

(Source: The 2018 ICPC Asia Qingdao Regional Programming Contest (The 1st Universal Cup, Stage 9: Qingdao) Problem I)

Let's describe the problem in another way.

> Use segments of length 1 or 2 to cover the whole sequence. The value of a segment is the sum of elements it covers. Minimize the difference between the maximum value and the minimum value.

Let's solve this problem using segment trees. Let $f(l, r, x \in \{0, 1\}, y \in \{0, 1\})$ indicate the smallest max-value to cover the sub-array $a_l, a_{l+1}, \cdots, a_r$. More precisely:

- $x = 0$ indicates that the segment covering a_l is completely included in interval $[l, r]$. That is, the segment covering a_l is either $[l, l]$ or $[l, l + 1]$.
- $x = 1$ indicates that the segment covering a_l is not completely included in interval $[l, r]$. That is, the segment covering a_l is $[l - 1, l]$.
- $y = 0$ indicates that the segment covering a_r is completely included in interval $[l, r]$. That is, the segment covering a_r is either $[r, r]$ or $[r - 1, r]$.
- $y = 1$ indicates that the segment covering a_r is not completely included in interval $[l, r]$. That is, the segment covering a_r is $[r, r + 1]$.

We can get the recursive equation on segment tree:

$$f(l, r, x, y) = \min_{k \in 0,1} \{\max(f(l, m, x, k), f(m + 1, r, k, y))\}$$

where $m = \lfloor \frac{l+r}{2} \rfloor$. The answer is $f(1, n, 0, 0)$. The initial values are:

- $f(i, i, 0, 0) = a_i$, this is the segment of length 1.
- $f(i, i, 0, 1) = a_i + a_{i+1}$, this is the segment of length 2.
- $f(i, i, 1, 0) = -\infty$, the cost covering a_i is calculated in the previous interval.
- $f(i, i, 1, 1) = +\infty$, a_i cannot be covered by two segments.

Next, we enumerate the min-value v, and remove all segments whose value is smaller than v (no need to really remove them, just change their value to $+\infty$), now the answer becomes $f(1, n, 0, 0) - v$.

There are $(2n - 1)$ possible values of v. For each v modify the value of segments with segment tree and calculate the value of f. The time complexity is $\mathcal{O}(n \log n)$.

Problem E

(source: 2021-2022 ICPC Southwestern European Regional Contest (SWERC 2021-2022) Problem H)

Let us start with some definitions that will make the presentation clearer. Given three points X, Y and Z in the plane, let $f(X, Y, Z)$ be the minimum of $\overline{XP} + \overline{YP} + \overline{ZP}$ over all points P. It is well known that this minimum actually exists and is achieved at a unique point called the Fermat point of the triangle XYZ.

For fixed X and Y, let $f_{XY}(Z) := f(X, Y, Z)$ and define

$$g(W) := \max(f_{AB}(W), f_{AC}(W), f_{BC}(W)),$$

where A, B and C are the positions of the three friends.

For a given parameter r, the condition that Z is a valid residence point is equivalent to $g(Z) \leq r$. Thus, no valid residence point exists if and only if $\min g > r$, and therefore the minimum value of r that works is equal to the minimum of g over all points of the plane.

We claim that g is convex, hence this minimum exists and we can find it efficiently. This relies on the following two observations:

- The pointwise maximum of a finite set of convex functions is convex.
- The function f_{XY} is convex.

The first observation is a standard fact. For the second observation, let X, Y be two points, $0 \leq t \leq 1$, and let P, Q be the respective Fermat points of the triangles ABX and ABY. Then, denoting $Z = tX + (1-t)Y$ and $R = tP + (1-t)Q$, we have that

$$\overline{AR} = |A - tP - (1-t)Q| \leq t\overline{AP} + (1-t)\overline{AQ},$$

$$\overline{BR} = |B - tP - (1-t)Q| \leq t\overline{BP} + (1-t)\overline{BQ},$$

and

$$\overline{ZR} = |t(X-P) + (1-t)(Y-Q)| \leq t\overline{XP} + (1-t)\overline{YQ},$$

so adding up the three inequalities we get that

$$f_{AB}(Z) \leq \overline{AR} + \overline{BR} + \overline{ZR} \leq t f_{AB}(X) + (1-t) f_{AB}(Y).$$

In view of these facts, if we know how to compute $f(X, Y, Z)$ efficiently we can solve the problem with a double ternary search on g. There are ways to achieve this in $\mathcal{O}(1)$ time by distinguishing two cases:

- If one of the angles of triangle XYZ is greater than $2\pi/3$, then the corresponding vertex is the Fermat point.
- Otherwise the Fermat point can be computed using the construction of the Napoleon triangle, or the sum of its distances can be computed more straightforwardly using the following formula:

$$f(X, Y, Z) = \sqrt{\frac{a^2 + b^2 + c^2 + 4\sqrt{3}S}{2}}.$$

  (Here a, b, c are the sidelengths and S is the area of the triangle.)

This solution has a total complexity of $\mathcal{O}(|\log \varepsilon|^2)$, where ε is the required precision. Alternatively, each of the three Fermat points can be computed using a double ternary search on its own, giving a total complexity of $\mathcal{O}(|\log \varepsilon|^4)$, which is also fast enough.