Informatyczny kącik olimpijski (57): Teleporty

W tej edycji kącika omówimy zadanie pt. *Podlewanie ogórków* (naprawdę!) z rundy 2A konkursu TopCoder Open 2012. W zadaniu dany jest pewien zbiór punktów ustawionych wzdłuż prostej i naszym celem jest odwiedzić wszystkie te punkty w pewnej zadanej kolejności. Innymi słowy, mamy n różnych punktów ponumerowanych od 1 do n (porządek numerów *niekoniecznie* od lewej do prawej) i powinniśmy odwiedzić je wszystkie w kolejności $1, 2, \ldots, n$. Pomóc nam w tym mogą teleporty – mamy możliwość rozstawić w dowolnych punktach prostej łącznie t teleportów. Teleporty działają tak, że w ułamku sekundy możemy teleportować się z dowolnego teleportu do dowolnego innego teleportu. Naszym zadaniem jest tak ustawić dostępne teleporty, aby zminimalizować długość trasy odwiedzającej po kolei punkty $1, 2, \ldots, n$, przy założeniu, że po drodze możemy teleportować się dowolnie wiele razy.

Rozwiązywanie każdego (nietrywialnego) zadania warto rozpocząć od wstępnej analizy problemu, czyli od stwierdzenia, o co w nim tak naprawdę chodzi. Niech $a_1, a_2, \ldots, a_n$ oznaczają współrzędne kolejnych punktów od lewej do prawej, a $k_1, k_2, \ldots, k_n$ – numery tych punktów. Niech wreszcie $p_1, p_2, \ldots, p_n$ oznaczają współrzędne punktów w porządku zadanych numerów $1, 2, \ldots, n$. Zauważmy, że poszukiwaną optymalną ścieżkę możemy podzielić na fragmenty: od p_1 do p_2, od p_2 do p_3 itd., przy czym każdy z fragmentów biegnie albo w linii prostej między punktami p_i a p_{i+1}, albo ścieżką od punktu p_i do najbliższego mu teleportu i od teleportu najbliższego punktowi p_{i+1} do punktu p_{i+1}.

Na pewno warto coś zrobić z faktem, że zgodnie z treścią zadania liczba możliwych rozstawień teleportów jest nieskończona. Naturalna hipoteza jest taka, że możemy ograniczyć się do teleportów ustawionych w pewnych spośród zadanych n punktów. Łatwo uzasadnić prawdziwość tej hipotezy: Załóżmy, że pewien teleport znajdowałby się gdzieś pomiędzy dwoma sąsiednimi punktami, czyli na pozycji x, takiej że $a_i < x < a_{i+1}$ dla pewnego i. Rozważmy wszystkie momenty, w których korzystamy z teleportu x. W każdym z nich musimy dojść do tego teleportu albo od punktu a_i (bądź innego, położonego na lewo), albo od punktu a_{i+1} (bądź innego, położonego na prawo). To oznacza, że odcinek $[a_i, x]$ przejdziemy l razy, a odcinek $[x, a_{i+1}]$ przejdziemy r razy. Jeśli więc $l \geqslant r$, to teleport przestawiamy na pozycję a_i, a w przeciwnym razie na pozycję a_{i+1}, i w obu przypadkach długość trasy nam nie wzrośnie.

Mamy już jakąś wizję tego, jak będzie wyglądała optymalna trasa oraz gdzie powinniśmy ustawiać teleporty. To pozwala już skonstruować pierwsze rozwiązanie zadania, rozpatrujące wszystkie możliwe rozstawienia teleportów. Złożoność czasowa takiego rozwiązania to z grubsza $O(t^{k+1})$.

Aby poprawić ten rezultat, powinniśmy jakoś „dobrze uchwycić" nasze zadanie. Problem ewidentnie stanowi w nim obecność dwóch porządków: zadanego porządku odwiedzania punktów oraz naturalnej kolejności „od lewej do prawej", odpowiadającej najkrótszym ścieżkom między kolejnymi punktami. Potrzebujemy jeszcze jakichś spostrzeżeń.

Załóżmy, że znamy pozycje wszystkich rozstawionych teleportów: $a_{j_1} < a_{j_2} < \ldots < a_{j_t}$. Teleporty te wyznaczałyby wówczas podział ciągu wszystkich punktów na fragmenty położone między kolejnymi teleportami. Czy na podstawie tego podziału umielibyśmy łatwo obliczyć wynik?

Rozważmy pewien punkt a_i, taki że $a_{j_m} \leqslant a_i \leqslant a_{j_{m+1}}$. Oznaczmy $L = a_{j_m}$, $R = a_{j_{m+1}}$. Wiemy, że poprzedni punkt w kolejności odwiedzania znajduje się na pozycji $x = p_{k_i - 1}$ (punkt ten może też nie istnieć). Jeśli teraz punkt x szczęśliwym trafem również znajduje się między teleportami L i R, to możemy bardzo łatwo wyznaczyć najkrótszą ścieżkę między nim a a_i: albo idziemy bezpośrednio po prostej, albo korzystamy z kombinacji teleportów L i R. A co, jeśli punkt x znajduje się zupełnie gdzieś indziej? Wiemy wówczas, że najkrótsza ścieżka od x do a_i biegnie od x do najbliższego mu teleportu i dalej od teleportu najbliższego a_i do a_i (zauważmy, że jeśli wynikiem jest najkrótsza ścieżka z x do a_i w linii prostej, to i tak możemy ją opisać w podanej postaci!). To oznacza, że wkład punktu a_i do wyniku możemy opisać całkiem lokalnie: jeśli mianowicie $L \leqslant x \leqslant R$, to do wyniku dodajemy

$$\min(|a_i - x|, \min(x - L, R - x) + \min(a_i - L, R - a_i)), \tag{1}$$

a w przeciwnym razie wynik zwiększamy tylko o

$$\min(a_i - L, R - a_i), \tag{2}$$

natomiast odległość od punktu x do najbliższego mu teleportu dodamy do wyniku w chwili rozpatrywania tego punktu. Aby ta metoda była poprawna, przy rozpatrywaniu a_i należy jeszcze stwierdzić, czy następny punkt w kolejności odwiedzania, $y = p_{k_i+1}$, znajduje się między teleportami L i R, a jeśli *nie*, jeszcze raz dodać do wyniku składnik (2). (Jeśli p_{k_i+1} nie istnieje, nie musimy niczego dodawać).

To oznacza, że do wyznaczenia wyniku wystarcza nam dla każdego punktu znajomość pozycji dwóch najbliższych mu teleportów. Możemy więc zastosować metodę programowania dynamicznego. W ramach stanu musimy na pewno jakoś zapamiętać zakres rozpatrzonych punktów wzdłuż prostej, liczbę już ustawionych teleportów oraz pozycje skrajnych ustawionych teleportów. Najlepiej jest to zrobić tak: pole $A[i, j]$ tablicy będzie odpowiadać minimalnej sumie wkładów punktów $a_1, a_2, \ldots, a_i$ do wyniku, przy założeniu, że rozstawiliśmy wśród nich j teleportów, z których ostatni znajduje się na pozycji a_i. Wówczas ze stanu $A[i, j]$ mamy $O(n)$ przejść, do stanów postaci $A[i', j + 1]$ dla $i' > i$, i możemy je wszystkie rozpatrzyć w czasie $O(n)$. Żeby uniknąć nieprzyjemnych przypadków brzegowych, postawimy łącznie $t + 2$ teleporty, z czego dwa pomocnicze: jeden na pozycji $a_0 = -\infty$, a drugi na pozycji $a_{n+1} = +\infty$.

Całe rozwiązanie działa w czasie $O(n^3)$ i pamięci $O(n^2)$.

Jakub RADOSZEWSKI