

B - Deque Bracket Optimization 解説

原案：potato167,noya2 問題準備：noya2

B：問題概要

- 長さ $2n$ の数列 w のスコアを次のように定める

$$\max_{\text{長さ } 2n \text{ の正しい括弧列 } s} \sum_{i=1}^{2n} w_i \cdot 1[s_i \text{ が開き括弧}]$$

- 各クエリでは w に `push_front`, `push_back` が合計 2 個ずつ飛んでくる
- そのたびにスコアを再計算せよ
- $Q \leq 4 \times 10^5, |w_i| \leq 10^7$

B：例

スコアの例

- $w = (1, 2, 3, 4)$ の場合： $s = ()()$ が最適で、スコアは $1 + 3 = 4$.
- $w = (1, 9, 3, 4)$ の場合： $s = (())$ が最適で、スコアは $1 + 9 = 10$.
- $w = (-1, -1, 0, 0, -1, 100)$ の場合： $s = ()(())$ が最適で、スコアは $-1 + 0 + 0 = -1$.

クエリでは整数 x, y が与えられ、次の 3 種類のいずれかで w を更新する

- $w \leftarrow (y, x, w_1, \dots, w_{|w|})$
- $w \leftarrow (x, w_1, \dots, w_{|w|}, y)$
- $w \leftarrow (w_1, \dots, w_{|w|}, x, y)$

B : push_back x 2 しか来ないとき

これは過去にいくらかでも出題例がある (ABC407-E, JAG 夏 2023-I)

問題の言い換え

- $w = (w_1, \dots, w_{2n})$ の各要素が答えに加算されるかを考える
- w_1 は加算される
- それ以降, $2i$ 個のうち i 個以上が加算される ($0 \leq i \leq n-1$)
- w_{2n} は加算されない
- 合計で n 個加算される

B : push_back x 2 しか来ないとき

- $\text{ans} \leftarrow w_1$
- max を取る優先度付きキュー $P \leftarrow \emptyset$
- for $i = 1, 2, \dots$
 - この時点での ans が $w[1, 2i]$ に対する答え
 - P に w_{2i}, w_{2i+1} を push する
 - P から max を pop して z とする
 - ans に z を足す

クエリにはオンラインで答えられる

= 2 個ずつ push_back してはスコアを再計算するデータ構造を構成できる

B：では，`push_front` x 2 しか来ないときは？

- 逆向きに見る
- `min` を取る優先度付きキューを使う
- キューから取り出した値を合計から引くと答えになる

それはそう

順向きの際は「取る要素の最大化」をしたのに対して，
逆向きの際は「捨てる要素の最小化」をしている

B : push_back x 2 と push_front x 2 だけなら？

重要な観察

- $X :=$ 順向きに見たときの max を取る優先度付きキューの最終的な中身
- $Y :=$ 逆向きに見たときの min を取る優先度付きキューの最終的な中身

としたとき、 $X \sqcup Y = \{w_2, w_3, \dots, w_{2n-1}\}$ が成り立つ

解釈：

- 順向きに見たときは、欲しい要素をキューから取り出した：要らない要素がキューの中に
- 逆向きに見たときは、要らない要素をキューから取り出した：欲しい要素がキューの中に

B : push_back x 2 と push_front x 2 だけなら？

push_back x 2 をしたとき

- 順向きのキューは普通に更新できる
- 逆向きのキューの更新は諦めるしかないのか？
- いいえ、**順向きのキューの中身の真反対になるように調整すればいい！**
- 具体的に：順向きのキューに x, y を push したあと z を pop したとする
- → 逆向きのキューに z を push する

pop_front x 2 は役割を入れ替えて同様に

B: push_back & push_front にも対応したい

パリティがずれて困ったと思ったら

現在の列 $(w_1, w_2, \dots, w_{2n})$ に対して

- $W = (w_2, w_3, \dots, w_{2n-1})$ に対するデータ構造 D
- $W' = (w_1, w_2, \dots, w_{2n})$ に対するデータ構造 D'

を同時に管理すればいい！

push_back & push_front は

- D, D' を swap
- D' への push_back $\times 2$ & push_front $\times 2$

とすればよい. 計算量は $O(Q \log Q)$ time

- AC teams / Trying teams
 - 3 + ? / 11
- First Acceptance
 - u_u ..zZZ (124 min)