北大合宿 2020 Day1 G 問題
Freqs

原案: tsutaj
問題文: rsk0315
解答: tsutaj, rsk0315
解説: tsutaj

2020 年 9 月 14 日

問題

Freqs

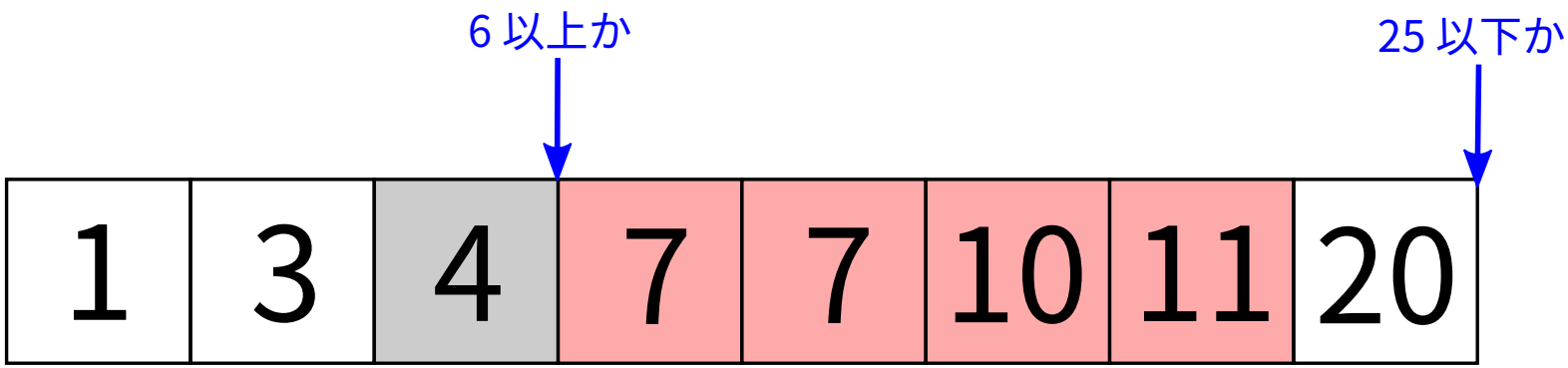
- 長さ N の数列 $A = (a_1, \ldots, a_N)$ がある
- 以下のクエリを処理
  1. $l \leq i \leq r$ について $a_i \leftarrow \min(a_i, x)$
  2. $l \leq i \leq r$ について $a_i \leftarrow \max(a_i, x)$
  3. $l \leq i \leq r$ について $a_i \leftarrow a_i + x$
  4. $l \leq i \leq r$ かつ $x \leq a_i \leq y$ を満たす i の個数を報告
- 制約
  - $1 \leq N, Q \leq 10^5$
  - $-10^9 \leq a_i \leq 10^9$ $(1 \leq i \leq N)$
  - クエリ 1, 2, 3: $-10^9 \leq x \leq 10^9$ $(1 \leq i \leq Q)$
  - クエリ 4: $-10^{15} \leq x \leq y \leq 10^{15}$ $(1 \leq i \leq Q)$

方針

- 二乗 $\rightarrow$ むり
- クエリ 4 がつらそう
  - 数列に対して、部分配列内にある要素のうち値の条件を満たすものを数える必要がある
  - いきなり一般の場合はつらいので、楽できるケースを考えてみよう

方針

- 二乗 $\rightarrow$ むり
- クエリ 4 がつらそう
  - 数列に対して、部分配列内にある要素のうち値の条件を満たすものを数える必要がある
  - いきなり一般の場合はつらいので、楽できるケースを考えてみよう
- クエリ 4 で楽できる場合 $\rightarrow$ 数列がソート済みであるとき
  - 二分探索 (“lower_bound”, “upper_bound”) で高速に処理可能
  - クエリあたり $O(\log N)$
- 「単調性があるなら高速にできる」という性質を使いたい！



Q. 区間 [3, 8) の中で 6 以上 25 以下の要素はいくつある？
A. 4 個

想定解法

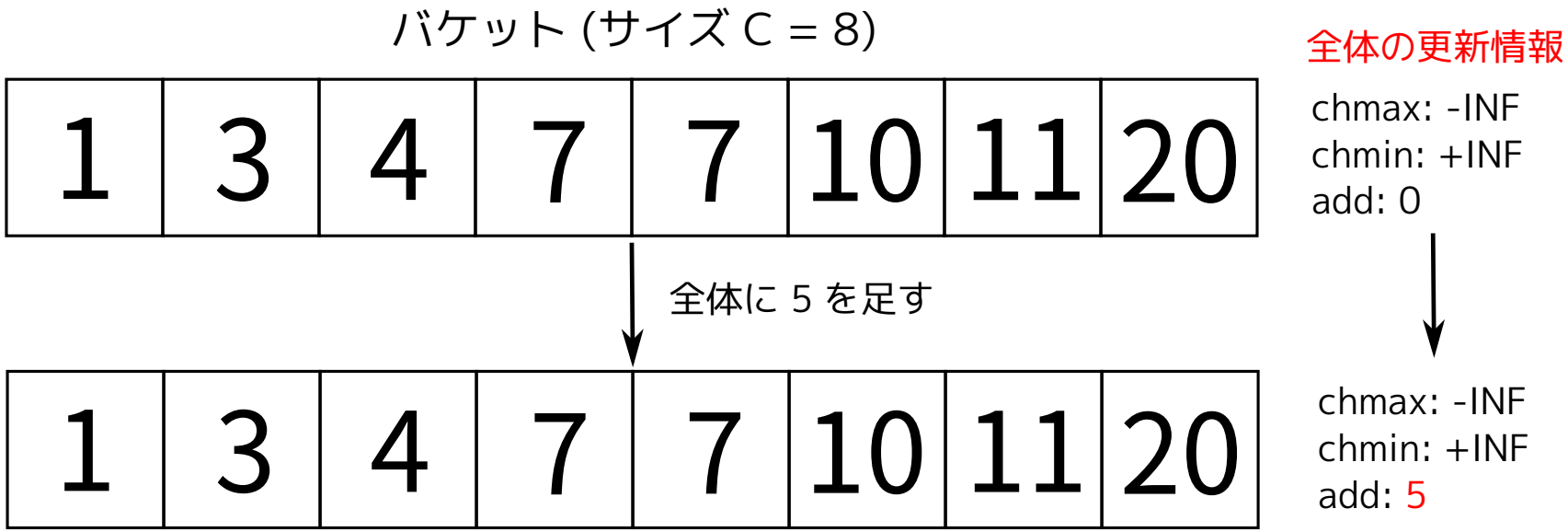
想定解法: 平方分割 ▸ Link

- 数列の要素を、先頭から B 個ずつまとめる (C 個のバケット)
- クエリ区間とバケットの位置関係で場合分けして処理していく
  - クエリ区間がバケットを完全に覆う場合
  - クエリ区間がバケットの一部を覆う場合

数列の長さ: N = 9, バケットサイズ: B = 3, バケット数: C = 3

想定解法: バケットを完全に覆う場合

- クエリ 1, 2, 3: バケットの中身は更新せず、「バケット全体に更新が来た」ことだけ覚える
  - 全体にかかる “chmin”, “chmax”, “add” の値更新だけやるので $O(1)$
- クエリ 4: バケットの中身 + 全体に来た更新情報 を使って答える
- 先程紹介したように二分探索が使えるので $O(\log N)$



想定解法: バケットの一部を覆う場合

- 更新 (クエリ 1, 2, 3): 全体に来た更新情報 を反映 $\rightarrow$ 更新後にソート
  - 下図のように処理すると $O(B \log B)$
  - バケットの情報だけだと値の位置が不明に $\rightarrow$ ソート前の情報も持つ
- 取得 (クエリ 4): 値を見るだけなので $O(B)$

想定解法: 計算量

- $B = C = \sqrt{N}$ とおく
  - $B \times C \simeq N$
- バケットを完全に覆う場合: $O(\log N)$
- バケットの一部を覆う場合: $O(\sqrt{N} \log N)$
- クエリは Q 回来る $\rightarrow$ 全体 $O(Q\sqrt{N} \log N)$
  - B に $\log$ が掛かることを考慮して B 少なめ・C 多めにするとちょっと速くなる気がします

実装 tips

- バケットごとに “chmin”, “chmax”, “add” の値を覚える必要があった
  - 適当に考えると、当該バケットに関しての操作列を全部覚える必要が出てきそうだが、もっと楽にできる
- “chmax”, “add” は以下のようにするとコンパクトにまとめられる
  - “add x” $\rightarrow$ “chmax y” $\Leftrightarrow$ “chmax $y-x$” $\rightarrow$ “add x”
  - “chmax” 同士・“add” 同士は簡単にまとめられる
  - 以上より “chmax”, “add” からなる有限長の操作列は “chmax a” $\rightarrow$ “add b” という形に正規化できる
- “chmin”, “add” に関しても同様にやればよい

余談

- Q. なぜ TL 2.0s なのか？
  - 定数倍高速化 (fastio, 自動ベクトル化) を頑張ると愚直 C++17 が 2.24s くらいで通る
  - さすがにこれが通ると興ざめなので調整しました
    - この制約でも愚直を通した人がいたら、すごい
    - 時間制約がギリギリになってしまいすみません
- Q. C++ 以外お断り？
  - Java でも AC を確認しています

Writer 解・統計

- Writer 解
  - tsutaj (C++・202 行・5504 bytes)
  - tsutaj (Java・213 行・6626 bytes)
  - rsk0315 (C++・160 行・3867 bytes)
- 統計
  - AC / tried: 3 / 47 (6.38 %)
  - First AC
    - latsatnat (87 min 05 sec)