

CF 1095 D. Reserved Reversals — 带教对话录

zzy & Claude

2026-04-29

题目链接: [CF 1095D Reserved Reversals](#) (Codeforces Round 1095 Div.2)。

第 1 步: 理解题目

Claude

先确认操作语义。题面给的合法操作是: 选 $[l, r]$ 满足 $\min(a_l \dots a_r) + \max(a_l \dots a_r)$ 是奇数, 然后把这一段反转。

样例 5 是 $a = [4, 2, 4, 2, 4]$, 题面提示这个 case 答案 NO 的理由是“找不到任何合法的 (l, r) ”。

问题: 对这个具体数组, 为什么任何 (l, r) 都不合法? 用一句话说出“min+max 奇数”这条约束实际上等价于 segment 满足什么条件。

zzy

对于一个具体数组来说, 如果找不到任何一个这个操作序列, 那么就说明数组全部为奇数, 或者全部为偶数是吧

Claude

对。再精确一点: $\min + \max$ 奇 \Leftrightarrow \min 与 \max 一奇一偶 \Leftrightarrow 这个 segment **至少含一个奇数且至少含一个偶数** (因为 \min/\max 都来自 segment 内部, $\text{segment 全奇} \Rightarrow \min/\max$ 都奇 \Rightarrow 和偶; 全偶同理)。

所以“找不到任何合法 (l, r) ”等价于“整个数组单一奇偶性”。样例 5 全偶就是这样。

记一个名字方便后面提: 把含“至少一奇一偶”的 segment 叫做**混色 segment** (mixed-parity segment)。**只有混色 segment 才能反转。**

接下来对一下数据范围: $1 \leq n \leq 2 \cdot 10^5$, $\sum n \leq 2 \cdot 10^5$, 时限 2s, $1 \leq a_i \leq n$ 。

问题: 你想往什么时间复杂度量级走?

zzy

这个显然是 $O(n \log n)$ 的这个时间复杂度

Claude

合理。 $n \leq 2 \cdot 10^5$, $O(n^2) = 4 \cdot 10^{10}$ 直接寄, 目标 $O(n \log n)$ 或更优 (很可能 $O(n)$ 就能搞定)。

第 2 步：找判定条件（不变量）

Claude

正常套路是“暴力 \rightarrow 优化”，但这题暴力 BFS 不现实——状态空间是 a 的所有排列， n 一大就爆。所以得直接找结构性判定条件：什么样的 a 是 YES，什么样的 NO。

样例 5 $[4, 2, 4, 2, 4]$ 全偶 \Rightarrow 没合法操作 \Rightarrow 没排好序就 NO，简单。但样例 6 $a = [3, 3, 1, 5, 5, 2]$ 也是 NO——这个数组有 5 个奇 + 1 个偶（位置 6 那个 2），随便找含 2 的段就能凑出混色段，明明可以做操作，为啥是 NO？

问题：手动模拟一下 $[3, 3, 1, 5, 5, 2]$ ，挑几个你觉得有用的合法 (l, r) 反转看看，看能不能感知出“这个 2 跑来跑去能不能把数组排好序”——以及为啥不能。

zzy

肯定是不可能的，因为 1 没有办法跑到最前面，1 要跑到最前面，肯定要有一个比较大的偶数，但是最大的偶数就是 2，在 3 的阻挠下，1 是没有办法跑到前面去的

Claude

对，“在 3 的阻挠下，1 跑不到前面”就是核心不变量的雏形。把它推广一步：

问题：考虑数组里两个同奇偶的值 $a < b$ 。在什么条件下， a 和 b 的相对顺序在所有合法操作下都不可能改变？（即：如果原数组里 b 在 a 前面，那永远没法变成 a 在 b 前面。）

zzy

这个有点困难，我想的就是，如果任意包含 a, b 的子数组中，满足可交换条件即可，具体的话，我就不细讲，总的来说就是最大值和最小值的这个奇偶性要不同

Claude

对，框架对了。我来补完最后一步形式化：

要 swap a, b ，必须有合法反转 $[l, r]$ 同时包含两人位置。这种反转的 $\min \leq a, \max \geq b$ 。合法 $\Leftrightarrow \min + \max$ 奇 \Leftrightarrow 一奇一偶。但 a, b 同奇偶，所以 \min 和 \max 至少有一个不能是 a 也不能是 b ，必须是某个“更极端”的异奇偶值 c ，且 $c < a$ （充当 \min ）或 $c > b$ （充当 \max ）。

所以 locked 条件（同奇偶 $a < b$ 永远换不了相对顺序）：

数组里没有任何异奇偶值 c 满足 $c < a$ 或 $c > b$

等价地：所有异奇偶值都落在闭区间 $[a, b]$ 内。

设 E 表数组里偶数集合、 O 表奇数集合（异奇偶视情况取另一个）。对同为奇数的 $a < b$ ：locked $\Leftrightarrow \min(E) \geq a$ 且 $\max(E) \leq b$ （约定 $E = \emptyset$ 时 $\min = +\infty, \max = -\infty$ ，那么 vacuously locked）。同奇偶为偶时对称。

样例 6 验证： $O = \{1, 3, 3, 5, 5\}, E = \{2\}, \min(E) = \max(E) = 2$ 。

- $(1, 3)$: $2 \geq 1, 2 \leq 3 \Rightarrow$ locked。原数组 1 在两个 3 之后，sorted 要 1 在前 \Rightarrow 违反 \Rightarrow NO
- $(1, 5)$: locked，同样违反。
- $(3, 5)$: $2 \geq 3 \Rightarrow$ 不 locked，可换。

问题：这个 locked 条件给了一个必要条件（违反就 NO）。它是不是也充分？换句话说：如果对所有 locked 同奇偶对 (a, b) 、原数组里 a 都已经在 b 前，是否一定 YES？想想看，怎么说服自己。

zzy

这个还能不充分？我认为，首先要确认这个问题，问题就是， $b > a$ ，而且这个先 b 后 a ，永远也解决不了，只要确认了有这样的对存在，那么肯定是这个错的。
 不过，我现在理解你的意思了，上面这段话，就说明了其是一个必要条件，但是没法说明这个是一个充分条件。
 呃，不过也只能猜一手了，我觉得这个条件挺对的

Claude

对，这个条件确实也充分（建立在数据范围内的合法操作总能凑出来）。直觉理由：

- 异奇偶相邻 swap 永远合法（长度 2 的混色段反转 = 直接交换），所以“奇偶交错”的局部错位都可以 bubble sort 修。
- 任何未 locked 同奇偶对 (a, b) 都存在某个“更极端”的异奇偶值 c ，可以选一个含 a, b, c 的反转、把 c 抬成 min 或 max 就把 a, b 交换过来。
- Locked 对原本就已正确（因为题设保证），不需要动。

严格证明要构造性 sorting 流程，这里就跳过——竞赛里这种“sortability”题判定通常都是：必要条件 = 充分条件，信一手即可，对拍验证就稳了。

第 3 步：把判定条件落到 $O(n)$ 算法

Claude

Locked 条件复述：同奇偶对 (a, b) ， $a < b$ 锁死 \Leftrightarrow 异奇偶集合 E 满足 $\min(E) \geq a$ 且 $\max(E) \leq b$ 。“判定 YES/NO” = 对每个 locked 对，检查原数组里 a 所有位置 $< b$ 所有位置。朴素地枚举所有同奇偶对就是 $O(n^2)$ 。但 locked 对其实有非常规整的结构——它们不是“任意”的同奇偶对。

问题：固定奇偶 = 奇（讨论奇与奇之间锁死的 pair）。设异奇偶（偶数）集合的 $\min(E) = M_-$ 、 $\max(E) = M_+$ 。把奇值按 $\leq M_-$ 、 $\in (M_-, M_+)$ 、 $\geq M_+$ 分三段——记为 L (low) / M (middle) / H (high)。

哪些 locked 奇-奇对都是属于哪两段之间的？想清楚这一步，你应该能直接给出一个 $O(n)$ 判定算法。

zzy

（直接写代码，跑了一遍，6 个样例里第 3、6 错——预期 NO 给了 YES）

Claude

算法本身正确（按当前判定条件，3 和 6 都能正确判 NO）。 $O(n)$ 形式：

对每个奇偶 p ：设异奇偶集合 E_p ，记 $M_- = \min(E_p)$ ， $M_+ = \max(E_p)$ 。

- $E_p = \emptyset$ ：奇偶 p 子序列必须已非递减，否则 NO。
- 否则记 $\text{lastL}_p = \max\{i : a_i \text{ 奇偶} = p, a_i \leq M_-\}$ 、 $\text{firstH}_p = \min\{i : a_i \text{ 奇偶} = p, a_i \geq M_+\}$ 。两者皆存在且 $\text{lastL}_p \geq \text{firstH}_p \Rightarrow \text{NO}$ 。

样例 3 $[5, 4, 3, 2, 1]$ ， $p = \text{奇}$ ： $E_p = \{2, 4\}$ ， $M_- = 2$ ， $M_+ = 4$ 。 $\text{lastL} = \text{奇} \leq 2$ 即 $\{1\}$ ，位置 5； $\text{firstH} = \text{奇} \geq 4$ 即 $\{5\}$ ，位置 1。 $5 \geq 1 \Rightarrow \text{NO}$

样例 6 $[3, 3, 1, 5, 5, 2]$ ， $p = \text{奇}$ ： $E_p = \{2\}$ ， $M_- = M_+ = 2$ 。 $\text{lastL} = \text{位置 3 (值 1)}$ ； $\text{firstH} = \text{位置 1 (值 3)}$ 。 $3 \geq 1 \Rightarrow \text{NO}$

算法层关键点：locked 配对的小值 a 必须 $\leq M_-$ ，不只是“某个比 b 小的同奇偶值”——比 b 小但 $> M_-$ 的同奇偶值不形成 locked 对，因为它们落在中间段 (M_-, M_+) 内，能被某个 $< M_-$ 的异奇偶值“夹出来”重排。

zzy

不是哥们，我们是不是对算法的认识不统一啊，我认为的这个算法就是长这样啊（贴 D_Reserved_Reversals.cpp）

Claude

进你的框架走——right-to-left 在线 stMat、找 nearSmall 当 inversion pair、用 findL/findR 检查 fixable，骨架没问题。分歧只在 findGloC 两个 arg 接哪个端点。

按命名：findL(x) = “存在异奇偶 $< x$ ”、findR(x) = “存在异奇偶 $> x$ ”。inversion pair (a, b) , $a < b$ ($a = \text{nearSmall}$ 小端、 $b = A[i]$ 大端)。可解锁条件：

$$\exists \text{ 异奇偶 } c \text{ 满足 } c < a \text{ 或 } c > b \Leftrightarrow \text{findL}(a) \vee \text{findR}(b)$$

findL 喂小端、findR 喂大端。但代码里 findGloC(A[i], nearSmall) 把大端喂给 findL、小端喂给 findR——反了。

样例 6 $i = 1, b = 3, a = 1$, 异奇偶 {2}:

检查	含义	结果
正确 findL(1) findR(3)	偶 < 1 或 > 3	都没有 \Rightarrow false (locked)
你的 findL(3) findR(1)	偶 < 3 或 > 1	$2 < 3 \Rightarrow$ true (误判 unlocked)

更糟：对任意异奇偶 c ($c \neq a, c \neq b$), $c < b$ 或 $c > a$ 总成立（三种情况 $c < a < b$ 、 $a < c < b$ 、 $a < b < c$ 都满足）。所以你这个 findGloC 异奇偶非空时恒为 true，退化成“opp 非空就 unlocked”——只有奇偶单一存在且非递减时才 NO，其他全 YES。3 / 6 都漏 NO 就是这个原因。

zzy

findGloC(nearSmall, A[i]) 顺序我搞错了，已修。但样例 3 [5,4,3,2,1] 还是不对，不符合的对是什么？

Claude

(1,5)。1 $\in L$ 、5 $\in H$, locked。原数组里 5 在位置 1、1 在位置 5，先 5 后 1，违反 sorted 要求的“先 1 后 5” \Rightarrow NO。