The 22nd Zhejiang Provincial Collegiate Programming Contest Analysis

GLORITY
睿琪软件

2025.4.26

A. Outer LIS

Shortest Judge Solution: 2209 Bytes

Description

给定一个长度为 n 的二元组序列，每个元素为 (a_i, w_i)。

q 次查询，每次查询给出一个区间 $[l, r]$，问该区间的元素一个都不选的情况下的带权 LIS。

- $n \le 10^5$。
- $q \le 10^5$。
- $1 \le a_i \le n$。

Solution

- 预处理出 f_i 表示以 i 为结尾的 LIS、g_i 表示以 i 为开头的 LIS。
- 对于每个询问，答案可能来自于:
  1. f 的前缀最大值或 g 的后缀最大值。
  2. 在 $[1, l-1]$ 选一个 i，并在 $[r+1, n]$ 选一个 j，满足 $a_i \le a_j$，对答案的贡献为 $f_i + g_j$。
- 方便起见，接下来只讨论第二种情况。
- 将序列分成 $m = \mathcal{O}(\sqrt{n})$ 块，预处理出 $pre_{i,j}$ 表示第 $[1, i]$ 块中 a 值不超过 j 的 f 的最大值、$suf_{i,j}$ 表示第 $[i, m]$ 块中 a 值至少为 j 的 g 的最大值。
- 再预处理出 $ans_{i,j}$ 表示第 $[1, i]$ 块与第 $[j, m]$ 块中的元素两两组合的最优解，配合 pre 或 suf 数组可以 $\mathcal{O}(n\sqrt{n})$ 得到。

Solution

- 对于一个询问 $[l, r]$，假设 l 位于第 x 块、r 位于第 y 块，答案可能来自于：
  1. 第 $[1, x-1]$ 块与第 $[y+1, m]$ 块的元素两两组合，即 $ans_{x-1,y+1}$。
  2. 第 x 块中零散的 $\mathcal{O}(\sqrt{n})$ 个元素与第 $[y+1, m]$ 块的元素两两组合。枚举零散元素后利用 suf 数组 $\mathcal{O}(1)$ 进行查询。
  3. 第 y 块的零散元素与第 $[1, x-1]$ 块的元素两两组合。同理利用 pre 数组进行查询。
  4. 两块的零散元素两两组合，排序后双指针计算。在预处理阶段对每个块按 a 进行排序，双指针时直接筛掉不在范围内的元素即可不带 $\log$。
- 总时间复杂度 $\mathcal{O}((n+q)\sqrt{n})$。

B. Turn on the Light 3

Shortest Judge Solution: 626 Bytes

Description

有 n 个灯，n 个开关，以及 m 次操作，每次操作会操作一个开关。操作编号为 x 的开关会打开所有编号为 x 的倍数的灯。

每次操作后输出有多少个灯被打开，或汇报没有被新打开的灯。

- $n \leq 2 \cdot 10^5$。
- $m \leq 2 \cdot 10^5$。
- $1 \leq a_i \leq n$。

Solution

- 操作一个按钮 x 不会打开任何新的灯当且仅当编号为 x 的灯已经打开。
- 每次操作先判断编号为 x 的灯是否打开，再枚举所有 x 的倍数去开灯即可。
- 时间复杂度 $O(n \log n + m)$。

C. RDDCCD

Shortest Judge Solution: 3751 Bytes

Description

$n \times n$ 网格中每个位置有一个数。每次翻转一条对角线上所有数的数位，求所有数 GCD。

- $n \leq 10^3$。
- $q \leq 10^6$。
- $1 \leq a_i \leq 10^9$。

Solution

- 先将盘面旋转 45°，变为行列操作。
- 考虑初始所有数全是 37 或 73 的情况（此时倒过来是 73，与 37 互质)。
- 如果想要让最后 GCD 不为 1，那么必须所有数字都是未翻过的状态或所有数字都是翻过的状态。我们先考虑如何判断所有数字都是未翻过的状态。
- 对于一个位置，未翻过当且仅当其对应行和对应列要么都翻过，要么都未翻过。
- 每个位置会对应一个限制，代表行列之间要么的翻转状况要么相同要么相反。

Solution

- 我们可以对每行建一个点，每列建一个点，然后将限制作为边连接对应行和列。每个连通块至多只会有两种形态满足限制。
- 我们可以使用 DFS 或并查集求出每个连通块满足限制的条件，然后使用数组记录满足条件的行列数，即可维护这种情况下的 GCD 信息。
- 考虑多个质因子的情况，最终 GCD 的因子一定只会出现在第一个数中或第一个数的翻转中。因此只需分解第一个数，然后对每个质因子（和质因子的幂）分别用上述方法维护即可。注意有无论如何无法满足的情况需要特判。
- 时间复杂度 $O((n^2+q)\log(\max a_i))$。

Problem D. Too Clever By Half

Shortest Judge Solution: 903 Bytes

Description

给定一个机器人，在数轴上 $[0, n]$ 之间的整点移动，从 0 出发在 0 结束，每次可以向左或向右一格。给定机器人到达每个坐标的次数 $c_0, c_1, \ldots, c_n$，构造字典序最小的左右移动序列满足到达次数。

- $n \le 2 \cdot 10^5$。
- $1 \le c_i \le 10^6, \sum c_i \le 2 \cdot 10^6$。

Solution

- 将机器人第 x 秒在 y 点看作坐标 (x, y)，那么构成了一个折线图。
- 折线图上连接坐标为 y 的层与坐标为 $y+1$ 的线段数量为 $2 \cdot \sum_{i=0}^{y} (-1)^{i-y} c_i$，其中一定是一半上升一半下降。
- 从前到后构造序列，只要保留一次折线让其回到 0，其他折线可以用来尽量放置 LR。
- 过程中折线数量为负，或有坐标大于 n 的折线即为无解。
- 时间复杂度 $O(\sum c)$。

Problem E. Gold Miner

Shortest Judge Solution: 4556 Bytes

Description

给定第四象限 n 个金块的坐标，有 q 个询问。

每次询问如果在 x 轴上 $[a_i, b_i]$ 随机选择一个实数点 $(p, 0)$，距离前 k_i 近的金块的距离的平方和。

- $n \le 2 \cdot 10^3$。
- $1 \le q \le 5 \cdot 10^5$。

Solution

- 对于两个金块 i, j，满足 $x_i < x_j$，连接这两点的线段的垂直平分线与 x 轴的交点横坐标不妨设为 e_{ij}。当 $p < e_{ij}$，那么 i 距离 p 更近，否则 j 距离 p 更近。
- 考虑 $a_i = b_i$ 的询问如何处理，在 x 轴上从小到大枚举 p，将 e_{ij} 看作一个事件。那么每个时间点对应的事件是若干个点的排名会发生变化，若干个排名区间的排名会反转。
- 询问前 k_i 大需要维护距离排序后的前缀和。反转长度为 l 的区间只需要修改对应区间前缀和的值，其他位置的前缀和值不会变化，这部分只需要 $O(l)$ 的时间。
- 可以在总时间复杂度 $O(n^2)$ 的时间内维护。

Solution

- 对于 $a_i \neq b_i$ 的情况，需要维护积分的前缀和。
- 当修改某个函数时，需要同步计算新的积分函数。
- 具体时间复杂度部分与 $a = b$ 的情况类似。
- 需要注意使用全整数排序事件，避免出现精度问题。
- 排序部分时间复杂度与排序算法相关，使用快速排序为 $O(n^2 \log n + q \log q)$，其余部分时间复杂度 $O(n^2 + q)$。

Problem F. Challenge NPC III

Shortest Judge Solution: 1411 Bytes

Description

给定一张有向图，求是否有长度不超过 k 的简单路径包含重复颜色的点。

- $n, m, k \le 10^5$。
- $1 \le c_i \le 50$。

Solution

- 对每种颜色求出同色最短路径，判断长度是否 $< k$ 即可。
- 具体实现上，对每种颜色求多源最短路（和次短路），并记录最短路的起点，确保用于更新答案的起点和终点不同即可。
- 时间复杂度 $O((n+m)\max(c_i))$。

G. Tariff-ied

Shortest Judge Solution: 1020 Bytes

Description

求整数序列 $\{l_i\}$ 使得 $\prod(1+\frac{l_ik_i}{100})$ 最大，且 $\sum l_i = m$ 其中 k_i 是给定的实数。

- $T \leq 5000$
- $n \leq 40$。
- $m \leq 10^6$。
- $1 \leq l_i \leq 100$。

Solution

- 在同一个位置”加点”次数越多，倍率越低。
- 存在一个贪心算法，每次选择倍率最高的一个, 调整法容易证明这是正确的.
- 一定存在一个倍率使得倍率大于这个的操作都被执行. 二分加点操作的倍率，然后判断加点次数是否足够即可。
- 时间复杂度 $O(n\log(\epsilon^{-1}))$。

H. Sweet Sugar IV

Shortest Judge Solution: 2079 Bytes

Description

$n \times m$ 的网格上有一些矩形障碍。这些矩形不会重叠，边界和角落也不会互相触碰。没有任何一个矩形贴着网格边缘。

在空白位置分布着 d 块 Sugar，每块 Sugar 的甜度不同。

q 次独立的询问，每次询问给出 S_x, S_y, k，问假如在 (S_x, S_y) 放置 k 个棋子，能收获的 Sugar 的甜度之和最多是多少。

棋子只能往右往下移动，可以重叠但不能进入障碍格，存在一个棋子与某块 Sugar 重合时才能拿走对应的 Sugar。

- $n \times m \leq 10^6$。
- $d \leq 2 \cdot 10^5$。
- $q \leq 3 \cdot 10^5$。
- $k \leq 5$。

Solution

- 矩形障碍不贴墙，也不八方向连通，因此：
  - 从 $(1,1)$ 出发一定可以到达每个空白位置。
  - 从每个空白位置出发一定可以到达 (n,m)。
  - 当然，这是个平面图。
- 从 $(1,1)$ 出发，按照先向右，再向下（顺时针）的优先级进行 DFS，得到每个格子的出栈顺序 $A(x,y)$。
- 再从 $(1,1)$ 出发，按照先向下，再向右（逆时针）的优先级进行 DFS，得到每个格子的出栈顺序 $B(x,y)$。
- 根据平面图的性质，从 (x,y) 出发能到达 (x',y') 当且仅当 $A(x,y) \geq A(x',y')$ 且 $B(x,y) \geq B(x',y')$。

Solution

- 事实上，我们只关心 Sugar 之间的可达性，因此只需记录每个格子出栈前已经出栈了多少个放置着 Sugar 的格子作为 $A(\cdot,\cdot)$ 和 $B(\cdot,\cdot)$。
- 分别以 $A(\cdot,\cdot)$ 和 $B(\cdot,\cdot)$ 为横纵坐标，建立新的 $d \times d$ 坐标系：对于位于格子 (x, y) 的甜度为 w 的 Sugar，在新坐标系的 $(A(x,y), B(x,y))$ 处放置 w 个点。
- 对于询问 $\langle S_x, S_y, k\rangle$，等价于在新坐标系的 $[1, A(S_x,S_y)] \times [1, B(S_x,S_y)]$ 范围中找出不超过 k 条不相交的上升序列，使得总长度最大。

Solution

- 在新坐标系中按横坐标从小到大依次考虑每个 Sugar 拆出来的点以及询问，动态增量维护关于纵坐标的 $\max\{k\} = 5$ 层杨氏图表，以消除横坐标这一维度的限制。
- 一个询问的答案等于杨表前 k 层中不超过 $B(S_x, S_y)$ 的元素总数，这是因为仅保留这些元素构造出的杨表一定是全量元素构造的杨表的前缀。
- 在维护杨表的时候，由于每个 Sugar 可能被拆出来非常多的点，需要把数值相等的一堆点合并在一起计数表示，比如通过 `std::map` 实现，这样杨表操作次数不超过 $2^k d$。
- 配合树状数组动态维护每层杨表的值域前缀和来回答询问。
- 总时间复杂度 $\mathcal{O}(nm + 2^k d \log d + qk \log d)$。

I. Version Number

Shortest Judge Solution: 134 Bytes

Description

给两个版本号，比较哪个更新。

- 保证数字不超过 9 位。
- 保证最多三个分隔符。

Solution

- 按题意模拟即可，按照点分段后依次比较每一段。
- 时间复杂度 $O(n)$。

Problem J. Chocolate Game

Shortest Judge Solution: 1726 Bytes

Description

给定一个 $n \cdot m$ 的巧克力块，第 i 行第 j 列的好吃程度为 $a_{i,j}$。

先后手轮流操作，每次可以吃掉剩余巧克力的最后若干行或最后若干列，不能不吃。

如果操作后剩下的总好吃程度 $\leq s$，那么操作的人输掉游戏。

给定 q 个 s_i，询问先后手胜负。

- $1 \leq n \cdot m \leq 2 \cdot 10^5$
- $1 \leq q \leq 2 \cdot 10^5$

Solution

- 考虑对于一个 s 怎么计算。
- 注意到每行每列只有一个必败态，假设存在一个必败态 $f(x_0, y_0)$，所有 $f(x, y_0)$ $(x > x_0)$ 和所有 $f(x_0, y)$ $(y > y_0)$ 一定是必胜态。
- 可以按照行从小到大依次推出必败态。这样可以在 $O(\min(n, m))$ 时间复杂度内处理询问，可以通过。
- 假设当前第 i 列的前 c_i 行不合法，且前 d 列完全不合法。那么先手必败的充要条件为对于 $i < m$，$i - d - (n - c_i) < 0$，且 $m - d - (n - c_m) = 0$。
- 按照 s 从小到大处理询问，动态维护这些量即可在 $O(nm \log(nm) + q \log q)$ 的时间复杂度内解决本题。

K. Wavelike Finding

Shortest Judge Solution: 2957 Bytes

Description

给定参数 k 和 W。

定义序列 $a_1, a_2, \ldots, a_{2k}$ 是 “摇晃的” 当且仅当 $\sum_{i=1}^{k}(k-i+1)(a_i+a_{2k-i+1}) \geq W$，且它的长度恰好为 $2k$。

定义序列 $b_1, b_2, \ldots, b_m$ 是 “波浪序列” 当且仅当它存在至少一个 “摇晃的” 子序列。

给定序列 $s_1, s_2, \ldots, s_n$，问从中最多能找出多少段不重叠的连续子串，使得每个子串都是 “波浪序列”。

- $n \leq 2 \cdot 10^5$。
- $k \leq \frac{n}{2}$。

Solution

- 首先考虑如何判断答案是否为 0。
- 定义一个长度为 m $(m \le k)$ 的序列 $a_1, a_2, \ldots, a_m$ 的 “半边波浪值” 为 $\sum_{i=1}^{m} w_i \cdot a_i$，其中 $w_i = k - i + 1$。
- 设 $f_{i,j}$ 表示序列 s 的 $[1, i]$ 前缀中选择 j 个数得到的子序列的最大可能 “半边波浪值”，$g_{i,j}$ 表示 s 的 $[i, n]$ 后缀中选择 j 个数逆序得到的子序列的最大可能 “半边波浪值”，则序列 s 是 “波浪序列” 当且仅当 $\max_{i=k}^{n-k} \{f_{i,k} + g_{i+1,k}\} \ge W$。
- 如果可以计算 $f_{\cdot,\cdot}$，就能倒着用同样的方法计算 $g_{\cdot,\cdot}$，接下来考虑如何计算 $f_{\cdot,\cdot}$。

Solution

- $f_{i,j} = \max\left\{f_{i-1,j}, f_{i-1,j-1} + w_j \cdot s_i\right\}$，两种转移分别表示：
  1. 维持 $f_{i-1,j}$ 的最优子序列。
  2. 在 $f_{i-1,j-1}$ 的最优子序列末尾新增 s_i。
- 固定 i，可以证明一定存在一个分界线 x 满足 $f_{i,<x}$ 取第一种转移，$f_{i,\geq x}$ 取第二种转移。
- 将 i 这一维滚动掉，设 $d_j = f_j - f_{j-1}$，二分找到最小的 x 满足 $d_x < w_x \cdot s_i$，根据状态转移方程可得 d 变化后为 d'，其中：
  - $d'_y = d_y \ (y < x)$
  - $d'_x = w_x \cdot s_i$
  - $d'_y = d_{y-1} - s_i \ (y > x)$
- 使用平衡树维护 d，需要支持树上二分、单点插入、后缀打标记减去一个数，并在点数超过 k 时删除最右边的元素。

Solution

- 最后对平衡树中所有元素求和即可还原出每个 $f_{\cdot,k}$。
- 至此可以在 $\mathcal{O}(n\log k)$ 时间内判断一个长度为 n 的序列是否是 “波浪序列”。
- 为了找出尽可能多不相交的波浪子串，重复贪心地二分找到最短的满足条件的前缀即可。
- 需要注意的是，这里在二分长度之前应该先从小到大倍增找到最小的 t 满足长度为 2^t 的前缀合法但是长度为 2^{t+1} 的前缀不合法，然后再在 $[2^t, 2^{t+1})$ 进行二分。
- 在 $[2^t, 2^{t+1})$ 二分的时间复杂度为 $O(2^t\log n\log k)$，而 $\sum 2^t = \mathcal{O}(n)$，因此总时间复杂度为 $O(n\log n\log k)$。

Appendix

- 附录：证明存在一个分界线 x 满足 $f_{i,<x}$ 取第一种转移，$f_{i,\geq x}$ 取第二种转移。
- 考虑按照某个顺序贪心地选择每个元素插入到最终的子序列中依次得到 $f_{n,1}, f_{n,2}, \ldots, f_{n,k}$。
- 令 $Q(i)$ 表示下一步如果加入 s_i 会对 “半边波浪值” 有多少贡献，令 i 左边已经选择了 cnt 个元素，i 右边已选择的元素之和为 sum，那么 $Q(i) = w_{cnt+1} \cdot s_i - sum$。
- 一开始所有元素都未选择，可以证明每次贪心挑选 Q 值最大的元素插入到最终的子序列中即可得到最优解。
- 在上述命题成立的前提下，自然可得存在一个分界线 x 满足 $f_{i,<x}$ 取第一种转移，$f_{i,\geq x}$ 取第二种转移。

Appendix

- 引理：如果 $i < j$ 且 $s_i < s_j$，那么 j 一定早于 i 被贪心算法选中。证明：
  - 如果 i 和 j 中间不存在别的已选元素，那么 $Q(i)$ 和 $Q(j)$ 的大小关系等价于 s_i 和 s_j 的大小关系，显然有 $Q(i) < Q(j)$。
  - 否则不妨先移除 i 和 j 中间的所有已选元素，此时 $Q(i) < Q(j)$，根据引理，中间早于 j 选中的每个元素 k 都满足 $s_k \geq s_j$，对 $Q(i)$ 的贡献为 $-s_k$，小于等于对 $Q(j)$ 的贡献 $-s_j$，所以 $Q(i) < Q(j)$ 依然成立。
- 接下来利用引理证明贪心算法的正确性。

Appendix

- 假设贪心算法先选择了集合 A 的元素（此时仍能达成未来的最优解），然后再选择了此时 Q 值最大的元素 x，导致未来无法选择 $|B|-1$ 个元素达成最优解 $A\cup B$：
  1. B 中所有元素都在 x 右侧，考虑 B 中最靠左的元素 y。在贪心算法选中 x 时 $Q(x)\geq Q(y)$，而 B 中其它元素对 $Q(x)$ 和 $Q(y)$ 的贡献相同，因此把 B 中的 y 换成 x 不会变劣，与假设矛盾。
  2. B 中存在一个元素在 x 左侧，考虑 B 中 x 左侧第一个元素 y。在贪心算法选中 x 时 $Q(x)\geq Q(y)$，B 中在 x 右侧的元素不改变 $Q(x)$ 与 $Q(y)$ 的大小关系；B 中在 y 左侧的元素对 $Q(x)$ 和 $Q(y)$ 的贡献分别为 $-s_x$ 与 $-s_y$。
     - 如果 $s_x\leq s_y$，那么 $Q(x)\geq Q(y)$ 依然成立，把 B 中的 y 换成 x 不会变劣，与假设矛盾。
     - 如果 $s_x>s_y$，根据引理，介于 y 与 x 之间的每个 A 中的数都至少为 s_x，全部放缩为 s_x 后，把 B 中的 y 换成 x 产生的贡献是若干倍 $s_x-s_y>0$，因此不会变劣，与假设矛盾。

L. Nailoongs Always Lie

Shortest Judge Solution: 1455 Bytes

Description

有 n 个生物，每个生物会说一个生物是奶龙。奶龙说的都是谎话，其他生物说的可能是真话可能是谎话。问至多有几个奶龙。

- $1 \leq n \leq 10^5$

Solution

- 限制相当于任意相邻两个生物不能同时为奶龙。同时，将 i 向 a_i 连边后图为基环树。
- 每次可以贪心将一个度数最小的点标记为奶龙，然后将和其相邻的点标记为不是奶龙，并断开这两个点和其余点的连边。统计这样贪心后有多少奶龙即可。
- 另一种做法是在环套树上 DP，也可以通过本题。
- 时间复杂度 $O(n)$。

Problem M. Master of Both VII

Shortest Judge Solution: 1274 Bytes

Description

有一个隐藏的 n 个点正多边形的三角剖分，你需要在 $n-3$ 次询问还原这个三角剖分。

可以询问 (x, y)，返回要么边 (x, y) 存在，要么返回和边 (x, y) 在非端点处相交的边数。

- $1 \leq n \leq 100$

Solution

- 考虑询问所有 $3 \le i \le n-1$ 的 $(1,i)$，结果为 d_i。对于 $d_i = 0$ 直接加入答案.
- 对于一条边 (x,y)，其会对 $x < i < y$ 的询问有 1 的贡献。
- 由于这些边构成了一些不会在非端点处相交的区间，按照 $2 \le i \le n$ 的顺序维护一个栈。
- 如果 $d_i > d_{i-1}$，那么入栈 $d_i - d_{i-1}$ 个 $i-1$。
- 如果 $d_i < d_{i-1}$，那么弹出 $d_{i-1} - d_i$ 个栈顶，构成边 (st_{top}, i)。
- 由于不会同时存在 $i-1$ 开始的区间与 i 结束的区间, 可以证明这个做法的正确性。
- 时间复杂度 $O(n)$。

Thank you!