2024 年（第十六届）四川省大学生程序设计大赛

2024 四川省赛命题组

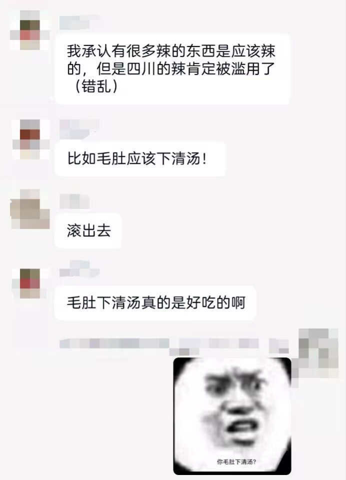
2024 年 6 月 13 日

Problem L: 毛肚下清汤？

题意

准备 2 口锅，然后给出下到两个锅熟的时间 + 能下到哪个锅，每个菜都会被下到他能被下到且煮熟时间最短的锅中，问两个锅中分别有哪些菜以及每个菜捞出来的顺序。$n \leq 10^5$

Fun fact:


我承认有很多辣的东西是应该辣的，但是四川的辣肯定被滥用了（错乱）
比如毛肚应该下清汤！
滚出去
毛肚下清汤真的是好吃的啊
你毛肚下清汤?

判断每个菜需要被下到哪个锅中，然后对于两口锅中的菜煮熟的时间进行排序后按题意输出即可。

Problem H: GG 和 YY 的石子游戏

题意

GG 和 YY 正在用一堆 n 个石子玩游戏。GG 和 YY 轮流操作，GG 先操作。在每一轮中，玩家可以从石子堆中移走 1 或 2 个石子。不能操作的一方输。两位玩家都想获胜，并得到尽可能多的石子。假设 GG 和 YY 都采用最优策略。请确定胜者，并回答胜者最后获得的石子数 v。

多组数据，其中 $1 \leq T \leq 10^4, 1 \leq n \leq 10^{12}$

若 $n \mod 3 = 0$，YY 有必胜策略，每当 GG 取 i 个石子时，YY 取 $3-i$ 个石子保证必胜，GG 由于不能获胜，则考虑贪心的方式来取得尽可能多的石子，因而不难得到结果。

若 $n \mod 3 > 0$，GG 有先手必胜策略，即选取 $n \mod 3$ 个石子，然后情况和前一种情况类似。因此，对于任意题目范围内的 n，均可 $O(1)$ 回答结果。

Problem E: L 型覆盖检查器

题意

给你一个 $W \times H$ 的矩形，有 UDLRC. 六种字母，. 表示未被覆盖，C 表示某个 L 的中心，U 表示它对应的 L 的中心在它上面一个，其他类似，问你是不是一个合法的 L 覆盖。

多组数据，其中 $W, H \leq 500$

模拟即可。

一个比较简单的写法是根据方向向中心点加值，统计每个点的值模 3 为 0 且总数 $= n \times m - 1$。

Problem I: 集装箱调度

题意

给你 n 个矩形放进一个大矩形里，不能旋转，矩形间不能重叠，每次选能放下它的位置中左下角 x 最小的，有多个就选 y 最小的，输出 n 个矩形最终的左下角坐标，放不下的输出 -1。

$n \leq 50$

模拟题，做法也较多，$O(n^4)$ 或者 $O(n^5)$ 的做法都可以通过，这里会介绍一种容易实现的 $O(n^4)$ 做法和 $O(n^3)$ 做法。

做法 1:

对于每个新的矩形，它的左边或者下边一定是靠在边界上或者某个已经放了的矩形的边界，对于每一维，候选的坐标有 $O(n)$ 个，我们维护每一维的坐标，每来一个矩形我们 $O(n^2)$ 枚举可能的左下角位置，再 $O(n)$ 去检查是否跟已放置的矩形有重合即可，总复杂度 $O(n^4)$。

做法 2:

沿用上面的结论，我们每放一个新矩形前，对每一维的坐标离散化，并且用已放置的矩形在二维网格上标 1，那么检查新位置是否合法就变成了去看新位置对应的矩形是否全为 0，这个可以用二维前缀和 $O(1)$ 判断，整体复杂度 $O(n^3)$。

Problem F: 小球进洞：平面版

题意

有一个固定的矩形，一个大小固定的圆沿一个方向运动，问有没有一个时刻圆完全在矩形里。

多组数据，其中

$1 \le T \le 10^4, -10^6 \le x, y, v_x, v_y\ l_x, l_y, r_x, r_y \le 10^6$,$1 \le r \le 10^6$

一个圆在矩形内部的充要条件是它的圆心到矩形四条边的距离都 $\geq r$。

因此我们把矩形每条边都向矩形中心移动 r 的距离得到新的矩形. 于是问题变成是否存在一个时刻，圆心在新的矩形内。

这个问题可以通过求解圆心与方向的射线和新矩形某条边是否有交点即可，一个简单的写法是直接直线求交，判断交点是否在线段上，以及交掉对应的时刻是否 ≥ 0。

Problem J: 罗马数字

题意

用罗马数字和阿拉伯数字在一定要求下表达一个十进制数，每种数字的使用需要一定代价，求表达某个数的最小代价。
多组数据，其中 $1 \leq T \leq 2 \cdot 10^3, 1 \leq n \leq 10^{18}, cost \leq 10^7$

做法 1:

从低位往高位考虑，$f(i,s)$ 表示在最低的 i 位填了数字后，十进制下需要向前进位 s 时的最小 $cost$，转移时枚举这一位填的数字，使得这一位模 10 与 n 这一位相同，即 $f(i,s) \to f(i+1, \frac{(s+val(j))}{10}) + cost$

总的复杂度为 $O(LSK)$，L 是数字位数，S 为最多进位的大小，K 为候选数字集大小，本题中 $L=19, S=111, K=17$ ，这种做法可以扩展到任意多个候选数字集上，只要候选数字不大，保证最多进位的大小不大即可。

做法 2:

从低位往高位考虑，$f(i, s, p0, p1, p2)$ 表示考虑最低的 i 位现在进位是 s ，比 i 更低的 3 位是否填了数时的最小 $cost$，每一位如果要填 ≥ 10 的数，因为这些数模 10 都为 0，可以都在更大的 i 去考虑，因此转移是枚举更低的三位填什么数，以及 i 这一位填个位数或者不填，因此 s 只能为 0 或者 1。

复杂度是 $O(L2^{n+1}m^n)$，L 是数字位数，n 是 ≥ 10 的数字的最大位数，m 是 ≥ 10 的数字在每一位的数量，本题 $n=3, m=2$，这种做法可以处理 ≥ 10 的数字都模 10 等于 0 的情况，并且数字的大小可以很大。

Problem B: 连接召唤

题意

给你能力值为 1 到 5 的精灵的数量，问你最多能连接召唤出多少能力值为 6 的精灵？

连接召唤规则：选 k 只精灵出来，每个能力值为 x 的精灵当做 x 或者 1，求和为 s，得到能力值为 s 的精灵一只。

多组数据，其中 $\sum a_i \leq 10^9, T \leq 10^5$

此题我们考虑贪心求解即可。

首先考虑只需要 2 只精灵就可以合成得到能力值为 6 的精灵的方式：$(3,3),(2,4),(x,5)$。

然后考虑 3 只精灵的情况 $(4,x,y),(3,2,x)$，这里所有的 x,y 都优先考虑能力小的。

最后考虑剩下的 $1,2,3$ 来合成能力值为 6 的精灵，它们能构成的数量是 $\frac{(a_1+2a_2+3a_3)}{6}$。

Problem G: 函数查询

题意

给定 n 个整数 $x_1, x_2, \ldots, x_n$。现有 q 次询问，每次询问给定一个函数 $f(x) = (a \oplus x) - b$，请判断是否存在 $1 \le i < n$ 满足 $f(x_i) \cdot f(x_{i+1}) \le 0$，如果存在请输出一个满足条件的 i，否则输出 -1。

$a \oplus b$ 表示 a 异或 b。

$2 \le n \le 3 \times 10^5$，$1 \le q \le 3 \times 10^5, 0 \le x_i \le 10^9, 0 \le a, b \le 10^9$

本题做法较多，有多种做法，都基于一个通用的结论，如果 $\max f(x) \cdot \min f(x) \leq 0$，那么一定是有解的，证明见做法 1，同时对于 $\max f(x)$ 和 $\min f(x)$ 的求解，可以通过 01Trie 在 $O(\log C)$ 的时间回答单次询问

做法 1:

在线二分，对于每个询问找出 $\max f(x)$ 的位置 l, $\min f(x)$ 的位置 r, 我们证明在 $[l, r]$ 内一定有解。

我们去看 $mid = \frac{l+r}{2}$ 的位置，因为 $f(x_l) \cdot f(x_r) \le 0$ ，那么 $f(x_l) \cdot f(x_{mid}) \le 0$ 和 $f(x_{mid}) \cdot f(x_r) \le 0$ 至少有一个成立，我们可以把区间减半，直到 $l + 1 = r$ ，此时 l 就是一个合法的位置。

这个做法可以强制在线，并且对于任意可以在 $O(1)$ 求出函数值，$O(\log C)$ 求出 min 或 max 的函数，都可以做到 $(n + q)(\log C + \log n)$ 的复杂度。

做法 2:

离线整体二分，对于每个询问，去看 $[1, r_i]$ 有没有解，那么每次二分时，我们按顺序把数添加进，在 r_i 时去找这个询问对应的 min 或 max，去看是否满足 $\max f(x) \cdot \min f(x) \le 0$，找到的最左边的 r_i 对应的 $r_i - 1$ 就是一个合法的答案。

这个做法总体复杂度 $(n+q)\log n \ \log C$。

这个做法对于任意可以在 $O(1)$ 求出函数值，$O(\log C)$ 求出 min 或 max 的函数，都可以对每个询问找出最左边的满足条件的位置，但这个做法在本题时限下会 TLE。

做法 3:

维护 01Trie 每个节点记录经过这个节点的数最左边的位置，对于每个询问，$f(x)=0$ 的值可以去查询 $a \oplus b$ 出现的最左边位置得到 p_{equal}。

然后我们分别去找最左边的 $f(x)>0$ 和 $f(x)<0$ 的位置 p_{big} 和 p_{small}，这个可以通过在字典树上去找某个节点满足前面的位 $= a \oplus b$，当前位 $\neq a \oplus b$ 去得到，如果这个 b 这一位为 1，$a \oplus x$ 这一位为 0，那么这个节点的位置是 p_{small} 的候选位置，同理如果这个 b 这一位为 0，$a \oplus x$ 这一位为 1，那么这个节点的位置是 p_{big} 的候选位置，在这 $\log C$ 个候选节点的位置分别取最小值就可以得到 p_{big} 和 p_{small} 。

做法 3:

如果 $p_{equal}=1$ 那么答案是 1 ，要么 p_{big} 和 p_{small} 有一个为 1 ，另一个就是答案，也就是答案为 $\max(p_{big},p_{small})-1$。

总体复杂度 $(n+q)\log C$, 这个做法可以强制在线并且回答最左边的满足条件的位置，但是只能回答形如 $f(x)=a\oplus x-b$ 的函数。

Problem K: 元素反应

题意

有 m 种元素，告诉你两两反应的伤害，同样的元素也可以反应。给你一个长度为 n 的串，表示给怪物施加元素的顺序，每次施加元素时，每次若怪物身上挂有元素就会发生反应对怪物造成伤害并且仅留下当前元素，否则将当前元素挂在怪物身上。
现在我们把 m 种元素设为惰性的，即不参与任何反应，也不会被挂在怪物身上。
问你 2^m 种方案的伤害分别是多少？
$n \leq 10^5, m \leq 17, a_{i,j} \leq 10^8$

做法 1:

用 $f(i,s)$ 表示某种元素 i 不惰性，且集合 s 的元素为惰性时，产生的所有 $a(i,x)$ 的伤害，那么我们可以枚举 i 出现的每个位置，去看它可能与哪些位置发生反应，显然每个元素 j 只会取 i 对应位置右边最左边的第一个 j ，因此最多 m 种可能的反应，并且每种反应需要 i,j 之间的其它元素集合 s_j 均为惰性，且 i,j 不为惰性。

做法 1:

那么所有的 $f(i,s)$ { s_j 是 s 的子集，$s_j \cup j$ 不是 s 的子集 } 都需要加上 $a(i,j)$，也就是所有的 $f(i,s)$ { s_j 是 s 的子集 } $+= a(i,j)$. 所有的 $f(i,s)$ { $s_j \cup j$ 是 s 的子集 } $-= a(i,j)$。

这是一个典型的 sos(sum of subset) 问题，可以 $m2^m$ 完成，找这 m 种可能反应位置可以 $O(m^2)$ 去做，也可以优化到 $O(m)$。

最后对每种元素 i 做一次 sos，再把答案累加即可，总复杂度 $O(nm^2 + 2^m m^2)$。

做法 2:

首先可以对于序列每个位置，考虑这个位置 i 和后面某个元素反应产生的贡献，并计算有多少种方案能够产生这个贡献。

设 s_i 表示序列第 i 个元素的种类。由于每种元素都会一起惰性，所以可以直接枚举后面和哪种元素 x 反应，那么 i 之后第一个这种元素的位置 j 就是所反应的。并且 i 到 j 之间其他所有种类的元素都一定是惰性,s_i, s_j 两个元素是一定不惰。其余的不在 $[i, j]$ 之间的所有元素种类都是可以惰性可以不惰。

由于每种元素只需要枚举第一个，复杂度是 $O(nm)$。

做法 2:

考虑设 f_s 表示 s 二进制位中为 1 的元素为是一定惰性，0 的元素可能惰性可能不惰时的方案数。那么如果在上面没有 s_i, s_j 两种元素一定不惰的要求的话，直接做一个高维前缀和（FMT）即可。现在有 s_i, s_j 一定惰性的要求。那么考虑容斥：

- s_i, s_j 可惰可不惰的加 a_{s_i,s_j}；
- s_i 惰，s_j 可惰可不惰的减 a_{s_i,s_j}；
- s_i 可惰可不惰，s_j 惰的情况减 a_{s_i,s_j}；
- s_i, s_j 都惰的情况加 a_{s_i,s_j}。

通过这样可以解决 s_i, s_j 一定惰性的要求。然后做一个 FMT 即可。

复杂度 $O(n * m + 2^m * m)$。

Problem D: L 型覆盖

题意

给你右上角空了一个 1×1 的矩形的 $W \times H$ 的网格图，为你能否用题中给定的 L 型进行覆盖，给出方案。

多组数据，其中 $W, H \leq 500$

$n \times m \bmod 3 \neq 1$ 的时候一定没有合法方案，因为每个 L 能覆盖三个格子，最后覆盖的格子数量一定是 3 的倍数。

$n \times m \bmod 3 = 1$ 的时候一定有可行的方案，可以通过多种方式构造出来，下面给出两种做法：

做法 1:

递归，因为 $n \times m \bmod 3 = 1$, 所以一定是 $n \equiv m \pmod 3$, 对于 $n \bmod 3 = 1$ 的情况我们可以拆成 $2 \times (n-2), 2 \times 2, (n-2) \times (m-2), (n-2) \times 2$ 的四个小矩形，2×2 是有. 的部分，直接填一个 L 就可以，然后在剩下三个矩形交界的地方，每个矩形角落出一个格子构成 L，那么剩下就是去求解三个 $n \bmod 3 = 2$ 的子问题。

对于 $n \bmod 3 = 2$ 的情况类似，我们可以拆成 $4 \times (n-4), 4 \times 4, (n-4) \times (m-4), (n-4) \times 4$ 的四个小矩形，变成求解 $n \bmod 3 = 1$ 的子问题。

注意递归到小范围的时候需要手动构造 $5 \times 5, 4 \times 4$ 的解。

做法 2:

我们容易构造出 2×3 的覆盖方案，那么对于 $6 \times X(X \geq 2)$ 的矩形我们都可以构造，奇数场景我们先用三个 2×3 构造 6×3，剩下偶数情况我们用两个 3×2 构造 6×2，拼接就好。

那么对于 $n \times m$ 的场景，我们先每 6 列去构造，再每 6 行去构造，那么剩下的一定是 $n, m \leq 7$ 的场景，手动构造 $4 \times 4, 5 \times 5, 7 \times 7$ 的解就好，4×7 可以由 1 个 $4 \times 4+$ 2 个 2×3 得到。

Problem C： 黑白立方格

题意

给定一个大小为 $N \times M \times L$ 的立方格，其中每个格点 (i,j,k) 要么是白色要么是黑色，格点的坐标均为正整数。设翻转格点 (i,j,k) 颜色的代价为 $a_{i,j,k}$。求翻转格点颜色的最小代价，使得左下角 $(1,1,1)$ 为黑色，右上角 (N,M,L) 为白色，且任意两个不同格点 (i_1,j_1,k_1) 和 (i_2,j_2,k_2) 至少满足以下条件之一:

- $i_1 > i_2$
- $j_1 > j_2$
- $k_1 > k_2$
- (i_1,j_1,k_1) 是黑色的。
- (i_2,j_2,k_2) 是白色的。

$1 \leq N,M,L \leq 5 \times 10^3$，$2 \leq N \times M \times L \leq 5 \times 10^3$，$cost \leq 10^5$

首先我们考虑什么样的情况是不满足条件的。

对于 (i,j,k) 的格子，如果它是黑色的，那么 $(i-1,j,k),(i,j-1,k),(i,j,k-1)$ 三个格子都得是黑色的，证明如下：

必要性：如果某个格子是白色的，那么已经违反了约束。

充分性：对于任意两个格子 $(i_1,j_1,k_1),(i_2,j_2,k_2)$，如果满足 $i_1 \le i_2 \ \& \ j_1 \le j_2 \ \& \ k_1 \le k_2 \ \& \ (i_1,j_1,k_1)$ 是白色 & (i_2,j_2,k_2) 是黑色，那么一定能找到一条从 (i_1,j_1,k_1) 到 (i_2,j_2,k_2) 的路径，使得相邻格子只在某一维差 1。因此可以有 (i_2,j_2,k_2) 是黑色逐步推理得到 (i_1,j_1,k_1) 是黑色，矛盾。

有了这个结论后，我们可以这样建一个网络流的图：

- 对于白色格子，它连向汇点，流量为翻转的代价。
- 对于黑色格子，源点连向它，流量为翻转的代价。
- 对于 (i, j, k) ，分别向 $(i-1, j, k), (i, j-1, k), (i, j, k-1)$ 连边，流量为 inf。

那么这个图的一个割就是一个合法的染色方案，因为不存在如下路径：

源点 $\rightarrow$ 黑色的 (i, j, k) $\rightarrow$
黑色的 $(i-1, j, k)/(i, j-1, k)/(i, j, k-1)$ $\rightarrow$ 汇点。

从而最小割就是答案。

Problem A: 逆序对染色

题意

给你一个长度为 n 的排列，对于每一对 $a_i > a_j$ 且 $i < j$，在 $n \times n$ 的网格图上把 (i, a_j) 染色，问你最终网格图上四连通的连通块数目。

$n \leq 3 \times 10^5$

按行从小到大考虑，每行去维护可能染色的区间段，初始为 $[1, n]$，后面每次处理第 i 行时，去查询 $[a_i + 1, n]$ 有多少可能的染色区间段，新增的连通块就是第 i 行有，第 $i - 1$ 行没有的段。

处理完第 i 行后，第 a_i 列往后都不会被染色，这会把某个连续的区间切开，查询可以用树状数组，更新可以直接维护某个点是不是染色区间的左端点即可，总复杂度 $O(n \log n)$。

Thanks for attention!