ICPC Asia EC Final 2024 题解

ICPC Asia EC Final 2024 命题组

December 29, 2024

本场比赛将在几周后的 Universal Cup 上作为 Stage: China 进行比赛，请不要将题解发给未参赛的选手，谢谢。

A. Hitoshizuku

$3n$ 个人要组成 n 个三人乐队，其中每个人有一个自己的魅力值 a_i，且所在队其他人的魅力值不能超过 b_i。构造方案或者判断无解。

也即：给定 $3n$ 个区间 $[a_i, b_i]$，要求分成 n 个三元组，使得每组内交集非空。

$n \leq 10^5, 1 \leq a_i \leq b_i \leq 10^9$.

Author: Kevin114514;

Prepared by Kevin114514, apiad & fstqwq

A. Hitoshizuku

考虑贪心：将所有区间的左右端点拿出来排序。

在考虑一个区间左端点时，将其右端点加入集合。

在考虑一个区间右端点时，如果其没有被删除，那么将其和集合内另外两个右端点最小的区间删除并匹配[1]。

由调整法可得这样做一定是最优的。

时间复杂度 $O(n \log n)$。

[1] 如果右端点两两不同（或者认为带有标号作为次关键字），那么事实上就是弹出并匹配堆顶的三个区间。

B. Guess the Polygon 2

乱序给出一个简单多边形的 n 个顶点，每次可以询问直线 $ax + by + c = 0$ 与简单多边形的相交部分总长，交互器以 $r\sqrt{a^2 + b^2}/s$ 的形式返回结果，其中 r 和 s 都是整数且 $r \geq 0, s \geq 1, \gcd(r, s) = 1$，要在 $(n - 2)$ 次询问内还原出简单多边形，按照逆时针顺序输出 n 个顶点。

$3 \leq n \leq 200$，坐标是 $[0, 1000]$ 内的整数。

Author: quailty & apiad; Prepared by quailty & cubercsl

B. Guess the Polygon 2

考虑字典序最小的点 P，从 P 连出去两条边构成一个有向角的方式有 $O(n^2)$ 种，选取斜率负充分大且足够接近 P 的直线就可以只截到这个角，其中只有 $O(n^4)$ 种斜率不能完全区分所有连边方式，那么随机一个斜率就可以大概率区分出所有连边方式。这样可以在询问前进行检查，如果直线不够好就重新随机。

现在已经确定了一个有向角 $Q \to P \to R$，那么截出三角形 QPR 之后得到了 $n-1$ 个点的多边形，但是这个多边形可能不再是简单的，要利用截出三角形之后新得到的有向边 $Q \to R$。

B. Guess the Polygon 2

考虑剩下的字典序最小的点 P'，如果在之前的过程中还没得到 P' 的前驱 Q'，就需要进行枚举，对 P' 的后继 R' 同理，这样还是有 $O(n^2)$ 种连边方式。仍然按照最开始的策略选取直线，但是额外要求直线与之前截出的三角形的边界没有重合部分，这样就可以将询问结果减去直线与之前已经截出的三角形的交线长得到一个值。通过这个值可以确定一个有向角 $Q' \to P' \to R'$，截出三角形 $Q'P'R'$（可能是顺时针绕向，也可能退化）然后重复上述过程即可。

B. Guess the Polygon 2

通过 $n-2$ 次询问依次从多边形截出了 $n-2$ 个三角形，只剩下一条边，反过来依次加入三角形就可以还原多边形。计算截线长度时涉及到的数值都是 $\sqrt{a^2+b^2}$ 的分数倍，可以通过分数取模避免高精度运算。如果使用 hash map，通过一次询问确定一个角的复杂度是 $O(n^2)$，总复杂度是 $O(n^3)$。

C. Norte da Universidade

给定一个 $n \times m$ 的网格，每格将属于东、南、西、或北区中的一个。要求北区的正北方向上的相邻格子必须是北区，其余区域类似。

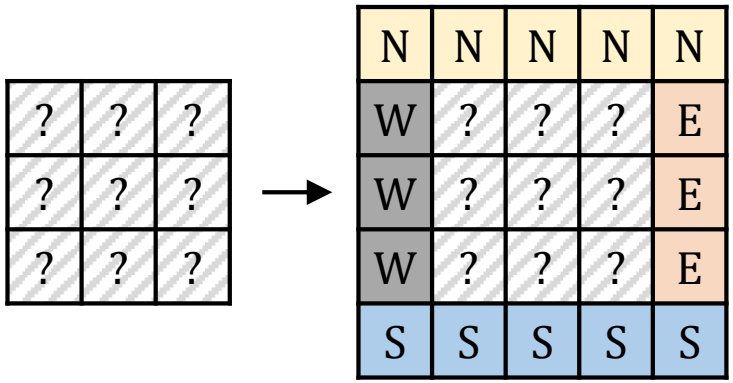
现在给定一个填了一部分的网格，其余格子可以任意填，问完成填空有多少种方案。

$1 \leq n, m \leq 1000$.

Author: fstqwq & apiad; Prepared by quailty & fstqwq

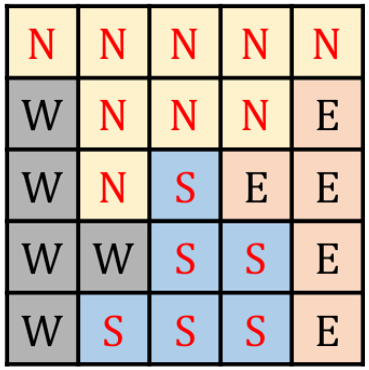
C. Norte da Universidade

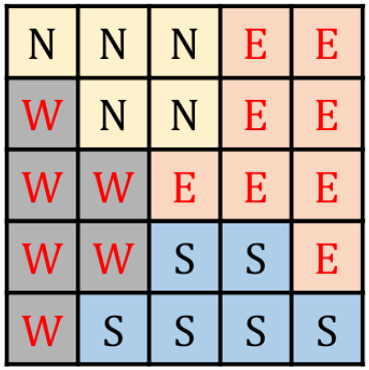
为了避免讨论有些区域在方案里可能不存在，可以先在 $n \times m$ 地图外围加一圈，保证各个区域都至少占据一个格子，如下图所示：

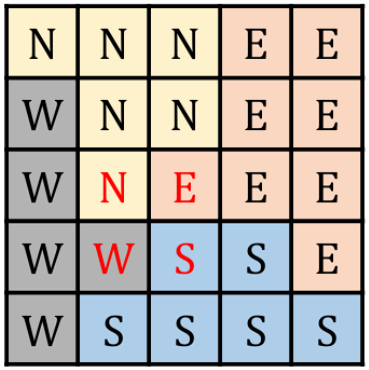


C. Norte da Universidade

考虑四个区域的分界线，如下图所示，有三种情况：







- N 区和 S 区有接壤，此时 W 区和 E 区一定没有接壤；
- W 区和 E 区有接壤，此时 N 区和 S 区一定没有接壤；
- N 区和 S 区没有接壤，W 区和 E 区也没有接壤。

C. Norte da Universidade

先考虑第一种情况，N-W 分界线和 N-E 分界线会收束到一个点，S-W 分界线和 S-E 分界线也会收束到一个点，这两个收束点之间通过一段 N-S 分界线连接。

首先观察 N-W 分界线，一定是从地图网格线的左上角出发，沿着网格线向下或者向右走出的一段折线，每次向下走要求左边都是 W 或者?，每次向右走要求上边都是 N 或者?，这样可以预处理出 N-W 分界线最后一条边是某条网格边的方案数组，类似可以预处理出另外三条相邻方向区域的分界线的方案数组。

C. Norte da Universidade

假设已经确定了两个收束点，现在需要一条 N-S 分界线连接这两个收束点，从 N-W 分界线和 N-E 分界线的收束点出发，每次可以向上、向下、向右走，每次向右走要求上边都是 N 的同时下边都是 S，在网格线上 dp 可以计算出所有可能的 N-S 分界线对答案的贡献。

C. Norte da Universidade

类似第一种情况可以计算出第二种情况的贡献。

对于第三种情况，四条相邻方向区域的分界线会收束在一个点，枚举这个点利用之前预处理的方案数组计算贡献即可，需要注意有两种可能的收束方向。

总复杂度 $O(nm)$。

D. Keystone Correction

一个投影仪被描述为一个源点的点光源和一个像矩形，投影到墙平面上。

投影可能不正，也可能不完全落在矩形荧幕上，因此可以选择像矩形里的四个角来确定一个四边形来校正画面，使得投影为一个与像矩形同宽高比且平行于地面的矩形，且完全落在荧幕上。

问校正后的画面最大面积是多少。

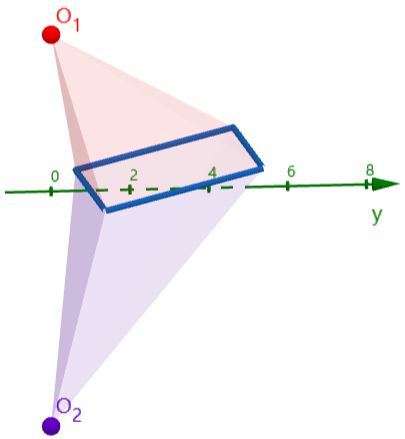
$T \leq 1000$，输入为整点，像矩形顶点值域 100，荧幕矩形值域 1000，并有一系列良定义保证。

Author: fstqwq; Prepared by fstqwq & quailty

D. Keystone Correction

算出 w 和 h 之后将透镜投影到屏幕所在平面上可以得到一个凸四边形 A，屏幕本身是一个矩形 B，问题转化成在 A 和 B 的交集内找一个长宽比是 $w:h$ 且长宽边对应和 B 的长宽边平行的矩形 C。

由于点光源发出的光可能从透镜的两面穿过，A 投影后的绕向可能是逆时针也可能是顺时针，在样例中已经有体现。



D. Keystone Correction

二分长边的长度 x，那么短边长度是 $y = xh/w$，考虑 C 的某个顶点的可行范围是 A 和 B 的交集按照 $(0,0)$、$(x,0)$、(x,y) 和 $(0,y)$ 分别偏移之后得到 4 个凸多边形区域的交集，可以使用半平面交相关算法判断可行范围是否非空。

二分后亦可以直接使用扫描线：同时维护两条相距为 x 的水平直线与凸包的交的范围，对范围求交后的长度是分段线性函数。只需要实现对平移前和平移后的分段线性函数取 min 即可，然后对上凸壳和（y 取负的）下凸壳进行平移取 min，随后在每个分段的端点求值判断。

D. Keystone Correction

使用半平面交需要注意精度。虽然所有输入均为整数，值域和面积均有限制，投影细长答案也不会太小。但是，投影、二分、直线求交点这三部分均会带来精度损失，累计精度可能会较差。

使用主流的排序后线性半平面交算法实现容易在一些平常忽略的细节（eps 的使用，直线求交点）上损失大量精度。

- 错误的写法会导致几乎平行或者退化的直线参与求交点，使得二分结果永远偏向 0 或者 ∞。
- 二分过程的末段因为 eps 的错误使用，导致判断交点在直线左右失效（或 eps 不一致），无法准确收敛于答案附近。

使用平方的直线切凸包算法（convex cut），以理论复杂度为代价，获得更大的精度宽容度，且在实现难度和常数上存在优势。

E. Corrupted Scoreboard Log

有一场有 m 道题的比赛[2]的排行榜中的空格丢失了，依据没有空格的排行榜信息还原 n 支队伍可能的信息。

$n \leq 500, m \leq 13$.

Author: cubercsl & Qingyu; Prepared by apiad & cubercsl

[2]题面中的故事为虚构。如有雷同，纯属巧合。

E. Corrupted Scoreboard Log

首先可以按照 try 或者 tries 将字符串划分开来。因为提交次数不超过 100，所以每个串最多两种划分（其中值得注意的是，100 tries 只有一种划分方式），然后搜索即可。注意对于一段可能需要枚举题数罚时，和提交的分隔符。

对于一组数据，枚举的量不会超过 $2^m \times 5$，可以通过。

可以通过折半搜索的方法做到更好的复杂度。

F. Boolean Function Reconstruction

构造一个只由不超过 $2^{n-1}+10$ 个二元 `and` `or` 组成的表达式，使得其满足给定的对于输入 n 个布尔变量输出一个布尔值的真值表。

$n \leq 15$.

Author: Lynkcat & apiad; Prepared by apiad & Lynkcat

F. Boolean Function Reconstruction

首先我们注意到 and 和 or 运算都有单调性：当输入的值从 0 变成 1 时，输出的结果是不降的，复合后的表达式亦然。因此，有解的充要条件是输入的真值表具有单调性。

先不考虑大小限制。对于一个单调的函数，我们有如下构造，找到所有 $f(x_1, x_2, \ldots, x_n) = 1$ 的状态，然后把这里面所有 $x_i = 1$ 的变量 and 起来。这样，我们得到了一个 $O(n2^n)$ 大小的满足要求的表达式。

为了进一步减小运算符的个数，我们需要两点优化：

- 只要找到极小的等于 1 的状态即可，因为这些状态等于 1，能直接推出包含它的集合也等于 1。

F. Boolean Function Reconstruction

- 把所有带 x_1 的子句中的 x_1 提取出来，写成

$$(x_1 \texttt{ and } f_1) \texttt{ or } f_2,$$

其中 f_1 和 f_2 分别为带 x_1 的子句提取出 x_1 之后的表达式和不带 x_1 的子句。然后对于这两个递归构造即可。

可以证明这样构造不会超过 $2\binom{n}{\lfloor n/2 \rfloor} = O(2^n n^{-1/2})$，所以可以通过。

如果没有注意到第一点，也可以对比较小的 n 进行搜索或者状压，得到最短的表达式。递归到变量比较少的时候，直接使用预处理的最短的表达式，这样也可以通过，甚至可能得到更好的结果。

G. Collatz Conjecture

从任意正整数 n 开始，然后按照以下步骤操作：

- 如果 n 是 A 的倍数，则将其除以 A：$n \to \frac{n}{A}$；
- 否则，加上 B：$n \to n + B$。

问 n 能否经过若干次操作回到自己。

$T \leq 10^5, 1 \leq A, B \leq 10^9, \gcd(A, B) = 1, 1 \leq n \leq 10^{18}$.

Author: Lynkcat & apiad; Prepared by apiad

G. Collatz Conjecture

我们把连续的 $+B$ 一直加到能 $/A$ 看成一组操作。这个操作在模 B 意义下可以看做 $\times A^{-1} \pmod{B}$。

因为 A, B 互质，根据欧拉定理有 $A^{\varphi(B)} \equiv 1 \pmod{B}$，所以在模 B 意义下一定会进入循环。

考虑数字 n 本身，一组操作之后会变成不超过 $(n + B(A-1))/A$。如果有 $n > B$，那么变换之后的数字一定小于 n。如果 $n \leq B$，那么变换之后的数字不超过 B。

所以按一组操作考虑，最后进入循环的一定是 $1 \sim B$ 之间的数字。

G. Collatz Conjecture

看 n 本身是否会进入循环，只需要考虑 $1 \sim B$ 之间与 n 同余的 n'。这个 n' 进循环时，如果在成为 A 的倍数前通过 $+B$ 能到达 n，n 就在循环里，否则 n 不可达。

也即，用 exgcd 求出最小的非负整数 t 满足 $A|(n' + tB)$，然后看是否有 $n \leq n' + tB$ 即可。

时间复杂度为 $O(T \log \min(A, B))$.

H. Staircase Museum

给定一个由二维平面上满足 $1 \le x \le n$ 且 $l_x \le y \le r_x$ 的格子 (x, y) 组成的阶梯形多边形，要在多边形内部和边界上选出最多的点，使得这些点两两之间的连线段都存在一个不在多边形内部和边界的点。

$n \le 5 \times 10^5$.

Author: xyz2606; Prepared by quailty & fstqwq

H. Staircase Museum

首先，选出的点满足二维偏序。

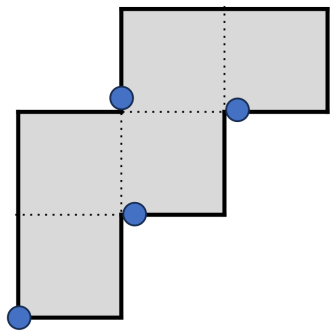
考虑将最右上的点先向左或者向下平移到边界上，再沿着边界移动到靠近拐点的位置。如果这两种平移方式都不合法，说明这两条边界各自能被某个点看到，那么这两个点之间原本就可以互相看到，产生矛盾。

这样将最右上的点尽可能向左下移动之后，不会影响剩下的点向左下移动的过程，直到将最左下的点移动到左下角。

H. Staircase Museum

这样所有点都是以下三种点之一：

- ⌞ 型拐角的拐点，也是阶梯的左下角；
- ⌟ 型拐角的竖直边界上靠近拐角的点，其左侧是多边形外部；
- ⌜ 型拐角的水平边界上靠近拐角的点，其下方是多边形外部；



只需要取出这 $O(n)$ 个点求解二维偏序，也就是按照 x 坐标排序之后计算 y 坐标的最长上升子序列即可，复杂度 $O(n \log n)$。

可以利用点列的特殊性做到 $O(n)$。

I. Color-Balanced Tree

给一棵 $2n$ 个点的树，对每个点青白染色，使得青点和白点都恰好有 n 个，并且对于树上任意一条简单路径，青点和白点个数相差不超过 3，构造一个方案，或判定无解。

$n \leq 2 \times 10^5$.

Author: apiad; Prepared by quailty & fstqwq

I. Color-Balanced Tree

Lemma

存在一个树上匹配，使得所有非匹配点均为叶节点。

Proof.

任选一个节点为根，从根向外进行贪心匹配：如果没有匹配，任选一个子节点进行匹配。无法匹配的点一定是叶节点。 □

I. Color-Balanced Tree

贪心求出一组只有叶子结点不在匹配里的树上匹配。将所有匹配点青白染色使得同色点不相邻，这样一条路径非叶节点的青白点数相差至多为 1，且至多有 2 个叶节点，相差必然不超过 3。每个匹配中恰好是一青一白，剩余的偶数个叶节点任取一半染青，剩余染白即可，因此总是存在解。

由上述分析[3]，亦可直接先对非叶节点青白相间染色，再对叶节点适当染色即可，复杂度 $O(n)$。

[3]或者直接对点数进行归纳：重复进行如下操作直到树被删空：删除所有度数为 1 的点。

J. The Mysterious Shop

$\{1, 2, \ldots, n\}$ 中每个价格的物品以 $1/2$ 概率出现。

一个顾客一开始有 n 块钱，对于所有出现的物品从大到小考虑能买就买。

问最终买到 $0, 1, \ldots, n$ 块钱总计价值的物品的概率。

$n \leq 2 \times 10^5$.

Author: zhoukangyang & Kevin114514;

Prepared by Kevin114514 & fstqwq

J. The Mysterious Shop

设 dp_n 表示 n 的答案。

假设选了 $a_0, a_1, a_2, \ldots, a_k$，其中 a_0 是剩下来的。那么如果 $a_i \le \sum_{j<i} a_j$，那么 S 中必然不能有 (a_i+1, a_i) 中的元素；否则，S 中可以包含 $\left(\sum_{j<i} a_j, a_i\right)$ 中的元素。

先考虑动态规划，记录 dp_m 为 $\sum a_i = m$ 的答案是多少。

J. The Mysterious Shop

如果 $a_i > \sum_{j<i} a_j$，那么可以把状态放在 $\sum_{j<i} a_j$ 上。此时，我们需要做的转移就是：选出一个 $f_0, f_1, \ldots, f_k$ 使得对于所有 $i > 1$，有 $\sum_{j<i} f_j \geq f_i$，且 $\sum f = n$。贡献是 $dp_{f_0} \times 2^{f_1 - f_0 - 1}$。

然后考虑容斥，钦定出一些位置满足 $\sum_{j<i} f_j < f_i$。要做的转移类似，因此可以干掉这个限制。

此时只有限制 $\sum f = n$。这个时候由于元素个数只有 $\sqrt{n}$ 个，可以设 $g(s, k)$ 表示总和是 s，选了 k 个数，每次操作给所有数加一或加新数。

DP 需要转移 $\log n$ 轮，第 i 轮只需要转移 $\sum f \leq \frac{n}{2^{i-1}}$ 的部分，故时间复杂度为 $O(n\sqrt{n})$。

K. Exploration Boundary

在 Dijkstra 运行过程中，把每一次优先队列出队前队内元素所代表的点的集合都拿出来，这些集合被称为探索边界。[4]

给定一张连通无向图，一个 Dijkstra 源点，以及一系列点集。要求构造每条边的权重，使得给定的点集均为从源点出发运行 Dijkstra 的探索边界。

$n, m \leq 2 \times 10^5, \sum |b_i| \leq 10^6$.

Author: fstqwq; Prepared by fstqwq

[4] Universal Optimality of Dijkstra via Beyond-Worst-Case Heaps, https://rihl.uralyx.cz/dijkstra-focs2024/

K. Exploration Boundary

当一个点集成为边界时，整张图被 Dijkstra 算法分为三部分：

- 边界内：已经被访问过的部分，也即当前已知距离的点；
- 边界上：在优先队列里的边界，也即当前已知距离的点的未知邻居的并集；
- 边界外：其余无法被已经访问过的部分的看见的点。

由于 Dijkstra 算法按照最短距离递增的顺序访问每个点，边界上和边界外的点距离不小于边界内的点的距离。基于这个观察，我们可以把边界看做城墙，那么从起点能到的点就应当是边界内的点点集，随后可以验证边界是否正确。这个过程可以通过 BFS 实现。

K. Exploration Boundary

现在考虑多个边界。如果我们把一系列相容的边界所围住的点写出来，一定是一系列互相包含的集合，从小到大的顺序就是他们出现的顺序，且可以通过不停添加边界点来得到这个集合顺序。

直接实现这个求集合的过程是平方的，但是我们事实上并不需要直接求出每个集合。考虑一开始将所有城墙全部建在图上，随后进行 BFS 来尝试访问所有能到的点，并记录阻挡住 BFS 的点（也即 Dijkstra 的优先队列内未出队点），直到 BFS 停下来。此时，阻挡住 BFS 的点应当构成一个边界集合（否则无解），将其拆掉继续 BFS 即可。

注意到，一个顶点上可能有多堵城墙，此时类似拓扑排序的处理入度的方法计数即可。

K. Exploration Boundary

按照上述方法，我们可以得到一个顶点顺序，满足这个出队顺序的边权即可构成所需边界序列。依照这个顺序，我们可以构造势能：第 i 个 BFS 出队的点势能是 i，那么对于在势能为 w_u 和 w_v 之间的边权值设为 $|w_u - w_v|$ 即可保证出队顺序。

由于要构造使得所有点距离互不相同，即使所有所需边界全部出现了，仍需继续运行 BFS 来获得正确的拓扑序，否则可能在最后一个边界外的点出现乱序，导致出现相同的最短距离。

如果实现了线性判断集合是否相同（如集合 hash），时间复杂度为 $O(n + m + \sum |b_i|)$。亦可以直接使用 `std::sort` 或 `std::set`，每次直接寻找给定集合是否出现，时间复杂度为 $O(n + m + \sum |b_i| \log \sum |b_i|)$。

L. Shiori

维护一个整数序列，支持区间赋值、区间将每个数同时加上区间 mex、区间求和。

$n, m, a_i \leq 5 \times 10^5$，保证任意时刻任意数非负。

Author: dXqwq; Prepared by dXqwq

L. Shiori

先考虑没有区间赋值怎么做。

我们认为 $<$ mex 的数“参与了 mex 的计算”，不难发现所有“参与了 mex 的计算”的数在加 mex 后都至少翻倍了。

如果我们能够找出所有 $<$ mex 的数，就可以直接暴力了。开一棵线段树每次找到区间的最小值，然后一旦发现下一个值比这个值大 > 1 就退出，找一个数是 $O(\log n)$ 的，而 mex 不超过 n，倍增次数不超过 $\log n$，总时间复杂度 $O(n \log^2 n)$。

L. Shiori

加上区间赋值之后，最简单的做法就是直接使用平衡树维护连续段再执行上面的算法，复杂度不变。

使用线段树的话，看似每次会更新 $\log n$ 个结点使复杂度到达 $O(n \log^3 n)$，但其实并不会多 log。考虑找最小值时搜索的每一个连续段，在线段树上只会遍历 $O(\log n)$ 个结点，只需要在线段树上 DFS 将所有全局最小值都暂时打标记赋值为无穷大即可。

时间复杂度 $O((n+m) \log^2 n)$。

Thank you!