2024 国际大学生程序设计竞赛亚洲区域赛（成都站）

电子科技大学出题组

2024 年 10 月 27 日

概况

- Easy: J,L
- Easy-Medium: A,B,G,I
- Medium: E,K
- Medium-Hard: C,D,F
- Hard: H,M

L. Recover Statistics

题目大意

- 构造一个数列，使得这个数列的 P50，P95 和 P99 为给定值。其中 Px 的意义为：数列中恰好有 $x\%$ 的数小于等于这个值。
- 关键词：简单构造

构造方法有很多，下面给出一种。

- 构造一个长为 $n = 100$ 的数列，其中有 50 个 a，45 个 b，4 个 c 和 1 个 $c + 1$。这样的数列是满足要求的。

注意题目中给出 Px 定义与 x 分位数不同。x 分位数是序列从小到大排列后位置为序列长度 $x\%$ 的数，但是本题中不是这个定义。

J. Grand Prix of Ballance

题目大意

- 给你一场比赛中会发生的一系列事件，输出该比赛结束后的排行榜
- 关键词：模拟

- 对于每次 2 ，3 操作，我们需要考虑当前操作是否合法，只统计合法的操作。
- 对于每次 1 操作，我们需要清空之前合法操作下记录下来的数据即可。
- 2 个易错点：
  - 每次暴力清除分数和是否完成一张图；
  - 存储分数未开 long long。

A. Arrow a Row

题目大意

- 定义箭串为一个长度大于等于 5 的，前面三个字符和最后一个字符为' $>$ ' 的，其余字符为'-' 的串
- 给定一个长度为 n 的只有' $>$ ' 和'-' 的串，询问它是否可以通过“由一个初始全为 * 的串，用若干长度任意的箭串替换该串一部分”得到，可以就用不超过 n 个箭串做构造。数据范围：$1 \leq n \leq 2 \times 10^5$。
- 关键词：构造

- 根据题目中说的“箭串”的要求，给定字符串能被构造显然需要满足以下三个条件：
  - 最左边一个字符要是“>”
  - 字符串中至少存在一个“-”
  - 最右边的三个字符都要是“>”
- 下面将通过构造的形式证明，满足上述三个条件的字符串可以被构造出来：
  - 首先找到最右边的那个连续“>”段，将箭串的长度固定为 5，右端从给定串最右边往左，固定在每一个“>”位置上，来做绘制操作，从而将这个连续段构造出来。
  - 找到上面所说的这个连续“>”段内，最左边的三个“>”，将接下来所有绘制操作的箭串的右侧固定在这三个“>”的右侧位置上，而左侧固定在最右边的“-”的左边的所有“>”位置上，按箭串从长到短的顺序做绘制操作，从而构造出最右边的连续“>”段以外的所有字符。
- 最终每个询问的复杂度为 $\mathcal{O}(|s|)$。

B. Athlete Welcome Ceremony

题目大意

- 给你一个 abc 和问号构成的串，你要将问号变成 abc 构成合法的串，合法的串是指没有相邻字符相同的串，Q 个询问，每次给你 x,y,z，问你最多用 x 个 a，y 个 b，z 个 c 能构成多少 distinct 的合法串
- 串长 ≤ 500，Q $\leq 10^5$，保证 x + y + z 大于等于问号个数
- 关键词：dp，前缀和

- 令 dp[i][j][k][p = a/b/c] 考虑前 i 个字符，当使用 j 个 a、k 个 b，且最后一个字符是 p 的合法串个数
- 我们考虑如何求得每次询问的答案可以发现，实际上这一步骤可以通过前缀和优化来进行，因此我们可以通过 dp 数组得到 f[x][y][z] 为询问 (x,y,z) 的答案，每次询问 $\mathcal{O}(1)$ 回答即可
- 复杂度 $\mathcal{O}(n^3 + q)$

G. Expanding Array

题目大意

- 给出一个整数数列 $a_1, \ldots, a_n$。可以按如下方式扩展序列：选择两个相邻的整数 a_i 和 a_{i+1}，将 a_i and a_{i+1}，a_i or a_{i+1} 或 $a_i \oplus a_{i+1}$ 三个值中的一个插入到这两个整数之间。你可以扩展无限次。最后统计扩展后的数组中不同元素的个数，问这个个数的最大值。
- 数据范围：$2 \le n \le 10^5$。
- 关键词：二进制

- 由于新插入的整数只能插入到相邻整数之间，因此新插入的整数不会对除这两个整数之外的其他整数产生影响，所以接下来我们对每两个相邻整数分别考虑。
- 考虑原序列中两相邻整数 x, y，利用异或可以将其无限扩展。扩展后如下所示：
  $x, y, x \oplus y, x, y, x \oplus y, x, y, x \oplus y, x, y, \ldots$
- 因此，可以构造出无限个 x 与 y 相邻的情况。
- 对于一对 (x, y) 的第 i 位，只有如下四种情况：
  $x_i = 0, y_i = 0$, $x_i = 0, y_i = 1$, $x_i = 1, y_i = 0$, $x_i = 1, y_i = 1$

- 不妨考虑 $x = 011_{(2)}, y = 101_{(2)}$，考虑 $000_{(2)}$ 到 $111_{(2)}$ 的所有数字是否可以构造出来。容易发现：
  - $000_{(2)} = (x \texttt{ or } y) \oplus (x \texttt{ or } y)$
  - $001_{(2)} = x \texttt{ and } y$
  - $010_{(2)} = x \texttt{ and } (x \oplus y)$
  - $011_{(2)} = x$
  - $100_{(2)} = y \texttt{ and } (x \oplus y)$
  - $101_{(2)} = y$
  - $110_{(2)} = x \oplus y$
  - $111_{(2)} = x \texttt{ or } y$

- 这里我们考虑了对应数位的所有可能情况，并生成了最多种类的整数。考虑 x 和 y 的二进制的第 i 位和第 j 位，如果 $x_i = x_j$ 且 $y_i = y_j$，那么操作后的得到的数 z 的二进制中一定有 $z_i = z_j$。对于如上对应的所有数位在做操作后，得到的数的对应数位一定一样。因此可以推广，对于相邻的两个整数 x, y，可以生成的整数有：
  $x, y, x \texttt{ and } y, x \texttt{ or } y, x \oplus y, x \texttt{ and } (x \oplus y), y \texttt{ and } (x \oplus y), 0$
- 考虑数列中所有相邻整数，将可以生成的数插入一个 set，或生成出这些数排序后去重，统计最后的容器大小即可。
- 时间复杂度：$\mathcal{O}(n \log n)$，空间复杂度：$\mathcal{O}(n)$。

I. Good Partitions

题目大意

- 给定一个长度为 n 的序列，称一个好的划分大小满足：按照给定方式划分的每份序列单调不降。q 次修改，每次修改序列中一个位置的值。询问每次修改前后的好的划分大小的数量。
- $n, q \le 2 \times 10^5$
- 关键词：简单数学，线段树

- 称满足 $a_i > a_{i+1}$ 的下标 i 为断点，显然好的划分大小需要使得 a_i 与 a_{i+1} 在不同的划分序列中，即划分大小是断点 i 的因数。答案即是所有断点的 $\gcd$ 的因数个数。
- 这启示我们可以维护 $\gcd$ 数组 g，对于断点设置 $g_i = i$，非断点设置 $g_i = 0$，线段树维护即可。提前预处理因数个数，可以做到 $\mathcal{O}(n \log n)$。

E. Disrupting Communications

题目大意

- 给定一棵 n 个点的树，有 q 次询问，第 i 次询问给定 u_i, v_i，求树上有多少个连通子图与 u_i 到 v_i 的简单路径至少有一个公共点，结果对 998244353 取模。
- 关键词：dp，lca，前缀和/线段树

预处理:

- 不妨设 1 为树的根。记 u 的父节点为 z ，z 的父节点为 w 。定义 $dp(u,0)$ 表示仅考虑 u 子树内的点，包含点 u 的连通子图数量，$dp(u,1)$ 表示仅考虑 u 子树以外的点，包含点 z 的连通子图数量。状态数为 $O(n)$。转移如下:

$$dp(u,0) = \prod_{v \in son_u} (dp(v,0)+1)$$

$$dp(u,1) = \prod_{(v,z) \in E, v \neq u} (dp(v,[v=w])+1)$$

- 朴素转移时间复杂度与点的度数有关，总复杂度最坏为 $O(n^2)$ 。不能直接求解逆元优化转移，需考虑逆元不存在的情况。数据特殊构造了 dp 值为 998244352 的情况。
- 将某个点的所有相邻点看成序列，可以通过维护序列的前后缀 dp 信息进行优化，同时回避求解逆元，此时总复杂度为 $O(n)$ 。

回答询问：
记询问给定的参数为 u, v 。回答询问有多种方法，大同小异，以下提供两种方法。

- 连通子图与 (u, v) 路径的交是一个连通块（链），点数总是比边数多 1，因此可容斥：包含 (u, v) 路径上某个点的连通子图数量之和减去包含 (u, v) 路径上某条边的连通子图数量之和即为答案。基于前述的预处理和简单的树上维护技巧容易求出。
- 考虑对补集计数，即与路径没有交点的连通子图数量。记 u 和 v 最近公共祖先为 LCA ，则与路径没有交点的连通子图数量等于 $\sum_{i\in subtree_{LCA}, i\notin path_{(u,v)}} dp(i, 0) + dp(LCA, 1)$ 。基于前述的预处理和简单的树上维护技巧容易求出。

K. Magical Set

题目大意

- 给定一个大小为 n（$n \leq 300$）的集合，每次操作可以把一个数字变成它的某个真因数，同时需要保证每次操作后集合内各元素互异，问最多操作次数。
- 关键词：网络流

- 定义初始集合内第 i 个数字为 a_i，考虑经过若干次操作后，最终集合内第 i 个数字变为 a'_i。
- 分别对 a_i、a'_i 进行质因数分解 $a_i = p_0^{k_0} \cdot p_1^{k_1} \cdot \ldots \cdot p_m^{k_m}$，$a'_i = p_0^{k'_0} \cdot p_1^{k'_1} \cdot \ldots \cdot p_m^{k'_m}$，满足 p_j 是质数，且 $k'_j \le k_j$。
- 假设我们已经知道所有 a'_i，贪心地操作，每次选择某个数字，令它的某个大于 k'_j 的 k_j 减一，则答案的最大可能取值为 $\sum_{i=1}^{n}\sum_{j=0}^{m}(k_j - k'_j)$，并且可以证明答案是可以取到最大取值的。操作过程中，如果发现操作后两个数字相同，则可以让另一个数字“代替”自己继续操作下去，以保证操作过程中满足集合内元素互异。
- 基于上述，定义 $f(x)$ 表示将 x 质因数分解后各个质数的指数和，则操作次数可被表示为 $\sum_{i=1}^{n}(f(a_i) - f(a'_i))$；
- 这个式子可以理解为，对每个 a_i，将其与 a'_i 匹配，匹配代价为 $f(a_i) - f(a'_i)$。

- 回到原题，a'_i 未知、互异，且需要最大化匹配代价的和。这个问题可以利用费用流求解。
- 用点集 V_0 代表初始集合中的每个数字，点集 V_1 代表所有可能的因数，建超级源点 S，向点集 V_0 中每个节点连一条流量为 1 的边；建超级汇点 T，点集 V_1 中每个节点向 T 连一条流量为 1 的边；对于点集 V_0 的每个点，向其所表示的初始集合中的数字的所有因数在点集 V_1 中对应的点连一条流量为 1 的边。
- 在这张图上跑最大流，可以满足 a'_i 互异的条件。
- 现在考虑加入费用，对每条从点集 V_0 连向点集 V_1 的边加上 $f(x)-f(y)$ 的费用，其余所有边的费用置 0，在这张图上跑最大费用最大流，得到的最大费用即为答案。

D. Closest Derangement

题目大意

- 给定一个长度为 n 的排列 p 。称满足 $p_i \neq q_i (i = 1, 2, 3, \ldots, n)$ 且 $\sum_{i=1}^{n} |p_i - q_i|$ 取到最小值的排列 q 为最近错位排列。求字典序第 k 小的最近错位排列。
- $2 \leq n \leq 2 \times 10^5$。
- 关键词：RMQ

- 设 $pos(x)$ 表示 x 在 p 中的下标。
- 当 n 为偶数时，$\sum_{i=1}^{n} |p_i - q_i|$ 的最小值为 n，所有的 $|p_i - q_i|$ 都等于 1。此时 q 是唯一的，通过两两配对交换位置得到。
$$q_{pos(x)} = \begin{cases} p_{pos(x)} + 1, x = 1, 3, 5, \dots \\ p_{pos(x)} - 1, x = 2, 4, 6, \dots \end{cases}$$
- 而当 n 为奇数时，由 $\sum_{i=1}^{n} (p_i - q_i) = 0$，可知 $\sum_{i=1}^{n} |p_i - q_i|$ 一定是偶数，故能够取到的最小值为 $n + 1$。
- 即存在一个 x 使得 $|p_{pos(x)} - q_{pos(x)}| = 2$，其他所有位置都满足 $|p_i - q_i| = 1$。
- 考虑最简单的情况：$p = [1, 2, 3]$，此时 q 有 $[2, 3, 1]$ 和 $[3, 1, 2]$ 两种取法。

- 可以观察到如下结论：当 n 为大于 3 的奇数时，q 一定满足这样的形式：
  - 值域上连续的有三个数 $x, x+1, x+2$（$x = 1, 3, 5, 7, \dots, n-2$）按照 $n = 3$ 的两种方法之一，置换位置：
    - $[q_{pox(x)}, q_{pox(x+1)}, q_{pox(x+2)}] = [p_{pox(x+1)}, p_{pox(x+2)}, p_{pox(x)}]$
    - $[q_{pox(x)}, q_{pox(x+1)}, q_{pox(x+2)}] = [p_{pox(x+2)}, p_{pox(x)}, p_{pox(x+1)}]$
  - 值域上剩下的 $1, 2, 3 \dots x-1$ 和 $x+3, x+4, \dots, n$ 两段，长度都为偶数，可以通过两两配对使得 $|p_i - q_i|$ 都等于 1。注意前一段从奇数开始，而后一段是从偶数开始，与 p 的两两配对的方式不同。
- 由于 x 有 $(n-1)/2$ 个，而每个 x 对应两种置换方法，故 q 共有 $n-1$ 种。
- 对于每一个 q，只要知道 $x, x+1, x+2$ 的值就能确定其他所有元素的值，故每一个 q 都可以用它对应的三置换来表示。因此，只要能够找到对两个排列 q 进行 $O(1)$ 字典序比较的方法，就能以 $O(n \log n)$ 的时间对所有的 q 排序，或采用 std::nth_ element $O(n)$ 确定第 k 大。

- 要想比较两个排列的字典序，只需要找到第一个不同的位置即可。
- 对于两个 q, 假设它们对应的三置换分别为 (x_1, x_1+1, x_1+2) 和 (x_2, x_2+1, x_2+2) ，可以发现两个排列中 1 到 $\min(x_1, x_2)-1$ 以及 $\max(x_1+2, x_2+2)+1$ 到 n 这些值所在的位置都是相同的，不需要比较。
- 而位于两个三置换中间的这段区间 ($[\min(x_1+2, x_2+2)+1, \max(x_1, x_2)-1]$，如果有）的每一个值，位置一定不同，因为它们在两个排列中一个位于三置换之前，一个位于三置换之后，与 p 两两配对方式是不同的。只需要找到最靠前的一个位置，可以通过对位置预先构建 ST 表查询区间最小值来得到。因此，两个三置换的位置，以及两个三置换之间的数最靠前的位置，至多 7 个位置之中一定包含了两个排列第一个不同的位置，从按位置小到大依次比较即可。
- 时间复杂度为 $O(n\log n)$，也可采用线性 RMQ 或其他解法做到 $O(n)$。

F. Double 11

题目大意

- n 个商品，每个销量是 s_i，你要把商品分成 M 类，每一类有一个共同的系数 k_j，如果令 c_i 为商品 i 所属的种类，则 $\sum k_{c_i} \times s_i \leq 1$，在此条件下最小化 $\sqrt{\sum_{i=1}^{n} \frac{1}{k_{c_i}}}$，精度误差不超过 10^{-9}
- $n, m \leq 2 \times 10^5$，$1 \leq s_i \leq 10^5$
- 关键词：wqs 二分

- 对答案有影响的是每一个 k_i 分配到多少个数记为 l_i，和这些数的和 s_i。
- 在固定 l_i 和 s_i 时，使用一些数学技巧（如拉格朗日乘数法）可以得到答案为 $\sum \sqrt{l_i s_i}$
- 因此我们就需要最小化上面的式子。继续观察，在固定 l_i 后，类似排序不等式，我们可以得到一定是将 s_i 排序，每个 l_i 对应选择一个区间。
- 此时可以设 $f_{i,j}$ 表示分了 i 段，最后一段结尾是 j 的最小代价，转移枚举最后一段是什么就可以。
- 这个暴力 dp 复杂度为 $O(n^2m)$，考虑优化。发现转移从 $f_{i,l} \to f_{i+1,r}$ 的代价 $w_{l,r} = \sqrt{(r-l)(sum_r - sum_l)}$。并且实际上这个 w 有决策单调性。
- 因此我们可以使用 wqs 二分，用二分栈去转移，复杂度 $O(n \log \epsilon^{-1} \log n)$。

C. Chinese Chess

题目大意

- 交互，给出一个位置集合，在一个中国象棋棋盘上某位置（不必符合象棋原位置，象只能在自己半区，将士只能在九宫格）藏了某种棋子（除了炮），多次询问某个位置，问该棋子到该位置的最小步数，最终编程回答棋子种类（先输出最小询问次数再交互）
- 关键词：构造，决策树

- 首先，我们可以求出对于每一种 6×90 可能的情况（6 种棋子 90 种不同的起始位置），求出其到棋盘上 90 个位置的最短路，（需要 bfs 及一定的模拟）。不难发现除去无法到达的情况最远的情况也不会超过 20 步。
- 回到问题，不难发现最多只有 $6\times 90=540$ 种情况。我们可以通过询问，不断缩小可能情况的集合。使用迭代加深搜索，不难发现即使是棋子可能在棋盘上任意位置的最坏情况，也只需要 3 步就可以保证将范围缩小到只有一种棋子。
- 由于本题还需要我们模拟交互的过程，故而还需要对于每一种情况下的最优策略记录加来。（可以使用手写的线性查找 map 实现，因为 std::bitset 没有重载比较运算）。
- 计算量大约为 $(6\times 90\times 90+90^3\times 20\times \frac{6\times 90}{64})\times C$，其中 C 是一个常数。

H. Friendship is Magic

题目大意

- $f(x)$ 表示把 x 的十进制拆成两个子串后最小的差的绝对值，求 $\sum f(i) \mod 10^9+7, L \le i \le R, 1 \le L, R \le 10^{18}$
- 关键词：数学，分类讨论

- 本题可能有多种不同做法，这里仅给出一个较直接的思路。
- 考虑拆分一个至少三位的数，通常可以写成 AxB 的形式，其中 A 和 B 都可以有若干位，而 x 是一个 0 到 9 之间的数，使得 (Ax)>B，A<(xB)。
- 显然，最优拆分下的差应该是 $\min(10A + x - B, x \times 10^{len(B)} + B - A)$。这里我们以 $10A + x - B < x \times 10^{len(B)} + B - A$ 为例，那么一组合法的 A, x, B 所需要满足的条件可以写成：
    - $B < 10A + B$
    - $B > A - x \times 10^{len(B)}$
    - $B > \frac{11A+(1-10^{len(B)})x}{2}$

- 考虑 A 为横坐标，B 为纵坐标的平面，这是平面中的三条直线。可以考虑枚举 A、B 的位数和 x 的值，需要解决的问题相当于给定平面上的一个矩形，要求统计矩形内部、并且满足三条直线限制的所有点的个数，横坐标和，纵坐标和。
- 可以注意到所有斜率都是 0.5 的倍数，因此这个问题不太难解决。可以通过分类讨论，将答案用若干等差数列求和以及等差数列各项平方求和来求出。
- 值得注意的是，其实只有两条在矩形内不相交的直线是实际具有限制作用的，因此满足限制的点其实是被两条直线夹在中间的区域，这可以一定程度简化计算。
- 对于 $10A + x - B > x \times 10^{len(B)} + B - A$ 的情况，同理。
- 时间复杂度可粗略估计为 $O(T \times 18^2 \times 10)$，具有较大常数，取决于具体实现细节。

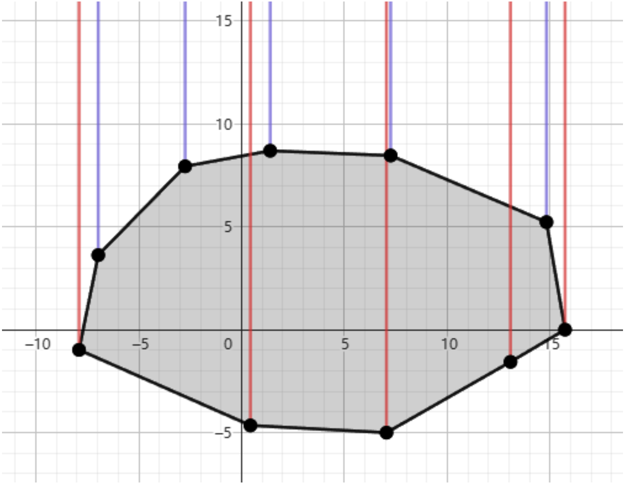
M. Two Convex Holes

题目大意

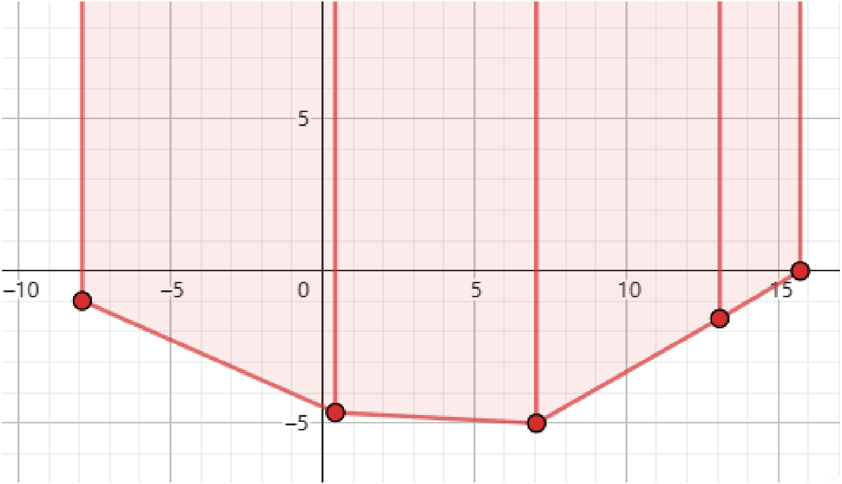
- 在一个三维空间 xOy 平面上有一个无限大的光屏，光屏上方有个垂直于 z 轴的无限大不透光平面，这两个平面上分别有一个透光的凸多边形孔。现在在这两个平面的上方有一个点光源沿着某个垂直于 z 轴的方向匀速直线运动（速度不为 0）。已知两个凸多边形孔和光源 $t = 0$ 时刻的坐标以及光源的速度，问在 $[t_1, t_2]$ 区间内随机等概率挑选一个时刻观测 xOy 平面，问这一平面上有光区域的期望时多少。
- 关键词：计算几何，积分

- 首先，由于光源是垂直于 z 轴移动的，所以问题其实是伪三维。只需将两个凸多变形孔投射到 xOy 平面，再做仿射变换（一些细节可以看后面的算法流程），即可转换为如下问题——
- 给你两个凸多边形 P_1 和 P_2（$|P_1| = n_1$，$|P_2| = n_2$），P_1 相对于 P_2 以 $\boldsymbol{v}$ 的速度匀速直线运动，求 $[t_1, t_2]$ 时间内两个多边形的面积交的平均值。
- 注意到在仿射变换过程中可以对坐标系进行放缩旋转，从而将 $\boldsymbol{v}$ 转换到一个相对容易处理的方向。在接下来的讨论中，我们默认 $\boldsymbol{v} = (0, -1)$，即沿 y 轴负方向每秒一个单位移动。

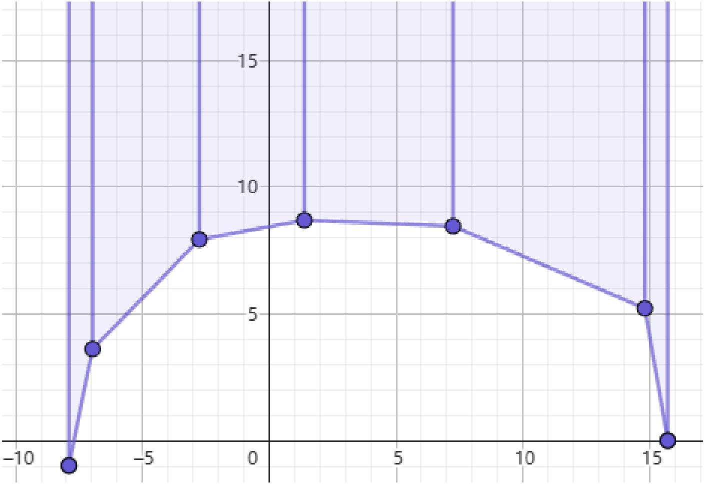
对于一个多边形，考虑拆分其面积。如对于某个 P_1



可以被拆解为四个红色且无限向 y 轴正方向延申的区域

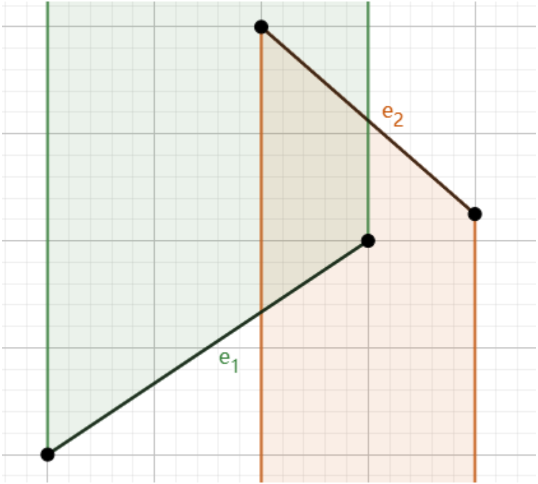


减去六个紫色的区域得到。



对于 P_2 我们也可以进行类似的拆解，只是区域的延申方向相反。

对应到多边形交，我们考虑对于边拆开计算其贡献。具体而言，对于每一条在 P_1 上的边 e_1 和 P_2 的上的边 e_2，每一对 (e_1, e_2) 的贡献即为如下图所示的两个凸区域的交集面积：



- 在经过上述转化之后，剩下的求两个区域（其中一个区域匀速直线运动）在任意时刻的交集的部分就很 trivial 了。在求面积交前可以先将永远不会成为交集的部分裁剪掉，这样可以避免一些如边平行于运动方向之类的细节，不过仍需根据 e_1 和 e_2 是否平行进行分类讨论。
- 最后算法实现的大致流程如下——
- 进行仿射变换，转化为二维问题
  - 将整个空间沿着 $(-x_0, -y_0)$ 方向平移，即将光源移动到 z 轴上
  - 计算出两个多边形孔在 xOy 平面上的投影，并计算出投影运动的相对速度 $\boldsymbol{v}' = (v'_x, v'_y)$
  - 【可选】将 xOy 上的所有点变换到以 $(-v'_y, v'_x)$ 为 x 轴方向的单位向量，原点不变的新坐标系上
- 将（变换后的）P_1 和 P_2 拆分，计算 $n_1 \times n_2$ 对边 (e_1, e_2) 的贡献（得到的应该是一个两段或三段的函数）

- 将所有的分段函数的加起来得到面积交关于时刻 t 的函数 $S(t)$，此外由于题目中所求的是期望，故还需要求出 $R(t) = \int_{-\infty}^{t} S(t)\,\mathrm{d}t$。这一过程中需要对函数分段点进行排序
- 对于询问 $[l_i, r_i]$，若 $l_i = r_i$ 则答案为 $S(l_i)$，否则为 $\frac{R(r_i)-R(l_i)}{r_i-l_i}$。显然 $S(\cdot)$ 和 $R(\cdot)$ 也是分段函数，故而这一步可以用二分实现
- 如果在做仿射变换时放缩了坐标系，那么答案还需乘以一个系数（通常是 $||\mathbf{v}'||^2$）
- 时间复杂度 $((n_1 + n_2 + q)\log(n_1 + n_2))$。
- 这一算法有如下特点：
  - 不需要写半平面交
  - 所有操作均可在四则运算范围内完成（故答案必为有理数）

Thank you!