2020牛客暑期多校训练营（第九场）

比赛链接

A -

Solved by .

题目描述

解题思路

C - Groundhog and Gaming Time

Upsolved by nikkukun.

题目描述

给定 $n \leq 5 \times 10^5$ 个区间 $[l_i, r_i]$，每个区间都会独立地以 $\dfrac 12$ 的概率被选中，求所有被选中的区间的交集长度的平方和的期望。

注意这里 $[L, R]$ 的长度是按 $R - L + 1$ 算，出题人不会好好说话，被坑了。

解题思路

令所有区间端点在数轴的投影将数轴分割成了长度为 $s_1, s_2, \ldots, s_m$ 的 m 个连续的线段，记 $p_i \in \{0, 1\}$ 表示其中的线段 i 是否在结果的交集中，则题目求的是

$$ \begin{aligned} E \left[\left(\sum_{i=1}^m s_i p_i \right)^2 \right] &= E \left[\sum_{i=1}^m (s_i p_i)^2 + 2\sum_{i \neq j} s_i p_i \cdot s_j p_j \right] \\ &= \sum_{i=1}^m E \left(s_i^2 \cdot p_i \right) + \sum_{i \neq j} E \left(s_i s_j \cdot p_i p_j \right) \end{aligned} $$

对于前面那部分，如果有 k 个原来的线段覆盖了 s_i，那么在所有 2^n 种情况里，只有恰好 $2^k - 1$ 种区间选择方法可以让 $p_i = 1$ 有贡献 $\dfrac {2^k - 1}{2^n} \cdot s_i^2$，否则为 0。这部分可以直接 $O(n)$ 枚举计算。

后面的部分类似，假设有 k 个线段恰好能覆盖掉 s_i 和 s_j，那么只有恰好 $2^k - 1$ 种区间选择方法可以让 $p_i = p_j = 1$ 有贡献 $\dfrac {2^k - 1}{2^n} \cdot s_i s_j$，否则为 0。

为了不暴力枚举 i, j 计算后面的部分，考虑从小到大枚举 j，一次计算 $i < j$ 的所有 i 的贡献。假设我们有一个线段树维护了 $i \in [1, j)$ 之间，既覆盖了 s_j 又覆盖了 s_i 的原线段个数 k_i，那么也可以同时维护 $\dfrac {2^{k_i} - 1}{2^n} \cdot s_i s_j$ 之和，单独把那个 1 提出来后就能维护。每次从 j 移动到 $j+1$ 时，更新 s_{j+1} 导致的区间 k 的增减即可计算答案。

总时间复杂度 $O(n \log n)$。

D - Groundhog Chasing Death

Solved by nikkukun.

题目描述

计算

$$ \prod _{i=a}^b \prod _{i=c}^d \gcd(x^i, y^j) \bmod 998244353 $$

其中 $0 \leq a, b, c, d \leq 3 \times 10^6$，$1 \leq x, y \leq 10^9$。

解题思路

为了方便表述，记 $A = \max\{a, b, c, d\},\ B = \max\{x, y\}$。

考虑每个质因子 p 对指数的贡献，并令 s, t 为使得 $p^s \mid x$ 与 $p^t \mid y$ 成立的最大整数，则贡献为 $$ \sum _{i=a}^b \sum _{i=c}^d \min(si, tj) = \sum _{u=1}^{\infty} \left(\sum _{i=a}^b [u \leq si]\right) \left(\sum _{i=c}^d [u \leq tj]\right) $$

这一部分暴力枚举 u 可以做到 $O(A \log B)$，而由于使得 $s, t > 0$ 的质数 p 只有 $O(\log B)$ 个，故对每个有贡献的质数都计算一次的总时间复杂度为 $O(A \log^2 B)$。

另外计算时用容斥把式子拆成四组前缀和的加加减减会好写很多。

K - The Flee Plan of Groundhog

Solved by nikkukun.

题目描述

给一个 10^5 结点的树，A 在 u 且速度是 1，B 在 $v = n$ 且速度是 2。现在 B 抓 A，A 可以到处逃，问最久可以逃多久。

解题思路

暴力做，枚举终点 x，计算 A 和 B 跑过去会不会在中途相遇（满足 $2 \cdot \mathrm{dist}(u, x) \leq \mathrm{dist}(v, x)$ 就不相遇），顺便计算所需时间即可。

赛后总结

nikkukun

1. 做题之前一定要先按题意把样例算一下，避免读错题或者是写错算法的情况发生；
2. 如果某个算法渐进意义明显超时了，那就不要加优化多次提交了，考虑能不能把复杂度降低一个等级；
3. 数量级很大的时候，不要用 `map<int, vector<int>`，会卡爆。

qxforever

Potassium