NOI模拟赛

SDSZ

2018 年 6 月 21 日

1 疑惑

1.1 问题描述

有N个非负整数，求随机选$N-x$个数的异或的期望是多少。$0 \le x \le N-1$，答案对998244353取模。

1.2 问题求解

我们发现期望就是所有方案的和除以方案的方案数，而方案数就是组合数C_n^k。所以以下算法中我们都是求所有方案的和。

1.2.1 $N \le 20$

对于不同的x，暴力枚举数的选法，求出它们的异或和，时间复杂度$O(2^N N)$，可以获得15分。

1.2.2 $N \le 1000$ ，$A_i \le 31$ 和$N \le 300$ ，$A_i \le 1023$

我们考虑对问题进行dp，令$f_{i,j,k}$表示前i个数，选了j个，异或和为k的方案数。转移就是枚举第i个数选或者不选，即$f_{i,j,k} = f_{i-1,j-1,k \oplus a_i} + f_{i-1,j,k}$，时间复杂度$O(N^2 max\{a_i\})$，可以获得30分。

1.2.3 所有a_i均相同

可以发现异或出来的数不是0就是a_i，所以选奇数个时所有方案数是$C_n^k \cdot a_i$，偶数个时是0，时间复杂度$O(N)$，结合前两个算法可以获得35 分。

1.2.4 $a_i \le 1$

我们发现此时我们只有0和1，而只有选择奇数个1才对答案有贡献。记1的个数为cnt，当我们要选k个数时，对答案有贡献的方案数（即选奇数个1的方案数）为：$\sum_{i=1,i=2t+1}^{cnt} C_{cnt}^{i} \cdot C_{N-cnt}^{k-i}$，对于这个式子，我们构造多项式$A(x) = \frac{1-(-1)^x}{2} \cdot C_{cnt}^x$，多项式$B(x) = C_{N-cnt}^x$，将两个多项式卷积起来就是最终要的不同项的答案啊。时间复杂度$O(NlogN)$，结合前三个算法可以获得45分。

1.2.5 $N \le 2000$

可以发现每一位是独立的，所有对每一位，单独去跑算法4，若你没有发现是卷积，直接暴力的复杂度$O(N^2 log(max\{a_i\}))$，可以获得40分。

1.2.6 $N \leq 100000$

对于每一位使用NTT即可获得满分，时间复杂度$O(NlogNlog(max\{a_i\}))$，可以获得100分。

2 MonkeySort

本题的数据比较有梯度，但是有一些部分分(~~是瞎出的~~)出题人并不会。

2.1 问题描述

给你一个排列，每次随机交换两个不同位置的数，问你得到的最终序列逆序对的期望对数是多少。

2.2 问题求解

2.2.1 算法1

对于$k = 0$的点，用传统的求逆序对的方法（归并，树状数组）求解。时间复杂度$O(nlogn)$

2.2.2 算法2

枚举k次MonkeySwap的结果，把每种结果的逆序对数相加。时间复杂度$O(C_n^k nlogn)$

2.2.3 算法3

考虑计算每一对$(a_i, a_j) i < j$的贡献，令$f_{kij0/1}$表示交换k轮后，两个关键点分别在i和j且关键点间的大小关系是0/1的方案数。

直接对于所有$(i, j)\ i < j$,将$f_{kij(key1>key2)}$相加即为答案。

初始状态可以通过初始排列a的每一对元素的大小关系得到。

例如$a_1 = 2$, $a_3 = 1$,则有初始状态$f_{0,1,3,1} = 1$和状态$f_{0,3,1,0} = 1$。

不用上述方法，而暴力枚举两个关键点再暴力转移，复杂度$O(kn^5)$

转移的时候枚举交换的方式，复杂度$O(kn^3)$

若使用前缀和优化转移，复杂度$O(kn^2)$

2.2.4 算法4

考虑两个关键点A, B以及它们的初始位置i, $j(i < j)$。它们的末状态可以简化成7种。

状态描述	编号
(1,2): (A in pos i)(B in pos j)	0
(1,0): (A in pos i)(B not in pos j)	1
(0, 2): (A not in pos i)(B in pos j)	2
(2, 1): (A in pos j)(B in pos i)	3
(2, 0): (A in pos j)(B in pos i)	4
(0, 1): (A not in pos j)(B in pos i)	5
(0, 0): (A in pos (i or j))(B not in pos(i or j))	6

将两个点的末状态拆分成这7种状态后，每一种状态的贡献单独统计并相加即为答案。

通过这样的拆分，我们并不需要枚举关键点，因为对于任意点对(i, j)，到达这些状态的方案数是相同的，可以先用$g_{k,7}$的dp将到达这7种末状态的方案数计算出来。显然这一步可以矩阵优化。

之后，考虑状态$0 - 5$，由于至少有一个点的位置是确定的，我们可以较容易的统计出每个点作为位置确定的那个关键点对这些状态的贡献。

例如，当前点为i, $[0, i\text{–}1]$中有k_1个数小于a_i, 有k_2个数大于a_i，那么点i对于状态2的贡献为：$(k_1 \cdot \frac{n-i}{n-2} + k_2 \cdot \frac{i-2}{n-2}) \cdot g_{k,2}$

即：令当前点i为关键点B，则关键点A可取$[0, i - 1]$，考虑关键点A的末位置，若末位置在i即关键点B前面，则有$i - 2$个位置可选（不能是初始位置），且A有贡献当且仅当$A > B$。若末位置在B后面，则有$n - i$个位置可选，且要求$A < B$。而对于状态6，我们发现每一对点恰有$\frac{1}{2}$的概率有贡献，把到状态6的方案数除2即可。可以用树状数组维护出k_1并计算相关值。复杂度$O(logk + nlogn)$

3 路灯

3.1 问题描述

二维平面上有N个点，按照1到n的顺序依次点亮每一个点，每个点都有一个相同的照亮范围，由两个向量组成。当一个点被照亮k_i次时，它也就被点亮了，问每个点被照亮的时刻。

3.2 问题求解

首先对于它所照亮的两个向量(x_1, y_1),(x_2, y_2)，若它们不共线（叉积不等于0），考虑这样一个变换：若原坐标为(x, y)，令

$x' = \frac{x \cdot y_2 - y \cdot x_2}{x_1 \cdot y_2 - x_2 \cdot y_1}$

$y' = \frac{x \cdot y_1 - y \cdot x_1}{x_2 \cdot y_1 - x_1 \cdot y_2}$

其实就是换基向量，这样我们就可以认为光线是水平和竖直的了。

若两向量共线，我们可以将向量无限放大，最后稍微让其中一个向量偏离一点，这样这两个向量就不共线了。

我们发现上面式子中的分母是一样的，于是我们就可以将分母去掉（但注意符号），只剩一个整数了。

3.2.1 算法1

我们可以暴力模拟题目的过程，对于第i个灯，若它还没亮，就把它点亮。然后去更新其他灯，时间复杂度$O(N^2)$

3.2.2 算法2

尝试底层优化（松松松），时间复杂度$O(N^2)$

3.2.3 算法3

考虑将所有点按照y坐标排序，相同的按照x轴排序，依次插入。显然一下式子成立:

$ans_i = min(i, ans_p)$，其中p为i左下角的点中ans第k_i小的数，特别的，若没有第k_i小的数，则认为第二项无穷大。

考虑树套树，在外层建线段树，每个线段树节点建一棵权值线段树。考虑在$O(logn)$棵权值线段树上二分，即可得知答案，时间复杂度$O(nlog^2n)$，空间复杂度$(nlog^2n)$

若是在外层建立权值线段树，里层建立平衡树，就可以将空间复杂度降到$O(nlogn)$

3.2.4 算法4

是不是觉得空间$64MB$很不爽，下面我来介绍一种空间$O(n)$的做法。考虑整体二分。对于在区间$[l, r]$内的所有点，先将它们按照y轴排序，维护一棵树状数组。若查询一个点的x前缀和比k_i小，我们将k_i减去前缀和，并将其放到右区间中。否则将其放在左区间中，并在树状数组对应位置+1。一直分治直到$l = r$，时间复杂度$O(nlog^2n)$，空间复杂度$O(n)$

4 祝愿

祝大家NOI能取得理想的成绩!