「比赛总结」实验舱

Jiayi Su (ShuYuMo)

2021-03-10 11:40:46

第一天的题都是好题啊 QAQ!!

异或

给出一个大小为 n 的集合 S 和一个整数 x ，求 S 中有多少子集满足其中任意两个元素之间的异或和不小于 x 。对质数取模。$n \leq 10^6$

设 x 的二进制最高位为 k 。

考虑两个数字异或的情况，如果两个数字大于 k 的二进制位不一样，其异或结果一定不小于 x 。

那么如果我们把大于 k 的二进制位相同的数取出来，分成一个组，然后单独求出每组内能取出的子集个数，最后答案只需要对每个组能取出的子集个数自由组合即可。

考虑求出每组内能取出的子集个数。因为要求每组取出的子集中任意两个数的异或和所以这些数字在第 k 位的取值一定要两两不同，所以每一组内取出的合法子集大小只有 $0, 1, 2$ 三种，前两种是平凡的，最后一种用 trie 维护即可。

计数

给定整数 x ，定义长度为 N 的序列 $\{a_i\}$ 的价值 $f(a)$ 如下：

$$\max\{L(a) - x, 0\}$$

其中 $L(a)$ 表示 a 最长的相同子段的长度。

给定整数 N, K, x ，求出长度为 N，各个元素在 $[1, K]$ 中的所有整数序列的价值之和。大质数取模。$N \leq 10^6, K \leq 10^8, x \geq 0$

首先一个显然的策略是枚举每个 $L(a)$ 的可能取值 x，算方案数即可。

每次算方案数时可以设 $dp[n][0/1]$ 考虑了长度为 n 的序列，达到了 / 未达到 x 的方案数。

这样就已经没办法优化了，可以考虑把等于的限制改宽改为大于的限制，注意到：

$$\max\{L(a) - x, 0\} = \sum_{i=x+1}^{n} [L[a] \geq i]$$

类似地贡献转化手法还有这道题。可能是一种常见套路

问题转化为对于一个 i 求有多少序列的连续子段长度大于 i。

可以考虑补集转化，求有多少序列的连续子段长度小于 i 的序列，只需要在每个位置枚举从这里出发连续字段有多长就可以了。(相当于一个存在性限制，转化为任意性限制，任意性限制显然结构更加简单)

可以设 $dp[n]$ 表示长度为 n 的序列，最长相同子段的长度小于 x 的方案数，转移显然：

$$dp[n]=\begin{cases}k^n & n<x\\ (k-1)\sum\limits_{i=1}^{x-1}dp[n-i] & otherwise.\end{cases}$$

可以用前缀和优化。这样做复杂度仍旧是 $\mathcal{O}(n^2)$ 的。

题解中给出了这样一份代码，其 $dp[n]$ 的意义是长度为 n 的合法序列，且最后两个元素不同的方案数：

```
int getans(int n, int k, int x) {
    if (x == 1) return power(k, n);
    static int dp[MAXN]; dp[1] = k;
    for (int i = 2; i <= n; i++)
        if (i < x) dp[i] = 1ll * k * dp[i - 1] % P;
        else dp[i] = (1ll * k * dp[i - 1] - 1ll * (k - 1) * dp[i - x] % P + P) % P;
    return (0ll + power(k, n) - dp[n] + P + dp[n - x + 1]) % P;
}
```

按照题解的转移方式，其方程可以写作：(转移可以理解为把最后一个数字扩充，然后随便放上一个不一样的)

$$dp[n]=\begin{cases}k & n=1\\ dp[n-1]\times k & i<x\\ dp[n-1]\times k+(1-k)\times dp[i-x] & otherwise.\end{cases}$$

考虑到这里的方程转移比较单一，可以试图画出转移图，然后考虑贡献的传递，第一个点值为 k ，每一条转移边都是乘某一个值，且这个值的取值仅有 2 种。

如果一条从 1 到 n 的转移路径所经过的两种转移路径数量相同，那么他们的贡献也相同。枚举第二条转移路径的数量即可计算贡献。这样做的复杂度是 $\mathcal{O}(\frac{n}{x})$ 。根据调和级数的理论，这样的做法是 $\mathcal{O}(n\log n)$ 的。

有一道平衡树优化 dp 转移的题目，比较有意思。

A

B

C. 最大价值

A 君有 n 个物品，每个物品有两个属性 a_i,b_i，现在 A 君想从所有物品中挑选 k 个，并按一定顺序摆放好。

对于一个方案中，被摆在第 j 个位置（位置从 1 开始标号）的物品为 i，它对这个方案产生的价值贡献为 $a_i\cdot(j-1)+b_i$，一个方案的价值和为它所含的所有物品的贡献之和。

现在 A 君想知道对于所有可能的 k（$1\le k\le n$ ），在最优的选取以及摆放情况下能得到的方案价值和最大是多少。

接下来 n 行，每行两个整数 a_i,b_i。

100% 的数据：$n\le 300\,000$, $0\le a_i\le 10^6$, $0\le b_i\le 10^{12}$

输出 n 行，每行两个整数 a_i,b_i ，表示 $k=1,2,3,4,\cdots,n$ 时的最大总价值。

对于一个大小为 k 的方案，如果确定了选择哪 k 个，就可以直接按照 a 排序，然后依次算出贡献（排序不等式）

那么显然可以考虑先对原物品按照的 a 排序，然后依次做背包即可。

$$dp[i][j]=\max\{dp[i-1][j],dp[i-1][j-1]+a_i\times j+b_i\}$$

观察到一条结论，选定 k 个物品的答案，一定是在原来 $k-1$ 个物品的答案基础上添加一个物品，而不会拿走。

如果考虑了 i 种物品，那么对于选取 j 个物品的方案，一定是从选 $j-1$ 个物品的方案添加一个物品得到的。也就是说，每新考虑一个物品，上面的转移一定是在某个 j 之前都是继承上一轮的答案，之后都是新增这个物品的答案，维护差分数组即可。可以平衡数维护每一个 j 处的答案的差分。

官方题解附有结论证明。