2020牛客暑期多校训练营（第六场）

李泽仁

牛客竞赛

AC.NOWCODER.COM

A. African sort

- **题意**
- 给定排列 p，每次可以选一个下标集合等概率打乱包含的数并花费集合大小的代价，求给 p 排升序最优策略下最小代价的期望，对 998244353 取模

A. African sort

令 i 向 p_i 连边得到由若干环构成的图，把操作集合大小花费之和转化成每个点被集合包含的期望次数之和，于是我们可以将每个点所在环的大小作为期望花费的参考。

首先思考恰好有一个大小为 n 的环（$n>1$）的最优策略。

> 引理：打乱大小为 n 的环，任一点将等概率出现在大小 $1\sim n$ 的环里。

考虑全排列，比如 1 所在环的大小只与排列中 1 的位置有关。

这似乎提示我们打乱整个环是最优的。如果操作集合仅为环中的一部分，一方面若仍保持 n 的花费，同时操作环的补集，点整体出现在更大环的概率高于前者，另一方面若多花费至少 1 的代价而去考虑下一步策略，后面我们会证明大小为 n 的环中任一点的期望被包含次数比 $n-1$ 仅多 $\frac{1}{n-1}$，小于多花费的代价。

A. African sort

如果有多个环，显然最优策略是独立地操作每个环。

设 f_n 为大小为 n 的环中任一点的期望被包含次数。

> 结论：$f_n = [n > 1] + H(n-1)$。

考虑归纳证明，$n = 1$ 显然成立，对于 $n > 1$ 根据引理有：

$$
\begin{aligned}
f_n &= 1 + \frac{1}{n} f_n + \frac{1}{n} \sum_{i=1}^{n-1} f_i = \frac{n}{n-1}(1 + \frac{1}{n} \sum_{i=1}^{n-1} f_i) = \frac{n}{n-1}(1 + \frac{n-2}{n} + \frac{1}{n} \sum_{i=1}^{n-1} \sum_{j=1}^{i-1} \frac{1}{j}) \\
&= \frac{n}{n-1}(1 + \frac{n-2}{n} + \frac{1}{n} \sum_{j=1}^{n-1} \frac{n-1-j}{j}) = \frac{n}{n-1}(1 + \frac{n-2}{n} + \frac{n-1}{n} H(n-1) - \frac{n-1}{n}) \\
&= H(n-1) + 1
\end{aligned}
$$

$O(n)$ 预处理 f，$O(mn)$ 找环求解即可。

B. Binary Vector

- **题意**
- 随机n个n维01向量，询问这个n个向量线性无关的概率

B. Binary Vector

由于这N个向量线性无关，则这N个向量张成的空间秩为N，考虑将每次随机的向量加入之前向量的空间

，那么最后N个向量秩为N当且仅当每次加入的向量都不属于之前的空间。

那么概率为$\prod_{i=0}^{N-1}\frac{2^N-2^i}{2^N}=\frac{\prod_{i=0}^{N-1}(2^N-2^i)}{2^{N^2}}=f_N$

容易发现$f_N=\frac{2f_{N-1}+(2^N-1)}{2^{2N-1}}$,所以可以线性求出所有$f_N$，时间复杂度为$O(n)$

牛客竞赛 AC.NOWCODER.COM

C. Combination of Physics and Maths

- **题意**
- 一个矩阵的底面积定义为最后一行的数的和，重量定义为所有数的和，给一个正整数矩阵，找一个“压强”最大的可非连续子矩阵

C. Combination of Physics and Maths

首先，若选择的子矩阵有多列，拆成两个行数不变的更小子矩阵，其中一个一定不劣。

证明是容易的，对于 $\frac{a}{b} \le \frac{c}{d}\ (a,b,c,d>0)$，必然有 $\frac{a}{b} \le \frac{a+c}{b+d} \le \frac{c}{d}$。

既然选择的子矩阵只有一列，不妨枚举唯一的底面积元素，选入其上方所有元素是最优的。不难做到 $O(Tnm)$。

D. Data Structure

- **题意**
- 给定一棵 n个节点的有根树，第 i 个点的编号是 i。
- 有m次询问，每次询问给出 l,r,x，求有多少点编号的二元组 (i,j) 满足 l <= i < j <= r 且 i 和 j 的最近公共祖先是节点 x。

D. Data Structure

这个题有矩阵乘法相关规约，难以polylog解决，考虑根号算法。

以下复杂度分析中 n 和 m 都写做 n。

这个询问转化一下，就是查询 x 子树在 $[l,r]$ 内的点个数平方，减去 x 的每个儿子的子树在 $[l,r]$ 内的点个数平方。

前者平凡，考虑后者：

这个是一个点所有儿子子树信息的查询，我们可以对原树轻重链剖分，用莫队维护每个点所有轻儿子的答案，这样每次修改沿着重链向上面跳，只会影响 $O(logn)$ 个位置，然后再求重儿子的答案，总时间复杂度 $O(n\sqrt{n}logn)$。

D. Data Structure

考虑优化这个的复杂度，我们对每个点，令重儿子为子树大小前 $O(\sqrt{n})$ 大的儿子，也就是说莫队只维护除了前 $O(\sqrt{n})$ 大的儿子的信息。

这样我们每次跳一次轻边，子树大小会增大 $O(\sqrt{n})$ 倍，于是只会跳 $O(1)$ 次轻边，莫队部分复杂度 $O(n\sqrt{n})$。

重儿子即查询 $O(n\sqrt{n})$ 次DFS序在一个区间中，小于一个数的元素个数，这个差分后用 $O(\sqrt{n})$修改，$O(1)$ 查询的分块即可 $O(n\sqrt{n})$ 解决掉。

可以用一些技巧优化到线性空间。

总时间复杂度 $O(n\sqrt{n})$，总空间复杂度 $O(n)$。

E. Easy construction

- **题意**
- 给定 n,k，问是否可以构造一个 1~n 的排列 P，使得对于 1~n 中任意的数 i，P 都存在一个长为 i 的子区间，其和模 n 余 k。有解输出任意一组，无解输出 -1。

E. Easy construction

- 首先如果要有解，k 必须是 n(n+1)/2 % n，因为长度为 n 的子区间只有一个，也就是 P 本身，而 P 本身的元素和就是 n(n+1)/2。

- 然后假设 k 满足上述条件，此时是一定有解的。如果 n 是奇数，可知 k=0，令 P = {n, 1, n-1, 2, n-2, ...} 即可；如果 n是偶数，可知 k=n/2，令 P = {n, n/2, 1, n-1, 2, n-2, ...} 即可。

F. Fibonacci Partition

- **题意**
- 对于给定的正整数n，令partition（n）为m的最大值，以使n可以表示为m个不同的斐波那契数之和。
- 每次把x加ai*Fbi，对于每次操作输出partition(x)

F. Fibonacci Partition

考虑X的 *Zenkorf representation*，如果有了这个就可以轻松计算出答案。假设X的*Zenkorf representation*中 1 出现的位置$p_1, p_2, \ldots, p_n$ $(p_i \geq 2)$，那么显然每个p_i都尽可能的拆成其他Fibonacci数之和是最好的。也就是说，答案几乎为:

$$\sum_{i=2}^{n} \lceil \frac{p_i - p_{i-1}}{2} \rceil$$

可能还需要关注p_1的值是不是2或者3。如果用Treap维护这个序列p，上式可以轻松维护。于是现在变成如何快速维护X的 *Zenkorf representation*。

F. Fibonacci Partition

把F的定义稍微改一下，$F_0 = 0, F_1 = 1, F_n = F_{n-1} + F_{n-2}$，并推广到负数下标上去，显然有$F_{-n} = (-1)^{n+1}F_n$。

考虑Lucas number，$L_0 = 2, L_1 = 1, L_n = L_{n-1} + L_{n-2}$，有个经典的等式:

$$L_k \cdot F_n = F_{n+k} + (-1)^k F_{n-k}$$

如果我们把a拆成若干个Lucas number的和，就可以用上面的式子变成加上或者减去若干个Fibonacci数，这就比较方便维护了。

F. Fibonacci Partition

经过一些观察发现，加入或者减去一个F_x的话，只和X的*Zenkorf representation*，那些 `10101...10101` 有关，因此可以用一个Treap来维护这些 `10101..101` 段。考虑加法的话，有以下几种情况：

1. F_{x-1}, F_x, F_{x+1}都不存在，那么直接加上一个F_x即可。
2. F_{x-1}, F_x, F_{x+1}至少有一个存在，那么假设包含x的 `101..101` 段中最高位1和最低位1分别是l和r，还可以分成下面几种情况：
   - $x = r + 1$，那么需要把F_r删掉，加入F_{r+2}。
   - $l \leq x \leq r$并且$x \equiv l \bmod 2$，那么需要把$F_l, F_{l+2}, \ldots, F_r$都删掉，加入$F_{r+1}$，$F_{l+1}, F_{l+3}, \ldots, F_{x-1}$和$F_{l-2}$。
   - $l \leq x \leq r$并且$x \not\equiv l \bmod 2$，那么需要把$F_{x+1}, F_{x+3}, \ldots, F_r$都删掉，加入$F_{r+1}$。

注意到在加入F_{r+2}或者F_{l-2}的时候可能会出现两个相邻的 `1`，需要递归重复以上过程。显然，递归有限次后就会结束。

考虑减法的话，也是类似的分析后可以搞定。

因此可以$O(\log A)$时间维护一次加入或者删除后的 *Zenkorf representation*。整体可以在$O(n \log^2 A)$的时间搞定这个题。

牛客竞赛 AC.NOWCODER.COM

G. Grid Coloring

- **题意**
- 给n*n的网格图的边染k种色，每种色染的边数相同，构造不存在同色环及整行整列不同色的方案

牛客竞赛
AC.NOWCODER.COM

G. Grid Coloring

$n=1$，$k=1$，$k \nmid 2(n+1)n$ 显然无解，其他情况可以构造出解。

若 k 为偶数且不为 2，横边竖边各分配一半的颜色。找到 $n+1$ 的约数 x 和 n 的约数 y 满足 $xy=k/2$，将 $(n+1)\times n$ 条横边或竖边划分为 $x\times y$ 块，每个块中染一种颜色。然而结果可能不满足限制 3，我们不妨把颜色矩阵的第 i 行交错，再循环移 $i-1$ 位。（如果 $x=1$，把调整行改为调整列）

$$\begin{bmatrix}1&1&1&1&2&2&2&2\\1&1&1&1&2&2&2&2\\1&1&1&1&2&2&2&2\\3&3&3&3&4&4&4&4\\3&3&3&3&4&4&4&4\\3&3&3&3&4&4&4&4\\5&5&5&5&6&6&6&6\\5&5&5&5&6&6&6&6\\5&5&5&5&6&6&6&6\end{bmatrix}\rightarrow\begin{bmatrix}1&2&1&2&1&2&1&2\\1&2&1&2&1&2&1&2\\1&2&1&2&1&2&1&2\\3&4&3&4&3&4&3&4\\3&4&3&4&3&4&3&4\\3&4&3&4&3&4&3&4\\5&6&5&6&5&6&5&6\\5&6&5&6&5&6&5&6\\5&6&5&6&5&6&5&6\end{bmatrix}\rightarrow\begin{bmatrix}1&2&1&2&1&2&1&2\\2&1&2&1&2&1&2&1\\1&2&1&2&1&2&1&2\\4&3&4&3&4&3&4&3\\3&4&3&4&3&4&3&4\\4&3&4&3&4&3&4&3\\5&6&5&6&5&6&5&6\\6&5&6&5&6&5&6&5\\5&6&5&6&5&6&5&6\end{bmatrix}$$

若 k 为计数或等于 2，横边竖边将共用所有的颜色。同样找到 $n+1$ 的约数 x 和 n 的约数 y 满足 $xy=k$，划分染色调整。既然两种边共用颜色，会存在同色环而不满足限制 2 吗？实际上对两种边输出同一个调整后的颜色矩阵是满足的，因为矩阵中任两个相邻的项不同，而任意同色环至少会导致一对相同的相邻项，反过来即成立。

H. Harmony Pairs

- **题意**
- 求1<=A<=B<=N,满足S(A)>S(B)的(A,B)个数 s是数码和

H. Harmony Pairs

考虑数位dp，$f(i,\Delta,f_0,f_1)$表示已经确定了A,B的最高的i位，两数的数码差是Δ,A与N的大小关系是f_0，A与B的大小关系是f_1的方案数，转移暴力枚举A,B下一位的数码即可。

时间复杂度$O(l^2d^3)$，其中l是N的长度，$d=10$

I. Interesting Stirling Task

- **题意**
- 求第一类斯特林数n行l到r项和%p

$$\left(\sum_{k=l}^{r} \left[{n \atop k} \right]\right) \bmod p$$

I. Interesting Stirling Task

显然可以拆成两个$(\sum_{k=0}^{m} \left[{n \atop k}\right]) \bmod p$的差。

考虑Lucas定理的证明

$$\binom{n'p+n_0}{k'p+k_0} \equiv \binom{n'}{k'}\binom{n_0}{k_0} \pmod p$$

根据$(x+1)^{n'p+n_0} = (x+1)^{n'p}(x+1)^{n_0}$和$(x+1)^p \equiv= x^p + 1 \pmod p$，然后二项式展开即可得到Lucas定理。

我们知道$s_n(x) = x(x+1)(x+2)\dots(x+n-1)$的系数就是$\left[{n \atop 0}\right], \left[{n \atop 1}\right], \dots, \left[{n \atop n}\right]$。

首先可以发现$s_p(x) \equiv x^p - x \pmod p$。

然后，令$n = n'p + n_0$，那么有

$$\begin{aligned} s_n(x) &= \prod_{t=0}^{n'-1}(x+tp)(x+tp+1)\dots(x+tp+p-1) \cdot \prod_{r=0}^{n_0-1}(x+n'p+r) \\ &\equiv \prod_{t=0}^{n'-1} x(x+1)\dots(x+p-1) \cdot \prod_{r=0}^{n_0-1}(x+r) \equiv s_p^{n'}(x) s_{n_0}(x) \\ &\equiv (x^p - x)^{n'} s_{n_0}(x) \pmod p \end{aligned}$$

配合二项式展开，我们可以得到

$$\left[{n \atop k}\right] \equiv (-1)^(n'-j)\binom{n'}{j}\left[{n_0 \atop i}\right] \pmod p$$

其中$k = n' + j(p-1) + i$，$(0 \le i < p-1 \text{ if } n_0 = 0$或者$0 < i \le p-1 \text{ if } n_0 > 0)$。

I. Interesting Stirling Task

可以发现，我们枚举i之后，仅需要求

Math </>

$$\sum_{j=0}^{\lfloor \frac{m-i-n'}{p-1} \rfloor} (-1)^{n'-j}\binom{n'}{j}$$

本质上就是给出n，m和x，求$\sum_{i=0}^{m}\binom{n}{i}x^i$。

注意到x^n也是可以和Lucas定理一样拆开的：$x^n \equiv x^{\lfloor \frac{n}{p} \rfloor} \cdot x^{n \bmod p} \pmod p$。因此，可以将$x^i$和$\binom{n}{i}$一起用Lucas定理拆开。

稍微化简下就可以得到如下的式子：

$$\sum_{i=0}^{m}\binom{n}{i}x^i = \sum_{t=0}^{\lfloor \frac{m}{p} \rfloor -1}\binom{\lfloor \frac{n}{p} \rfloor}{t}x^t \cdot \sum_{r=0}^{p-1}\binom{n \bmod p}{r}x^r + \binom{n/p}{m/p} \cdot x^{m/p} \cdot \sum_{r=0}^{m \bmod p}\binom{n \bmod p}{r}x^r$$

由于n是固定的，总共只有$O(\log n)$种$n \bmod p$，都预处理一个$\binom{n \bmod p}{r}x^r$的前缀和即可。然后再预处理出$s_{n_0}(x)$，就可以解决这个题了。

预处理$s_{n_0}(x)$可以用分治的方法做到$O(n_0 \log n_0)$。

总体来说，时间复杂度为$O(p \log p)$差不多。

J. Josephus Transform

- **题意**
- 给一个长度为 n 的排列和 m 次操作，每个操作可以表示为 (k, x)，即进行 x 次以 k-约瑟夫变换。问最后排列长啥样。

J. Josephus Transform

- 首先考虑如何求一个 k-约瑟夫变换。注意到每次取出来的数字是剩下所有数字的第几个其实是可以算出来的。不妨设上一个被取出来的数字是当时的第 pos 个（初始设为 1），当前还剩下 cnt 个数字，那么下一个被选出来的数应该是当前剩下的所有数字中的第 (pos-1+k-1) % cnt + 1 个，这个可以用线段树或者树状数组来求，所以求一个k-约瑟夫变换的复杂度是 O(n log n) 的。
- 然后要连续进行 x 次置换，注意到置换是满足结合律的，所以可以用快速幂来算，这部分时间复杂度就是 O(n log x) 的。
- 所以总的时间复杂度是 O(nm(log n + log x)) 的。

K. K-bag

- **题意**
- K-bag序列定义为由多个1-k的排列顺序连接起来的序列
- 想问你给定序列是不是k-bag的连续子序列

K. K-bag

对于一个部分 k-bag 序列，一定存在一个 x 满足：对于任意整数 y 都有 $[yk+x, yk+x+k-1]$ 这一段区间里所有的数都不相等（这个区间可能不被 $[1,n]$ 完全包含，此时指它和 $[1,n]$ 的交），并且可以限制 x 取 0 到 $k-1$ 之间的某个整数。容易发现只要上面的条件也是充要的。定义 pre_i 表示最大的满足 $a_p=a_i, p<i$ 的 p，如果不存在则未定义。对于一个有定义的 pre_i，对 x 取值的限制为：存在一个值 j 满足 $j \bmod k = x$ 且 $pre_i \le j \le i$，由此合法的 x 将是某一段或某两段区间。具体来说，当 $pre_i \le i-k$ 时对 x 没有限制；否则当 $pre_i \bmod k < i \bmod k$ 时要求 $pre_i \bmod k \le x < i \bmod k$；当 $pre_i \bmod k > i \bmod k$ 时要求 $pre_i \bmod k \le x < k$ 或 $0 \le x < i \bmod k$。对于这些限制求交或者求补集的并即可判断是否存在一个合法的 x，复杂度 $O(n)$ 或 $O(n\log n)$。

Thanks

牛客竞赛
AC.NOWCODER.COM