Solution

环游世界(travel)

假设我们得到了 a 数组，做一个前缀和，这 m 次旅行，每一次就是对前缀和数组的两个数的异或和提出了要求。

我们将这种要求抽象到图上的一条边，显然，在图中，任意一个数确定了，那么他所在的联通块内所有的数都确定了。

判无解很简单，随便赋一个值看看冲不冲突即可。

需要注意的是，$Sum_0 = 0$，所以 0 所在的联通块方案唯一。有解情况下，答案为 $(2^k)^{cnt-1}$，cnt 为联通块个数。

加固(reinforce)

考虑怎么写暴力，显然，我们需要枚举出现的所有字母的排列，在这个排列下，如果某一个字母的字典序比它前一个字母的字典序小，那么它和它前一个字母必然不在同一段咒语中。

那么最小的答案必然就是这样的相邻字母对数+1。显然存在一种方案答案可以取到。

接下来我们考虑正解，不妨考虑从小到大确认字典序。我们记 f_S 表示 S 中的字典序已经被确认，且 $|S|$ 中的字典序就是字典序第 1 到第 S 的字符。这种情况下，已经可以确定的相邻的不在同一段咒语的最小点对数量。

转移的时候直接枚举字典序第 $|S|+1$ 的字符即可。预处理统计所有相邻的字符对 $cost_{i,j}$ 表示 i 比 j 字典序大时产生的贡献。我们可以预处理 $cost_{i,S}=\sum_{j\in S} cost_{i,j}$。

这样预处理可以做到 $O(20\times 2^{20})$，DP 复杂度也是 $O(20\times 2^{20})$。

最终的复杂度就是 $O(|\sum|\times 2^{|\sum|})$,其中 $|\sum|$ 为字符集大小，本题为20。

图（graph）

首先考虑树的情况。

定义某个点的权值 val_u,即其所有儿子的子树大小以及当前整个子树的大小的异或和。这样删去某条路径的答案就变成了路径上所有点的异或和，再异或上 val_{LCA} 和 $n - sz_{LCA}$即可。

现在放到图上，删掉两点间的所有简单路径上的点相当于是删除两点之间的所有点双。建出圆方树，相当于是删除路径上的所有方点及其周围的所有圆点。

定义圆方树的子树大小为这个子树内的圆点个数。

类似地定义圆点的权值为其所有儿子方点子树大小的异或和。方点的权值为所有儿子圆点的权值的异或和。这样，方点的权值就是删去这个点双之后，子图内所有联通块大小的异或和。

再类似地给每个方点异或上当前子树内的圆点个数，令 sum 为某两点 u, v 路径上所有方点的权值异或和，c 为最近公共祖先。

如果 c 是圆点，那么这个圆点相邻的方点中 u, v 方向上的子节点的子树大小在 sum 中被重复计算了一次，那么 $sum \oplus val_c$ 刚刚好消除了重复计算的那两个子树大小的影响，同时异或上了删除这个圆点之后，除了 u, v 方向的子节点的答案。这个时候还剩下 c 之外的子树的答案没有算，答案就是 $ans = sum \oplus val_c \oplus (n - sz_c)$ 。

如果 c 是方点，要注意它的父亲也会被删除，当前方点被多计算了一次，$sum \oplus val_{fa_c}$ 同样地消除影响加统计其余子树答案。剩下的是 c 的父亲之外的子树的答案，

$ans = sum \oplus val_{fa_c} \oplus (n - sz_{fa_c})$。

用树上差分来维护路径权值异或和，Tarjan 构建点双，圆方树, 离线求 Lca 可以做到 $\mathcal{O}(n + m)$。

排列变换(permutation)

首先分析一下运算，我们有 $(p+q)_{p_i} = q_i$。写成置换形式就是：

$$p+q = \begin{pmatrix} p_1 & p_2 & \cdots & p_n \\ q_1 & q_2 & \cdots & q_n \end{pmatrix}$$

考虑左乘一个 p，那么有：

$$p(p+q) = \begin{pmatrix} 1 & 2 & \cdots & n \\ p_1 & p_2 & \cdots & p_n \end{pmatrix} + \begin{pmatrix} p_1 & p_2 & \cdots & p_n \\ q_1 & q_2 & \cdots & q_n \end{pmatrix} = q$$

这意味着：$p(p+q) = q$，即 $p^{-1}p(p+q) = p^{-1}q$，于是 $p+q = p^{-1}q$。

然后：

$$\begin{aligned} &f_0 = p \\ &f_1 = q \\ &f_2 = p^{-1}q \\ &f_3 = q^{-1}p^{-1}q \\ &f_4 = q^{-1}\underline{pq^{-1}p^{-1}q} \\ &f_5 = q^{-1}pqq^{-1}pq^{-1}p^{-1}q = q^{-1}p\underline{pq^{-1}p^{-1}q} \end{aligned}$$

注意到后面的下划线部分持续出现，我们令 $A = pq^{-1}p^{-1}q$，有：

$$\begin{aligned} &f_4 = q^{-1}A \\ &f_5 = q^{-1}pA \\ &f_6 = A^{-1}qq^{-1}pA = A^{-1}pA \\ &f_7 = A^{-1}p^{-1}qA^{-1}pA = A^{-1}qA \end{aligned}$$

容易发现 $f_n = A^{-1}f_{n-6}A$。

于是可以通过类似快速幂的倍增思想解决本题。

骰子(dice)

如果获知了一轮游戏中平局的概率 draw，则在 R 局内获胜的概率为 $\frac{1-\mathrm{draw}^R}{2}$。

问题转化为获知平局的概率 draw。

将每个骰子上的点数减 1，即点数范围为 $[0,K)\cap\mathbb{Z}$。

则平局概率为 $\frac{1}{K^{2N}}\sum_{i=0}^{N(K-1)}\mathrm{f}[N][i]^2$，其中 $\mathrm{f}[n][m]$ 表示投掷 n 个 k 面骰子，点数和恰好为 m 的方案数。

有转移方程 $\mathrm{f}[n][m]=\sum_{i=0}^{K-1}f[n-1][m-i]$，边界为 $\mathrm{f}[0][0]=1$。

使用前缀和优化到 $\mathcal{O}(N^2K)$ 即可获得 45 分。

接下来给出一个结论：$\sum_i\mathrm{f}[n][i]^2=\mathrm{f}[2n][n(K-1)]$。

这个结论的证明：

- 首先 $\mathrm{f}[2n][k]=\sum_{i+j=k}\mathrm{f}[n][i]\cdot\mathrm{f}[n][j]$。
- 所以有 $\sum_i\mathrm{f}[n][i]\cdot\mathrm{f}[n][n(K-1)-i]=\mathrm{f}[2n][n(K-1)]$。
- 但是因为 $\mathrm{f}[n][i]=\mathrm{f}[n][n(K-1)-i]$（每个骰子取的点数从原来的 x 变成 $K-1-x$ 即可）。
- 所以有 $\sum_i\mathrm{f}[n][i]^2=\mathrm{f}[2n][n(K-1)]$。

那么只需计算 $\frac{1}{K^{2N}}\mathrm{f}[2N][N(K-1)]$ 即可。

$\mathrm{f}[n][m]$ 相当于 m 个无标号物品放入 n 个有标号盒子中，但每个盒子不能装大于等于 K 个物品的放置方案数。

我们知道，没有限制时的答案为 $\binom{m+n-1}{n-1}$。

考虑容斥：至少有 i 个盒子超出了限制，则答案为 $\sum_{i=0}^{n}(-1)^i\binom{n}{i}\binom{m-K\cdot i+n-1}{n-1}$。

代回原式得到：$\mathrm{draw}=\frac{1}{K^{2N}}\sum_i(-1)^i\binom{2N}{i}\binom{K(N-i)+N-1}{2N-1}$。

$\mathcal{O}(N\cdot K)$ 预处理阶乘以及阶乘的逆元即可在 $\mathcal{O}(N)$ 的时间内计算答案。

注意两个 $5\cdot 10^7$ 的 `int` 数组是开不进 256MB 的，但是可以发现有用的阶乘也就 $\mathcal{O}(N)$ 个，可以开若干个 10^7 的数组进行存储，以避免 MLE。