trappola bewitching

答案是树的直径长度除以 2。

证明：

> 方便起见，下面将该值称为“半径”，记作 R。
>
> 首先容易证明答案不小于半径：对于任意和为 1 的非负实数序列 x，设直径两端点分别为 v_1, v_2，则有：
>
> $$\begin{aligned}&\sum_{i=1}^{n} x_i \operatorname{dis}(i, v_1) + \sum_{i=1}^{n} x_i \operatorname{dis}(i, v_2)\\ =&\sum_{i=1}^{n} x_i (\operatorname{dis}(i, v_1) + \operatorname{dis}(i, v_2))\\ \geq&\sum_{i=1}^{n} x_i \operatorname{dis}(v_1, v_2)\\ =&\operatorname{dis}(v_1, v_2)\end{aligned}$$
>
> 于是 v_1, v_2 中的一个必满足
>
> $$\sum_{i=1}^{n} x_i \operatorname{dis}(i, v) \geq \frac{\operatorname{dis}(v_1, v_2)}{2} = R$$
>
> 其次，如果存在点 c 使得 c 到 v_1, v_2 的距离均为半径，则使 $x_c = 1$，其余项为 0 即可取到下界。
>
> 若不存在，设直径上点 p, q 满足：
>
> - p, q 之间有连边，设该边边权为 w_0。
> - $\operatorname{dis}(p, v_1) < \operatorname{dis}(p, v_2)$。
> - $\operatorname{dis}(q, v_1) > \operatorname{dis}(q, v_2)$。
>
> 使 $x_p = \dfrac{w_0 - (R - \operatorname{dis}(p, v_1))}{w_0}$，$x_q = 1 - x_p$，其余项均为 0 即可取到下界。

conflict

设 $f_{u,s}$ 为选出一个以 u 为根，大小为 s 的连通块的方案数。设 $g_{u,s}$ 为选出一个包含 u 的父亲，不包含 u，大小为 s 的连通块的方案数。（也就是对普通的树上背包 dp 进行换根）

直接这样做，转移 g 的复杂度是 $O(n^3)$ 的。

注意到答案是 $\sum f_{u,s} \times g_{u,s}$，也就是只有 $f_{u,s}$ 有值的位置的 $g_{u,s}$ 才有用。因此可以将 $g_{u,s}$ 的长度强行截断到 u 的子树大小，然后暴力转移。这样可以沿用树上背包的复杂度分析，为 $O(n^2)$。

Axium Crisis

一个点集的 LCA 等于其中 dfs 序最小的点和 dfs 序最大的点的 LCA，于是考虑在线段树上维护 dfs 序。

对树重链剖分得到一个 dfs 序，对给定的 m 个点建线段树。线段树上维护：

- dfs 序最小值
- dfs 序最大值

- 区间内所有点距离最靠上的轻边的距离的最小值（记为 d_m）

执行 1 操作时，如果 $d_m = 0$，则向两侧递归暴力修改。如果 $d_m > 0$，则直接将整个区间的三个值都打上一个 -1 的标记。这样每个点只有在移动到一条轻边下面的时候会被暴力修改一次，复杂度即为 $O(n + m \log n \log m + q \log m)$。

另外一种方法是线段树上维护区间点的 LCA 和区间深度最小值，每次相当于对后者区间 -1，然后如果后者的值小于前者的深度则将前者向上移动一位。这样的复杂度是 $O(n + m \log n + q \log m)$。注意不要在线段树合并孩子信息的时候更新 LCA，否则会求 $O(q \log m)$ 次 LCA，可能会由于常数原因无法通过。

Grievous Lady

原题 Luogu P8946 ~~虽然也是我出的。~~

考虑暴力 DP：

设 $f_{i,j}$ 表示填到前 i 位，运算结果是 j 的方案数，结果大于 m 统一记为 $m+1$（因为再接一位一定是 0）。朴素实现可得 24，精细实现可得 40。（事实上 40 分很提示正解）

考虑所有转移。对于 A：

1. 所有数都可以往后接一个更小的数转移到 0。
2. 1 可以接任意数转移到对应的数。
3. 0 接任意数转移到 1。
4. 其余转移不多。

对于 C：

5. 所有数都可以往后接一个更小的数转移到 0。
6. 所有数都可以往后接自己转移到 1。
7. 所有数都可以往后接自己 +1 转移到自己 +1。
8. 1 可以接任意数转移到该数。
9. 其余转移不多。

这里会有几个重复转移（例如 $1\text{C}1 = 1$ 会被多次转移），需要减掉。重复部分很少所以可以直接归到最后一类里面。

依次考虑每一种转移：

1. 全局 $\sum a_i(i-1)$，单点修改
2. 全局加
3. 单点修改，单点求值
4. 单点修改，单点求值
5. 同 1
6. 全局和，单点修改
7. 整体向右平移一位
8. 全局加
9. 同 3

所以数据结构需要支持：

- $O(1)$ 单点修改；
- $O(1)$ 单点求值；
- $O(\sqrt{m})$ 以下全局加；
- $O(\sqrt{m})$ 以下全局和；
- $O(\sqrt{m})$ 以下全局 $\sum a_i(i-1)$；

- $O(\sqrt{m})$ 以下向右平移一位。

注意这个转移的过程中前一列的 DP 不能继承到后一列，所以还要支持：

- $O(\sqrt{m})$ 以下全部清零。

使用一个 $O(1)$ 数据结构维护：保存 DP 数组 f 和一个时间标记数组 t，$s_1=\sum_{i=0}^{m} f_i$，$s_2=\sum_{i=0}^{m} f_i(i-1)$，以及一个清零值 v_0，清零时间 t_0，整体加的值 v_A。考虑所有操作：

- 单点修改：这里是单点加，下放标记（如果 $t_i<t_0$，将时间修改到 t_0，f_i 修改到 v_0）之后直接改数组并对应更新 s_1 和 s_2 就行，$O(1)$。
- 单点求值：下放标记后返回 f_i+v_A 即可，$O(1)$。
- 全局加：修改 v_A, s_1, s_2 即可，$O(1)$。
- 全局和：返回 s_1 即可，$O(1)$。
- 全局 $\sum a_i(i-1)$：返回 s_2+f_0 即可（注意 s_2 中 f_0 的系数是 -1），$O(1)$。
- 向右平移一位：修改 f, t 的起始指针的位置（可以开一个两倍长的数组，初始把指针指在中间，这样每次直接自减 1 即可），并对应修改 s_1 和 s_2 即可，$O(1)$。
- 清零：将 t_0 设为当前时间，v_0 设为 $-v_A$，s_1 和 s_2 均设为 0 即可，$O(1)$。

最后一个运算符特殊处理，需要求组合数上指标前缀和，是经典问题，可以 $O(1)$ 求。

所以这题就做完了。最后复杂度是 $O(n\sum_{i=2}^{\sqrt{m}}\sqrt[i]{i!m})$，实际 $m=10^5$ 时 A/C 转移分别只有 $393/1195$ 个，可以通过。