HNOI 2018 省队集训
Day1 Solution

June 20, 2018

1 Tree

几个月前CF Div2的一道题，拿来作签到题。

如果没有换根操作，那么这就是一道裸题：只需要用线段树/树状数组维护DFS上的权值和即可。需要支持换根操作时，我们也并不需要真的换个根，而只需要用一个变量记录当前的根$root$。

经过一些简单的讨论可以知道，u和v在以$root$为根时的lca为：$lca(u,v)$, $lca(u,root)$, $lca(v,root)$中原来深度最大的一个，其中$lca(a,b)$代表a,b 在以1为根时的lca。根据$root$是否在原来u的子树内，我们可以轻易提取出以$root$为根时u的子树对应的区间。

于是这还是一道裸题，$O(q\log n)$.

2 Function

我们重新描述一下这道题：一个$10^9\times n$的网格，每个格子有一个权值，每一列格子的权值都是相同的。从一个起点开始，每次可以向上走一格或者向左上角走一格，直到走到最上面一行为止，你需要最小化经过的格子的总权值。

首先我们可以发现一些显然的性质，最优的路径之一一定形如：先往左上走若干步（可能不走），到达权值较小的一列后，一直往上走到顶。对于每个询问，枚举从起点出发最终会到达哪一列，就可以得到一个$O(nq)$的做法。

对于任意$1\le i\le j\le n$, 从(x,j)出发最终到达第i列然后走到顶的代价，可以表示为一个关于x的一次函数，我们只关心这些一次函数的最小值，也就是这些直线形成的下凸壳。我们得到一个思路：将询问离线，按y从小到大排序，从最左边开始每次加入一条直线，维护下凸壳，然后在凸壳上二分即可得到答案。

怎么维护下凸壳呢？对于一个点(x,y)，它要么继承上一列$x-1$的决策，要么就直接往上走到顶。并且我们发现，第二种情况只会出现在从顶端开始连续的一段中。于是我们只需要用栈维护凸壳即可。

$O((n+q)\log n)$.

3 Or

考了才知道是毛爷爷论文里的题，直接送分了...

一个合法的A序列满足这样的条件：对于任意$i \in [1,n]$，A_i中一定存在某一位为1，而之前的A_i中这一位都为0。

记$dp[i][j]$表示：前i个数，有j位出现过1时的方案数，显然有转移：

$$dp[i][j] = \sum_{k=1}^{j} dp[i-1][j-k] \times 2^{j-k} \times \binom{j}{k}$$

写出这个再加个FFT就能做到$O(nk\log k)$，将上面的式子写成生成函数的形式(F_i表示$dp[i]$的指数型生成函数)：

$$\frac{dp[i][j]}{j!} = \sum_{k=1}^{j} \frac{dp[i-1][j-k] \times 2^{j-k}}{(j-k)!} \times \frac{1}{k!}$$

$$F_i(x) = F_{i-1}(2x) \times (e^x - 1)$$

$$F_n(x) = \prod_{i=0}^{n-1} e^{2^i x} - 1$$

最后一个式子可以倍增求，假设n是偶数，设

$$G(x) = \prod_{i=0}^{\frac{n}{2}-1} e^{2^i x} - 1$$

那么

$$F_n(x) = G(x) \times G(2^{\frac{n}{2}} x)$$

.

于是就只要log次多项式乘法了，$O(k \log n \log k)$.