Prime Distance On Tree题解

雅礼中学 毛啸

2015 年 10 月 8 日

、

题意

给定一棵树，问从中选两个点他们之间距离是质数的概率。

树的大小不超过50000。

提示

这个题目表面上是问概率，实际上我们只要求出距离是质数的点对个数然后除以$\binom{n}{2}$即可。

题解

一个朴素的算法是，暴力枚举两个点，然后用线性时间求出他们直接的距离，再判断是不是质数从而决定是否更新答案。

判断是不是质数可以用筛法预处理，数据范围较小也可以直接枚举数然后暴力。

求距离的话，可以以每个点为根BFS，也可以用递推解决。

上述算法都难以通过本题。

经过一些思考之后我们发现，质数这个条件，套在距离上，是没有任何作用的。唯一的办法就是摘掉质数这个帽子，看清题目的真谛。

其实这个题目是要求我们求两两距离为任意值的点对个数。

很多要求这个值的题目，都有一个通用的解法，本题也不例外，那就是——树分治。

树分治的过程是选一个点为根，处理所有经过根的路径，然后递归每个子树继续处理。

处理经过根的路径时，复杂度可以是整棵树的大小相关的复杂度。

复杂度会是怎样的呢，假设每一次的复杂度是树的大小，如果树是一条链而我们每次都是选链的一个端点，那么复杂度是$O(n^2)$，不能忍受。

而如果我们每次选中点，根据主定理，复杂度只有$O(n\log n)$。

因此这个点的选取是十分重要的。

我们发现，只要取树的重心，就可以保证复杂度是$O(n\log n)$，因为重心的最大子树大小不超过$\lfloor\frac{n}{2}\rfloor$，可以继续使用主定理。

选定重心之后，如何统计经过他的路径呢。

一个朴素的方法是对于他的每一个子树，求出深度为每一个值的点个数，如果在这棵子树之前的子树中(根可以算作第一棵子树)，深度为i 的子树个数是a_i，这棵子树中深度为i的子树个数是b_i，那么将给长度为$i+j$的路径条数贡献a_ib_j。最后我们将每个b_i加到对应的a_i中去即可。

这样复杂度是每次分治都要$O(n^2)$，显然无法忍受。

我们发现，给长度为$i+j$的路径条数贡献a_ib_j，其实就是序列的卷积。FFT是OI中常用的计算卷积的算法，本题也不例外。

由于$c_k = \sum_i \sum_j [i+j = k]a_ib_j$，对于一个大于两个序列长度之和的n, $i+j = (i+j) \mod n$，所以$c_k = \sum_i \sum_j [(i+j) \mod n = k]a_ib_j$也就是$c_k = \sum_i \sum_j [(i+j-k) \mod n = 0]a_ib_j$，而$[a \mod b = 0]$可以拆成单位复根的形式，最后我们可以化成一个反演变换之后乘起来再逆变换回去的过程，而这就是FFT的思路，FFT的介绍有很多，这里不再赘述。

我们直接将上述卷积改成FFT，即可通过CC的数据。

当我写完上述做法时，马上提交获得的结果却是WA，而对拍并没有找出我的错误，后来我发现我是用了NTT(数论变换)来做的卷积，模数设的10^9左右的数，而考虑一个点连着另外$n-1$个点的数，长度为2的路径个数是超过了模数的，所以我的算法会得到错误的答案。

当我造完清橙上要用的输入之后运行我自己的程序获得输出时，却发现我的程序卡在了一个点一直无法运行出答案，我开始怀疑是不是我数据的问题。后来事实证明是我程序的漏洞，CC的数据是不强的。

事实上，做卷积的时候，如果我们先处理一棵最大深度较大的子树，然后处理很多棵深度很小的子树，那么，由于我们记录的a数组是一个前缀和，而FFT的复杂度是和两个序列的长度和相关的，所以我的算法会执行很多次复杂度非常高的FFT，于是无法在时限内得出答案。

解决方法很简单，只要按子树最大深度从小到大处理即可，我们发现虽然是前缀和，但是序列长度只是取max，所以序列和不超过当前子树最大深度的两倍，因此很容易证明每次分治的复杂度为$O(n \log n)$，根据主定理总复杂度为$O(n \log^2 n)$。

官方题解

CC 上的题解