1000 pt

我觉得这个题挺难。

首先能想到的是这是一个树上的DP。于是要先把树建出来，由于这种输入第一次见。所以建树也花了点时间。建树的算法如下：

BFS，任选一个起始结点s，标记为访问，然后扫A,B数组，找到他的邻居结点，如果没有被访问过，则将其加入到s的孩子列表中，并相应的入队。因为没有环，所以每个结点都只访问一次，标记是为了不要访问父亲结点。

然后是递归，考虑以i为根的子树的放置k个狐狸的方案，分两种情况：

1. i没有fox，那么i只可能有一个子树放置k个狐狸，这样得到了子问题。
2. I上有fox，那么这些子树的根首先必须放置狐狸，然后这些子树放置的狐狸个数之和等于k-1。

但是这样不好DP，首先这个方法需要考虑子树内部是否把fox放在根部。所以是否放在根部需要外部传入，用s来标记，所以把子问题定为：根i已经确定(为状态s)的情况下，子树（不含i）里面放k个狐狸的方案有多少个？

这样可以DP，但是需要枚举x1+x2+…xn=k的所有可能。

题解里有一个比较好的方案，可以通过状态s来同时记录需要考虑的子树从最左边哪个开始。

于是，可以只考虑最左边的子节点ch[s]，他的所有可能，乘上这种可能下，最左为s+1的以i为根的子树的所有放置方案。

所以，用s区分后，原问题变为：

1. S=0时，i上没有fox，就可以对每个子树单独考虑，加和即可。
2. S>0时，表明最左边的结点是ch[s]，枚举这个结点的可能值，和余下的fox个数。乘上该条件下在下一个最左结点的以i为根的子树的放置情况数。

然后需要考虑边界情况，边界情况一直是我处理不好的地方。

F(p , s , n)

有三个参数，边界分别是

1. P是叶子结点
2. S>当前结点的儿子个数
3. N=0

P是子节点时，讨论s和n的情况，是返回1还是0.

S不合法时，讨论p和n的情况。

N不合法时，讨论p和s的情况。

有更好的做法么？