Solution

T1：

如果是树上的情况很容易能发现答案就是，因为确定根之后儿子节点的颜色只与父亲节点有关。

如果是一个环的情况也可以类似地先确定一个点为根然后推导：如果不考虑首尾颜色相同的情况答案为，然后减去首尾相同的情况、加上三个点相同的情况……

上述过程，也即容斥可得答案为。观察得其乘上即为。

所以不同大小的环对答案的贡献不一样，而最多只会有种不同大小的环，所以复杂度为

T2：

容易看出一个点到另一个点的期望步数只与两点之间的距离有关，而无关两个点的位置。所以我们设为距离终点为的点有多少个。

设为距离终点小于等于的点的距离和为多少。

容易看出存在一个阈值，使得距离大于的点都随机跳跃更优，而其它点都沿最短路最优。

设大于的点的期望为，大于的点有 个

则满足： ，解得

并且

根据以上条件有可二分性，所以二分出合适的然后用点分治查询就行了。

时间复杂度

T3：先考虑Ax/B的情况 设A/B=k 作射线y=kx(k>0) 从原点出发 每次和y=int相交记一个1遇到x=int+记一个0 先不管过整点的问题

那么这样一个01string 第x个0之前有y个1 就是全部的pairs(x,y) 如果能解构 那么就容易了

对比k=13/8的串A和k=5/8的串B 相当于B中每个0->10 1->1形成A 对比k=5/8和k=8/5则是01互换

反过来就是相当于从k=1/1的串开始 每次将0和1分别替换成一些东西 做O(log)次 得到最终串

这里有个过整点的细节问题 根据整除定义的话我们最终应该先1后0 所以如果01互换要做偶数次 初始串则一致是10 否则应该是01

其实到这部分和一般的类欧几里得做法是基本等价的 即用这种（插入）替换形式的解构就能计算出答案 但是这里比较困难

\*

其实还有一个AwD的思路是同时维护\sum a^x\*b^y和\sum a^x\*(b^0+...+b^(y-1))

利用合适的差分手段 以及等比求和公式 能把两部分和相互转换 也是能求解的 但是必须要b-1有逆才能做 矩阵的话就不太行了

能拿到60-80分

\*

基于刚才的替换解构 逆向求解 考虑某层如果0和1最终会变成s0 s1 那么前一层的0和1最终会变成s0,s1的一些拼接

形式一点就是可以列出S0='0',S1='1',S2,S3,...,St，每个Si(i>1)可以表示成p\*Sj+q\*Sk(j,k<i)这样

并且最终串就是St 且t=O(log) 也就是你得到了一个O(log)层的对整个串的解构

那么就能（xjb上）跟线段树一样做了 把一个区间扔下去递归解构就好了

大概还需要记一个struct node表示对于一个01串需要维护的信息 写一个拼接函数就好了

还有一些没有解决的事情就是实际上是(Ax+C)/B 如果对着这个直接翻转细节会比较多 然而一定能找到一个等价起始位置

具体说应该是As%B/gcd(A,B)=C%B/gcd(A,B) 这样的话（其实本来只要前缀）就必须要做区间 似乎会导致递归解构时的更多细节

如果做前缀相减的话会要求b可逆 矩阵的情况下可能还是会不太行

最后就是注意一下不要爆64位存储类型