XJTU Contest 7 solution

author： ddd , SunshinePie

Animism

首先我们设$f_{i,j}$：只能走i的子树中的点，从i出发最后回到i，一开始有j升油，回到i的时候最多能有几升油?

那么想象一下一个从x到y的旅程，设$x->y$在树上的路径为$a_1,a_2,a_3....$,一开始有$f_{a_1,p}$升油（记为q_1），到达a_2时有$q_2=f_{a_2,q_1-1}$升油......$q_i=f_{a_i,q_{i-1}-1}$

性质一：$f_{i,j}=j$或者$f_{i,j+1}$，此性质比较显然

因此我们可以把f_i表示成一些块$[l_1,r_1],[l_2,r_2].....[l_k,r_k]$，满足$\forall u\in[l_j,r_j],f_{i,u}=r_j$

我们可以把这些块用set维护（每个点维护一个set），特别地，这个set里可以只维护长度大于1的块（因为长度等于1的块是平凡的，但结合下文，虽然我们不必维护平凡块，但是我们可以允许set里有平凡块存在，不影响）

我们可以考虑一下一个set传给它父亲时set里的内容会发生什么变化

1.这个set里长度为2的块会消失

2.这个set里长度大于2的块$[l,r]$会变成$[l+1,r-1]$

发现我们并不方便直接朴素地修改整个set，所以我们对每个区间记$[l,r,dep]$表示这个区间是在深度为dep的点处加入的，那么当你在深度为dep_2的点访问它时，它的实际值为$[l+dep-dep_2,r-(dep-dep_2)]$

对这个set进行启发式合并，合并两个子树的set其实就是把这些区间做$union$生成的新区间集合，用STL即可完成。

至于为什么是取$union$的，形象的理解一下，大概是它可以多次经过i，比如一开始油量是x，在一个子树里走了一圈回到i变成$x+1$升油，然后又去另一个子树里走一走回来变成$x+2$.....抽象一下就会发现本质上就是区间$union$

现在我们$O(n\log^2 n)$求出了所有的f

现在着手处理询问，我们把标记打在树上，(x,y,p)，把起点标记打在x上，把终点标记打在y上

我们考虑对于每个i的子树，维护：

1.对于子树中的每个起点，它走到i以后，最多还有多少油

2.对于子树中的每个终点，只走i子树中的点，从i出发，至少一开始需要有多少油，才能走到终点。

考虑维护这些信息，以处理起点信息为例，处理终点是相似的：

考虑一个起点从x的答案转移到x的父亲fa_x的答案，有两部分

1.首先它减1，因为走了一条树边

2.找到fa_x的set中包含这个值的区间$[l,r]$，这个值更新为r(即此题解第三行所谓的$q_i=f_{a_i,q_{i-1}-1}$)

第一步自不必说，我们直接来说第二步可以暴力在set中查找，找到的所有起点的值都被设为了r，因此用带权并查集将他们合并即可。

因此，启发式合并中的siz应视为子树内起点/终点数+子树大小。维护起点信息是类似的。

在一个询问的lca处看一下这趟旅程是否已经能满足，但注意，这时的是否满足我们还没考虑lca子树外的加油站。为了考虑上这一点，我们需要进行更多的工作。我们下面称仅使用lca子树内部加油站，无法完成的询问为坏询问

std的做法是，给关于起点的带权并查集中的每个块开一个set2，用来存储**这个块中所有坏询问的起点对应的终点**。当然这时并查集的合并也就需要按照set2的大小启发式合并。发现set2中的这些坏终点一旦插进来虽然值会改变，但是相对顺序不会改变，因此我们就以每个元素插进来时和其他元素比较的顺序作为set2的顺序。同一个set2中的坏终点值更新时，由于内部有序，因此可达成旅行的那些坏终点一定是一个前缀，因此暴力查找这些可达成的询问即可，并将它们删掉。

$O((n+m+q)\log^2(n+m+q))$

备注：本题由于过于难写，所以目前除了std，没有验题人验过，所以有小概率数据错误（不过数据大概还是对的，std跟暴力对拍过很多数据）

Bitwise Xor

考虑从第r个人到第l个人倒着插入线性基，然后从高位往低位贪心，看看这一位上的基是哪个人贡献的，那么这一位就归这个人管（按照这个人的喜好p贪心）。因为我们是倒着插的线性基，所以这个人一定是最后一个能掌握这个二进制位的，所以他掌控这一位。

因此其实修改p是非常简单的，我们要做的就是对于n个人的每个前缀，求出从后往前插入的线性基，并记录每个基是哪个人提供的，这是一个经典算法。

$O((\sum m+q)\log A)$

Counting

1.设P_1,P_2为两个$1-n$排列,首先我们把第一个排列中的数字视为红色，第二个排列中的数字视为蓝色

我们考虑将这两个排列进行归并排序，当相等时可随意取一个元素，首先考虑**区分颜色，最后得到的数列A有多少种可能**.答案是$\prod_{i=1}^n(i^2+1)$。这是因为考虑从n到1倒着插入数字，红i和蓝i要么是头元素（见下文），要么插在某个同颜色的$j,(j>i)$后面，总共$(n-i+1)^2$种，但当两者都成为头元素时，顺序有红前蓝后和蓝前红后两种，因此总方案数为$(n-i+1)^2+1$，将这些贡献全部乘起来即可。

2.如果A中一个元素A_i是A的前缀i的不严格最大值，那么我们称这个元素为头元素，设这些头元素的下标为$b_1....b_k$,对于一个合法数列A（此处忽略颜色），头元素为$b_1....b_k$，分割成了k个子段
$S=\{T\in A\text{的子段}|T_i=A[b_i,b_{i+1})\}$。$S$有唯一一种划分
$S=S_1\cup S_2......\cup S_m,\forall S_i,\text{每个}S_i\text{中存在的数字都出现了正好两次，并且这个}S_i\text{不可再分}$ （即m尽可能大）

3.设原问题答案的指数型生成函数为G,问题1的答案的指数型生成函数是$F(x)=\sum_{i=0}^{\infty}(\prod_{j=0}^{i}(j^2+1))\frac{x^i}{i!}$

那么对于一个不区分颜色的合法排列A,它在有颜色版本中被计算了2^m次（划分的每个S_i都可以取一个为红色另一个为蓝色）。它在G^2这个生成函数中也被计算了2^m次（前面的G提供一个划分的子集，后面的G提供其他部分，共2^m个子集）

因此$G^2=F,G=\sqrt{F}$，多项式开根即可

Decision

本质上就是求对于 $v_1\in[0,t],v_2\in[0,t]$ 的一共 $(t+1)^2$ 种情况，统计有多少种情况是偶数。

首先考虑对于 v_1+v_2 的各种取值，$|v_1-v_2|$ 的可能取值都是什么：当 v_1+v_2 为偶数时，$|v_1-v_2|$ 的取值为 $0,2,4,\cdots,v_1+v_2$；当 v_1+v_2 为奇数时，$|v_1-v_2|$ 的取值为 $1,3,5,\cdots,v_1+v_2$。这是v_1+v_2小于t的情况，大于t时是类似的。

那么枚举 v_1+v_2 的取值，对于每种取值，只需要求出从这个值往后跳若干步的值的奇偶性之和。由于产生的有向图是环套树森林，倍增即可。时间复杂度 $O(m\log m)$。

Expectation

一共$2n$段路，设第i段路最终期望被经过a_i次，算出所有a_i即可根据期望线性性算出答案

显然，$\forall k>0, a_{2k-1}=a_{2k}$

我们先计算a_1,显然第一段路被经过最多一次，有$\frac{(2n-1)!!}{2^n n!}$的概率不会被经过，因此a_1好算。

考虑$a_{2k+1}-a_{2k}$，不难发现第$(2k+1)$段路比第$(2k)$段多经过1次，-1次或0次。因此我们要计算的实际上是前者比后者先被经过的概率减去后者比前者先被经过的概率

设$dp[n][m]$表示一个空的洞，左边有n个球，右边有m个球，右边球掉下来的概率减去左边球掉下来的概率。

$dp[0][n]=1-\frac{(2n-1)!!}{2^n n!}$

$dp[n][0]=-dp[0][n]$

$dp[n][m]=\frac{(2n-1)dp[n-1][m]+(2m-1)dp[n][m-1]}{2(n+m)}$

有了这些dp值，我们即可将a递推出来$a_{2k+1}=a_{2k}+dp[k][n-k]$,也就得到了最终答案。

这是本题的$O(n^2)$做法，已经可以通过此题

bonus：事实上，本题还存在$O(n\log n)$的做法，不过验题时发现此做法难度可能较大，经验题人建议只把n^2版本出到了比赛中，大家可以自行思考一下。

Flower

转化成对偶问题：给每个节点赋一个权值，使得每朵花内部点的权值和大于等于这朵花的权值，总点权最小是多少。

按照花的深度最低点从高到低进行排序，每次都尽量把权值放在深度低的点即可，询问一朵花内的点当前权值和，给一个点权值增加，可用动态点分治套数据结构实现

$O(m\log^2 n)$

Game

考虑如下伪代码：

```
S = 点集
vector<点集> V
while (S非空)
    d = S中的最远点对的距离
    S2 = 空集
    for (auto p: S)
        if (S中存在点q, 使得dis(p,q) == d)
            S2.insert(p)
    将S2中的所有点从S中删去
    vec.push_back(S2)
```

如果 vec.back() 的点集中只有一个点，且为 1 号点，那么先手败；否则先手胜。

必胜策略：如果 1 号点所在集合还有别的点，那么往同集合内最远的点走。这样后手必须往前面的集合走，先手可以继续在集合内选，后手必须跨集合，以此类推。最后走到第一个集合，后手没法走了，但是先手总能在同集合内找到可以走的点。

方法二：考虑直接$dp[i][j]$表示不考虑不能走重复点的限制，当前在i，上一步是j能否可以赢。可以证明直接这样dp最终得到的就是真实答案。

Heart

前置知识：子集卷积算法，设多项式$A=\sum_{i=0}^{2^n-1} a_i x^i, B=\sum_{i=0}^{2^n-1} b_i x^i$

求$C=A*B=\sum_{i=0}^{2^n-1} x^i \sum_{d\subseteq i} a_d b_{i-d}$可以在$O(n^2 2^n)$时间内完成

那么这里考虑倍增子集卷积，将所有弹幕按照所需碎片集合b的最高二进制位分类，得到多项式$A_1,A_2,...,A_{21}$，A_i的度数为2^i

将这些多项式依次子集卷积起来得到多项式即可$O(1)$计算答案，子集卷积的复杂度为$O(\sum_{i=1}^{21} i^2 2^i)=O(w^2 2^w)$

好像有些3^n+剪枝水过去了，大概是因为评测机波动大加上数据确实没卡剪枝，非常抱歉。

LIS/LDS

引理：对于一个长度为 $a\times b+1$ 的序列，如果 $\text{LIS}\le a$，那么 $\text{LDS}>b$。证明显然。因此讨论三种情况：

- 如果 $n\ge a\times b+1$，由引理知，无解。
- 如果 $n<a+b-1$，显然无解。
- 否则 $a+b-1\le n\le a\times b$。把序列分成 a 块，设第 i 块的长度为 x_i，由 $x_a=b, 1\le x_i\le b, x_1+x_2+...+x_a=n$ 得到一组解 $(x_1,x_2,\cdots,x_a)$，那么逐个枚举块，块内从后往前依次填入数字即可。题目要求字典序最小，那么只需要在求 x 时让越往后的块尽可能大即可。

Jogging

首先如果起点与对角线连通，那么答案就是 0。否则，可以证明起点所在连通块的大小不会很大，暴力 bfs 出所有点建图即可。建出图来之后，答案就是 "起点的度数+1" 除以 "总度数+n"，其中 n 表示连通块大小，证明可以考虑：假设极限存在且唯一，那么此答案满足对应的约束关系。至于证明极限的存在性，需要涉及关于邻接矩阵的特征值等内容，可以自行查阅“图上随机游走”相关书籍文献。

证明连通块大小不会很大的过程：对于一个点 (x, y)，假设其不与对角线连通，那么不妨设 $x > y$。设 $x_1 \le x \le x_2$ 满足 x_1, x_2 是相邻的两个质数，那么对于 $x = x_1$ 和 $x = x_2$ 这两条线来说，地图上对角线下方的部分全部为墙。因此可以证明连通块大小一定不会大于 $O(g_n^2)$，其中 g_n 表示不大于 n 的相邻质数之差的最大值，而 $MAX(g_{10^{12}})$ 只有几百。同时这是一个极松的估算，实际测试的结果远远小于这个数字。

Kcats

考虑笛卡尔树，原排列中的1的右边，单调栈个数都会+1，然后左边部分和右边部分就无关了

$dp[i][j][k]$表示区间$[i, j]$是笛卡尔树中深度为k的子树，有多少种方案，转移时，枚举根在哪个位置，并乘上对应组合数dp即可。设枚举的根的位置是p，合法的转移一定形如$0 \le k \le i \le p \le j$，因此时间复杂度$O(n^4)$，常数很小，为$\frac{1}{24}$