IOI2014 DAY2

徐寅展 <xuyinzhan@gmail.com>

杭州学军中学

February 9, 2015

Overall Statement

初始有一个序列，由 $1 \sim n$ 循环位移得到，即可能为 $(i, \cdots, n, 1, \cdots, i-1)$，$i$ 是 1 到 n 范围内的任意一个数字。
之后有若干操作，每次操作时，首先找到当前最小的还未用过的编号 id，将序列中的某个数字替换为 id。
定义替换序列为每次操作中被替换的数字组成的序列。

Overall Statement

$$\begin{array}{ccccc} 4 & 5 & 1 & \underline{2} & 3 \\ & & \Downarrow & & \\ 4 & \underline{5} & 1 & 6 & 3 \\ & & \Downarrow & & \\ 4 & 7 & 1 & 6 & \underline{3} \\ & & \Downarrow & & \\ 4 & 7 & 1 & 6 & 8 \end{array}$$

Statement

判断一个操作完之后的序列是否合法。

$1 \leq n \leq 100000, \ 1 \leq inputSeq[i] \leq 250000$

Solution

首先要所有数字互不相同。
如果一个数字大于n，那么总可以是合法的。
对于小于等于n的位置要判断相对位置。

Statement

将被删除的数字依次写成一个序列，称这个序列为替换序列。
给定一个合法的最终序列，求出一个替换序列。
$1 \le n \le 100000, 1 \le inputSeq[i] \le 250000$

Solution

如果存在 $1 \sim n$ 中任意一个，那么可以确定最开始的序列；否则任选一个初始序列。
从小到大枚举之后放进去的数字，如果出现在最终序列中，那么放在该位置，否则放在任意一个非确定的位置。

Statement

求替换序列个数，对1000000009取模。

$1 \leq n \leq 100000, 1 \leq inputSeq[i] \leq 1000000000$

Solution

如果存在1到n中的任意一个，那么可以确定最开始的序列；否则任选一个初始序列，最后答案乘以n即可。
将所有数字排序后，每一段可以填的位置个数是相等的。

Statement

有一个点带权的无向图，最开始只有点0，随后点1至点$n-1$依次加入。点i加入时，会有一个已经加入的点作为它的$host$，记为$host_i$，它会在点i和其他一些已经加入的点之间连边。具体连边方式有以下三种：

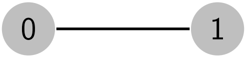
- I方式：只将i与$host_i$连边。
- M方式：只将i与$host_i$的所有邻居连边（$host_i$本身不与i连边）。
- W方式：将i与$host_i$及其所有邻居连边。

Statement

初始时只有一个点0.

Statement

1号点的*host*为0，连边方式为I:

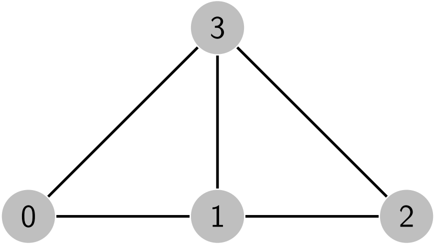


Statement

2号点的$host$为0，连边方式为M:

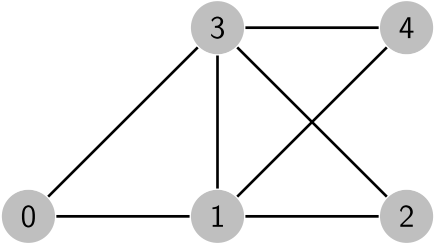
Statement

3号点的$host$为1，连边方式为W:



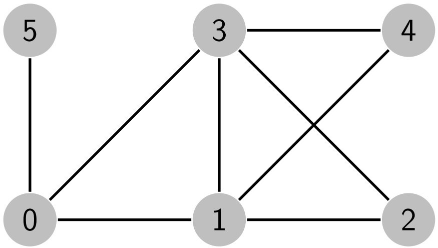
Statement

4号点的$host$为2，连边方式为M:



Statement

5号点的*host*为0，连边方式为*I*:



Statement

现在已知每个点的点权，*host*以及连边方式，求该图的最大点权独立集。
$1 \le n \le 100000$, 时间限制$1s$.

Solution

注意到如果将$host_i$当做点i的父亲，我们就能得到一棵以点0为根的树，其中每个孩子节点根据相应的点的连边方式不同可以分为I、M和W三种类型。考虑树形DP。

Solution

令$f(i)$表示点i可以被选的情况下以i为根的子树的最大权值，$g(i)$表示点i不能被选的情况下以i为根的子树的最大权值（这里的能否被选是在不考虑以i为根的子树的情况下）。

To Calculate $g(i)$

首先考虑$g(i)$的计算。因为点i是不能选的，所以点i的所有M类和W类孩子都是不能选的，I类孩子是可以选的。因此

$$g(i) = \sum_{u} g(u) + \sum_{v} f(v)$$

其中u是i的M类或W类儿子，v是i的I类孩子。

To Calculate $f(i)$

对于$f(i)$，我们有i号节点被选和未被选两种情况。
下面对于两种情况分别求出$f(i)$，再取两者中的较大值作为$f(i)$的值。

Case 1

第一种情况，我们选择了点i，那么点i的所有I类和W类孩子就不能选了，而M类孩子仍是可以选的，所以在这种情况下

$$f(i) = \sum_u g(u) + \sum_v f(v) + w(i)$$

其中u是i的I类或W类孩子，v是i的M类孩子。

Case 2

第二种情况，也就是不选择点i的情况稍微复杂一些，我们需要决定i的每个孩子u是否能够被选择。
唯一的限制是，一旦我们决定了一个I类或W类儿子是可以选择的，那么在它之后加入（编号比它大）的M类孩子和W类孩子就是不能选择的了。我们可以对i的所有孩子进行一个简单的DP来作出最优决定。

Complexity

最终，$f(0)$即为答案。该算法的时间复杂度为$O(n)$。

Statement

n个城市依次排开，编号为0到$n-1$。i号城市与$i-1$号城市和$i+1$号城市相邻（0号与$n-1$号特殊）。
一开始你在$start$号城市。每一天，你可以选择移动到相邻的一个城市，或者游玩当前城市，并获得a_i的娱乐值，其中i是你现在所在的城市编号。
你不能游玩同一个城市多次，但是能多次经过同一个城市。
问d天你能获得的最大愉悦值是多少。
$1 \le n \le 100000$，$1 \le d \le 2n + \lfloor n/2 \rfloor$，所有城市的愉悦值均为非负整数。时间限制5s.

Special Case

起点在0号城市
可以发现，在这种情况下，只会一直往右移动，而不会回头。因此可以枚举往右走到的最右边那个城市，然后再从剩余的天数中，选择愉悦值最大的若干个城市进行游览。
后者可以用可持久化线段树实现。

Special Case EXT

从 0 出发，但需要对 $1 \sim d$ 天都求出答案。

Property

Theorem

令 f_d 为可以游览 d 天，最优的那个右端点（有多个相同取最左边一个）。我们有如果 $x < y$ ，那么 $f_x \leq f_y$.

Proof

Proof.

考虑 f_x 与 f_{x+1} ，假设 $f_{x+1} < f_x$ 。
先把能游览$x+1$天的右端点与游览x天的右端点放在f_x处。右端点每次向左移动一格，游览x天的与游览$x+1$天的，都能游览一个新的城市（当最右端的那个城市本来也被选时，是两个新的城市）。但由于$x+1$天的本来就比x天的游览了更多的城市，所以这个x天的新城市的愉悦值要大于等于$x+1$ 天的新城市的愉悦值。
当它们移动到f_{x+1}时，x天的增加的愉悦值大于等于$x+1$天的增加的愉悦值。于是就矛盾了。因此有$f_x \leq f_{x+1}$。 □

Algorithm

有了这条性质，接下去就容易解决了。
考虑分治，要求d在$[l, r]$中的所有f_d，并且知道这些f_d在$[a, b]$中。令$mid = \lfloor \frac{l+r}{2} \rfloor$，首先暴力找到$f_{mid}$,接下去递归分别找$f_l \ldots f_{mid-1}$,$f_{mid+1} \ldots f_r$。其中，$f_l \ldots f_{mid-1}$都在$[a, f_{mid}]$之间；$f_{mid+1} \ldots f_r$都在$[f_{mid}, b]$之间。这样总的时间复杂度是$O(n \log^2 n)$的。

Left Part

考虑原问题。
最优策略肯定是往右走再折回左边，或者往左走再折回右边。
既然可以对每一个d都求出只在左边或者只在右边的答案，也可以求出对于每一个d往一个方向走，再折回$start$的答案（具有类似上文中的单调性）。
那么只要求出两边分别对于所有d的答案后，就能求出整个问题的最优解了。时间复杂度为$O(n\log^2 n)$.