水题选讲

ljfcnyali

2020 年 11 月 25 日

CF Gym 101190I

这是一道交互题

现在有一张有向图，保证所有点出度均为 m，但并不清楚点数和具体边集，你位于未知的某点，你需要遍历所有边集（可以重复经过）。

每个点有一个石头，初始状态为中，你每次到达一个点时，可以修改石头的状态为左或右（但不能为中），同时，石头可以标记一条出边，初始状态为任意的一条出边。

每次你可以将当前的石头标记的边顺时针转 x 条边，并且可以调整石头状态，再选择调整前石头标记的边顺时针转 y 条边的出边移动。

注意，一个点的入边个数不清楚也不会在当前点观测到，同时保证原图强连通且所有出边相对顺序从未改变，每次交互库给出当前点石头状态左中右三种，你输出 $x, L/R, y$ 表示一次移动。

$m \leq 20$ 且可以假设点数 ≤ 20，允许重边自环，且交互次数 ≤ 20000

可以发现，石头指向的边一定为上次遇到改点时的出边，否则该信息没有用处。

可以发现，石头指向的边一定为上次遇到改点时的出边，否则该信息没有用处。

很明显我们需要依次将所有点的出边全部走完，不妨假设我们现在需要将我们当前所在的点的所有出边走完，且我们按照某种 DFS 顺序处理，那么还未走完的点一定组成一条链。

可以发现，石头指向的边一定为上次遇到改点时的出边，否则该信息没有用处。

很明显我们需要依次将所有点的出边全部走完，不妨假设我们现在需要将我们当前所在的点的所有出边走完，且我们按照某种 DFS 顺序处理，那么还未走完的点一定组成一条链。

那么我们假设所有出边走完的点状态为 L，未走完的为 R，未访问过的为 C，且当前点在一条 R 组成的链上。

可以发现，石头指向的边一定为上次遇到改点时的出边，否则该信息没有用处。

很明显我们需要依次将所有点的出边全部走完，不妨假设我们现在需要将我们当前所在的点的所有出边走完，且我们按照某种 DFS 顺序处理，那么还未走完的点一定组成一条链。

那么我们假设所有出边走完的点状态为 L，未走完的为 R，未访问过的为 C，且当前点在一条 R 组成的链上。

我们将当前点指向上次访问的边，依次处理其所有出边重复 m 次，并且每次我们试图回到当前点。

可以发现，石头指向的边一定为上次遇到改点时的出边，否则该信息没有用处。

很明显我们需要依次将所有点的出边全部走完，不妨假设我们现在需要将我们当前所在的点的所有出边走完，且我们按照某种 DFS 顺序处理，那么还未走完的点一定组成一条链。

那么我们假设所有出边走完的点状态为 L，未走完的为 R，未访问过的为 C，且当前点在一条 R 组成的链上。

我们将当前点指向上次访问的边，依次处理其所有出边重复 m 次，并且每次我们试图回到当前点。

考虑如果当前出边走到一个 R 点，即形成了一个 R 环，那么标记此点为 L，而我们可以遍历整个环知道该环大小，再将 L 修改回 R 且走环大小-1 步回到正在处理的点

可以发现，石头指向的边一定为上次遇到改点时的出边，否则该信息没有用处。

很明显我们需要依次将所有点的出边全部走完，不妨假设我们现在需要将我们当前所在的点的所有出边走完，且我们按照某种 DFS 顺序处理，那么还未走完的点一定组成一条链。

那么我们假设所有出边走完的点状态为 L，未走完的为 R，未访问过的为 C，且当前点在一条 R 组成的链上。

我们将当前点指向上次访问的边，依次处理其所有出边重复 m 次，并且每次我们试图回到当前点。

考虑如果当前出边走到一个 R 点，即形成了一个 R 环，那么标记此点为 L，而我们可以遍历整个环知道该环大小，再将 L 修改回 R 且走环大小-1 步回到正在处理的点

否则走到一个 L 点，那么 L 点一直按照其标记的点走一定会回到某一个 R 点上，再一直走即可以得到环大小，同理可以回到正在处理的点

可以发现，石头指向的边一定为上次遇到改点时的出边，否则该信息没有用处。

很明显我们需要依次将所有点的出边全部走完，不妨假设我们现在需要将我们当前所在的点的所有出边走完，且我们按照某种 DFS 顺序处理，那么还未走完的点一定组成一条链。

那么我们假设所有出边走完的点状态为 L，未走完的为 R，未访问过的为 C，且当前点在一条 R 组成的链上。

我们将当前点指向上次访问的边，依次处理其所有出边重复 m 次，并且每次我们试图回到当前点。

考虑如果当前出边走到一个 R 点，即形成了一个 R 环，那么标记此点为 L，而我们可以遍历整个环知道该环大小，再将 L 修改回 R 且走环大小-1 步回到正在处理的点

否则走到一个 L 点，那么 L 点一直按照其标记的点走一定会回到某一个 R 点上，再一直走即可以得到环大小，同理可以回到正在处理的点

如果走到的是 C 点，拓展 R 链即可

现在考虑当前处理的点所有出边访问完时，我们需要回到上一个 R 链上的点

现在考虑当前处理的点所有出边访问完时，我们需要回到上一个 R 链上的点

若我们可以记录某一条出边组成的环，那么我们顺着那个环走即可回到上一个点

现在考虑当前处理的点所有出边访问完时，我们需要回到上一个 R 链上的点

若我们可以记录某一条出边组成的环，那么我们顺着那个环走即可回到上一个点

因为存在自环或 L 链与当前点组成的环，所以并不是所有环均可以回到上一个点，可以试图记录一个经过 R 点个数最多的环，这样的环一定合法

现在考虑当前处理的点所有出边访问完时，我们需要回到上一个 R 链上的点

若我们可以记录某一条出边组成的环，那么我们顺着那个环走即可回到上一个点

因为存在自环或 L 链与当前点组成的环，所以并不是所有环均可以回到上一个点，可以试图记录一个经过 R 点个数最多的环，这样的环一定合法

具体的，开一个栈记录 R 链上的所有点的信息，每次顺带维护以上信息即可。