C 相对位置

题目描述

在二维平面直角坐标系中，共有 n 个点，其中第一个点的位置在**原点**。给出 m 组两个点之间的相对位置。请你求出全部点的具体坐标。

输入

第一行有两个整数 $n(1 \leq n \leq 10^4)$ 和 $m(1 \leq m \leq 2 \times 10^5)$，表示点的个数和关系的个数。

接下来有 m 行，每一行有四个整数 $p1, p2, dx, dy$, 代表 $p1$ 和 $p2$ 两个点的位置关系。满足关系式:

$$dx = x_{p2} - x_{p1},\ dy = y_{p2} - y_{p1}$$

保证 $1 \leq p1, p2 \leq n, 0 \leq |dx|, |dy| \leq 10^9$。且每个点坐标如果可以确定，只有唯一解。

输出

对于每组数据，输出 n 行，第 i 行表示第 i 个点的坐标。

如果可以通过上述关系确定该点具体位置，则输出对应的 x 和 y 坐标值。 反之则输出 `undecidable`。

输入样例

```
5 3
1 2 2 1
1 3 -1 -2
4 5 9 6
```

输出样例

```
0 0
2 1
-1 -2
undecidable
undecidable
```

思路分析

一句话题意：给定m组两点之间的关系和初始已知点原点，求其他点是否可求，若可求则输出具体坐标。

将每条关系看成边，可以组成图。每个点的坐标是否可求是图的连通性问题，点的具体求解依靠于是否拥有前驱节点。因此可以使用深搜加回溯的方式遍历所有可达点并求出具体坐标。

具体实现

用结构体nodes数组承装输入的所有点之间关系，并进行反向存储。

用数组edge将输入的关系与点进行对应进行剪枝. (蒟蒻语:不进行剪枝纯暴力貌似会t🙄)

使用visit数组记录某个节点是否搜索到过 x,y数组分别记录每个点的x,y坐标.

从原点进行深度优先搜索,在搜索到新的节点时根据前继节点和两点间的关系求出坐标,并继续深搜,直到将可求点坐标全部求出.

核心代码

```
void dfs(int k){
    for(auto a :edge[k]){
        int f=a.first,s=a.second;
        if(visitt[f]){
            continue;
        }
        else{
            visitt[f]=true;
            if(nodes[s].p2==f){
                x[f]=x[k]+nodes[s].dx;
                y[f]=y[k]+nodes[s].dy;
            }
            else{
                x[f]=x[k]-nodes[s].dx;
                y[f]=y[k]-nodes[s].dy;
            }
            dfs(f);
        }
    }
}
```

完整实现

```
#include<bits/stdc++.h>

#define P pair<int,int>
#define ll long long
#define N 200010
using namespace std;
int m, n;
struct node {
    int p1, p2, dx, dy;
} nodes[1000000];
vector<P > edge[N];
ll x[N], y[N];
bool visitt[N];

void dfs(int k) {
    for (auto a: edge[k]) {
        int f = a.first, s = a.second;
        if (visitt[f]) {
            continue;
        } else {
            visitt[f] = true;
            x[f] = x[k] + nodes[s].dx * (nodes[s].p2 == f ? 1 : -1);
            y[f] = y[k] + nodes[s].dy * (nodes[s].p2 == f ? 1 : -1);
            dfs(f);
        }
    }
}

int main() {
```

```
    ios::sync_with_stdio(false);
    cin.tie(0),cout.tie(0);

    cin >> n >> m;
    for (int i = 1; i <= m; i++) {
        cin >> nodes[i].p1 >> nodes[i].p2 >> nodes[i].dx >> nodes[i].dy;
        edge[nodes[i].p1].push_back({nodes[i].p2, i});
        edge[nodes[i].p2].push_back({nodes[i].p1, i});
    }
    visitt[1] = true;
    dfs(1);
    for (int i = 1; i <= n; i++) {
        if (visitt[i]) {
            cout << x[i] << " " << y[i] << "\n";
        } else {
            cout << "undecidable" << "\n";
        }
    }
    return 0;
}
```

运行结果

交上去就会发现ac!😊 (大概150-250ms之间)

Accepted	1	c++	1188	155	13704
Accepted	1	c++	1365	255	13584

题外话

推荐几道深搜回溯的题目,大家感兴趣可以做做

P1219 [USACO1.5] 八皇后 Checker Challenge

P1294 高手去散步

P1706 全排列问题

谢谢大家😎