> 计算学部寒假十大打卡活动——“龙舞编程新春会”编程打卡
>
> 2024年1月23日

一、题目：AcWing P3200 无线网络

目前在一个很大的平面房间里有 n 个无线路由器，每个无线路由器都固定在某个点上。

任何两个无线路由器只要距离不超过 r 就能互相建立网络连接。

除此以外，另有 m 个可以摆放无线路由器的位置。

你可以在这些位置中选择至多 k 个增设新的路由器。

你的目标是使得第 1 个路由器和 2 个路由器之间的网络连接经过尽量少的中转路由器。

请问在最优方案下中转路由器的最少个数是多少?

输入格式

第一行包含四个正整数 n,m,k,r。

接下来 n 行，每行包含两个整数 xi 和 yi，表示一个已经放置好的无线路由器在 (xi,yi)点处。输入数据保证第 1 和第 2 个路由器在仅有这 n 个路由器的情况下已经可以互相连接(经过一系列的中转路由器)。

接下来 m 行，每行包含两个整数 xi 和 yi，表示 (xi,yi) 点处可以增设一个路由器。

输入中所有的坐标的绝对值不超过 10^8，保证输入中的坐标各不相同。

输出格式

输出只有一个数，即在指定的位置中增设 k 个路由器后，从第 1 个路由器到第 2 个路由器最少经过的中转路由器的个数。

测试样例

样例 1：

输入：

```
5 3 1 3
0 0
5 5
0 3
0 5
3 5
3 3
4 4
3 0
```

输出

```
2
```

样例 2：

```
13 3 1 3
0 0
5 5
-3 0
-3 2
-3 4
-3 6
-1 7
0 5
3 5
1 -2
3 -2
5 -2
5 1
3 3
4 4
3 0
```

输出

```
5
```

数据范围

$2 \le n \le 100,$
$1 \le k \le m \le 100,$
$1 \le r \le 10^8$

二、解答

1. 整体思路：dijsktra+拆点

一眼可知最短路问题，根据问题不妨将每条边长视为1，使用dijsktra算法作为单源最短路径问题处理。由于到达每个点时增设路由器的不确定数，我们需要同时记录到达一个点的增设路由器数与此时的最小中转路由器数，即将一个点拆分为多个点。

显然，由于我们设边长为1，因此一个点在某一增设路由器数下的最小中转路由器数就是我们第一次通过另一个点访问到该状态时的数量，故每个点仅需访问1次。且无需采用优先队列，用正常的queue即可。

2. 代码

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=205;
int n,m,k,r,head[maxn],ek,nk,vis[maxn][maxn],dis[maxn][maxn];

struct EDGE{//边长记录
    int next,v;
}e[10005];
```

```
struct Node{//记录每个点的坐标
    int x,y;
}node[maxn];

struct que{
    int x,cnt;
};

void add(int x,int y){//双向加边操作
    e[++ek].v=y;e[ek].next=head[x];head[x]=ek;
    e[++ek].v=x;e[ek].next=head[y];head[y]=ek;
}

bool judge_dis(int a,int b)//判断a,b两个路由器是否可以互相建立网络
{
    double disx=node[a].x-node[b].x,disy=node[a].y-node[b].y;
    double dis=sqrt(disx*disx+disy*disy);
    return dis<=r;
}

int main()
{
    //freopen("in.txt","r",stdin);

    //按照1~n,n+1 ~ m分别读入这n+m个路由器，根据其编号可判断其类型
    scanf("%d %d %d %d",&n,&m,&k,&r);
    for(nk=1;nk<=n;nk++)
        scanf("%d %d",&node[nk].x,&node[nk].y);
    for(int i=1;i<=m;i++,nk++)
        scanf("%d %d",&node[nk].x,&node[nk].y);
    nk--;//总路由器数多算了1

    //计算数组中每两个点间的距离，如果距离小于r，就在这两点之间建立一条无向边。
    for(int i=1;i<nk;i++)
        for(int j=i+1;j<=nk;j++)
        if(judge_dis(i,j))
            add(i,j);


    //dijsktra算法
    queue<que>q;
    q.push((que){1,0});
    vis[1][0]=1;

    while(!q.empty())
    {
        que qf=q.front();
        q.pop();
        int cnt=qf.cnt;

        for(int i=head[qf.x];i;i=e[i].next)
        {
            int y=e[i].v;
            int ycnt=cnt+(y>n);//判断该点是否为增设路由器，是否需要改变到达y点的增设路由器
数
```

```
            if(y==2)//到达终点
            {
                printf("%d",dis[qf.x][cnt]);//第一次访问即最小，也就是答案
                return 0;
            }
            else if(ycnt<=k&&(!vis[y][ycnt]))//在满足条件的情况下，将
            {
                vis[y][ycnt]=1;
                dis[y][ycnt]=dis[qf.x][cnt]+1;
                q.push((que){y,ycnt});//压入队列
            }
        }
    }

    return 0;
}
```

三、总结

题目是较为常见的dijsktra变种问题，值得注意的是对细节的一些处理：

1. 在计算两点之间的距离的时候，虽然单点的坐标数字的绝对值不超过10^8，但在计算两点距离$z=\sqrt{x^2+y^2}$的时候还是存在可能导致数字范围超过int，加上结果很可能不是整数，因此改为double类型。
2. 将每条边长视为1，大大简化了我们的运算过程，不必将到达点2 的每个状态（每种增设路由器数）都运行完再结束，第一次到达即是最短
3. 对于边长不等的正常情况，需要使用优先队列；同时由于使用的是自定义的结构体类型，也需要重载结构体运算符。