C++算法

现场挑战
快快编程1056

12权图

SSSP

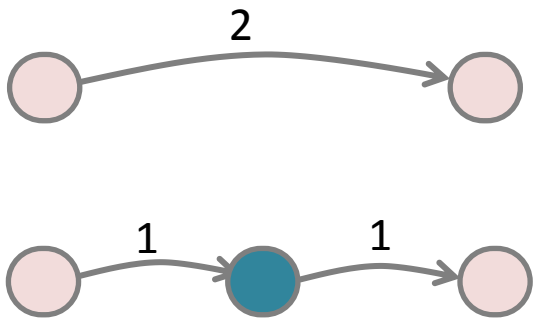
Dijkstra+堆优化

复杂度
O(ElogE)

能否再加快速度?

边变链

每条2权边变两条1权边



建新图/增加点

```
 9 void add(int a,int b){
10     to[a].push_back(b);
11 }
```

易错点

原图点数边数和
新图点数边数混淆

```
33     int x,y;
34     cin>>nOld>>x>>y;
35     int a,b;
36     for(int i=0;i<x;i++)
37         cin>>a>>b,add(a,b);
38     int nNew=nOld;
39     for(int i=0;i<y;i++)
40         cin>>a>>b,add(a,++nNew),add(nNew,b);
```

建新图/增加点

第一行包含正整数n,x和y
n<=10000,x,y<=200000

```
3 const int N=10009;
4 const int M=          ;
5 const int INF=2e9;
6 int nOld,d[N+M];
7 bool vst[     ];
8 vector<int> to[     ];
```

易错点

图储存数组太小

拆点易错点

易错点
原图点数边数和 新图点数边数混淆

易错点
图储存数组太小

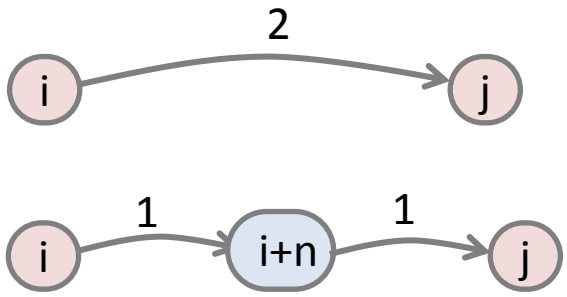
12权图

能否再加快速度?

拆点

每个节点加一个辅助点

i号节点对应增加(i+n)号



```
33      cin>>nOld>>x>>y;
34      int nNew=nOld*2;
35      int a,b;
36      for(int i=0;i<x;i++)
37          cin>>a>>b,add(
38      for(int i=1;i<=nOld;i++)
39          add(
40      for(int i=0;i<y;i++)
41          cin>>a>>b,add(
```

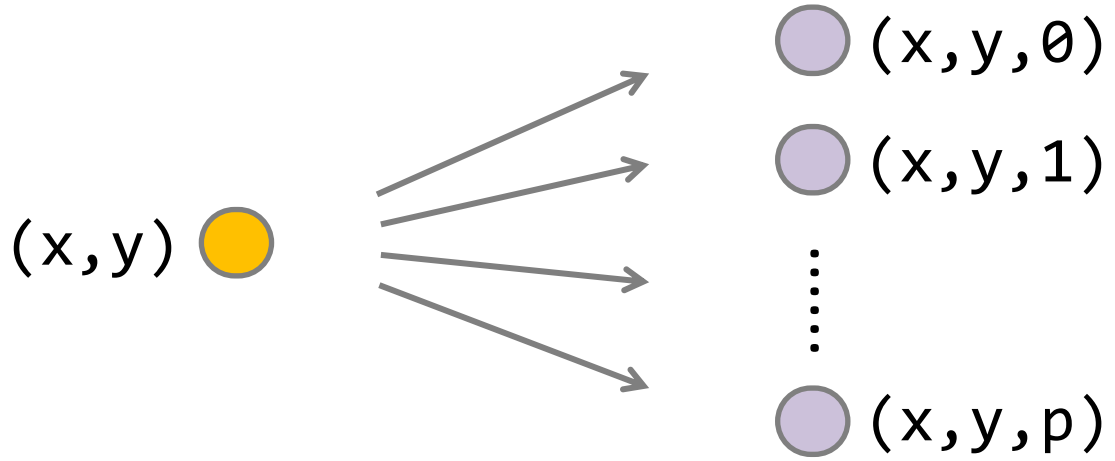
建新图/增加点

```
3 const int N=10009;
4 const int M=400009;
5 const int INF=2e9;
6 int nOld,d[      ];
7 bool vst[      ];
8 vector<int> to[      ];
```

现场挑战
快快编程1048

拆点

原图中1个节点对应
新图中多个节点



拆点 | 分层图 | 状态具体化

算法建模

从(1,1)格子
到(n,m)格子
最多撞墙p次
最少几步

从(1,1,0)状态
到(n,m,?)
其中之一的状态
最少几步

使用什么算法?

无权图最短路
用BFS

```
38      cin>>n>>m>>p;
39      for(int i=1;i<=n;i++)
40          for(int j=1;j<=m;j++)
41              cin>>mp[i][j];
42      int ans=bfs();
43      if(ans!=-1)cout<<ans<<endl;
44      else cout<<"mission impossible"<<endl;
```

```
3  const int N=509;
4  const int P=11;
5  int dx[4]={0,1,0,-1};
6  int dy[4]={1,0,-1,0};
7  struct Node{int x,y,z;};
8  int n,m,p,d[N][N][P];
9  bool vst[N][N][P];
10 char mp[N][N];
```

d[x][y][k]表示从起点到(x,y)时
剩k次撞墙机会最少要几步

vst[x][y][k]表示什么含义?

```
int bfs(){
    queue<Node> q;
    vst[1][1][p]=1;
    d[1][1][p]=0;
    q.push((Node){1,1,p});
    while(!q.empty()){
        Node now=q.front(); q.pop();
        int x=now.x,y=now.y,z=now.z;
        for(int k=0;k<4;k++){
            int nx=x+dx[k],ny=y+dy[k],nz=z;
            if(nx<1||nx>n||ny<1||ny>m)continue;
            if(mp[nx][ny]=='#'){
                if(!z)continue;
                nz=z-1;
            }
            if(vst[nx][ny][nz])continue;
            vst[nx][ny][nz]=1;
            d[nx][ny][nz]=d[x][y][z]+1;
            q.push((Node){nx,ny,nz});
            if(nx==n&&ny==m)return d[nx][ny][nz];
        }
    }
    return -1;
}
```

讨论题

单源单汇，无向图，
节点代表城市，非负边权代表通行时间
可以重复走，但不能停留
能否恰好在时刻T从源到汇?

时空状态

T较小时

i号节点拆成 (i,0),(i,1),...,(i,T)

vst[i][t]代表能否恰好在时刻t来到i

判断连通性

T较大时

i号节点拆成 (i,0),(i,1),...,(i,T)
内存不够?
速度太慢?

讨论题

单源单汇，无向图，
节点代表城市，非负边权代表通行时间
可以重复走，但不能停留
能否在时刻$[T_1,T_2]$从源到汇?

现场挑战
快快编程1057

```
struct station{int x,y;} p[M];
```

								起点	终点
1	2	3	4	…	…	…	m	m+1	m+2

拆成两层

									起点	终点
横向	1	2	3	4	…	…	…	m	m+1	m+2
纵向	1 +m+2	2 +m+2	3 +m+2	4 +m+2	…	…	…	m +m+2	m+1 +m+2	m+2 +m+2

```
struct station{int x,y;} p[M];
```

1	2	3	4	…	…	…	m	起点 m+1	终点 m+2

拆成两层

横向

1	2	3	4	…	…	…	m	起点 m+1	终点 m+2

纵向

1 +m+2	2 +m+2	3 +m+2	4 +m+2	…	…	…	m +m+2	m+1 +m+2	m+2 +m+2

用下标数组id[]排序

id[i]表示排序后第i个车站原来是几号

```
struct station{int x,y;} p[M];
```

1	2	3	4	…	…	…	m	起点 m+1	终点 m+2

拆成两层

	1	2	3	4	…	…	…	m	起点 m+1	终点 m+2
横向										
纵向	1 +m+2	2 +m+2	3 +m+2	4 +m+2	…	…	…	m +m+2	m+1 +m+2	m+2 +m+2

用下标数组id[]排序

行号优先排序cmpX

同一行邻居双向边

p[id[i+1]].x==p[id[i]].x

(p[id[i+1]].y-p[id[i]].y)*2

列号优先排序cmpY

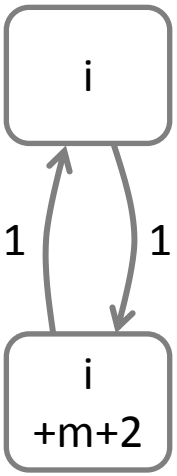
同一列邻居双向边

p[id[i+1]].y==p[id[i]].y

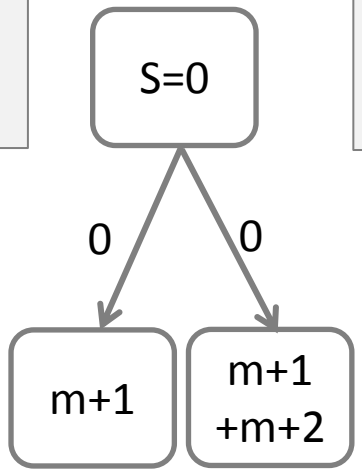
(p[id[i+1]].x-p[id[i]].x)*2

换乘
双向边

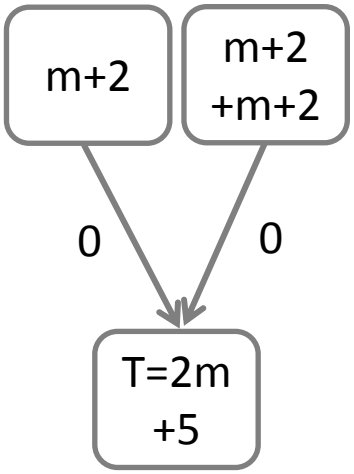
i=1,2,...m



超级源
S=0



超级汇
T=2m+5



用
下
标
数
组
id[]
排
序

行号优先排序cmpX

同一行邻居双向边
p[id[i+1]].x==p[id[i]].x

(p[id[i+1]].y-p[id[i]].y)*2

i=1,2,...m+1

列号优先排序cmpY

同一列邻居双向边
p[id[i+1]].y==p[id[i]].y

(p[id[i+1]].x-p[id[i]].x)*2

i=1,2,...m+1

新建图后，共有几个点?

新建图后，共有几条有向边?

```
3  const int M=100009;
4  const int INF=2e9;
5  struct station{int x,y;} p[M];
6  struct Node{
7      int u,c;
8      bool operator<(const Node&a)const{
9          return c>a.c;
10     }
11 };

12 vector<int> to[M*2],w[M*2];
13 bool ok[M*2];
14 int n,m,nNode,S,T,d[M*2],id[M];
15 void add(int a,int b,int c){
16     to[a].push_back(b); w[a].push_back(c);
17     to[b].push_back(a); w[b].push_back(c);
18 }
```

```
41 bool cmpX(const int&a,const int&b){
42     return p[a].x<p[b].x||p[a].x==p[b].x&&p[a].y<p[b].y;
43 }
44 bool cmpY(const int&a,const int&b){
45     return p[a].y<p[b].y||p[a].y==p[b].y&&p[a].x<p[b].x;
46 }
```

```
36 void build(){
37     for(int i=1;i<=m+2;i++)id[i]=i;
38     sort(id+1,id+1+m+2,cmpX);
39     for(int i=1;i<=m+1;i++)if(p[id[i+1]].x==p[id[i]].x)
40         add(id[i],id[i+1],(p[id[i+1]].y-p[id[i]].y)*2);
41     sort(id+1,id+1+m+2,cmpY);
42     for(int i=1;i<=m+1;i++)if(p[id[i+1]].y==p[id[i]].y)
43
44     for(int i=1;i<=m;i++)
45         add(i,i+m+2,1);
46     S=0; T=m+2+m+2+1;
47     add(S,m+1,0); add(S,m+1+m+2,0);
48     add(m+2,T,0);
49     nNode=2*m+6;
50 }
```

```
19  int dijkstra(){
20      fill(d,d+1+nNode,INF);
21      priority_queue<Node> q;
22      d[S]=0;
23      q.push((Node){S,0});
24      while(!q.empty()){
25          int u=q.top().u;    q.pop();
26          if(u==T)return d[T];
27
28          ok[u]=1;
29          for(int i=0;i<to[u].size();i++){
30              int v=to[u][i];
31              int cost=
32              if(d[v]<=cost)continue;
33              d[v]=cost;
34              q.push((Node){v,d[v]});
35          }
36      }
37      return -1;
38  }
```

快快编程作业

1056
1048
1057

拓展题	1485,392,1130