

1.题目名称

最小生成树

2.代码行数

161行

3.算法思想

分别调用Kruskal算法和Prim算法求最小生成树以及它的权值即可。

4.主要/核心函数分析

Prim

```
1 void Prim(){
2     dist[0]=0;
3     visit[0]= true;
4     path[0]="v1";
5     for(int i=1;i<n;i++){
6         dist[i]= min(dist[i],Matrix[0][i]);
7         path[i]= "v"+to_string(1)+"->"+"v"+ to_string(i+1);
8     }
9     cout<<"v"<<1<<"加入顶点集合"<<endl;
10    cout<<path[0]<<endl;
11    Print();
12    for(int i=1;i<n;i++){
13        int temp=-1;
14        int min=INF;
15        for(int j=1;j<n;j++){
16            if(!visit[j]&&dist[j]<min){
17                min=dist[j];
18                temp=j;
19            }
20        }
21        if(temp===-1){
22            flag= true;
23            return;
24        }
25
26        visit[temp]= true;
27        sum+=dist[temp];
28        for(int j=1;j<n;j++){
29            if (dist[j]>Matrix[temp][j]){
30                dist[j]=Matrix[temp][j];
31                path[j]=path[temp]+"->"+"v"+ to_string(j+1);
32            }
33        }
34        cout<<"v"<<temp+1<<"加入顶点集合"<<endl;
35        cout<<path[temp]<<endl;
36        Print();
37    }
38 }
```

1. 从源点出发，将所有与源点连接的点加入一个待处理的集合中
2. 从集合中找出与源点的边中权重最小的点，从待处理的集合中移除标记为确定的点
3. 将找到的点按照步骤1的方式处理
4. 重复2，3步直到所有的点都被标记

Kruskal

```

1 void kruskal(){
2     Initial();
3     for(int i=0;i<m;i++){
4         int x= find(e[i].u);
5         int y= find(e[i].v);
6         if(x!=y){
7             cout<<e[i].u<<"->"<<e[i].v<<endl;
8             if(Union(x,y)){
9                 sum+=e[i].w;
10            }
11        }
12    }
13 }
```

对边的权值按从小到大排。

从小到大依次进行插入，如果边关联的两个点属于同一个树则直接跳过不执行插入。

5.测试数据(规模,测试次数)

规模:边的权值为float类型，运算结果不超过float类型最大值

测试次数:1

测试用例:见测试文件

6.运行结果

```

1 F:\data_structure\Choice\question22\cmake-build-debug\question22.exe
2 v1加入顶点集合
3 v1
4 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
5 0 3.7 2 INF INF INF INF INF INF
6
7 v3加入顶点集合
8 v1->v3
9 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
10 0 3.7 2 INF INF 4 6 INF INF INF
11
12 v2加入顶点集合
13 v1->v2
14 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
15 0 3.7 2 4 5.5 4 6 INF INF INF
16
17 v4加入顶点集合
18 v1->v2->v4
19 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
20 0 3.7 2 4 5.5 4 6 6 INF 8
```

```

21
22 v6加入顶点集合
23 v1->v3->v6
24 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
25 0 3.7 2 4 5.5 4 1.2 2 4 8
26
27 v7加入顶点集合
28 v1->v3->v6->v7
29 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
30 0 3.7 2 4 5.5 1.2 1.2 2 4 8
31
32 v8加入顶点集合
33 v1->v3->v6->v8
34 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
35 0 3.7 2 4 3 1.2 1.2 2 4 4.4
36
37 v5加入顶点集合
38 v1->v3->v6->v8->v5
39 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
40 0 3.7 2 4 3 1.2 1.2 2 4 4.4
41
42 v9加入顶点集合
43 v1->v3->v6->v9
44 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
45 0 3.7 2 4 3 1.2 1.2 2 4 4.4
46
47 v10加入顶点集合
48 v1->v3->v6->v8->v10
49 v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
50 0 3.7 2 4 3 1.2 1.2 2 4 4.4
51
52 ALL:28.3
53
54 6->7
55 1->3
56 6->8
57 5->8
58 1->2
59 2->4
60 6->9
61 3->6
62 8->10
63 ALL:28.3
64
65 进程已结束,退出代码0
66

```

7.时间复杂度分析

Prim算法时间复杂度为 $O(n^2)$

Krustal算法的时间复杂度为 $O(n \log n)$

8.结果截屏图片

```
运行: question22 x
0 3.7 2 4 3 1.2 1.2 2 4 4.4
ALL:28.3
6->7
1->3
6->8
5->8
1->2
2->4
6->9
3->6
8->10
ALL:28.3
进程已结束,退出代码0
```

9.心得体会

对于两种求最小生成树的算法更加了解与熟悉。同时也明白了两种算法的时间复杂度优劣。