

# 天津大学本科实验报告专用纸

学院 智能与计算学部 年级 2018 级 专业 软件工程 班级 6 姓名 王传安  
学号 3018216301 课程名称 算法设计与分析 实验日期 2019/11/22  
同组实验者 无 成绩           

一、实验目标：

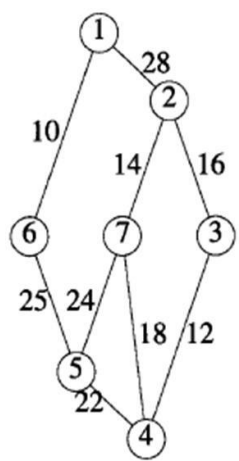
求出所给图的最小生成树。

二、实验内容：

最小生成树是指将给出的所有点连接起来（即从一个点可到任意一个点），且连接路径之和最小的图。

输入描述：用邻接矩阵描述下图。

输出描述：最小权值和 sum，对应的边的集合 S。



三、实验步骤：

克鲁斯卡尔算法的具体思路是：将所有边按照权值的大小进行升序排序，然后从小到大一一判断，条件为：如果这个边不会与之前选择的所有边组成回路，就可以作为最小生成树的一部分；反之，舍去。

# 天津大学本科实验报告专用纸

直到具有  $n$  个顶点的连通网筛选出来  $n-1$  条边为止。筛选出来的边和所有的顶点构成此连通网的最小生成树。

判断是否产生回路：在初始状态下给每个顶点赋予不同的标记，对于遍历过程的每条边，其都有两个顶点，判断这两个顶点的标记是否一致，如果一致，说明它们本身就处在一棵树中，如果继续连接就会产生回路；如果不一致，说明它们之间还没有任何关系，可以连接。

定义边：

```
struct edge
{
    int u, v;           // 边的两个端点编号
    int cost;           // 边权
    edge(int x,int y, int c):u(x),v(y),cost(c){}
};
```

findFather 函数返回 x 的根节点：

```
int findFather(vector<int> father, int x)
{
    while (x != father[x])
        x = father[x];
    return x;
}
```

这里用根节点来对每一个顶点进行标记。

初始化 father:

```
int NumEdge = 0;
for (int i = 0; i < n; i++)
    father[i] = i;
```

判断当前边的两个顶点的根节点是否相同，若不同，合并，然后加入这条边:

```
int faU = findFather(father, E[i].u);
int faV = findFather(father, E[i].v);
if (faU != faV) {
    father[faU] = faV;
    ans += E[i].cost;
    NumEdge++;
    cout << E[i].u << "--" << E[i].v << endl;
    if (NumEdge == n - 1)
        break;
}
```

当边的数目=顶点数-1 时，说明最小生成树已经生成，此时 break:

```
if (NumEdge == n - 1)
    break;
```

当把边遍历完（或者已经生成树 break）后，如果边的数目！= 顶点数-1，说明没有连通，返回-1:

```
if (NumEdge != n - 1) //无法连通时返回-1
    return -1;
else
    return ans; //返回最小生成树边权之和
```

四、 实验结果:

如下图:

```
最小生成树的边为:
1--6
3--4
2--7
2--3
4--5
5--6
最小权值为: 99
```

教师签字:

年 月 日

