最小生成树P算法

主讲老师：党东

【最小生成树问题】

生成树：一个|V|个点的无向连通图中，取其中|V|-1条边，并连接所有的顶点，则为原图的一棵生成树。

树的属性：树是图的一种特殊形态。一个图G是树当且仅当以下任意一个条件成立：

①G有V-1条边，无圈；

②G有V-1条边，连通；

③任意两点只有唯一的简单路径；

④G连通，但任意删除一条边后不连通；

【最小生成树问题】

无向图的最小生成树（贪心思想）

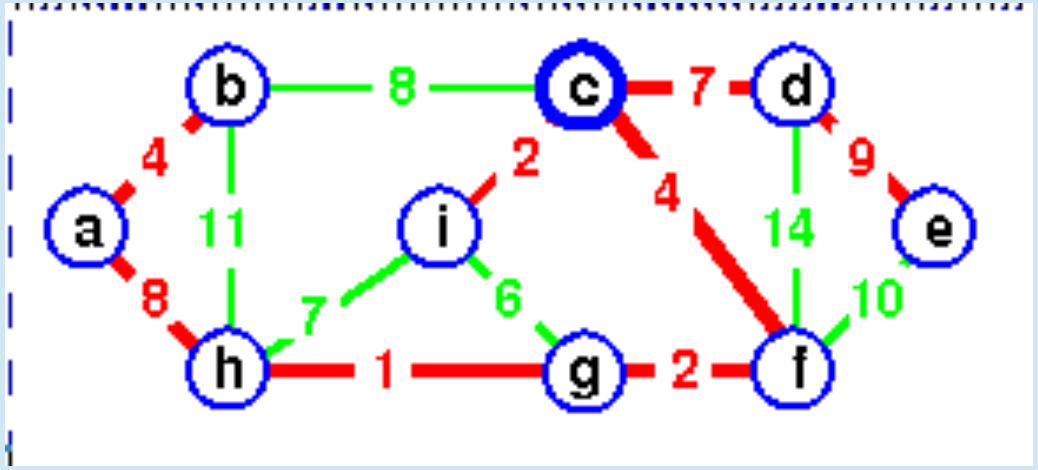
- Prim算法，适用于点少的图
- Kruskal算法，适用于边少的图

有向图的最小树形图

【最小生成树问题】

最小生成树：在一张带权的无向连通图中，各边权和为最小的一棵生成树即为最小生成树。

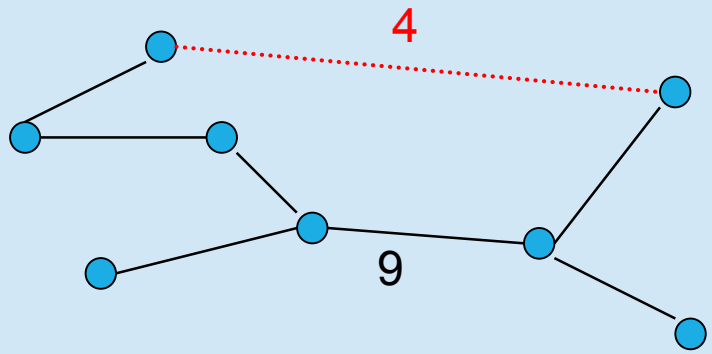
简单讲：找出连接所有点的最低成本路线



红边连接了所有顶点，
所以构成一棵生成树
权和=1+2+2+4+4+7+8+9

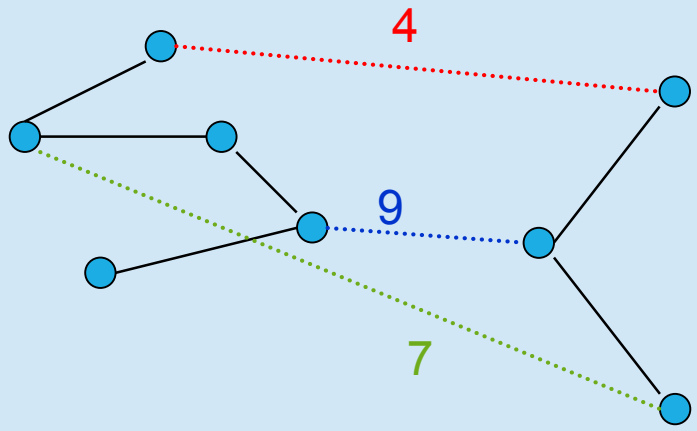
最小生成树(MST)--算法原理

环属性：一棵生成树上，增加一条边e，再删除e所在环上的最大边，会得到另一棵“更好”的生成树(如果e不是最大边)



最小生成树(MST)--算法原理

剪切属性：在图中，剪切将顶点划分成两个不相交集合。交叉边为这些顶点在两个不同集合的边。对于任何一个剪切，各条最小的交叉边都属于某个MST，且每个MST中都包含一条最小交叉边。



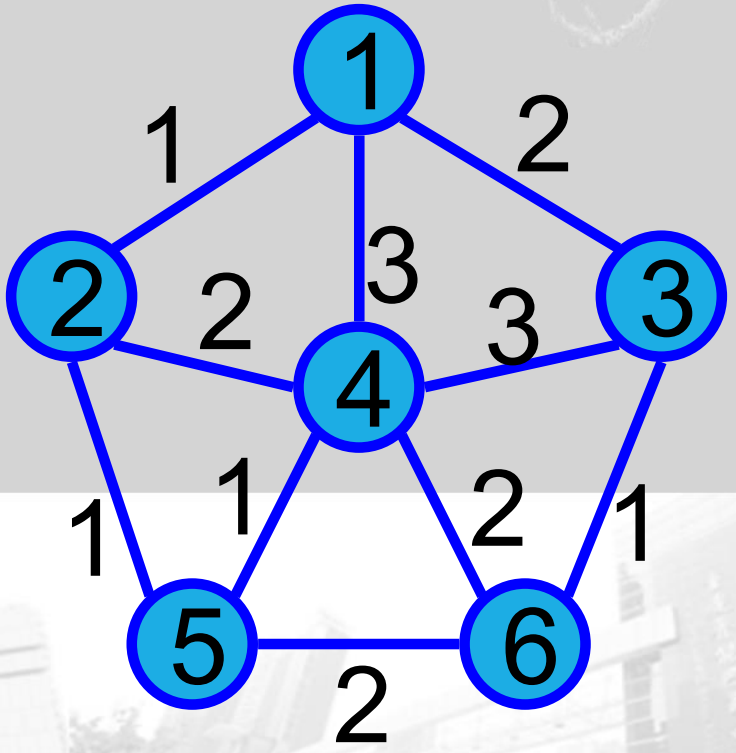
最小生成树(MST)--算法原理

最小边原则： 图中权值最小的边(如果唯一的话)一定在最小生成树上。

唯一性： 一棵生成树上，如果各边的权都不相同，则最小生成树是唯一的。反之不然。

Prim算法---构建过程

1. 将1号节点置入集合S中。
2. 找到所有连接S中的节点和非S中的节点的边中的权值最小的那一条，并标记这条边，同时将连接的非S中的节点加入S集合。
3. 重复2步骤，直到所有节点都在S中了。



Prim算法---算法步骤

选定图中的任意一个顶点 V_0，从 V_0 开始生成最小生成树：

①初始化 dist[v0]=0，其他点的距离值 dist[i]=∞。其中 dist[i]表示集合 V_B 中的点到 V_A 的距离值。

②经过 N 次如下步骤操作，最后得到一棵含 N 个顶点，N-1 条边的最小生成树：

A.选择一个未标记的点 k，并且 dist[k]的值是最小的；

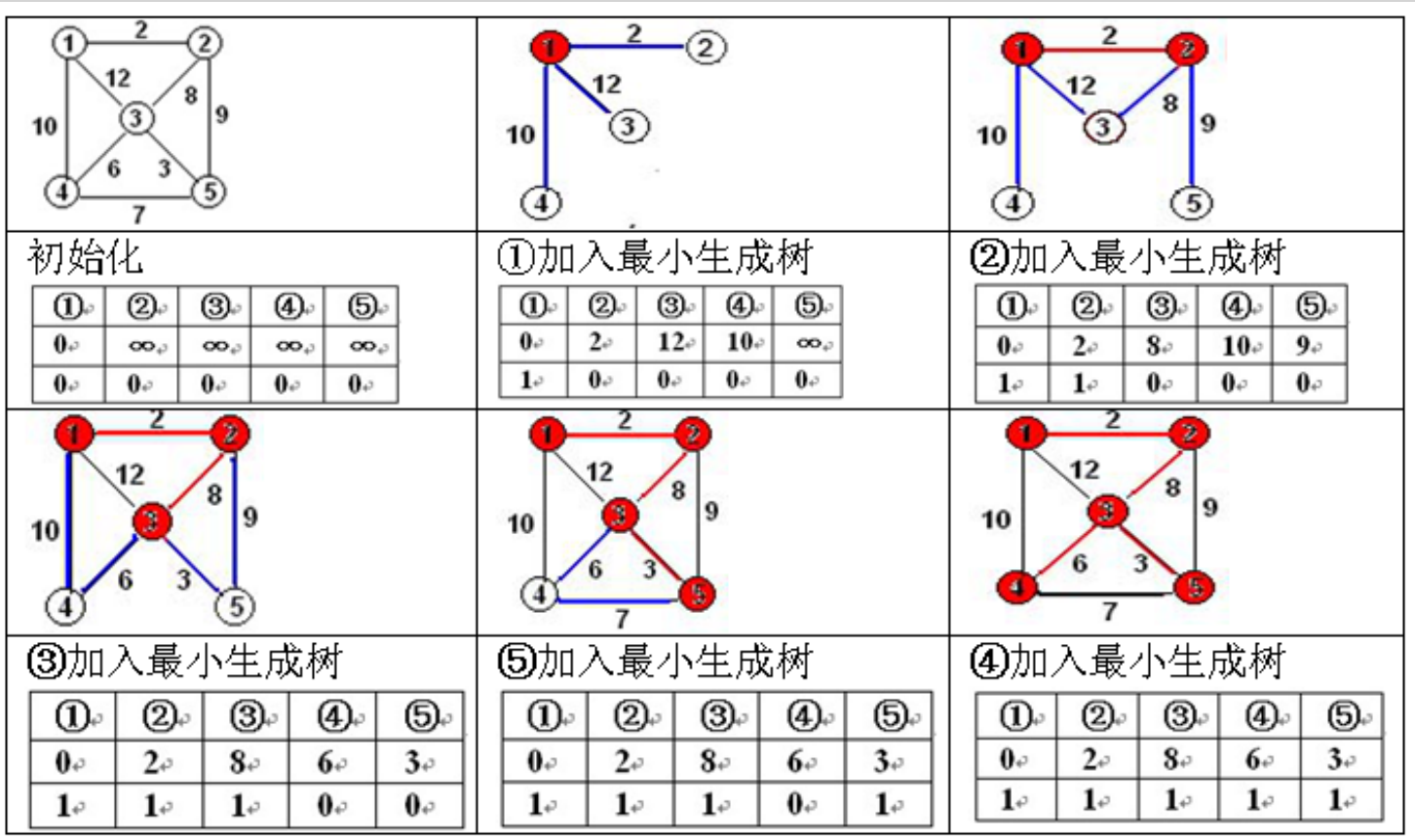
B.标记点 k 进入集合 V_A；

C.以 k 为中间点，修改未标记点 j，即 V_B 中的点到 V_A 的距离值；

③最后得到最小生成树 T。

Prim算法---算法步骤

下面图为从顶点1开始生成最小生成树的过程：



初始化

①	②	③	④	⑤
0	∞	∞	∞	∞
0	0	0	0	0

①加入最小生成树

①	②	③	④	⑤
0	2	12	10	∞
1	0	0	0	0

②加入最小生成树

①	②	③	④	⑤
0	2	8	10	9
1	1	0	0	0

③加入最小生成树

①	②	③	④	⑤
0	2	8	6	3
1	1	1	0	0

⑤加入最小生成树

①	②	③	④	⑤
0	2	8	6	3
1	1	1	0	1

④加入最小生成树

①	②	③	④	⑤
0	2	8	6	3
1	1	1	1	1

任意时刻的中间结果都是一棵树：从某一个点开始，每次都花最小的代价，用一条边加进一个新的点。

Prim算法---算法证明

这样做是对的吗？——下面用数学归纳法证明

- ①当只取了一个点K时（边集为空），一定存在一个MST，包含当前的点集和边集
- ②假设存在一个MST包含当前的点集和边集。当前点集为S，剩下的点集为S’。设跨越S-S’的最小代价的边为(u,v)
- （反证法）假设取的是跨越S-S’的某边(u’,v’)，删除(u’,v’)加入(u,v)，S和S’分别连通，且S-S’通过(u,v)也能够连通，还得到了一个权值更小的MST！所以新加入的边一定是代价最小的边(u,v)。
- 根据① ②，prim算法的正确性得证。

Prim算法---核心代码（邻接矩阵实现）


```
int vst[505];//vst[i]标记顶点i是否加入最小生成树中
int d[505];//d[i]表示不是生成树中点i到当前生成树中点到最小值
int g[505][505],n,m,ans=0;
void Read()//读入数据,构建图
{   int i,j,x,y,w;
    cin>>n>>m;
    for(i=1;i<=n;i++)
          for(j=1;j<=n;j++)g[i][j]=INF;
    for(i=1;i<=m;i++){cin>>x>>y>>w;g[x][y]=g[y][x]=w;}
}
int main()
{   Read();
    Prim(1);
    cout<<ans<<endl;
}
```

Prim算法---核心代码（邻接矩阵实现）

```
void Prim(int v0)
{   int i,j,k,minn;
    memset(vst,0,sizeof(vst));  //初始化生成树点集合
    for(i=1;i<=n;i++)d[i]=INF;d[v0]=0;
    ans=0;
    for(i=1;i<=n;i++) //选择n个点
    {   minn=INF;
        for(j=1;j<=n;j++)//选择最小边
            if(vst[j]==0 && minn>d[j]){minn=d[j];k=j;}
        vst[k]=1;  //标记
        ans+=d[k];
        for(j=1;j<=n;j++)   //修改d数组
            if(vst[j]==0&&d[j]>g[k][j])d[j]=g[k][j];
    }
}
```

Prim算法---核心代码（邻接表实现）

```
void Prim(int v0)
{    int i,j,y,k;
     for(i=1;i<=N;i++)d[i]=INF;
     d[v0]=0;//初始化d数组
     ans=0;
     for(i=1;i<=N;i++)
     {    minn=INF;
          for(j=1;j<=N;j++)
               if(!vst[j]&&minn>d[j]){k=j;minn=d[j];}//选
          ans+=d[k];  //累加最小生成树的边值
          vst[k]=1;  //标记顶点y被选入最小生成树
          for(j=h[k];j;j=a[j].next) //修改
          {    y=a[j].to;
               if(!vst[y]&&a[j].v<d[y])d[y]=a[j].v;
          }
     }
}
```

```
void Read()
{   int i,x,y,v;
    cin>>N>>M;
    for(i=1;i<=M;i++)
    {   cin>>x>>y>>v;
        AddEdge(x,y,v);
        AddEdge(y,x,v);
    }
}
```

Prim算法---优化方法

上述算法的时间复杂度为O(N^2)，能进行优化么?

关键在于第2步：求出最小生成树的n-1条边(重复操作n-1次)，由于边数一定，肯定要重复操作n-1次，只能优化每次选取的最小值。

【思考】：怎样快速找到权值最小的边？——我们需要借助于堆

使用堆来保存已选集合中点到未选集合中点的最短边长并维护其最小值，并在访问每条边的时候更新。

每次取最小值为O(logm)，总时间复杂度为O((n+m)logm)。

【例1】局域网(net) --1453

Description

某个局域网内有n(n<=100)台计算机，由于搭建局域网时工作人员的疏忽，现在局域网内的连接形成了回路，我们知道如果局域网形成回路那么数据将不停的在回路内传输，造成网络卡的现象。因为连接计算机的网线本身不同，所以有一些连线不是很畅通，我们用f(i,j)表示i,j之间连接的畅通程度(f(i,j)<=1000)，f(i,j)值越小表示i,j之间连接越通畅，f(i,j)为0表示i,j之间无网线连接。现在我们需要解决回路问题，我们将除去一些连线，使得网络中没有回路，并且被除去网线的Σf(i,j)最大，请求出这个最大值。

Input

第一行两个正整数n k

接下来的k行每行三个正整数i j m表示i,j两台计算机之间有网线联通，通畅程度为m。

Output

一个正整数，Σf(i,j)的最大值。

【例1】局域网(net) --1453

Sample Input

5 5

1 2 8

1 3 1

1 5 3

2 4 5

3 4 2

Sample Output

8

【例1】局域网(net) --1453

【参考代码】

```
int n,m,x,y,map[105][105],dis[105],vis[105],ans,total;
void prim(int s){
        for(int i=1;i<=n;i++) dis[i]=map[s][i];
        for(int i=1;i<=n;i++){
                int minn=0x7fffff;
                for(int j=1;j<=n;j++) if(minn>dis[j]&&!vis[j]) { s=j;minn=dis[j];}
                vis[s]=1;ans+=dis[s];
                for(int j=1;j<=n;j++)
                        if(map[s][j]&&!vis[j]&&dis[j]>map[s][j]) dis[j]=map[s][j];
        }
}
```

【例1】局域网(net) --1453

【参考代码】

```
int main() {
        memset(map,0x3f,sizeof(map));
        n=Read(),m=Read();
        for(int i=1;i<=n;i++) map[i][i]=0;
        for(int i=1;i<=m;i++)
                x=Read(),y=Read(),map[y][x]=map[x][y]=Read(),total+=map[x][y];
        prim(1);
        cout<<total-ans;
        return 0;
}
```

【例2】繁忙的都市 --1454

Description

城市C是一个非常繁忙的大都市，城市中的道路十分的拥挤，于是市长决定对其中的道路进行改造。城市C的道路是这样分布的：城市中有n个交叉路口，有些交叉路口之间有道路相连，两个交叉路口之间最多有一条道路相连接。这些道路是双向的，且把所有的交叉路口直接或间接的连接起来了。每条道路都有一个分值，分值越小表示这个道路越繁忙，越需要进行改造。但是市政府的资金有限，市长希望进行改造的道路越少越好，于是他提出下面的要求：

1．改造的那些道路能够把所有的交叉路口直接或间接的连通起来。

2．在满足要求1的情况下，改造的道路尽量少。

3．在满足要求1、2的情况下，改造的那些道路中分值最大值尽量小。

作为市规划局的你，应当作出最佳的决策，选择那些道路应当被修建。

Input

第一行有两个整数n,m表示城市有n个交叉路口，m条道路。接下来m行是对每条道路的描述，u, v, c表示交叉路口u和v之间有道路相连，分值为c。(1≤n≤300，1≤c≤10000)。

Output

两个整数s, max，表示你选出了几条道路，分值最大的那条道路的分值是多少。

【例2】繁忙的都市 --1454

Sample Input

4 5

1 2 3

1 4 5

2 4 7

2 3 6

3 4 8

Sample Output

3 6

【例2】繁忙的都市 --1454

【参考代码】

```
int n,m,x,y,map[305][305],dis[305],vis[305],ans,total;
void prim(int s){
        for(int i=1;i<=n;i++) dis[i]=map[s][i];
        for(int i=1;i<=n;i++){
                int minn=0x7fffff;
                for(int j=1;j<=n;j++) if(minn>dis[j]&&!vis[j]) { s=j;minn=dis[j];}
                vis[s]=1;ans=max(ans,dis[s]);
                for(int j=1;j<=n;j++)
                        if(map[s][j]&&!vis[j]&&dis[j]>map[s][j]) dis[j]=map[s][j];
        }
}
```

【例2】繁忙的都市 --1454

【参考代码】

```
int main() {
        memset(map,0x3f,sizeof(map));
        n=Read(),m=Read();
        for(int i=1;i<=n;i++) map[i][i]=0;
        for(int i=1;i<=m;i++)
                x=Read(),y=Read(),map[y][x]=map[x][y]=Read(),total+=map[x][y];
        prim(1);
        cout<<n-1<<" "<<ans;
        return 0;
}
```

【练习1】最优布线问题 --1450

Description

学校有n台计算机，为了方便数据传输，现要将它们用数据线连接起来。两台计算机被连接是指它们有数据线连接。由于计算机所处的位置不同，因此不同的两台计算机的连接费用往往是不同的。

当然，如果将任意两台计算机都用数据线连接，费用将是相当庞大的。为了节省费用，我们采用数据的间接传输手段，即一台计算机可以间接的通过若干台计算机（作为中转）来实现与另一台计算机的连接。

现在由你负责连接这些计算机，任务是使任意两台计算机都连通（不管是直接的或间接的）。

Input

第一行为整数n（2<=n<=100），表示计算机的数目。此后的n行，每行n个整数。第x+1行y列的整数表示直接连接第x台计算机和第y台计算机的费用。

Output

一个整数，表示最小的连接费用。

Sample Input

```
3
0 1 2
1 0 1
2 1 0
```

Sample Output

```
2
```

Hint

注：表示连接1和2，2和3，费用为2。

【练习2】最短网络 --1451

Description

农民约翰被选为他们镇的镇长！他其中一个竞选承诺就是在镇上建立起互联网，并连接到所有的农场。当然，他需要你的帮助。约翰已经给他的农场安排了一条高速的网络线路，他想把这条线路共享给其他农场。为了用最小的消费，他想铺设最短的光纤去连接所有的农场。你将得到一份各农场之间连接费用的列表，你必须找出能连接所有农场并所用光纤最短的方案。每两个农场间的距离不会超过100000。

Input

第一行：农场的个数，N（3<=N<=100）。

第二行..结尾:后来的行包含了一个N*N的矩阵,表示每个农场之间的距离。理论上，他们是N行，每行由N个用空格分隔的数组成，实际上，他们限制在80个字符，因此，某些行会紧接着另一些行。当然，对角线将会是0，因为不会有线路从第i个农场到它本身。

Output

只有一个输出，其中包含连接到每个农场的光纤的最小长度。

Sample Input

```
4
0  4  9  21
4  0  8  17
9  8  0  16
21 17 16  0
```

Sample Output

```
28
```