# 第一章 最小生成树问题

树是图论中非常重要的一类图。最小生成树问题则是其中的经典问题之一，有着极其广泛的应用。在实际应用中，许多问题的图论模型都是最小生成树，如通信网络建设、有线电缆铺设、加工设备分组等。

## 1.1 树的概念

**定义 1.1**

树：连通的无圈图称为树。

例如，图中的 $G_1$ 是树，$G_2$ 和 $G_3$ 不是树。

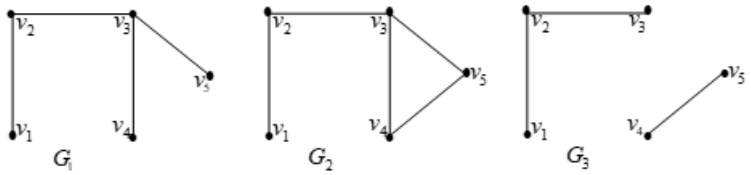



图 1.1: 树与非树

**定理 1.1**

设 $G$ 是具有 $n$ 个顶点 $m$ 条边的图，则有如下的结论：

(1) 图 $G$ 是树。

(2) 图 $G$ 中任意两个不同顶点之间存在唯一的路。

(3) 图 $G$ 连通，删除任一条边均不连通。

(4) 图 $G$ 连通，且 $n = m + 1$。

(5) 图 $G$ 无圈，添加任一条边可得唯一的圈。

**定义 1.2**

生成树：若图 $G$ 的生成子图 $H$ 是树，则称 $H$ 为 $G$ 的生成树。

连通图的生成树一定存在。一个图的生成树通常不唯一，如 $n$ 个顶点的完全图，其不同生成树的个数为 $n^{n-2}$。

### 1.1.1 最小生成树

**定义 1.3**

最小生成树：在赋权图 $G$ 中，边权之和最小的生成树。

♣

说明：

(1) 不同的方法可能得到不同的生成树。

(2) 按照生成树的定义，$n$ 个顶点的连通网络的生成树有 $n$ 个顶点、$n-1$ 条边。

构造最小生成树的准则：

(1) 必须只使用该网络中的边来构造最小生成树。

(2) 必须使用且仅使用 $n-1$ 条边来联结网络中的 $n$ 个顶点。

(3) 不能使用产生回路的边。

应用实例：欲在 $n$ 个城市间建立通信网，则 $n$ 个城市应铺 $n-1$ 条线路。注意到，铺设的选择有 $\frac{n(n-1)}{2}$ 种方式。如何选择出其中的 $n-1$ 条线路进行铺设，使得总费用最少？

分析：用顶点表示城市；边表示线路；边的权值表示线路的经济代价；连通网表示城市间的通信网。则数学模型为：$n$ 个顶点的生成树很多，需要从中选一棵代价最小的生成树，即该树各边的代价之和最小。此树便称为最小生成树 MST(Minimum cost Spanning Tree)。

## 1.2 最小生成树的构造：克鲁斯卡尔（Kruskal）算法

Kruskal 算法的基本思想是：将图中所有边按权值递增顺序排列，依次选定取权值较小的边。要求后面选取的边不能与前面选取的边构成回路，若构成回路，则放弃该条边，再去选后面权值较大的边。若顶点的个数为 $n$，则按以上法则选够 $n-1$ 条边即可。

Kruskal 算法如下。

Step 1. 选 $e_1 \in E$，使得 $e_1$ 是权值最小的边。

Step 2. 若 $e_1, e_2, \ldots, e_i$ 已选好，则从 $E \setminus \{e_1, e_2, \ldots, e_i\}$ 中选取 $e_{i+1}$，满足：

1) $\{e_1, e_2, \ldots, e_i, e_{i+1}\}$ 中无圈

2) $e_{i+1}$ 是 $E \setminus \{e_1, e_2, \ldots, e_i\}$ 中权值最小的边。

Step 3. 直到选好 $e_{n-1}$ 为止。

**例题 1.1** 一个乡有 9 个自然村，其间道路及各道路长度如图所示，各边上的数字表示距离，问架设通信线时，如何拉线才能使用线最短。

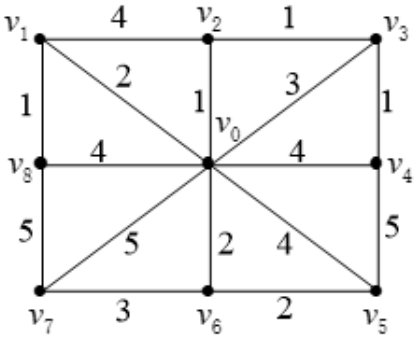



图 1.2: 道路示意图

用 Kruskal 算法求解过程如下。按边长由小至大的顺序排列：

$$l = 1 : (v_0, v_2), (v_2, v_3), (v_3, v_4), (v_1, v_8)$$

$$l = 2 : (v_0, v_1), (v_0, v_6), (v_5, v_6)$$

$$l = 3 : (v_0, v_3), (v_6, v_7)$$

$$l = 4 : (v_0, v_4), (v_0, v_5), (v_0, v_8), (v_1, v_2)$$

$$l = 5 : (v_0, v_7), (v_7, v_8), (v_4, v_5)$$

按照边的排列顺序，取：

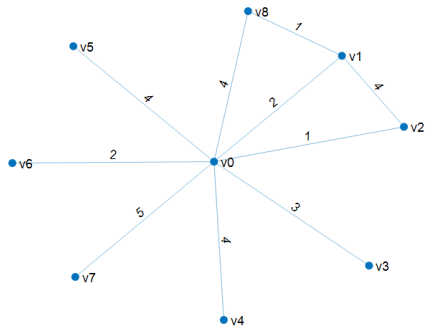
$$e_1 = (v_0, v_2), e_2 = (v_2, v_3), e_3 = (v_3, v_4), e_4 = (v_1, v_8)$$

$$e_5 = (v_0, v_1), e_6 = (v_0, v_6), e_7 = (v_5, v_6)$$

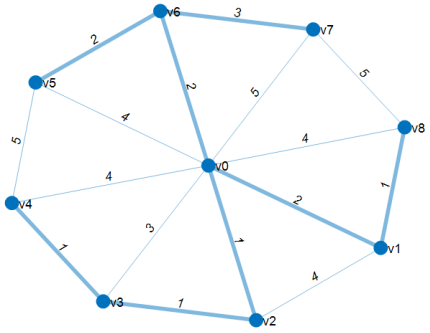
按照先小后大的原则，避圈。由于下一个未选边中的最小权边 $(v_0, v_3)$ 与已选边 $e_1, e_2$ 构成圈，排除。选中符合条件的边：

$$e_8 = (v_6, v_7)$$

原图等效于：




得到的最小生成树如下图所示，其权为 13。




Matlab 程序如下。

```
clc
clear
close all
A = zeros(9);
A(1,[2:9]) = [2 1 3 4 4 2 5 4];
A(2,[3 9]) = [4 1]; A(3,4)=1;
A(4,5) = 1; A(5,6) = 5; A(6,7) = 2;
A(7,8) = 3; A(8,9) = 5;
s = cellstr(strcat('v', int2str([0:8]'))); % 生成顶点记号
G = graph(A, s,'upper'); % 生成图
P = plot(G, 'EdgeLabel', G.Edges.Weight);
T = minspantree(G, 'Method', 'sparse') % Kruskal算法
W = sum(T.Edges.Weight)
highlight(P, T)
```

## 1.3 模型的应用

参见优秀论文 2007-B-09。

## 1.4 最小生成树的构造：普里姆算法（Prim 算法）

Prim 算法的核心是基于贪心策略，和对边进行操作的 Kruskal 算法不同，Prim 算法则是对节点进行操作，每次遍历添加一个点，直至所有的节点添加完毕。具体步骤如下表所示。

| |
|---|
| 步骤 1. 初始化阶段 |
| 1）任选起始顶点加入 MST 集合 |
| 2）创建候选边集合（连接已选与未选顶点的边） |
| 步骤 2. 迭代扩展过程 |
| 1）每次选择候选边中权重最小的边 |
| 2）将新顶点加入 MST 集合 |
| 3）更新候选边集合 |
| 步骤 3. 判断终止条件 |
| 1）当所有顶点都加入 MST 时结束 |
| 2）最终边集构成最小生成树 |

如上例中，如记初始 $u = \{v_0\}$，$v = \{v_1, v_2, \ldots, v_8\}$，则最小生成树的算法过程如下。

1. $u = \{v_0, v_2\}$, $v = \{v_1, v_3, v_4, v_5, v_6, v_7, v_8\}$
2. $u = \{v_0, v_2, v_3\}$, $v = \{v_1, v_4, v_5, v_6, v_7, v_8\}$
3. $u = \{v_0, v_2, v_3, v_4\}$, $v = \{v_1, v_5, v_6, v_7, v_8\}$
4. $u = \{v_0, v_2, v_3, v_4, v_1\}$, $v = \{v_5, v_6, v_7, v_8\}$
5. $u = \{v_0, v_2, v_3, v_4, v_1, v_8\}$, $v = \{v_5, v_6, v_7\}$
6. $u = \{v_0, v_2, v_3, v_4, v_1, v_8, v_6\}$, $v = \{v_5, v_7\}$
7. $u = \{v_0, v_2, v_3, v_4, v_1, v_8, v_6, v_5\}$, $v = \{v_8\}$
8. $u = \{v_0, v_2, v_3, v_4, v_1, v_8, v_6, v_5, v_7\}.v = \emptyset$

## 1.5 基于着色问题的模型

已知图 $G=(V,E)$，对图 $G$ 的所有顶点进行着色，要求相邻的两顶点的颜色不一样，问至少需要几种颜色？

**定理 1.2**

记 $\chi(G)$ 为图 $G=(V,E)$ 的顶点进行着色所需最少的颜色数目，则：

$$\chi(G) \leq \max\{d(v)|v \in V\} + 1.$$

♡

上面的结论给出了色数的上界，然而着色算法目前还没有找到最优算法，可以利用整数线性规划来求解这类问题。

**例题 1.2**（物资存储问题）一家公司制造 $n$ 种化学制品 $A_1, A_2, \ldots, A_n$，其中有些化学制品若放在一起可能产生危险，如引发爆炸或产生毒气等，称这样的化学制品是不相容的。为安全起见，在存储这些化学制品时，不相容的不能放在同一储存室内。问至少需要多少个存储室才能存放这些化学制品？

思路：构造图 $G$，用顶点 $v_1, v_2, \ldots, v_n$ 分别表示 $n$ 种化学制品，顶点 $v_i$ 与 $v_j$ 相邻，当且仅当化学制品 $A_i$ 与 $A_j$ 不相容。于是存储问题就化为对图 $G$ 的顶点着色问题，对图 $G$ 的顶点最少着色数目便是最少需要的储存室数。

**例题 1.3**（无线交换设备的波长分配问题）有 5 台设备，要给每一台设备分配一个波长。如果两台设备靠得太近，则不能给它们分配相同的波长，以防干扰。已知 $v_1$ 和 $v_2, v_4, v_5$ 靠得近，$v_2$ 和 $v_3, v_4$ 靠得近，$v_3$ 和 $v_4, v_5$ 靠得近，问至少需要几个发射波长。

思路：以设备为顶点构造图 $G = (V, E)$，其中，$V = \{v_1, v_2, \ldots, v_5\}$，$v_1, v_2, \ldots, v_5$ 分别代表 5 台设备。$E$ 为边集，如果两台设备靠得太近，则用一条边连接它们。于是图 $G$ 的着色给出一个波长分配方案：给着同一种颜色的设备同一个波长。画出着色图如下图所示，可知需要 3 个发射波长。

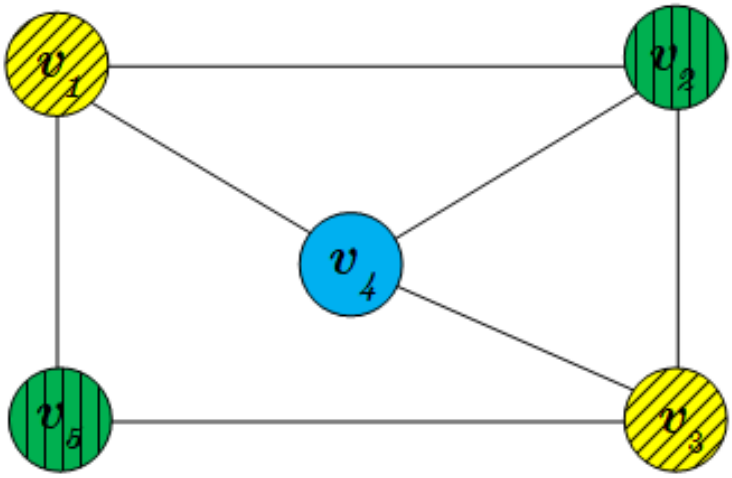



图 1.3: 5 台设备的关系图

**例题 1.4** （会议安排）学校的学生会下设 6 个部门，部门的成员如下：部门 1={张，李，王}，部门 2={李，赵，刘}，部门 3={张，刘，王}，部门 4={赵，刘，孙}，部门 5={张，王，孙}，部门 6={李，刘，王}，每个月每个部门都要开一次会，为了确保每个人都能参加他所在部门的会议，这 6 个会议至少需要安排在几个不同的时段？

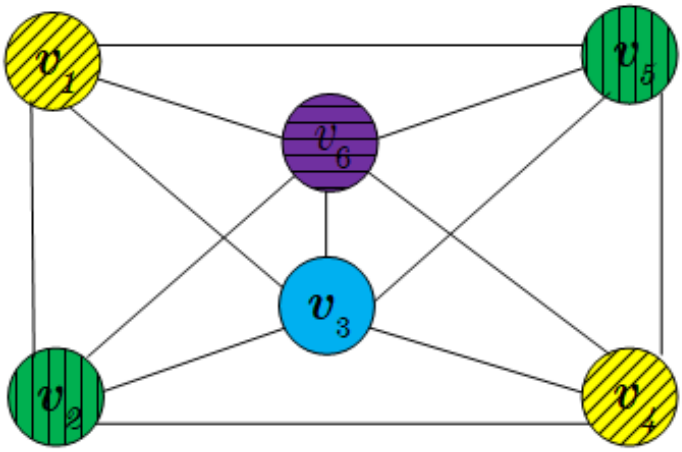


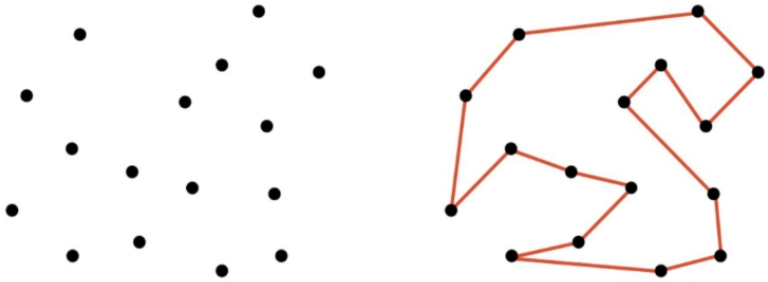

图 1.4: 部门之间关系图

构造图 $G = (V, E)$，其中 $V = \{v_1, v_2, \ldots, v_6\}$，其中，$v_1, v_2, \ldots, v_6$ 分别表示部门 1 到部门 6。$E$ 为边集，两个顶点之间有一条边当且仅当它们代表的委员会成员中有共同的人。该图可以用 4 种颜色着色。

着同一种颜色的顶点代表的部门会议可以安排在同一时间段，而不同颜色代表的部门会议必须安排在不同的时间，故这 6 个会议至少要安排在 4 个不同的时间，其中，部门 1 和部门 4，部门 2 和部门 5 的会议可以安排在同一时间段。

## 1.6 旅行商问题

旅行商问题是指旅行家要旅行 $n$ 个城市，要求每个城市经历一次且仅经历一次然后回到出发城市，并要求所走路程最短。



问题分析：从初始点出发的周游路线一共有 $(n-1)!$ 条，即等于除初始结点外的 $(n-1)$ 个结点的排列数，因此旅行商问题是一个排列问题。通过枚举 $(n-1)!$ 条周游路线，从中找出一条具有最小成本的周游路线的算法，其计算时间为 $O(n!)$。

建模思路 1：通过所给出的一个无向图，即 $n$ 个顶点，$m$ 个无向边，每条边有一个权值代表两个点之间的距离，要求把每一个点都走一遍并回到原点，求路径的最短值。

建模思路 2：将问题建模为 0-1 规划问题。

求解：通过动态规划、贪心法、分支限界等算法进行相应算法的求解。

## 1.7 模型的应用

参见优秀论文 2013-B-01、2013-B-02。

## 1.8 最大流问题

许多系统包含了流量问题，如公路系统中有车辆流、物资调配系统中有物资流、金融系统中有现金流等。这些流问题都可归结为网络流问题，且都存在一个如何安排使流量最大的问题，即最大流问题。

**定义 1.4**

网络：给定一个有向图 $D=(V,A)$，其中 $A$ 为弧集，在 $V$ 中指定了一点，称为发点或源，该点只有发出的弧；同时指定一个点称为收点，该点只有进入的弧；其余的点叫中间点。对于每一条弧 $(v_i,v_j)\in A$，对应有一个值 $c(v_i,v_j)\geq 0$，简记为 $c_{ij}$，称为弧的容量。通常把这样的有向图 $D$ 叫作一个网络，记作 $D=(V,A,C)$，其中，$C=\{c_{ij}\}$。

**定义 1.5**

网络上的流：指定义在弧集合 $A$ 上的一个函数 $f=\{f_{ij}\}=\{f(v_i,v_j)\}$，并称 $f_{ij}$ 为弧 $(v_i,v_j)$ 上的流量。

**定义 1.6**

可行流为满足下列条件的流：

（1）容量限制条件：对每一弧 $(v_i,v_j)\in A, 0\leq f_{ij}\leq c_{ij}$

（2）平衡条件：中间点的流出量与流入量相等，即：

$$\sum_{j:(v_i,v_j)\in A} f_{ij}-\sum_{j:(v_j,v_i)\in A} f_{ji}=0$$

发点 $v_s$：

$$\sum_{(v_s,v_j)\in A} f_{sj}=v$$

收点 $v_t$：

$$\sum_{(v_j,v_t)\in A} f_{jt}=v$$

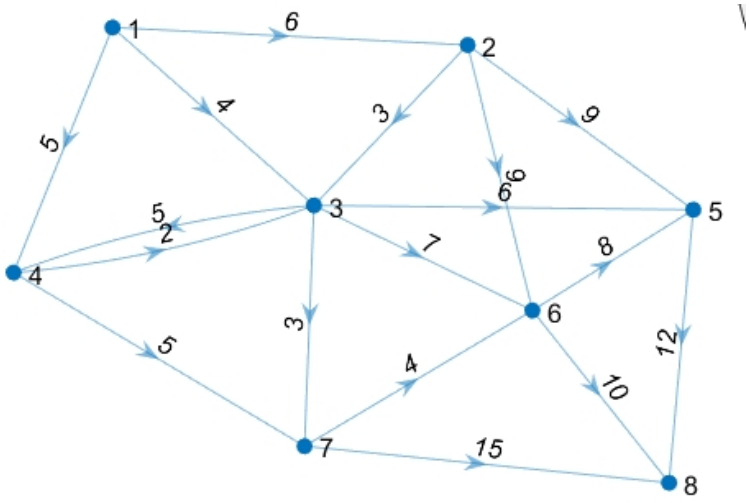
其中，$v$ 称为这个可行流的流量，即发点的净输出量。

最大流问题可以建模为线性规划模型。

$$
\begin{aligned}
&\max v \\
&\text{s.t.} \begin{cases}
\sum\limits_{(v_s,v_j)\in A} f_{sj} = v, \\
\sum\limits_{j:(v_i,v_j)\in A} f_{ij} - \sum_{j:(v_j,v_k)\in A} f_{jk} = 0,\ j \neq s,t \\
\sum\limits_{(v_j,v_t)\in A} f_{jt} = v, \\
0 \leq f_{ij} \leq c_{ij} \\
(v_i, v_j) \in A
\end{cases}
\end{aligned}
$$

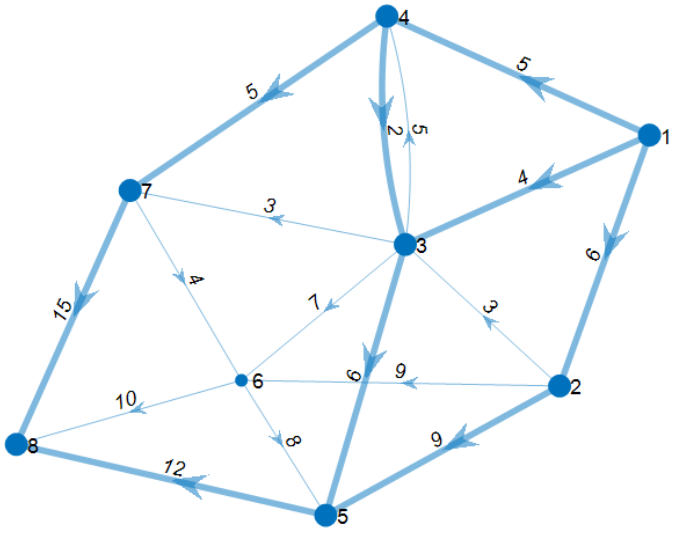
## 1.9 用 MATLAB 求网络最大流

**例题 1.5** 求下图中从 1 到 8 的最大流。




用有向图的方法求解最大流的程序如下。

```
clc
clear
clf
A = zeros(8); A(1,[2:4])=[6,4,5];
A(2,[3,5,6])=[3,9,9]; A(3,[4:7])=[5,6,7,3];
A(4,[3,7])=[2,5]; A(5,8)=12;
A(6,[5,8])=[8,10]; A(7,[6,8])=[4,15];
G = digraph(A);
P = plot(G, 'EdgeLabel', G.Edges.Weight);
[M, F] = maxflow(G,1,8) % 求有向图的最大流
highlight(P, F)       % 显示最大流并画出最大流
```



**最小费用流问题**

在许多实际问题中，往往还要考虑网络上流的费用问题。例如，在运输问题中，人们总是希望在完成运输任务的同时，寻求一个使总的运输费用最小的运输方案。

## 1.10 最小费用流的模型

设 $f_{ij}$ 为弧 $(v_i, v_j)$ 上的流量，$b_{ij}$ 为弧 $(v_i, v_j)$ 上的单位费用，$c_{ij}$ 为弧 $(v_i, v_j)$ 上的容量，则最小费用流问题可以用如下的线性规划问题描述：

$$\min \sum_{(v_i,v_j)\in A} b_{ij} f_{ij}$$

$$\text{s.t.} \begin{cases} \sum\limits_{(v_s,v_j)\in A} f_{sj} = v \\ \sum\limits_{j:(v_i,v_j)\in A} f_{ij} - \sum\limits_{j:(v_j,v_k)\in A} f_{jk} = 0, \quad j \neq s,t \\ \sum\limits_{(v_j,v_t)\in A} f_{jt} = v \\ 0 \le f_{ij} \le c_{ij} \\ (v_i, v_j) \in A \end{cases}$$

若 $v = v_{\max}$ 时，本问题就是最小费用最大流问题。若 $v > v_{\max}$，本问题无解。

### 1.10.1 最小费用流的对偶算法

1961 年，Busacker 和 Gowan 提出了一种求最小费用流的迭代法，其步骤如下。

Step 1. 求出从发点到收点的最小费用通路 $\mu(v_s, v_t)$。

Step 2. 对该通路 $\mu(v_s, v_t)$ 分配最大可能的流量：

$$\bar{f} = \min_{(v_i,v_j)\in\mu(v_s,v_t)} \{c_{ij}\}$$

并让通路上的所有边的容量相应减少 $\bar{f}$。对于通路上的饱和边，其单位流费用相应改为 $\infty$。

Step 3. 作该通路 $\mu(v_s, v_t)$ 上所有边 $(v_i, v_j)$ 的反向边 $(v_j, v_i)$。令 $c_{ji} = \bar{f}$，$b_{ji} = -b_{ij}$。

Step 4. 在新网络中，重复上面的 3 个步骤，直到从发点到收点的全部流量等于 $v$ 为止。