离散数学基础：图论

Fundamentals of Discrete Mathematics: Graph Theory

周晓聪(isszxc@zsu.edu.cn)
中山大学计算机科学系, 广州 510275

2008 年 12 月 26 日

目录

第一章　图的基本概念

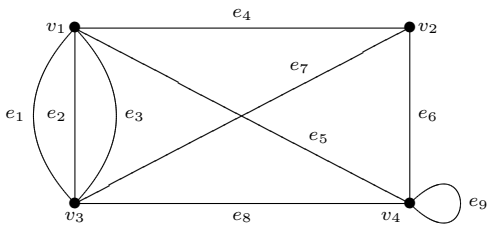
这一章介绍有关图论的一些基本概念，包括无向图（有向图）的定义、顶点与边之间的关系、顶点度数、握手定理、图的道路与回路等。

1.1　图的基本定义

定义 1.1.1 （无向）图(graph)是二元组$G=(V,E)$，其中$V\neq\varnothing$是图G的顶点(vertex)集，其中的元素称为G的顶点(vertex)，E是图G的边(edge)集，其中的元素称为G的边(edge)，且满足，对图G的任意边$e\in E$，都有且仅有两个顶点$u,v\in V$与e关联，称为e的两个端点，通常将e记为$e=(u,v)$或$e=(u,v)$。

对于边$e=(u,v)$，这里u,v没有顺序，因此边$e=(u,v)$和$e=(v,u)$是同一条边。我们只考虑有限图$G=(V,E)$，也即其顶点集V和边集E都是有限集。上面的定义是说，**在定义一个图的时候，要给出它的顶点集和边集，并说明每一条边的两个端点是那两个顶点**。通常，我们可以对图作最为直观的理解，则画出这个图，并用V和E中的元素对这个图进行标记。

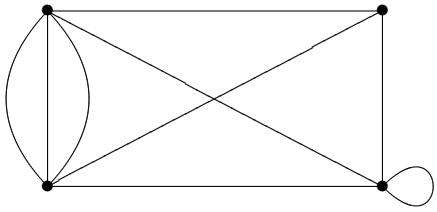
例子 1.1.2 下面是一个图的直观表示：



若将这个图用比较严格的数学形式给出来，则是：图$G=(V,E)$，其中：

$$\begin{aligned} V &= \{\, v_1, v_2, v_3, v_4 \,\} \\ E &= \{\, e_1=(v_1,v_3),\ e_2=(v_1,v_3),\ e_3=(v_1,v_3),\ e_4=(v_1,v_2),\ e_5=(v_1,v_4), \\ &\qquad e_6=(v_2,v_4),\ e_7=(v_2,v_3),\ e_8=(v_3,v_4),\ e_9=(v_4,v_4) \,\} \end{aligned}$$

有时为了简便，我们可能在给出图的直观形式时，没有对其顶点和边进行标记，例如下面是上面的图不带标记的形式：

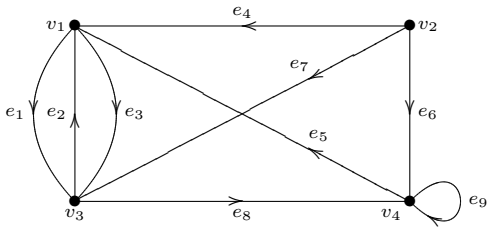


这通常是因为我们这时只关注整个图的性质，不关心图的顶点或边的性质。如果需要关心某些顶点或边的性质时，我们也可能将某些顶点和边标记，而不标记其他顶点和边。

定义 1.1.3 **（有向）图**(directed graph, 或简称digraph)是二元组$G=(V,E)$，其中$V\neq\varnothing$是图D的**顶点**(vertex)集，其中的元素称为G的**顶点**(vertex)，E是图G的**边**(edge)集，其中的元素称为G的**边**(edge)，且满足，对图G的任意边$e\in E$，都有一个顶点$u\in V$是e的**起点**，同时有一个顶点$v\in V$是e的**终点**，通常将e记为$e=\langle u,v\rangle$。

类似，我们也只考虑有限有向图，有向图与无向图的区别，在于有向图中与每一条边相关联的两个顶点区分为起点和终点。

例子 1.1.4 下面是一个有向图的直观表示：



若将这个有向图用比较严格的数学形式给出来，则是：图$G=(V,E)$，其中：

$$\begin{aligned} V &= \{\, v_1, v_2, v_3, v_4 \,\} \\ E &= \{\, e_1=\langle v_1,v_3\rangle,\ e_2=\langle v_3,v_1\rangle,\ e_3=\langle v_1,v_3\rangle,\ e_4=\langle v_2,v_1\rangle,\ e_5=\langle v_4,v_1\rangle, \\ &\qquad e_6=\langle v_2,v_4\rangle,\ e_7=\langle v_2,v_3\rangle,\ e_8=\langle v_3,v_4\rangle,\ e_9=\langle v_4,v_4\rangle \,\} \end{aligned}$$

对于无向图或有向图，还有下面一些简单概念和规定：

1. 通常使用G表示无向图，D表示有向图，但有时用G泛指无向或有向图的，不过D总是表示有向图；

2. 有时使用$V(G),E(G)$分别表示图G的顶点集和边集，$|V(G)|,|E(G)|$分别表示G的顶点数和边数；若$|V(G)|=n$，则称G是n**阶图**；

3. 若$|V(G)|$和$|E(G)|$都是有限数，则称G是有限图，我们只讨论有限图；

4. 对于图G，若$|E(G)|=0$，则称G是**零图**，此时若G有n个顶点，则称G是n阶零图，记为N_n，特别地，称N_1为**平凡图**；

5. 在图的定义中要求顶点集非空，但在图进行运算时，可能使得顶点集为空，因此特别称顶点集为空的图为**空图**；

6. 对于图的直观表示，如果使用了符号标记顶点与边，则称为标定图，否则称为非标定图；

7. 有向图的**基图**是指去掉所有有向边的方向后得到的无向图。

下面如果只说图，则是指无向图。我们下面继续给出一些简单的术语定义：

定义 1.1.5 对于无向图或有向图$G=(V,E)$，
(1) 对于边$e\in E$，若$e=(u,v)$（或$e=\langle u,v\rangle$），则称边e与顶点u,v关联(incident)；
(2) 对于顶点$u,v\in V$，如果存在边$e\in E$，使得$e=(u,v)$（或$e=\langle u,v\rangle$），则称u与v邻接(adjacent)，或说u和v是相邻的顶点；
(3) 对于边$e_1,e_2\in E$，若e_1和e_2关联共同的顶点，则称e_1和e_2邻接，或说e_1和e_2是相邻的边。

这些术语给出了边与边、顶点与顶点之间的邻接（或说相邻）关系，以及顶点与边之间的关联关系。实际上，我们无需纠缠于这些术语的严格定义，直观理解即可，上述定义只是如果在有歧义的情况下，给出一个标准的说法而已。

定义 1.1.6 对于无向图或有向图$G=(V,E)$，
(1) 若边$e=(v,v)$（或$e=\langle v,v\rangle$），即它关联同一个顶点，则称e是环(loop)；
(2) 若边$e=(u,v)$（或$e=\langle u,v\rangle$）且$e'=(u,v)$（或$e'=\langle u,v\rangle$），即e和e'都关联相同的顶点，则称e和e'是重边(parallel edges)。

注意，在有向图中，边$e=\langle u,v\rangle$和$e'=\langle v,u\rangle$并不是重边。下面我们定义图的一些特殊类别：

定义 1.1.7 对于无向图$G=(V,E)$，如果G既没有环也没有重边，则称G是简单图，在后面我们提到图时，如果没有特别声明，都是指简单图。对于有向图$G=(V,E)$，如果G既没有环也没有重边，则称为有向简单图，同样，如果没有特别声明，我们所讲的有向图，都是有向简单图。

备注 1.1.8 在G. Chartrand, Ping Zhang的教材《图论导引》中，所谓的“graph”就是指简单图，而一般的图则采用术语“multigraph”。

定义 1.1.9 对于无向图$G=(V,E)$，如果对V的任意两个顶点u,v，都存在边$e=(u,v)$或$e=(v,u)$，则称G是完全图(complete graph)。具有n个顶点的完全图记为K_n。对于有向图$G=(V,E)$，如果对V的任意两个顶点u,v，都存在边$e=\langle u,v\rangle$及$e'=\langle v,u\rangle$，则称G是有向完全图。

利用子集的概念可定义子图的概念：

定义 1.1.10 给定图$G=(V,E)$，如果图$G'=(V',E')$满足$V'\subseteq V$且$E'\subseteq E$，则称G'是G的子图(subgraph)。特别地，如果$V=V'$，则称G'是G的生成子图(spanning subgraph)。对于G的顶点集V的任意子集$V'\subseteq V$，取$E'=\{e\in E\mid e=(u,v)\text{ 且 }u\in V'\text{ 且 }v\in V'\}$，也即$E'$是$E$中那些两个端点都在$V'$中边构成的集合，则称$G'=(V',E')$是$G$的由$V$导出的子图(induced subgraph)。

上述定义是针对无向图定义的，读者不难对有向图也定义子图、生成子图及导出子图的概念。可在图$G=(V,E)$上定义一些操作，而得到它的一些子图：

定义 1.1.11 对于图$G=(V,E)$，
(1) 删除顶点集：设$V'\subseteq V$，则$G-V'=(V-V',E')$是G的子图，$G-V'$的顶点集是$V-V'$，而$E'=\{e\in E\mid e=(u,v)\text{ 且 }u\notin V'\text{ 且 }v\notin V'\}$，也即$E'$是$E$中那些两个端点都不在$V'$中的边构成的集合；
(2) 删除边集：设$E'\subseteq E$，则$G-E'=(V,E-E')$是G的子图，$G-E'$的顶点集仍是V，而边集是$E-E'$。

(3) 删除子图: 设$G'=(V',E')$是G的子图，则$G-G'=(V,E-E')$;

(4) 边的收缩: 设边$e=(u,v)\in E$，$G\backslash e$表示从G中删除边e后，将e的两个端点u和v用一个新顶点w(或用u或v充当w)代替，使w关联e以外u和v关联的所有边，称为边e的收缩;

(5) 添加新边: 设$u,v\in V(u,v$可能相邻，也可能不相邻，$G\cup(u,v)$或$G+(u,v)$表示在u,v之间加一条边(u,v)，称为添加新边。

注意，$G-V'$不仅要将G中那些在V'中的顶点删除，而且要将这些顶点相关联的边也删除，因为边是不难单独存在的，它一定要与两个顶点关联，而$G-E'$只将G中那些在E'中的边删除，不改变G的顶点。特别地，当$V'=\{v\}$，只有一个顶点时，将$G-V'$记为$G-v$，类似地有$G-e$。对于有向图，也可类似地定义有向图删除顶点集、边集及子图的操作。

定义 1.1.12 对于无向图$G=(V,E)$，设$|V|=n$，即G有n个顶点，则称$\overline{G}=K_n-G$为G的补图。

下面我们定义顶点的度数：

定义 1.1.13 对于无向图$G=(V,E)$，定义顶点$v\in V$的度数(degree)$d(v)=|\{e\in E\mid e=(u,v)$或$e=(v,u)\}|$，也即$d(v)$是$v$所关联的边数，但若$e=(v,v)$是环，则边$e$要计算两次。

对于有向图$G=(V,E)$，定义顶点$v\in V$的入度$d^-(v)=|\{e\in E\mid e=\langle u,v\rangle\}|$，则$d^+v$是以$v$为终点的边数; 定义顶点$v\in V$的出度$d^+(v)=|\{e\in E\mid e=\langle v,u\rangle\}|$，则$d^+v$是以$v$为起点的边数; 定义定点$v\in V$的度数$d(v)=d^-(v)+d^+(v)$。

度数为0的顶点称为孤立顶点，度数为1的顶点称为悬挂顶点，与它关联的边称为悬挂边。很简单地，可证明下面的定理：

定理 1.1.14 握手定理：对于任意的无向图$G=(V,E)$，有：$\sum_{v\in V}d(v)=2|E|$; 对于任意的有向图$G=(V,E)$，有：$\sum_{v\in V}d^-(v)=\sum_{v\in V}d^+(v)=|E|$。 □

后面称度数为奇数的定点为奇度顶点，度数为偶数的顶点为偶度顶点。很容易地，根据握手定理，有如下推论：

推论 1.1.15 任何图（无向图或有向图）的奇度顶点个数是偶数。

例子 1.1.16 很显然，对于n个顶点的完全图K_n，因为每个顶点都与其它$n-1$个顶点有边，因此每个顶点的度数都是$n-1$，从而$\sum d(v)=n(n-1)$，因此K_n的边有$\dfrac{n(n-1)}{2}$条。

例子 1.1.17 （耿素云教材[1]p213习题七第2题）：设9阶无向图G中，每个顶点的度数不是5就是6，证明G中至少有5个6度顶点，或至少有6个5度顶点。

证明 由握手定理，G的度数为5的顶点个数为偶数个，也即可能为0,2,4,6,8个，由于图G的顶点度数不是5就是6,因此相应地，度数为6的顶点就为9,7,5,3,1个，显然这五种情况都表明G中至少有5个6度顶点或至少有6个5度顶点。 □

定义 1.1.18 设$G=(V,E)$是无向图，$V=\{v_1,v_2,\cdots,v_n\}$，称$(d(v_1),d(v_2),\cdots,d(v_n))$为图$G$的度数列(degree sequence)。对于任意给定的非负整数列$\mathbf{d}=(d_1,d_2,\cdots,d_n)$，如果它是某个图的度数列，则称$\mathbf{d}$是可图化的(graphical)；如果它是某个简单图的度数列，则称$\mathbf{d}$是可简单图化的。

很容易证明下面的定理：

定理 1.1.19 非负整数列$\mathbf{d}=(d_1,d_2,\cdots,d_n)$是可图化的当且仅当$(\sum_{i=1}^n d_i) \mod 2=0$。

证明 由握手定理易知，当$\mathbf{d}$是某个图的度数列时有$(\sum_{i=1}^n d_i) \mod 2=0$。反之，若$(\sum_{i=1}^n d_i) \mod 2=0$，则可构造图$G$，其顶点集是$V=\{v_1,v_2,\cdots,v_n\}$，对任意的$1\le i\le n$，若$2k_i\le d_i<2k_i+1$，则在$v_i$上设置$k_i$条环，这样非负整数列$\mathbf{d}'=(d_1-2k_1,d_2-2k_2,\cdots,d_n-2k_n)$中的数要么是0,要么是1,而且由于$(\sum_{i=1}^n d_i) \mod 2=0$，则其中必定有偶数个1,将这些1所对应的顶点成对地连一条边，显然得到的图的度数列就是$\mathbf{d}$。 □

要判断一个非负整数列是否可简单图化则要困难很多，不过下面的定理可以帮助我们将非负整数列进行递减以得到一个简单的、容易判断的非负整数列（下面定理的证明主要参考[2]）：

定理 1.1.20 设$\mathbf{d}=(d_1,d_2,\cdots,d_n)$是非递增的非负整数列，即$d_1\ge d_2\ge\cdots\ge d_n$。不失一般性，假定$d_1\ge 1$。我们有：$\mathbf{d}$是可简单图化的当且仅当非负整数列

$$\mathbf{d}'=(d_2-1,\ d_3-1,\cdots,\ d_{d_1+1}-1,\ d_{d_1+2},\cdots,d_n)$$

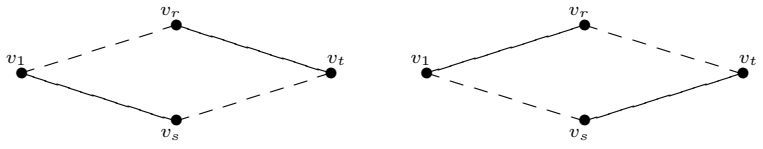
是可简单图化的。

证明 首先，很容易地，当$\mathbf{d}'$是可简单图化的，设简单图G的度数列是$\mathbf{d}'$，则可以新增一个顶点v，使v分别与G中度数为$d_2-1,d_3-1,\cdots,d_{d_1+1}-1$的$d_1$个顶点有边相连，得到的新图$G'$的度数列显然是$\mathbf{d}$，而且$G'$是简单图，因此$\mathbf{d}$是可简单图化的。

反过来的证明要复杂一些。我们证明：在所有度数列为$\mathbf{d}$的图中，必定有一个图G，设其顶点集$V=\{v_1,v_2,\cdots,v_n\}$，$d(v_i)=d_i$，满足：v_1与且只与度数为$d_2,d_3,\cdots,d_{d_1+1}$的d_1个顶点有边相连。

使用反证法：假定所有度数列为$\mathbf{d}$的图都不满足这种性质。我们选一个图G，设其顶点集$V=\{v_1,v_2,\cdots,v_n\}$，$d(v_i)=d_i$，它满足：与v_1相邻的顶点的度数之和最大。我们来证明，如果这个图不满足上面所说的性质，则会导出矛盾！（也即，实际上我们证明了这样的图必满足上面的性质）。

假设选中的图G不满足上面的性质，也即v_1不是与且只与度数为$d_2,d_3,\cdots,d_{d_1+1}$的顶点有边相连，由于$\mathbf{d}$是非递增的，也就是说，存在两个顶点v_r和v_s，$d_r=d(v_r)>d(v_s)=d_s$，v_1与v_r没有边相连，反而与v_s有边相连。由于$d(v_r)>d(v_s)$，因此存在顶点v_t使得v_r与v_t有边相连，而v_s与v_t没有边相连。v_1,v_r,v_s,v_t之间的关系可用下面左边的图表示，其中顶点之间的虚线表示没有边相连：



G G'

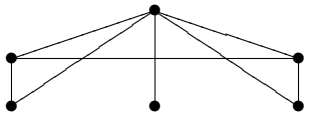
这样我们可将图G进行改变，删除边$(v_1,v_s),(v_r,v_t)$，而增加新边$(v_1,v_r),(v_s,v_t)$，得到图G'。显然G'的度数列仍为$\mathbf{d}$，但G'中与v_1相邻的顶点度数之和却比G大（因为$d(v_r)>d(v_s)$），这与G是与v_1相邻的顶点度数之和最大矛盾！

这就证明了，度数列为$\mathbf{d}$的图中，必定有一个图G，设其顶点集$V=\{v_1,v_2,\cdots,v_n\}$，$d(v_i)=d_i$，G满足：v_1与且只与度数为$d_2,d_3,\cdots,d_{d_1+1}$的d_1个顶点有边相连，这样删除v_1及其相邻的边得到的图的度数列就是$\mathbf{d}'$，因此$\mathbf{d}'$也是可简单图化的。 □

例子 1.1.21 耿素云教材[1]p213习题七第7题：

1. 整数列$(6,6,5,5,3,3,2)$是可简单图化的当且仅当$(5,4,4,2,2,1)$是可简单图化的，当且仅当$(3,3,1,1,0)$是可简单图化的，当且仅当$(2,0,0,0)$是可简单图化，而$(2,0,0,0)$显然不可简单图化，因此整数列$(6,6,5,5,3,3,2)$不是可简单图化的。

2. 整数列$(5,3,3,2,2,1)$是可简单图化的当且仅当$(2,2,1,1,0)$是可简单图化的，当且仅当$(1,0,1,0)$是可简单图化的，显然$(1,0,1,0)$是可简单图化的，因此整数列$(5,3,3,2,2,1)$是可简单图化的。下面的简单图的度数列就是$(5,3,3,2,2,1)$（但好像非同构的简单图就只有下面这一个图!）：



3. 整数列$(3,3,2,2,2,2)$是可简单图化的，当且仅当$(2,1,1,1,2)$是可简单图化的，当且仅当$(2,2,1,1,1)$是可简单图化的，当且仅当$(1,0,1,1)$是可简单图化的，而$(1,0,1,1)$显然是不可简单图化的，因此整数列$(3,3,2,2,2,2)$不是可简单图化的。

定义 1.1.22 给定图$G=(V,E)$，定义图G的最小度$\delta(G)=\min\{d(v)\mid v\in V\}$，即$\delta(G)$是度数最小的顶点的度数；定义图$G$的最大度$\Delta(G)=\max\{d(v)\mid v\in V\}$，即$\Delta(G)$是度数最大的顶点的度数。

引理 1.1.23 若G是简单图，则$\Delta(G)\leq n-1$。

定义 1.1.24 若图$G=(V,E)$任意顶点的度数都等于k，则称G为k正则图(regular graph)。

最后，我们定义图的同构这个概念：

定义 1.1.25 对于图$G=(V,E)$和$G'=(V',E')$，如果存在双函数$f:V\to V'$，满足：对E的任意边$e=(u,v)$，E'存在惟一的边$e'=(f(u),f(v))$，而且对E'的任意边$e'=(u',v')$，E存在惟一的边$e=(f^{-1}(u'),f^{-1}(v'))$，则称$G$和$G'$同构，记为$G\sim G'$。

定义 1.1.26 如果图$G=(V,E)$与它的补图$\overline{G}$同构，则称G是自补图(self-complementary graph)。

由于目前尚没有发现两个图同构的充分必要条件，也没有发现判断两个图是否同构的有效算法，因此我们只能直观地理解两个图同构的含义，即在不考虑图的标记的情况，这两个图在适当移动顶点和边之后有相同的图形表示，或者说在不考虑图的标记的情况下，这两个图的顶点和边之间的关联情况完全一致。

1.2 道路与回路

定义 1.2.1 给定无向图$G=(V,E)$，G中的一条道路（或说路、通路、链等）$\Gamma=v_0e_1v_1e_2\cdots e_nv_n$是$G$中顶点和边的序列，且满足，对任意的$1\leq i\leq n$有$e_i=(v_{i-1},v_i)$（或$e_i=(v_i,v_{i-1})$），这时称$v_0$和$v_n$是$\Gamma$的两个端点，$n$是$\Gamma$的长度。如果还满足$v_0=v_n$，则称$\Gamma$是回路（或说圈）。

定义 1.2.2 给定有向图$G=(V,E)$，G中的一条有向道路$\Gamma=v_0e_1v_1e_2\cdots e_nv_n$是$G$中顶点和边的序列，且满足，对任意的$1\leq i\leq n$有$e_i=\langle v_{i-1},v_i\rangle$，这时称$v_0$是$\Gamma$的起点，而$v_n$是$\Gamma$的终点，$n$是$\Gamma$的长度。如果还满足$v_0=v_n$，则称$\Gamma$是有向回路。

定义 1.2.3 给定无向图$G=(V,E)$，Γ是G的一条道路（回路），如果Γ中不存在重复的边，则称Γ是简单道路（简单回路）；如果除Γ的两个端点可能相同之外没有重复的顶点，则称Γ是初级道路（初级回路），初级道路也称为路径，初级回路也称为圈。类似地可定义有向简单道路、有向简单回路、有向初级道路、有向初级回路。

后面为方便起见，我们用记号$v\in\Gamma$表示顶点v出现在Γ的顶点与边的序列中，类似地用$e\in\Gamma$表示边e在Γ的顶点与边序列中。

对于道路$\Gamma=v_0e_1v_1e_2\cdots e_nv_n$，由于边的两个端点在图中是固定的，因此我们可以只用边的序列$e_1e_2\cdots e_n$表示道路Γ。进一步，在简单图中，由于顶点之间最多有一条边，因此可只用顶点的序列$v_0v_1\cdots v_n$表示道路Γ。很容易证明图中的道路（回路）的如下性质：

引理 1.2.4 给定图$G=(V,E)$有n个顶点，若存在以u和v为端点的道路，则存在以u和v为端点的长度小于等于$n-1$的初级道路。

推论 1.2.5 给定图$G=(V,E)$有n个顶点，若存在从u到其自身的回路，则存在从u到其自身的长度小于等于n的初级回路。

在图论的证明中，很多时候需要利用极长道路的概念：

定义 1.2.6 给定无向图$G=(V,E)$，称G中的一条简单道路$\Gamma=v_0e_1v_1e_2\cdots e_nv_n$是极长道路，如果不存在顶点$v\in V$，$v\neq v_i(0\leq i\leq n)$使得$v$与$\Gamma$的端点$v_0$或$v_n$邻接。进一步，若极长道路$\Gamma$又是初级道路，则称为极长初级道路。

也就是说，极长道路Γ的两个端点只与在Γ中的顶点相邻。注意，只有在简单道路（则不重复边）的意义下才有所谓的极长道路。如果允许重复边，则任意一条边$e=(u,v)$都可看作是一条无穷长的道路$\Gamma=ueveue\cdots$。

例子 1.2.7 若简单图G的最小度$\delta(G)\geq 2$，则G必含有回路。

证明 设$\Gamma=v_0e_1v_1e_2\cdots e_nv_n$是$G$中一条极长道路，由于$d(v_0)\geq 2$，因此除$v_1$之外，还有$v_i,\in\Gamma$使得$v_0$与$v_i$相邻，则$\Gamma'=v_0v_1\cdots v_iv_0$是回路。 □

例子 1.2.8 若Γ是简单图$G=(V,E)$的中含顶点数大于3的初级回路，如果存在边$e=(u,v)\in E$满足$u,v\in\Gamma$而$e\notin\Gamma$，则称e是Γ的弦，而Γ是带弦回路。证明若简单图$G=(V,E)$的最小度$\delta(G)\geq 3$，则G必存在带弦的回路。

证明 设$\Gamma=v_0e_1v_1e_2\cdots e_nv_n$是$G$中一条极长初级道路，由于$d(v_0)\geq 3$，因此除$v_1$之外，还有$v_i,v_j\in\Gamma$使得$v_0$与$v_i$和$v_j$相邻，不妨假设$i<j$，则$\Gamma'=v_0v_1\cdots v_i\cdots v_jv_0$是一条初级回路，而边$e=(v_0,v_i)$是该回路的弦。 □

例子 1.2.9 给定$G=(V,E)$是简单图，$|V|=n\geq 4$，$|E|=m\geq 2n-3$，证明G有带弦回路。

证明 用数学归纳法，不难看到当$n=4$时，命题成立；假设命题对于$n=k$时成立，也即对任意的图，如果其顶点数是k，而边数大于等于$2k-3$，则它含有带弦回路。考虑$n=k+1$，即考虑对于顶点数是$k+1$，边数大于等于$2(k+1)-3$的图G。设Γ是G的一条极长初级道路，v是Γ的一个端点，如果$d(v)\geq 3$，则由上面的例子可得G有带弦回路，否则（即$d(v)<3$）不难看到$G-v$是有k个顶点，边数大于等于$2(k+1)-3-2=2k-3$的图，根据归纳假设，$G-v$含有带弦回路，那么G也必含有带弦回路。 □

1.3 图的连通性

定义 1.3.1 给定无向图$G=(V,E)$，顶点$u,v\in V$，称u和v之间有道路，或称u和v是**可达**的，如果G存在以u和v为端点的道路。进一步，我们约定u和u自己是可达的。

在V上定义关系$\leftrightarrow\subseteq V\times V$，$u\leftrightarrow v$当且仅当$u$和$v$是可达的，则$\leftrightarrow$是等价关系，称$V$关于$\leftrightarrow$的等价类$[u]_{\leftrightarrow}\subseteq V$导出的子图为$G$的**连通分支**，我们将$G$的连通分支数记为$p(G)$。若$p(G)=1$，则称$G$是**连通图**(connected graph)。

引理 1.3.2 给定简单图$G=(V,E)$，$|V|=n,|E|=m$，若G是连通的，则$m\geq n-1$。

证明 使用第二数学归纳法证明：当$n=1$时，显然$m\leq 0$，类似地，当$n=2$时，要使得G连通至少要有一条边。设$n<k$时成立，即任意少于k个顶点的简单连通图的边数大于等于顶点数减一。

考虑任意含有k个顶点的简单连通图G。设v是G的任意顶点，记$p(G-v)=j$，$G-v$的每个连通分支的顶点数分别是$n_1,n_2,\cdots,n_j$，显然对任意的$1\leq i\leq j$有$1\leq n_j<k$，因此$G-v$的每个连通分支都是少于k个顶点的连通简单图，根据归纳假设，$G-v$至少含有

$$\sum_{i=1}^{j}n_j-1=k-1-j$$

条边。而为保持G是连通图，v与$G-v$的每个连通分支都至少有一条边相连，因此G至少含有$k-1-j+j=k-1$条边。 □

这个引理给出了连通简单图的边数的一个下界，也即使得简单图连通的在边数方面的一个必要条件。显然简单图的边数的上界在完全图达到，利用这个上界，我们可以证明使得简单图连通的一个边数方面的充分条件：

例子 1.3.3 给定简单图$G=(V,E)$，$|V|=n,|E|=m$，若$m>\frac{1}{2}(n-1)(n-2)$，则G连通。

证明 反证法：若G不连通，则$k=p(G)\geq 2$，设其k个连通分支的顶点数分别为$n_1,n_2,\cdots,n_k$，则对任意的$1\leq i\leq k$有$1\leq n_i\leq n-1$，且$\sum_{i=1}^{k}n_i=n$。当G的每个连通分支都是完全图时边数最大，因此G最多有

$$\sum_{i=1}^{k}\frac{n_i(n_i-1)}{2}\leq\sum_{i=1}^{k}\frac{(n-1)(n_i-1)}{2}=\frac{n-1}{2}\sum_{i=1}^{k}(n_i-1)=\frac{(n-1)(n-k)}{2}$$

若$k\geq 2$，则与$m>\frac{1}{2}(n-1)(n-2)$矛盾，因此G只有一个连通分支，即G是连通图。 □

下面从顶点度数的角度给出了简单图连通的一个充分条件：

引理 1.3.4 给定简单图$G=(V,E)$，$|V|=n$，若对任意的两个顶点$u,v\in V$都有$d(u)+d(v)\geq n-1$，则G连通。

证明 反证法：若G不连通，则至少有两个连通分支，从其中两个连通分支中分别取顶点u和v，设这两个连通分支的顶点数分别为n_u和n_v，则$d(u)<n_u$且$d(v)<n_v$，从而$d(u)+d(v)<n_u+n_v\leq n$，这与$d(u)+d(v)\geq n-1$矛盾，因此G是连通图。 □

有向图的连通性可类似定义：

定义 1.3.5 给定有向图$G=(V,E)$，对G的两个顶点$u,v\in V$，若存在以u为起点，v为终点的有向道路，则称u**可达**v，记为$u\to v$。进一步记$u\leftrightarrow v$为$u\to v$且$v\to u$。约定总有$u\leftrightarrow u$。

若对G的任意两个顶点u,v都有$u\leftrightarrow v$，则称G是**强连通**的；若有$u\to v$或$v\to u$之一成立，则称G是**单向连通**的；若不考虑有向图G中边的方向时是（无向）连通图，则称G是**弱连通**的。

有些连通图当删除若干个顶点或若干条边之后就不连通了，从某种意义上说，能够删除多少顶点或边使得图不连通，从另一个角度刻画了图的连通程度。

定义 1.3.6 给定无向图$G=(V,E)$（不一定是简单图），$V'\subset V$是顶点集V的真子集，若V'满足：(i) $p(G-V')>p(G)$，即从G中删除V'中的顶点（及其相关联的边）之后的图的连通分支数比G大；(ii) 对V'的真子集$V''\subset V'$，都有$p(G-V'')=p(G-V')$，即只删除V'中某些顶点（及关联的边）不会再增加图的连通分支数；则称V'是图的**点割集**。若点割集$V'=\{v\}$只有一个顶点，则称v是图G的**割点**。

定义 1.3.7 给定无向图$G=(V,E)$（不一定是简单图），$|V|=n$，设G连通且不含完全图K_n为子图，称：

$$\kappa(G)=\min\{|V'|\mid V'\text{是}G\text{的点割集}\}$$

为G的**点连通度**，简称**连通度**。规定完全图K_n的点连通度为$n-1$，非连通图的点连通度为0。若$\kappa(G)\geq k$，则称图G是k连通图。

实际上，点连通度就是顶点数最少的点割集的顶点数目，显然若图有割点的话，则其点连通图就是1。若G是k连通图，则从G中任意删除k个顶点之后，图仍然是连通的。也可从边的角度刻画图的连通程度：

定义 1.3.8 给定无向图$G=(V,E)$（不一定是简单图），$E'\subset E$是顶点集V的真子集，若E'满足：(i) $p(G-E')>p(G)$，即从G中删除E'中的边之后的图的连通分支数比G大；(ii) 对E'的真子集$E''\subset E'$，都有$p(G-E'')=p(G-E')$，即只删除E'中某些边不会再增加图的连通分支数；则称E'是图的**边割集**。若边割集$E'=\{e\}$只有一条边，则称e是图G的**割边**，或称为**桥**。

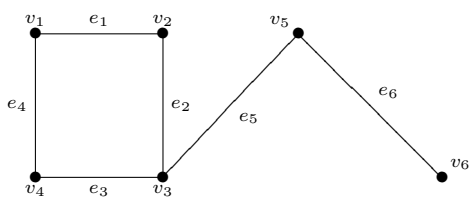
定义 1.3.9 给定无向图$G=(V,E)$（不一定是简单图），$|V|=n$，设G连通，称：

$$\lambda(G)=\min\{|E'|\mid E'\text{是}G\text{的边割集}\}$$

为G的**边连通度**，简称**连通度**。规定非连通图的点连通度为0。若$\lambda(G)\geq r$，则称图G是r边连通图。

类似地，若G是r边连通图，则从G中任意删除r条边之后，它仍然是连通的。

例子 1.3.10 考虑下面的图：



显然$\{v_2,v_4\},\{v_3\},\{v_5\}$都是点割集，其中$v_3$和$v_5$都是割点，注意$v_1$和$v_6$都不在任何点割集中，因此该图的点连通度是1。

另一方面，$\{e_6\},\{e_5\},\{e_2,e_3\},\{e_1,e_2\},\{e_3,e_4\},\{e_1,e_3\},\{e_2,e_4\}$都是边割集，其中$e_6$和$e_5$都是桥，显然该图的边连通度是1。

由于删除顶点要同时删除其关联的边，因此不难证明如下定理：

定理 1.3.11 设G是连通图，则：$\kappa(G)\le\lambda(G)\le\delta(G)$。

证明 设v是G中度数最少的顶点，删除与v相关联的边之后，图就不连通，因此$\lambda(G)\le\delta(G)$。下面再证$\kappa(G)\le\lambda(G)$。设$E'=\{e_1,e_2,\cdots,e_\lambda\}$是$G$的一个最小边割集，则$G-E'$至少有两个不连通的顶点集$V_1$和$V_2$。这时再考虑图$G$，若将$V_1$中与$e_1,e_2,\cdots,e_{\lambda-1}$相关联的至多$\lambda-1$个顶点删除（注意这也删除了边$e_1,e_2,\cdots,e_{\lambda-1}$），又将$V_2$中与$e_\lambda$相关联的一个顶点删除（这也删除了边$e_\lambda$），显然我们至多删除了$\lambda(G)$个顶点，但得到的图同样不连通，因此$\kappa(G)\le\lambda(G)$。 □

练习 1.3.12 对照耿素云教材[1]知，**上述证明是有问题的**，请读者思考一下，上述简单证明有什么问题？

耿素云教材[1]使用一些定理进一步讨论了边连通度、点连通度和图的最小度之间的联系，由于我尚未发现这些定理给出的联系的一些直观的解释，因此这里不对这些定理进行讨论。下面就割点和桥的一些性质进行简单的讨论。

定理 1.3.13 设$G=(V,E)$，$v\in V$，v是G的割点当且仅当存在$V-v$的一个划分$\{V_1,V_2\}$，使得对任意的$u\in V_1,w\in V_2$，v在每一条u到w的路径上。

证明 若v是G的割点，则$G-v$至少有两个连通分支，假设G_1是其中的一个连通分支，V_1是G_1的顶点集，令$V_2=V-V_1-\{v\}$，则$\{V_1,V_2\}$是$V-v$的一个划分，而且对任意的顶点$u\in V_1,w\in V_2$，u,w处在$G-v$的不同的连通分支，从而从u到w的每条路径必经过v，否则如果它们之间存在不经过v的路径P，则P在$G-v$中，这与u,w处在$G-v$的不同连通分支矛盾！

反之，若存在$G-v$的划分$\{V_1,V_2\}$，使得对任意的$u\in V_1,w\in V_2$，v在每一条u到w的路径上。假设这时v不是割点，则$G-v$连通，根据划分的要求$V_1\neq\varnothing,V_2\neq\varnothing$，因此存在顶点$u\in V_1,w\in V_2$，$u$与$w$连通，即这时它们存在不经过$v$的路径，与前提条件矛盾！ □

推论 1.3.14 设v为无向连通图G的一个顶点，v为割点当且仅当存在与v不同的两个顶点u和w，使得v处在每一条从u到w的路径上。

证明 若v是G的割点，则根据上一定理，存在$V-v$的一个划分$\{V_1,V_2\}$，使得对任意的$u\in V_1,w\in V_2$，v在每一条u到w的路径上，而由划分的要求，V_1和V_2不空，因此确实存在$u\in V_1,w\in V_2$，而且u,w与v不同，使得v处在每一条从u到w的路径上。

反之，若存在与v不同的两个顶点u和w，使得v处在每一条从u到w的路径上。假设这时v不是割点，则$G-v$连通，即u与w连通，从而它们之间存在不经过v的路径，这与前提条件矛盾！ □

定理 1.3.15 设e为无向连通图G的一条边，e为桥当且仅当e不在G的任意圈上。

证明 设$e=(u,v)$，若e是桥，我们证明它不可能在G的某个圈上。否则，设G中存在圈$v_0e_0v_1\cdots uev\cdots v_ne_nv_0$，则对$G$中的任意两个定点$w,w'$，若$w$到$w'$存在不经过$e$的路径，则$w$与$w'$在$G-e$中仍连通，否则，若$w$到$w'$的每条路径都经过$e$，则在$G-e$中，$w$到$w'$存在经过$u\cdots v_1e_0v_0\cdots v_ne_nv_0$的路径，从而也是连通的，即这时$G-e$总是连通的，从而与$e$是桥矛盾！

反之，若e不在G的任意圈上，而e不是桥，即$G-e$连通，从而在$G-e$中存在u到v的路径，那么加上e就得到G的一个含有e的圈，与e不在G的任意圈上矛盾！ □

定理 1.3.16 设e为无向连通图$G=(V,E)$的一条边，e为桥当且仅当存在V的一个划分$\{V_1,V_2\}$，使得对任意的$u\in V_1,w\in V_2$，e在每一条u到w的路径上。

证明 设e是桥，则这时$G-e$有且只有两个连通分支G_1,G_2，设它们的顶点集分别是V_1,V_2，则$\{V_1,V_2\}$就是V的一个划分，而且对任意的$u\in V_1,w\in V_2$，e在u到w的每一条路径上，否则若存在一条不经过e的路径，则u和w在$G-e$中连通，这与它们处在$G-e$的不同连通分支矛盾。

反之，设存在V的一个划分$\{V_1,V_2\}$，使得对任意的$u\in V_1,w\in V_2$，e在每一条u到w的路径上。若这时e不是桥，则$G-e$仍连通，则存在$u\in V_1,w\in V_2$（划分块非空集），且u到w之间在$G-e$中存在路径，该路径不经过e，矛盾！ □

下面的定理对于判断有向图的强连通性非常有用。

定理 1.3.17 设D是有向图，D强连通当且仅当D存在经过D中每个顶点至少一次的回路。

证明 显然，当D存在这样的回路时，D是强连通的。反之，当D强连通时，设D的顶点集是$V=\{v_1,\cdots,v_n\}$，由于D是强连通的，从而对任意的$i=1,2,\cdots,n-1$，v_i到v_{i+1}有道路，设为Γ_i。同样地，v_n到v_1也有道路，设为Γ_n，这样$\Gamma_1,\cdots,\Gamma_{n-1},\Gamma_n$就是一条经过$D$中每个顶点至少一次的回路。 □

下面的引理给出单向连通的有向图的一个重要性质：

引理 1.3.18 设$D=(V,E)$是单向连通的有向图，则对V的任意非空子集$V'\subseteq V$，都存在顶点$u\in V'$，使得对任意的$v\in V'$，都有从u到达v的有向道路。

证明 使用反证法。假设D不满足上述所说性质，则D必存在不满足此性质的顶点子集。设$V_1=\{v_1,v_2,\cdots,v_k\}\subseteq V$是不满足上述性质的极小顶点集，显然$|V_1|\geq 2$。令$V_2=V_1-\{v_k\}$，则$V_2$非空，且满足上述性质，也即存在$u\in V_2$，对任意的$v\in V_2$，都有从$u$到达$v$的有向道路。从而，对于$V_1$而言，它不满足上述性质就是因为不存在$u$到$v_k$的道路。

但我们能证明这时也不存在从v_k到u的道路，因为若存在v_k到u的道路，则由于u到V_1的其他顶点都有道路，从而v_k到V_1的所有顶点就都有道路，这与V_1不满足上述性质矛盾！但若既不存在u到v_k的道路，又不存在v_k到u的道路，则与D是单向连通的有向图矛盾！因此D不存在满足上述性质顶点子集，也即引理成立。 □

根据上述引理，可以给出判断有向图单向连通的一种方法：

定理 1.3.19 设D是有向图，D单向连通当且仅当D存在经过D中每个顶点至少一次的通路。

证明 设$D=(V,E)$，且$|V|=n$。显然当D存在经过D中每个顶点至少一次的通路时，D是单向连通的。反之，若D是单向连通的，根据上面的引理，D存在顶点v_1，v_1可到达其他所有的顶点。类似地，$V-\{v_1\}$存在顶点v_2可到达$V-\{v_1\}$中的所有顶点，等等，$V-\{v_1,v_2,\cdots,v_{n-2}\}$中存在顶点$v_{n-1}$可到达$v_n$，从而$v_1v_2v_3\cdots v_{n-1}v_n$是经过$D$中每个顶点至少一次的通路。 □

1.4 邻接矩阵与可达矩阵

在对图进行描述或运算时，常常要采用图的矩阵表示。图的矩阵表示有两种，一种是邻接矩阵，一种是关联矩阵。邻接矩阵表示顶点之间的邻接关系，而关联矩阵表示顶点与边之间的关联关系。这一节只介绍邻接矩阵，关联矩阵将在树的一章介绍。

定义 1.4.1 给定无向图$G=(V,E)$无多重边，$V=\{v_1,v_2,\cdots,v_n\}$。无向图G的邻接矩阵A是一个n阶方阵$A=[a_{ij}]_{n\times n}$，定义为：

$$a_{ij}=\begin{cases}1 & \text{若 } (v_i,v_j)\in E\\ 0 & \text{否则}\end{cases}$$

显然无向图的邻接矩阵是一个对称矩阵，它的行的非零元素个数是该行对应顶点的度数。相比而言，对于邻接矩阵，我们主要关心有向图的邻接矩阵。

定义 1.4.2 给定有向图$G=(V,E)$无多重边，$V=\{v_1,v_2,\cdots,v_n\}$。有向图G的邻接矩阵A是一个n阶方阵$A=[a_{ij}]_{n\times n}$，定义为：

$$a_{ij}=\begin{cases}1 & \text{若 } \langle v_i,v_j\rangle\in E\\ 0 & \text{否则}\end{cases}$$

有向图邻接矩阵的行的非零元素个数是该行对应顶点的出度，列的非零元素个数是该列对应顶点的入度。进一步，有如下定理：

定理 1.4.3 设A是n阶有向标定图的邻接矩阵，A的l次幂$(l\geq 1)$$A^l$中的元素$a_{ij}^{(l)}$是$v_i$到$v_j$长度为$l$的通路数，$\sum_i\sum_j a_{ij}^{(l)}$是$D$中长度为$l$的通路总数，而$\sum_i a_{ii}^{(l)}$是$D$中长度为$l$的回路总数。

证明 对长度l实施数学归纳法。 □

注意，这里说的通路是指一般的通路，即可能存在环和重复边。而对于回路总数，则当起点和终点不同时，被看作不同的回路，例如设有回路$v_0v_1v_2\cdots v_nv_0$，则它既是v_0到v_0的回路，也是v_i到$v_i(i=1,\cdots,n)$的回路，因此在回路总数中被计为$n+1$条回路。进一步，令

$$B_r=A+A^2+\cdots+A^r=\left[b_{ij}^{(r)}\right]_{n\times n},\quad r=1,2,\cdots$$

则有推论：

推论 1.4.4 设A是n阶有向标定图的邻接矩阵，B_r的定义如上，则B_r中的元素$b_{ij}^{(r)}$是v_i到v_j长度小于等于r的通路数，$\sum_i\sum_j b_{ij}^{(r)}$是$D$中长度小于等于为$r$的通路总数，而$\sum_i b_{ii}^{(r)}$是$D$中长度小于等于$r$的回路总数。 □

如果我们只关心v_i是否可达v_j，则可在矩阵乘法中使用逻辑运算，即该用实数乘法时使用逻辑与（或称为逻辑乘），该用实数加法时使用逻辑或（或说逻辑加）。我们定义有向图的可达矩阵：

定义 1.4.5 给定有向图$G=(V,E)$，$V=\{v_1,v_2,\cdots,v_n\}$。有向图G的可达矩阵P是一个n阶方阵$P=[p_{ij}]_{n\times n}$，定义为:

$$p_{ij}=\begin{cases}1 & \text{若}v_i\text{可达}v_j\\ 0 & \text{否则}\end{cases}$$

根据定义，显然对任意的$1\le i\le n$，有$p_{ii}=1$，而对任意的$i\ne j$，则$p_{ij}=1$当且仅当$b_{ij}^{(n-1)}\ne 0$。利用邻接矩阵使用矩阵乘法计算可达矩阵虽然可行，但是计算起来比较复杂，因为每计算一次矩阵乘法，则需要做n^3次逻辑乘，而且要计算到A^{n-1}，因此需要做$n^3(n-1)$次逻辑乘。一个更有效的计算方法是Warshall算法，其算法步骤如下：设A是n阶有向图的邻接矩阵，要计算它的可达矩阵P，

(1). $P\leftarrow A$

(2). $k\leftarrow 1$

(3). $i\leftarrow 1$

(4). $p_{ij}\leftarrow p_{ij}\ \vee\ (p_{ik}\ \wedge\ p_{kj}),\quad j=1,2,\cdots,n$

(5). $i\leftarrow i+1$，若 $i\le n$ 转(4)

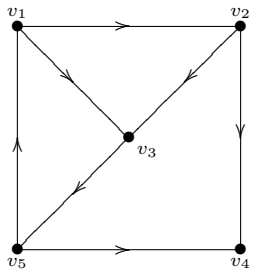
(6). $k\leftarrow k+1$，若 $k\le n$ 转(3)，否则停止

实际上，Warshall算法的基本思想是：

第k次循环计算的$p_{ij}^{(k)}$是顶点v_i是否能经过$v_1,v_2,\cdots v_k$中的某些顶点到达v_j。从而第n次循环时，$p_{ij}^{(n)}=1$表示顶点v_i可达v_j。

从而对于矩阵$P^{(k)}$，若$p_{ij}^{(k)}=1$，当且仅当，v_i可直接到达v_j，或者v_i经过$v_1,v_2,\cdots,v_k$中的某些顶点可到达v_j。下面使用例子说明Warshall 算法计算可达矩阵的基本思想。

例子 1.4.6 戴一奇教材[3]p13的例2.2.1。考虑下面的图：



首先

$$A=\begin{bmatrix}0&1&1&0&0\\0&0&1&1&0\\0&0&0&0&1\\0&0&0&0&0\\1&0&0&1&0\end{bmatrix}$$

P初始化为A，当$k=1$时，实际上是看$p_{i1}, i=1,\cdots,n$，若$p_{i1}=1$，则1行的值与第i行的值进行逻辑加。对于上述矩阵，只有$p_{51}=1$，因此将第1行的值与第5行的值进行逻辑加，得到$P^{(1)}$：

$$P^{(1)}=\begin{bmatrix}0&1&1&0&0\\0&0&1&1&0\\0&0&0&0&1\\0&0&0&0&0\\1&\mathbf{1}&\mathbf{1}&1&0\end{bmatrix}$$

矩阵$P^{(1)}$与它的初值A不同的就是p_{52}和p_{53}，从原来的0变成了1，实际上这意味着，v_5可以经过v_1分别到达v_2和v_3，而其他的顶点都不能通过v_1到达另外一个顶点，因此值就没有改变，从前面的图看也确实如此。

当$k=2$时，则看$p_{i2}, i=1,\cdots,n$，若$p_{i2}=1$，则将第2行的值与第i行的值进行逻辑加。对于矩阵$P^{(1)}$，有$p_{12}=1$和$p_{52}=1$，因此将第2行的值与第1行的值进行逻辑加，将第2行的值与第5行的值进行逻辑加，得到的结果如下：

$$P^{(2)}=\begin{bmatrix}0&1&1&\mathbf{1}&0\\0&0&1&1&0\\0&0&0&0&1\\0&0&0&0&0\\1&1&1&1&0\end{bmatrix}$$

这次影响到的值包括p_{13}, p_{14}, p_{53}和p_{54}，表明v_1可经过v_1, v_2到达v_3和v_4，v_5可经过v_1, v_2到达v_3和v_4。矩阵$P^{(2)}$和$P^{(1)}$中值不同的是p_{14}，这表明v_1不可直接到达v_4，也不能经过v_1到达v_4，只能通过v_1, v_2到达v_4。

依次计算，可得到Warshall算法的计算结果如下，上面将每一次循环对原来矩阵作了改变的数字用了黑体字标出：

$$A=\begin{bmatrix}0&1&1&0&0\\0&0&1&1&0\\0&0&0&0&1\\0&0&0&0&0\\1&0&0&1&0\end{bmatrix}\quad P^{(1)}=\begin{bmatrix}0&1&1&0&0\\0&0&1&1&0\\0&0&0&0&1\\0&0&0&0&0\\1&\mathbf{1}&\mathbf{1}&1&0\end{bmatrix}\quad P^{(2)}=\begin{bmatrix}0&1&1&\mathbf{1}&0\\0&0&1&1&0\\0&0&0&0&1\\0&0&0&0&0\\1&1&1&1&0\end{bmatrix}$$

$$P^{(3)}=\begin{bmatrix}0&1&1&1&\mathbf{1}\\0&0&1&1&\mathbf{1}\\0&0&0&0&1\\0&0&0&0&0\\1&1&1&1&\mathbf{1}\end{bmatrix}\quad P^{(4)}=P^{(3)}\quad P^{(5)}=\begin{bmatrix}\mathbf{1}&1&1&1&1\\\mathbf{1}&\mathbf{1}&1&1&1\\\mathbf{1}&\mathbf{1}&\mathbf{1}&\mathbf{1}&1\\0&0&0&0&0\\1&1&1&1&\mathbf{1}\end{bmatrix}$$

计算结果表明，除了v_4不能到达其他顶点之外（这很显然，因为v_4的出度为0），其他顶点都是相互可达的，而且任意顶点也可到达v_4，因此这个图是单向连通的，但不是强连通的。如果可达矩阵的所有元素都为1，则相应的有向图是强连通的。

备注 1.4.7 实际上，当有向图不存在多重边时，有向图与顶点集上的关系是一一对应的，这时有向图的可达矩阵实际上给出了该关系的传递闭包。

作 业

作业 1.1 设无向图$G=(V,E)$，$V=\{v_1,v_2,\cdots,v_6\}$，

$$E=\{(v_1,v_2),(v_2,v_2),(v_2,v_4),(v_4,v_5),(v_3,v_4),(v_1,v_3),(v_3,v_1)\}$$

(1) 画出G的图形；
(2) 求出G中各顶点的度及奇数度顶点的个数；
(3) 指出G中的平行边、环、孤立点、悬挂边及悬挂顶点。G是简单图吗？

作业 1.2 下面的非负整数列，哪些是可简单图化的？如果是可简单图化的，请给出一个简单图使得其度数列是该整数列：
(1) (1, 3, 3, 4, 5, 6, 6)
(2) (0, 1, 1, 2, 3, 3)
(3) (3, 3, 3, 3)
(4) (2, 3, 3, 4, 5, 6)

作业 1.3 设无向图G有12条边，已知G的3度顶点有6个，而其他顶点的度数都小于3，问G至少有多少个顶点？为什么？

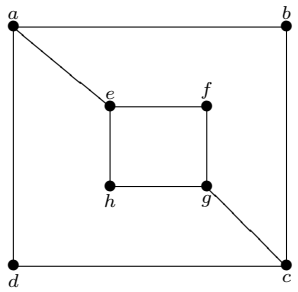
作业 1.4 设n阶图G有m条边，证明

$$\delta(G)\leq\frac{2m}{n}\leq\Delta(G)$$

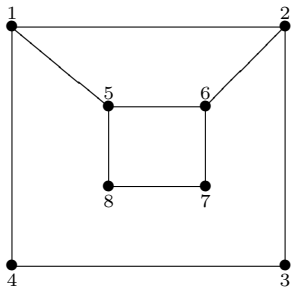
作业 1.5 设G是n个顶点m条边的简单图，v是G中度数为k的顶点，e是G的一条边。
(1) $G-v$有多少个顶点和多少条边？
(2) $G-e$有多少个顶点和多少条边？
(3) $G\backslash e$有多少个顶点和多少条边？

作业 1.6 设G为至少有两个顶点的简单图，证明G至少有两个顶点度数相同。

作业 1.7 下面两个图G_1和G_2同构吗？如果同构，请写出它们顶点之间的对应关系；如果不同构，请说明理由。



G_1



G_2

作业 1.8 判断下面两个命题是否正确，并说明理由：
(1) 任何两个同构的图都有相同的顶点数和相同的边数；
(2) 任何具有相同顶点数和相同边数的图都是同构的。

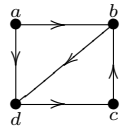
作业 1.9 (1) 给出所有非同构的无向4阶自补图；(2) 给出所有非同构的无向5阶自补图。

作业 1.10 (1) 画出4阶无向完全图K_4的所有非同构的生成子图，并指出其中的自补图；(2) 画出3阶有向完全图所有非同构的4条边的子图及其它们的补图。

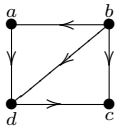
作业 1.11 设G_1, G_2, G_3是3个无向简单图，都有4个顶点，2 条边，证明他们之中至少有两个是同构的。

作业 1.12 无向图G是3-正则图，且顶点数n与边数m满足：$2n-3=m$，试问G有几种非同构的情况？并证明你的结论！

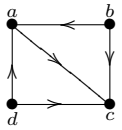
作业 1.13 对于下面六个图：



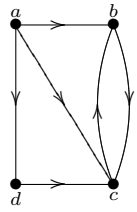
(1)



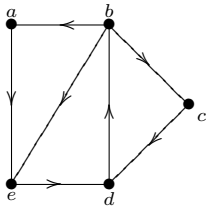
(2)



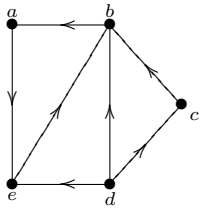
(3)



(4)



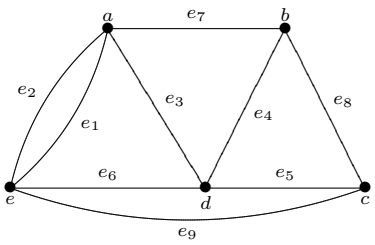
(5)



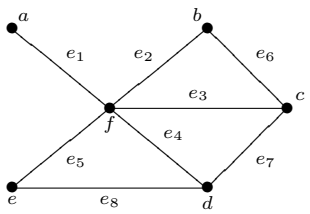
(6)

(i) 哪几个是强连通的？(ii) 哪几个是单向连通的？(iii) 那几个是弱连通的？

作业 1.14 对于下面两个图：



(1)



(2)

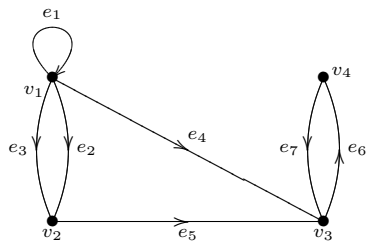
(1) 求图(1)中3条边和4条边的边割集各一个，求图(2)的一个最小边割集和一个最大的边割集；
(2) 求图(1)中的一个最小的点割集，在(2)中求含1个顶点和2个顶点的点割集各一个；
(3) 求图(1),(2)的点连通度、边连通度。

作业 1.15 分别构造连通图G_1, G_2, G_3，使得：
(1) $\kappa(G_1) = \lambda(G_1) < \delta(G_1)$；
(2) $\kappa(G_2) < \lambda(G_2) = \delta(G_2)$；
(3) $\kappa(G_3) < \lambda(G_2) < \delta(G_3)$。

作业 1.16 (1) 证明：若无向图G恰有两个奇度顶点，则这两个顶点是连通的；(2) 若有向图D中只有两个奇度顶点，他们一个可达另一个或相互可达吗？

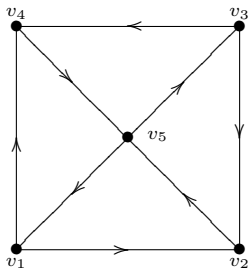
作业 1.17 设$n \geq 6$，将无向完全图K_n的边涂上蓝色或红色，证明任何一种随意的涂法，都存在红色的K_3或蓝色的K_3；以此证明任何6个人中，或有三人彼此都认识，或有三人彼此都不认识。

作业 1.18 对于下面的有向图D：



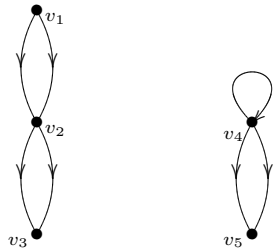
(1) 给出图D的邻接矩阵A；(2) D中v_1到v_4长度为4的通路数有多少？(3) D中v_1到v_1自身长度为3的回路数为多少？(4) D中长度为4的通路总数有多少？其中有几条回路？(5) D中长度小于等于4的通路有多少条？其中有多少条回路？

作业 1.19 对于下面的有向图D：



(1) 给出图D的邻接矩阵A；(2) D中长度为4的通路有多少条，其中有几条为回路？(3) 利用Warshall算法求图D的可达矩阵，请写出计算的中间结果。图D是哪种类型的有向连通图？

作业 1.20 对于下面的有向图D：



(1) 给出图D的邻接矩阵A；(2) 求v_1到各顶点的距离；(3) 求图D的可达矩阵。

第二章　树的基本概念

这一章我们考虑与树有关的基本概念，包括无向树的定义及基本性质、图的关联矩阵与生成树的计数、有向树（根树）和二叉树的性质，以及最优二叉树(Huffman树)的构造及应用等。

2.1　树的基本定义

定义 2.1.1 若简单连通图$G=(V,E)$没有回路，则称G是（无向）树。无向树中度数为1的顶点称为树叶，度数大于等于2的顶点称为分支点。只有一个顶点（无边）的简单图称为平凡树；若G含有多个无回路的连通分支，则称G是森林。

也就是说，（无向）树就是连通无回路的简单图。后面在提到树时，如果没有特别说明都是指无向树。在证明树的性质之前，先会议一下桥（或说割边）的定义，我们将看到树的每一条边都是桥。

定义 2.1.2 给定图$G=(V,E)$，说边$e\in E$是G的桥（割边），如果$p(G-e)>p(G)$，即G中删除e之后会增加连通分支数。

显然，如果e是G的桥，则有$p(G-e)=p(G)+1$，即删除一条边，最多增加一个连通分支。另一方面，若e是G的桥，则e不属于G的任意回路。根据这些基本常识，下面证明树的基本性质。

定理 2.1.3 给定简单图$G=(V,E)$，$|V|=n,|E|=m$，下面各命题等价：

(1) G是树（即G连通且无回路）；

(2) G的任意两个顶点存在惟一的道路；

(3) G连通且任意边都是桥；

(4) G无回路，但在任何两个不相邻顶点之间加一条新边，则得到惟一一个含新边的回路。

证明　我们采用循环论证法，即证明(1) $\Rightarrow$ (2) $\Rightarrow$ (3) $\Rightarrow$ (4) $\Rightarrow$ (1)。

(1) $\Rightarrow$ (2)：由于G是连通的，因此G的任意两个顶点之间都存在道路，而若G的两个顶点之间存在两个不同的道路，则这两个道路构成回路，与G无回路矛盾，因此G的任意两个顶点必存在惟一的道路。

(2) $\Rightarrow$ (3)：当G的任意两个顶点存在惟一的道路时，显然G是连通的。对G的任意边$e=(u,v)$，如果e不是G的桥，则$G-e$仍是连通的，即存在以u和v为端点的道路Γ，那么G就存在以u和v的两条道路Γ和$\Gamma'=uev$，矛盾，因此G的任意边必是桥。

(3) $\Rightarrow$ (4)：因为G的任意边都是桥，因此G无回路，因此G连通且无回路，即G是树，从而由(1) $\Rightarrow$ (2)得G的任意两个顶点之间只存在惟一的道路，从而若在G的两个不相邻顶点之间加一条

新边，必得到含有该新边的回路，且该回路是惟一的（不然则这两个顶点之间原来就会有两条不同的道路，矛盾!）。

(4) ⇒ (1)：只要证明G是连通的，对G的任意两个顶点u, v，如果u和v相邻，则u和v是可达的；如果u和v不相邻，则在u和v之间增加一条新边可得到惟一一个含该新边的回路，这表明在G中u和v也是可达的。因此G的任意两个顶点都是可达的，即G是连通的。 □

定理 2.1.4 给定简单图$G = (V, E)$，$|V| = n, |E| = m$，下面各命题等价：

(1) G是树（即G连通且无回路）；

(2) G连通且任意边都是桥；

(3) G无回路且$m = n - 1$；

(4) G连通且$m = n - 1$；

证明 由于(1)和(2)的等价性上面已经证明，因此这里只要证明(2) ⇒ (3) ⇒ (4) ⇒ (2)即可。

(2) ⇒ (3)：显然这时G没有回路，我们使用归纳法证明$m = n - 1$。当$n = 1$时，G为平凡图（平凡树），$m = 0$，显然$m = n - 1$。假定对任意少于k个顶点的连通图G，若G的任意边都是桥时，G的边数等于顶点数减一。考虑具有k个顶点的图，对G的任意边e，因为G连通且e是桥，因此$p(G - e) = 2$，设G的两个连通分支分别是G_1和G_2，其顶点数分别是n_1和n_2，边数分别是m_1和m_2，则显然G_1和G_2的任意边仍然是桥，且$n_1 < k$和$n_2 < k$，根据归纳假设有$m_1 = n_1 - 1$且$m_2 = n_2 - 1$，从而G共有$n_1 - 1 + n_2 - 1 + 1 = n_1 + n_2 - 1 = k - 1$条边，命题成立。因此当$G$连通且任意边是桥时，$G$无回路且边数等于顶点数减1。

(3) ⇒ (4)：这只要证明G是连通的，用反证法，设G有$k > 1$个连通分支，每个连通分支的顶点数分别是$n_1, \cdots, n_k$，则G的每个连通分支无回路，即G的每个连通分支是树，从而G的每个连通分支的边数分别是$n_1 - 1, \cdots, n_k - 1$，从而G的边数m为$n_1 - 1 + n_2 - 1 + \cdots + n_k - 1 = n - k$，当$k > 1$时与$m = n - 1$矛盾！因此必有$k = 1$，即$G$是连通图。

(4) ⇒ (2)：只要证明当G连通且$m = n - 1$时，G的任意边都是桥。对G的任意边e，若$G - e$是连通的，则由于$G - e$的顶点数仍然是n，而连通简单图的边数要大于顶点数减1，因此$G - e$的边数要大于等于$n - 1$，但G的边数$m = n - 1$，从而$G - e$的边数是$n - 2$，因此必有$G - e$是不连通的，这就说明e是桥。 □

总的来说，上面两个定理说明树具有这样的性质：(i) **连通无回路**；(ii) **边数等于顶点数减一**；(iii) **每条边都是桥**；(iv) **任意两个顶点之间有惟一的道路**；(v) **任何两个不相邻顶点之间加新边得惟一的回路**。我们通常用$T = (V, E)$表示树。

例子 2.1.5 若$T = (V, E)$是树，且$|V| = n \geq 2$，则T至少有两片树叶，即至少有两个度数为1的顶点。

证明 因为T是连通图，所以对T的任意顶点v都有$d(v) \geq 1$，如果T至少有$n - 1$个顶点的度数大于等于2，设T的边数为m，则

$$2m = \sum_{v \in V} d(v) \geq 2(n - 1) + 1$$

即$m \geq n$，这与$m = n - 1$矛盾，因此T至多只有$n - 1$个顶点的度数大于等于2，也即至少有两个顶点的度数为1。 □

2.2 生成树

任何无向连通图都存在一棵树作为它的子图，我们称这种树为无向连通图的树，特别地，若这种树是图的生成子图（即顶点数一样的子图），则称其为生成树：

定义 2.2.1 给定无向连通图$G=(V,E)$（不一定是简单图），若$T=(V,E')$是G的生成子图且是树，则称T是G的生成树。若T是G的生成树，则对任意的边$e\in E$，若$e\in E'$，则称e是T的树枝，否则称e是T的弦，并称由G的顶点和T的弦构成的G的子图（即$G-T$）称为T的余树，记为$\overline{T}$。

注意G的生成树T的余树$\overline{T}$不一定连通，也不一定不含有回路，因此$\overline{T}$不一定是树。很容易证明：

引理 2.2.2 任意无向图$G=(V,E)$具有生成树当且仅当G是连通图。

证明 显然当G具有生成树时，G是连通图，因此生成树本身是连通图。反之，当G是连通图时，若G无回路，则G就是自己的生成树；若G含有回路，任取一回路，随意地删除回路上的一条边，若再有回路则继续删除回路上的边，直到不含有任意回路为止，最后得到的图仍然是连通的（因为删除回路上的边不会增加连通分支），而且是无回路的，也即就是G的生成树。通常人们称这种产生生成树的方法为破圈法。 □

推论 2.2.3 设图G是n个顶点m条边的无向连通简单图，则$m\geq n-1$。

推论 2.2.4 设图G是n个顶点m条边的无向连通简单图，T是G的生成树，则T的余树$\overline{T}$有$m-n+1$条边。

推论 2.2.5 设T是连通图G的一棵生成树，$\overline{T}$是它的余树，则G的任意回路C必含$\overline{T}$中的边（即T的弦）。

证明 因为否则的话，回路C的边都在树T中，这与T无回路矛盾。 □

更进一步可证明下面的定理：

定理 2.2.6 设T是连通图G的一棵生成树，e为T的弦，则$T\cup\{e\}$必含有一个回路使得该回路除了含有弦e之外不再有T的其他弦。

证明 设$e=(u,v)$，则T中有u到v的惟一道路Γ，显然$\Gamma\cup\{e\}$是含有弦e的回路。 □

显然不同的弦含的回路也不同，我们有如下的定义：

定义 2.2.7 设T是n个顶点m条边的无向连通简单图G的一棵生成树，设$e'_1,e'_2,\cdots,e'_{m-n+1}$是$T$的弦，设$C_r$是$T$添加弦$e'_r$后产生的只含弦$e'_r$，其他边都是树枝的回路，称$C_r$是$G$的对应$T$的弦$e'_r$的基本回路，称$\{C_1,C_2,\cdots,C_{m-n+1}\}$是$G$对应$T$的基本回路系统。

注意，不同生成树的基本回路系统可能不同，但不同生成树的基本回路系统中回路数目是一样的，即都是$m-n+1$。从弦导出基本回路系统，从树枝则可得到基本割集系统。首先有如下定理：

定理 2.2.8 设T是连通图G的一棵生成树，e是T的树枝，则G存在只含树枝e，其他边都是弦的边割集，且不同的树枝对应的边割集也不同。

证明 根据树的基本性质，e是T的桥，因此$T-e$有两个连通分支T_1, T_2，令

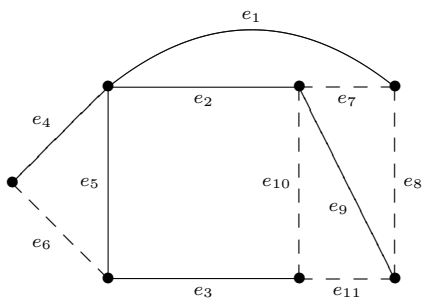
$$S_e = \{(u,v) \mid u \in T_1 \wedge v \in T_2\}$$

即S_e种是哪些两个端点分属于T_1和T_2的边构成的边集，显然S_e是边割集，而且$e \in S_e$，且S_e中的其他边都是弦（否则作为树T的连通分支T_1和T_2有边相连）。 □

定义 2.2.9 设T是有n个顶点的连通图G的一棵生成树。$e_1, e_2, \cdots, e_{n-1}$是$T$的树枝。$S_i$是$G$的只含树枝$e_i$的边割集，则称$S_i$是$G$的对应$T$由树枝$e_i$生成的**基本割集**，称$\{S_1, S_2, \cdots, S_{n-1}\}$为$G$对应$T$的**基本割集系统**。

同样，不同生成树的基本割集系统也可能不同，但不同生成树的基本割集系统中的边割集数目都是一样的，即$n-1$。

例子 2.2.10 考虑下面的图（参见耿素云、屈婉玲教材[4]p309的图16.3）：



其中实线给出的边构成了上图的生成树，它的弦包含$e_6, e_7, e_8, e_{10}, e_{11}$，对应的基本回路系统是：

$$C_6 = e_6 e_4 e_5 \qquad C_7 = e_7 e_2 e_1$$
$$C_8 = e_8 e_9 e_2 e_1 \qquad C_{10} = e_{10} e_3 e_5 e_2$$
$$C_{11} = e_{11} e_3 e_5 e_2 e_9$$

该生成树的树枝包括$e_1, e_2, e_3, e_4, e_5, e_9$，对应的基本割集系统是：

$$S_1 = \{e_1, e_7, e_8\} \qquad S_2 = \{e_2, e_7, e_9, e_{10}, e_{11}\}$$
$$S_3 = \{e_3, e_{10}, e_{11}\} \qquad S_4 = \{e_4, e_6\}$$
$$S_5 = \{e_5, e_6, e_{10}, e_{11}\} \qquad S_6 = \{e_9, e_8, e_{11}\}$$

为了得到连通图的所有生成树及其数目，我们考虑图的关联矩阵：

定义 2.2.11 给定有向图$G = (V, E)$，设$V = \{v_1, v_2, \cdots, v_n\}$，而$E = \{e_1, e_2, \cdots, e_m\}$，则$G$的关联矩阵$B = [b_{ij}]$是$n \times m$的矩阵，对任意的$1 \le i \le n, 1 \le j \le m$有：

$$b_{ij} = \begin{cases} 1 & \text{若} v_i \text{是} e_j \text{的起点} \\ -1 & \text{若} v_i \text{是} e_j \text{的终点} \\ 0 & \text{否则} \end{cases}$$

显然，任何一个有向图的关联矩阵具有如下特征：(i) 孤立点对应的行向量为零向量；(ii) 每个列向量有且仅有两个非零元−1和1；(iii) 如果该有向图不连通，则通过适当重排之后，其关联矩阵可变成对角分块矩阵。

关联矩阵的秩与有向图的弱连通性有密切关系，为方便起见，我们将有向图不考虑边之后得到的无向图称为该有向图的基图。我们不加证明地给出如下命题，有关证明可参考耿素云教材[1]的10.1节（定理10.1）。

命题 2.2.12 弱连通有向图$G=(V,E)$的关联矩阵B的秩$r(B)=|V|-1$。

有向图的关联矩阵划去一行向量后得到的矩阵称为该有向图的基本关联矩阵。

定义 2.2.13 给定有向图$G=(V,E)$，设$V=\{v_1,v_2,\cdots,v_n\}$，而$E=\{e_1,e_2,\cdots,e_m\}$。其关联矩阵$n\times m$矩阵B，称划去B中对应顶点v_k对应的行向量得到的$(n-1)\times m$矩阵B_k为G的对应v_k的基本关联矩阵。

基本关联矩阵与有向图的基图中的回路有如下关系：

命题 2.2.14 给定有向图$G=(V,E)$，设$V=\{v_1,v_2,\cdots,v_n\}$，而$E=\{e_1,e_2,\cdots,e_m\}$。B_k是G的基本关联矩阵。设$\Gamma=v_{i1}e_{i1}v_{i2}e_{i2}\cdots e_{ij}v_{i1}$是$G$的基图的一个简单回路，则$e_{i1},\cdots,e_{ij}$对应的$B_k$的各列向量线性相关。

证明 参见戴一奇教材[3]3.2节定理3.2.5。 □

根据这个性质可以证明：

命题 2.2.15 给定有向弱连通图$G=(V,E)$，$|V|=n$，其基本关联矩阵B_k的任一$(n-1)$阶子阵非零的充要条件是：该子阵各列对应的图G的边（不考虑方向）构成G的基图的一棵生成树。

证明 参见戴一奇教材[3]3.2节定理3.2.6，或参考耿素云教材[1]的10.1节（定理10.3）。 □

根据在矩阵理论中的Binet-Cauchy定理可以对图的生成树进行计数：

命题 2.2.16 设B_k是有向弱连通图$G=(V,E)$的某一基本关联矩阵，则G的不同树的生成树数目是（下面的B_i是B_k的某一$n-1$阶子阵，而下面的求和则取遍B_k的所有可能$n-1$阶子式）：

$$|B_kB_k^T|=\sum_i|B_i||B_i^T|=\sum_i|B_i|^2$$

如果要给出图G的所有生成树，则可在B_k中用E中相应的元素标记，得到矩阵$B_k^e=(b_{ij}^e)$，即令：

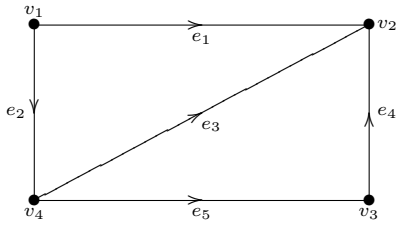
$$b_{ij}^e=\begin{cases}e_j & \text{若 } b_{ij}=1\\ -e_j & \text{若 } b_{ij}=-1\\ 0 & \text{否则}\end{cases}$$

则$|B_k^eB_k^T|$给出所有生成树的清单。

证明 参见戴一奇教材[3]3.3节定理3.3.1。 □

我们以一个比较简单的例子来说明上述定理的使用：

例子 2.2.17 求下图的生成树的数目：



解：任取该图的一个基本关联矩阵，例如B_4：

$$B_4 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

如果直接计算$|B_4 B_4^T|$则有：

$$|B_4 B_4^T| = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 3 & -1 \\ 0 & -1 & 2 \end{bmatrix} = 8$$

如果利用Binet-Cauchy定理则共有$C_5^3 = 10$种可能，从而有：

$$\begin{aligned} |B_4 B_4^T| = &\begin{vmatrix} 1 & 1 & 0 \\ -1 & 0 & -1 \\ 0 & 0 & 0 \end{vmatrix}^2 + \begin{vmatrix} 1 & 1 & 0 \\ -1 & 0 & -1 \\ 0 & 0 & 1 \end{vmatrix}^2 + \begin{vmatrix} 1 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{vmatrix}^2 + \\ &\begin{vmatrix} 1 & 0 & 0 \\ -1 & -1 & -1 \\ 0 & 0 & 1 \end{vmatrix}^2 + \begin{vmatrix} 1 & 0 & 0 \\ -1 & -1 & 0 \\ 0 & 0 & -1 \end{vmatrix}^2 + \begin{vmatrix} 1 & 0 & 0 \\ -1 & -1 & 0 \\ 0 & -1 & -1 \end{vmatrix}^2 + \\ &\begin{vmatrix} 1 & 0 & 0 \\ 0 & -1 & -1 \\ 0 & 0 & -1 \end{vmatrix}^2 + \begin{vmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{vmatrix}^2 + \\ &\begin{vmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & -1 \end{vmatrix}^2 + \begin{vmatrix} 0 & 0 & 0 \\ -1 & -1 & 0 \\ 0 & -1 & -1 \end{vmatrix}^2 \end{aligned}$$

显然除了第一个和最后一个行列式为0之外，其他的行列式的值都等于1。

为了给出所有生成树，我们计算：

$$|B_4^e B_4^T| = \begin{vmatrix} e_1 & e_2 & 0 & 0 & 0 \\ -e_1 & 0 & -e_3 & -e_4 & 0 \\ 0 & 0 & 0 & e_4 & -e_5 \end{vmatrix} \begin{vmatrix} 1 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{vmatrix} = \begin{vmatrix} e_1+e_2 & -e_1 & 0 \\ -e_1 & e_1+e_3+e_4 & -e_4 \\ 0 & -e_4 & e_4+e_5 \end{vmatrix}$$

$$= (e_1+e_2)((e_1+e_3+e_4)(e_4+e_5)-e_4^2)-e_1^2(e_4+e_5)$$

$$= (e_1+e_2)(e_1e_4+e_1e_5+e_3e_4+e_3e_5+e_4e_5)-e_1^2e_4-e_1^2e_5$$

$$= e_1e_3e_4+e_1e_3e_5+e_1e_4e_5+e_2e_1e_4+e_2e_1e_5+e_2e_3e_4+e_2e_3e_5+e_2e_4e_5$$

这就得到了所有的生成树，即由边e_1,e_3,e_4构成的生成树，由e_1,e_3,e_5构成的生成树，等等。

对于无向连通图，我们可以将该无向图的每一条边随意地给一个方向，然后利用上面的方法计算所得到的弱连通有向图的所有生成树，去掉树边的方向，则得到无向图的生成树。

我们也可使用另外一种方法得到无向连通图的所有生成树。对于无向连通图G，由于环不会在生成树中出现，因此我们将G中的环去掉，或者不妨假设G不含环。

设e是G的一条边，则G的含e的生成树则与图$G\backslash e$的生成树一一对应，这里$G\backslash e$是图G对边e的收缩，则将e的两个端点u,v缩成一个顶点w，原先与u,v关联的边都与w关联。

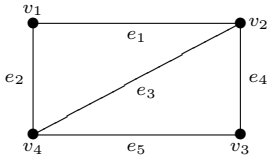
容易证明，G中存在含e的回路当且仅当$G\backslash e$中存在过w的回路，因此G的含e的生成树T在不考虑边e的时候是$G\backslash e$的生成树，因为T显然是$G\backslash e$的生成子图（即是$G\backslash e$的子图且有相同的顶点），而且T连通（因为在G中连通），而且不含有回路，因为在G中没有回路，则在G中没有含边e的回路，从而T在$G\backslash e$中没有过w的回路，而T作为G的生成树不可能在$G\backslash e$中含不过w的回路。类似地，可证明$G\backslash e$的生成树T'，加上边e后是G的生成树，即G的必含e的生成树与$G\backslash e$的生成树一一对应。

另一方面，G的不含e的生成树则显然与图$G-e$的生成树一一对应。注意，若这是e是割边，则$G-e$是非连通图，从而不存在生成树，也即G的生成树必含有e。这样我们证明了：

定理 2.2.18 设G是无向连通图，e是G的一条非环边，则G的生成树数目等于图$G-e$的生成树数目与图$G\backslash e$的生成树数目之和。 □

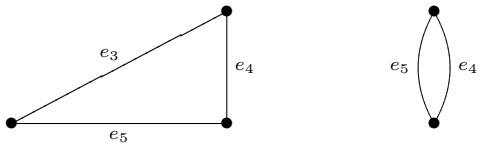
注意，在利用$G-e$和$G\backslash e$计算G的生成树时，遇到环（通常是图$G\backslash e$产生环）时，应自动忽略它们。下面以例子说明这种计算生成树的方法。

例子 2.2.19 求下图的生成树的数目：



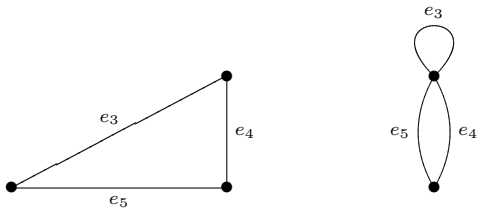
根据上述定理，该图不含e_1的生成树与下面左图（即$G-e_1$）的生成树一一对应，而含e_1的生成树与下面右图（即$G\backslash e_1$）的生成树一一对应（由于顶点在求生成树不起关键作用，下面的图都不再标出顶点）：

对于上面左图（即图$G-e_1$），由于e_2是割边，因此它不存在不含e_2的生成树，而它含e_2的生成树与下面左图（即图$(G-e_1)\backslash e_2$）一一对应：



显然上面左图（即图$(G-e_1)\backslash e_2$）中不含e_3的生成树就是e_4e_5，对应地，原图G就有生成树$e_2e_4e_5$，而上面左图（即图$(G-e_1)\backslash e_2$）中含e_3的生成树与上面右图（即图$((G-e_1)\backslash e_2)\backslash e_3$）的生成树一一对应，显然这个图的生成树分别是$e_4$和$e_5$，对应地原图$G$就有生成树$e_2e_3e_4$和$e_2e_3e_5$。

类似地，对于图$G\backslash e_1$，其不含e_2的生成树与下面左图（即图$(G\backslash e_1)-e_2$）的生成树一一对应，而含e_2的生成树与下面右图（即图$(G\backslash e_1)\backslash e_2$）的生成树一一对应：



注意，在图$(G\backslash e_1)\backslash e_2$中，$e_3$收缩成了环，我们将它忽略后，上面两个图看似与前面的两个图一样，但对于原图而言得到的生成树不同，这时得到原图G的生成树分别是$e_1e_4e_5$（不含e_3的生成树），$e_1e_3e_4$和$e_1e_3e_5$（两棵含e_3的生成树），以及$e_1e_2e_4$和$e_1e_2e_5$（两棵含e_2但不含e_3的生成树）。综上我们得到G的所有生成树：

$e_2e_4e_5$	$e_2e_3e_4$	$e_2e_3e_5$	$e_1e_4e_5$
$e_1e_3e_4$	$e_1e_3e_5$	$e_1e_2e_4$	$e_1e_2e_5$

可以看出，用这种方法计算生成树比用关联矩阵的计算更为繁琐，程序实现起来也会更困难。

2.3 根树

定义 2.3.1 如果一个弱连通有向图不考虑边的方向是树，则称这个图是**有向树**。若有向树T有且仅有一个顶点v的入度为0，则称T是以v为**根**的**根树**。我们将根树中出度为0的顶点为**叶子**。

定义 2.3.2 对于根树$T=(V,E)$，T的两个顶点u,v。若有有向边$e=\langle u,v\rangle\in E$，则称$u$是$v$的**父亲**，而$v$是$u$的**儿子**；若存在$u$到$v$的一条有向道路，则称$u$是$v$的**祖先**，而$v$是$u$的**后代**。

定义 2.3.3 对于根树$T=(V,E)$，设其根为v，则对T的任意顶点u，从v到u的惟一道路的长度称为u的**深度**。定义T的**高度**是从v到T的叶子最长道路的长度。

显然根节点的深度为0，我们称之为第0层节点；深度同为i的节点称为第i层节点，具有最大深度的节点的深度为树的深度（高度）。对于根树，如果将每个节点的儿子按照某种特定的次序（例如在画的时候从左至右的次序）排列，则称该根树为有序根树，如果有序根树的所有节点的出度不大于m，则称为m元树（或说m叉树）；如果有序根树的所有节点的出度都是0或m，则称为m元正则树（或说m叉正则树）。

根树在计算机科学及实际生活中有许多应用，例如语法树、判定树（决策树）等都是根树的例子。也许在计算机科学中用得最多的是二元（叉）树。

定义 2.3.4 二叉树是二元有序树，即每个节点的出度不大于2，且每个节点的儿子按一定顺序排列的根树。每个节点的出度为0或2的二叉树称为正则二叉树。

通常我们将二叉树某个节点的两个儿子节点区分为左儿子和右儿子节点。不难看出，二叉树具有这样的特点：(i) 第k层最多有2^k个节点；(ii) 高度为k的二叉树最多有$2^{k+1}-1$个节点。进一步，m元有序树都可转化为一棵二叉树：以某个结点的最左儿子作为该结点的左儿子，而以该结点的右兄弟作为该结点的右儿子。

例子 2.3.5 设T是二叉树，其出度为2的节点数目是n_2，叶子结点的数目是n_0，证明$n_0 = n_2 + 1$。

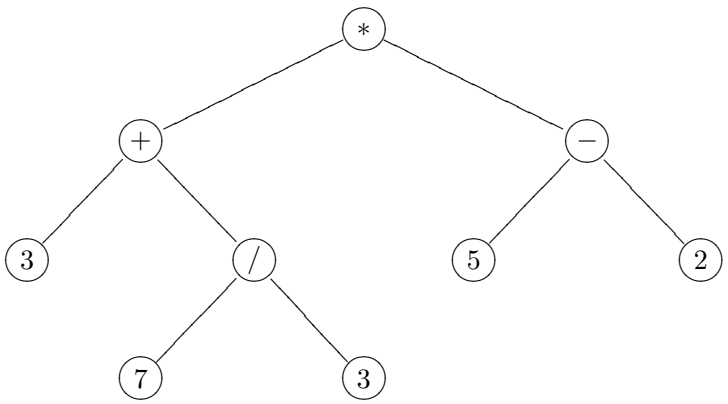
证明 对T的结点个数n进行归纳，显然当$n = 1$时，一个叶子结点，零个出度为2的节点，因此命题成立；设对于任意小于k个结点的二叉树叶子数等于出度为2的节点数加1。考虑有k个节点的任意二叉树T，若T的根v的出度为1，则T与$T - v$的叶子数和出度为2的节点数都相同，而根据归纳假设，$T - v$的叶子数等于出度为2的节点数加1，因此这时命题成立；若T的根v的出度为2，则v的左子树T_1和v的右子树T_2都是结点数小于k的二叉树，设其叶子数分别为n_{01}和n_{02}，出度为2的节点数为n_{21}和n_{22}，则根据归纳假设有：

$$n_{01} = n_{21} + 1 \qquad n_{02} = n_{22} + 1$$

也即$n_{01} + n_{02} = n_{21} + n_{22} + 2$，注意到这时$T$恰好有$n_{01} + n_{02}$个叶子节点，而有$n_{21} + n_{22} + 1$个出度为2的结点（比$T - v$多$v$这个出度为2的节点），因此这时命题也成立。 □

二叉树的遍历是程序设计中常常遇到的一个问题，数据课程中也常常研究二叉树的各种遍历方法。我们这里以一个简单的例子说明二叉树的中序、前序和后序遍历方法。

例子 2.3.6 考虑下面的二叉树：



对上述二叉树进行中序遍历则得到：

$$3+7/3*5-2$$

当然这不加上括号不会得到正确的算术表达式，加上适当括号则得到$(3+7/3)*(5-3)$。

对上述二叉树进行后序遍历则得到

$$373/+52-*$$

对这个串作一遍扫描即可计算表达式的值。对上述二叉树进行前序遍历则得到

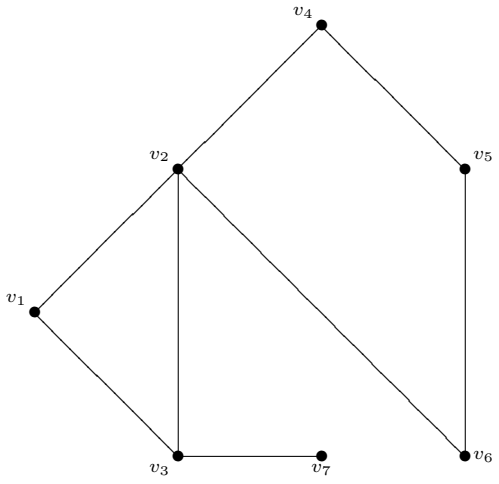
$$*+3/73-52$$

同样也可利用这个串计算该表达式的值，但不常用。实际上，二叉树的中序遍历常用于说明表达式的语法，而后序遍历常用于计算表达式的值。

至于m元树，我们可以按照上述方法将m元树转换为2元树再进行遍历。如果直接遍历m元树，从根节点出发，有两种策略：一是先深搜索：先不断地搜索结点的儿子，直到叶子结点，再回溯到父亲结点，再从该父亲结点的另外儿子开始继续策略；二是先广搜索：先搜索所有儿子结点，然后再根据一定的顺序（从左到右）选择一个儿子节点再继续搜索。

实际上，上面的中序遍历采用的就是先深搜索策略，而前序遍历采用的就是先广搜索策略。对于一般的图（有向图或无向图）的遍历也可采用这两种策略：先深搜索是从一个结点开始不断搜索其相邻结点，直到没有还未搜索过的相邻结点，则回溯到上一结点继续搜索，而先广搜索是先搜索一个结点的所有相邻结点，然后选一个结点再继续搜索。

例子 2.3.7 考虑下图的顶点遍历（边的遍历与顶点遍历类似），也即按照某种策略访问下图的所有顶点：



我们分别使用先深搜索策略和先广搜索策略从v_1开始遍历上图的所有顶点。在遍历时，我们总是使用序列T表示哪些顶点已经被遍历，并假定我们用某种方式记住了顶点与顶点之间的相邻关系（例如使用邻接矩阵），因此我们总可很容易地得到某个顶点的相邻顶点。读者可将序列T理解为一些元素的集合，但这些元素有顺序以便给出我们遍历顶点的顺序。

为了使用先深策略，我们还需要用一个栈S保存刚遍历过但还有相邻节点没有遍历的节点，这里大家无需深究栈到底是什么东西，你可将它理解为也是一个序列，即其中元素有顺序的集合，而且我们按照所谓的“先进后出”的策略访问其中的元素即可，下面的遍历过程将展示我们如何使用和访问栈。

我们从v_1开始遍历，因此开始时$T=\langle v_1\rangle$，$S=\langle\rangle$，这里用$\langle\rangle$表示空序列。下一步找到v_1的某个相邻顶点，比如说v_2（当然v_3也可，但那将得到一个不同的遍历顺序），将v_2加入T，而因为v_1有相邻结点，从而也将其加入S（按栈的术语称为“压入栈”），得到：

$$T=\langle v_1,v_2\rangle \qquad S=\langle v_1\rangle$$

再找v_2的相邻顶点（当然是除T中顶点之外的相邻顶点），比如说v_4，将v_4加入T，而将v_2压入栈S，得到：

$$T=\langle v_1,v_2,v_4\rangle \qquad S=\langle v_1,v_2\rangle$$

再找v_4的除在T中顶点之外的相邻顶点，上图只有v_5，加入v_5到T，而加入v_4压入栈S得到：

$$T=\langle v_1,v_2,v_4,v_5\rangle \qquad S=\langle v_1,v_2,v_4\rangle$$

类似地，下一步v_5在T中顶点之外的相邻顶点只有v_6，将v_6加入到T，而将v_5加入压入栈S得到：

$$T=\langle v_1,v_2,v_4,v_5,v_6\rangle \qquad S=\langle v_1,v_2,v_4,v_5\rangle$$

好了，对于v_6，它的所有相邻顶点都已经在T中（也即v_6没有尚未访问的相邻顶点），这时就需要回溯，也即要返回到上一个我们访问过的节点，这里就体现了栈的作用，我们上一个访问的节点现在在S的最后（按栈的术语就是在栈顶），也即这时返回到v_5。注意，v_5是最后入栈的顶点，而弹出的时候最先弹出，这就是栈的所谓的“后进先出”的访问策略。

现在考虑v_5，注意到它也没有不在T中的相邻顶点，那么它留在S中已经没有用了，将其删除（按栈的术语称为“弹出”v_5），则得到：

$$T=\langle v_1,v_2,v_4,v_5,v_6\rangle \qquad S=\langle v_1,v_2,v_4\rangle$$

现在再考虑在栈顶的顶点（S的最后）v_4，同样它也没有不在T中的相邻顶点，继续弹出v_4，得到：

$$T=\langle v_1,v_2,v_4,v_5,v_6\rangle \qquad S=\langle v_1,v_2\rangle$$

现在再考虑在栈顶的顶点v_2，它还存在不在T中的相邻顶点v_3，因此得到：

$$T=\langle v_1,v_2,v_4,v_5,v_6\rangle \qquad S=\langle v_1\rangle$$

现在再考虑刚弹出的v_2，它还存在不在T中的相邻顶点v_3，因此将v_3加入T，得到：

$$T=\langle v_1,v_2,v_4,v_5,v_6,v_3\rangle \qquad S=\langle v_1\rangle$$

现在考虑v_3，由于它有不在T中的相邻顶点v_7，因此将v_7加入T，而将v_3压入栈S，得到：

$$T=\langle v_1,v_2,v_4,v_5,v_6,v_3,v_7\rangle \qquad S=\langle v_1,v_3\rangle$$

现在对于v_7也没有不在T中的相邻顶点，因此再回溯到v_3，由于v_3已经没有不在T中的相邻顶点，将v_3弹出：

$$T = \langle v_1, v_2, v_4, v_5, v_6, v_3, v_7 \rangle \qquad S = \langle v_1 \rangle$$

这时栈顶的顶点v_1也没有不在T中的相邻顶点，再将v_1弹出。好了，现在栈空了，表明我们整个遍历的过程就结束了，而顶点的访问顺序就在T中。

若使用先广策略，则除了用T记住遍历结点的顺序之外，这回使用队列Q来帮助我们，实际上队列也是一个序列，只是我们采用与栈不同的访问策略访问其中的元素，即使用“先进先出”的策略访问队列。

从v_1开始遍历，同样将v_1加入到T，同样将v_1加入到Q，得到：

$$T = \langle v_1 \rangle \qquad Q = \langle v_1 \rangle$$

下一步我们考虑Q的最前面顶点v_1，将v_1的所有（不在T中的）相邻顶点v_2, v_3加入到T（即表示v_1的所有相邻顶点都已经访问），同时在Q中删除v_1（v_1出队列），而将v_2, v_3加入到Q（入队列），得到：

$$T = \langle v_1, v_2, v_3 \rangle \qquad Q = \langle v_2, v_3 \rangle$$

继续考虑Q的最前面顶点v_2，将v_2的所有（不在T中的）相邻顶点v_4, v_6加入到T和Q，同时删除v_2，得到：

$$T = \langle v_1, v_2, v_3, v_4, v_6 \rangle \qquad Q = \langle v_3, v_4, v_6 \rangle$$

注意将v_4和v_6加入Q时是加入（入队列）到Q的尾部，而删除（出队列）的v_2是在Q的最前面顶点，这就是队列的所谓“先进先出”访问策略（v_2是先入队列的顶点，因此也是最先出队列的顶点）。

继续考虑Q的最前面顶点v_3，将v_3的所有不在T中的相邻顶点v_7加入T和Q，并在Q中删除v_3，得到：

$$T = \langle v_1, v_2, v_3, v_4, v_6, v_7 \rangle \qquad Q = \langle v_4, v_6, v_7 \rangle$$

继续考虑Q的最前面顶点v_4，将v_4的所有不在T中的相邻顶点v_5加入T和Q，并在Q中删除v_4，得到：

$$T = \langle v_1, v_2, v_3, v_4, v_6, v_7, v_5 \rangle \qquad Q = \langle v_6, v_7, v_5 \rangle$$

继续考虑RQ的最前面顶点v_6，它没有不在T中的相邻顶点，直接在Q中删除，得到：

$$T = \langle v_1, v_2, v_3, v_4, v_6, v_5 \rangle \qquad Q = \langle v_7, v_5 \rangle$$

继续考虑RQ的最前面顶点v_7，它也没有不在T中的相邻顶点，直接在Q中删除，得到：

$$T = \langle v_1, v_2, v_3, v_4, v_6, v_5 \rangle \qquad Q = \langle v_5 \rangle$$

继续考虑RQ的最前面顶点v_5，它也没有不在T中的相邻顶点，直接在Q中删除，得到了空队列，这表明所有的顶点已经访问到，遍历过程结束，而节点的遍历顺序也在T中。

练习 2.3.8 读者应通过上述例子理解先深搜索和先广搜索的区别，并将上述例子中描述的方法提炼为一般的方法，并自己画一个图进行先深搜索和先广搜索。进一步，读者可通过这里例子，理解栈和队列的区别，为以后的数据课程学习打下基础。

2.4 哈夫曼树

这一章的最后，我们利用一种特殊的根树，即所谓的哈夫曼(Huffman)树考虑一类实际问题的解决。假定现在要在网络传送许多英文文件，需要对文件的内容进行编码。我们知道，对于英文而言，不是每个字母都是等概率出现的，例如c, e这些字母通常出现得最多，而z, x这些字母则出现得很少，所以如果我们都用同样长的二进制编码所有的英文字母，那么对文件编码得到二进制文件长度就不是最优的。

为简单起见，假设我们要传送的文件中只含有a, b, c, d, e, f, g, h等八个字母，每个字母在文件中出现的概率分别为：

a： 15%	b： 12%	c： 25%	d： 8%
e： 20%	f： 6%	g： 8%	h： 6%

如果我们使用相同长度的二进制对这八个字母进行编码，那么每个字母需要三位二进制，从而若要传送长度为1万个字母的英文文件，那么总共要传送3万个二进制数。

下面我们利用Huffman树对这八个字母做一种不等长的编码，使得当要传送1万个字母时，传送的二进制数可以少于3万，从而节省了网络传送的费用。为此，我们先引入有关概念。

定义 2.4.1 给定有k个叶子节点的二元树$T = (V, E)$，若k个叶子分别赋以非负实数权$w_1, w_2, \cdots, w_k$，并设这k个叶子对应的高度分别是$l_1, l_2, \cdots, l_k$，则称

$$w(T) = \sum_{i=1}^{k} l_i * w_i$$

为二元树T的带权外部通路总长。

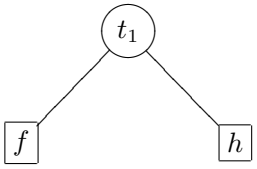
称在有k个叶子节点并分别赋以非负实数权的二元树中，具有最小的带权外部通路总长的二元数为最优二元树。

回到前面的问题，我们将要编码的字母看作二元树的叶子，字母在文件中出现的概率作为赋予叶子的权，通过构造一个最优二元树可以确定这些字母的一个比较节省网络传送费用的二进制编码方案。Huffman给出了一个构造最优二元树的算法，从而我们将利用Huffman算法构造出来的最优二元树称为Huffman树。

例子 2.4.2 我们利用上面的例子来说明Huffman算法的基本思想。首先将字母的权从小到大排序（省略百分比，实际上，Huffman算法对于权也没有特别的要求，只要非负即可）：

$$6(f) \quad 6(h) \quad 8(g) \quad 8(d) \quad 12(b) \quad 15(a) \quad 20(e) \quad 25(c)$$

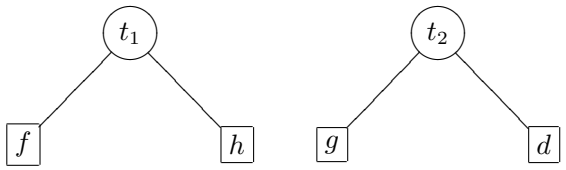
然后我们开始构造二元树。取所有权中最小的两个权，以这两个权所对应的字母作为儿子，新添加一个节点t_1作为根，得到一个子树（注意，我们特意用方框表示叶子节点，而用园框表示分支节点）：



为了便于后面的构造，我们令t_1的赋为它的两个儿子节点的权的和，即为12，而且在上面的权序列中，将f和g的权删除，而将t_1的权加入，并仍保持从小到大排序，即得到如下的权序列：

$$8(g) \quad 8(d) \quad 12(t_1) \quad 12(b) \quad 15(a) \quad 20(e) \quad 25(c)$$

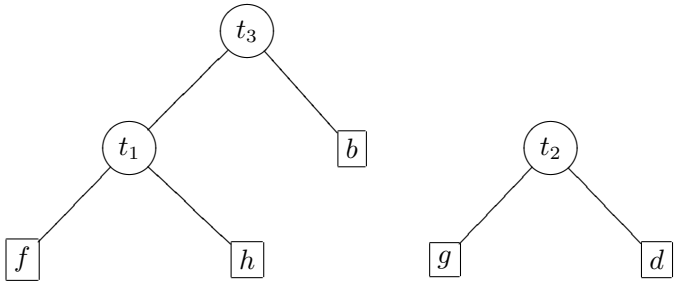
下一步跟前面类似，我们再选两个权最小的字母g和d作为儿子，并新添加一个节点t_2作为根，得到一个子树（为清楚起见，我们将先前构造的树t_1也画在下面）：



同样地，在权序列中删除g和d的权，并令t_2的权为它们的和（即16），并加入到权序列，保持从小到大的序，即得到：

$$12(t_1) \quad 12(b) \quad 15(a) \quad 16(t_2) \quad 20(e) \quad 25(c)$$

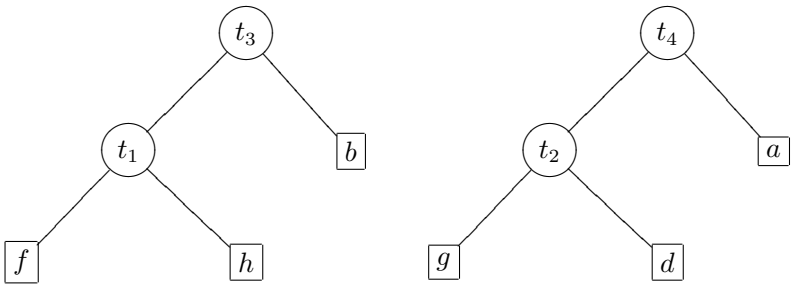
这时权最小的是t_1和b的权，也没有关系，我们将t_1看作一个单独的字母好了，取这两个权对应的节点（这时称节点更为恰当了）作为儿子，添加一个新节点t_3作为根，也就是说，现在我们得到如下的树（准确地说，是森林）。注意，以t_3是以t_1作为左儿子还是以b作为左儿子没有什么关系，随便选择即可，下面为了画图的方便，选择t_1作为左儿子：



同样地，在权序列中删除t_1和b的权，并令t_3的权为它们的和（即24），并加入到权序列，保持从小到大的序，即得到：

$$15(a) \quad 16(t_2) \quad 20(e) \quad 24(t_3) \quad 25(c)$$

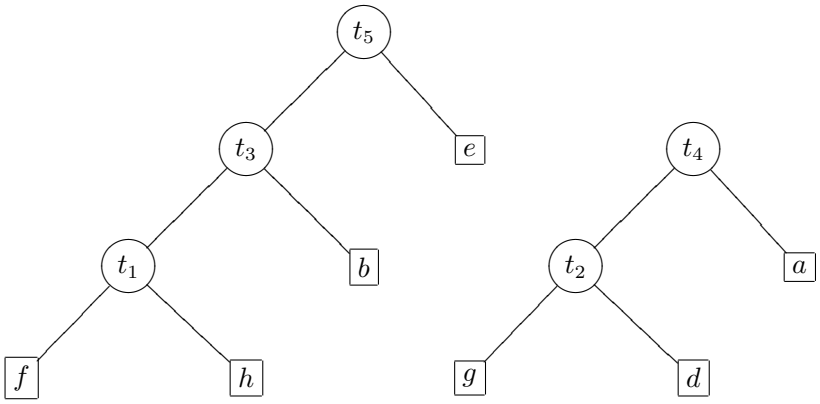
继续选择两个最小的权对应的节点（即a和t_2）作为儿子，添加一个新节点t_4作为根，也就是说，得到如下的森林：



删除a和t_2对应的权，加入t_4对应的权（即a和t_2的权之和31），并保持权序列的从小到大顺序得到：

$$20(e) \quad 24(t_3) \quad 25(c) \quad 31(t_4)$$

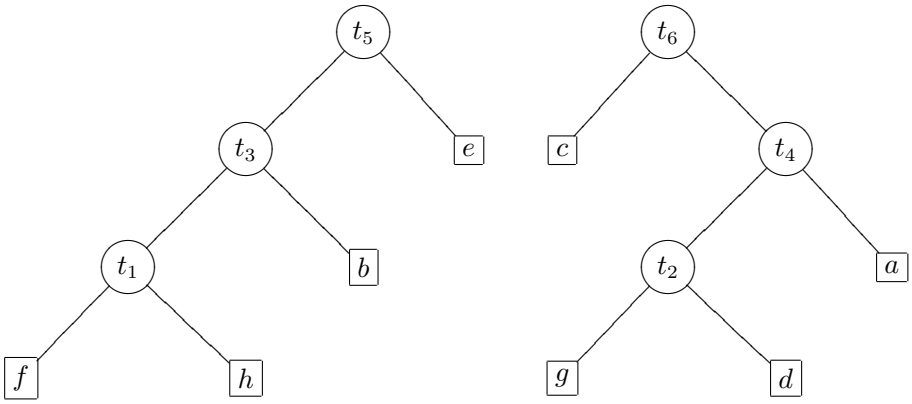
继续选择两个最小的权对应的节点（即e和t_3）作为儿子，添加新节点t_5作为根，得到如下森林：



删除e和t_3对应的权，加入t_5对应的权（即e和t_3的权之和44），保持权序列的从小到大顺序得到：

$$25(c) \quad 31(t_4) \quad 44(t_5)$$

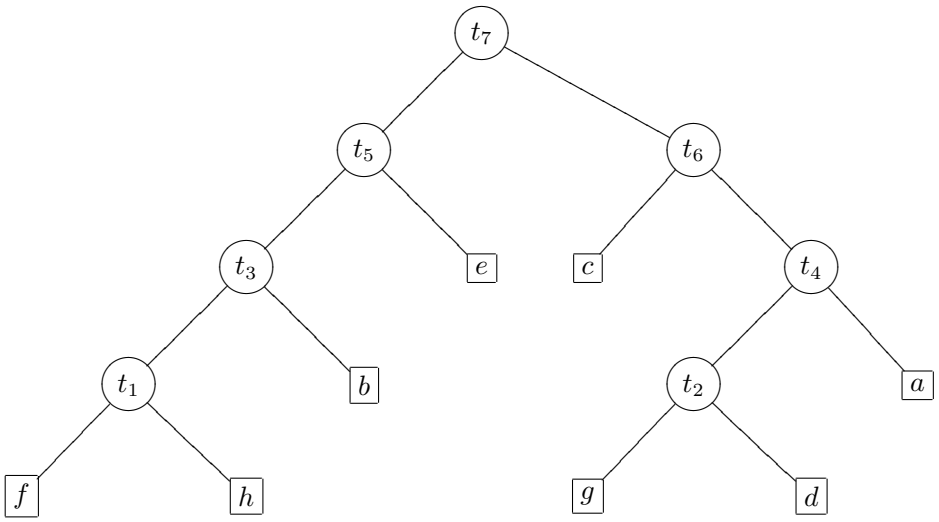
构造一个新的节点t_6，以c和t_4对应的节点作为儿子，得到：



删除c和t_4对应的权，加入t_6对应的权（即c和t_4对应的权之和56），保持权序列的从小到大顺序得到：

$$44(t_5) \quad 56(t_6)$$

构造一个新的节点t_7，以t_5和t_6对应的节点作为儿子，得到：



在权序列中删除t_5和t_6对应的权之后，再加入t_7对应的权（即t_5和t_6的权之和100）后得到的权序列中只有一个节点（即t_7），这表明我们的最优二元树的构造已经完成。但要注意，这个最优二元树T的外部通路总长并不是t_7的权，而是：

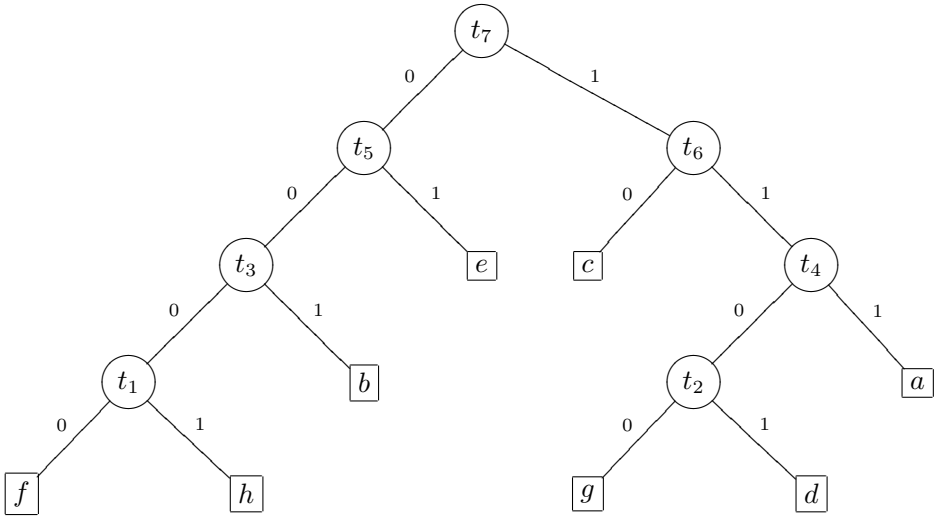
$$w(T) = 6*4+6*4+8*4+8*4+12*3+15*3+20*2+25*2 = 283$$

练习 2.4.3 根据上述例子的描述，抽象出Huffman算法的一般描述（或者说，根据这个例子，理解戴一奇教材p57页对Huffman算法的描述）。

定理 2.4.4 由Huffman算法得到的二元树是最优二元树。

证明 参见戴一奇教材[3]p58页的定理3.6.1。 □

例子 2.4.5 继续上面的例子，根据上面构造得到的Huffman树，我们可以对要传送的字母a,b,c,d,e,f,g,h进行编码。编码方法很简单，对树的每一条边赋一个二进制数：如果某条边是从父亲节点连向左儿子节点，则赋二进制数0；如果是从父亲节点连向右儿子节点，则赋二进制数1。对于上面的Huffman树，我们得到：



我们令每个叶子节点对应的字母的编码就是从根节点到该叶子节点所经过的边上的二进制数构成的串，也即得到：

a 的编码是:	111	b 的编码是:	001	c 的编码是:	10	d 的编码是:	1101
e 的编码是:	01	f 的编码是:	0000	g 的编码是:	1100	h 的编码是:	0001

与前面所给出的每个字母的出现频率相比较，不难发现，出现频率越高的字母的编码越短。因此，如果传送具有1万个字母的英文文件，该文件只出现这8个字母，而且每个字母出现的概率正如上面所给出的那样，也就是说，在1万个字母中，a大概出现15%，也即1500个，b大概出现12%，也即1200个等等，那么传送这1万个字母英文文件需要传送的二进制编码数目是：

$$\begin{aligned} W = & \ 600*4+600*4+800*4+800*4+1200*3+1500*3+2000*2+2500*2 \\ = & \ w(T)*10000/100 = 28300 \end{aligned}$$

比使用等长的三位编码要传送的二进制数3万要少1700个。

要使用不等长的二进制编码进行传输，我们还需要解决一个问题，即传送信息的分割，也即那几个二进制位是代表一个字符。很显然如果使用等长编码的话，例如3位编码，那么文件接受方在接受到数据之后，每三位三位进行译码即可，但如果是使用不等长的编码进行传送，那么接受方接收到数据之后该如何译码呢？也就是说，我们需要一个能够正确分割接收到的信息并进行译码的方法。解决这个问题的方法就是让对要传送的字符的编码满足前缀码的要求。

定义 2.4.6 设$a_1a_2\cdots a_n$是长度为n的串，我们称其子串$a_1, a_1a_2, a_1a_2a_3, \cdots, a_1a_2\cdots a_{n-1}$分别为该符号串长度为$1,2,3,\cdots,n-1$的前缀。也就是说，符号串$\beta$是符号串$\alpha$的前缀，当且仅当$\beta$是$\alpha$的从第一个字符开始的子符号串。

定义 2.4.7 设$A=\{\beta_1,\beta_2,\cdots,\beta_n\}$是一些符号串构成的集合，如果$A$满足，对任意的$i\neq j, 1\leq i,j\leq n$，都有$\beta_i$不是$\beta_j$的前缀，也即$A$中的任意两个符号串都不互为前缀，则称$A$是前缀码。进一步，若前缀码$A$中任意的串$\beta_i$都只有两种符号构成，则称$A$是二元前缀码。

例子 2.4.8 对于串110010101，其前缀有$1, 11, 110, 1100, 11001$等等，而10010虽然是它的子串，但不是它的前缀。下面的符号串集合不是前缀码：

$$A=\{11, 100, 1101, 001, 0100, 0111, 0011\}$$

因为其中11是1101的前缀，001是0011的前缀。而我们前面根据Huffman树得到的编码集（符号串集）

$$H=\{111(a), 001(b), 10(c), 1101(d), 01(e), 0000(f), 1100(g), 0001(h)\}$$

就是前缀码。一般地，如果按照上面的编码方法，即将连向左儿子的边编码为0，右儿子的边编码为0，叶子结点的编码为根到该叶子结点的路径上的二进制串，那么任意一棵二元树的叶子结点的编码构成的集合一定是前缀码，因为根到任何叶子结点的路径都是惟一的，绝不会是根到另外一个叶子结点的一部分。

对于二元前缀码，由于任何一个码都不是另外一个码的前缀，从而对于一些字母的编码，如果传送正确的话，那么在接收方都能正确地译码，例如我们要传送的字符串是$faceaddheadgeaf$，根据上面的编码我们要传送如下的二进制串：

$$0000111100111111011101000101111110111000111110000$$

接收方只要知道每个字母的编码，就能正确地将上述二进制串翻译回英文字符串。例如，看到第一个二进制0时，那么可能的字符是b, e, f, h，再看到下一个0时，可能的字符只有b, f, h，再看到一个0，可能的字符剩下f, h，再看到一个0，就只有f了，而且由于0000不会是任何其他字符编码的前缀，因此0000只可能译码成f。

如果用于编码字符的不是前缀码，那么就无法正确分割信息并进行译码，例如，若利用上述编码集A进行编码，传送如下的二进制串：

$$1101110010011$$

那么当接收方接收到11时，它不知道是将110111分割成110111呢，还是分割成110111，而使用前缀码进行编码则只可能存在惟一的一种分割方式。

作 业

作业 2.1 设无向简单图G是由$k(k \geq 2)$棵树构成的森林，已知G有n个顶点，m条边，证明$m = n - k$。

作业 2.2 设无向简单图G有n个顶点，$n-1$条边，则G为树。这个命题正确吗？为什么？

作业 2.3 设无向简单图G有n个顶点，$n-1$条边，则G是连通当且仅当G无回路。

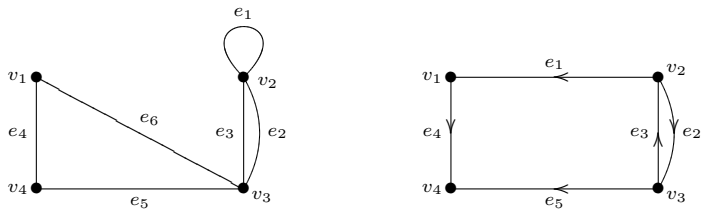
作业 2.4 已知无向树T有三个3度顶点，一个2度顶点，其余都是1度顶点。(1) T有几个叶子节点？(2) 画出两棵满足上述度数要求的非同构的无向树。

作业 2.5 一棵无向树有n_i个顶点的度数为i，$i = 1, 2, \cdots, k$。如果已知$n_2, n_3, \cdots, n_k$，试问n_1应为多少？

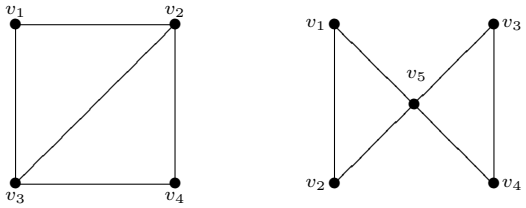
作业 2.6 设T是一棵非平凡的无向树，$\Delta(T) \geq k$，证明：T至少含有k片树叶。

作业 2.7 设$d_1, d_2, \cdots, d_n$是n个正整数，$n \geq 2$，若$\sum_{i=1}^{n} d_i = 2n - 2$，证明存在一棵顶点度数分别为$d_1, d_2, \cdots, d_n$的无向树。

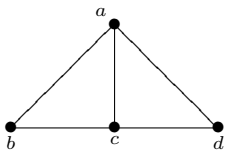
作业 2.8 给出下面无向图G和有向图D的关联矩阵：



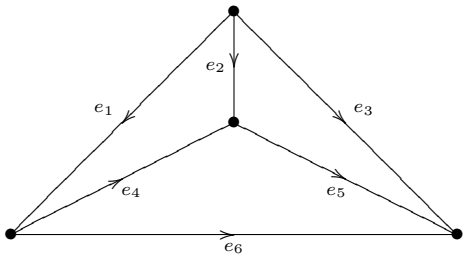
作业 2.9 求下面两个图的所有生成树，它们各有几棵非同构的生成树？



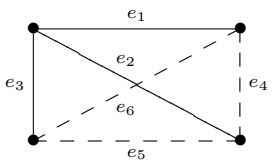
作业 2.10 求下图的所有生成树



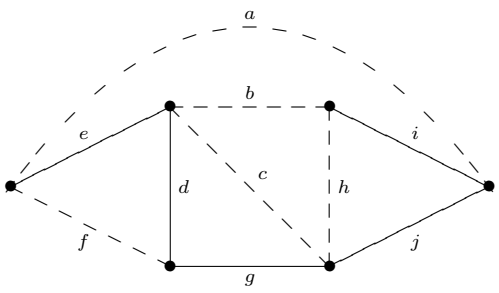
作业 2.11 求下面有向图的所有生成树：



作业 2.12 下图实线给出了它的一棵生成树，请给出该图关于此生成树的基本割集系统和基本回路系统。

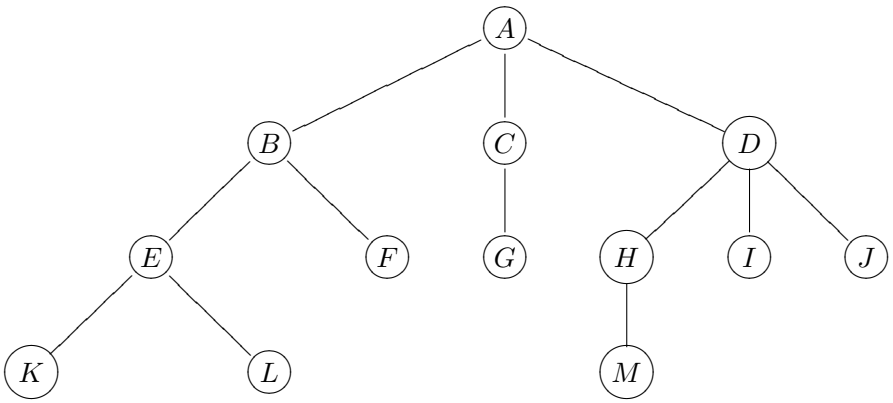


作业 2.13 下图实线给出了它的一棵生成树，请给出该图关于此生成树的基本割集系统和基本回路系统。



作业 2.14 有向图D仅有一个顶点入度为0,其余顶点的入度都是1，D一定是有向树吗？

作业 2.15 对于下面根树T：



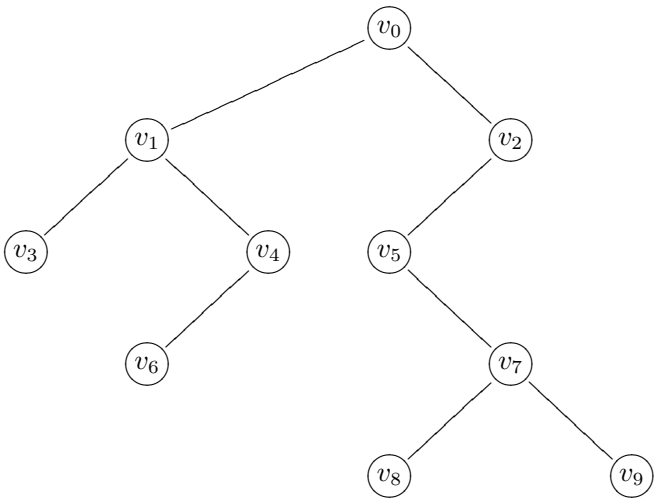
(1) 给出它所有的根节点、树叶节点、分支节点、内点；
(2) 给出每个节点的父节点、子节点；
(3) 给出每个节点的层；
(4) 给出树高和最大出度；
(5) 给出所有子（根）树；
(6) 给出它的中序、前序和后序遍历结果。

作业 2.16 设T是一棵二元正则树，m是其边数，t是树叶数且$t \geq 2$，证明：$m = 2t - 2$。

作业 2.17 设T是一棵r元正则树，i是T的分支点数，t是树叶数，证明：$(r-1)i = t - 1$。

作业 2.18 设T是一棵高为h的r元正则树，t是树叶数，证明：$r + (r-1)(h-1) \leq t \leq r^h$。

作业 2.19 请给出下面二叉树的中序、前序和后序遍历结果。



作业 2.20 给出表达式：

$$((a+(b*c)*d)-e)\div(f+g)+(h*i)*j$$

的波兰符号法和逆波兰符号法表示。

作业 2.21 下面给出的三个符号串集中，哪些是前缀码，哪些不是？为什么？

(1) $B_1=\{0,10,110,1111\}$

(2) $B_2=\{1,01,001,000\}$

(3) $B_3=\{1,11,101,0001,0011\}$

作业 2.22 求带权为2, 3, 5, 7, 8的最优二叉树，并写出其对应的二元前缀码。

作业 2.23 在通信中要传输八进制数字0, 1, 2, 3, 4, 5, 6, 7，这些数字出现的频率分别是：

0 : 30%	1 : 20%	2 : 15%	3 : 10%
4 : 10%	5 : 6%	6 : 5%	7 : 4%

编一个最佳前缀码，使通信中出现的二进制数字尽可能少：

(1) 画出相应的二叉树；

(2) 写出每个数字对应的前缀码；

(3) 传输按上述比例出现的数字10000个时，至少要用多少个二进制数字？

第三章　路径问题

这一章集中考虑几个与图的路径有关的几个问题，包括有向图中任意两点之间的最短路径，图的最小生成树，以及有向网络关键路径问题，这些问题都是在实际应用中经常遇到的问题，例如求网络的关键路径是项目管理（包括软件开发项目）中的重要问题之一。

3.1　最短路径

定义 3.1.1 给定有向图$G=(V,E)$，设$V=\{v_1,v_2,\cdots,v_n\}$。如果给每一条有向边$e=\langle v_i,v_j\rangle$赋一个非负实数权w_{ij}，则称图G为有向网络。

假定这里考虑的有向网络都是简单有向图，即没有多重边（或说有多重边时取权最小的边），也没有环。

定义 3.1.2 给定有向网络$G=(V,E)$，设$V=\{v_1,v_2,\cdots,v_n\}$。定义G的距离矩阵$D_{n\times n}=[d_{ij}]$，其中

$$d_{ij}=\begin{cases} w_{ij} & \text{若存在有向边 } e=\langle v_i,v_j\rangle\in E \\ \infty & \text{否则} \end{cases}$$

定义 3.1.3 给定有向网络$G=(V,E)$，设$V=\{v_1,v_2,\cdots,v_n\}$。定义有向路径$\Gamma=v_{i_1}v_{i_2}\cdots v_{i_k}$的带权路径长度为:

$$d(\Gamma)=\sum_{j=1}^{k-1}w_{i_ji_{j+1}}$$

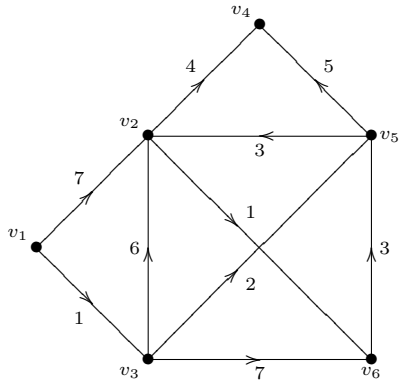
若定点v_i可达v_j，则称v_i到v_j的所有有向路径中具有最小带权路径长度的路径为v_i到v_j的最短路径，v_i到v_j的最短路径的带权路径长度称为v_i到v_j的最短距离。

显然我们有：

定理 3.1.4 给定有向网络$G=(V,E)$，设$V=\{v_1,v_2,\cdots,v_n\}$。若路径$v_1,v_2,\cdots,v_{k-1},v_k$是$v_1$到$v_k$的最短路径，则路径$v_1,v_2,\cdots,v_{k-1}$是$v_1$到$v_{k-1}$的最短路径。

根据这一点，Dijkstra 设计了一个求v_1到其他各顶点的最短距离。我们以一个例子说明Dijkstra算法的基本思想。

例子 3.1.5 考虑下图所示的有向网络：



其距离矩阵为：

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{bmatrix} \infty & 7 & 1 & \infty & \infty & \infty \\ \infty & \infty & \infty & 4 & \infty & 1 \\ \infty & 6 & \infty & \infty & 2 & 7 \\ \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & 3 & \infty & 5 & \infty & \infty \\ \infty & \infty & \infty & \infty & 3 & \infty \end{bmatrix}$$

我们利用Dijkstra算法求v_1到各点的最短路径。需要三个集合（或数组）来记录求解过程中的信息。

(1) $\overline{S}$记录尚未求得最短路径顶点的集合；

(2) $U=(u_i)_{1\le i\le n}$是以顶点编号（顶点v_i的编号是i）为下标的数组，u_i记录v_1到v_i的最短距离；

(3) $Q=(q_i)_{1\le i\le n}$是以顶点编号为下标的数组，q_i记录v_1到v_i的最短路径中的v_i的直接前趋。

开始时，自然$\overline{S}=\{v_2,v_3,v_4,v_5,v_6\}$（表示$v_2$到$v_6$都没有求得最短路径），而$u_1=0$，$u_i=d_{1i}(2\le i\le n=6)$，也即：

$$U=\langle\infty,7,1,\infty,\infty,\infty\rangle$$

而对任意的$1\le i\le n$，都令$q_i=v_1$，即目前考虑的可能前趋都是v_1：

$$Q=\langle v_1,v_1,v_1,v_1,v_1,v_1\rangle$$

或者说，按照耿素云教材，我们使用下表给出U和Q：

	v_1	v_2	v_3	v_4	v_5	v_6
第0步	$\boxed{0}$	$7/v_1$	$1/v_1$	∞/v_1	∞/v_1	∞/v_1

上表中的值都是以u_i/q_i的形式给出。其中以盒子框住的u_1，这只是表示v_1到v_1的最短路径约定为0，不用再计算。而u_i是无穷时，q_i的值实际上暂时没有参考意义。

在U中那些上未求得最短路径顶点的编号所对应的u_i值中的最小者u_j，即令

$$u_j=\min_{i\in\overline{S}}u_i$$

具体到上面的有向网络的距离矩阵，我们得到最小的是$u_3=1$，这表示v_1到v_3的最短路径就是边$e=\langle v_1,v_3\rangle$，其长度就是1，因此将v_3从$\overline{S}$中删除，同时修改U中的u_i值，只针对v_3在$\overline{S}$中的邻接顶点即可，也就是说，v_1原来不能到达的顶点，现在可能可经过v_3到达了，或者说虽然原来v_1能到达的顶点的路径长度可能比以v_3为中间顶点的路径长度要短。

在这里，我们看到v_3的邻接顶点有v_2, v_5, v_6，比较$u_2 = 7$和$u_3 + d_{32} = 7$，因此仍保持u_2的值为7；比较$u_5 = \infty$和$u_3 + d_{35} = 3$，因此将u_5修改为3，相应地将q_5的值改为v_3（即v_1经过v_3可以比较快地到达v_5）；比较$u_6 = \infty$和$u_3 + d_{36} = 8$，因此将u_6修改为8，相应地将q_6的值也修改为v_3，因此得到：

$$\overline{S} = \{v_2, v_4, v_5, v_6\} \qquad U = \langle 0, 7, 1, \infty, 3, 8\rangle \qquad Q = \langle v_1, v_1, v_1, v_1, v_3, v_3\rangle$$

到此为至的计算结果也可用下表给出：

	v_1	v_2	v_3	v_4	v_5	v_6
第0步	$\boxed{0}$	$7/v_1$	$1/v_1$	∞/v_1	∞/v_1	∞/v_1
第1步		$7/v_1$	$\boxed{1}/v_1$	∞/v_1	$3/v_3$	$8/v_3$

上表以盒子框住u_3的值，表示v_3已经求得最短路径，在后续的步骤中我们不用再考虑它。

下一步我们考虑u_2, u_4, u_5, u_6中的最小者，显然最小者是$u_5 = 3$，因此将5从$\overline{S}$中删除，并修改v_5在$\overline{S}$中的邻接顶点，即v_2和v_4所对应的u_2和u_4的值，即比较$u_2 = 7$和$u_5 + d_{52} = 6$，将u_2的值修改为6，q_2的值修改为v_5；比较$u_4 = \infty$和$u_5 + d_{54} = 8$，将u_4的值修改为8，q_4的值修改为v_5，因此得到：

$$\overline{S} = \{v_2, v_4, v_6\} \qquad U = \langle 0, 6, 1, 8, 3, 8\rangle \qquad Q = \langle v_1, v_5, v_1, v_5, v_3, v_3\rangle$$

到此为至的计算结果也可用下表给出：

	v_1	v_2	v_3	v_4	v_5	v_6
第0步	$\boxed{0}$	$7/v_1$	$1/v_1$	∞/v_1	∞/v_1	∞/v_1
第1步		$7/v_1$	$\boxed{1}/v_1$	∞/v_1	$3/v_3$	$8/v_3$
第2步		$6/v_5$		$8/v_5$	$\boxed{3}/v_3$	$8/v_3$

由于v_3已经不在$\overline{S}$中，所以为了简洁起见，在新增的一行中不再给出v_3对应的u_3和q_3值。当然耿素云教材给出的表更为简洁：除了在获得最短路径的顶点处标记上Q值以外，其他地方均不给出Q值。

下一步考虑u_2, u_4, u_6中的最小者，即$u_2 = 6$，将2从$\overline{S}$中删除，并修改v_2的邻接点v_4和v_6所对应的u_4和u_6的值，即比较$u_4 = 8$和$u_2 + d_{24} = 10$，仍保持u_4和q_4的值；比较$u_6 = 8$和$u_2 + d_{26} = 7$，修改u_6的值为7，q_6的值为v_2，因此得到：

$$\overline{S} = \{v_4, v_6\} \qquad U = \langle 0, 6, 1, 8, 3, 7\rangle \qquad Q = \langle v_1, v_5, v_1, v_5, v_3, v_2\rangle$$

到此为至的计算结果也可用下表给出：

	v_1	v_2	v_3	v_4	v_5	v_6
第0步	$\boxed{0}$	$7/v_1$	$1/v_1$	∞/v_1	∞/v_1	∞/v_1
第1步		$7/v_1$	$\boxed{1}/v_1$	∞/v_1	$3/v_3$	$8/v_3$
第2步		$6/v_5$		$8/v_5$	$\boxed{3}/v_3$	$8/v_3$
第3步		$\boxed{6}/v_5$		$8/v_5$		$7/v_2$

最后考虑u_4, u_6中的最小者，即$u_6 = 7$，将6从$\overline{S}$中删除，并修改v_6的邻接顶点，但v_6的邻接顶点只有v_5，而v_5已经不再$\overline{S}$中，所以无需修改U和Q。最后考虑删除u_4时，u_4的邻接顶点都已不在$\overline{S}$中，因此这时不会对U和Q产生影响，从而最后得到的U和Q如下：

$$U = \langle 0, 6, 1, 8, 3, 7\rangle \qquad Q = \langle v_1, v_5, v_1, v_5, v_3, v_2\rangle$$

到此为至的计算结果也可用下表给出：

	v_1	v_2	v_3	v_4	v_5	v_6
第0步	$\boxed{0}$	$7/v_1$	$1/v_1$	∞/v_1	∞/v_1	∞/v_1
第1步		$7/v_1$	$\boxed{1}/v_1$	∞/v_1	$3/v_3$	$8/v_3$
第2步		$6/v_5$		$8/v_5$	$\boxed{3}/v_3$	$8/v_3$
第3步		$\boxed{6}/v_5$		$8/v_5$		$7/v_2$
第4步				$8/v_5$		$\boxed{7}/v_2$
第5步				$\boxed{8}/v_5$		

U给出了v_1到各顶点之间的最短距离，即v_1到$v_i(2 \le i \le 6)$的最短距离是u_i，而Q可给出v_1到各顶点的最短路径，例如v_1到v_2的最短路径可这样得到：q_2的值是v_5，而q_5的值是v_3，q_3的值是v_1，因此v_1到v_2的最短路径是$v_1 \to v_3 \to v_5 \to v_2$。注意，如果$u_i$的值是$\infty$，则表示$v_1$到$v_i$没有路径。总之，$v_1$到各顶点之间的最短路径和最短距离如下：

v_1到v_2的最短距离是 6, 相应的路径是 $v_1 \to v_3 \to v_5 \to v_2$

v_1到v_3的最短距离是 1, 相应的路径是 $v_1 \to v_3$

v_1到v_4的最短距离是 8, 相应的路径是 $v_1 \to v_3 \to v_5 \to v_4$

v_1到v_5的最短距离是 3, 相应的路径是 $v_1 \to v_3 \to v_5$

v_1到v_6的最短距离是 7, 相应的路径是 $v_1 \to v_3 \to v_5 \to v_2 \to v_6$

显然利用Dijkstra算法，我们可以求得任意两个顶点之间的最短距离。如果要直接求有向网络中任意两个顶点之间的最短距离，与利用邻接矩阵计算有向图的可达矩阵类似，我们也可通过修改矩阵乘法计算任意两个顶点之间的最短距离，进一步，也可通过修改计算可达矩阵的Warshall算法计算任意两个顶点之间的最短距离。

例子 3.1.6 同样考虑戴一奇教材p25图2.23所示的有向网络。其距离矩阵D为：

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{bmatrix} \infty & 7 & 1 & \infty & \infty & \infty \\ \infty & \infty & \infty & 4 & \infty & 1 \\ \infty & 6 & \infty & \infty & 2 & 7 \\ \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & 3 & \infty & 5 & \infty & \infty \\ \infty & \infty & \infty & \infty & 3 & \infty \end{bmatrix}$$

我们通过修改矩阵乘法计算任意两个顶点之间的最短距离。

我们知道，距离矩阵$D = [d_{ij}]$中的元素d_{ij}给出了v_i到v_j的长度为1的带权路径长度，而计算矩阵乘法$D \times D = D^{(2)} = [d_{ij}^{(2)}]$时$d_{ij}^{(2)}$的值是通过考察对任意的顶点$v_k$，$v_i$经过$v_k$再到$v_j$的长度为2的路径，如果要考虑最小的带权路径长度，我们在计算$d_{ij}^{(2)}$时可使用如下方法：

$$d_{ij}^{(2)} = \min\{d_{ij}, \min_{1 \le k \le n}(d_{ik} + d_{kj})\}$$

这里+是普通实数加法，但约定对任意的实数r有$r + \infty = \infty$。实际上，$d'_{ij} = \min_{1 \le k \le n}(d_{ik} + d_{kj})$是$v_i$到$v_j$的长度为2的最小带权路径长度，从而$d_{ij}^{(2)} = \min\{d_{ij}, d'_{ij}\}$就是$v_i$到$v_j$长度不超过2的最

小带权路径长度。类似地计算$D^{(3)}=D^{(2)}\times D$等等，一直到$D^{(n-1)}$，得到长度不超过$n-1$的最小带权路径长度。显然，任意两个顶点之间的最短路径长度不会超过$n-1$，因此$D^{(n-1)}$将给出任意两个顶点之间的最短距离。

根据这个方法，我们针对上述有向网络进行计算，得到：

$$D^{(2)}=\begin{bmatrix}\infty & 7 & 1 & 11 & 3 & 8\\ \infty & \infty & \infty & 4 & 4 & 1\\ \infty & 5 & \infty & 7 & 2 & 7\\ \infty & \infty & \infty & \infty & \infty & \infty\\ \infty & 3 & \infty & 5 & \infty & 4\\ \infty & 6 & \infty & 8 & 3 & \infty\end{bmatrix}$$

$$D^{(3)}=\begin{bmatrix}\infty & 6 & 1 & 8 & 3 & 8\\ \infty & 7 & \infty & 4 & 4 & 1\\ \infty & 5 & \infty & 7 & 2 & 6\\ \infty & \infty & \infty & \infty & \infty & \infty\\ \infty & 3 & \infty & 5 & 7 & 4\\ \infty & 6 & \infty & 8 & 3 & 7\end{bmatrix}$$

$$D^{(4)}=D^{(5)}=\begin{bmatrix}\infty & 6 & 1 & 8 & 3 & 7\\ \infty & 7 & \infty & 4 & 4 & 1\\ \infty & 5 & \infty & 7 & 2 & 6\\ \infty & \infty & \infty & \infty & \infty & \infty\\ \infty & 3 & \infty & 5 & 7 & 4\\ \infty & 6 & \infty & 8 & 3 & 7\end{bmatrix}$$

最后矩阵$D^{(5)}$的对角线数字，如果不是∞，则意味着v_i到v_i有一条最短的回路，不过这对于求两顶点之间的最短距离没有什么意义，因为我们可以约定任意顶点到它自己的最短距离是0。

例子 3.1.7 上述计算虽然直接，但比较繁琐，Warshall 的计算可达矩阵的方法也可用来计算任意两个顶点之间的最短距离。Warshall 算法的基本思想是：

第k次循环计算的$d_{ij}^{(k)}$是顶点v_i到v_j之间可能经过$v_1,v_2,\cdots,v_k$这k个顶点的路径中带权长度最短的路径长度，从而第n次循环时，$d_{ij}^{(n)}$将给出顶点v_i到v_j之间可能经过所有顶点的路径中带权长度最短的路径长度，也即$d_{ij}^{(n)}$是顶点v_i到v_j之间的最短距离。

同样考虑戴一奇教材p25图2.23所示的有向网络。其距离矩阵D为：

$$\begin{array}{c}\\ v_1\\ v_2\\ v_3\\ v_4\\ v_5\\ v_6\end{array}\begin{bmatrix}\infty & 7 & 1 & \infty & \infty & \infty\\ \infty & \infty & \infty & 4 & \infty & 1\\ \infty & 6 & \infty & \infty & 2 & 7\\ \infty & \infty & \infty & \infty & \infty & \infty\\ \infty & 3 & \infty & 5 & \infty & \infty\\ \infty & \infty & \infty & \infty & 3 & \infty\end{bmatrix}$$

第一次循环是对任意的$1\le i\le n,1\le j\le n$，用d_{ij}与$d_{i1}+d_{1j}$之间的小者修正d_{ij}，由于上述有向网

络对任意的i都有$d_{i1}=\infty$，因此第一次循环不会对距离矩阵作任何修改，即

$$D^{(1)}=D=\begin{bmatrix}\infty & 7 & 1 & \infty & \infty & \infty\\ \infty & \infty & \infty & 4 & \infty & 1\\ \infty & 6 & \infty & \infty & 2 & 7\\ \infty & \infty & \infty & \infty & \infty & \infty\\ \infty & 3 & \infty & 5 & \infty & \infty\\ \infty & \infty & \infty & \infty & 3 & \infty\end{bmatrix}$$

第二次循环是对任意的$1\le i\le n, 1\le j\le n$，用$d_{ij}^{(1)}$与$d_{i2}^{(1)}+d_{2j}^{(1)}$之间的小者修正$d_{ij}$得到$d_{ij}^{(2)}$，为此，只要将第2行中非$\infty$的$d_{2j}^{(1)}$与第2列中非$\infty$的$d_{i2}^{(1)}$相加，并与$d_{ij}^{(1)}$比较即可得到$d_{ij}^{(2)}$，也即，令：

$$d_{ij}^{(2)}=\min\{d_{ij}^{(1)}, d_{i2}^{(1)}+d_{2j}^{(1)}\}$$

针对上述有向网络，我们得到：

$$D^{(2)}=\begin{bmatrix}\infty & 7 & 1 & 11 & \infty & 8\\ \infty & \infty & \infty & 4 & \infty & 1\\ \infty & 6 & \infty & 10 & 2 & 7\\ \infty & \infty & \infty & \infty & \infty & \infty\\ \infty & 3 & \infty & 5 & \infty & 4\\ \infty & \infty & \infty & \infty & 3 & \infty\end{bmatrix}$$

$$D^{(4)}=D^{(3)}=\begin{bmatrix}\infty & 7 & 1 & 11 & 3 & 8\\ \infty & \infty & \infty & 4 & \infty & 1\\ \infty & 6 & \infty & 10 & 2 & 7\\ \infty & \infty & \infty & \infty & \infty & \infty\\ \infty & 3 & \infty & 5 & \infty & 4\\ \infty & \infty & \infty & \infty & 3 & \infty\end{bmatrix}$$

$$D^{(5)}=\begin{bmatrix}\infty & 6 & 1 & 8 & 3 & 7\\ \infty & \infty & \infty & 4 & \infty & 1\\ \infty & 5 & \infty & 7 & 2 & 6\\ \infty & \infty & \infty & \infty & \infty & \infty\\ \infty & 3 & \infty & 5 & \infty & 4\\ \infty & 6 & \infty & 8 & 3 & 7\end{bmatrix}$$

$$D^{(6)}=\begin{bmatrix}\infty & 6 & 1 & 8 & 3 & 7\\ \infty & 7 & \infty & 4 & 4 & 1\\ \infty & 5 & \infty & 7 & 2 & 6\\ \infty & \infty & \infty & \infty & \infty & \infty\\ \infty & 3 & \infty & 5 & 7 & 4\\ \infty & 6 & \infty & 8 & 3 & 7\end{bmatrix}$$

得到的结果$D^{(6)}$与上面得到的$D^{(5)}$相同，都给出了任意两顶点之间的最短距离，但显然，使用Warshall算法计算更为简单。

上面考虑的是有向网络，类似地也可定义无向网络：

定义 3.1.8 给定无向图$G=(V,E)$，设$V=\{v_1,v_2,\cdots,v_n\}$。如果给每一条有向边$e=\langle v_i,v_j\rangle$赋一个非负实数权w_{ij}，则称图G为**无向网络**。

同样我们考虑的无向网络都是简单无向图，即没有多重边，也没有环。Dijkstra算法可直接用于求无向网络中任意一个顶点到其他所有顶点之间的最短距离，而根据距离矩阵使用矩阵乘法或Warshall算法来计算任意两个顶点之间的最短距离时，只要将距离矩阵中的d_{ij}值设为与d_{ji}值一样即可。

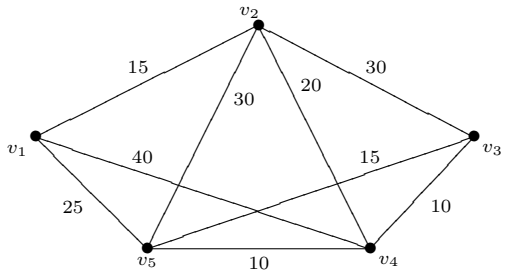
3.2 最小生成树

这一节要考虑的主题是求无向连通网络的最小生成树：

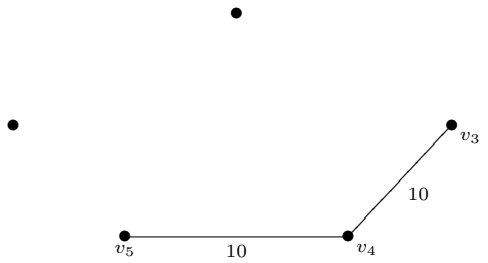
定义 3.2.1 无向连通网络G的所有生成树中树枝权值之和最小的生成树称为G的**最小生成树**（或最短树）。

Kruskal提供了一个求无向连通网络$G=(V,E)$的最小生成树的算法，其基本思想是将G的所有边按权值从小到大排序，然后依次考察每条边作为最小生成树T的候选树枝，如果某条边不与已经选中的边构成回路，则合乎要求，否则就被放弃而考虑下一条边，直到所有的边考虑完毕，或更简单地，只要T中已经有$|V|-1$条边即可终止算法。

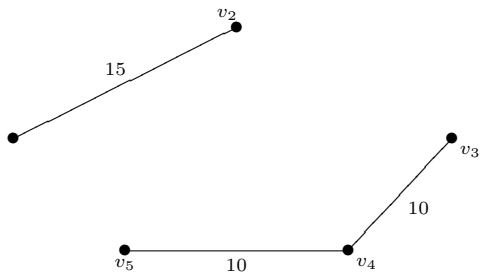
例子 3.2.2 我们以例子来说明Kruskal算法。考虑下图所示的无向网络：



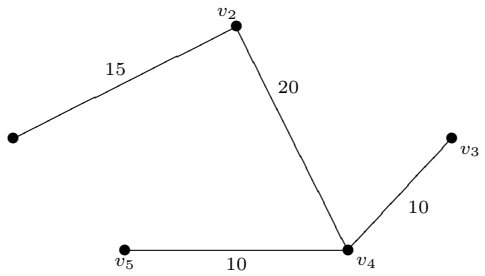
根据Kruskal算法的选边的顺序如下。首先选择权最小的边(v_5,v_4)，然后选择边(v_4,v_3)，这两个边的权都是10，而且不构成回路，我们初步得到的生成树：



下一步考虑的边的权是15，这时如果加入边(v_5,v_3)，则会构成回路，所以下一步只能加入边(v_1,v_2)，得到的生成树如下：



下一步考虑的边的权是20，即加入边(v_2, v_4)，这时就已经得到了$4 = |V| - 1$条边，从而算法终止，得到如下的最小生成树：

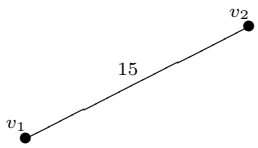


Prim提供了另一种得到最小生成树的算法，其基本思想是：首先任选一个顶点v_0构成顶点集V'，然后不断在$V - V'$中选一条到V'中某顶点（例如是v）最短的边（例如是(v, u)），也即

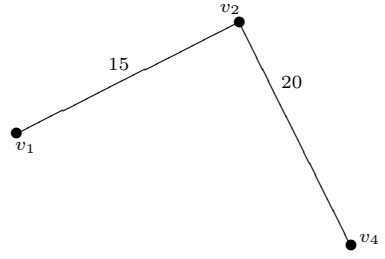
$$w(v, u) = \min_{t \in V', w \in V - V'} w(t, w)$$

将边(v, u)加入到树T，并令$V' = V' \cup \{u\}$，直到$V' = V$。

例子 3.2.3 同样考虑上面的无向网络，使用Prim算法选择顶点和边的顺序如下：假定第一步选择顶点v_1，令$V' = \{v_1\}$，与v_1距离最短的边是(v_1, v_2)，加入边(v_1, v_2)到生成树T，加入顶点v_2到V'，得到：

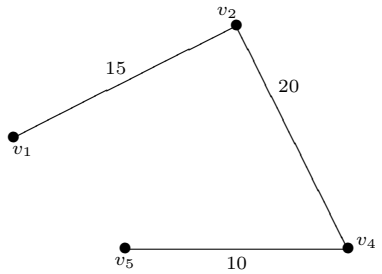


显然与v_1, v_2距离最短的边是(v_2, v_4)，加入边(v_2, v_4)到生成树T，加入顶点v_4到V'，得到：

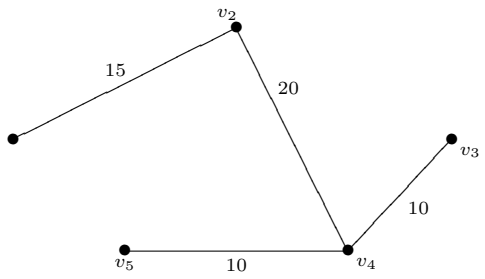


下一步与v_1, v_2, v_4距离最短的边是(v_5, v_4)（或(v_3, v_4)也类似），加入边(v_5, v_4)到生成树T，加入顶

点v_5到V'，得到：



下一步与v_1, v_2, v_4, v_5距离最短的边是(v_3, v_4)，加入边(v_3, v_4)到生成树，加入顶点v_3到V'，得到：



这时$V' = V$，算法终止。这里我们看到得到的生成树与Kruskal算法的结果一样，但由于最小生成树是不惟一的，因此这两个算法得到的最小生成树不一定相同。

练习 3.2.4 根据上面例子，理解教材中对这两个算法的描述，并思考这两个算法的关键步骤（或最复杂步骤，最费时间步骤）各是什么。教材中还给出了其他两种算法，同学们可自行学习。

3.3 关键路径

在许多工程项目管理中，我们都会遇到与关键路径有关的问题。通常一项工程都会分为很多工序，这些工序之间有一些约束，最简单的约束是时间约束，例如工序j必须在工序$i_1, i_2, \cdots, i_k$完成之后才能进行等等。

我们以后将在软件工程课程中也会遇到与关键路径有关的问题，因为按照软件生命周期，软件开发项目通常分为需求分析、总体设计、详细设计、实现、单元测试、集成测试、验收测试等各个阶段，而且当软件规模比较大的时候，又会分成几个子系统，每个子系统都有这些阶段，从而个子系统的各阶段之间存在复杂的时间约束关系，软件项目经理通常需要找出其中的关键步骤，从而控制软件开发的进度。我们先引入一些概念来对这种与关键路径有关问题进行建模。

定义 3.3.1 PERT图是一个有向连通网络$G = (V, E)$，且：

(1) 每条边表示一个作业（工序），边的非负实数权表示完成该作业所需的时间；

(2) 每个顶点表示以该顶点为起点的作业的开始，也表示以该顶点为终点的作业结束；

(3) 作业开始条件：某条有向边$e = \langle v, u \rangle$所代表的作业可以开始当且仅当以v为终点的有向边所代表的作业全部完成；

(4) G没有有向回路，也没有环；

(5) G中有且仅有一个顶点的入度为0，称该顶点为源点；同时G中有且仅有一个顶点的出度为0，称该顶点为汇点。

实际上，我们有：

引理 3.3.2 没有回路的有向连通网络G存在入度为0的顶点，同时也存在出度为0的顶点。

证明 考虑G的一条极长有向道路Γ，设其起点为v_1，终点为v_n。若$d^-(v_1) \neq 0$，则存在边$\langle\rangle v_i, v_1)$，若$v_i \in \Gamma$，则G存在有向回路，若$v_i \notin \Gamma$，则Γ不是极长道路，总之必有$d^-(v_1) = 0$。类似地也有$d^+(v_n) = 0$。 □

定理 3.3.3 设$G = (V, E)$不存在有向回路，可以将G的顶点重新编号为$v'_1, v'_2, \cdots, v'_n$使得对任意边$(v'_i, v'_j) \in E$，都有$i < j$。

根据这个定理，我们总是假定PERT图的顶点编号满足上述定理给出的条件。对于PERT图，我们主要关心其关键路径，以及每个作业的最早启动时间和最晚启动时间：

定义 3.3.4 给定PERT图$G = (V, E)$，G中从源点到汇点最长带权路径称为关键路径。关键路径的长度T为完成整个任务（即包括所有作业）所必需的最少时间。关键路径上的边所代表的作业称为关键作业。

对于某个作业$e = \langle v_k, -\rangle$，源点到该作业起点$v_k$的最长路径长度$\pi_k$称为该作业的最早起动时间。设$v_k$到汇点的最长路径长度为$\pi'_k$，则该作业可最晚于$\tau_k = T - \pi'_k$时间启动而不影响整个任务完成的预期时间$T$，$\tau_k$称为该作业的最晚启动时间。

很显然，作业$e = \langle v_k, -\rangle$是关键作业当且仅当$\tau_k = \pi_k$，因为这时$e$是处于从源点到汇点最长带权路径上。所以，我们可通过求每个作业的最早启动时间和最晚启动时间而得到PERT图的关键路径。

定理 3.3.5 设PERT图$G = (V, E)$的顶点编号$v_1, v_2, \cdots, v_n$使得边$e = \langle v_i, v_j\rangle \in E$蕴涵$i < j$。令：

$$\pi_1 = 0 \qquad \pi_j = \max\{\pi_l + w(v_l, v_j) \mid 1 \leq l < j\}$$

则$\pi_i (1 \leq i \leq n)$是作业$e = \langle v_i, v_j\rangle$的最早启动时间。这里$w(v_l, v_j)$定义为：

$$w(v_l, v_j) = \begin{cases} \text{边}\langle v_l, v_j\rangle\text{的非负实数权} & \text{若}\langle v_l, v_j\rangle \in E \\ -\infty & \text{否则} \end{cases}$$

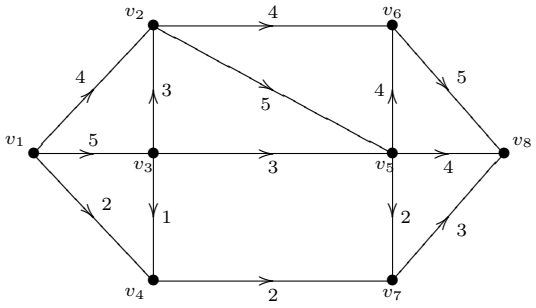
令：

$$\tau_n = \pi_n \qquad \tau_k = \min\{\tau_k, \tau_l - w(v_k, v_l) \mid k < l \leq n\}$$

则$\tau_i (1 \leq i \leq n)$是作业$e = \langle v_i, v_j\rangle$的最晚启动时间。

我们无需详细证明上述定理，下面以一个简单的例子说明如何给出PERT图中每个作业的最早启动时间和最晚启动时间，从而得到整个PERT图的关键路径。

例子 3.3.6 考虑下面的PERT图：



注意顶点可看作一种状态，在该状态，所有以它为终点的作业都已经完成，而所有以它为起点的作业准备启动。为计算每个作业（边）的最早启动时间和最晚启动时间，我们实际上计算每个顶点的最早启动时间和最晚启动时间，也即，某顶点的最早启动时间（最晚启动时间）就是所有以该顶点为起点的作业的最早启动时间（最晚启动时间）。

我们先计算每个顶点的最早启动时间。最早启动时间的计算是从源点开始计算，源点的最早启动时间为0，对于顶点v_i，只有在所有以v_i为终点的边$\langle v_l, v_i\rangle$的起点v_j的最早启动时间都计算出来之后，才能计算v_i的最早启动时间，而且v_i的最早启动时间π_i是：

$$\pi_i = \max\{\pi_j + w(v_j, v_i) \mid \langle v_j, v_i\rangle \in E\}$$

对于上述PERT图，首先$\pi_1 = 0$，下一步，我们看到，只有v_3能够计算最早启动时间，且：

$$\pi_3 = \max\{\pi_1 + w(v_1, v_3)\} = \max\{0 + 5\} = 5$$

再下一步，我们看到现在可以计算v_2和v_4的最早启动时间：

$$\pi_2 = \max\{\pi_1 + w(v_1, v_2),\ \pi_3 + w(v_3, v_2)\} = \max\{4, 5 + 3\} = 8$$
$$\pi_4 = \max\{\pi_1 + w(v_1, v_4),\ \pi_3 + w(v_3, v_4)\} = \max\{2, 5 + 1\} = 6$$

再下一步，我们看到现在可以计算v_5的最早启动时间：

$$\pi_5 = \max\{\pi_3 + w(v_3, v_5),\ \pi_2 + w(v_2, v_5)\} = \max\{5 + 3, 8 + 5\} = 13$$

再下一步，我们看到现在可以计算v_6, v_7的最早启动时间：

$$\pi_6 = \max\{\pi_2 + w(v_2, v_6),\ \pi_5 + w(v_2, v_6)\} = \max\{8 + 4, 13 + 4\} = 17$$
$$\pi_7 = \max\{\pi_4 + w(v_4, v_7),\ \pi_5 + w(v_5, v_7)\} = \max\{6 + 2, 13 + 2\} = 15$$

最后可计算v_8的最早启动时间：

$$\begin{aligned}\pi_8 &= \max\{\pi_5 + w(v_5, v_8),\ \pi_6 + w(v_6, v_8),\ \pi_7 + w(v_7, v_8)\} \\ &= \max\{13 + 4, 17 + 5, 15 + 3\} = 22\end{aligned}$$

这就得到源点v_1到汇点v_8的最长带权路径长度为22，这也是完成整个任务的必需时间。

现在可以计算每个顶点的最晚启动时间。最晚启动时间的计算是从汇点开始计算，汇点的最晚启动时间等于其最早启动时间，对于以顶点v_i，只有在所有以v_i为起点的边$\langle v_i, v_j\rangle$的终点v_j的最晚启动时间都计算出来之后，才能计算v_i的最晚启动时间，而且v_i的最晚启动时间τ_i是：

$$\tau_i = \min\{\pi_j - w(v_i, v_j) \mid \langle v_i, v_j\rangle \in E\}$$

对于上述PERT图，首先$\tau_8 = \pi_8 = 22$，下一步，我们看到，可以计算v_6, v_7的最晚启动时间：

$$\tau_6 = \min\{\tau_8 - w(v_6, v_8)\} = \min\{22 - 5\} = 17$$
$$\tau_7 = \min\{\tau_8 - w(v_7, v_8)\} = \min\{22 - 3\} = 19$$

再下一步，我们看到现在可以计算v_5的最晚启动时间：

$$\begin{aligned}\tau_5 &= \min\{\tau_6 - w(v_5, v_6),\ \tau_7 - w(v_5, v_7),\ \tau_8 - w(v_5, v_8)\} \\ &= \min\{17 - 4,\ 19 - 2,\ 22 - 4\} = 13\end{aligned}$$

再下一步，我们看到可以计算v_2, v_4的最晚启动时间：

$$\tau_2 = \min\{\tau_6 - w(v_2, v_6),\ \tau_5 - w(v_2, v_5)\} = \min\{17 - 4, 13 - 5\} = 8$$
$$\tau_4 = \min\{\tau_7 - w(v_4, v_7)\} = \min\{19 - 2\} = 17$$

再下一步，我们看到可以计算v_3的最晚启动时间：

$$\begin{aligned}\tau_3 &= \min\{\tau_2 - w(v_3, v_2),\ \tau_4 - w(v_3, v_4),\ \tau_5 - w(v_3, v_5)\} \\ &= \min\{8 - 3, 17 - 1, 13 - 3\} = 5\end{aligned}$$

最后可以计算源点v_1的最晚启动时间（实际上，源点的最晚启动时间应该是0）：

$$\begin{aligned}\tau_1 &= \min\{\tau_2 - w(v_1, v_2),\ \tau_3 - w(v_1, v_3),\ \tau_4 - w(v_1, v_4)\} \\ &= \min\{8 - 4, 5 - 5, 17 - 2\} = 0\end{aligned}$$

这样我们得到了各个顶点的最早启动时间和最晚启动时间：

顶点V	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
最早启动时间π	0	8	5	6	17	13	15	22
最晚启动时间τ	0	8	5	17	17	13	19	22
缓冲时间τ	0	0	0	11	0	0	4	0

根据各个顶点的最早启动时间和最晚启动时间可以确定PERT图的关键路径，即凡是最早启动时间等于最晚启动时间的顶点就是处于关键路径的顶点。对于上述PERT图，我们得到的关键路径就是：

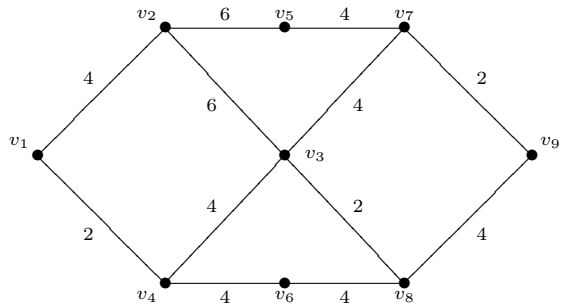
$$\Gamma = v_1 v_3 v_2 v_6 v_5 v_8$$

处于关键路径上的作业（边）就是关键作业，这些作业的完成时间必须按照预期的时间完成，否则就会影响整个任务的完成时间，而处于非关键路径上的作业$\langle v_i, v_j\rangle$就允许有延误时间，其可以延误的时间等于$\tau_j - \pi_i - w(v_i, v_j)$，例如，作业$\langle v_1, v_4\rangle$的允许延误时间就是$\tau_4 - \pi_1 - w(v_1, v_4) = 15$，也就是说，该作业即使比原来预计的2个时间再晚15个时间也能保证整个任务在22个时间内完成（当然前提是其他作业，特别是作业$\langle v_4, v_7\rangle$不再延迟）。而作业$\langle v_4, v_7\rangle$的允许时间是$\tau_7 - \pi_4 - w(v_4, v_7) = 11$。有时，我们简单地讨论顶点的缓冲时间，顶点$v_i$的缓冲时间等于$\tau_i - \pi_i$，各顶点的缓冲时间如上表所示。

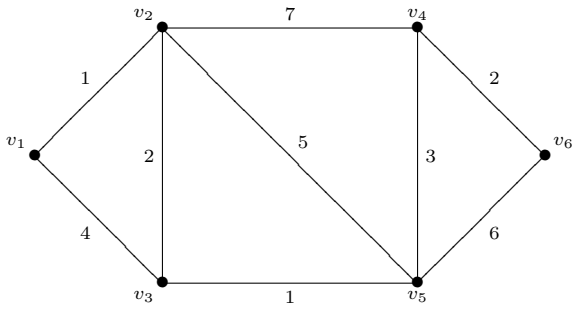
耿素云教材使用的术语是最早完成时间和最迟完成时间，这实际上与上面的说法没有本质差别，因为一个顶点实际上既代表上一工序的完成，也代表下一工序的启动。

作 业

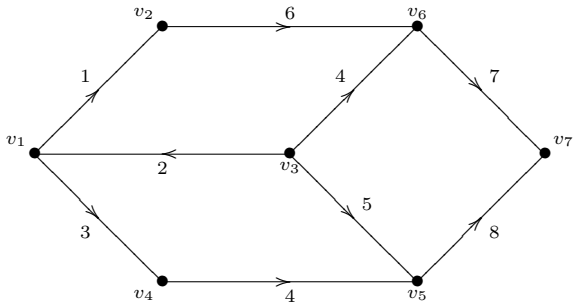
作业 3.1 使用Dijkstra算法求下面带权图中v_1到v_9的最短路径：



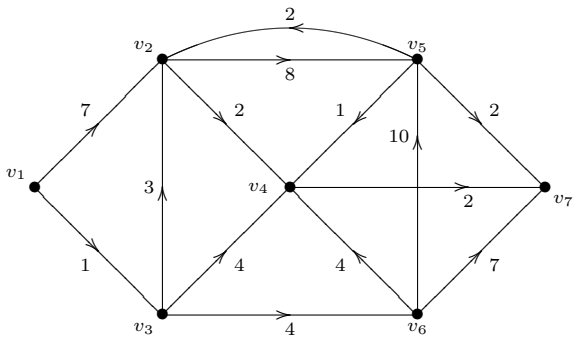
作业 3.2 写出下面的带权无向图的距离矩阵，并使用Dijkstra算法求下面图中v_1到其余各顶点的最短路径，然后用Warshall算法求任意两个顶点之间的最短距离：



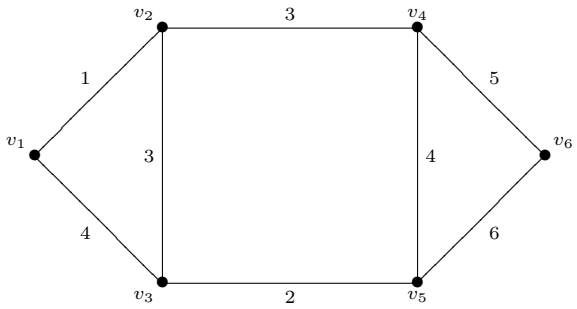
作业 3.3 写出下面的带权有向图的距离矩阵，并使用Dijkstra算法求v_1到各点的最短距离，然后用Warshall算法求任意两点之间的最短距离：



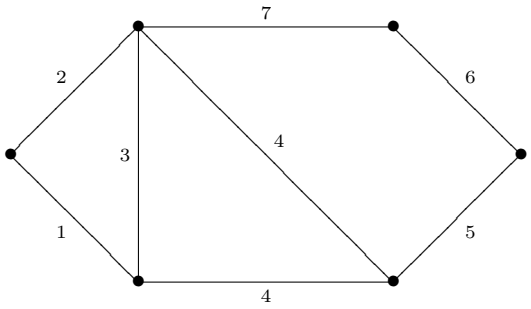
作业 3.4 使用Dijsktra算法计算v_1到v_7的最短路径：



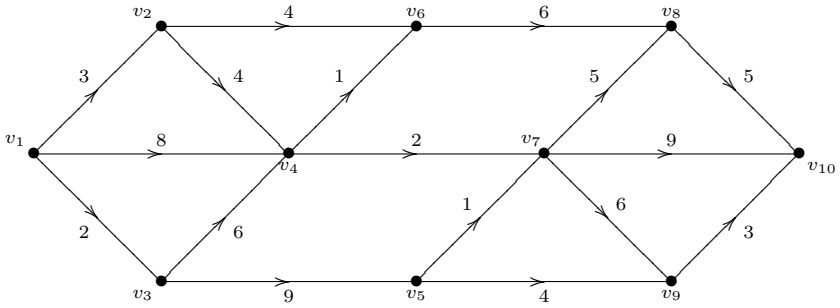
作业 3.5 分别使用Kruskal算法和Prim算法求下图的最小生成树：



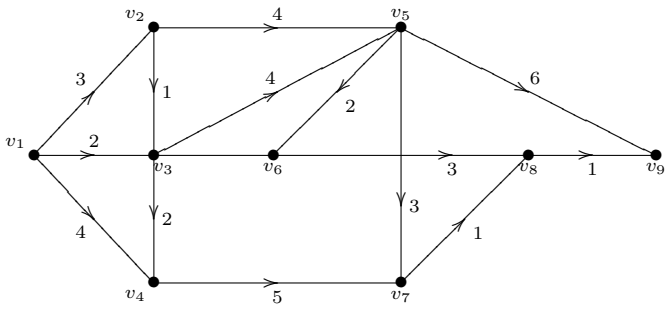
作业 3.6 分别使用Kruskal算法和Prim算法求下图的最小生成树：



作业 3.7 对于下面的PERT图，计算每个顶点的最早启动时间和最晚启动时间，并给出其关键路径：



作业 3.8 对于下面的PERT图，计算每个顶点的最早启动时间和最晚启动时间，并给出其关键路径：



第四章　平面图与着色

在实际问题中有时要涉及到图的平面性的讨论，比如印刷电路板的设计、大规模继承电路的布局布线等，都离不开图的平面性研究。著名的四色猜想也属于平面性范畴。这一章讨论平面图的基本性质，以及与平面图相关的着色问题。

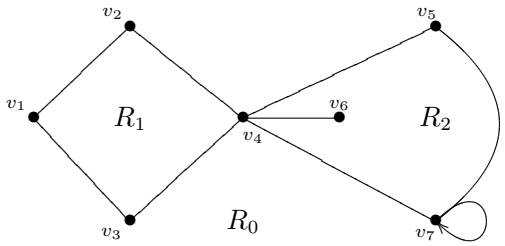
4.1　平面图及其性质

定义 4.1.1 给定图$G=(V,E)$，若能将其画在平面上，且任意两条边的交点只能是G的顶点，则称G可嵌入平面，或称G是可平面的。可平面图在平面上的一个嵌入称为一个平面图。

实际上，树就是一类重要的平面图，因为树没有回路，也就是说，树没有一个封闭的区域，因此总能通过适当排列其顶点，使得任意两条边之间都不相交。

定义 4.1.2 设G是平面图，由它的若干边包围而成，且其中不含图的顶点和边的区域称为G的一个面或域。包围域R的所有边组成的回路称为该面的边界，回路长度称为该面的度，记为$deg(R)$。由平面图的边包围且无穷大的域称为平面图的外部面，其他的面都称为内部面。

例子 4.1.3 考虑下面的平面图：



这个平面图共有四个面，其中R_0是外部面，R_3由v_7上的环围成，上面没有标出，这些面的边界和度分别是：

R_0 的边界：	$v_1v_2v_4v_5v_7v_7v_4v_3v_1$	$deg(R_0)=8$
R_1 的边界：	$v_1v_2v_4v_3v_1$	$deg(R_1)=4$
R_2 的边界：	$v_4v_5v_7v_4v_6v_4$	$deg(R_2)=5$
R_3 的边界：	v_7v_7	$deg(R_3)=1$

注意v_4和v_6之间的边作为R_2的边界需要计算两次。

类似于握手定理，由于平面图的任意边都作为两个面的边界，因此我们有：

定理 4.1.4 平面图所有面的度之和等于边数的两倍。

在平面图研究中，最重要的是**欧拉公式**：

定理 4.1.5 设平面连通图G有n个顶点，m条边，d个域，则有$n-m+d=2$。

证明 G是连通图，有生成树T，它包含$n-1$条边，不产生回路，因此对T而言只有一个外部面。由于G是平面图，每加入一条余树边，它一定不与其他边相交，也就是说一定是跨在某个面的内部，把该面分成两个面，因此加入$m-n+1$条余树边就得到$m-n+2$个面。 □

推论 4.1.6 设平面图G的连通分支数为k，并有n个顶点，m条边，d个域，则有$n-m+d=k+1$。

利用欧拉公式，我们有：

定理 4.1.7 设G是连通的平面图，且G的各面度数大于等于l，则G的边数m与顶点数n满足：

$$m \leq \frac{l}{l-2}(n-2)$$

证明 设G的面数为r，则由欧拉定理有$r=2+m-n$，从而利用面的度数之和是边数两倍可得：

$$2m \geq l*r \implies 2m \geq l(2+m-n) \implies m \leq \frac{l}{l-2}(n-2)$$

□

推论 4.1.8 定义$g(G)$是图G长度最小的回路的长度，称为图G的**围长**。若连通平面图G的围长大于等于l，则

$$m \leq \frac{l}{l-2}(n-2)$$

证明 因为平面图G的面的度数大于图G的围长。 □

下面考虑极大平面图：

定义 4.1.9 设$G=(V,E)$为简单平面图，$|V|>3$，若对任意$v_i,v_j \in V$且$(v_i,v_j) \notin E$，有$G'=(V,E\cup(v_i,v_j))$为非平面图，即在G的任意两个不相邻顶点之间加一条新边都得到非平面图，则称G为一个**极大平面图**。

根据这个定义，我们看到，这里的“极大性”是针对固定顶点数的图的边的数目而言。很容易证明极大平面图的如下性质：

引理 4.1.10 (1) 极大平面图是连通图。

(2) 极大平面图没有桥。

(3) 极大平面图的每个面都由3条边组成。

(4) 极大平面图有$3d=2m$（d为面数目，m为边数目）。

(5) 设v是极大平面图中任意一顶点，则与v相邻的顶点必构成一个回路；

(6) 对于极大平面图$G=(V,E)$，若$|V|>4$，则$\delta(G)\geq 3$。

证明

(1) 若极大平面图不是连通图，则在分别取两个连通分支上的一个顶点，将这两个顶点相连不会增加面数，也就是说不会改变图的平面性，从而与极大性相矛盾。

(2) 若$e=(v,u)$是极大平面图的一个桥（割边），则v的一个相邻的顶点v'与u不相邻，但显然在v'与u之间增加一条边不会改变图的平面性，从而与极大性矛盾!

(3) 极大平面图的每个面都由3条边组成，否则设G有面R至少有四条边界，不妨设围成该面的回路是$v_1e_1v_2e_2v_3e_3v_4e_4\cdots e_kv_1$。这时若$v_1$与$v_3$不相邻，则在$v_1$与$v_3$之间可在面$R$内增加一条边而不破坏图的平面性，这与$G$是极大平面图矛盾，因此$v_1$与$v_3$相邻，且边$(v_1,v_3)$在面$R$之外，同理$v_2$与$v_4$也相邻，且边$(v_2,v_4)$也在面$R$之外，但这时边$(v_1,v_3)$和边$(v_2,v_4)$必相交，与$G$是平面图矛盾。

(4) 由极大平面图的每个面都由3条边组成，及面的度数之和等于边数两倍可得。

(5) 设与v相邻的顶点不构成一个回路，则在与v相邻的顶点中必存在两个顶点u,w，u与w不相邻，那么这时以边(v,u)和(v,w)为边界的面的度必大于等于4,这与极大平面图的每个面的度为3矛盾!

(6) 因为当$|V|>4$时，与任意一个顶点相邻的顶点都构成一个回路，显然这个回路的长度大于等于3，因此任意顶点的度数大于等于3，即$\delta(G)\geq 3$。

□

进一步，利用欧拉公式我们有：

定理 4.1.11 设极大平面图G有n个顶点，m条边，d个面，则有$m=3n-6$且$d=2n-4$。

证明 将$2m=3d$代入$n-m+d=2$得$n-\frac{3}{2}d+d=2$，即$d=2n-4$，从而$m=3n-6$ □

推论 4.1.12 设简单平面图G有n个顶点，m条边，d个面，则有$m\leq 3n-6$且$d\leq 2n-4$。

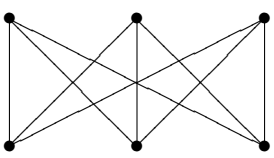
定理 4.1.13 简单平面图G中存在度小于6的顶点。

证明 如果G的任意度都大于等于6，即G的总度数大于等于$6n$，由边数等于顶点度数两倍得，边数大于等于$3n$，与$m\leq 3n-6$矛盾! □

库拉图斯基定理是在理论上判别一个图是否是平面图的重要定理。为介绍该定理，我们先定义二部图的概念：

定义 4.1.14 给定图$G=(V,E)$，如果V能分为两个不相交的非空子集V_1,V_2，即$V_1\neq\varnothing,V_2\neq\varnothing,V_1\cup V_2=V,V_1\cap V_2=\varnothing$，且对$G$的任意边$e=(u,v)\in E$，都有$u\in V_1$且$v\in V_2$，或者$v\in V_1$且$u\in V_2$，也即$G$的任意边的两个端点都在不同的子集，则称$G$是二部图。若$|V_1|=s,|V_2|=r$，且$V_1$的任意顶点都与$V_2$的任意顶点有边，则称该二部图为完全二部图，记为$K_{s,r}$。

对于平面图而言，最重要的是$K_{3,3}$：



定理 4.1.15 完全图K_5和完全二部图$K_{3,3}$都不是平面图。

证明 因为K_5有10条边，5个顶点，而且$g(K_5)=3$，若它是平面图则必须满足：

$$10 \le \frac{3}{3-2}(5-2)=9$$

矛盾，因此K_5不可能是平面图。同样的，不难看到$g(K_{3,3})=4$，而它有6个顶点9条边，若它是平面图则必须满足：

$$9 \le \frac{4}{4-2}(6-2)=8$$

矛盾，因此$K_{3,3}$也不可能是平面图。 □

定义 4.1.16 K_5和$K_{3,3}$分别记为$K^{(1)}$和$K^{(2)}$图，在$K^{(1)}$和$K^{(2)}$图上任意增加一些度为2的顶点之后得到的图称为$K^{(1)}$和$K^{(2)}$型图，统称为K型图。

定理 4.1.17 **库拉图斯基定理**：图G是可平面的当且仅当G不存在K型子图。

这个定理只是在理论上给出了一个图是可平面的充分必要条件，但实际上，很难应用该定理判断一个图是否是可平面的，目前判断一个图是否是可平面的算法比较复杂，这里不再详细讨论。最后我们讨论平面图的对偶图的概念。

定义 4.1.18 给定图$G=(V,E)$，满足下列条件的图$G^*=(V^*,E^*)$称为图G的**对偶图**：G的任一域f_i内有且仅有一点v_i^*；对G的域f_i,f_j的共同边界e_k，画一条$e_k^*=(v_i^*,v_j^*)$且只与e_k交于一点；若e_k完全处于f_i中，则v_i^*有一自环e_k^*且与e_k相交与一点。上述过程称为求对偶图的D过程，得到的对偶图称为原图的拓扑对偶。

对平面图G，D过程构造的G^*是唯一的；对于非平面图，D过程可能不成立；对平面图G，D过程构造的G^*也是平面图；不论图G是否连通，D过程得到的G^*是连通的；若图G连通，且存在G^*，则$(G^*)^*=G$；对图G，若存在G^*，则G中回路相对应于G^*中割集，G中割集相对应于G^*中回路；若$G\simeq G^*$，称G为**自对偶图**。

定理 4.1.19 图G有对偶图当且仅当G是平面图。

定理 4.1.20 对图G施行D过程得到G^*，设n,m,d和n^*,m^*,d^*分别表示G和G^*的顶点数、边数及域数，则有：

$$m^*=m \qquad n^*=d \qquad n=d^*$$

定理 4.1.21 设G为平面图，施行D过程得到G^*，设n,m,d和n^*,m^*,d^*分别表示G和G^*的结点数、边数及域数，k为G的连通分支数，则有：$d^*=n-k+1$

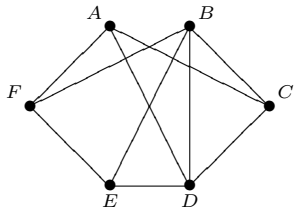
4.2 图的着色

图的着色包括顶点着色、边着色以及平面图的面着色，这里只讨论简单无向图的顶点着色。图的着色可用于解决一些实际问题：

例子 4.2.1 耿素云教材[1]p306 例12.4给出了一个考试安排问题：某系二年级学生共选修全校性的选修课程n门，期末考试前要将这n门课程先考完，要求每天每个学生至多只能参加一门课程的考试，问至少需要几天才能使每个学生将所选的课程都考完？

为解决这个问题，我们可以使用图来建立一个模型。设$V=\{c_1,c_2,\cdots,c_n\}$为课程集合，设$S(c_i)$为学习课程c_i的学生集合，若$S(c_i)\cap S(c_j)\neq\varnothing, i\neq j$，我们就在$c_i$和$c_j$之间连一条边，这样得到一个以$V$为顶点集的无向图，对这个图的顶点进行着色，使得相邻顶点着不同颜色，那么需要的最少颜色数就是使每个学生都考完其选修课程所需要的天数。

例子 4.2.2 戴一奇教材[3]p83页例4.6.1给出了一个货物存储问题：六种货物要存放到仓库，它们之间的关系如下图：



上图中，顶点A,B,C,D,E,F是货物，两个顶点有边相连表示这两种货物不能存放在同一个仓库。此问题与考试安排问题类似，也可用图的着色来解决，对这个图的顶点进行正常着色所需要的最少颜色数就是存放这些货物所需的最少仓库数，而着色方案的数目就是存放货物的方案数。

下面我们先讨论图的顶点着色的一些基本性质，然后再以一个具体例子说明这一类问题的求解方法。

定义 4.2.3 图$G=(V,E)$的一个k顶点着色指用k种颜色对G的各顶点的一种分配方案。若着色使得相邻顶点的颜色都不同，则称该着色正常，或称G存在一个正常的k顶点着色（或称一个k着色）。此时称G为k-可着色的。

定义 4.2.4 给定图$G=(V,E)$，使该图k-可着色的最小k值称为G的色数，记为$\gamma(G)$。若$\gamma(G)=k$，称G为k色图。

显然我们可得到一些图的色数：

例子 4.2.5 (1) 若图G是零图（即只有一个顶点没有边的图），则$\gamma(G)=1$；
(2) 若图G是n个顶点的完全图，因为任意一个顶点都与$n-1$个顶点相邻，因此$\gamma(G)=n$；
(3) 若图G是二部图，则由于任意边都是连接两个不同集合中的顶点，因此$\gamma(G)=2$；
(4) 若图G是$2n$个顶点的回路，则$\gamma(G)=2$；
(5) 若图G是$2n+1$个顶点的回路，则$\gamma(G)=3$；

引理 4.2.6 若图T是$n(\geq 2)$个顶点的树，则$\gamma(T)=2$。

证明 任取树T的一个顶点v_0，定义：

$$V_1=\{u\mid v_0\text{ 到 }u\text{ 的惟一路径长度为奇数}\}\qquad V_2=\{v\mid v_0\text{ 到 }v\text{ 的惟一路径长度为偶数}\}$$

那么显然V_1之中的任意两个顶点都不相邻，因为若$v_i,v_j\in V$，v_0到v_i的惟一路径是Γ_i，v_0到v_j的惟一路径是Γ_j，显然$\Gamma_i\neq\Gamma_j$，若还有边$e=(v_i,v_j)\in E$，则Γ_i,Γ_j以及边e构成回路，与T是树矛盾，类似地V_2之中的任意两个顶点也都不相邻，从而T是二部图，即$\gamma(G)=2$。 □

定理 4.2.7 给定$G=(V,E)$是非空图（即$|V|>1,|E|>0$），则$\gamma(G)=2$当且仅当它没有奇数长的回路。

证明 如果G有奇数长的回路，则显然$\gamma(G)\geq 3$，因此$\gamma(G)=2$意味着图G没有奇数长的回路；反之，若它没有奇数长回路，若G有多个连通分支，则不同连通分支之间的着色不会互相影响，因此不妨假设它只有一个连通分支。

任取树G的一个顶点v_0，定义：

$$V_1=\{u\mid v_0\text{ 到 }u\text{ 的最短路径长度为奇数}\}\qquad V_2=\{v\mid v_0\text{ 到 }v\text{ 的最短路径长度为偶数}\}$$

那么显然V_1之中的任意两个顶点都不相邻，因为若$v_i,v_j\in V$，v_0到v_i的最短路径是Γ_i，v_0到v_j的最短路径是Γ_j，显然$\Gamma_i\neq\Gamma_j$，若还有边$e=(v_i,v_j)\in E$，则Γ_i,Γ_j以及边e构成奇数长的回路，矛盾！类似地V_2之中的任意两个顶点也都不相邻，从而G是二部图，即$\gamma(G)=2$。 □

下面的定理给出了一个图的色数的上界：

定理 4.2.8 对任意图$G=(V,E)$，$\gamma(G)\leq\Delta(G)+1$，$\Delta(G)$是G中度数最大的顶点的度数。

证明 对G的顶点数作归纳。$n=1$时命题显然成立。假设$n=k-1$时命题成立，即对任意具有$k-1$个顶点的图G，$\gamma(G)\leq\Delta(G)+1$。对于具有k个顶点的图G，任取一个顶点v，图$G-v$具有$k-1$个顶点，根据归纳假设有$\gamma(G-v)\leq\Delta(G-v)+1$。由于$\Delta(G-v)\leq\Delta(G)$，因此$\gamma(G-v)\leq\Delta(G)+1$，即$\Delta(G)+1$种颜色可对$G-v$着色。放回$v$，由于$v$的度数小于$\Delta(G)$，与$v$相邻的顶点之多占用$\Delta(G)$种颜色，因此用$\Delta(G)+1$种颜色也可对$G$着色。 □

对于一个具体的图G，如何确定它的色数呢？下面介绍色数的一种求解方法。先引入一个定义：

定义 4.2.9 给定图$G=(V,E)$，设i,j是图G的两个不相邻的顶点。定义$\overline{G}_{ij}=G+e_{ij}$，这里$e_{ij}=(i,j)$，即$\overline{G}_{ij}$是$G$在顶点$i$和$j$之间增加一条新边而得到的图。

定义$\overset{\circ}{G}_{ij}$也是一个简单图，其顶点集$\overset{\circ}{V}_{ij}=V-\{i,j\}+\{ij\}$，边集:

$$\overset{\circ}{E}_{ij}=E-\{(k,i)\mid(k,i)\in E\}-\{(k,j)\mid(k,j)\in E\}+\{(k,ij)\mid(k,i)\in E\vee(k,j)\in E\}$$

直观地说，$\overset{\circ}{G}_{ij}$是将G中顶点i,j收缩成一个顶点ij，原先与i或j相邻的顶点都与顶点ij相连，并合并多重边而得到的简单图。

定理 4.2.10 设i,j是简单图G的两个不相邻的顶点，则：

$$\gamma(G)=\min\{\gamma(\overline{G}_{ij}),\ \gamma(\overset{\circ}{G}_{ij})\}$$

证明 对G中顶点的任何着色，顶点i和j要么着以同色，要么着以异色，二者必居其一。设i,j着相同颜色时G的最少着色数是$\gamma(G(i,j\text{同色}))$，i,j着不同颜色时G的最少着色数是$\gamma(G(i,j\text{异色}))$，则显然：

$$\gamma(G)=\min\{\gamma(G(i,j\text{同色})),\ \gamma(G(i,j\text{异色}))\}$$

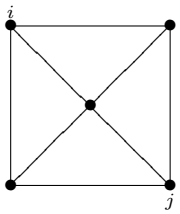
而显然又有：

$$\gamma(G(i,j\text{同色}))=\gamma(\overset{\circ}{G}_{ij})\qquad\gamma(G(i,j\text{异色}))=\gamma(\overline{G}_{ij})$$

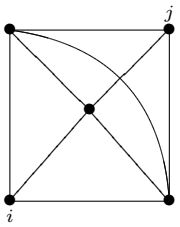
□

根据上述定理，我们通过不断在G的不相邻顶点之间增加边，以及收缩不相邻顶点而将图G变换成完全图，而得到任意图G的色数。

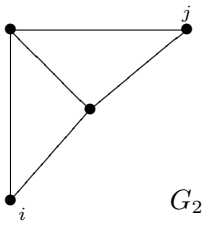
例子 4.2.11 计算下面图G的色数$\gamma(G)$：



如上图，选择i,j两个不相邻的顶点进行变换，增加这两个顶点之间的边得到图$G_1 = \overline{G}_{ij}$，如下图左边所示；收缩这两个顶点得到图$G_2 = \overset{\circ}{G}_{ij}$，如下图右边所示。

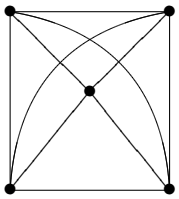


G_1

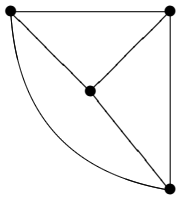


G_2

如上图，在G_1中选择i,j两个不相邻的顶点进行变换，增加这两个顶点之间的边得到图$\overline{G}_1$，它是完全图K_5；收缩这两个顶点得到图$\overset{\circ}{G}_1$，得到的是完全图K_4，这两个图如下：

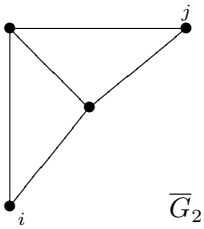


$\overline{G}_1$

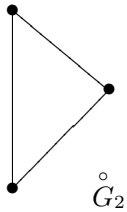


$\overset{\circ}{G}_1$

在G_2中选择i,j两个不相邻顶点进行变换，增加这两个顶点之间的边得到图$\overline{G}_2$，它也是完全图K_4；收缩这两个顶点得到图$\overset{\circ}{G}_2$，得到是完全图K_3，这两个图如下：



$\overline{G}_2$



$\overset{\circ}{G}_2$

从而原图的色数为：

$$\gamma(G) = \min\{\gamma(G_1),\ \gamma(G_2)\} = \min\{\gamma(\overline{G}_1),\ \gamma(\overset{\circ}{G}_1),\ \gamma(\overline{G}_2),\ \gamma(\overset{\circ}{G}_2)\} = 3$$

给定一个图，最多使用k种颜色对其进行着色，使得其相邻顶点着不同颜色，那么会有多少种不同的着色方案呢？显然方案数与k有关，可以证明方案数是k的多项式，因此称为色多项式：

定义 4.2.12 图G使用k种颜色进行正常着色的方案数称为图G的**色多项式**，记为$f(G,k)$。

例子 4.2.13 不难得到一些特殊图的色多项式：

(1) 对于零图：$G=(V,E)$，$n=|V|$，$|E|=0$，其色多项式显然是：$f(G,k)=k^n$；

(2) 对于树T：$T=(V,E)$，$n=|V|$，根节点可在k种颜色中任取，非根节点选取与其父亲节点不同的颜色，也就是说，剩下的$n-1$个顶点都有$k-1$种选择，因此$f(G,k)=k(k-1)^{n-1}$；

(3) 对于完全图K_n：$f(K_n,k)=k(k-1)(k-2)\cdots(k-n+1)$。

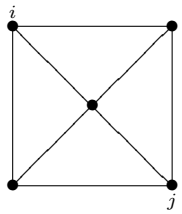
类似求图G的色数的方法，我们有如下定理：

定理 4.2.14 给定图G，i,j是其两个不相邻的顶点，则：

$$f(G,k)=f(\overline{G}_{ij},k)+f(\overset{\circ}{G}_{ij},k)$$

利用这个公式，以及完全图的色多项式，可以求任意图的色多项式。

例子 4.2.15 根据上一例子的变换，下图G



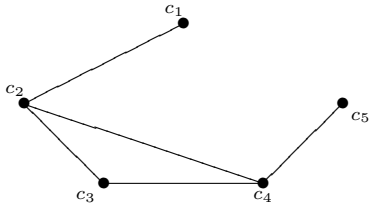
的色多项式是：

$$\begin{aligned} f(G,k) &= f(K_5,k)+2f(K_4,k)+f(K_3,k) \\ &= k(k-1)(k-2)(k-3)(k-4)+2k(k-1)(k-2)(k-3)+k(k-1)(k-2) \\ &= k(k-1)(k-2)(k^2-7k+12+2k-6+1) \\ &= k(k-1)(k-2)(k^2-5k+7) \end{aligned}$$

特别地，$f(G,3)=6$，即以3种颜色着色有6种不同的方案。

例子 4.2.16 现在回到这一节一开始提出的考试安排问题：假定有5门课程c_1,c_2,c_3,c_4,c_5，已知有的学生既选c_1又选c_2，有的学生既选c_2又选c_3，有的既选c_3又选c_4，有的既选c_4又选c_2，也有的既选c_4又选c_5，问至少要安排多少天考试，在安排最少天数的情况下至多有多少种安排方案？

解：我们以c_1,c_2,c_3,c_4,c_5为顶点，若有学生既选课程c_i又选c_j则在c_i和c_j之间连一条边，这样根据题目的陈述，我们得到如下无向简单图：



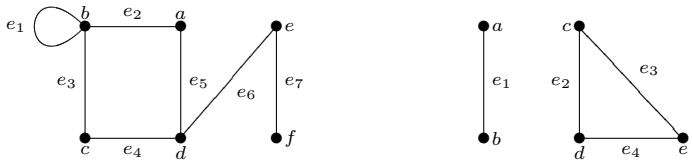
很容易知道上图的色数$\gamma(G)=3$，然后根据定理4.2.14可得到其色多项式是：

$$f(G,k)=f(K_5,k)+5f(K_4,k)+4f(K_3,k)=k^5-5k^4+9k^3-7k^2+2k$$

从而$f(G,3)=24$，也就是说至少要安排3天考试，至多有24种安排方案。

作 业

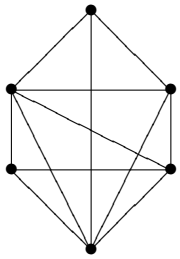
作业 4.1 下面是两个平面图G_1,G_2，请分别给出它们各个面的边界及度：



作业 4.2 设G是有k个连通分支的平面图，G的每个面至少由l条边围成，则：

$$m\leq\frac{l}{l-2}(n-p-1)$$

作业 4.3 下图是否是平面图，如果是请给出它的一个平面嵌入。是否是极大平面图？为什么？



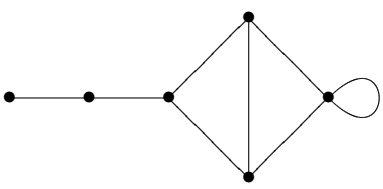
作业 4.4 设简单平面图G的面数$r<12$，图G的最小度$\delta(G)\geq 3$，证明G至少由一个面的度小于5。

作业 4.5 设G是边数m小于30的简单平面图，试证明G中存在度数小于等于4的顶点。

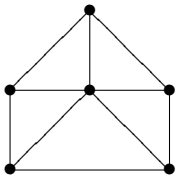
作业 4.6 设简单平面图G的顶点数$n=7$，边数$m=15$，证明G的每个面的度都是3。

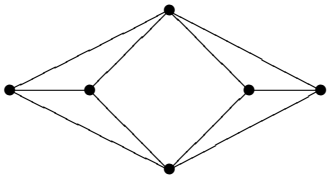
作业 4.7 设G是连通的简单平面图，顶点数为n，边数为m，面数为r，试证明：(1) 若$n\geq 3$，则$r\leq 2n-4$；(2) 若G的最小度$\delta(G)=4$，则G中至少有6个顶点的度数小于等于5。

作业 4.8 求下图的对偶图：

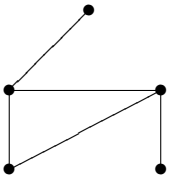


作业 4.9 求下两个图的对偶图，并验证$n^* = r,\ m^* = m,\ r^* = n$：

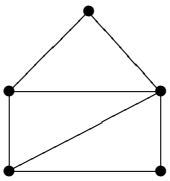




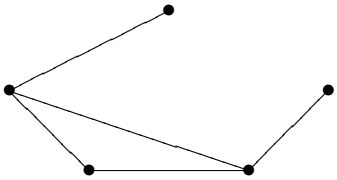
作业 4.10 求下图G的色数$\gamma(G)$及色多项式$f(G,k)$，并计算$f(G,\gamma(G)), f(G,4)$：



作业 4.11 求下图G的色数$\gamma(G)$及色多项式$f(G,k)$：



作业 4.12 求下面图的色多项式$f(G,k)$：



作业 4.13 设G是n阶k-正则图，证明：$\gamma(G) \geq \dfrac{n}{n-k}$。

作业 4.14 设G是不含K_3的连通简单平面图，证明：(1) G中含度数小于等于3的顶点；(2) G是4-可着色的。

作业 4.15 设G是连通简单平面图，G的最短初级回路长度$l \geq 4$，证明：(1) G中存在度数小于等于$l-1$的顶点；(2) G是l- 可着色的。

第五章　支配集、覆盖集、独立集和匹配

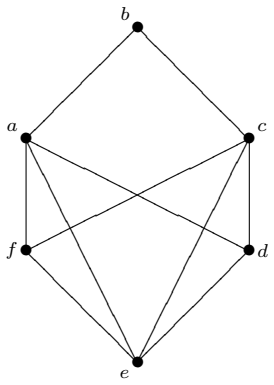
这一章讨论支配集、覆盖集、独立集以及匹配，这些问题的共同点是在图中找出满足一定性质顶点集或边集。目前还没有有效的算法给出任意图的支配集、覆盖集和独立集等，相关问题的算法及其复杂度研究为理论计算机，特别是可计算性理论提供了诸多素材。匹配问题则有着丰富的应用背景，也有比较有效的算法，因此这一章将重点讨论与匹配问题有关的算法。

这一章讨论的图都是无向简单图，如果没有特别说明，下面定义和定理中所说的图都是指无向简单图。这一章的大部分内容来自耿素云、屈婉玲编著的教材:《离散数学》(修订版)，高等教育出版社，2004年。

5.1　支配集、点独立集和点覆盖集

定义 5.1.1 给定图$G=(V,E)$, $V^*\subseteq V$, 若对于任意的$v_i\in V-V^*$, 都存在$v_j\in V^*$使得$(v_i,v_j)\in E$, 则称V^*是G的支配集。若V^*是G的支配集，且V^*的任意真子集都不是G的支配集，则称V^*是G的极小支配集。顶点数最少的支配集称为最小支配集，最小支配集的顶点个数称为图G的支配数，记为$\gamma_0(G)$。

例子 5.1.2 考虑下图：



上图的支配集有：

$$\{a,c\},\ \{b,e\},\ \{c,f\},\ \{b,d,f\},\ \{a,b,c\},\ \{a,b,c,d,e,f\},\cdots$$

极小支配集有：

$$\{a,c\},\ \{b,e\},\ \{c,f\},\ \{b,d,f\}$$

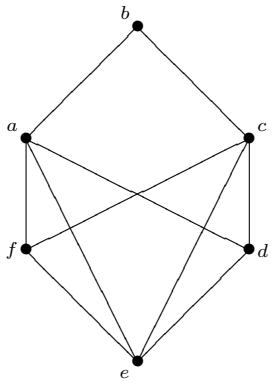
而最小支配集有：

$$\{a,c\},\ \{b,e\},\ \{c,f\},\ \{b,d,f\}$$

注意，最小支配集不是惟一的，图G的支配数$\gamma_0(G)=2$。

定义 5.1.3 给定$G=(V,E)$，$V^*\subseteq V$，若V^*中任意两个顶点都不相邻，则称V^*为G的（点）独立集。若V^*再加入任何顶点都不再是独立集，则称V^*为极大点独立集，顶点数最多的点独立集称为最大点独立集，其顶点数称为图G的独立数，记为$\beta_0(G)$。

例子 5.1.4 考虑例子5.1.2中的图：



其独立集有：

$$\{b,d\},\ \{b,f\},\ \{a,c\},\ \{b,d,f\},\cdots$$

其极大独立集有：

$$\{b,d\},\ \{a,c\},\ \{b,d,f\}$$

而$\{b,d,f\}$是最大独立集，因此图G的独立数是$\beta_0(G)=3$。

独立集与支配集有如下联系：

定理 5.1.5 给定图$G=(V,E)$无孤立顶点，则G的极大点独立集都是G的极小支配集。

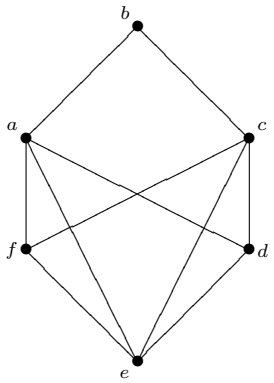
证明 设V^*是G的一个极大点独立集，我们证明它也是G的极小支配集。首先证明它是G的支配集，用反证法。若V^*不是G的支配集，即存在顶点$u\in V-V^*$，使得不存在顶点$v\in V^*$，使得有边$(u,v)\in E$，也即u不与V^*的任何顶点相邻，从而$V^*\cup\{u\}$也还是独立集，与V^*是极大独立集矛盾，因此V^*必是G的支配集。

其次证明V^*是极小支配集，这显然成立，因为对V^*的任意真子集$V_1^*\subset V^*$，在$V^*-V_1^*$中的顶点都不与V_1^*中的顶点相邻，因此V_1^*不可能是支配集。 □

上述定理的逆不成立，对于例子5.1.2中的图G，$\{c,f\}$是极小支配集，但它不是点独立集，更不是极大点独立集。

定义 5.1.6 给定图$G=(V,E)$，$V^*\subseteq V$，若对于任意的边$e\in E$，都存在顶点$v\in V^*$，使得v与e相关联，则称V^*为G的点覆盖集，或简称点覆盖。若V^*是G的点覆盖，而V^*的任何真子集都不是G的点覆盖，则称V^*为极小点覆盖，顶点个数最少的点覆盖称为最小点覆盖，其顶点个数称为图G的点覆盖数，记为$\alpha_0(G)$。

例子 5.1.7 考虑例子5.1.2中的图：



其点覆盖有：

$$\{a,b,c,d,e\},\ \{a,b,c,e\},\ \{a,c,e\},\cdots$$

极小点覆盖有：

$$\{a,c,e\},\ \{b,d,e,f\},\ \{a,c,d,f\}$$

而$\{a,c,e\}$是图G的最小点覆盖，因此图G的点覆盖数$\alpha_0(G)=3$。

点覆盖集和点独立集有如下联系：

定理 5.1.8 给定图$G=(V,E)$无孤立点，$V^*\subset V$，则V^*是G的点覆盖当且仅当$\overline{V^*}=V-V^*$是G的独立集。

证明 先证V^*是G的点覆盖蕴涵$\overline{V^*}$是G的独立集，用反证法。若$\overline{V^*}$不是G的独立集，即存在两个顶点$u,v\in\overline{V^*}$，u,v相邻，这表明边$e=(u,v)$的两个端点都不在V^*，这与V^*是点覆盖集矛盾!

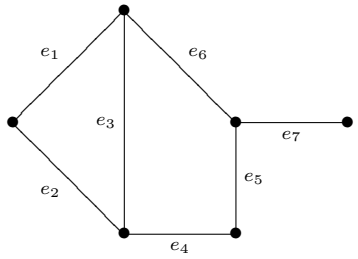
再证$\overline{V^*}$是G的独立集蕴涵V^*是点覆盖集。由$\overline{V^*}$是G的独立集，表明G的任意边的两个端点都至少有一个属于V^*（否则这两个端点在$\overline{V^*}$中相邻），也即V^*是点覆盖集。 □

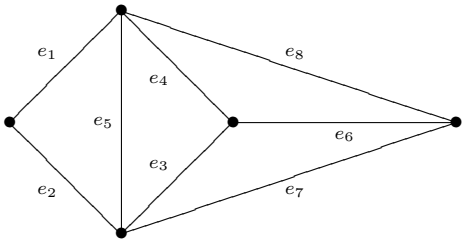
推论 5.1.9 给定$G=(V,E)$无孤立点，$|V|=n$，则V^*是G的极小点覆盖当且仅当$\overline{V^*}=V-V^*$是G的极大独立集，从而有$\alpha_0+\beta_0=n$。

5.2 边覆盖与匹配

定义 5.2.1 给定图$G=(V,E)$，$E^*\subseteq E$，若对G的任意顶点$v\in V$，都存在边$e\in E^*$与v关联，则称E^*为图G的边覆盖。设E^*为边覆盖，若E^*的任何真子集都不是边覆盖，则称E^*为极小边覆盖。边数最小的边覆盖称为图G的边覆盖数，记为$\alpha_1(G)$。

例子 5.2.2 考虑下面两个图：





对于左边的图，$\{e_1, e_4, e_7\}$, $\{e_2, e_5, e_6, e_7\}$等都是极小边覆盖，其中$\{e_1, e_4, e_7\}$是最小边覆盖，该图的边覆盖数是3。对于右边的图，$\{e_1, e_3, e_6\}$, $\{e_2, e_4, e_8\}$等都是极小边覆盖，而且也都是最小边覆盖，其边覆盖数也是3。

定义 5.2.3 给定图$G = (V, E)$，$E^* \subseteq E$，若E^*中的任意两条边都不相邻，即任意两条边都不存在公共端点，则称E^*为G的**边独立集**，也称E^*为G的**匹配**。若在E^*中再添加任意边后都不再是匹配，则称E^*为**极大匹配**。边数最多的匹配称为**最大匹配**，其边数称为**边独立数**或**匹配数**，记为$\beta_1(G)$。

例子 5.2.4 考虑例子5.2.2中的两个图。对于左边的图，$\{e_2, e_6\}, \{e_3, e_5\}, \{e_1, e_4, e_7\}$等都是极大匹配，其中$\{e_1, e_4, e_7\}$是最大匹配，其匹配数是3。对于右边的图，$\{e_1, e_3\}, \{e_2, e_4\}, \{e_4, e_7\}$等都是极大匹配，同时也都是最大匹配，其匹配数是2。

匹配问题有很强的应用背景，例如在国际会议中，需要两人一组编成讨论小组进行讨论，有的与会人员会一门外语，有的与会人员会很多门外语，假设要求小组中的两人要会一门共同的外语。我们以人为顶点，以能够编成一组的两个人之间连一条边，则编组方案就是求图的匹配，能将最多人进行编组的方案就是最大匹配。

匹配问题中应用得最多的是二部图的匹配，因为常见的任务分配问题都与二部图的匹配问题有关。例如，m项工作准备分配给n个人做，以工作为一个顶点集，以人员为另一个顶点集，两个顶点之间有边表明某项工作可以由某人完成。如果要求每个人至多从事一项工作，一项工作也只能由一个人承担，则工作安排方案就是求二部图的匹配。

为了更好地研究匹配，下面引入更多有关匹配的概念。

定义 5.2.5 设M是图$G = (V, E)$的一个匹配。

(1) 若$e = (u, v) \in M$，则称u和v被M所匹配;

(2) 对于顶点v，若存在边$e \in M$使得e与v关联，则称v为**M饱和点**，否则称v为**M非饱和点**;

(3) 若G的每个顶点都是M饱和点，则称M为**完美匹配**;

对于例子5.2.2的两个图，左边图的完美匹配是$\{e_1, e_4, e_7\}$，右边图不存在完美匹配，对于它的任意一个最大匹配，例如$M = \{e_2, e_4\}$，通过增加关联非饱和顶点的边可产生最小边覆盖，例如$M \cup \{e_6\}, M \cup \{e_7\}, M \cup \{e_8\}$都是最小边覆盖；而对于它的任意一个最小边覆盖，例如$W = \{e_1, e_3, e_6\}$，从中删除相邻边的其中一条边都可得到最大匹配，例如$W - \{e_3\}, W - \{e_6\}$都是最大匹配。一般地说，我们有：

定理 5.2.6 给定图$G = (V, E)$无孤立点，$|V| = n$：

(1) 设M为G的一个最大匹配，对于G的每个M非饱和点都取一条与其关联的边组成边集N，则$W = M \cup N$是G的最小边覆盖；

(2) 设W_1是G的一个最小边覆盖，若W_1中存在相邻的边就删除其中的一条，继续这个过程，直到无相邻的边，设删除的边构成集合N_1，即$M_1 = W_1 - N_1$是G的最大匹配；

(3) G的边覆盖数α_1与匹配数β_1满足：$\alpha_1 + \beta_1 = n$。

证明 因为M是最大匹配，所以M中的边数等于β_1，那么G有$n - 2\beta_1$个M非饱和点。显然N中的边只关联一个M非饱和点（否则的话M就不是最大匹配），因此N中的边数为$n - 2\beta_1$。显然$W = M \cup N$是边覆盖，且：

$$|W| = |M| + |N| = \beta_1 + n - 2\beta_1 = n - \beta_1$$

另一方面，由W_1是最小边覆盖，W_1中的任意边至多只有一个端点与W_1中的其他边关联（否则从W_1删除这条边仍然是边覆盖，与W_1是最小边覆盖矛盾），因而在由W_1构造M_1时，每删除相邻两条边中的一条时，产生并只产生一个M_1非饱和点，也就是说，删除的边数等于M_1非饱和点数，即：

$$|N_1| = |W_1| - |M_1| = n - 2|M_1|$$

整理后得到$|W_1| = \alpha_1 = n - |M_1|$。显然$M_1$是匹配，即$|M_1| \leq \beta_1$，而$W$是覆盖，即$|W| \geq \alpha_1$，从而：

$$\alpha_1 = n - |M_1| \geq n - \beta_1 = |W| \geq \alpha_1$$

这表明$|M_1| = \beta_1$，即M_1是最大匹配，而且$|W| = \alpha_1$，即W是最小边覆盖，且$\alpha_1 + \beta_1 = n$。 □

推论 5.2.7 给定图$G = (V, E)$无孤立点，$|V| = n$。M是G的匹配，W是G的边覆盖，则$|M| \leq |W|$，等号成立时M是G的完美匹配而W是G的最小边覆盖。

证明 上面定理表明最小边覆盖删除相邻边之一后可得到最大匹配，因此$\beta_1 \leq \alpha_1$，从而$|M| \leq \beta_1 \leq \alpha_1 \leq |W|$。当等号成立时，说明$M$是最大匹配而$W$是最小边覆盖（否则与$\beta_1 \leq \alpha_1$矛盾），从而$\alpha_1 = |W| = |M| = \beta_1$，从而$\alpha_1 + \beta_1 = 2\beta_1 = n$，从而$G$没有$M$非饱和点（因为$M$非饱和点个数等于$n - 2\beta_1$），这表明$M$是完美匹配。 □

下面我们考虑怎样判断一个匹配是否是最大匹配，首先我们定义匹配的可增广的交错路径：

定义 5.2.8 称G中在M和$E - M$中交替取边的回路称为M的交错回路。称G中在M和$E - M$中交替取边的初级道路 称为M的交错路径，起点和终点都是M非饱和点的交错路径称为可增广交错路径。

引理 5.2.9 设M是图G的匹配，Γ是M的可增广交错路径，则$M' = \Gamma \oplus M$也是G的匹配，且$|M'| = |M| + 1$。这里$\oplus$是异或操作，也即M'是取在M或在Γ中，但不同时在M和Γ中的边构成的集合。

证明 对M'中的任意两条边u, v，有四种情况：

(1) $u \in M - \Gamma$且$v \in M - \Gamma$，那么由于M是匹配，从而u, v不相邻；

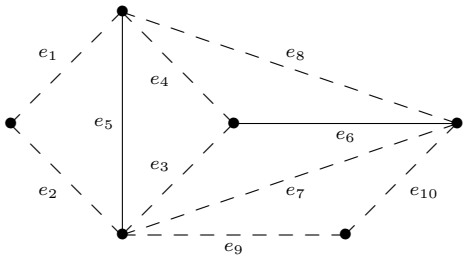
(2) $u \in \Gamma - M$且$v \in \Gamma - M$，那么由于Γ是交错路径，在M和$E - M$中交替取边，而且不存在重复的顶点（是初级道路），从而u, v不相邻；

(3) $u \in \Gamma - M$而$v \in M - \Gamma$，那么注意到v的两个端点都是M饱和点，而u的两个端点不都是M饱和点，因此若u的两个端点都不是M饱和点时，显然v与u不相邻，而当u的一个端点也是M饱和点时，它也不可能是v的某个端点，因为否则的话，由于v不在Γ中，而u的这个M饱和点又不是Γ的终点，因此必还有一条在M中的边与u相邻，而这时这条边也与v相邻，与M是匹配矛盾，因此v不与u相邻；

(4) $v\in\Gamma-M$而$u\in M-\Gamma$，与上一情况类似，v与u不相邻。

因此M'中的任意两条边都不相邻，也即M'是图G的匹配。又Γ的两个端点都是M非饱和点，从而Γ中在M中的边比不在M中的边少一条，从而$|M'|=|M|+1$。 □

例子 5.2.10 考虑下面的图：



其中$M=\{e_5,e_6\}$是匹配，$\Gamma_1=e_1e_5e_3e_6e_8$是M的交错路径，但不是可增广交错路径，因为它的两个端点不都是M非饱和点，而$\Gamma_2=e_1e_5e_3e_6e_{10}$是$M$的可增广交错路径，显然$M'=\Gamma_2\oplus M=\{e_1,e_3,e_{10}\}$是比$M$更大的匹配。

上面的引理实际上给出了在图中寻找最大匹配的方法，即从任意一个初始匹配开始（例如只有一条边的匹配），通过寻找该匹配的可增广交错路径可得到一个更大的匹配，直到得到一个不存在可增广交错路径的匹配。首先，我们有如下简单引理：

引理 5.2.11 设M,M'是图G的两个匹配，则由$M\oplus M'$导出的子图的每个连通分支要么是在M和M'中交替取边的交错回路，要么是在M和M'中交替取边的交错路径。注意，$M\oplus M'$是取或者在M或者在M'中，而不同时在M和M'中的边，其导出的子图则是这些边加上与这些边关联的顶点得到的G的子图。

证明 因为M导出的子图的顶点度数都是1，而M'导出的子图的顶点度数也都是1，因此$M\oplus M'$导出的子图的顶点度数不是1就是2，如果它的某个连通分支存在两个度数为1的顶点（注意，不可能存在多于两个度数为1的顶点），则该连通分支是一条初级路径；如果不存在度数为1的顶点则就是初级回路。不管是初级道路还是初级回路，都必然是在M和M'中交替取边（否则与M和M'都是匹配矛盾）。 □

下面的定理说明了不存在可增广交错路径的匹配确实是最大匹配：

定理 5.2.12 M是G的最大匹配当且仅当G不含M可增广的交错路径。

证明 由上面的引理我们知道，当M是最大匹配时，它不含M可增广交错路径。下面只要证明若匹配M不含可增广交错路径时是最大匹配。设M_1是G的最大匹配，只要证明$|M|=|M_1|$即可。

为此考虑M_1和M的对称差得到的边集导出的子图（即这些边，以及与这些边关联的顶点构成的图），设该子图是H。如果$H=\varnothing$是空图，即$M=M_1$，于是M就是最大匹配。若$H\neq\varnothing$，则根据上一引理，H的各连通分支导出的子图的每个连通分支要么是在M和M_1中交替取边的交错回路，要么是在M和M_1中交替取边的交错路径。

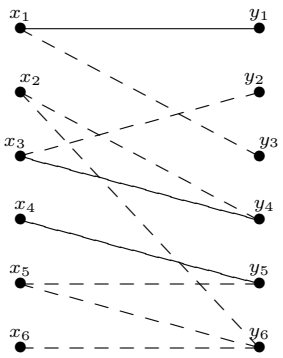
当某个连通分支是在M和M_1中交替取边的交错回路时，显然该连通分支中在M和M_1中的边数相等。而当它是在M和M_1中交替取边的交错路径时，则该交错路径的两个端点不可能同时是M饱

和点（否则该交错路径就是M_1的可增广交错路径，与M_1是最大匹配无可增广交错路径矛盾），也不可能同时是M_1饱和点（否则该交错路径是M的可增广交错路径），这样该交错路径中M中的边和M_1中的边的数目也相同。

总之，无论何种情况，总有M的边数与M_1的边数相等，也即$|M| = |M_1|$，也即M是最大匹配。□

上面的定理表明可通过寻找匹配的可增广交错路径扩大匹配，但在一般图中，寻找可增广交错路径并不是一件简单任务，所以我们下面只讨论在二部图中寻找最大匹配的算法。一个有效的算法称为匈牙利算法，其基本思想是在给定一个初始匹配之后，通过不断地寻找其可增广交错路径而扩大该匹配，直到不存在可增广交错路径。

例子 5.2.13 我们使用戴一奇教材[3]p90的例子5.1.3来说明匈牙利算法的基本思想。考虑下面的二部图$G = (X, Y, E)$，其中$X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$，$Y = \{y_1, y_2, y_3, y_4, y_5, y_6\}$。在图中我们用实线给出了初始的匹配$M = \{(x_1, y_1), (x_3, y_4), (x_4, y_5)\}$。匈牙利算法的基本思想是寻找匹配的可增广交错路径，从而扩大该匹配。



为了寻找匹配的可增广交错路径，对X中的每个顶点进行标记，饱和点标记1，没有搜索的点标记0，无法扩大匹配的顶点标记为2。对于上述初始匹配，初始标记是将M饱和点标记1，其他顶点都标记为0，即：

X：	标记1的顶点：$\{x_1, x_3, x_4\}$	标记0的顶点：$\{x_2, x_5, x_6\}$	标记2的顶点：$\varnothing$
Y：	标记1的顶点：$\{y_1, y_4, y_5\}$	标记0的顶点：$\{y_2, y_3, y_6\}$	标记2的顶点：$\varnothing$

在X中选一个标记为0的顶点u，寻找以该顶点为起点的可增广交错路径。寻找的方法是使用集合U记录X中可能在该交错路径中的顶点，用V记录Y中可能在该交错路径中的顶点。这两个集合中的顶点的添加方式满足：

1. V的初始值是空集，V中的顶点都是U中顶点的相邻顶点，且除了最后一个添加的顶点可能是非饱和点之外，其他顶点都是饱和顶点，如果最后发现一个非饱和顶点，则找到一个可增广交错路径，否则初始选的顶点v无法扩大匹配；

2. U中的初始顶点是u，其后加入的顶点都是根据V中的顶点v，选择一条在匹配M中的边(v, u')而加入u'。

具体来说，对于上面的初始匹配，我们选择标记为0的顶点x_2，记$U = \{x_2\}$，$V = \varnothing$。考虑x_2的相邻顶点y_4和y_6，由于y_6标记为0，是M非饱和顶点，优先考虑y_6，就得到一条可增广交错路径(x_2, y_6)，

将这条边加入M，扩大M为$M=\{(x_1,y_1),(x_2,y_6),(x_3,y_4),(x_4,y_5)\}$。将$x_2$和$y_6$的标记改为1，得到：

X： 标记1的顶点：$\{x_1,x_2,x_3,x_4\}$　　标记0的顶点：$\{x_5,x_6\}$　　标记2的顶点：$\varnothing$

Y： 标记1的顶点：$\{y_1,y_4,y_5,y_6\}$　　标记0的顶点：$\{y_2,y_3\}$　　标记2的顶点：$\varnothing$

对于扩大后的匹配，选择标记为0的顶点x_5，令$U=\{x_5\},V=\varnothing$。考虑x_5的相邻顶点y_5,y_6，由于它们都已经有标记1，因此随意选择一个顶点，例如y_5加入到V，然后选择y_5关联的在M中的边(x_4,y_5)，将x_4加入U，得到：

$$U=\{x_5,x_4\}\qquad V=\{y_5\}$$

下一步考虑与x_5,x_4相邻的所有顶点y_5,y_6，由于y_5已经在V中，这一次考虑y_6，它的标记也为1，将其加入V，并选择y_6关联的在M中的边(x_2,y_6)，将x_2加入U，得到：

$$U=\{x_5,x_4,x_2\}\qquad V=\{y_5,y_6\}$$

再考虑与x_5,x_4,x_2相邻且不在V中的顶点y_4，y_4的标记也为1，将其加入V，并选择y_4关联的在M中的边(x_3,y_4)，将x_3加入U，得到：

$$U=\{x_5,x_4,x_2,x_3\}\qquad V=\{y_5,y_6,y_4\}$$

考虑与U中顶点相邻且不在V中的顶点y_2，y_2标记为0，是M非饱和顶点，因此存在以x_5为起点的可增广路径，该路径可通过回溯加入y_2的过程得到：

$$\Gamma=y_2x_3y_4x_2y_6x_5$$

将M扩大为$M=M\oplus\Gamma=\{(x_1,y_1),(x_2,y_4),(x_3,y_2),(x_4,y_5),(x_5,y_6)\}$。将$x_5$和$y_2$的标记改为1，得到：

X： 标记1的顶点：$\{x_1,x_2,x_3,x_4,x_5\}$　　标记0的顶点：$\{x_5\}$　　标记2的顶点：$\varnothing$

Y： 标记1的顶点：$\{y_1,y_2,y_4,y_5,y_6\}$　　标记0的顶点：$\{y_3\}$　　标记2的顶点：$\varnothing$

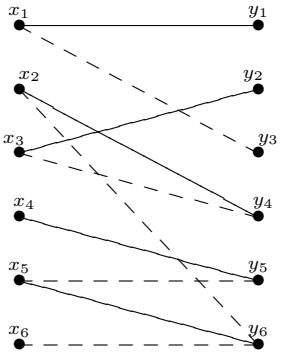
对于扩大后的匹配，选择标记为0的顶点x_6，令$U=\{x_6\},V=\varnothing$，考虑x_6的相邻顶点y_6，y_6已经标记为1，将其加入V，并选择y_6关联的在M中的边(x_5,y_6)，将x_5加入U，得到：

$$U=\{x_6,x_5\}\qquad V=\{y_6\}$$

考虑与U中的顶点相邻且不在V中的顶点y_5，y_5也标记为1，将其加入V，并选择y_5关联的在M中的边(x_4,y_5)，将x_4加入U，得到：

$$U=\{x_6,x_5,x_4\}\qquad V=\{y_6,y_5\}$$

这时U中的顶点没有不在V中的相邻顶点，这意味着以x_6为起点的交错路径都只会以饱和点结束，因此不存在以x_6为起点的可增广交错路径，这表明x_6无法扩大匹配，标记为2。到此，所有在X中的顶点已经被标记为1或2，算法结束，而在Y中仍标记为0的顶点也是无法参与匹配的顶点。最后得到的最大匹配就是：$M=\{(x_1,y_1),(x_2,y_4),(x_3,y_2),(x_4,y_5),(x_5,y_6)\}$，下图用实线给出了该匹配：



5.3 二部图中的匹配

下面我们转为讨论二部图中的匹配，因为二部图中的匹配问题有更强的应用背景，而且寻找二部图的最大匹配也比寻找一般图的最大匹配简单。首先回忆二部图的概念：

定义 5.3.1 给定图$G=(V,E)$，如果V能分为两个不相交的非空子集V_1,V_2，即$V_1\neq\varnothing,V_2\neq\varnothing,V_1\cup V_2=V,V_1\cap V_2=\varnothing$，且对$G$的任意边$e=(u,v)\in E$，都有$u\in V_1$且$v\in V_2$，或者$v\in V_1$且$u\in V_2$，也即$G$的任意边的两个端点都在不同的子集，则称$G$是**二部图**。若$|V_1|=s,|V_2|=r$，且$V_1$的任意顶点都与$V_2$的任意顶点有边，则称该二部图为**完全二部图**，记为$K_{s,r}$。

我们通常记二部图为$G=(V_1,V_2,E)$，以明确给出它的两个不相交的非空顶点子集。上一章证明了二部图的色数是2，实际上我们可以得到：

引理 5.3.2 一个图是二部图当且仅当该图没有奇数长的回路。

证明 上一章证明了一个图的色数是2当且仅当它没有奇数长的回路，而显然一个图的色数是2当且仅当它是二部图。 □

在二部图中，我们通常考虑其完备匹配：

定义 5.3.3 给定二部图$G=(V_1,V_2,E)$，设$|V_1|\leq|V_2|$，若M是G的最大匹配，且$|M|=|V_1|$，则称M为V_1到V_2的**完备匹配**（或称完全匹配）。

直观地说，图的匹配就是将顶点进行某种配对，最大匹配就是参与的顶点数尽可能多的匹配，完美匹配就是所有顶点都参与的匹配，而对于二部图而言，我们往往更关注其中的一个顶点集（当然是顶点数集少的那个顶点集）是否都参与了匹配，因此有所谓的完备匹配。显然在上述定义中，当$|V_1|=|V_2|$时，因为二部图的每条边一定分别关联V_1和V_2的一个顶点，因此完备匹配就是完美匹配，而当$|V_1|<|V_2|$时，二部图不可能有完美匹配，这时完备匹配只是最大匹配而已。

对于二部图，我们可以判断其是否存在完备匹配：

定理 5.3.4 Hall定理：给定二部图$G=(V_1,V_2,E)$，设$|V_1|\leq|V_2|$，G存在V_1到V_2的完备匹配当且仅当V_1中任意$k(k=1,2,\cdots,|V_1|)$个顶点至少与V_2中的k个顶点相邻。该条件常称为二部图存在完备匹配的**相异性条件**。

证明 若G中存在V_1的k个顶点之多与V_2中的$k-1$个顶点相邻，显然，这V_1的这k个顶点不可能都由V_2的顶点与它匹配，从而G不存在从V_1到V_2的完备匹配。

下面我们证，当V_1的任意k个顶点至少与V_2的k个顶点相邻时，G的最大匹配就是V_1到V_2的完备匹配。反证法，设M是G的最大匹配，但不是G的完备匹配，也即存在$v_0 \in V_1$是M非饱和点，那么首先在V_2中存在与v相邻的顶点，因为根据相异性条件，v_0不是孤立点，其次与v_0相邻的V_2中的顶点都是M饱和点，因为若有顶点$u \in V_2$与v_0相邻，且是M非饱和点，则$M \cup \{(v_0, u)\}$仍是G的匹配，与M是最大匹配矛盾。

现在考虑从v_0出发的尽可能长的交错路径，由于M是最大匹配，因此这些交错路径都不是可增广交错路径，即每条路径的另一个端点必然是M饱和点，于是这些交错路径的长度为偶数长，从而由二部图的性质，这些路径的端点全部在V_1中。定义：

$$S = \{u \mid u \in V_1 \text{ 且}u\text{在从}v_0\text{出发的交错路径上}\}$$
$$T = \{u \mid u \in V_2 \text{ 且}u\text{在从}v_0\text{出发的交错路径上}\}$$

由于所有交错路径的端点全在S中，S中除了v_0之外，其它顶点均与T中的顶点配对，所以$|S| = |T| + 1$。显然与S中顶点相邻的顶点都在T中，从而V_1中的$|S| = |T| + 1$个顶点只与V_2中$|T|$个顶点相邻，与相异性条件矛盾！所以V_1中不存在M非饱和点，即M是完备匹配。 □

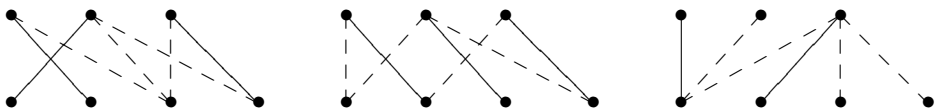
由上述定理，我们还可以得到一个更容易判断的条件，称为二部图完备匹配的t条件：

推论 5.3.5 给定二部图$G = (V_1, V_2, E)$，设$|V_1| \leq |V_2|$。若V_1中每个顶点至少关联$t(t \geq 1)$条边，而V_2中每个顶点至多关联t条边，则G存在V_1到V_2的完备匹配。

证明 由于V_1中每个顶点至少关联t条边，从而V_1的任意$k(1 \leq k \leq |V_1|)$个顶点至少关联kt条边，又由于V_2的每个顶点之多关联t条边，所以kt条边至少关联V_2中k个顶点，根据相异性条件，G存在V_1到V_2的完备匹配。 □

显然t条件比相异性条件要强，也就是说，满足相异性条件不一定满足t条件。

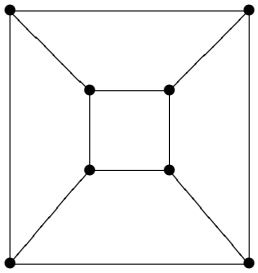
例子 5.3.6 考虑下面三个图：



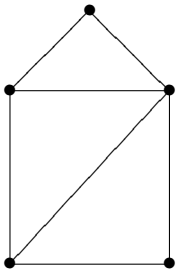
上面最左边的图满足相异性条件，但不满足t条件，中间的图既满足相异性条件，又满足t条件，这时$t = 2$，因此这两个图都有完备匹配，正如图中实线所示。上面最右边的图不满足相异性条件，有V_1（上排顶点）的两个顶点只与V_2（下排顶点）的一个顶点相邻，因此它不存在完备匹配。

作 业

作业 5.1 下面两个图哪个是二部图？哪个不是二部图？为什么？

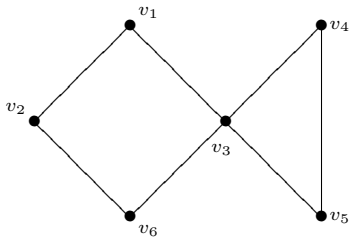


(a)



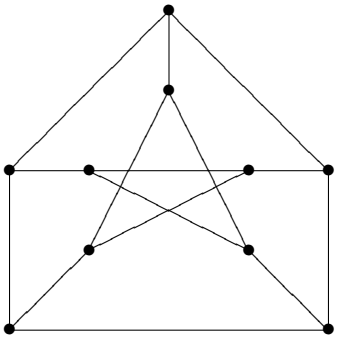
(b)

作业 5.2 考虑下面的图：

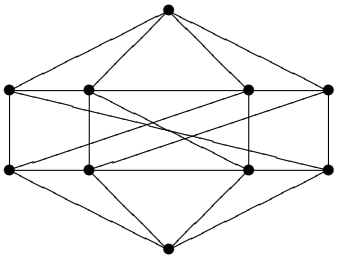


(1) 给出上图的两个不同的极小支配集、一个最小支配集及支配数γ_0；
(2) 给出上图的两个不同的极大点独立集、一个最大点独立集及点独立数β_0；
(3) 给出上图的两个不同的极小点覆盖集、一个最小点覆盖集及点覆盖数α_0。

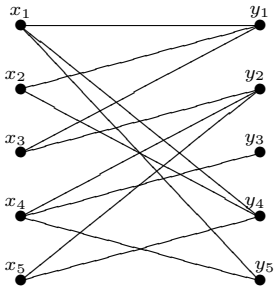
作业 5.3 在下面的彼得森图中找一个最小点覆盖集N，最大点独立集Y，最小边覆盖集W和一个最大边独立集M。然后计算出图中的点覆盖数α_0，点独立数β_0，边覆盖数α_1和边独立数β_1。



作业 5.4 求下图的点覆盖数α_0，点独立数β_0，边覆盖数α_1和边独立数β_1。



作业 5.5 给出初始匹配$M=\{(x_1,y_1),(x_4,y_2)\}$，求下面二部图的最大匹配：



作业 5.6 某用人单位要招聘7个人，设7个工作岗位分别为：$P_1, P_2, P_3, P_4, P_5, P_6, P_7$，现有10个申请者$a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}$，这10个人可胜任的工作岗位集合依次是$\{P_1, P_5, P_6\}$, $\{P_2, P_6, P_7\}, \{P_3, P_4\}, \{P_1, P_5\}, \{P_6, P_7\}, \{P_3\}, \{P_2, P_3\}, \{P_1, P_3\}, \{P_5\}, \{P_1\}$，问如何安排它们的工作使得无工作的人最少？

作业 5.7 现有4名教师，张、王、李、赵，要求他们去教4门课程：数学、物理、英语和化学。已知张能胜任数学和化学；王能胜任物理和英语；李能胜任数学、物理和英语，而赵只能胜任英语。请问应该如何安排，才能使每位教师都能教一门自己能够胜任的课程，并且每门课都有人教？

作业 5.8 现有6位教师：张、王、李、赵、孙、周，要安排他们去教6门课程：数学、化学、物理、语文、英语和计算机。张老师可教数学、计算机和英语；王老师可教英语和语文；李老师可教数学和物理；赵老师只能教化学；孙老师可教物理和计算机；周老师可教数学和物理。问怎样安排课程才能使得每门课程都有人教，每个人都只教一门课？

作业 5.9 现有三个课外小组：物理组、化学组和生物组。今有张、王、李赵陈5名同学。已知：
(1) 张、王为物理组成员，张、李、赵为化学组成员，李、赵、陈卫生物组成员；
(2) 张为物理组成员，王、李、赵为化学组成员，王、李、赵、陈为生物组成员；
(3) 张为物理组和化学组成员，王、李、赵、陈为生物组成员。
问在(1),(2),(3)三种情况能否各选出三名不兼职的组长，为什么？

作业 5.10 某年级共开设了7门课，有7位教师承担。已知每位教师都可担任其中的3门课。他们将自己能够担任的课上报教务处之后，教务员发现每门课都恰有3位教师能承担。问教务员能否安排这7位教师每人担任1门课，且每门课都有人承担？

第六章　欧拉图与哈密顿图

6.1　欧拉图

欧拉图来源于“哥尼斯堡七桥问题”，是图论研究的发源之一。

定义 6.1.1 无向连通图$G=(V,E)$的一条经过所有边的简单回路（道路）称为G的欧拉回路（道路）。

定理 6.1.2 无向连通图$G=(V,E)$存在欧拉回路的充要条件是G的所有顶点的度数都为偶数。

证明　必要性。若G中有欧拉回路C，则C过每一条边一次且仅一次。对任一顶点v来说，如果C经由e进入v，则一定通过另一条边e'离开v，因此v的度数是偶数。

充分性。由于G的所有顶点度数都是偶数，因此对G的任意顶点v_0，一定存在从v_0出发的G的一个简单回路C。如果C已经包含G的所有边，则得到一个欧拉回路，否则考虑$G-C$，它的所有顶点度数仍然是偶数，而且一定存在一个顶点v与C有边相连，而且顶点v的度数大于0，从而这一点开始又存在一个简单回路C'，这样$C\cup C'$就是一个新的回路，比C的边多，一直这样扩大回路，直到包含所有的边，则得到一个欧拉回路。　□

根据这个定理可找出一个所有顶点都是偶数的图的欧拉回路，具体例子可参见戴一奇教材p16的例2.3.1。

推论 6.1.3 若无向连通图G中只有2个度为奇数的顶点，则G存在欧拉道路。

欧拉回路有着十分有趣的应用，例如戴一奇教材p17的例2.3.4给出了其在计算机科学中一个十分有趣的应用。我国图论专家管梅谷先生提出的中国邮路问题也与欧拉回路有着十分密切的联系。

6.2　哈密顿图

哈密顿图来源于19世纪英国数学家哈密顿提出的周游世界的问题，后来成为图论研究中一个重要问题，并且有许多重要的应用。

定义 6.2.1 无向图的一条经过全部顶点的初级回路（道路）称为它的哈密顿回路（道路），简称H回路（道路）。

至今没有找到一个图存在哈密顿回路的充分且必要的条件，但可以证明下面的充分条件：

定理 6.2.2 如果简单图G的任意两个顶点v_i, v_j的度数之和大于等于$n-1$，这里n是图G的顶点个数，则G存在哈密顿道路。

证明 参见戴一奇教材p16定理2.4.1的证明。先证G是连通图。若G非连通，则至少有2个连通分支H_1, H_2，其顶点数分别为n_1, n_2，从中各取一个顶点v_1, v_2，则$d(v_1) \leq n_1 - 1$，而$d(v_2) \leq n_2 - 1$，从而$d(v_1) + d(v_2) \leq n_1 + n_2 - 2 \leq n - 1$，矛盾！所以$G$必然是连通图。

下面证明G存在哈密顿道路。设$\Gamma = v_{i1}v_{i2}\cdots v_{il}$是$G$的一条极长初级道路，即$\Gamma$的两个端点$v_{i1}$和$v_{il}$的相邻顶点都在$\Gamma$中。若$l = n$，则$\Gamma$就是一条哈密顿道路。

若$l < n$，我们可用反证法证明G中存在经过顶点$v_{i1}, v_{i2}, \cdots, v_{il}$的回路。因为若不存在这样的回路，考察$v_{i1}$和$v_{il}$的相邻顶点，注意它们的相邻顶点都在$\Gamma$中，若假设不存在这样的回路，那么对任意的$v_{ip} \in \Gamma$，$v_{i1}$与$v_{ip}$相邻意味着$v_{il}$就不能与$v_{ip-1}$相邻（否则$v_{i1}v_{i2}\cdots v_{ip-1}v_{il}v_{il-1}\cdots v_{ip}$就是一条回路），于是当$d(v_{i1}) = k$时，$d(v_{il}) \leq l - k - 1$（其中减1因为不能与自身相邻，注意图$G$是简单图），从而$d(v_{i1}) + d(v_{il}) < n - 1$，与题设条件矛盾！这就证明了当$l < n$时存在经过顶点$v_{i1}, \cdots, v_{il}$的回路$C$。

由于G是连通图，所以存在C之外的顶点v_t与C的某个顶点v_{iq}相邻，删除(v_{iq-1}, v_{iq})，则$\Gamma' = v_t v_{iq}\cdots v_{ip-1}v_{il}\cdots v_{iq-1}$是$G$中比$\Gamma$更长的初级道路，以$\Gamma'$的两个端点可继续扩充，得到更长的初级道路，重复此过程，最后必得到一条包含G的所有顶点的初级道路，即G的哈密顿道路。 □

推论 6.2.3 若简单图G的任意两个顶点v_i, v_j的度数之和大于等于n，这里n是图G的顶点个数，则G存在哈密顿回路。

证明 由上面的定理知其G存在哈密顿道路H，而且该道路是G的一条极长道路，设其端点是v_1和v_n，若G不存在包含H中（也就是G中）所有顶点的回路，那么与在上述定理中的证明类似，$d(v_1)$和$d(v_n)$的和小于等于H的长度减1，即$d(v_1) + d(v_n) \leq n - 1$，与题设条件矛盾！所以$G$一定存在哈密顿回路。 □

哈密顿回路在生活以及一些研究中有十分重要的应用，其中旅行商问题与哈密顿回路的关系十分密切，而旅行商问题是一个典型的NP问题，在计算机科学中的可计算性理论及算法理论的研究中有着十分重要的地位。

参考文献

[1] 耿素云. 集合论与图论–离散数学第二分册. 北京大学出版社, 1998年2月.

[2] G. Chartrand and Zhang Ping. *Introduction to Graph Theory.* McGraw-Hill Companies, Inc., 2005. 《图论导引》(英文影印版)，人民邮电出版社，2006年6月第一版.

[3] 戴一奇，胡冠章，陈卫. 图论与代数结构. 清华大学出版社, 1995.

[4] 耿素云、屈婉玲. 离散数学(修订版). 高等教育出版社, 2004年1月.