


第八章 图论

(Graph Theory)

- 
- § 1. 图论朔源
 - § 2. 图的基本概念
 - § 3. 路与圈
 - § 4. 图的矩阵表示 (*)
 - § 5. 带权图的最短路径
 - § 6. Euler图
 - § 7. Hamilton图
 - § 8. 二分图
 - § 9. 平面图
 - § 10. 树 (*)

§ 1. 图论朔源

图论最早处理的问题是哥尼斯堡（konigsberg）城普雷格尔(pregel)河上的七桥问题。

问题：在十八世纪的东普鲁士有个哥尼斯堡城（后属于前苏联的立陶宛苏维埃社会主义共和国，其名为加里宁格勒。现属于立陶宛共和国）。哥尼斯堡城位于普雷格尔河畔，河中有两个岛，城市中的各个部分由七座桥相连。

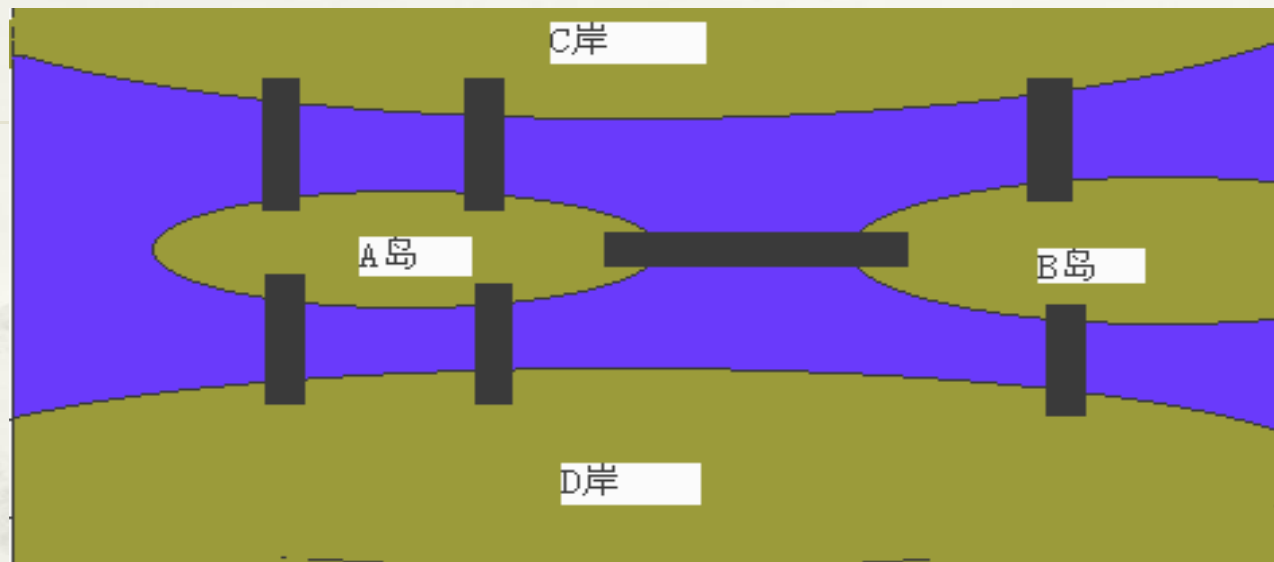
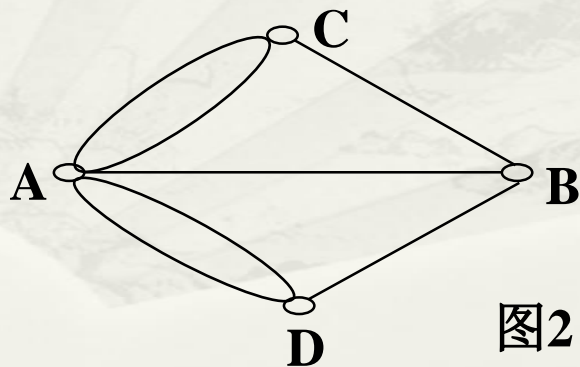


图1. 七桥图

问题：从四块陆地的任一块出发，怎样才能做到经过每座桥一次且仅一次，然后回到出发点。

1736年，瑞士数学家列昂哈德·欧拉(Leonhard·Euler) 发表了他的著名论文“哥尼斯堡七座桥”。在这篇文章中他阐述了解决七桥问题的方法，引出了图论的观点，从而被誉为图论之父，成为图论的创始人。

哥尼斯堡七桥问题归结为：在图2 所示的图中，从 A, B, C, D 任一点出发，通过每条边一次且仅一次而返回出发点的回路是否存在？后人称此问题为Euler环游。



欧拉断言这样的回路是不存在的。从图2中的任一结点出发，为了要回到原来的出发点，要求与每个结点相关联的边数均为偶数。这样才能保证从一条边进入某结点后，可从另一条边出去，而不经已走过的边。从一个结点的不同的两条边一进一出才能回到原出发点。而图2中的A, B, C, D全是与奇数条边相连，由此可知所要求的回路是不可能存在的。

欧拉给出了一个判定准则：若有Euler环游，则图中每个结点都必须是偶结点(与偶数条边相关联)；若不限定到回原出发点，则只能有两个奇结点(与奇数条边相关联)，一个起点，一个终点。

这是图论的第一篇文献。时年欧拉22岁。

注 ●列昂哈德·欧拉Leonhard . Euler (1707-1783) 瑞士数学家，后移居俄罗斯。27岁双目失明，主要靠秘书帮助工作。

本世纪40年代的数学游戏：

某人挑一担菜、并带一只狗、一只羊，要从河的北岸到南岸。由于船小，只允许带狗、羊、菜三者中的一种过河；当人不在场时狗与羊、羊与菜不能呆在一起。问此人应采取怎样的办法才能将这三样东西安全地带过河去？

方法一：不对称状态空间法

将人(person)、狗(dog)、羊(sheep)、菜(cabbage)中任意几种在一起的情况看作是一种状态，则北岸可能出现的状态共有十六种，其中安全状态有下面十种：

(人, 狗, 羊, 菜), (空); (P,D,S,C) , (\emptyset) ;

(人, 狗, 羊), (菜); (P,D,S,) , (C) ;

(人, 狗, 菜), (羊); (P,D,C) , (S) ;

(人, 羊, 菜), (狗); (P,S,C) , (D) ;

(人, 羊), (狗, 菜); (P,S) , (D,C) 。

不安全的状态有如下六种：

(人), (狗, 羊, 菜); (P), (D,S,C) ;

(人, 菜), (狗, 羊); (P,C), (D,S) ;

(人, 狗), (羊, 菜); (P,D), (S,C) 。

可将十种安全状态表示成十个结点，而渡河的全过程则看作是状态间的转移。这样，上述问题就转化为求一条从(人, 狗, 羊, 菜)或(P,D,S,C) 状态到(空) 或(\emptyset)状态的路径。图3中黑色箭头所表示的路径就是其中的一条。

另一条从 (P,D,S,C) 到 (\emptyset) 状态的路径不同的部分由图3中红色箭头所示的路径给出。

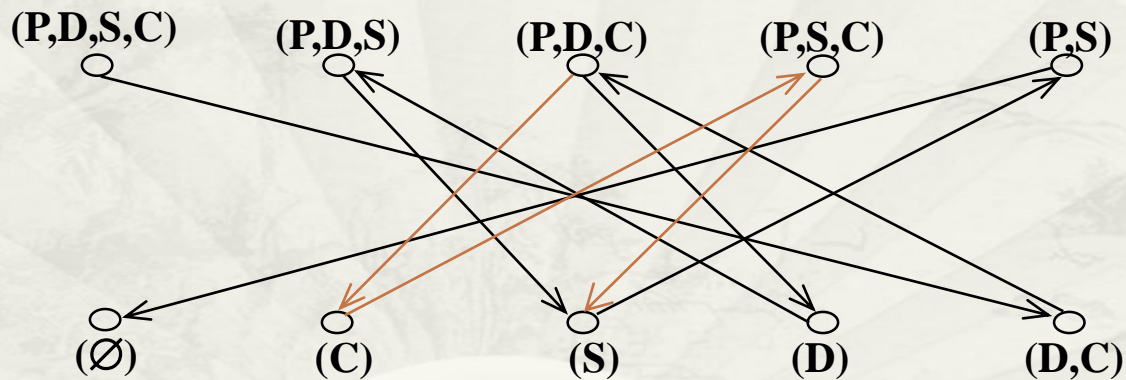


图3

方法二：对称状态空间法

方法一仅考虑了河北岸的状态，没有考虑河南岸的状态。现在将用字符串表示的两岸状态放入一个二元组中，以表示两岸状态的变化，其前者表示河北岸的状态，后者表示河南岸的状态。其图示见下面图4。

它具有对称性是明显的(状态的对称性，图的对称性，路径的对称性)。

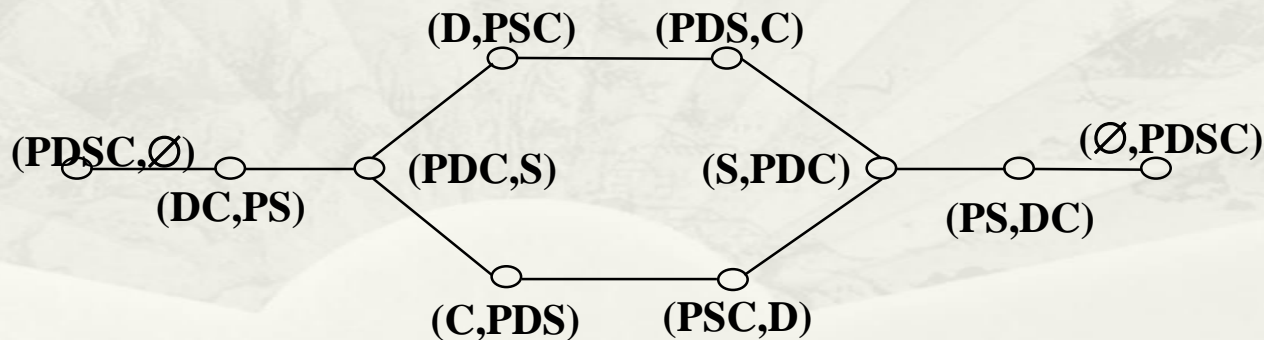


图4

注：●上述问题统称“渡河问题”。

●“三对忌妒的夫妇渡河问题”参见《离散数学基础》
[美]C. L. Liu著 刘振宏译 P162;

●“三个传教士与三个吃人肉的野人渡河问题”参见《Prolog高级程序设计》[美]L. 斯特林 E. 夏皮罗著 刘家全 邓佑译 郑守淇校 P197;

●渡河问题的条件也是可变的。比如夫妇的对数可以是四对，五对；渡河能力或渡河工具一小船的容量也是可变的。

§ 2. 图的基本概念

- 图的定义
- 图论的概念术语
- 子图与补图
- 结点的度
- 图的同构

定义1.(图的定义一) 图 $G = (V, E)$ 是一个系统，其中

(1) $V \neq \emptyset$ 是一个有限集合； V 中的每一元素 $v \in V$ 都称为图 G 的一个结点 (node ,vertex)， V 称为图 G 的结点集；

(2) E 是一个有限集合； E 中的每一元素 $e \in E$ 都称为图 G 的一条边(edge)； E 称为图 G 的边集。

注：●此定义的优点是简单，适应面广；缺点是没有规定清楚点、线之间的关系。

● 图示：图与画的联系。

定义2. (图的定义二) 图 $G = (V, E)$ 是一个系统，其中

(1) $V \neq \emptyset$ 是一有限集合； V 中的每一元素 $v \in V$ 都称为图 G 的一个结点 (node ,vertex)； V 称为图 G 的结点集；

(2) $E \subseteq V \times V$ 是一有限集合，一个 V 上的关系； E 中的每一元素 $(u, v) \in E$ 都称为图 G 的一条边(edge) (这里 $u, v \in V$)； E 称为图 G 的边集。

注：●此定义的优点是简单，规定了清楚的点、线之间的关系，很适合简单图、特别是有向图（比如第二章的关系图、哈斯图）；缺点是无法表示平行边，因此不适合多重图（比如上节的七桥图）。

例1.有四个城市 v_1 , v_2 , v_3 , v_4 , 其中 v_1 与 v_2 间有公路 e_1 相连, v_1 与 v_4 间有公路 e_2 相连, v_2 与 v_3 间有公路 e_3 相连。

该事实可用图 $G = (V, E)$ 表示。图 G 中结点集

$V = \{v_1, v_2, v_3, v_4\}$, 边集 $E = \{e_1, e_2, e_3\}$ 。

例2. 有四个程序，它们之间存在如下的调用关系： P_1 能调用 P_2 ， P_2 能调用 P_3 ， P_2 能调用 P_4 。

上述事实也可用一图 $G = (V, E)$ 来表示。图中结点集 $V = \{v_1, v_2, v_3, v_4\}$ ，边集 $E = \{(v_1, v_2), (v_2, v_3), (v_2, v_4)\}$ 。

定义3. (图的定义三) 图 $G = (V, \Sigma, E)$ 是一个系统, 其中

(1) $V \neq \emptyset$ 是一有限集合; V 中的每一元素 $v \in V$ 都称为图 G 的一个结点(node ,vertex);

V 称为图 G 的结点集;

(2) Σ 是一有限集合; Σ 中的每一元素 $\sigma \in \Sigma$ 都称为图 G 中的一个标号(label); Σ 称为图 G 的标号集;

(3) $E \subseteq V \times \Sigma \times V$ 是一有限集合, 一个三元关系; E 中的每一元素 $(u, \sigma, v) \in E$ 都称为图 G 的一条边(edge) 或弧(arc), 此边起自 u 而终于 v ; 称 u 是此边的起点, 称 σ 是此边的标号, 称 v 是此边的终点, 起点和终点统称为边的端点(这里 $u, v \in V, \sigma \in \Sigma$); E 称为图 G 的边集。

注： ●此定义是由美国哈佛大学爱伦堡教授给出的；

●此定义规定了严格的点、线之间的关系，适应面很广、特别适合多重图（比如上节的七桥图）；缺点是边表示比较复杂，简单图一般不采用。

●标号实际上是为了区别两点间的平行边而设的；标号集的大小一般就是图中平行边的最大条数(图的重数，参见下面概念)。

●当图的重数为1，即图无平行边时(简单图，参见下面概念)，有 $\Sigma = \{\sigma_1\}$ ，各边标号一样，全为 σ_1 ，这时可取掉各边标号及标号集，定义3就变成了定义2；所以定义3适合于图的一般情况，特别是(有平行边的)多重图，而定义2适合于(无平行边的)简单图。

例3. 七桥图(见图3), 按定义3, 可用一图 $G = (V, \Sigma, E)$ 来表示。

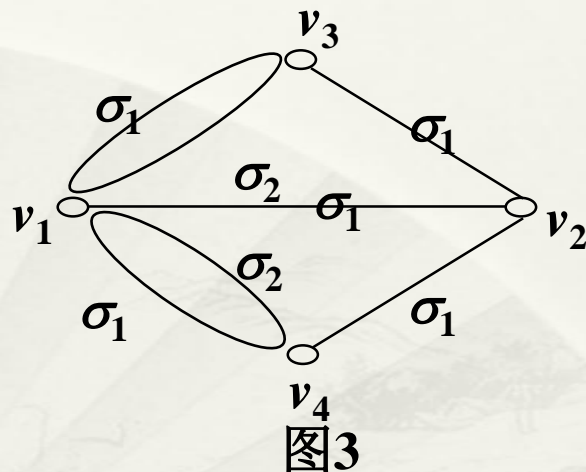
图中结点集 $V=\{v_1, v_2, v_3, v_4\}$,

图中标号集 $\Sigma=\{\sigma_1, \sigma_2\}$,

图中边集

$$E=\{(v_1, \sigma_1, v_3), (v_1, \sigma_2, v_3), (v_1, \sigma_1, v_4), \\ (v_1, \sigma_2, v_4), (v_1, \sigma_1, v_2), (v_2, \sigma_1, v_3), \\ (v_2, \sigma_1, v_4)\},$$

它的图示如图3所示。



图论的基本概念性术语和一些特殊图：

(1)(n,m)图： $|V| = n, |E| = m$ ， 即有n个结点和m条边的图称为 (n, m) 图。

(2)无向边： (undirected edges简edges)在定义3下， 若边 (u, σ, v) 与边 (v, σ, u) 表示同一条边， 则称此边为无向边。

(3)无向图： (undirected graph简graph)

所有的边都是无向边的图称为无向图。记为G。

(4)有向边: (directed arc简arc或arrow)

在定义3下, 若边 (u, σ, v) 与边 (v, σ, u) 表示不同的边, 则称此边为有向边。

(5)有向图: (directed graph简digraph)

所有的边都是有向边的图称为有向图。记为D。

(6)混和图: (mixed graph)

既有有向边又有无向边的图称为混和图。

(7)空图: (empty graph)

$V=\emptyset$ (当然 $E=\emptyset$), 即没有一个结点的图称为空图。

(8)零图: (null graph)

$E=\emptyset$, 即没有一条边的图称为零图。

(9)平凡图: (trivial graph)

$|V|=1$, 即只有一个结点的图称为平凡图。

(10)二边相邻: (adjacent)

在图中, 若两条边有一公共端点, 则称此二边相邻。

(11)二结点相邻: (adjacent)

若两个结点是同一条边的两个端点, 则称此二结点相邻。

(12)一结点与一边相关联: (incident)

若一结点是一边的一个端点, 则称此结点与该边相关联。

(13)孤立点: (isolated vertex)

不与任何边相关联的结点称为孤立点。

(14)自环: (loop)

两个端点相同的边称为自环。

(15)平行边: (parallel edges)

有相同端点(相同的起点, 相同的终点)的两条边称为平行边。

(16)重数： (multiplicity)

两结点间平行边的条数称为平行边的重数。

(17)多重图： (multiply graph)

具有平行边的图称为多重图；多重图的重数是图中平行边重数的最大者。

(18)简单图： (单图、单纯图(**simple graph**))

无平行边、无自环的图称为简单图。

(19)图的阶： (**order**)

图中结点的个数 $|V|$ 称为图的阶。

(20)完全图： (complete graph)

每一对不同的结点间都有一条边的简单图称为完全图。

n 个结点 m 条边的无向完全图： $m = \frac{n(n-1)}{2}$

n 个结点 m 条边的有向完全图： $m=n(n-1)$

n 个结点的无向完全图记为： K_n

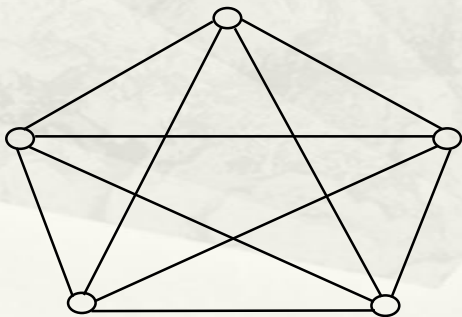


图8. K_5

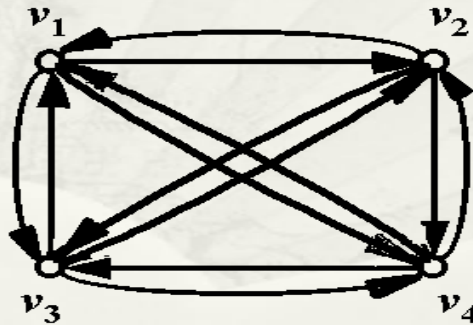


图9

定义4. (图的定义四) 图 $G=(V,E, \gamma)$ 是一个系统，其中

(1) $V \neq \emptyset$ 是一有限集合； V 中的每一元素 $v \in V$ 都称为图 G 的一个结点(node ,vertex)； V 称为图 G 的结点集；

(2) E 是一个有限集合； E 中的每一元素 $e \in E$ 都称为图 G 的一条边(edge)； E 称为图 G 的边集。

.....

(3) γ 是边到结点集的一个关联函数，即

$\gamma:E \rightarrow 2^V$ (无向图) 或 $\gamma:E \rightarrow V \times V$ (有向图)。

γ 将 E 中的每条边 $e \in E$ 与结点集 V 中的一个二元子集 $\{u, v\} \in 2^V$ (或 $\{u, v\} \subseteq V$)相关联，或与结点集 V 上的一个二元组 $(u, v) \in V \times V$ 相关联，即

$\gamma(e) = \{u, v\}$ (无向图) 或 $\gamma(e) = (u, v)$ (有向图)，

称 u 是此边的起点，称 v 是此边的终点，结点 u 和 v 统称为边的端点。

注： ●此定义是对美国库曼教授所给定义的一个修正；

●此定义的优点是适应面较广，尤其是将边看作是和结点同样的独立的研究对象，边不再是由结点表示的一个附属对象，用函数概念规定了点、线之间的严格关联关系，这样一来，就便于边概念的进一步推广（比如引出超图概念）；缺点是关联函数表示比较烦琐，简单图一般不采用。

例4. 七桥图(见图10), 按定义4, 可用一图 $G = (V, E, \gamma)$ 来表示。

图中结点集 $V = \{v_1, v_2, v_3, v_4\}$,

图中边集 $E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$,

图中关联函数 $\gamma: E \rightarrow 2^V$ 使

$$\gamma(e_1) = \{v_1, v_3\}, \gamma(e_2) = \{v_1, v_3\},$$

$$\gamma(e_3) = \{v_1, v_2\}, \gamma(e_4) = \{v_1, v_4\},$$

$$\gamma(e_5) = \{v_1, v_4\}, \gamma(e_6) = \{v_2, v_3\},$$

$$\gamma(e_7) = \{v_2, v_4\}.$$

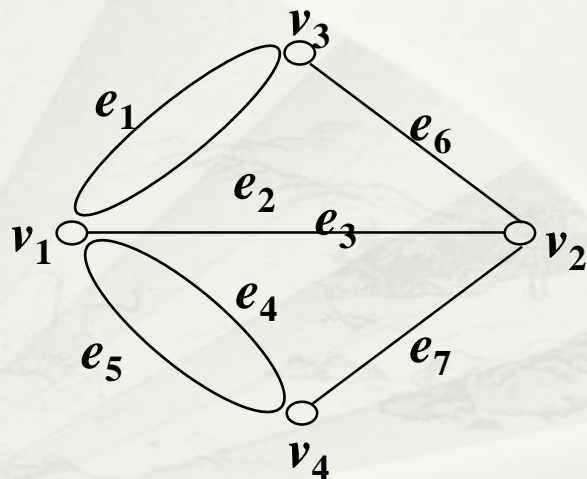


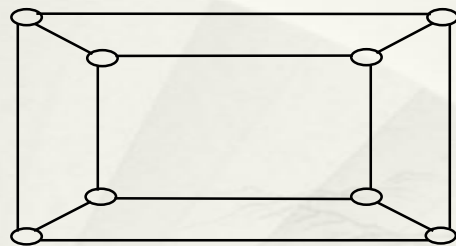
图10

定义5.子图(subgraph)

设 $G=(V, E)$ 和 $G'=(V', E')$ 是两个(有向的或无向的)图。

- (1) 若 $V' \subseteq V$ 且 $E' \subseteq E$, 则称 G' 为 G 的子图;
- (2) 若 $V' \subset V$ 且 $E' \subset E$, 则称 G' 为 G 的真子图(proper-);
- (3) 若 $V'=V$ 且 $E' \subseteq E$, 则称 G' 为 G 的生成子图(spanning-);
- (4) 当 $V'=V$ 且 $E'=E$, 或 $V'=V$ 且 $E'=\emptyset$ 时, 称 G' 为 G 的平凡子图(trivial-); 即, 图 G 本身和 G 的零图是 G 的平凡子图。

例5. 图G及其子图、生成子图、真子图、平凡子图.....



G

图11

定义6.商图(quotient graph)

设 $G = (V, E)$ 是一(有向的或无向的)图, $R \subseteq V \times V$ 是一结点集 V 上的等价关系。

那么, 定义图 G 关于等价关系 R 的商图为简单图

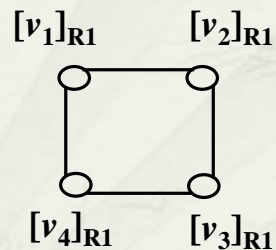
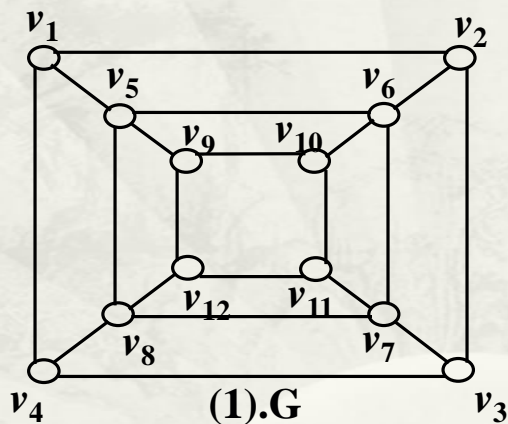
$G^R = (V^R, E^R)$ 其中:

$V^R = V/R = \{[v]_R \mid v \in V\}$ 是 V 关于等价关系 R 的商集;

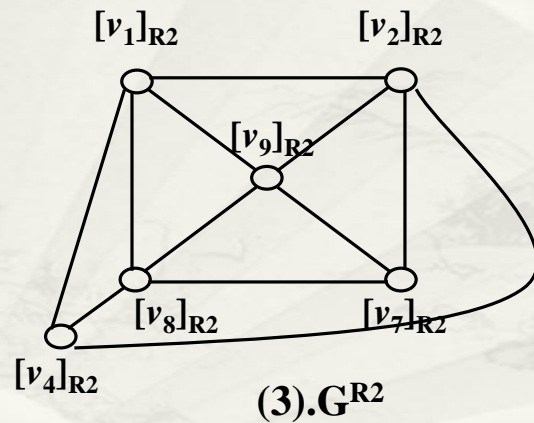
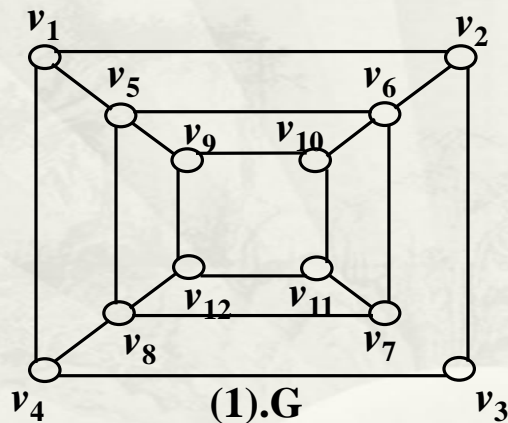
$E^R = \{([u]_R, [v]_R) \mid [u]_R \in V^R \wedge [v]_R \in V^R \wedge (\exists u' \in [u]_R)(\exists v' \in [v]_R) ((u', v') \in E)\}。$

例6. 图G如(1)图所示;

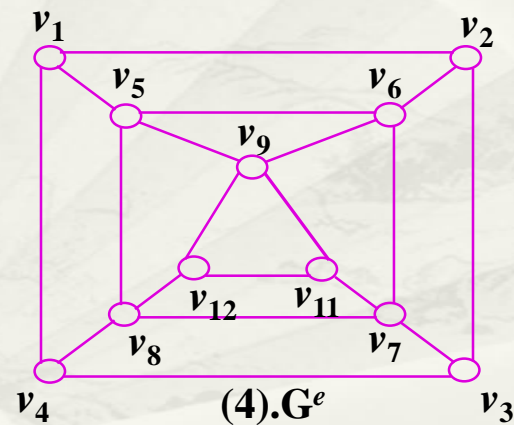
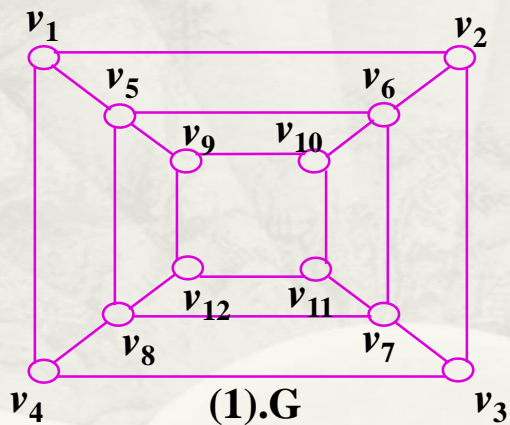
其关于等价关系 R_1 (以划分 $\{\{v_1, v_5, v_9\}, \{v_2, v_6, v_{10}\}, \{v_3, v_7, v_{11}\}, \{v_4, v_8, v_{12}\}\}$ 给出)的商图 G^{R_1} 如(2)图所示;



关于等价关系 R_2 (以划分 $\{\{v_1, v_5\}, \{v_2, v_3, v_6\}, \{v_4\}, \{v_7\}, \{v_8\}, \{v_9, v_{10}, v_{11}, v_{12}\}\}$ 给出)的商图 G^{R_2} 如(3)图所示;



关于等价关系 R_3 (以划分 $\{\{v_1\}, \{v_2\}, \{v_3\}, \{v_4\}, \{v_5\}, \{v_6\}, \{v_7\}, \{v_8\}, \{v_9, v_{10}\}, \{v_{11}\}, \{v_{12}\}\}$ 给出)的商图 G^{R_3} 如(4)图所示(称为由一条边 $e=(v_9, v_{10})$ 所导出的商图, 记为 G^e);

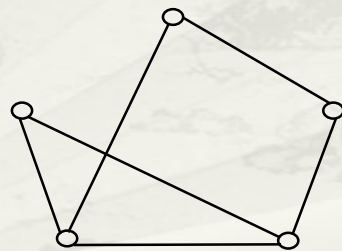


定义7.补图(complement graph)

设 $G = (V, E)$ 为一简单图， $G^* = (V, E^*)$ 是与图 G 相应的完全图。定义图 G 的补图 $\overline{G} = (V, \overline{E})$ ，其中：

$$\overline{E} = E^* \setminus E。$$

例7.图 G 及其相应的完全图、补图
分别如下图13所示.....



G

图13

(21)结点的出度： (out-degree)

有向图中以结点 v 为起点的有向边的条数称为结点 v 的出度。记为 $\overleftarrow{\deg}(v)$ 。

(22)结点的进度： (入度(in-degree))

有向图中以结点 v 为终点的有向边的条数称为结点 v 的进度。记为 $\overrightarrow{\deg}(v)$ 。

(23)结点的度： (degree)

图中与结点 v 关联的边的条数称为结点 v 的度。记为 $\deg(v)$ 。

(24)奇结点: (odd vertex)

度数为奇数的结点称为奇结点。

(25)偶结点: (even vertex)

度数为偶数的结点称为偶结点。

(26)图G的最小度: (minimal degree)

图G中各结点度数的最小者。记为 $\delta(G)$ 。

(27)图G的最大度: (maximum degree)

图G中各结点度数的最大者。记为 $\Delta(G)$ 。

(28)正则图: (regular graph)

若图G中各结点的度数都相等, 则称图G 是正则图。 显然, 这时 $\delta(G)=\Delta(G)$ 。

(29) k -正则的: (k - regular)

若图G中各结点的度数都相等, 且为 k , 则称图G 是 k -正则的, 或 k 度正则的。 显然, 这时 $\delta(G)=\Delta(G)= k$ 。

(30)悬挂点: (hang vertex)

度数为1的结点称为悬挂点。

(31)悬挂边: (hang edge)

与悬挂点关联的边称为悬挂边。

例8.在有向图14中，其中各结点的度数：

$$\overleftarrow{\deg}(v_1)=3, \overrightarrow{\deg}(v_1)=4, \deg(v_1)=7 ;$$

$$\overleftarrow{\deg}(v_2)=2, \overrightarrow{\deg}(v_2)=2, \deg(v_2)=4 ;$$

$$\overleftarrow{\deg}(v_3)=3, \overrightarrow{\deg}(v_3)=1, \deg(v_3)=4 ;$$

$$\overleftarrow{\deg}(v_4)=1, \overrightarrow{\deg}(v_4)=1, \deg(v_4)=2 ;$$

$$\overleftarrow{\deg}(v_5)=0, \overrightarrow{\deg}(v_5)=0, \deg(v_5)=0 ;$$

$$\overleftarrow{\deg}(v_6)=0, \overrightarrow{\deg}(v_6)=1, \deg(v_6)=1 ;$$

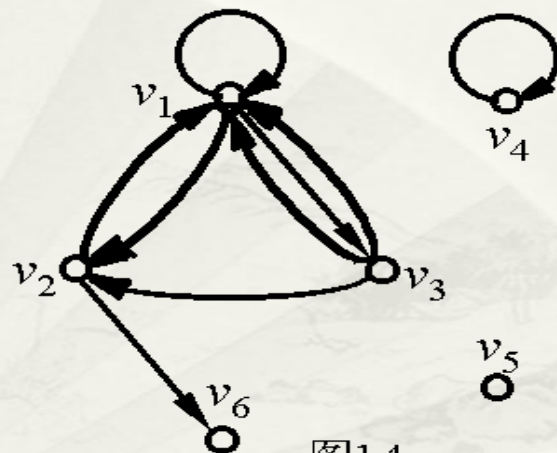


图14

奇结点： v_1, v_6 ; 偶结点： v_2, v_3, v_4, v_5 ;

悬挂点： v_6 ; 悬挂边： (v_2, v_6) 。

有如下结论：

(1)所有结点的度数之和等于边数的二倍；

$$7+4+4+2+0+1=2 \times 9$$

(2)所有结点的进度之和等于出度之和等于边数；

$$4+2+1+1+0+1=3+2+3+1+0+0=9$$

(3)所有奇结点的度数之和是偶数： $7+1=8$ 。

例9.在下面的无向图中，其中：

$\deg(v_1)=5, \deg(v_2)=3, \deg(v_3)=3,$

$\deg(v_4)=2, \deg(v_5)=0, \deg(v_6)=1$;

奇结点： v_1, v_2, v_3, v_6 ;

偶结点： v_4, v_5 ;

悬挂点： v_6 ; 悬挂边： (v_2, v_6) 。

并且有如下结论：

(1)所有结点的度数之和等于边数的二倍；

$$5+3+3+2+0+1=2 \times 7$$

(2)所有奇结点的度数之和是偶： $5+3+3+1=12$ 。

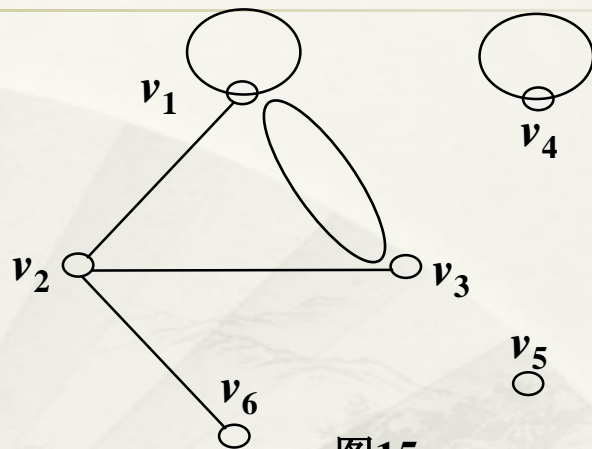


图15

定理1. 设 $G = (V, E)$ 是 (n, m) 图, 则

(1) 无向图 G 中, 所有结点的度之和等于边数的二倍; 即

$$\sum_{i=1}^n \text{dev}(v_i) = 2m \text{ 。}$$

(2) 有向图 G 中, 所有结点的进度之和等于出度之和等于边数;

即

$$\sum_{i=1}^n \overrightarrow{\text{dev}}(v_i) = \sum_{i=1}^n \overleftarrow{\text{dev}}(v_i) = m \text{ 。}$$

[证]. (采用 ‘数钱法’)

(1) 因为无向图G中的每条无向边都与两个结点相关联, 所以每条边都能给G中结点的度数贡献2, 因而G中所有的m条边能给G中结点的度数总计贡献 $2m$, 故G中所有结点的度数之和为 $2m$, 即

$$\sum_{i=1}^n \text{dev}(v_i) = 2m \quad .$$

(2)对于有向图G，每条有向边都与一个终点和一个起点相关联，因此每条有向边都能给G中结点的进度贡献1，给出度贡献1，因而G中所有的m条边能给G中结点的进度总计贡献m，给出度总计贡献m，故G中所有结点的进度之和等于出度之和等于边数m，即

$$\sum_{i=1}^n \overrightarrow{\text{dev}}(v_i) = \sum_{i=1}^n \overleftarrow{\text{dev}}(v_i) = m。$$

定理2.任何图中所有奇结点的度数之和是偶数。

[证].设图中共有 n 个结点；其中奇结点的个数为 n_1 ，并且不妨设奇结点为 u_1, u_2, \dots, u_{n_1} ；偶结点的个数为 n_2 (当然有 $n_1 + n_2 = n$)，并且不妨设偶结点为 w_1, w_2, \dots, w_{n_2} ，其对应的度数为 $2k_1, 2k_2, \dots, 2k_{n_2}$ ；于是奇结点度数之和

$$\begin{aligned}\sum_{i=1}^{n_1} \text{dev}(u_i) &= \sum_{i=1}^n \text{dev}(v_i) - \sum_{i=1}^{n_2} \text{dev}(w_i) \\ &= 2m - (2k_1 + 2k_2 + \dots + 2k_{n_2}) \quad (\text{定理1}) \\ &= 2[m - (k_1 + k_2 + \dots + k_{n_2})]\end{aligned}$$

是偶数。

推论1.(握手引理) 在一次集会上和奇数个人握过手的人的数目是偶数。

[证]. 用结点表示人，用边表示两人相互握过手，从而便可得到一个图。这个图表达了参加集会的人之间彼此握手打招呼的情况。于是直接应用定理2，即可知推论的结论成立。

例题：



下列选项中的数值表示一个简单无向图中各结点的度数，能画出图的是()。

☐ A (1, 2, 2, 3, 4, 5)

☐ B (1, 2, 3, 4, 5, 5)

☐ C (2, 3, 3, 4, 5, 6)

☒ D (1, 1, 1, 2, 2, 3)

提交

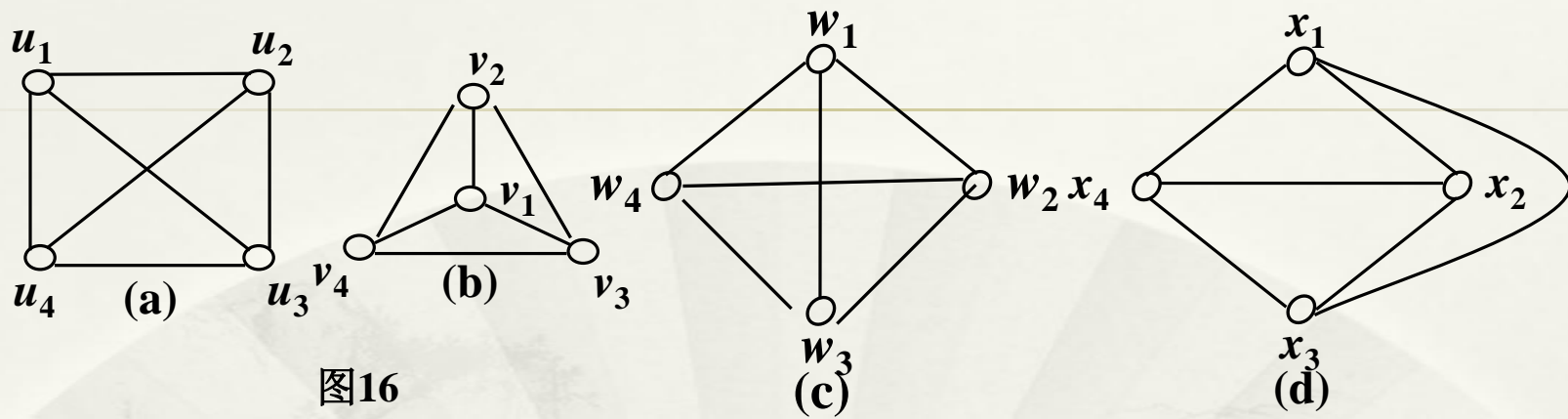


图16

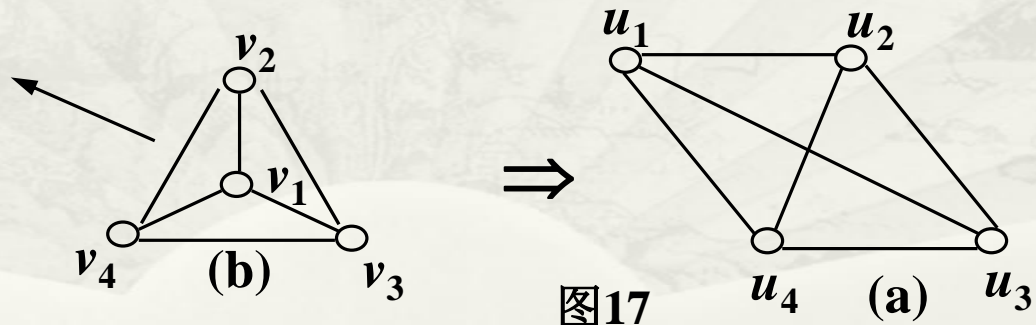


图17

定义8.图的同构(isomorphism of graphs)

(1)称 $G=(V,E)$ 及 $G'=(V',E')$ 二图同构, 记为 $G \cong G' \Leftrightarrow$ 存在着两个双射函数 φ 及 ψ ,

$\varphi: V \rightarrow V'$, $\psi: E \rightarrow E'$, 使得 $\psi(u,v)=(u',v') \Rightarrow \varphi(u)=u' \wedge \varphi(v)=v'$ (*)

(2)称 $G=(V,E,\gamma)$ 及 $G'=(V',E',\gamma')$ 二图同构, 记为 $G \cong G' \Leftrightarrow$ 存在着两个双射函数 φ 及 ψ ,

$\varphi: V \rightarrow V'$, $\psi: E \rightarrow E'$, 使得

$$\psi(e)=e' \wedge \gamma(e)=\{u,v\} \wedge \gamma'(e')=\{u',v'\} \Rightarrow \varphi(u)=u' \wedge \varphi(v)=v'$$

或 $\psi(e)=e' \wedge \gamma(e)=(u,v) \wedge \gamma'(e')=(u',v') \Rightarrow \varphi(u)=u' \wedge \varphi(v)=v'$

} (**)

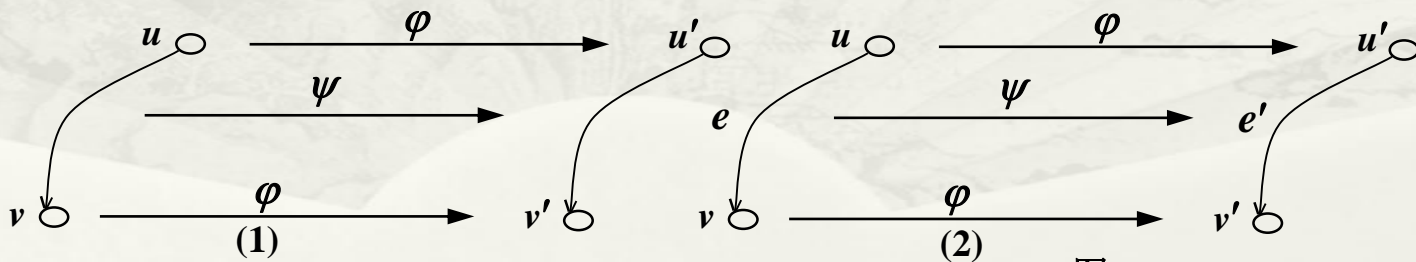


图18

注： ●此定义(1)中(*)式也可改写为：

$$\psi(u,v) = (\phi(u), \phi(v)) \quad (*)'$$

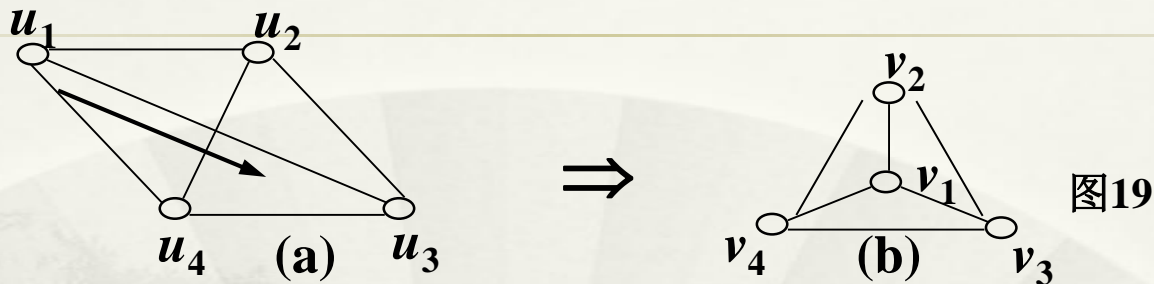
此定义(2)中(**)式也可改写为：

$$\left. \begin{aligned} \text{或} \quad \gamma(e) = \{u,v\} &\Rightarrow \gamma'(\psi(e)) = \{\phi(u), \phi(v)\} \\ \gamma(e) = (u,v) &\Rightarrow \gamma'(\psi(e)) = (\phi(u), \phi(v)) \end{aligned} \right\} \quad (**')$$

这实际上就是图的同态公式；

●图的同构远比代数系统的同构复杂。这是因为图的同构在同态公式中牵扯着两个(甚至三个,考虑定义三)双射函数的交叉关系,而代数系统的同构在同态公式中只有一个双射函数。因此图的同构问题不象代数系统的同构问题那样有许多进展、有几个定理好用,迄今为止,没有任何进展,没有任何定理可用,仅仅只能用定义。

例10.图19中的(a)和(b)二无向图是同构的。



(采用变形法)其同构函数为 φ 及 ψ ，使得

$$\varphi(u_1)=v_1, \varphi(u_2)=v_2, \varphi(u_3)=v_3, \varphi(u_4)=v_4;$$

$$\psi(u_1, u_2)=(v_1, v_2), \psi(u_1, u_3)=(v_1, v_3), \psi(u_1, u_4)=(v_1, v_4),$$

$$\psi(u_2, u_3)=(v_2, v_3), \psi(u_2, u_4)=(v_1, v_2), \psi(u_3, u_4)=(v_3, v_4);$$

从而 φ 及 ψ 是两个双射函数，且满足同态公式(*)，因此(a)和(b)二图是同构的。

例11.图20中的(a)和(b)二无向图是同构的。

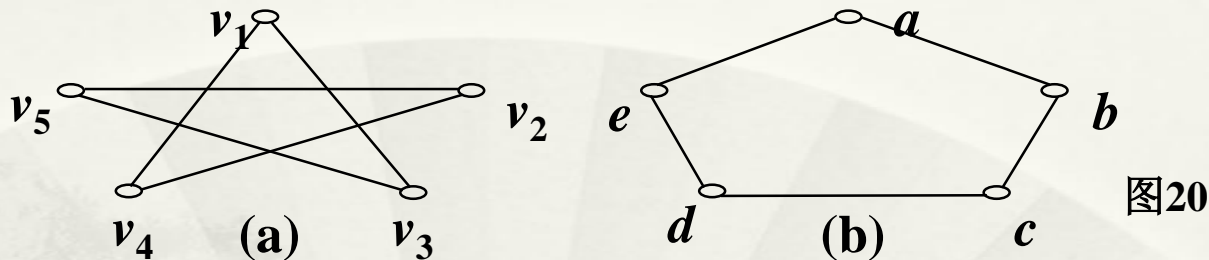


图20

(采用抽路蹦圈法)其同构函数为 ϕ 及 ψ ，使得

$$\phi(v_1)=a, \phi(v_2)=c, \phi(v_3)=e, \phi(v_4)=b, \phi(v_5)=d ;$$

$$\psi(v_1, v_3)=(a, e), \psi(v_1, v_4)=(a, b), \psi(v_2, v_4)=(c, b),$$

$$\psi(v_2, v_5)=(c, d), \psi(v_3, v_5)=(e, d) ;$$

从而 ϕ 及 ψ 是两个双射函数，且满足同态公式(*)，因此(a)和(b)二图是同构的。

两个图能否同构，从直观的意义上来讲是能否将其中的一个图示通过某些“变形”后使它成为另一个图示的形状。如将(a)图示作如下变形一蹦圈后就可得到(b)图示：

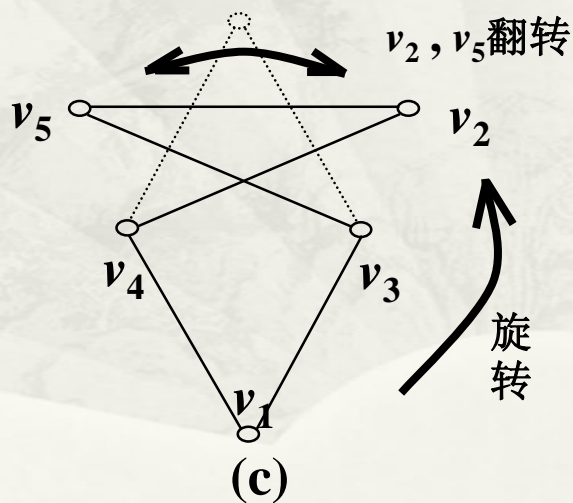
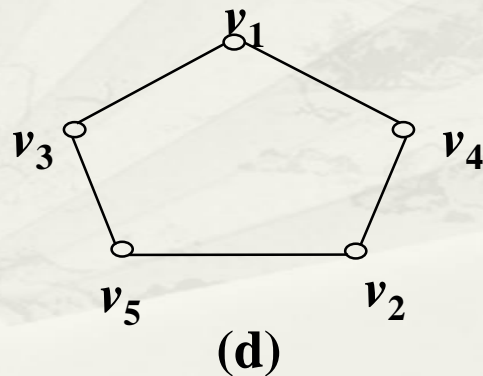


图21



例12.图22中的(a)和(b)二无向图是同构的。

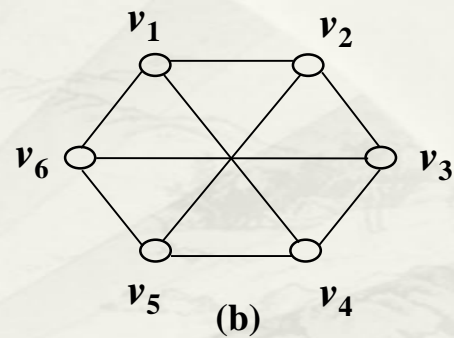
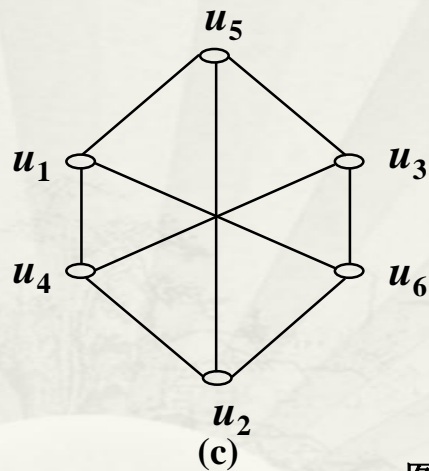
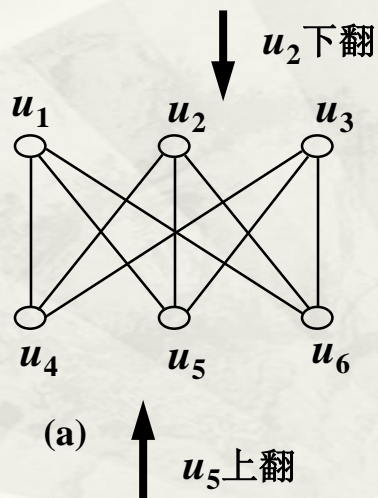


图22

(采用抽路蹦圈法)其同构函数为 φ 及 ψ ，使得

$$\varphi(u_1)=v_1, \varphi(u_2)=v_5, \varphi(u_3)=v_3,$$

$$\varphi(u_4)=v_6, \varphi(u_5)=v_2, \varphi(u_6)=v_4;$$

$$\psi(u_1, u_4)=(v_1, v_6), \psi(u_1, u_5)=(v_1, v_2), \psi(u_1, u_6)=(v_1, v_4),$$

$$\psi(u_2, u_4)=(v_5, v_6), \psi(u_2, u_5)=(v_5, v_2), \psi(u_2, u_6)=(v_5, v_4),$$

$$\psi(u_3, u_4)=(v_3, v_6), \psi(u_3, u_5)=(v_3, v_2), \psi(u_3, u_6)=(v_3, v_4);$$

从而 φ 及 ψ 是两个双射函数，且满足同态公式(*)，因此(a)和(b)二图是同构的。

例13.图23中的(a)和(b)二有向图是同构的。

(采用抻路蹦圈法)其同构函数为 φ 及 ψ ，使得

$$\varphi(u_1)=v_1, \varphi(u_2)=v_2, \varphi(u_3)=v_4, \varphi(u_4)=v_3;$$

$$\psi(u_1, u_2)=(v_1, v_2), \psi(u_1, u_4)=(v_1, v_3),$$

$$\psi(u_2, u_3)=(v_2, v_4), \psi(u_4, u_3)=(v_3, v_4);$$

从而 φ 及 ψ 是两个双射函数，且满足同态公式(*)，因此(a)和(b)二图是同构的。

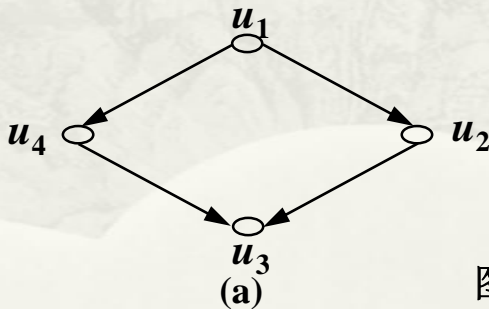
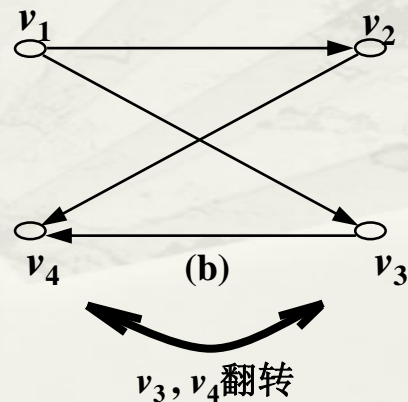


图23



若两个图同构，则它们必须满足：

- (1) 结点个数相等； (2) 边数相等；
- (3) 对应结点的进度、出度、度数均相等；
- (4) 度数相同的结点个数相等；
- (5) 平行边对应，重数相等；
- (6) 自环对应；悬挂点对应；孤立点对应；
- (7) 结点间的相邻关系对应；边间的相邻关系对应；结点与边的关联关系对应；
- (8) 圈对应；路对应；
- (9) 对应圈的长度相等；对应路的长度相等；
-

例14.图24中(a)、(b)两图不同构。

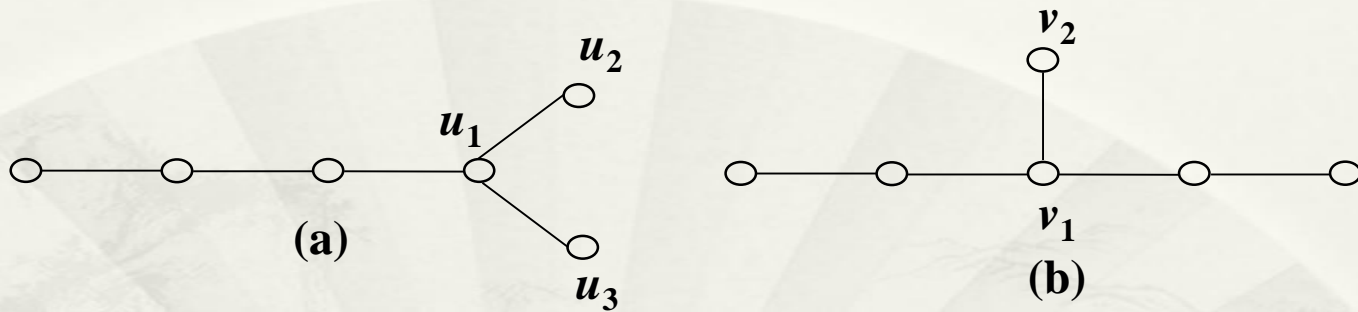


图24

Ulam猜想(1960).

1960年 Ulam提出如下关于图同构的著名猜想，至今无人能够证明或否定。

设 $G=(V,E)$ 及 $G'=(V',E')$ 是两个图，其结点集分别为

$V=\{v_1, v_2, \dots, v_n\}$ 及 $V'=\{v_1', v_2', \dots, v_n'\}$ ($n \geq 3$)，则

$(\forall i \in N)(1 \leq i \leq n)(H_i \cong H_i') \Rightarrow (G \cong G')$ ，即

若 G 和 G' 的 n 对特殊真子图对应同构，则 G 和 G' 同构。

这里： $H_i = G \setminus \{v_i\} = (V_i, E_i)$,

$V_i = V \setminus \{v_i\}$, $E_i = \{(u, v) \mid (u, v) \in E \wedge u \neq v_i \wedge v \neq v_i\}$;

$H_i' = G' \setminus \{v_i'\} = (V_i', E_i')$,

$V_i' = V' \setminus \{v_i'\}$, $E_i' = \{(u', v') \mid (u', v') \in E' \wedge u' \neq v_i' \wedge v' \neq v_i'\}$ 。

§ 3.路与圈

- 定义与实例
- 基本概念
- 可达性
- 图的连通性

§ 3.路与圈

定义1.途径(way)

设 $G=(V,E,\gamma)$ 是一图。 G 的有限非空点边交错序列

$$w=v_0, e_1, v_1, e_2, v_2, \dots, e_k, v_k$$

若满足条件:

$$(1)\gamma(e_i)=\{v_{i-1}, v_i\}(\text{或}(v_{i-1}, v_i)) \quad (1\leq i\leq k)$$

(2)当 e_i 和 e_{i+1} 不是自环时, 有 $e_{i+1}\neq e_i \quad (1\leq i\leq n)$ (无向图), 则称其为 G 的一条从 v_0 到 v_k 的途径。 v_0 称为途径 w 的起点, v_k 称为途径 w 的终点, k 称为途径 w 的长度。

注：•当 $G=(V,E)$ 是一简单图时，在实际应用中，常用**纯边列**或**纯点列**的方式表示**途径**：

(1) $w=(e_1, e_2, \dots, e_k)$;

(2) $w=((v_0, v_1), (v_1, v_2), \dots, (v_{k-1}, v_k))$;

(3) $w=(v_0, v_1, v_2, \dots, v_{k-1}, v_k)$ 。

•途径 w 实际上是一个**动态的过程**；其在图 G 上的**静态轨迹**是图 G 的一个子图 $G(w)=(V(w), E(w), \gamma(w))$ ，其中：

$V(w)=\{v_0, v_1, v_2, \dots, v_k\}$ （去掉重复）

$E(w)=\{e_1, e_2, \dots, e_k\}$ （去掉重复）

$\gamma(w): E(w) \rightarrow 2^{V(w)}$ （无向图） 或 $\gamma: E(w) \rightarrow V(w) \times V(w)$ （有向图）

使得 $\gamma(w)(e_i)=\{v_{i-1}, v_i\}$ (或 (v_{i-1}, v_i)) ($1 \leq i \leq n$) （去掉重复）。

- 途径定义1中条件(1)、(2)的情况如下图1所示:

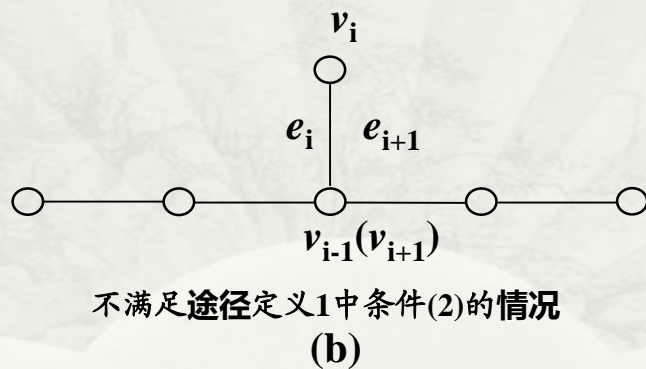
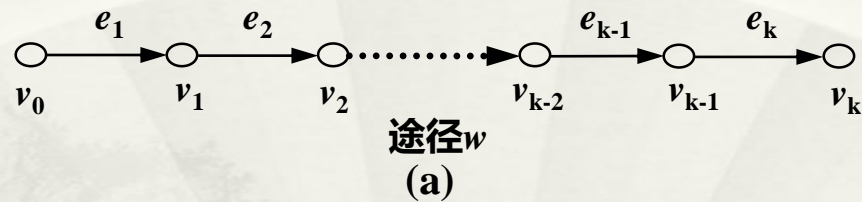


图1

定义2.路(path)、圈(cycle)

设 $G=(V,E)$ 是一图, $w=(v_0, v_1, v_2, \dots, v_{k-1}, v_k)$ 是一途径。

(1)若 $v_0 \neq v_k$, 则称此途径 w 是从 v_0 到 v_k 的一条路; 记为

$P=(v_0, v_1, v_2, \dots, v_{k-1}, v_k)$, 并称 k 为路 P 的长度(即 $|P|=k$)。

(2)若 $v_0 = v_k$, 则称此途径 w 是一个圈; 记为

$C=(v_0, v_1, v_2, \dots, v_{k-1}, v_k)$, 并称 k 为圈 C 的长度(即 $|C|=k$)。

定义3.可达性(reachablility)、连通性(connectivity)

设 $G=(V,E)$ 是一无向图, $u, v \in V$

(1)若存在着从结点 u 到结点 v 的一条路 P , 则称从结点 u 到结点 v 是可达的;

(2)若图 G 中任何两结点都是可达的, 则称此图 G 是连通的。否则, 称图 G 是非连通的。

注一： ●可达概念可以看作结点间的一个二元关系——可达关系；

●在无向图中，规定任一结点自己到自己总是可达的，即可达关系具有自反性；

●在无向图中，可达性是相互的，从结点 u 可达结点 v ，则从结点 v 也可达结点 u ，即可达关系是对称的；

●可达关系是传递的，即若从结点 u 可达结点 v ，又从结点 v 可达结点 w ，则从结点 u 也可达结点 w ；

●一般地，当从结点 u 可达结点 v 时，它们之间不一定只有一条路，可能会有若干条路。称从结点 u 到结点 v 的所有路中长度最短的那一条为短程线，并将短程线的长度叫做从结点 u 到结点 v 的距离，用 $d(u, v)$ 表示。

规定：

(1) $d(u, u)=0$ ；

(2) 若结点 u 到结点 v 不可达，则 $d(u, v)=\infty$ 。

• **短程线**不一定是唯一的,有时可能会有**好几条**；

• **按照通常的理解，距离概念一般都具有下列性质：**

(1) **非负性**： $d(u, v) \geq 0$ ；

(2) **对称性**； $d(u, v) = d(v, u)$ ；

(3) **三角不等式**： $d(u, v) + d(v, w) \geq d(u, w)$

• **对无向图，上述性质全成立；**

• **对有向图来说，第二条，对称性质不成立。**

注二：●连通性是有强弱之分的；

(1) 若图G中任二结点间都至少存在着一一条路可达，则称图G是1-连通的；

(2) 若图G中任二结点间都至少存在着k条不同的路可达，则称图G是k-连通的 ($k \geq 2$)；

● 通常所说的连通性实际上是指1-连通。1-连通的连通性较差，重要的信道图网络，比如军事信道图网络，其连通性至少在2-连通以上。

例1.在图2中由结点 v_1 到结点 v_3 的路有：

$$P_1=(v_1,v_2,v_3)$$

$$P_2=(v_1,v_4,v_3)$$

$$P_3=(v_1,v_2,v_4,v_3)$$

$$P_4=(v_1,v_2,v_4,v_1,v_2,v_3)$$

$$P_5=(v_1,v_2,v_4,v_1,v_4,v_3)$$

$$P_6=(v_1,v_1,v_2,v_3)$$

....

在图2中，由结点 v_1 到 v_1 的圈有：

$$C_1=(v_1,v_1)$$

$$C_2=(v_1,v_2,v_1)$$

$$C_3=(v_1,v_2,v_3,v_1)$$

$$C_4=(v_1,v_4,v_3,v_1)$$

$$C_5=(v_1,v_2,v_3,v_2,v_1)$$

$$C_6=(v_1,v_2,v_3,v_2,v_3,v_1)$$

.....

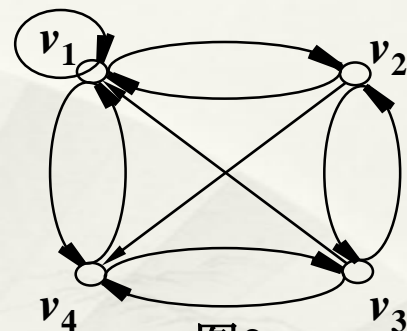


图2

例2.路的概念可以用来描述很多东西，如：

(1)在Pascal语言中，一个复合语句以BEGIN开始，END结束，其中若干个语句用分号隔开。所以，执行一个Pascal语言中的一条复合语句，可以认为是从图3(a)中的BEGIN开始到END结束走完一条路。

(2)Pascal语言中的“标识符”可以认为是图3(b)中从“入口”到“出口”的一条路。

(3)一个二进制数可以认为是图3(c)中从“入口”到“出口”的一条路。

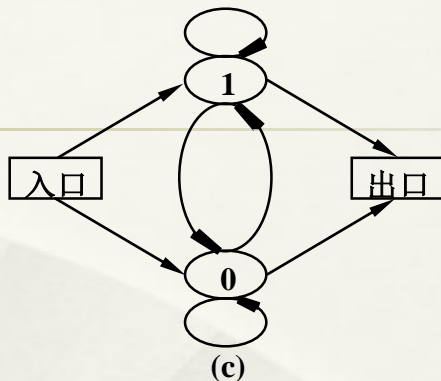
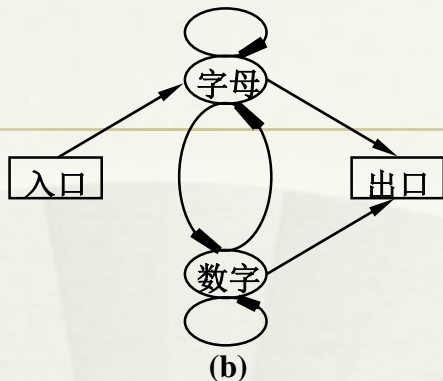
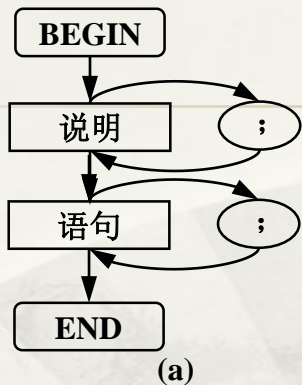


图3

注：•实际上，分程序是如下的点列：

BEGIN 说明 ; 说明 ; ... ; 说明 ; 语句 ; 语句 ; ... ; 语句 ; END

而不是边列。

•各种程序设计语言的文法都可用图来表示，而且比通常所用的Backus元语言表示的更好、更直观。

定义4.简单路(simple path)简单圈(simple cycle)

无重复边的路称为简单路;

无重复边的圈称为简单圈。

定义5.初级路(elementary path)初级圈(elementary cycle)

无重复点的路称为初级路;

无重复点的圈称为初级圈。

例4.在图4中由结点 v_1 到结点 v_5 的路有：

$$P_1=(v_1,v_2,v_5) ;$$

$$P_2=(v_1,v_2,v_6,v_2,v_5) ;$$

$$P_3=(v_1,v_4,v_5,v_6,v_2,v_6,v_2,v_5) \circ$$

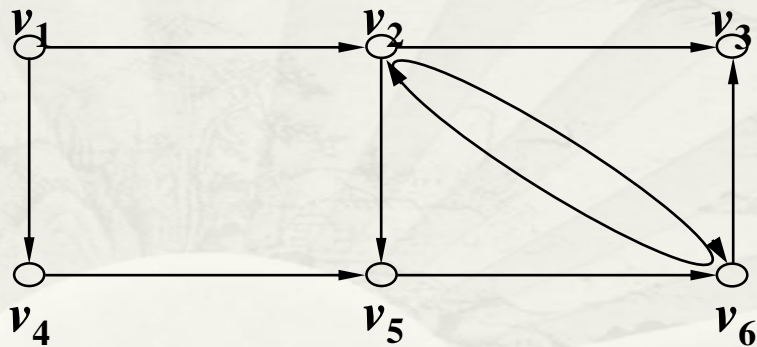


图4

定理1. 设 $G=(V,E)$ 为一简单图，若 $|V|=n$ ，则

(1) G 中任一初级路的长度均不超过 $n-1$ ；

(2) G 中任一初级圈的长度均不超过 n 。

[证]. (1)首先, 在任一初级路中, 各结点都是互不相同的; 其次, 在任一长度为 k 的初级路中, 不同结点的个数必然为 $k+1$ 。由于图 G 中只有 n 个不同的结点, 所以 G 中任一初级路的长度不会超过 $n-1$ 。

(2)在任一长度为 k 的初级圈中, 其不同的结点的个数也为 k 。而图 G 中仅有 n 个不同的结点, 故任一初级圈的长度不会超过 n 。

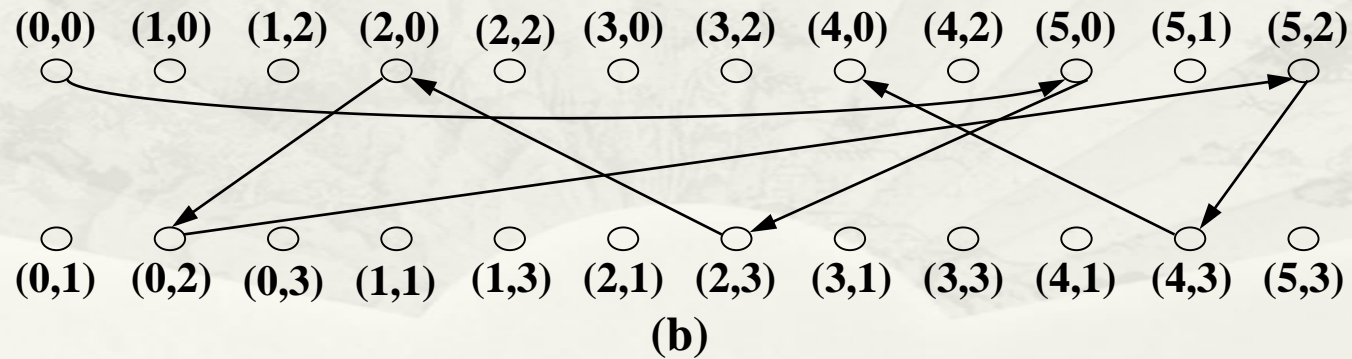
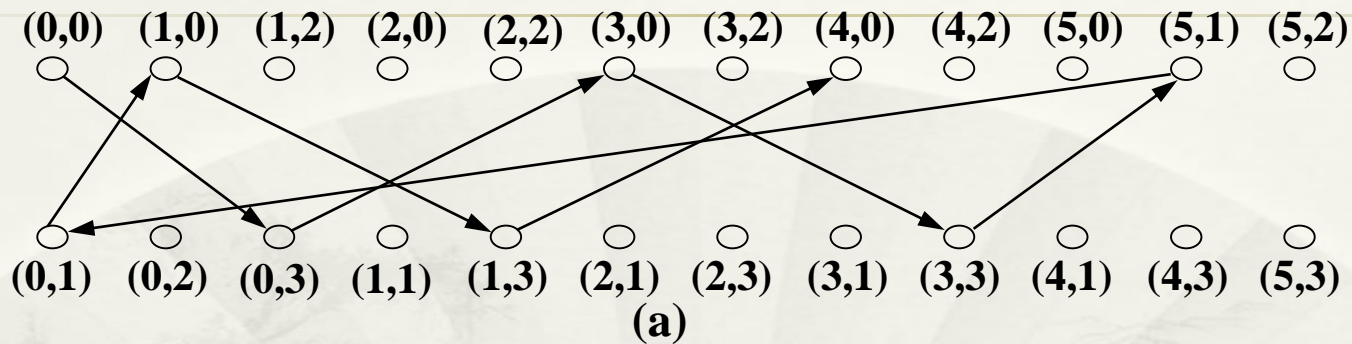
例5.分油问题(三倒油葫芦): 有三个油桶a,b,c分别可装8斤,5斤,3斤油。假设a桶已装满了油, 在没有其它度量工具的情况下, 如何将油平分?

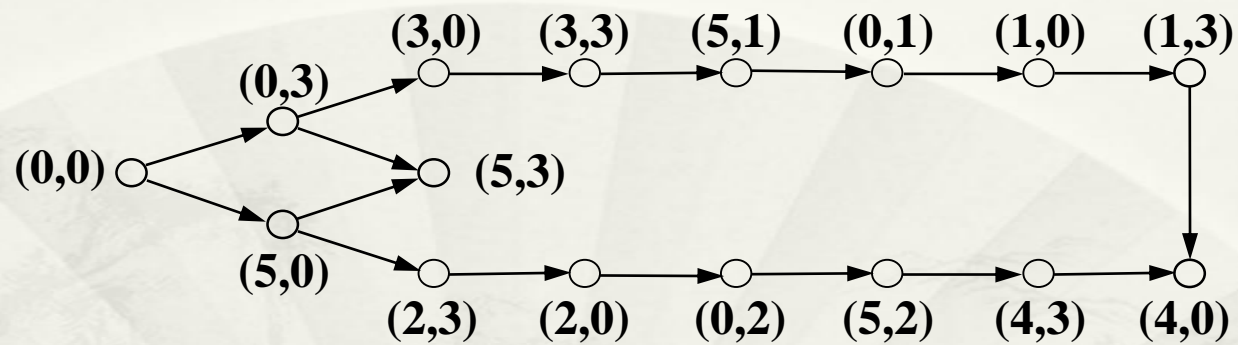
[解].首先, 由于没有其它工具, 只能利用这三只油桶来回倾倒, 以达到分油的目的; 为了分得准确, 规定倒油时必须将油桶倒满或倒空。

.....

其次，用二元组 (b,c) 来表示 b,c 两个桶中装油的各种可能状态，由于 $0 \leq b \leq 5, 0 \leq c \leq 3$ ，故 (b,c) 共有24种不同的状态。每一种状态用一结点表示，两结点间有边相连当且仅当这两种状态可通过倒油的方法互相转换。起始状态为 $(0,0)$ ，终结状态为 $(4,0)$ ，其它为中间状态。同时，为了尽快将油分好，要求中间状态不重复出现。这样，分油问题就转化为：在具有24个形如 (b,c) 的结点的图中，寻找由结点 $(0,0)$ 到结点 $(4,0)$ 的初级路。

这样的初级路有两条，如图5中的(a),(b)所示。(a),(b)可合并为(c)。





(c)

图5

定义6.连通支(分图(connected component))

无向图中极大的连通子图称为一个连通支。

例6.在图6中有三个连通支(分图)；因此不是连通图。

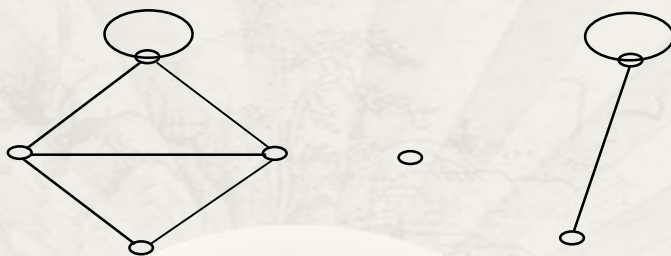


图6

设 $G = (V, E)$ 是一个简单无向图, $|V| = 9$, 如果 $m=29$, 那么 G 一定是连通图。

- ☒ A 此命题成立
- ☐ B 此命题不成立

定义7.强连通 单向连通 弱连通

设 $G=(V,E)$ 是一有向图。如果 G 中

(1)任意两结点间都是相互可达的，则称图 G 是强连通的(strongly connected);

(2)任意两结点间至少有一结点可达另一结点，则称图 G 是单向连通的(single directed connected);

例7.

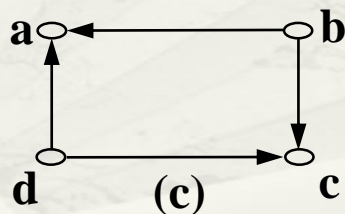
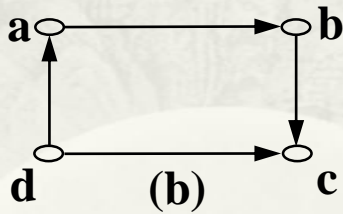
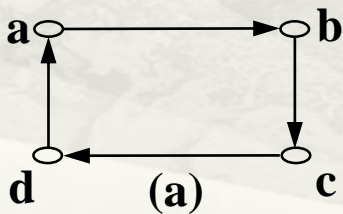


图7

.....

(3)略去边的方向后，任意两结点间都是可达的(即图G的无向图是连通的)，则称图G是弱连通的(weakly connected)。

注：●强连通 \Rightarrow 单向连通；反之？单向连通 \Rightarrow 弱连通；反之？

定义8.强分图 单向分图 弱分图

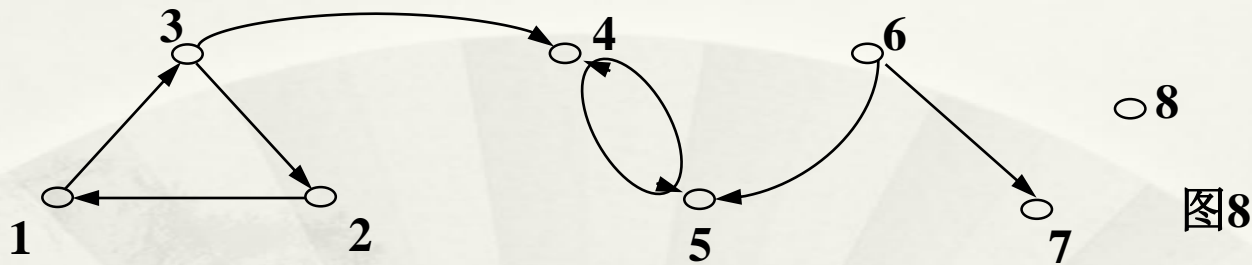
设 $G=(V,E)$ 是一简单有向图。

(1)称 G 的极大的强连通子图为 G 的强连通支(强分图(strongly fragments));

(2)称 G 的极大的单向连通子图为 G 的一个单向连通支(单向分图(single directed fragments));

(3)称 G 的一个极大的弱连通子图为 G 的一个弱连通支(弱分图(weakly fragments))。

例8.求图8的强连通支、单向连通支、弱连通支。



注：●有向图中的强连通性建立了图 G 的结点集 V 上的一个等价关系，因而诱导出了图 G 的结点集 V 上的一个划分，图 G 的每一个强连通支就是一个“划分块”；但是却不能在图 G 的边集 E 上建立一个划分；如在例8的图中，边 $(3, 4)$ 和 $(6, 5)$ 不属于 G 中任何一个强分图。

●有向图中的弱连通性在图 G 的结点集 V 上以及边集 E 上都建立了一个等价关系，因而在图 G 的结点集 V 上以及边集 E 上都诱导出了一个划分，图 G 的每一个弱连通支就是一个“划分块”；

•有向图中的单向连通性，由于单向可达关系不具有对称性，所以这种关系并不能在图 G 的结点集 V 上及边集 E 上建立一个等价关系，因而也不能在图 G 的结点集 V 上及边集 E 上诱导出一个划分，图 G 的每一个单向连通支也就不会是(由某种等价关系所确定的)一个“划分块”。

定理2.在简单有向图中，

- (1)每一个结点及每一条边都恰在一弱连通支中；
- (2)每一个结点都恰在一个强连通支中；
- (3)每一个结点、每一条边都至少属于一个单向连通支。

例9.图的强连通性概念在检测死锁问题上的应用。

计算机操作系统在同一时间内要处理许多程序请求(索取)资源(如CPU、内存、外存、输入输出设备、编译程序等等)的信息(命令)。但这些程序在占有有一些资源的同时，又要请求另外一些资源。在同一时间，既不释放占有的资源，又不放弃资源的请求。在程序动态执行的过程中，有时就可能会产生冲突。

如程序 P_1 已占有资源 R_1 ，又要请求资源 R_2 ；而程序 P_2 已占有资源 R_2 ，又要请求资源 R_1 ，这时两个程序都无法进行工作。称计算机的这种状况为“死锁”状态。计算机操作系统中有专门的进程来负责动态的检测和处理死锁状态。

资源分配图G是一个有向图，它表达了时刻t系统中申请资源的分配状态。在图G中结点表示系统的资源，边表示程序占有资源和请求资源的情况。当程序 P_k 占有资源 R_i 而要申请资源 R_j 时，则从结点 R_i 到结点 R_j 连一条有向边，并标记上 P_k 。

现设时刻t运行的程序集合为 $\{P_1, P_2, P_3, P_4\}$ ，时刻t资源集合为 $\{R_1, R_2, R_3, R_4\}$ 。

时刻t系统中资源的分配状况如右表1所示。

程序	占有资源	请求资源
P_1	R_4	R_1
P_2	R_1	R_2, R_3
P_3	R_2	R_3
P_4	R_3	R_1, R_4

表1

其资源分配图是有向图G，如图9所示。

产生死锁的特征：

时刻t系统产生死锁

⇔资源分配图G中含有
强连通支(有向圈)

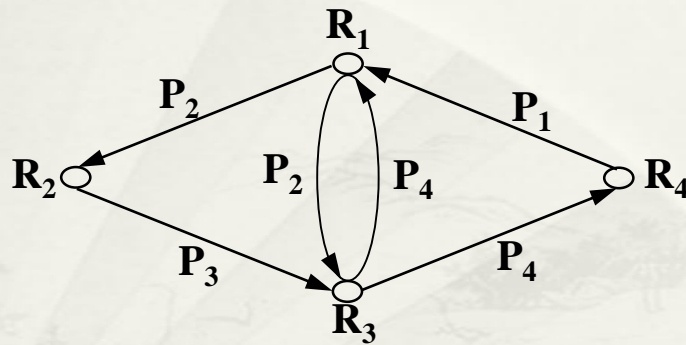


图9

例10.图的强连通性概念在检测递归调用问题上的应用。

程序设计中，程序设计人员一个很重要的任务是需要弄清一个程序中的调用关系是否具有递归性。但在多个过程中，其递归性往往不是很清楚的。

过程调用图G是一个有向图，它表达了程序中过程间的调用状态。在图G中结点表示程序的过程，边表示过程间的调用情况。当过程 P_i 调用过程 P_j 时，则从结点 P_i 到结点 P_j 连一条有向边。

现设程序P中的过程集合为 $P=\{P_1, P_2, P_3, P_4, P_5\}$ 。

程序P中过程间的调用关系为

$C=\{(P_1, P_2), (P_2, P_4), (P_3, P_1), (P_4, P_5), (P_5, P_2)\}$ 。

其过程调用图G是有向图G,

如图10所示。

含有递归的特征:

程序中含有过程递归调用 \Leftrightarrow

过程调用图G中含有强连通支(有向圈)。

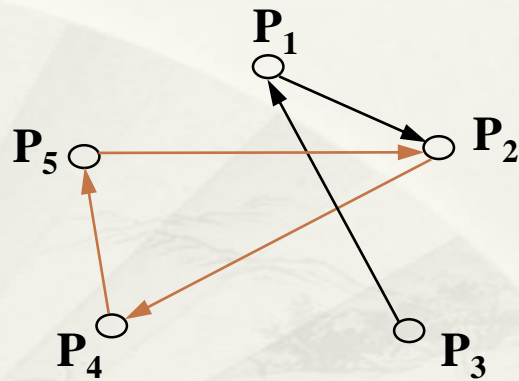


图10

§ 4.图的矩阵表示(*)

- 邻接矩阵
- 关联矩阵
- 可达矩阵
- Warshall*算法
- 可达矩阵与图的连通性

§ 4.图的矩阵表示

定义1.邻接矩阵(adjacency matrix 或 connection matrix)

设 $G=(V,E)$ 是一简单图(有向或无向), $|V|=n$, 并且设 $V=\{v_1,v_2,\dots,v_n\}$ 已被强行命名。则 定义 n 阶方阵

$A=(a_{ij})_{n \times n}$, 其中

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

为图 G 的邻接矩阵。

例1.有向图G的图示如图1所示。

$V = \{v_1, v_2, v_3, v_4\}$ 已强行命名。

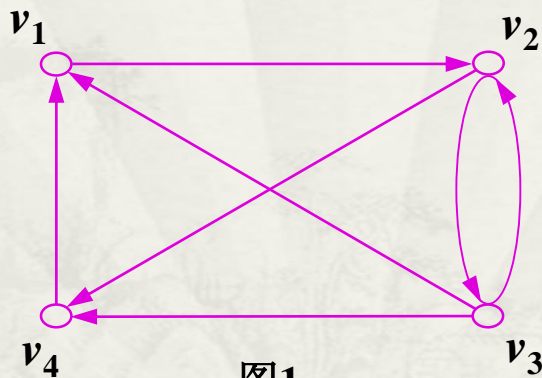
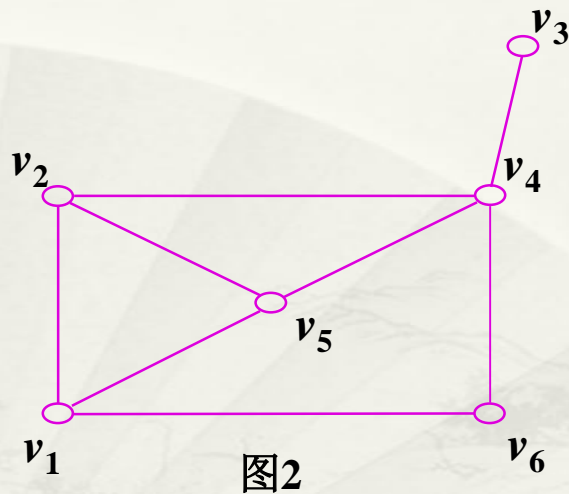


图1

例2.无向图G的图示如图2所示。

$V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ 已强行命名。



注：图G是无向图 \Leftrightarrow 其邻接矩阵是对称矩阵。

结论：

(1)每个简单(有向或无向)图都与一个主角线元素全为零的 $0\sim 1$ 方阵对应；每个主角线元素全为零的 $0\sim 1$ 方阵都与一个简单图对应；

(2)对于一个 (n,m) -图来说，由于 n 个结点有 $n!$ 种强行命名法，故应有 $n!$ 个邻接矩阵。

从《线性代数》的知识可知：矩阵的行与列顺序对调，是对矩阵进行了所谓的基本初等变换。而对矩阵施行基本初等变换，不会改变矩阵的本质性质——即这 $n!$ 个主角线元素全为零的 $0\sim 1$ 方阵都表示同一个图；

(3)两个简单图同构 \Leftrightarrow 其邻接矩阵经过行与列的顺序对调后，
相同；

(4)一图G是零图 \Leftrightarrow 其邻接矩阵A是全0矩阵；

(5)图G是完全图 \Leftrightarrow 其邻接矩阵A除主对角线外均是1；

(6)在有向图G的邻接矩阵A中

其行和等于对应结点的出度。即 $\sum_{k=1}^n a_{ik} = \overrightarrow{dev}(v_i)$

其列和等于对应结点的进度。即 $\sum_{k=1}^n a_{ki} = \overleftarrow{dev}(v_i)$

同一编号的行和列的和相加等于对应结点的度。即

$$\sum_{k=1}^n a_{ik} + \sum_{k=1}^n a_{ki} = \sum_{k=1}^n (a_{ik} + a_{ki}) = dev(v_i)$$

全部1的个数等于边数。即 $\sum_{j=1}^n \sum_{i=1}^n a_{ij} = m。$

(7) 在无向图G的邻接矩阵A中

其行和等于对应结点的度。即 $\sum_{k=1}^n a_{ik} = \text{dev}(v_i)$

其列和等于对应结点的度。即 $\sum_{k=1}^n a_{ik} = \text{dev}(v_i)$

全部1的个数等于边数的2倍。即 $\sum_{j=1}^n \sum_{i=1}^n a_{ij} = 2m$

定义2.关联矩阵(incidence matrix)

设 $G=(V,E)$ 是一简单图(有向或无向), $|V|=n$, $|E|=m$, 并且设 $V=\{v_1,v_2,\dots,v_n\}$, $E=\{e_1,e_2,\dots,e_m\}$ 已被强行命名。则 定义 $n \times m$ 阶矩阵 $B=(b_{ij})_{n \times m}$ 为图 G 的关联矩阵。其中: (1)若 G 是有向图, 则

$$b_{ij} = \begin{cases} 1, & \text{如果结点 } v_i \text{ 是边 } e_j \text{ 的起点} \\ 0, & \text{如果结点 } v_i \text{ 与边 } e_j \text{ 不关联} \\ -1, & \text{如果结点 } v_i \text{ 是边 } e_j \text{ 的终点} \end{cases}$$

(2)若 G 是无向图, 则

$$b_{ij} = \begin{cases} 1, & \text{如果结点 } v_i \text{ 与边 } e_j \text{ 关联} \\ 0, & \text{如果结点 } v_i \text{ 与边 } e_j \text{ 不关联} \end{cases}$$

例3.有向图G的图示如图3。

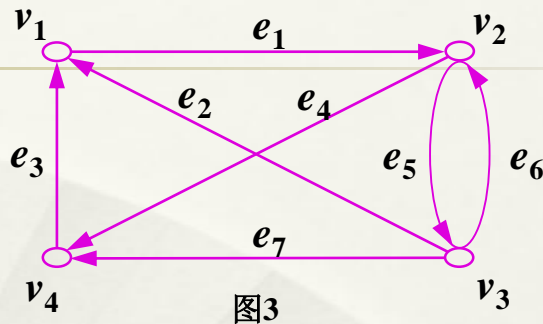
$$V = \{v_1, v_2, v_3, v_4\},$$

$$E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$$

已强行命名。

则图G的关联矩阵为

$$B = \begin{matrix} & \begin{matrix} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \end{matrix} \\ \begin{matrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{matrix} & \begin{pmatrix} 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 1 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 & 1 & 1 \\ 0 & 0 & 1 & -1 & 0 & 0 & -1 \end{pmatrix} \end{matrix}$$



注：有向图的关联矩阵的特点是每列一个正1，一个负1。这正好对应相应边的起点和终点。

例4.无向图G的图示如图4所示。

$V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ 已强行命名。

则图G的邻接矩阵为

$$B = \begin{matrix} & \begin{matrix} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{matrix} \\ \begin{matrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{matrix} & \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{matrix}$$

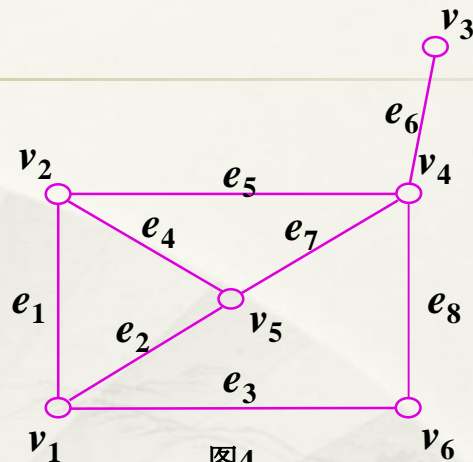


图4

注：•无向图的关联矩阵的特点是每列两个1。这正好对应相应边的两个端点。

定义3.可达矩阵(reachable matrix)

设 $G=(V,E)$ 是一简单图(有向或无向), $|V|=n$, 并且设 $V=\{v_1,v_2,\dots,v_n\}$ 已被强行命名。则定义 n 阶方阵

$R=(r_{ij})_{n \times n}$, 其中

$$r_{ij} = \begin{cases} 1, & \text{若从结点 } v_i \text{ 可达结点 } v_j \\ 0, & \text{若从结点 } v_i \text{ 不可达结点 } v_j \end{cases}$$

为图 G 的可达矩阵。

注：(1)对于有向图

从结点 v_i 可达结点 $v_j \Leftrightarrow$ 从结点 v_i 到结点 v_j 至少有一条有向路

从结点 v_i 不可达结点 $v_j \Leftrightarrow$ 从结点 v_i 到结点 v_j 没有有向路

(2)对于无向图

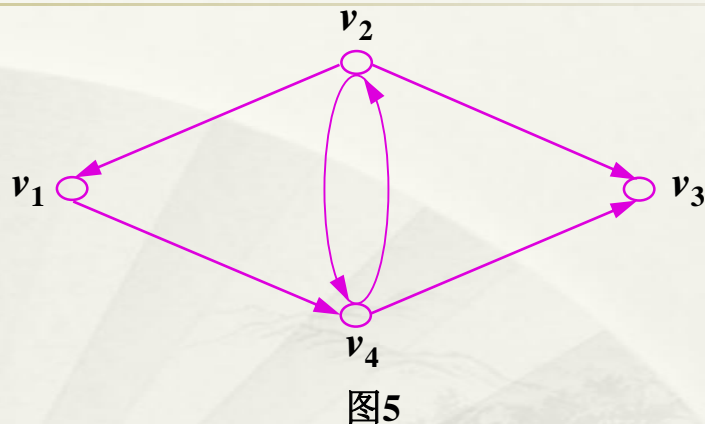
从结点 v_i 可达结点 $v_j \Leftrightarrow$ 从结点 v_i 到结点 v_j 至少有一条路

从结点 v_i 不可达结点 $v_j \Leftrightarrow$ 从结点 v_i 到结点 v_j 没有路。

例5.有向图G的图示如图5。 $V = \{v_1, v_2, v_3, v_4\}$ 已强行命名。

则图G的可达矩阵为

$$R = \begin{matrix} & \begin{matrix} v_1 & v_2 & v_3 & v_4 \end{matrix} \\ \begin{matrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{matrix} & \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} \end{matrix}$$



可达矩阵的计算

直接从图的图示求其可达矩阵，一般情况下是一件很困难的事，可达矩阵的计算是要从图的邻接矩阵求出其可达矩阵。

定义4.邻接矩阵的布尔幂

设 $G=(V,E)$ 是一简单图(有向或无向), n 阶方阵 A 是图 G 的邻接矩阵。则递归的定义邻接矩阵 A 的布尔幂

$$(1) A^{(1)}=A;$$

$$(2) A^{(m+1)}= A^{(m)} \wedge A;$$

其中: $a^{(m+1)} = \bigvee_{k=1}^n (a_{ik}^{(m)} \wedge a_{kj}) \quad (1 \leq i \leq n, 1 \leq j \leq n)。$

注: ●这里 \wedge 和 \vee 都是二元布尔运算: 布尔乘和布尔和。

●其运算表如下:

\wedge	0	1
0	0	0
1	0	1

表1

\vee	0	1
0	0	1
1	1	1

*可达矩阵的计算方法一：(矩阵幂算法)

$$\text{可达矩阵 } R = A \vee A^{(2)} \vee A^{(3)} \vee \dots \vee A^{(n)}$$

注：•这里 \vee 是矩阵的布尔和运算。

•两个矩阵的布尔和运算就是其对应元素做二元布尔和运算。

例6.用矩阵幂算法求图5所示的有向图G的可达矩阵。

图G的邻接矩阵及其矩阵幂为

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

$$A^{(2)} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix}$$

$$A^{(3)} = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix}$$

$$A^{(4)} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

有向图G的可达矩阵

$$R = A \vee A^{(2)} \vee A^{(3)} \vee A^{(4)} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} \circ$$

*矩阵幂算法的计算工作量：

从表2可得矩阵幂算法的(时间)复杂性是 n^4 级的；(空间)复杂性是 n^3 级的。

	布尔乘(次)	布尔和(次)	存贮量(单元)
算一个矩阵 $A^{(m)}$	$n^2 \cdot n = n^3$	$n^2 \cdot (n-1)$	n^2
算可达矩阵 R	$(n-1) \cdot n^3$	$(n-1) \cdot n^2 \cdot (n-1) + n^2 \cdot (n-1) = n^3 \cdot (n-1)$	$n \cdot n^2 + n^2 = (n+1) \cdot n^2$

表2

*可达矩阵的计算方法二：Warshall算法(1962年)

No1.R:=A

No2.k:=1

No3.i:=1

No4.for j:=1 step 1 until n do

$r_{ij} := r_{ij} \vee (r_{ik} \wedge r_{kj})$

No5.i:=i+1;if i≤n then goto No4

No6.k:=k+1;if k≤n then goto No3

No7.if k>n then exit

注： •这里，在第k步,如果 记 r_{ij} 为 $r_{ij}^{(k)}$ ， R 为 $R^{(k)}$ ，则有 $r_{ij}^{(k)} := r_{ij}^{(k-1)} \vee (r_{ik}^{(k-1)} \wedge r_{kj}^{(k-1)})$ ；

•于是有 $r_{ij}^{(k)}=1 \Leftrightarrow$ 从结点 v_i 到结点 v_j 有直接边或者有通过结点 v_1, v_2, \dots, v_k 的路，而这点正是Warshall算法正确性的要点；

• Warshall算法实际上是一个无层次的、前步结果即用、一步输出结果的迭代过程。这里的分层是人为的、勉强的、不自然的。

例7.用Warshall算法求图5所示的有向图G的可达矩阵。

图G的邻接矩阵及各层可达矩阵幂为.....

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix} = \mathbf{R}^{(1)}$$

$$\mathbf{R}^{(3)} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\mathbf{R}^{(2)} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\mathbf{R}^{(4)} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} = \mathbf{R}$$

Warshall算法的计算工作量:

从表3可得Warshall算法的(时间)复杂性是 n^3 级的; (空间)复杂性是 n^2 级的。比矩阵幂算法降低约一个幂级!

	布尔乘(次)	布尔和(次)	存贮量(单元)
内层循环	n	n	n
中层循环	n^2	n^2	n^2
外层循环	n^3	n^3	n^2

表 3

可达矩阵与图的连通性

- (1) 有向图 G 强连通 \Leftrightarrow 可达矩阵 R 是全1矩阵;
- (2) 有向图 G 单向连通 $\Leftrightarrow R \vee R^T$ 除主对角线外均是1;
- (3) 有向图 G 弱连通 \Leftrightarrow 由矩阵 $A_1 = A \vee A^T$ 所确定的可达矩阵 R 是全1矩阵;
- (4) 有向图 G 中有有向圈 \Leftrightarrow 可达矩阵 R 的某些主对角线元素 $r_{ii} = 1$;

§ 5.带权图的最短路径

- 定义
- *Dijkstra*算法
- 算法思想及实质
- 计算实例

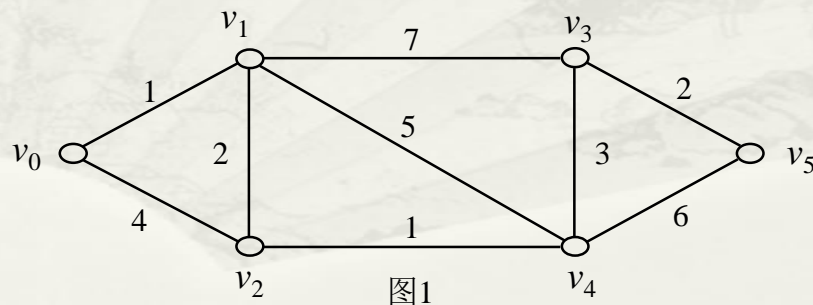
§ 5. 带权图的最短路径

定义1.带权图 (weighted digraph)

设 $G=(V,E)$ 是一简单图。称一元函数 $w:E\rightarrow R^+$ 为图 G 的权函数，使得对于每一条边 $e\in E$ ，均有一正实数 $w(e)\in R^+$ 与之对应，称图 G 为带有权 w 的图，简称为带权图。并记为 $G=(V,E, w)$ 。

例1.图1所示的图 G 是一带权图。

注：带权图有时也称为
网络(network)。



***带权图研究的一个重要方向：寻求带权图的某类极值子图。分为两个方面：**

(1)寻找某类具有最大权的子图。其中之一、且最常见的就是最长路问题。例如，在一给定的军用或民用工程网络中的寻求最大施工时间问题。解决此类问题的著名算法是关键路径法(CPM(Critical Path Method)(1957))，它是由美国著名的杜邦公司和兰德公司提出的。此算法和由美国海军军械部特种计划研究室在研究“北极星导弹潜艇”时提出的著名的计划评审技术(PERT(Program Evaluation and Review Technique)(1958))一起，利用电子计算机，可完成对大型工程项目的施工时间分析。

(2)寻找某类具有最小权的子图。其中之一、且比较常见的就是最短路问题。例如，在一给定的交通网络(铁路网络、公路网络等)或通讯网络中寻求两点间最短路问题。解决此类问题的算法之一是最著名的*Dijkstra*算法(1959)。这是迄今为止公认的求最短路长的最好算法。

定义2.最短路 (shortest path)

设 $G=(V,E,w)$ 是一带权图。

(1) 设 $P=(e_{i1}, e_{i2}, \dots, e_{ik})$ 是 G 中的一条路。则路 P 的路长定义为：

$$w(P) = \sum_{j=1}^k w(e_{ij});$$

(2) 从结点 u 到结点 v 的最短路 P_0 是满足下述条件的路：

$$w(P_0) = \min\{w(P) \mid P \text{ 为从 } u \text{ 到 } v \text{ 的路}\}。$$

注：当带权图中各边的权值都为1时，则路长的定义就退化为 § 3 一般图的路长定义了。

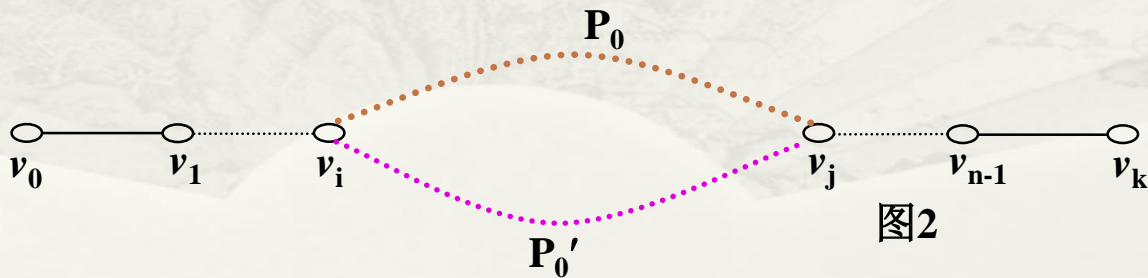
E.W.Dijkstra算法(1959):

算法的目标是求出带权图G中从结点 v_0 到结点 v_n 的最短路。

*基本思想：最短路上的任何一段(局部)也都是最短的。即，如果 $P_0=(v_0, v_1, v_2, \dots, v_n)$ 是一条最短路，则其中任一段 $(v_i, v_{i+1}, v_{i+2}, \dots, v_j)$ ($0 \leq i < j \leq n$)也是最短的。

特别地，最短路 P_0 的任一真前缀 $(v_0, v_1, v_2, \dots, v_k)$ ($k < n$)也是最短的。

否则，将局部段换为最短的，则可得一比 P_0 更短的路 P_0' ，这与已知 P_0 是最短路矛盾。



No1. $P := \{v_0\}$; $T := V \setminus \{v_0\}$; /*设定P-T界面*/

$d(v_0) := 0$; $(\forall t \in T)(d(t) := \infty)$; /*做圆标号(用一重for循环可实现)*/

$p := v_0$; $\text{mark}(v_0) := d(v_0)$; /*做方标号*/

No2. $(\forall t \in T)(d(t) := \min\{d(t), d(p) + w(p, t)\})$;

 /*按离界面最近方向修改圆标号(用一重for循环可实现)*/

$(\exists t_0 \in T)(\forall t \in T)(d(t_0) \leq d(t))$; /*寻求T中圆标号最小的结点 t_0 (用一重for循环及if比较可实现)*/

No3. $P := P \cup \{t_0\}$; $T := T \setminus \{t_0\}$; /*修改P-T界面*/

$p := t_0$; $\text{mark}(t_0) := d(t_0)$; /*改圆为方*/

No4. if $t_0 = v_n$ then goto exit ;

 else goto No2 ;

*Dijkstra算法的工作原理:

将带权图G的结点集V分成两部分：一部分称为具有P标号(方标号)的集合；另一部分称为具有T标号(圆标号)的集合。所谓结点 v_i 的P标号是指从 v_0 到 v_i 的最短路的路长；而结点 v_i 的T标号是指从 v_0 到 v_i 的某条路径的长度。

Dijkstra算法首先将 v_0 取为 P 标号结点；其余的结点均取为T标号结点；然后按离P-T界面最近方向逐步地将具有T标号的结点改为P标号结点。当结点 v_n 也被改为P标号结点时，也就求出了从(起始)结点 v_0 到各个结点 v_i 的最短路长；尤其是求出了从(起始)结点 v_0 到(终结)结点 v_n 的最短路长。

例2. 利用Dijkstra算法求出例1图1所示的带权图G中从结点 v_0 到结点 v_5 间的最短路及最短路长。

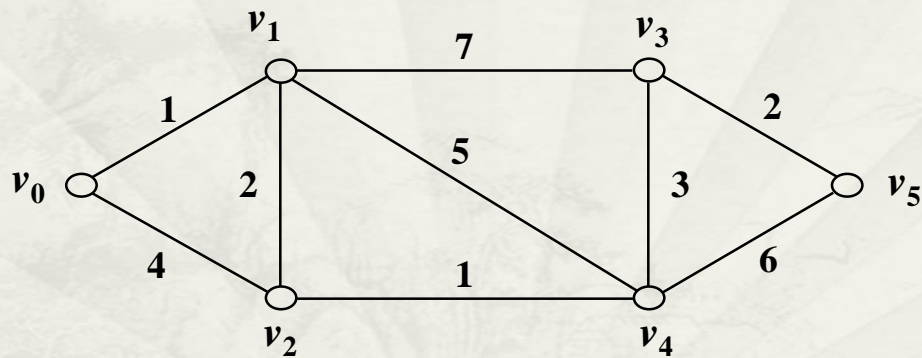
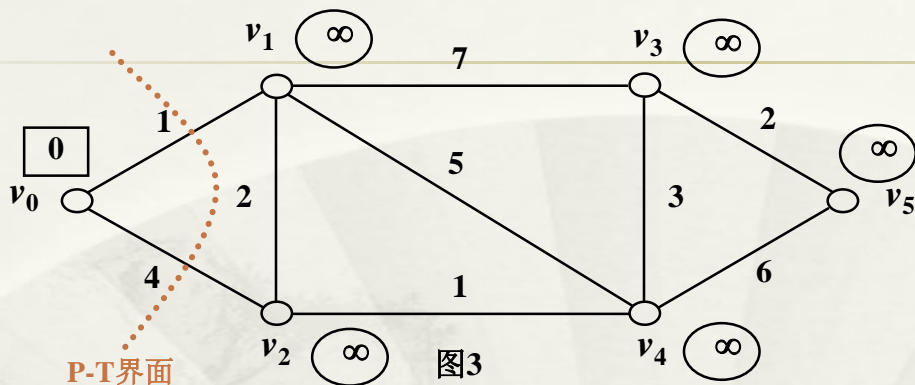
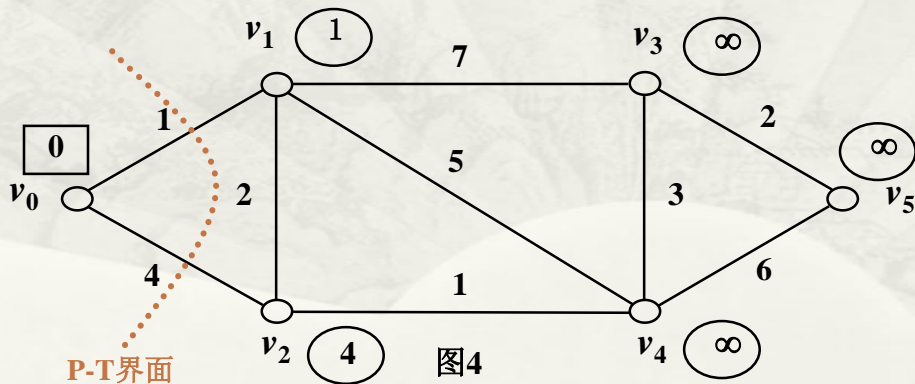


图1

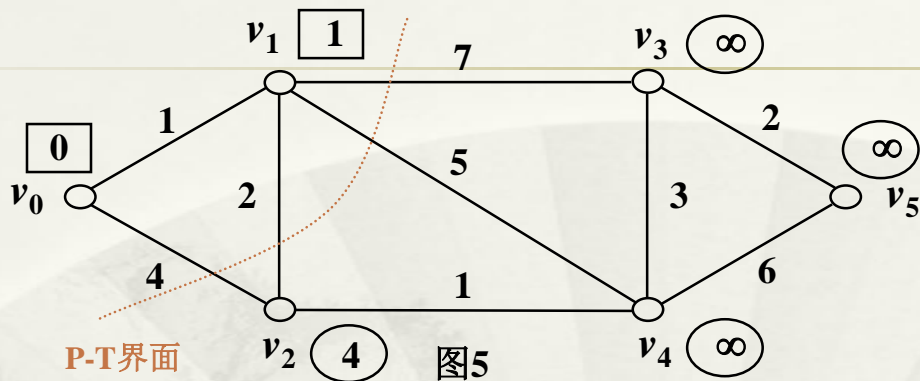
(1)图上作业法:



设定P-T界面:
 结点 v_0 做方标号;
 其余结点做圆标号;



修改圆标号(只修改P-T
 界面近前的结点 v_1, v_2):
 $v_1: \infty, \underline{0+1=1};$
 $v_2: \infty, \underline{0+4=4};$

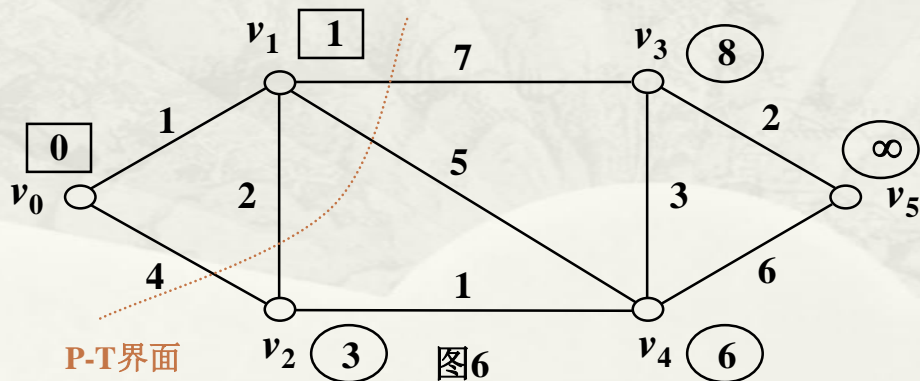


改圆为方:

寻求T中圆标号最小的结点
 $t_0 = v_1$:

v_1	v_2	其余
<u>1</u>	4	∞

修改P-T界面;



修改圆标号(只修改离P-T
 界面最近的结点 v_2 , v_3 ,
 v_4):

$$v_2: \underline{4, 1+2=3};$$

$$v_3: \infty, \underline{1+7=8};$$

$$v_4: \infty, \underline{1+5=6};$$

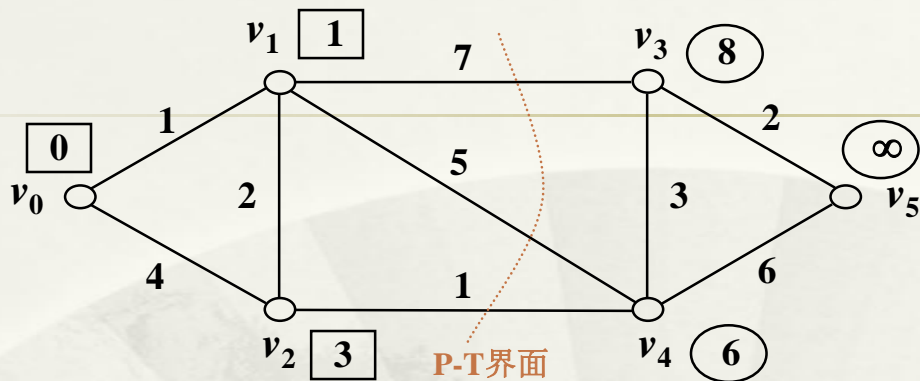


图7

改圆为方:

寻求T中圆标号最小的结点 $t_0 = v_2$:

v_2	v_3	v_4	v_5
<u>3</u>	8	6	∞

修改P-T界面;

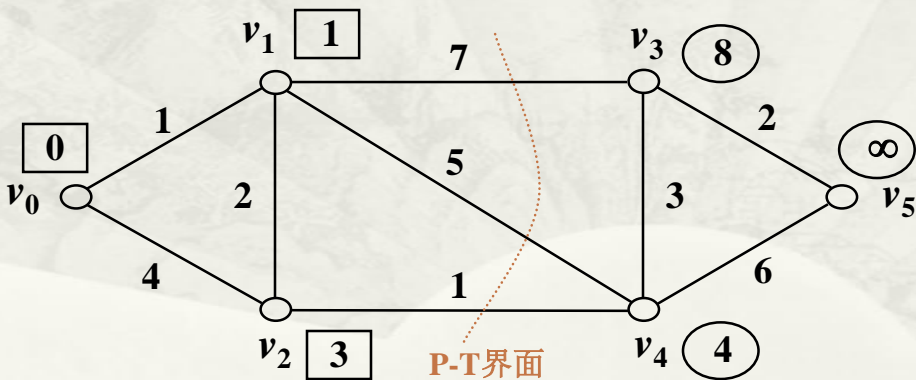
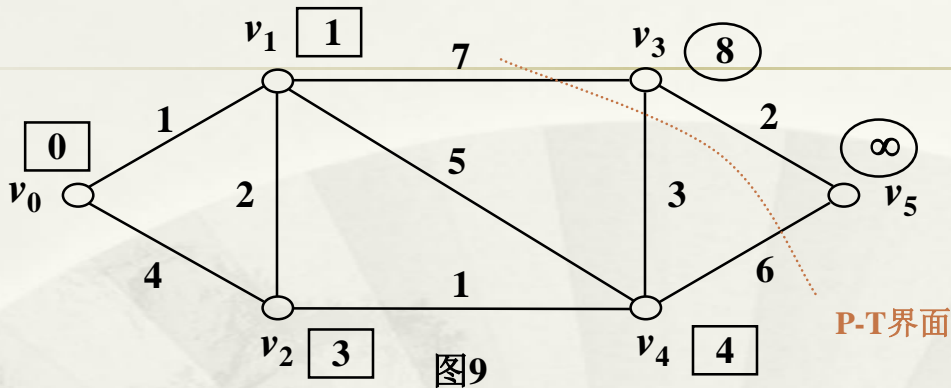


图8

修改圆标号 (只修改离P-T界面最近的结点 v_3, v_4):

v_3 : 8;
 v_4 : 6, 3+1=4;

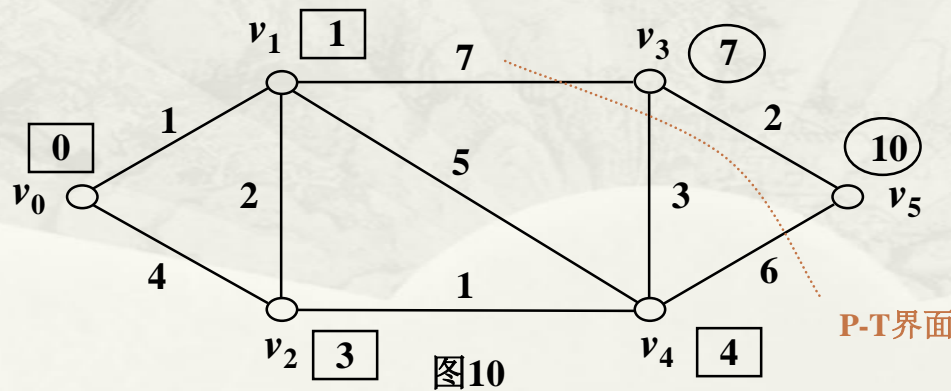


改圆为方:

寻求T中圆标号最小的结点
点 $t_0 = v_4$:

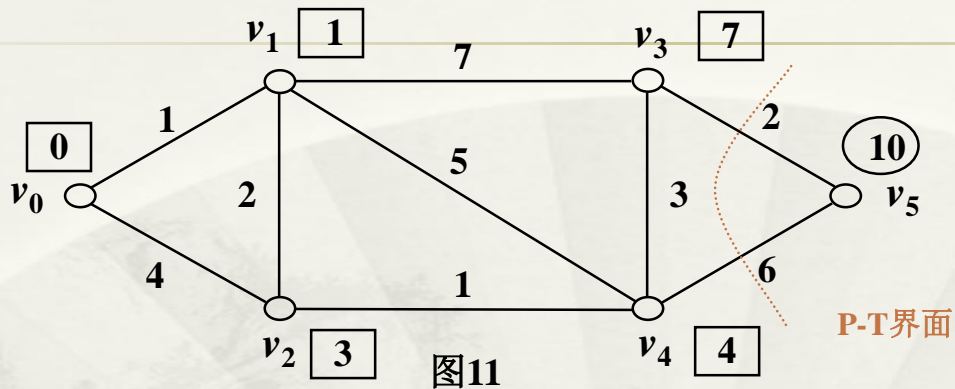
v_3	v_4	v_5
8	<u>4</u>	∞

修改P-T界面;



修改圆标号(只修改离P-T界面最近的结点 v_3, v_5):

v_3 : 8, 4+3=7 ;
 v_5 : $\infty, 4+6=10$;

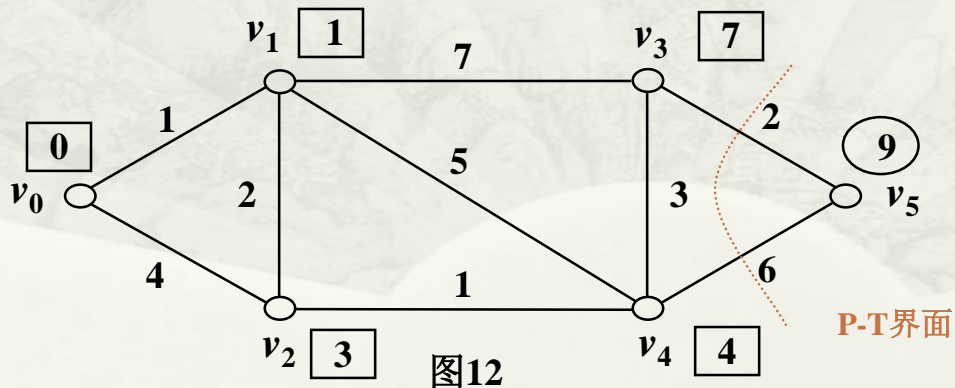


改圆为方:

寻求T中圆标号最小的结点
点 $t_0 = v_3$:

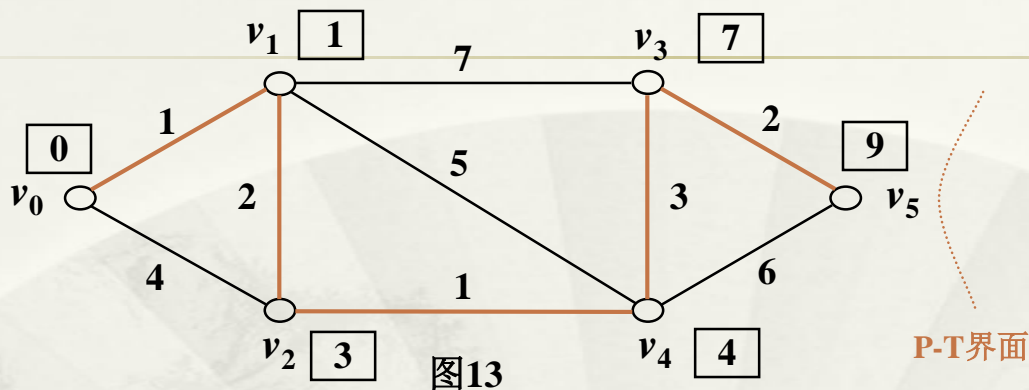
v_3	v_5
<u>7</u>	10

修改P-T界面;



修改圆标号(只修改离P-T界面最近的结点 v_5):

$$v_5: \underline{10, 7+2=9};$$



改圆为方：
 寻求T中圆标号最小的
 的结点 $t_0 = v_5$ ；
 修改P-T界面；

于是，从结点 v_0 到结点 v_5 间的最短路为

$$P_0 = (v_0, v_1, v_2, v_4, v_3, v_5) \quad (\text{图13红线所示})$$

从结点 v_0 到结点 v_5 间的最短路长为

$$w(P_0) = w(v_0, v_1) + w(v_1, v_2) + w(v_2, v_4) + w(v_4, v_3) + w(v_3, v_5) = 1 + 2 + 1 + 3 + 2 = 9$$

(2)程序算法:

1. $P=\{v_0\}$; $T=\{v_1, v_2, v_3, v_4, v_5\}$;

$d(v_0)=0$; $d(v_1)=d(v_2)=d(v_3)=d(v_4)=d(v_5)=\infty$; $p=v_0$;

2. $d(v_1)=1$; $d(v_2)=4$; $d(v_3)=d(v_4)=d(v_5)=\infty$;

$t_0=v_1$;

3. $P=\{v_0, v_1\}$; $T=\{v_2, v_3, v_4, v_5\}$; $p=v_1$;

4. $t_0 \neq v_5$; go to 2;

2. $d(v_2)=3$; $d(v_3)=8$; $d(v_4)=6$; $d(v_5)=\infty$;

$t_0=v_2$;

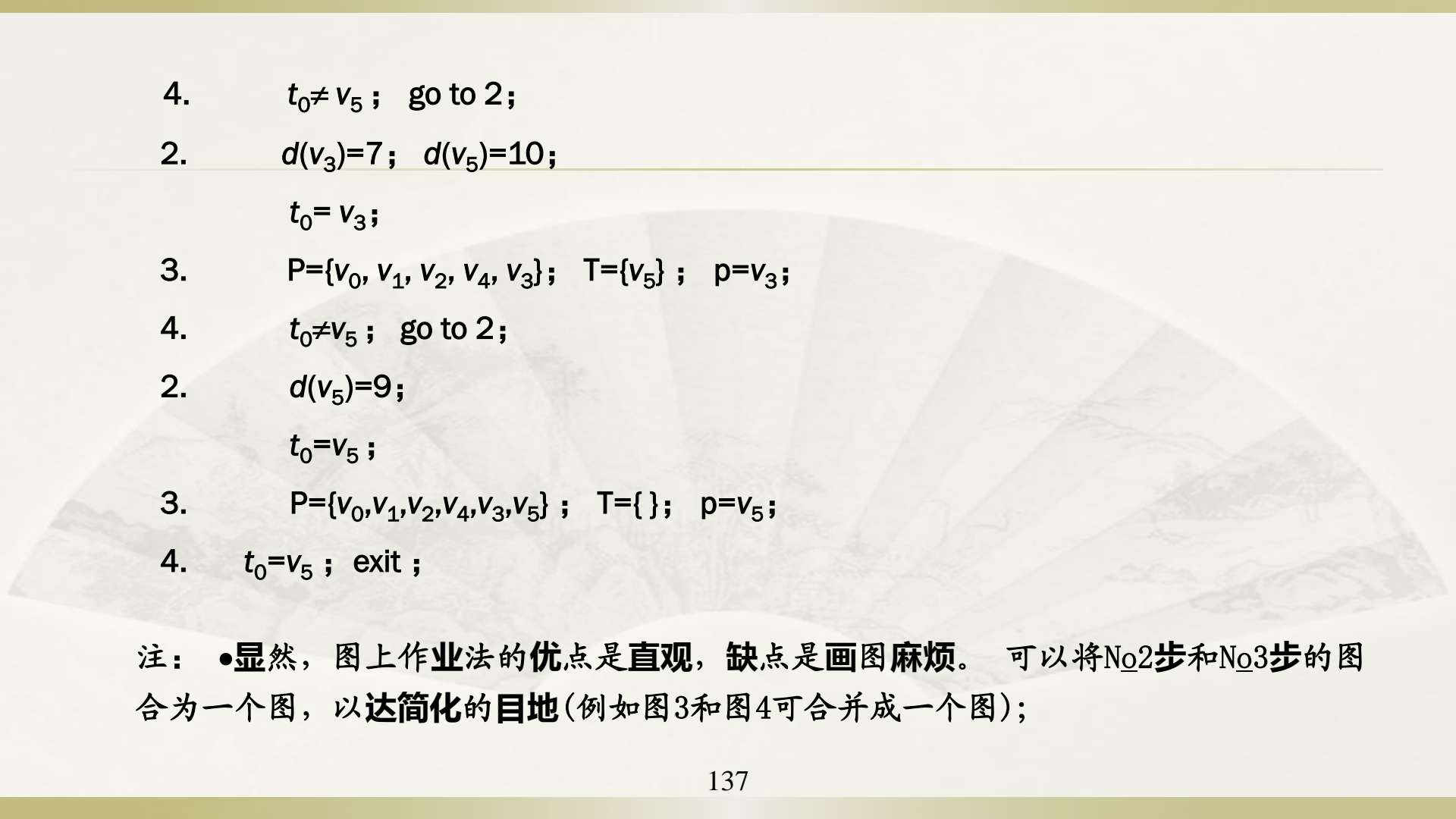
3. $P=\{v_0, v_1, v_2\}$; $T=\{v_3, v_4, v_5\}$; $p=v_2$;

4. $t_0 \neq v_5$; go to 2;

2. $d(v_3)=8$; $d(v_4)=4$; $d(v_5)=\infty$;

$t_0=v_4$;

3. $P=\{v_0, v_1, v_2, v_4\}$; $T=\{v_3, v_5\}$; $p=v_4$; . 136

- 
4. $t_0 \neq v_5$; go to 2;
 2. $d(v_3)=7$; $d(v_5)=10$;
 $t_0 = v_3$;
 3. $P=\{v_0, v_1, v_2, v_4, v_3\}$; $T=\{v_5\}$; $p=v_3$;
 4. $t_0 \neq v_5$; go to 2;
 2. $d(v_5)=9$;
 $t_0 = v_5$;
 3. $P=\{v_0, v_1, v_2, v_4, v_3, v_5\}$; $T=\{\}$; $p=v_5$;
 4. $t_0 = v_5$; exit;

注：●显然，图上作业法的优点是直观，缺点是画图麻烦。 可以将No2步和No3步的图合为一个图，以达简化的目地(例如图3和图4可合并成一个图);

•显然，程序计算法的优点是简便、机械，缺点是缺乏直观性、计算出错不易检查。

**Dijkstra*算法的效果：

•*Dijkstra*算法能够一次性的求出从起点 v_0 到带权图 G 中其余每一个结点的最短路长(就是各结点方标号中的数值)。

•*Dijkstra*算法没有直接给出从起点 v_0 到带权图 G 中其余各结点的最短路。因此，要直接给出从起点 v_0 到带权图 G 中其余各结点的最短路，需要修改*Dijkstra*算法，或者①给其嵌入记忆系统(用表结构或仅记前一个结点)；或者②给其增加回溯功能。

Dijkstra算法的特点:

•Dijkstra算法是沿着P-T界面(按离界面最近方向)向前推进的; 而不是沿着最短路向前推进的。

*Dijkstra算法的计算工作量:

Dijkstra算法的计算工作量主要是集中在No2。在 $|P|=k$, $|T|=n-k(0 \leq k \leq n-1)$ 时, 当P再增加一个结点, 需进入一次循环:

	加法(次)	比较(次)	存贮量(单元)
算一个结点	1	1	1
算一次循环	$n-k-1$	$2(n-k-1)$	1
完成计算	$\frac{1}{2}n(n-1)$	$n(n-1)$	$n^2 + 2n + 1$

表1

从表1可得Dijkstra算法的(时间)复杂性是 n^2 级的; (空间)复杂性也是 n^2 级的。

求从结点 v_1 到结点 v_8 的所有最短路。

