离散数学

西安交通大学
电子与信息工程学院
计算机系

第八章 图论
（Graph Theory）

离散数学

§ 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

离散数学

§1. 图论朔源

- 图论的创始人
- 图论应用的几个例子

离散数学

§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。

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。

问题是这样的：在公元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。当时，城中的居民热衷于这样一个问题，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。问题看来并不复杂，

离散数学

但当地的居民和游人做了不少的尝试，却都没有取得成功。于是，有好事者便向当时居住在该城的大数学家欧拉请教。

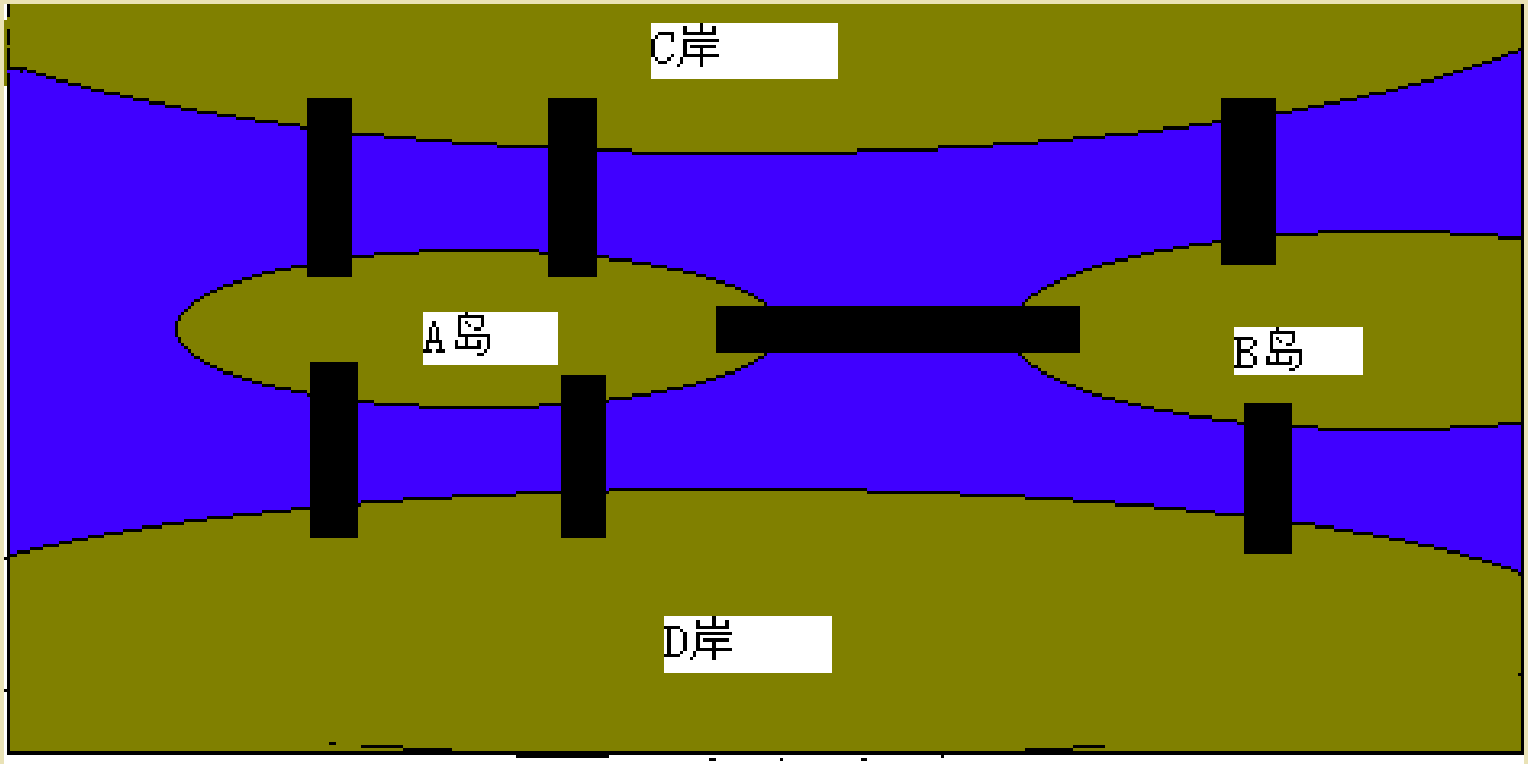


图1.-七桥图

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。

离散数学

1736年，瑞士的数学家L.Euler解决了这个问题。他将四块陆地表示成四个结点，凡陆地间有桥相连的，便在两点间连一条线，这样图1就转化为图2了。

此时，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？后人称如此的问题为Euler环游。

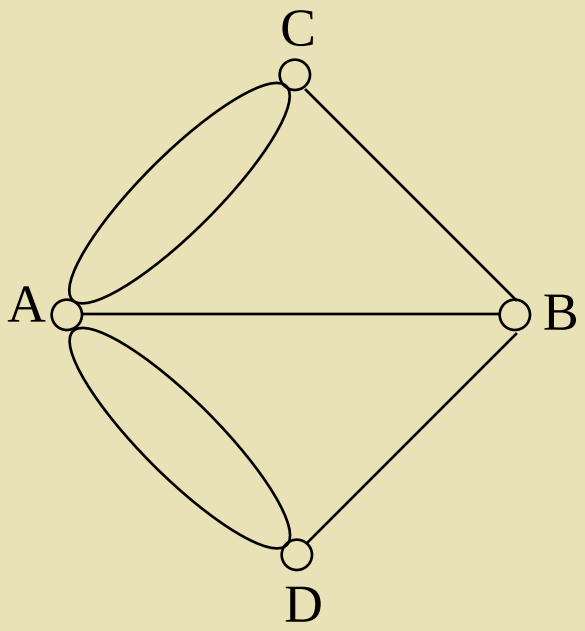


图2

离散数学

欧拉断言这样的回路是不存在的。理由是：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，而不经过已走过的

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。

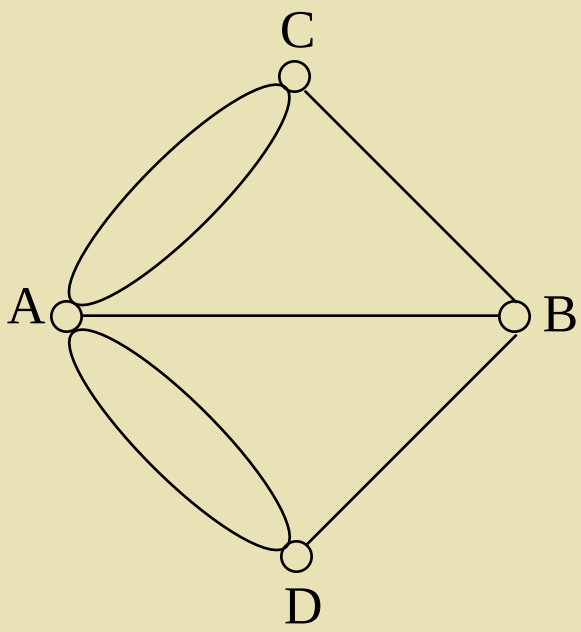


图2

离散数学

由此，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。

有关这方面的内容，我们将在§6. Euler图来详细讨论。

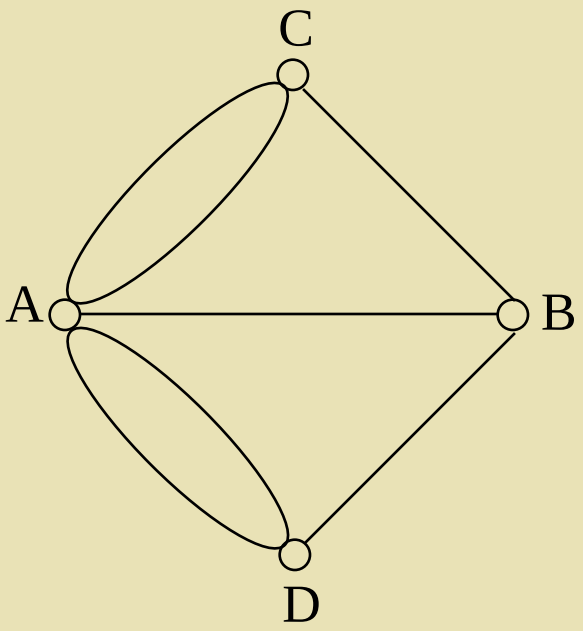


图2

此图实际上是反映了客观事物之间的相互关系

离散数学

本世纪40年代，一个数学游戏也使用类似的方法得到了解决：某人挑一担菜、并带一只狼、一只羊，要从河的北岸到南岸。由于船小，只允许带狼、羊、菜三者中的一种过河；而由于明显的原因，当人不在场时狼与羊、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)、狼(wolf)、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，狼，羊，菜)，(空)；　(P,W,S,C)，(∅)；

(人，狼，羊)，(菜)；　(P,W,S,)，(C)；

(人，狼，菜)，(羊)；　(P,W,C)，(S)；

离散数学

(人，羊，菜)，(狼)；　　(P,S,C)，(W)；

(人，羊)，(狼，菜)；　　(P,S)，(W,C)。

不安全的状态有如下六种：

(人)，(狼，羊，菜)；　　(P)，(W,S,C)；

(人，菜)，(狼，羊)；　　(P,C)，(W,S)；

(人，狼)，(羊，菜)；　　(P,W)，(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，狼，羊，菜)或 (P,W,S,C) 状态到(空) 或(∅)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,W,S,C)到(∅)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。

离散数学

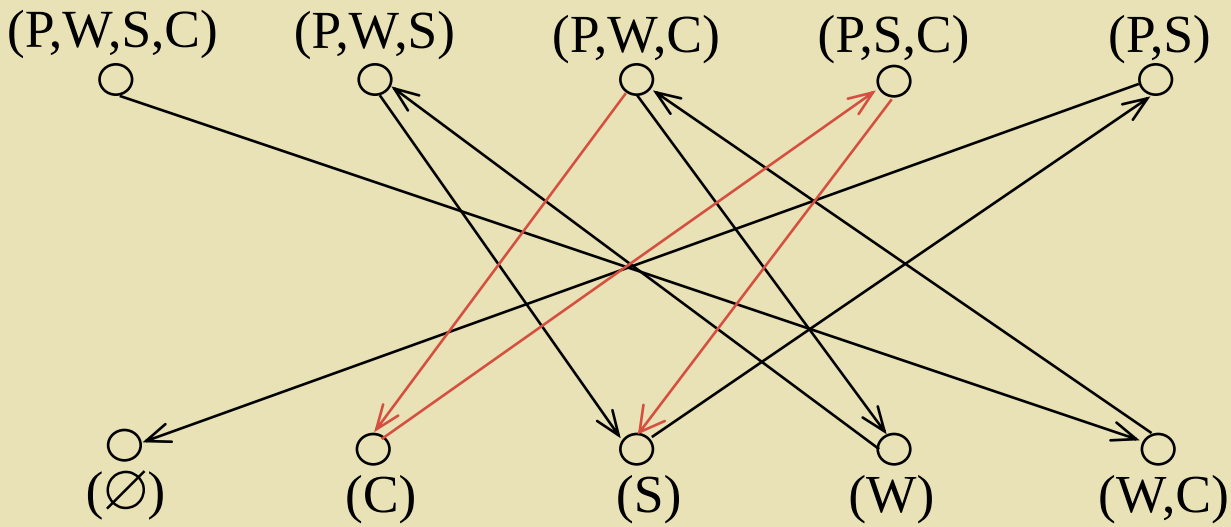


图3

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。我们现在的方法是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)。

离散数学

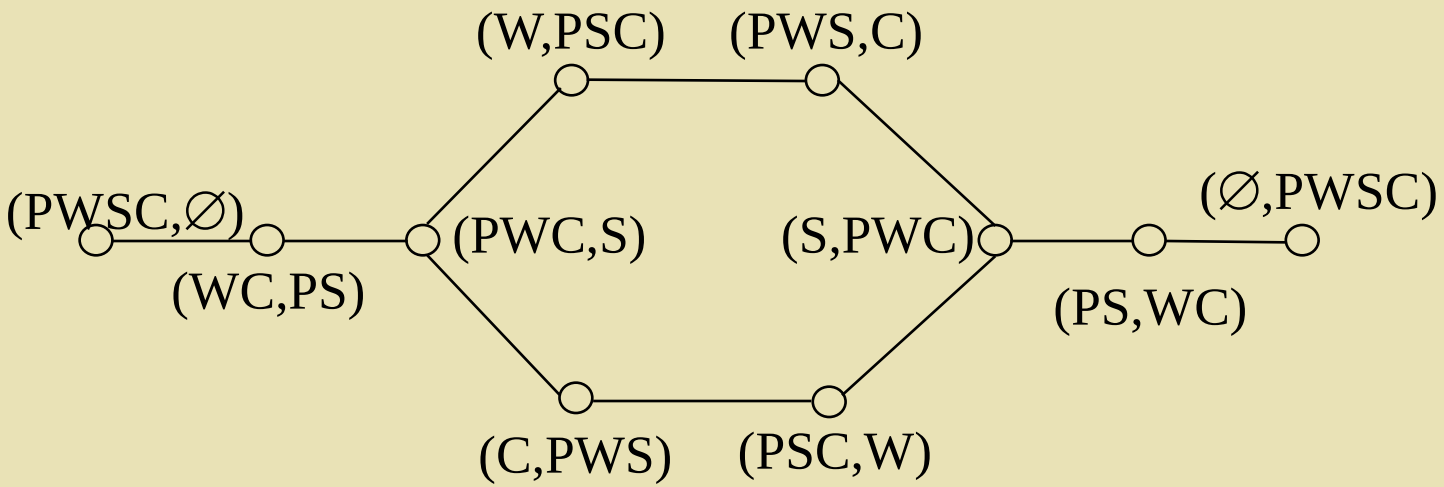


图4

注：•上述问题统称“渡河问题”。渡河问题很多，比如以下；

• “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 刘家佺 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—小船的容量也是可变的。

离散数学

从上面的两上例子可以看到：对有些问题的研究可以归结为对图的研究，图是由点和线构成的。在图中我们感兴趣的事情是给定的两点间是否有连线，至于如何连结则是无关紧要的，这样的处理方式可以使问题变得简洁明了。下节，我们将给出图的正式定义。

§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

离散数学

由上节的例子，对什么是图我们有了初步的了解。我们看到：一个图有两个基本的成分——点与线，并且组成图的点与线之间有一定的联系。如在§1的图2中与点对(A,B)相联系的有一条边，与点对(A,D)相联系的有两条边，而点对(C,D)间却无边相联，作为图的定义，则需将点与边的关系描述清楚。下面，我们给出图的几种不同的定义。

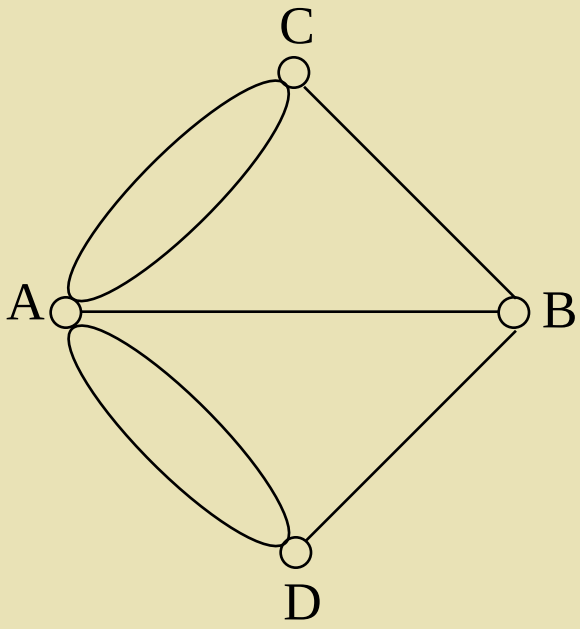


图2

离散数学

定义1.(图的定义一)

图G = (V, E)是一个系统 ， 其中

(1)V≠∅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。

离散数学

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。

上述事实可用图G = (V, E)表示。图G中结点集

$V=\{v_1, v_2, v_3, v_4\}$，图G中边集 $E=\{e_1, e_2, e_3\}$，它的图示如图1所示。

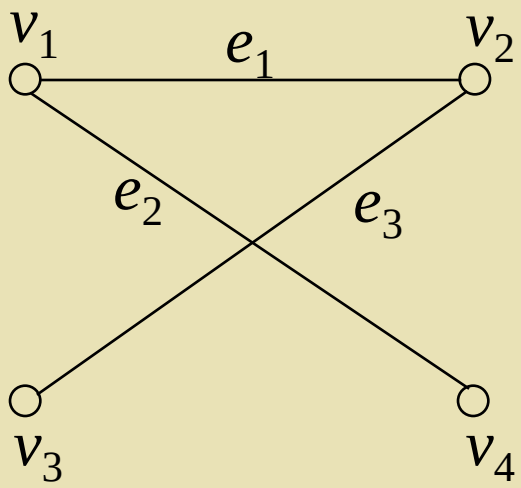


图1

离散数学

定义2. (图的定义二)

图G = (V, E)是一个系统，其中

(1)V ≠∅是一有限集合； V中的每一元素v∈V都称为图G的一个结点(node ,vertex)； V称为图G的结点集；

(2)E ⊆V×V是一有限集合，一个V上的关系； E中的每一元素(u,v)∈E都称为图G的一条边(edge) (这里u, v∈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）。

•根据这个定义，上面的例1可表示为：图G = (V, E)。图G中结点集合V={v_1, v_2, v_3, v_4}，图G中边的集合 E={(v_1, v_2), (v_1, v_4), (v_2, v_3)}，它的图示仍如图1所示。

离散数学

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$V=\{v_1, v_2, v_3, v_4\}$，图中边集$E=\{(v_1, v_2), (v_2, v_3), (v_2, v_4)\}$，它的图示如图2所示。

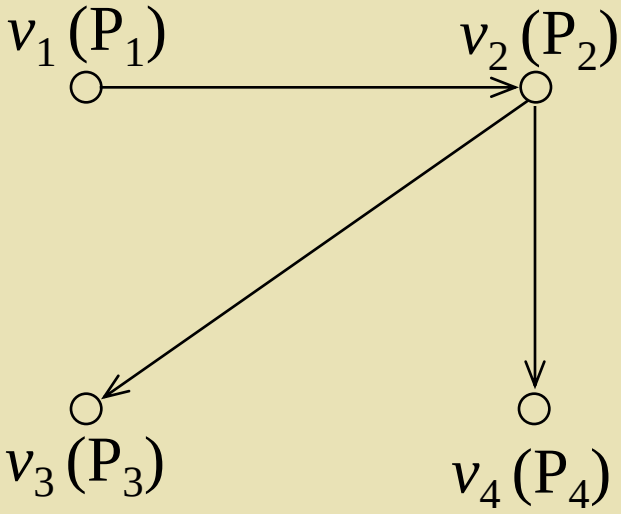


图2

离散数学

定义3. (图的定义三)

图G = (V,Σ, E)是一个系统，其中

(1)V ≠∅是一有限集合； V中的每一元素v∈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σ∈Σ都称为图G中的一个标号(label)； Σ称为图G的标号集；

(3)E ⊆V×Σ×V是一有限集合，一个三元关系； E中的每一元素(u,σ,v)∈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∈V , σ∈Σ)； E称为图G的边集。

离散数学

注：•此定义是由美国哈佛大学爱伦堡教授给出的；

•此定义的优点是规定了严格的点、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。

离散数学

例3. 七桥图(见图3), 按定义3,
可用一图G = (V, Σ, E)来表示。
图中结点集V=$\{v_1, v_2, v_3, v_4\}$,
图中标号集Σ=$\{\sigma_1, \sigma_2\}$,
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。

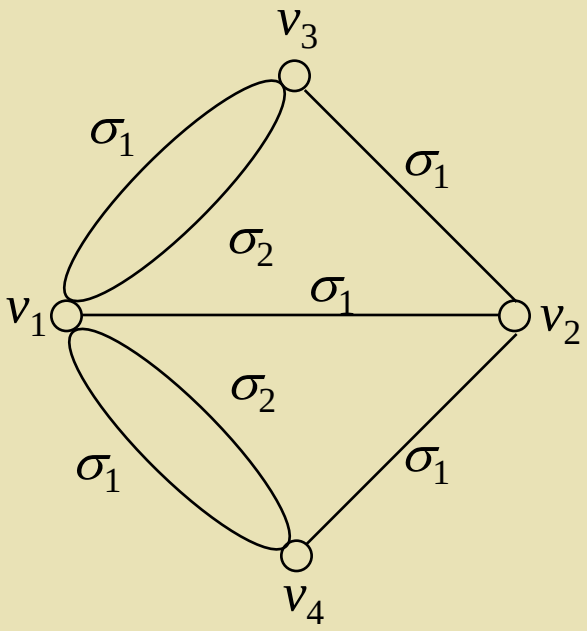


图3

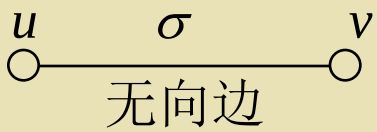
离散数学

- **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简称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。

例如边(u,σ,v)就是一无向边。



(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。

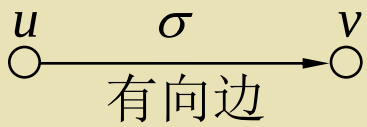
例如图1与七桥图(图3)都是无向图。

离散数学

(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。

例如边(u,σ,v)就是一有向边。



(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。

例如图2是有向图。半序关系的Hasse图是有向图。

(6)混和图： (mixed graph)

既有有向边又有无向边的图称为混和图。

例如下边左图是一混和图。

离散数学

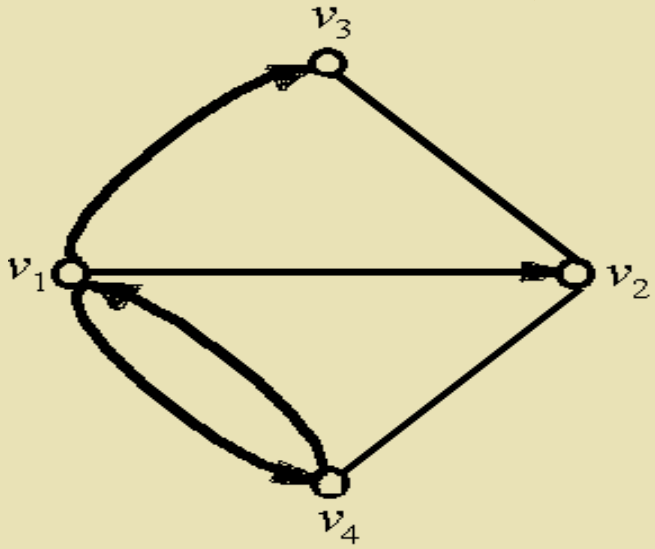


图4.混合图

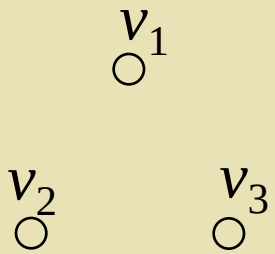


图5.零图

(7)空图：(empty graph)

V=∅(当然 E=∅)，即没有一个结点的图称为空图。

(8)零图：(null graph)

E=∅，即没有一条边的图称为零图。

例如上边右图是三个结点的零图。

离散数学

(9)平凡图： (trivial graph)

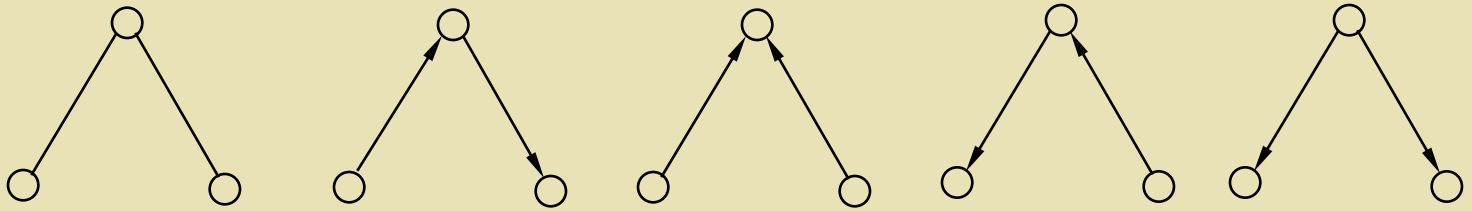
|V| = 1，即只有一个结点的图称为平凡图。

例如下边是三个不同的平凡图。

(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。

下边是五种二边相邻的情况。

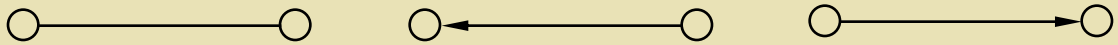


离散数学

(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。

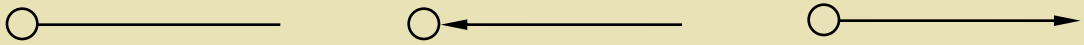
下边是三种二结点相邻的情况。



(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。

下边是三种结点与边相关联的情况。



(13)孤立点： (isolated vertex)

不与任何边相关联的结点称为孤立点。

离散数学

右图是一含有孤立点的无向图，v_4是孤立点。

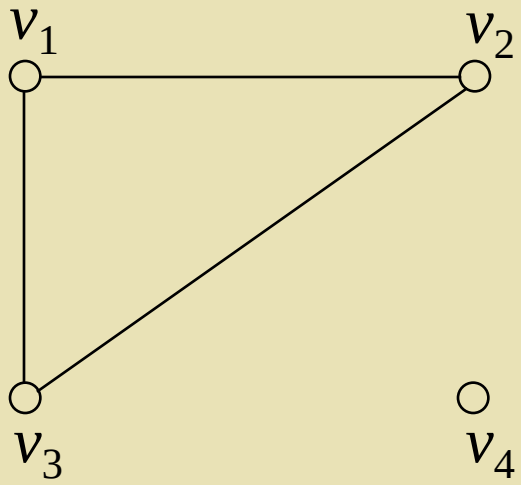


图6.含孤立点的图

(14)自环： (loop)

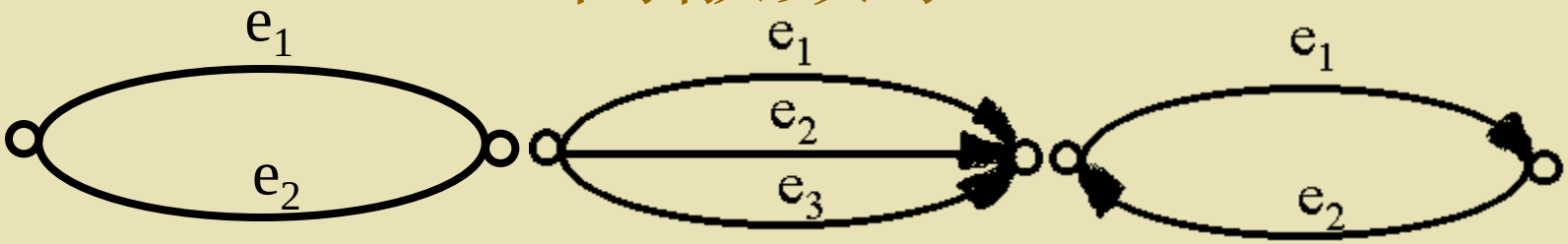
两个端点相同的边称为自环。

例如下边是二个不同的自环。

(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。

例如下边是二种平行边的情况和非平行边的情况。



无向边e_1和e_2是平行边

有向边e_1和e_2及e_3是平行边

有向边e_1和e_2不是平行边

(16)重数： (multiplicity)

两结点间平行边的条数称为平行边的重数。

例如上边左图平行边的重数是2，中图平行边的重数是3，右图平行边的重数是1。

(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。

例如前边七桥图(图3)是多重图，重数是2。

离散数学

(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。

例如前边图1、图2是简单图；七桥图(图3)不是简单图(因其有平行边)；右图不是简单图(因其有自环) 。

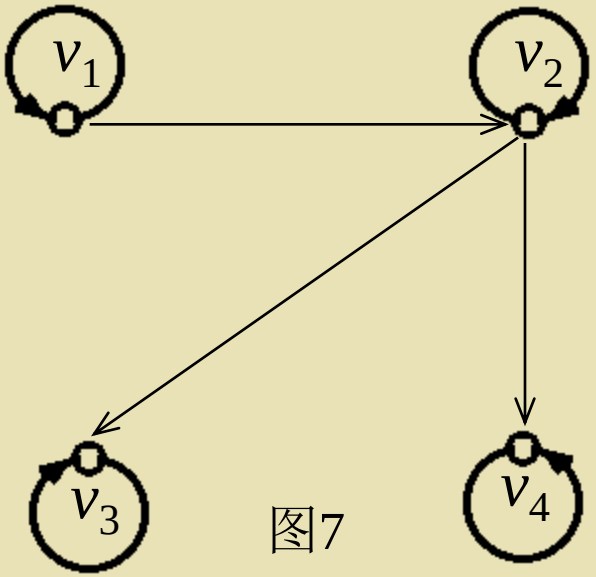


图7

(19)图的阶： (order)

图中结点的个数|V|称为图的阶。

例如前边图1、图2及七桥图(图3)都是4阶图。

(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。

离散数学

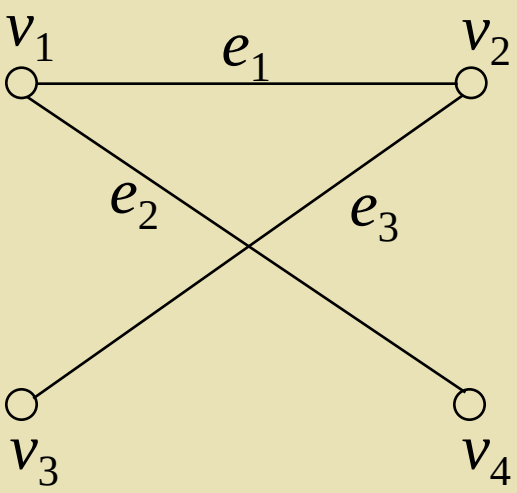

v_1
e_1
v_2
e_2
e_3
v_3
v_4

图1

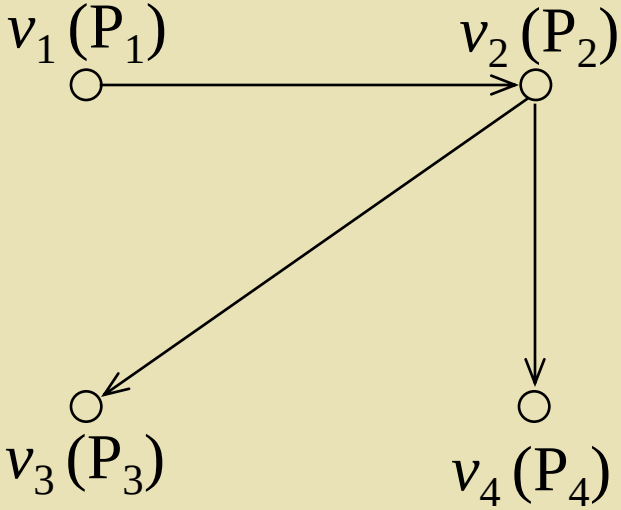

v_1 (P$_1$)
v_2 (P$_2$)
v_3 (P$_3$)
v_4 (P$_4$)

图2

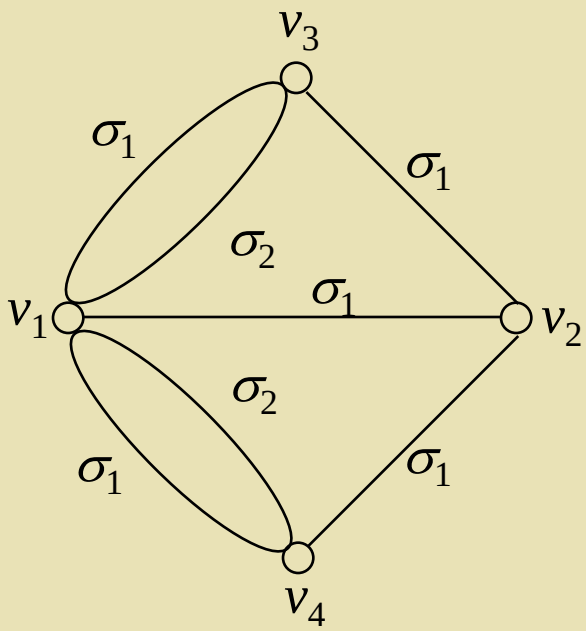

v_3
σ_1
σ_1
σ_2
σ_1
v_1
v_2
σ_2
σ_1
σ_1
v_4

图3

离散数学

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：$\boldsymbol{K}_n$

例如下面左图是5个结点的无向完全图；右图是4个结点的有向完全图。

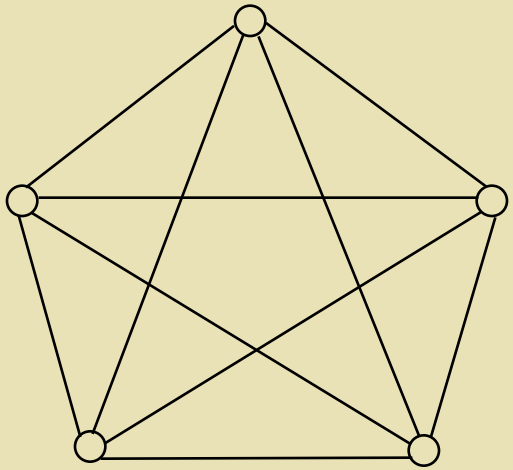


图8. $\boldsymbol{K}_5$

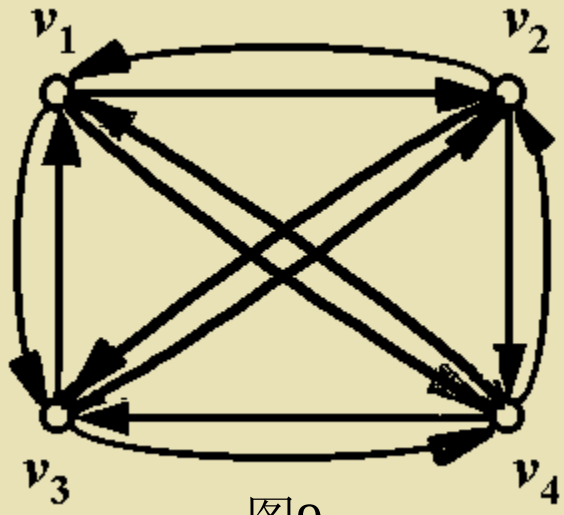


图9

离散数学

定义4. (图的定义四)

图G=(V,E, γ)是一个系统，其中

(1)V ≠∅是一有限集合； V中的每一元素v∈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∈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

γ:E→2^V (无向图) 或 γ:E→ V×V (有向图) 。

一般来说，它将E中的每条边e∈E与结点集V中的一个二元子集{u,v}∈2^V (或{u,v}⊆V)相关联或与结点集V上的一个二元组(u,v)∈V×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。

离散数学

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, γ)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 γ:E→2^V 使

$\gamma(e_1)=\{v_1,v_3\}, \gamma(e_2)=\{v_1,v_3\}$,
$\gamma(e_3)=\{v_1,v_2\}, \gamma(e_4)=\{v_1,v_4\}$,
$\gamma(e_5)=\{v_1,v_4\}, \gamma(e_6)=\{v_2,v_3\}$,
$\gamma(e_7)=\{v_2,v_4\}$。

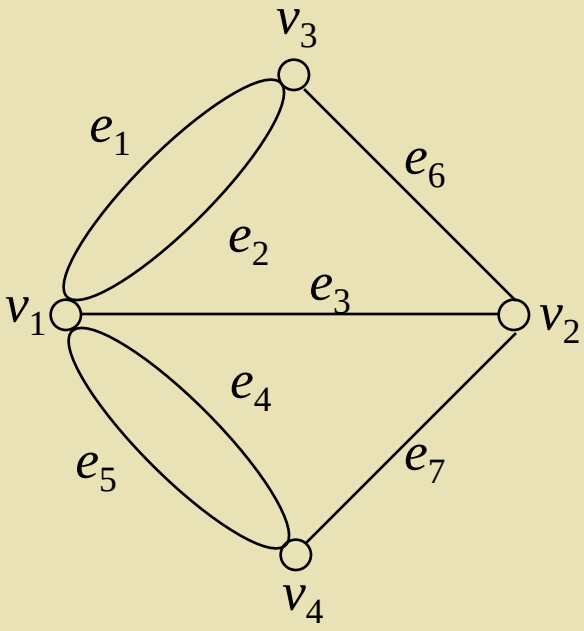


图10

离散数学

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′⊆V且E′⊆ E，则称G′为G的子图；

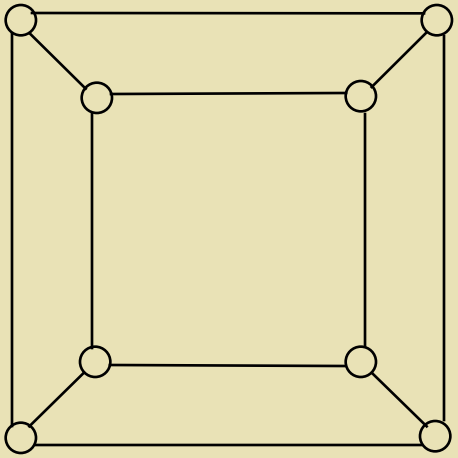
(2) 若V′⊂V或 E′⊂ E，则称G′为G的真子图(proper-)；

(3) 若V′=V且E′⊆ E，则称G′为G的生成子图(spanning-)；

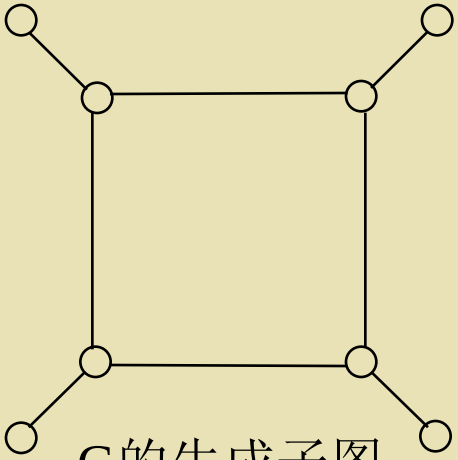
(4)当V′=V且E′=E ，或 V′=V且E′=∅时，称G′为G的平凡子图(trivial-)；即，图G 本身和G的零图是G的平凡子图。

例5. 图G及其子图、生成子图、真子图、平凡子图分别如下图11所示：

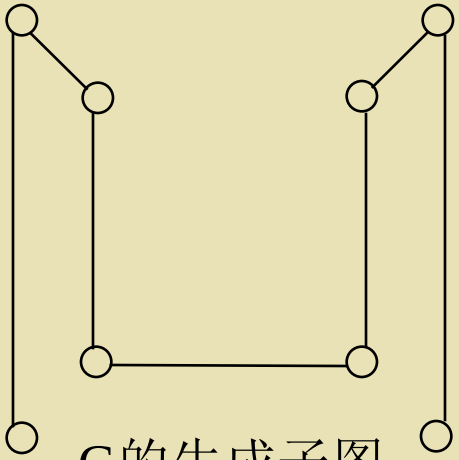
离散数学



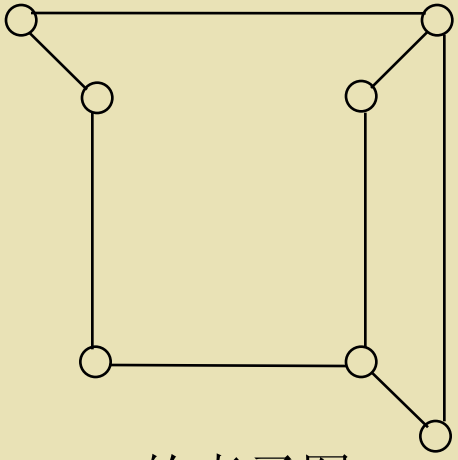
G



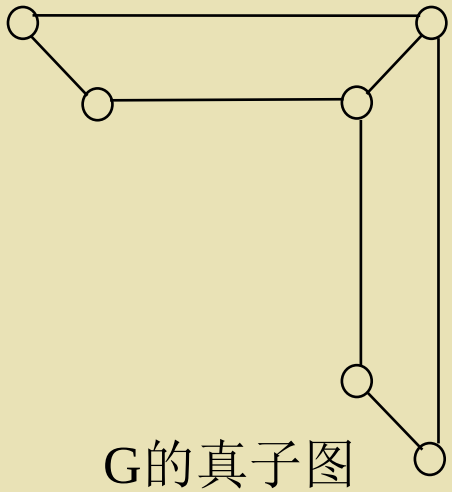
G的生成子图



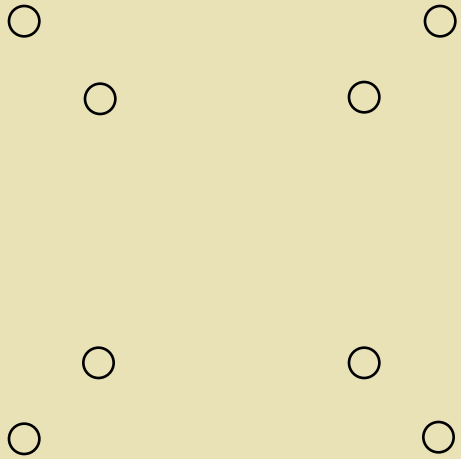
G的生成子图



G的真子图



G的真子图



G的零图

图11

离散数学

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。

那么，我们定义图G的补图 $\overline{G} = (V, \overline{E})$，其中：

$\overline{E} = E^* \backslash E$。

例7.图G 及其相应的完全图、补图分别如下图13所示：

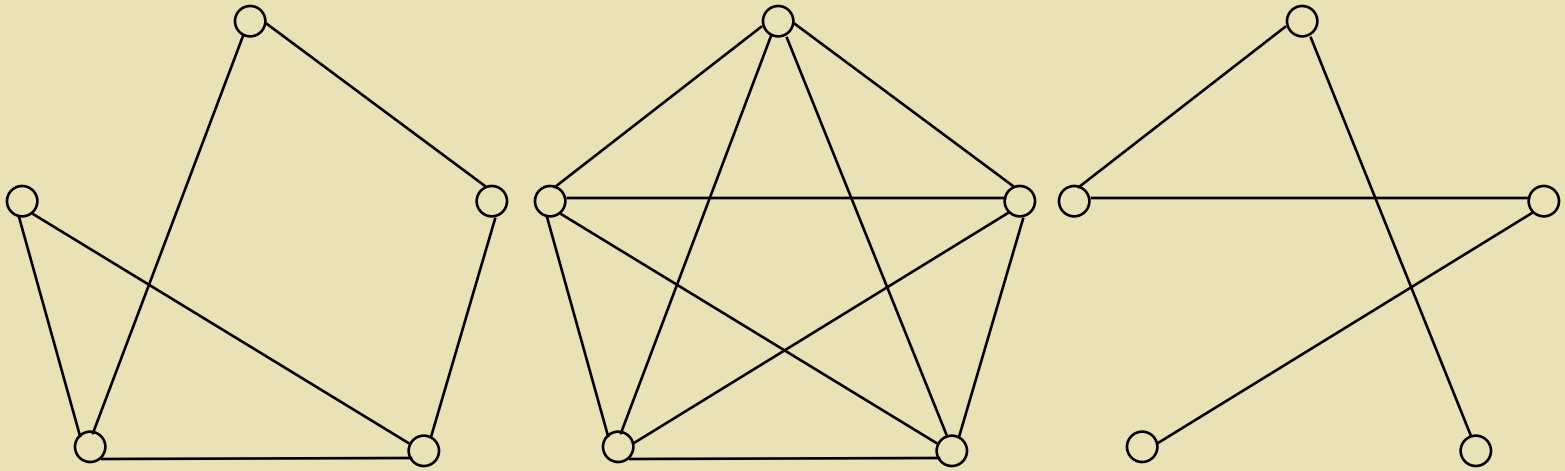


图13

离散数学

(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。

离散数学

(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 $\Delta(G)$ 。

例如前边七桥图(图3)的 $\delta(G)=3, \Delta(G)=5$ 。

(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 $\delta(G)=\Delta(G)= k$ 。

例如前边图11的图G是3-正则的，图G是正则图，这时 $\delta(G)=\Delta(G)= 3$ 。

离散数学

(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.在右面的有向图中，其中：

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

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

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

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

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

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

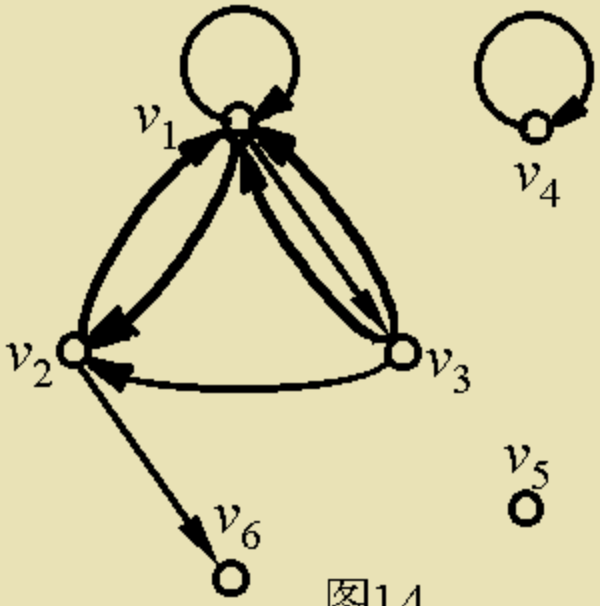


图14

离散数学

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) 。

并且有如下结论：

(1)所有结点的度数之和等于边数的二倍；

$7+4+4+2+0+1=2\times9$

(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：

离散数学

$\deg(v_1)=5,\deg(v_2)=3,\deg(v_3)=3,$
$\deg(v_4)=2,\deg(v_5)=0,\deg(v_6)=1$ ；
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) 。

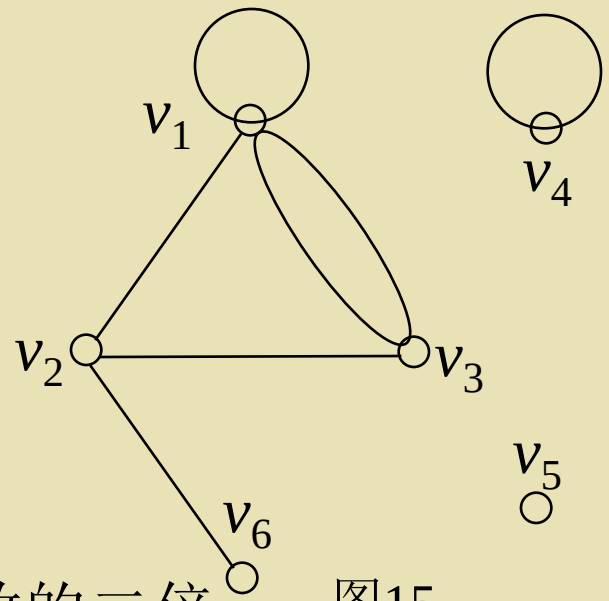


图15

并且有如下结论：

(1)所有结点的度数之和等于边数的二倍；

$5+3+3+2+0+1=2\times7$

(2)所有奇结点的度数之和是偶数；

$5+3+3+1=12$ 。

例8和例9中的结论是否具有一般性呢？下面的定理1及定理2回答了这个问题，答案是肯定的。

离散数学

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

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

(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

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

[证].

(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

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

(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

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

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$u_1, u_2,\ldots, u_{n1}$；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$w_1, w_2, \ldots, w_{n2}$，其对应的度数为$2k_1, 2k_2, \ldots, 2k_{n2}$；于是有奇结点度数之和

$$\sum_{i=1}^{n_1}\mathrm{den}(u_i)=\sum_{i=1}^{n}\deg(v_i)-\sum_{i=1}^{n_2}\deg(w_i)$$

$$=2m-(2k_1+2k_2+\ \ldots\ +2k_{n2}) \qquad \text{(定理1)}$$
$$=2[m-(k_1+k_2+\ \ldots\ +k_{n2})]$$

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].我们用结点表示人，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。

在代数系统中我们研究了代数同构这一概念，从而得知：两个代数系统同构标志着它们在结构上是完全相同、没有什么区别。在图论中我们同样可以定义图的同构这一概念，从而使我们可以研究图的结构的异同。

离散数学

我们知道，一个图可以用它的图示来描述，对于图示，我们最关心的问题是结点的个数、边的条数以及结点与边之间的相互关联关系，而对于结点的大小、边的长短、形状、标记等我们都没有作限制。观察下面四个图示

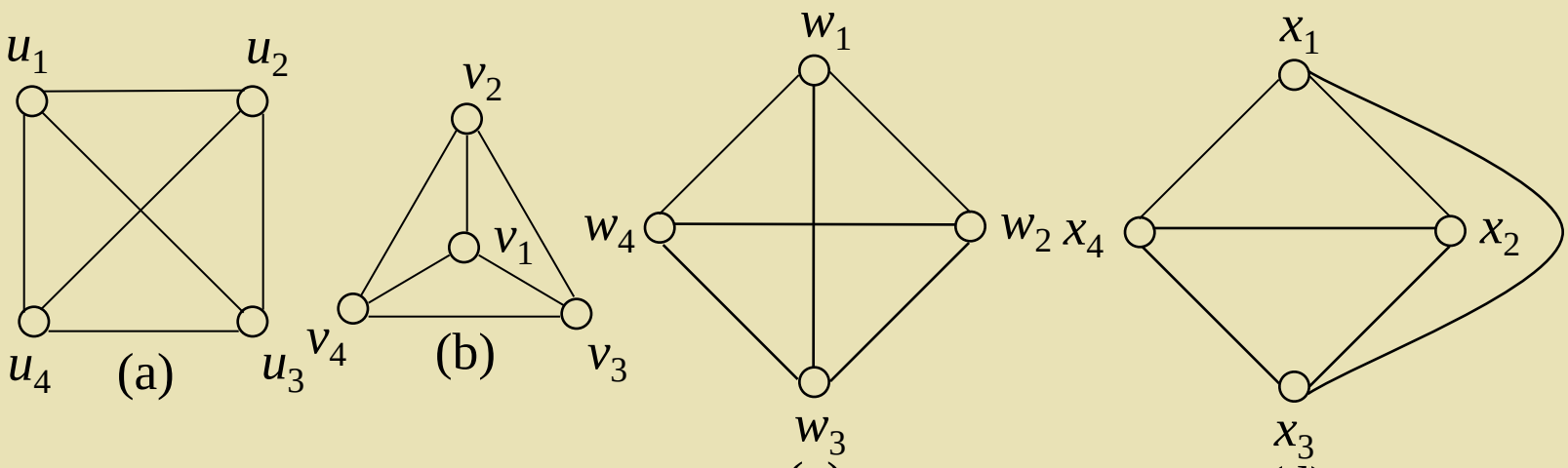


图16

离散数学

我们知道，一个图可以用它的图示来描述，对于图示，我们最关心的问题是结点的个数、边的条数以及结点与边之间的相互关联关系，而对于结点的大小、边的长短、形状、标记等我们都没有作限制。观察下面四个图示

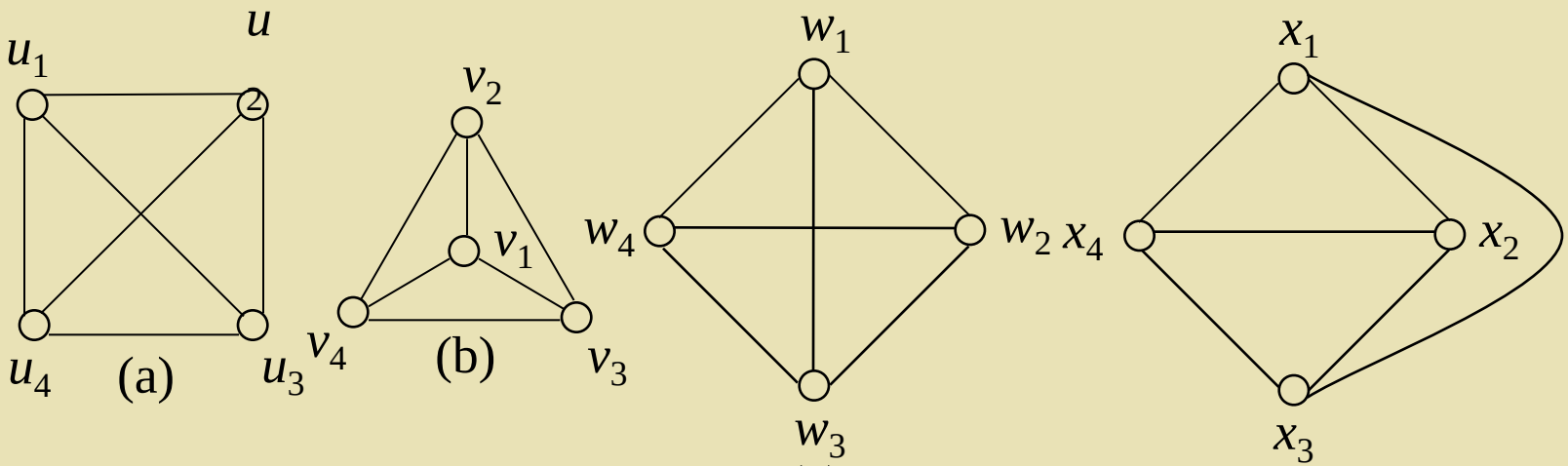


图16

图16中的四个图示，表面形状虽然不同，但这四个图示反映的结点的个数、边的条数以及结点与边之间的相互关联关系是相同的。

离散数学

若将(b)中的边(v_1,v_3)拉长(如图17所示)，即可发现(b)与(a)是一个图；(c)只是将(a)中的结点换了一个名字而已；而(d)也只是将(a)中的结点名字换了一下且改变了边(x_1,x_3)的位置。所以，图16中的四个图示所反映问题的实质是一样的，因此我们将它们看作是同构的、没有什么区别的图。

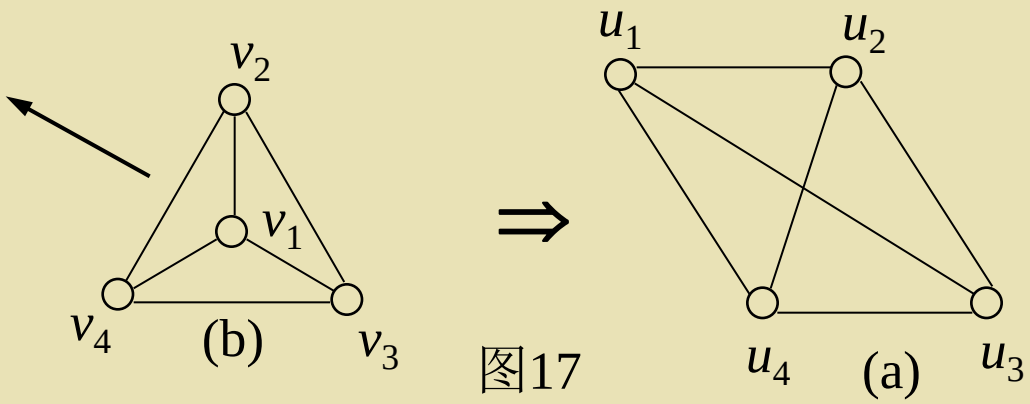


图17

离散数学

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≅G′⇔存在着两个双射函数φ及ψ，φ：V →V′，ψ：E→E′，使得

$\psi(u,v)=(u',v')\Rightarrow\varphi(u)=u'\wedge\varphi(v)=v'$ (*)

(图示如下图(1))。

2)称G=(V,E,γ)及G′=(V′,E′,γ')二图同构，记为G≅G′⇔存在着两个双射函数φ及ψ，φ：V →V′，ψ：E→E′，使得

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

(图示如下图(2)) 。

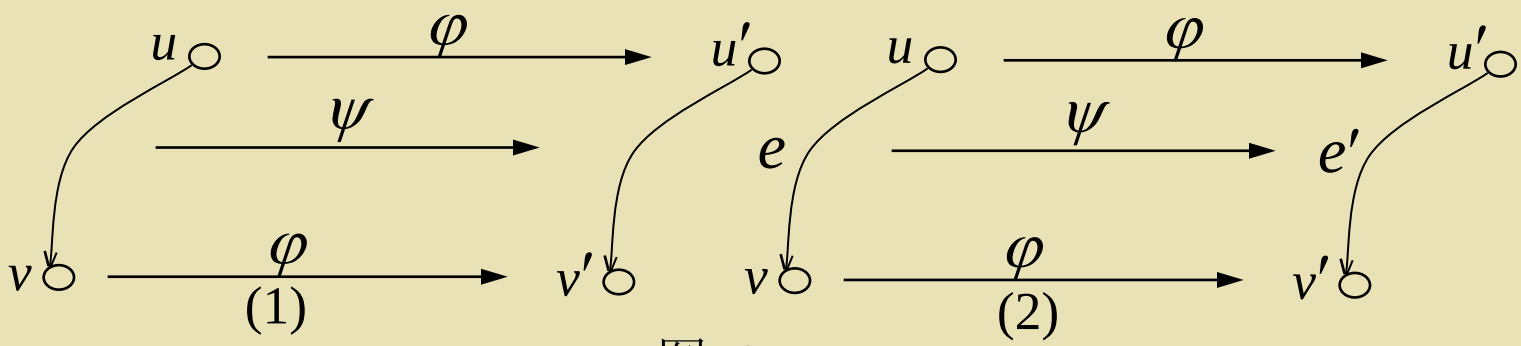


图18

离散数学

注： •此定义(1)中(*)式也可改写为：

$$\psi(u,v)=(\varphi(u),\varphi(v)) \qquad (*')$$

此定义(2)中(**)式也可改写为：

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

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；

•有关图的定义三的同构概念我们留给读者作为一个练习。

离散数学

例10.图19中的(a)和(b)二无向图是同构的。

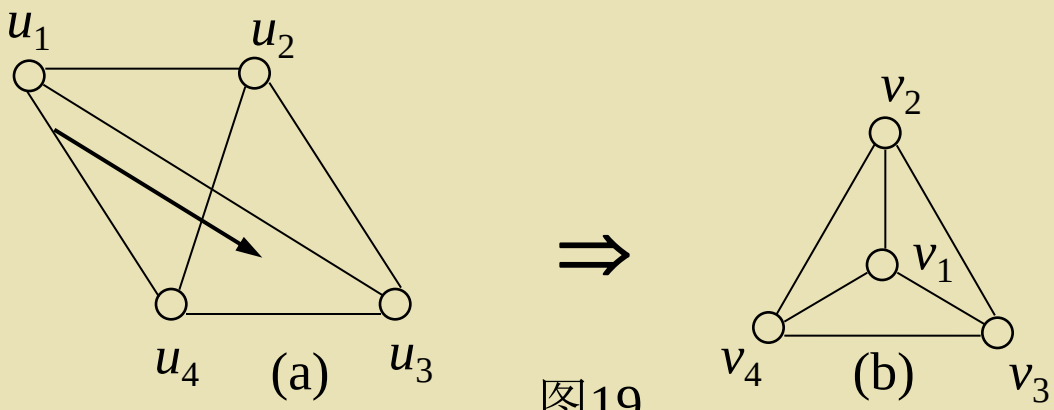


图19

(采用变形法)其同构函数为φ及ψ，使得

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

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

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

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。

离散数学

例11.图20中的(a)和(b)二无向图是同构的。

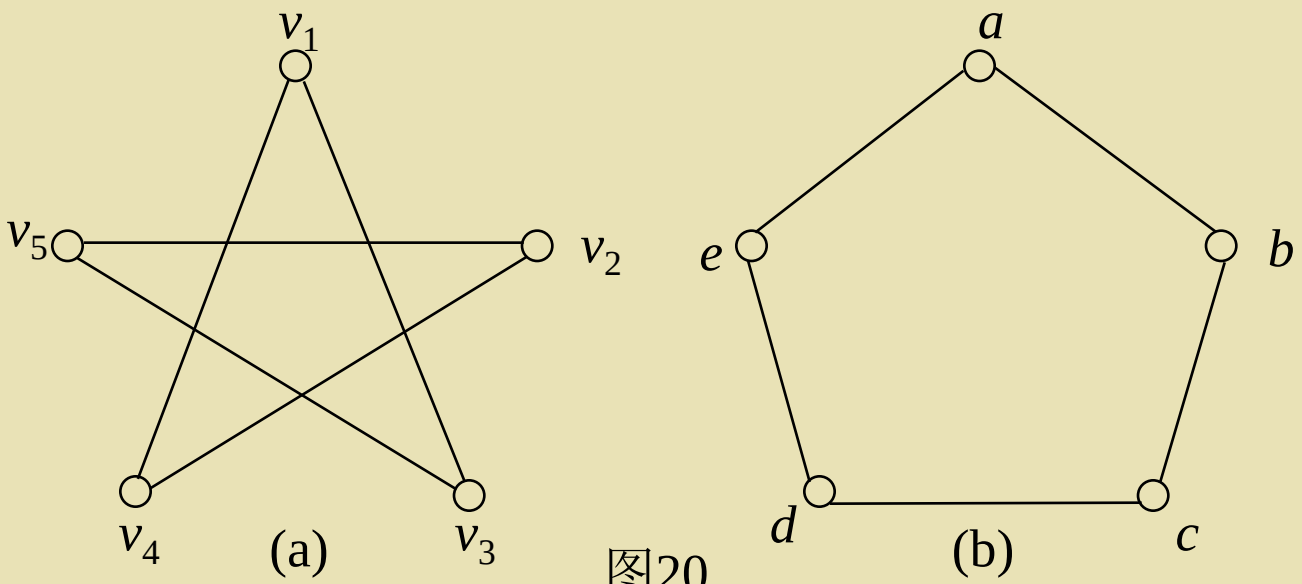


图20

(采用抻路蹦圈法)其同构函数为φ及ψ，使得

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

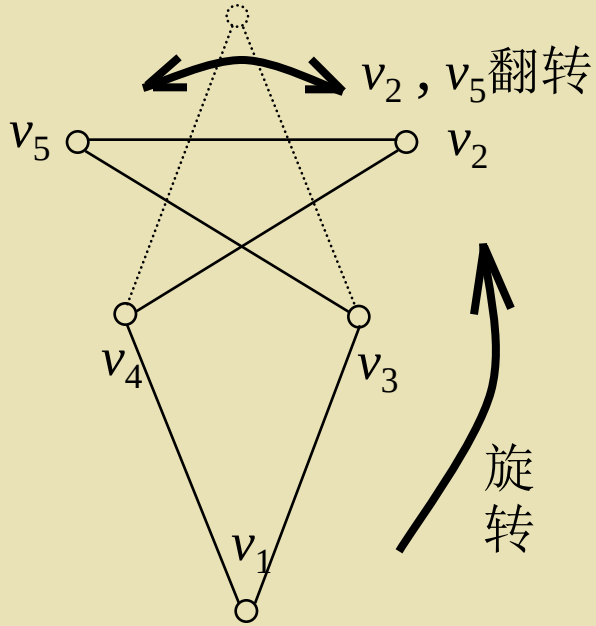
$\psi(v_1,v_3)=(a, e)$, $\psi(v_1,v_4)=(a, b)$, $\psi(v_2,v_4)=(c, b)$,

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

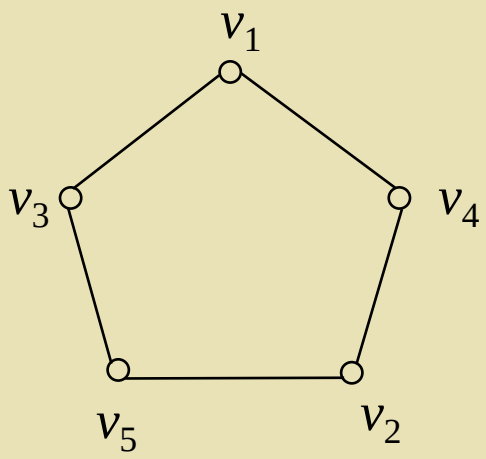
从而φ及ψ是两个双射函数，且满足同态公式(*)，因此(a)和(b)二图是同构的。

离散数学

两个图能否同构，从直观的意义上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。在上面的例子中，将(a)图示作如下变形一蹦圈后就可得到(d)图示：



(c)



(d)

图21

此时，我们可以发现，寻找同构函数的事情就变得很容易了。

离散数学

例12.图22中的(a)和(b)二无向图是同构的。

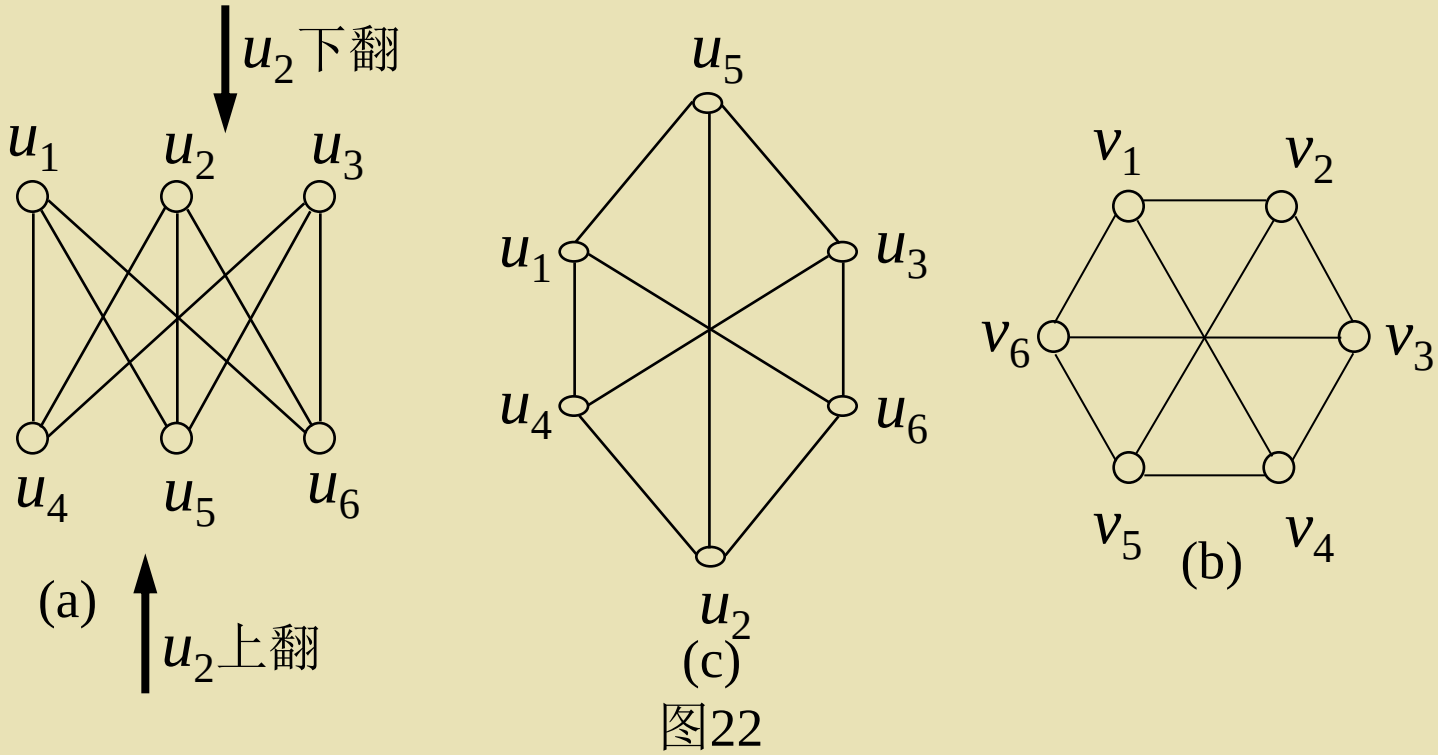


图22

(采用抻路蹦圈法)其同构函数为φ及ψ，使得

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

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

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

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

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

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。

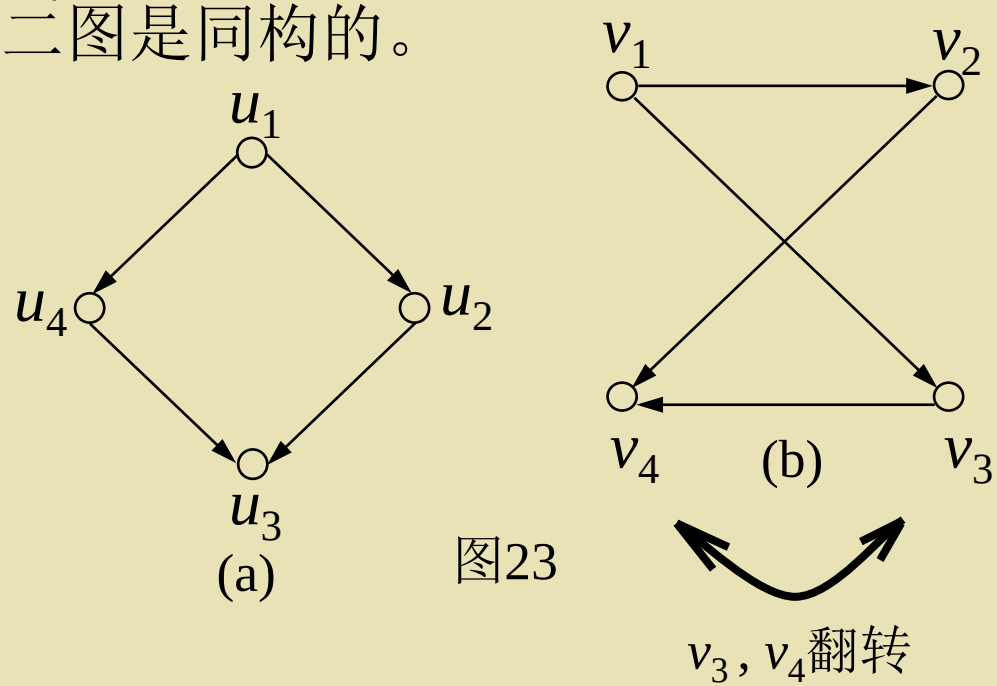


图23

离散数学

由上面的讨论我们不难看出，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；

… … …

然而，上面的那些条件只是两图同构的必要条件，并不是充分条件。

例14.图24中(a)、(b)两图不同构。

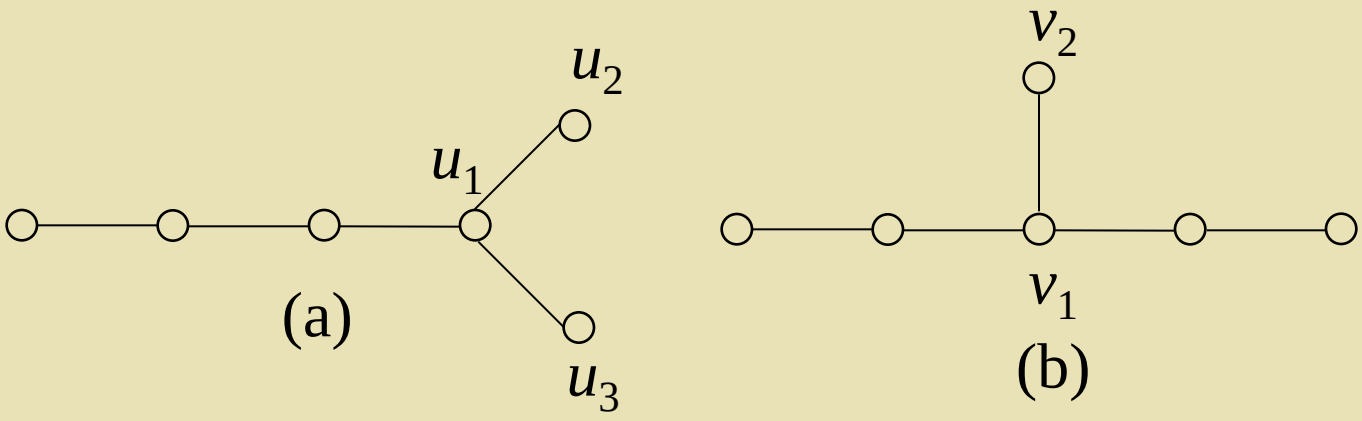


图24

图24中(a)、(b)两图满足除(6)和(7)以外的上述条件，但是，由于(a)中唯一的度为3的结点u_1与两个悬挂点u_2、u_3相邻；而在(b)中唯一的度为3的结点v_1只与一个悬挂点v_2相邻，所以(a)、(b)两图不满足上述条件(6)和(7)，故(a)、(b)不同构。

注：•因此，这些条件只能用来否定图同构，不能用来证明图同构。即

两图不满足这些条件之一$\Rightarrow$ 两图不同构；

两图满足全部这些条件$\not\Rightarrow$ 两图同构。

容易看出：同构关系是一个等价关系。对于图，我们感兴趣的问题是边与点的连接关系，所以在画图示时有时略去标号。有了同构的概念以后，我们可以把一个无标号的图看作是同构图的等价类的代表元。

离散数学

- **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

$V=\{v_1, v_2, \ldots, v_n\}$ 及 $V'=\{v_1', v_2', \ldots, v_n'\}$ $(n\geq 3)$，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：$H_i=G\backslash\{v_i\}=(V_i,E_i)$,

$V_i=V\backslash\{v_i\}$,

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

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

$V_i'=V'\backslash\{v_i'\}$,

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

§3.路与圈

- 定义与实例
- 基本概念
- 可达性
- 图的连通性

离散数学

§3.路与圈

定义1.途径(way)

设G=(V,E,γ)是一图。G的有限非空点边交错序列

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

, 若满足条件:

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

(2)当e_i和e_{i+1}不是自环时，有$e_{i+1}\neq e_i$ $(1\le i\le n)$(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：•需说明的是：当G=(V,E)是一简单图时，在实际应用中，为方便起见，常用纯边列或纯点列的方式表示途径:

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

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

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

离散数学

•值得注意的是：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：

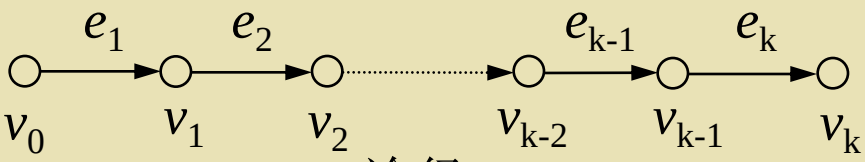
$V(w)=\{v_0, v_1, v_2, \ldots, v_k\}$ (去掉重复)

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

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

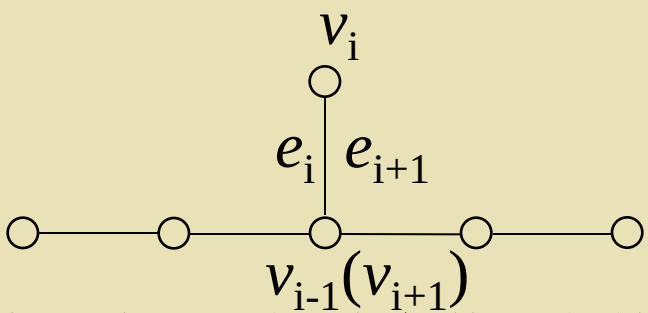
使得 $\gamma(w)(e_i)=\{v_{i-1}, v_i\}$(或$(v_{i-1}, v_i)$) $(1\le i\le n)$ (去掉重复) 。

•途径定义1中条件(1)、(2)的情况如下图1所示：



途径w

(a)



不满足途径定义1中条件(2)的情况

(b)

图1

离散数学

定义2.路(path)、圈(cycle)

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

$P=(v_0, v_1, v_2, \ldots, v_{k-1}, v_k)$，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

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

离散数学

例1.在图2中由结点v_1到结点v_3的路有：

$P_1=(v_1,v_2,v_3)$
$P_2=(v_1,v_4,v_3)$
$P_3=(v_1,v_2,v_4,v_3)$
$P_4=(v_1,v_2,v_4,v_1,v_2,v_3)$
$P_5=(v_1,v_2,v_4,v_1,v_4,v_3)$
$P_6=(v_1,v_1,v_2,v_3)$
… … …

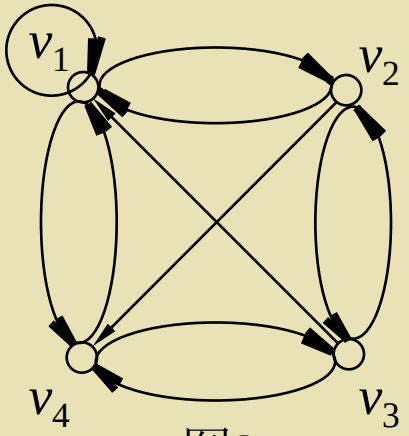


图2

在图2中，由结点v_1到v_1的圈有：

$C_1=(v_1,v_1)$
$C_2=(v_1,v_2,v_1)$
$C_3=(v_1,v_2,v_3,v_1)$
$C_4=(v_1,v_4,v_3,v_1)$
$C_5=(v_1,v_2,v_3,v_2,v_1)$
$C_6=(v_1,v_2,v_3,v_2,v_3,v_1)$
… … …

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。

离散数学

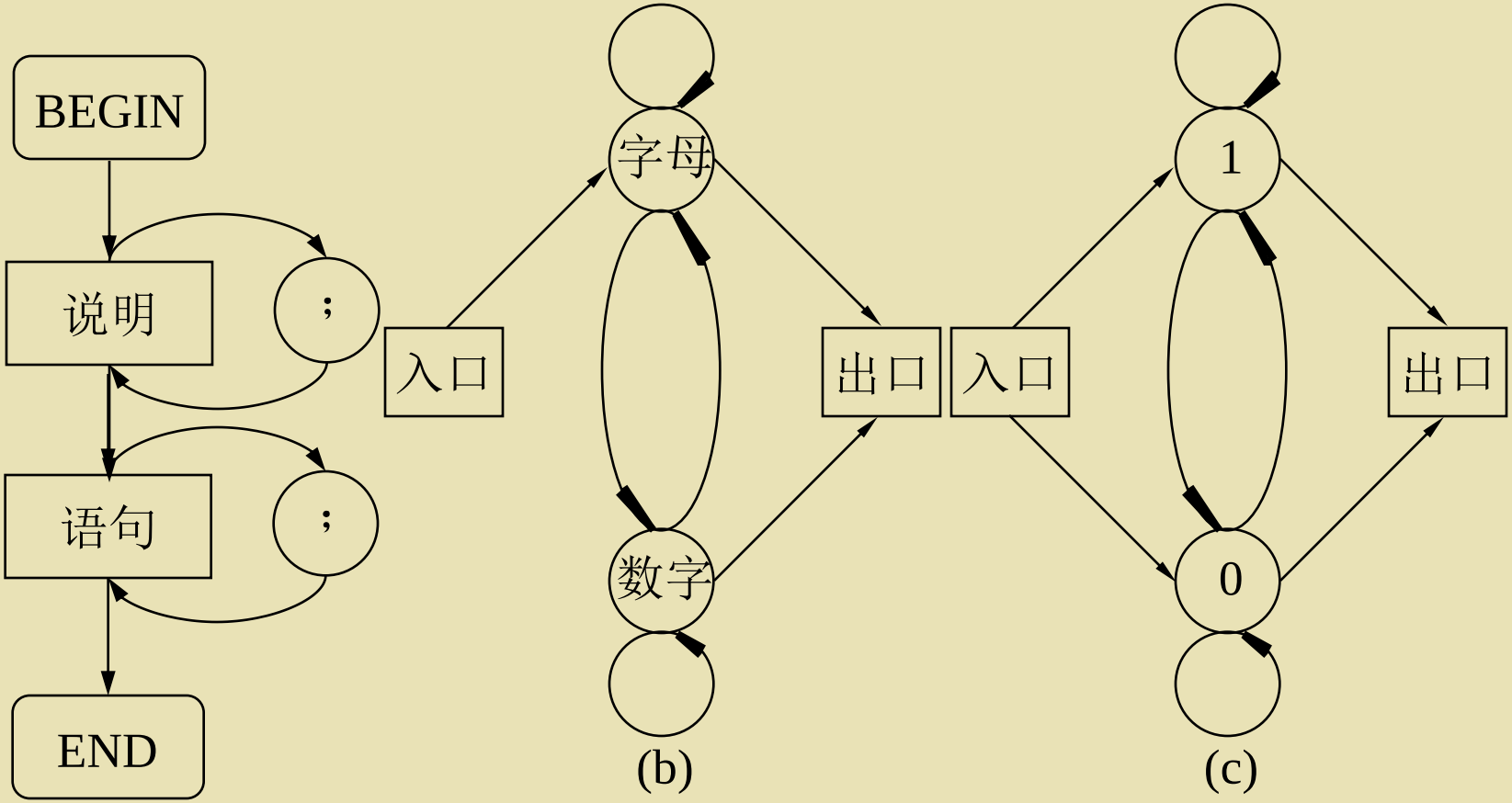


图3

离散数学

定义3.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4.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。

离散数学

例3. 在例2的路和圈中：

P_1, P_2, P_3是初级路；

P_5 , P_6是简单路；

P_4既非初级路又非简单路。

C_1, C_2, C_3, C_4是初级圈；

C_5是简单圈；

C_6既非初级圈又非简单圈。

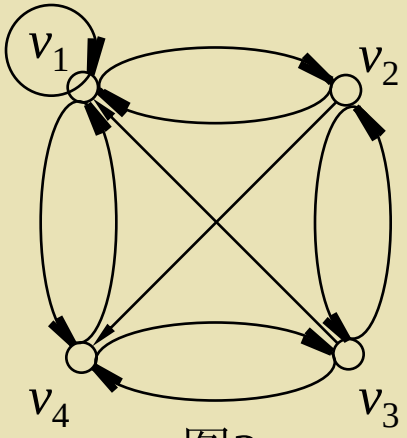


图2

$P_1=(v_1,v_2,v_3)$

$P_2=(v_1,v_4,v_3)$

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

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

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

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

$C_1=(v_1,v_1)$

$C_2=(v_1,v_2,v_1)$

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

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

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

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

例4.在图4中由结点v_1到结点v_5的路有：

$P_1=(v_1,v_2,v_5)$是初级路；

$P_2=(v_1,v_2,v_6,v_2,v_5)$是简单路；

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

既非初级路又非简单路。

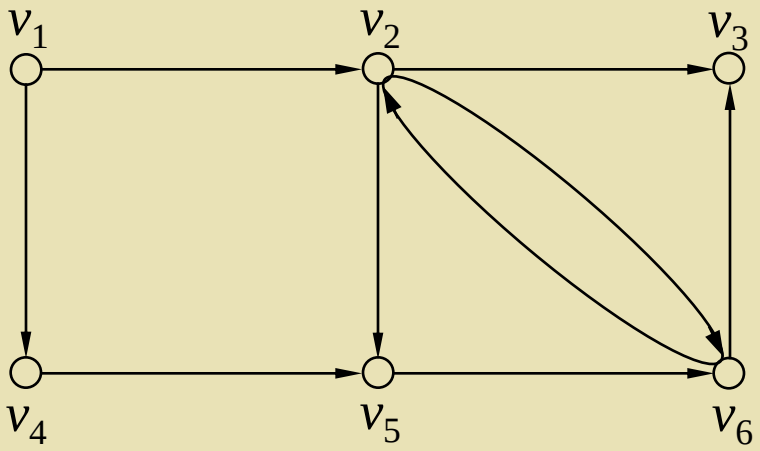


图4

注：•初级路(圈)一定是简单路(圈)；
简单路(圈)不一定是初级路(圈)。

•任给一条路，都可通过去掉其中所有的圈而形成一条初级路。
任给一个圈，都可通过去掉其中所有的“小圈”而形成一个初级圈。

离散数学

关于初级路及初级圈的长度有如下定理：

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。

注： •利用路的概念可以解决一些有趣的问题。

例如，在§1中人、狼、羊、菜安全渡河问题就是求一条从(人、狼、羊、菜)状态到(空)状态的路径，而且为了要使渡河所花费的时间达到最少，故要求所求的路径应是一条初级路。

这里，我们再举一个例子：

离散数学

例5.分油问题：

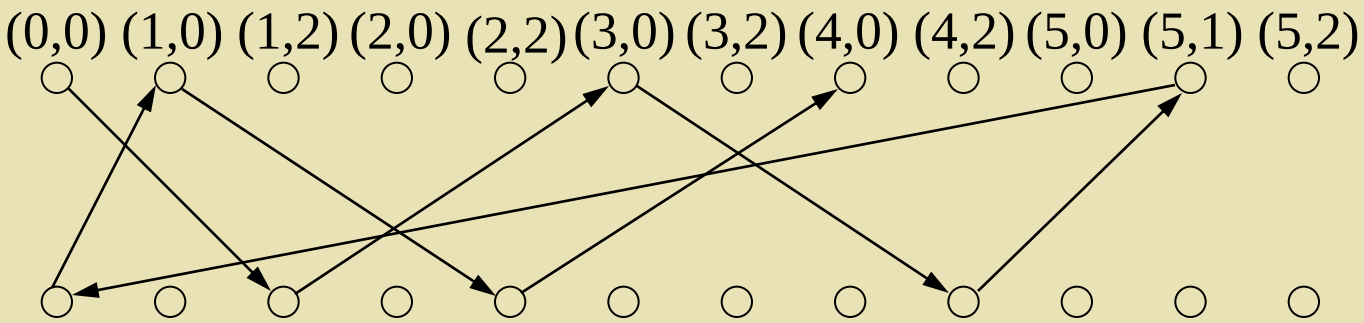
有三个油桶a,b,c分别可装8斤,5斤,3斤油。假设a桶已装满了油，在没有其它度量工具的情况下，如何将油平分？

[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。

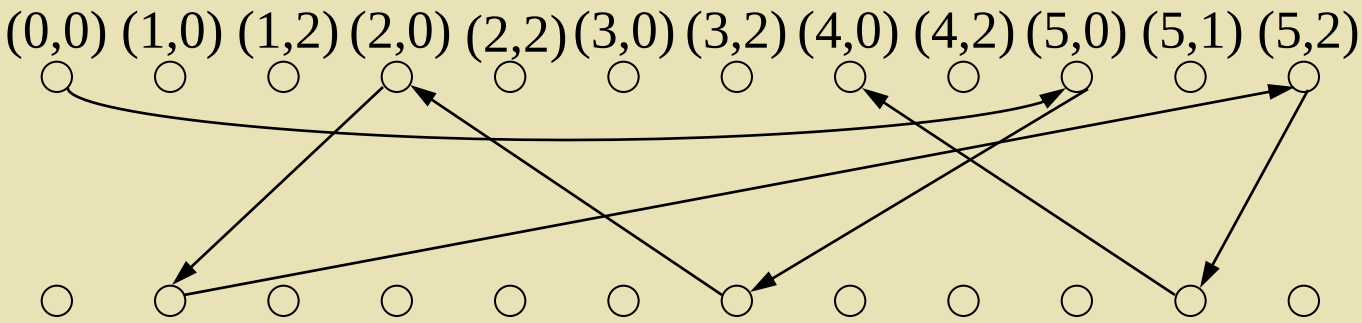
其次，我们用二元组(b,c)来表示b,c两个桶中装油的各种可能状态，由于$0\le b\le 5$, $0\le c\le 3$，故(b,c)共有24种不同的状态。每一种状态用一结点表示，两结点间有边相连当且仅当这两种状态可通过倒油的方法互相转换。规定起始状态为(0,0)，终结状态为(4,0)，其它则为中间状态。

同时，为了尽快将油分好，要求中间状态不重复出现。

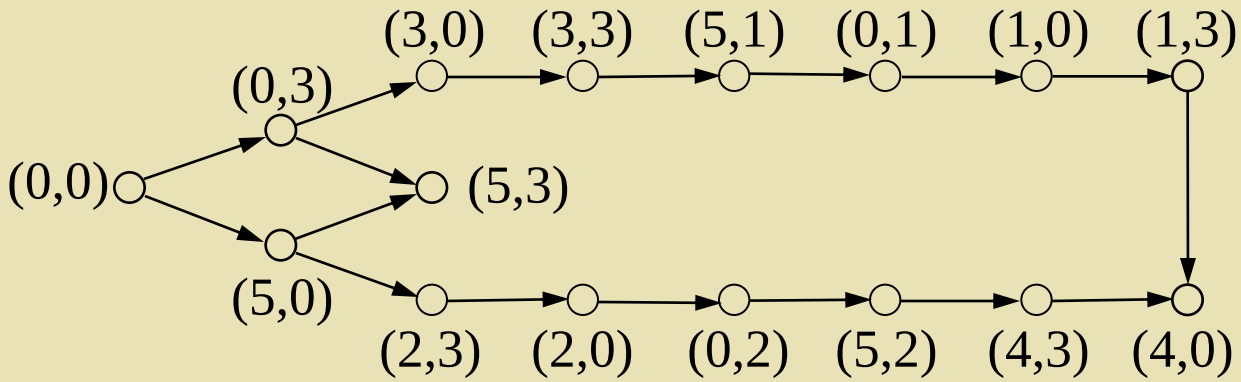
离散数学


(0,0) (1,0) (1,2) (2,0) (2,2) (3,0) (3,2) (4,0) (4,2) (5,0) (5,1) (5,2)
(0,1) (0,2) (0,3) (1,1) (1,3) (2,1) (2,3) (3,1) (3,3) (4,1) (4,3) (5,3)

(a)


(0,0) (1,0) (1,2) (2,0) (2,2) (3,0) (3,2) (4,0) (4,2) (5,0) (5,1) (5,2)
(0,1) (0,2) (0,3) (1,1) (1,3) (2,1) (2,3) (3,1) (3,3) (4,1) (4,3) (5,3)

(b)



(c)

图5

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)。

离散数学

定义5.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。

离散数学

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，可能会有好几条路。我们称从结点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。我们规定：

  (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；
- 然而对有向图来说，第二条，对称性质不成立。

注二：
- 需要说明的是：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≥2)；

- 我们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。

离散数学

注：•由于无向图中的可达性是自反的、对称的和传递的，所以可达性实际上是建立了图G的结点集V上的一种等价关系。从而确定了图G的结点集V上的一个划分。每一个划分块都是所给定的图G的一个连通子图，而且在每一个划分块中结点间的可达性已达到极大限度。

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。

离散数学

例6.在图6中有三个连通支(分图)；因此不是连通图。

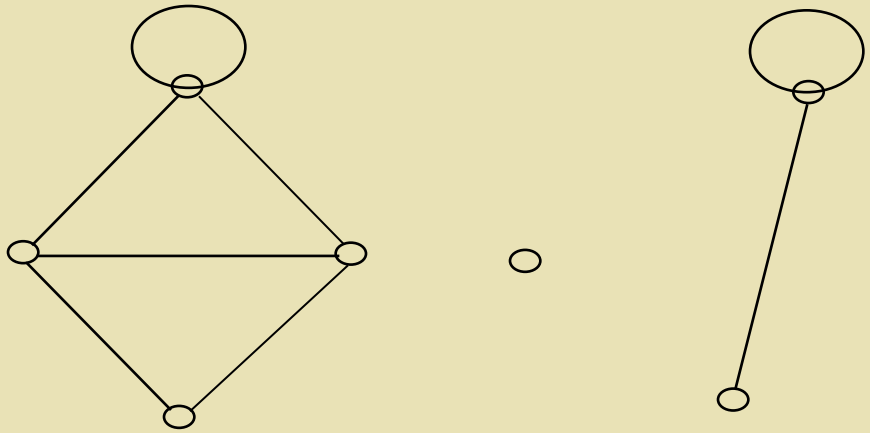


图6

注：•图G是连通的⇔图G的连通支数是1；

•对于有向图，由于其可达性不具有对称的性质，所以有向图连通性的概念相对于无向图要复杂一些。

离散数学

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)；

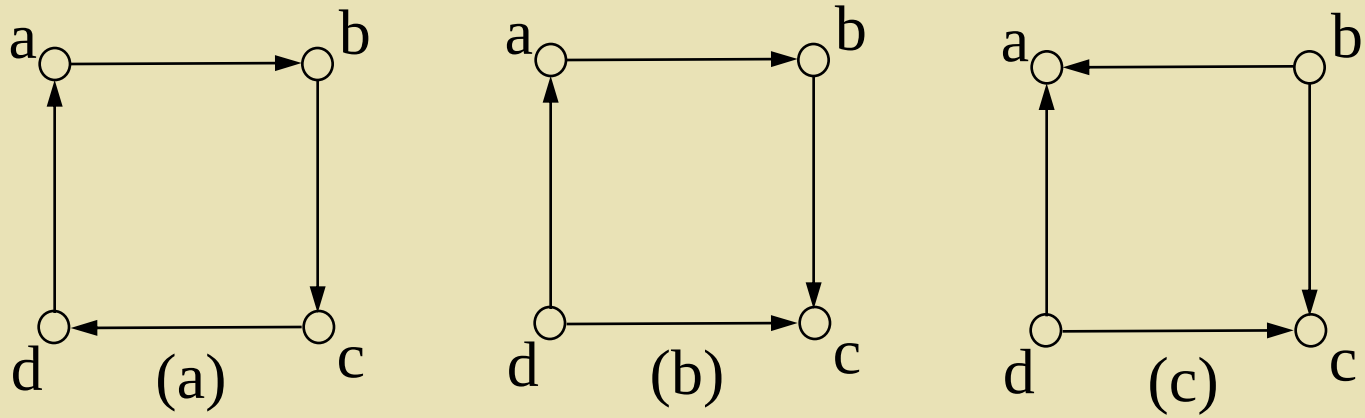
(3)略去边的方向后，任意两结点间都是可达的(即图G的无向图是连通的)，则称图G是弱连通的(weakly connected)。

注：•强连通⇒单向连通；反之不真；

•单向连通⇒弱连通；反之不真。

离散数学

例7.在图7中



(a)是强连通的；

(b)是单向连通的；但不是强连通的，因为从结点c不可达结点a,b,d；

(c)是弱连通的；但不是强连通的，因为结点a,c相互不可达；也不是单向连通的，因为从结点b不可达结点d。

离散数学

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))。

离散数学

例8.在图8中

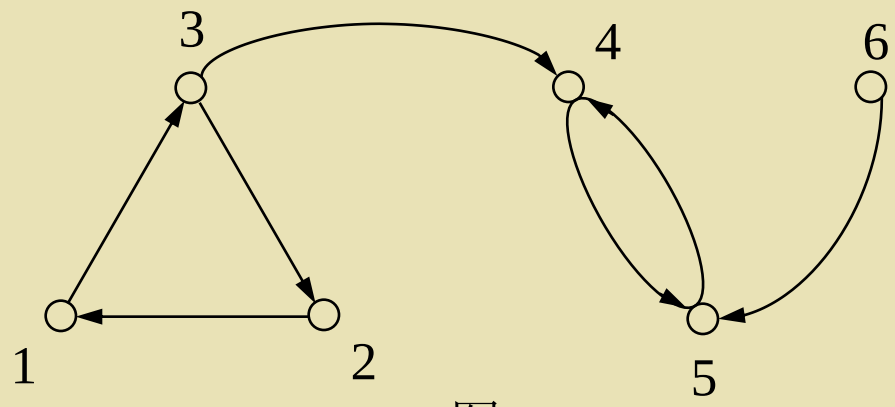


图8

(1)强分图有三个：

$G_1=(\{1,2,3\},\{(1,3),(3,2),(2,1)\})$

$G_2=(\{4,5\},\{(4,5),(5,4)\})$

$G_3=(\{6\},\varnothing)$

(2)单向分图有二个：

$G_4=(\{1,2,3,4,5\},\{(1,3),(3,2),(2,1),(3,4),(4,5),(5,4)\})$

$G_5=(\{4,5,6\},\{(4,5),(5,4),(6,5)\})$

(3)弱分图就一个，是G本身：

$G_6=G$

$=(\{1,2,3,4,5,6\},\{(1,3),(3,2),(2,1),(3,4),(4,5),(5,4),(6,5)\})$。

离散数学

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”；

•对于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”；

如在例8图8中，两个单向连通支G_4,G_5的结点集及边集都是相交的，不是划分块。

离散数学

基于上述思想，有如下定理：

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。

[证].证明留给大家作为一个习题。

下面我们介绍两个在计算机科学中应用连通性的两个例子。

离散数学

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)。但这些程序在占有一些资源的同时，又都要请求一些资源。在同一时间，既不释放占有的资源，又不放弃资源的请求。这样，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。我们称计算机的这种状况为“死锁”状态。

计算机操作系统中应该有专门的进程来负责动态的检测和处理死锁状态。

离散数学

下面我们将同一时刻多个程序占有并请求资源的状况抽象成一个有向图，然后利用图论知识来分析产生死锁的特征，从而给出检测死锁的算法。

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，并表记上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)

显然本例中的有向图G是强连通的，所以必然产生死锁现象。

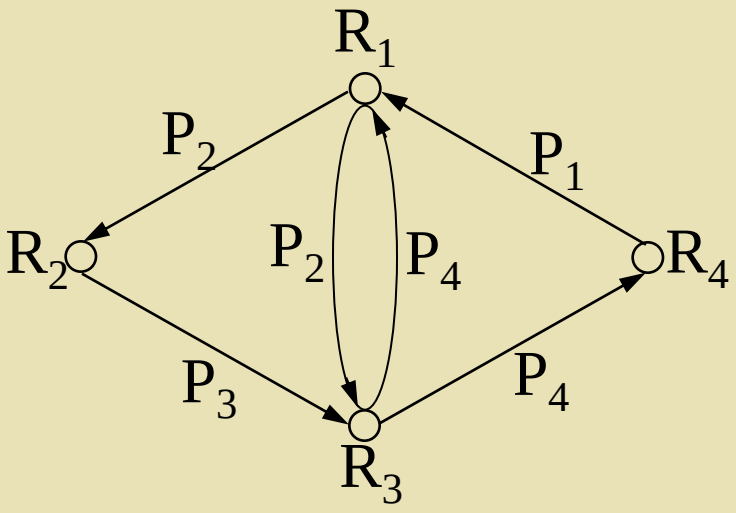


图9

例10.图的强连通性概念在检测递归调用问题上的应用

在程序设计中，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。

我们的思想是将过程间的调用关系用有向图来表示，

离散数学

从而可用有向图的某种图论性质来刻划某过程的递归性。下面我们将某一程序多个过程间的调用关系抽象成一个有向图，然后利用图论知识来分析含有递归的特征，从而给出检测递归性的算法。

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Pi调用过程P_j时，则从结点Pi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。

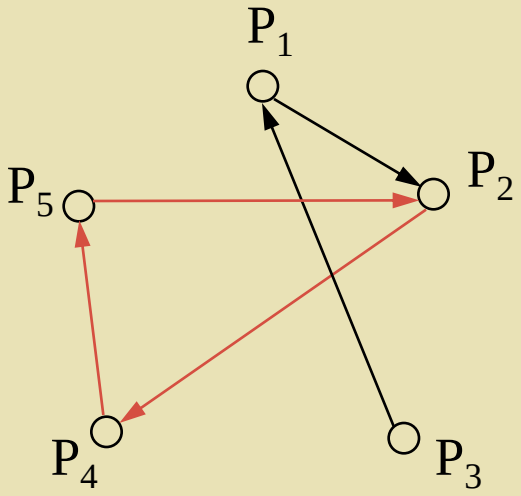


图10

离散数学

含有递归的特征：

程序中含有过程递归调用⇔过程调用图G中含有强连通支(有向圈)

显然本例中的有向图G含有强连通支

$$G'=(\{P_2,P_4,P_5\}\{(P_2,P_4),(P_4,P_5),(P_5,P_2)\})$$

所以过程P_2,P_4,P_5是递归的。

§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$V=\{v_1,v_2,\ldots,v_n\}$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。

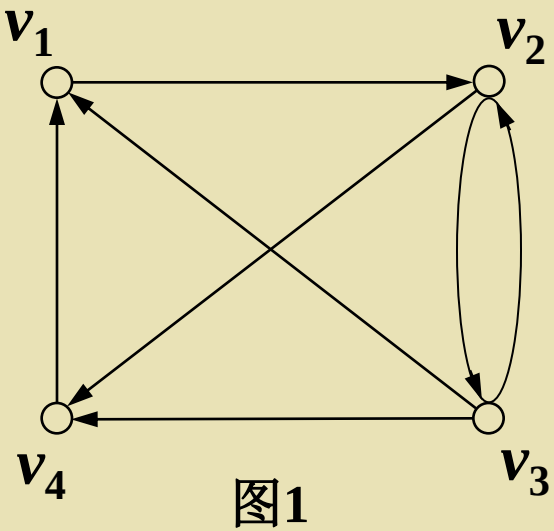


图1

V ={v1,v2, v3, v4}已强行命名。

$$B=\begin{array}{c} \\ v_1\\ v_2\\ v_3\\ v_4\end{array}\begin{array}{c}\begin{array}{cccc}v_1 & v_2 & v_3 & v_4\end{array}\\ \begin{pmatrix}0&1&0&0\\0&0&1&1\\1&1&0&1\\1&0&0&0\end{pmatrix}\end{array}$$

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。

$$B=\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array}\begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

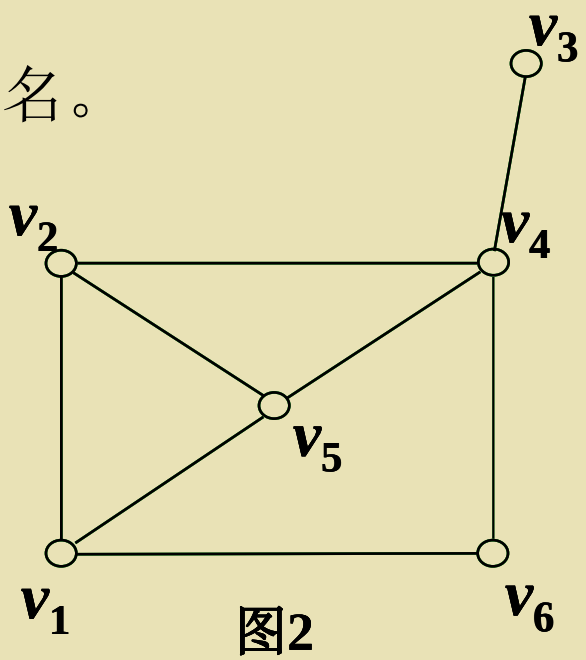


图2

注： 图G是无向图⇔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0～1方阵对应；每个主角线元素全为零的0～1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0～1方阵都表示同一个图；

(3)两个简单图同构⇔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⇔其邻接矩阵A是全0矩阵；

(5)图G是完全图⇔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 $\sum_{k=1}^{n} a_{ik} = \overleftarrow{dev}(v_i)$

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

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

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

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={$v_1,v_2,\ldots,v_n$}， E={$e_1,e_2,\ldots,e_m$}已被强行命名。则 定义n×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

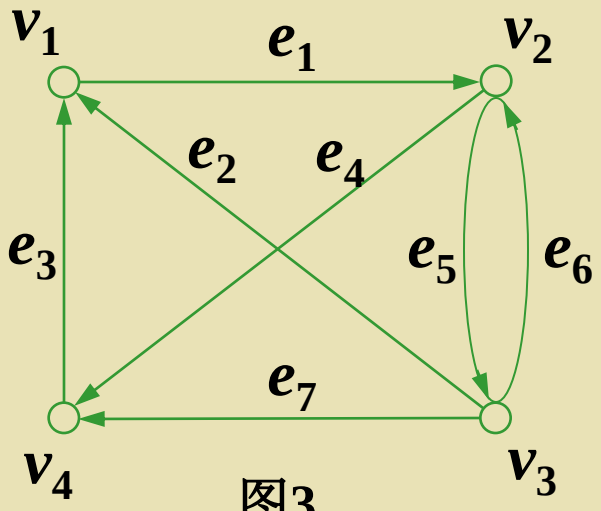


图3

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

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

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

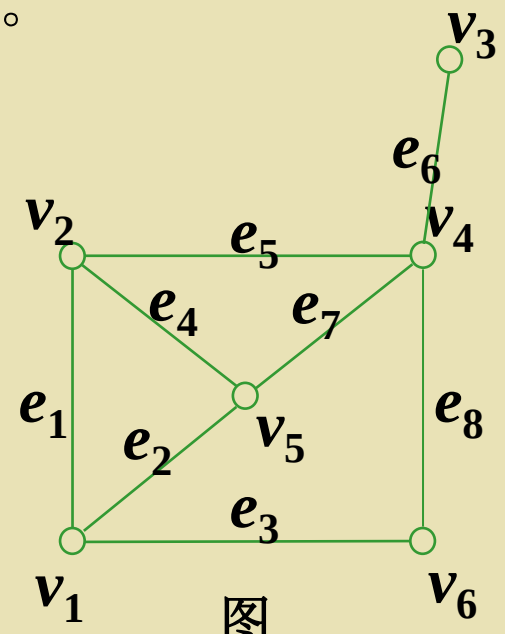


图4

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={$v_1,v_2,\ldots,v_n$}已被强行命名。则 定义n阶方阵

$R=(r_{ij})_{n\times n}$ ，其中

$$r_{ij}=\begin{cases}1, & \text{若从结点}\boldsymbol{v}_i\ \text{可达结点}\boldsymbol{v}_j\\ 0, & \text{若从结点}\boldsymbol{v}_i\text{不可达结点}\boldsymbol{v}_j\end{cases}$$

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⇔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⇔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⇔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⇔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

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

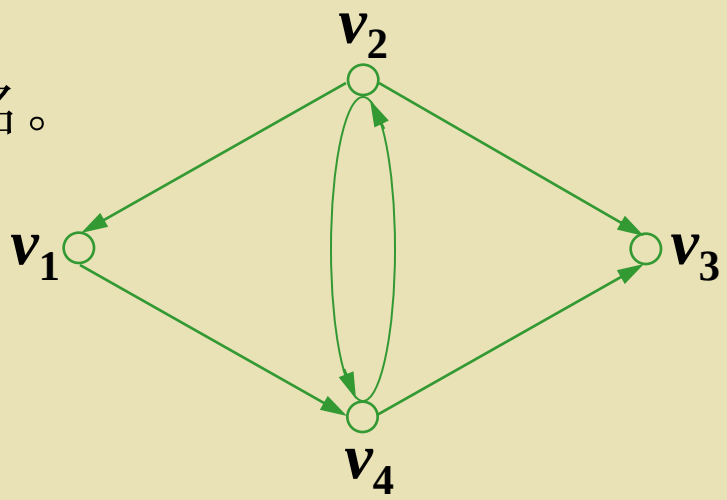


图5

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，因为需要找出任二结点间的一条(有向、无向)路。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： $a^{(m+1)}=\bigvee_{k=1}^{n}(a_{ik}^{(m)}\wedge a_{kj})$ $(1\le i\le n,1\le j\le n)$。

注：•这里∧和∨都是二元布尔运算：布尔乘和布尔和。

•其运算表如下：

∧	0	1
0	0	0
1	0	1

∨	0	1
0	0	1
1	1	1

表1

- **可达矩阵的计算方法一：(矩阵幂算法)**

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

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。

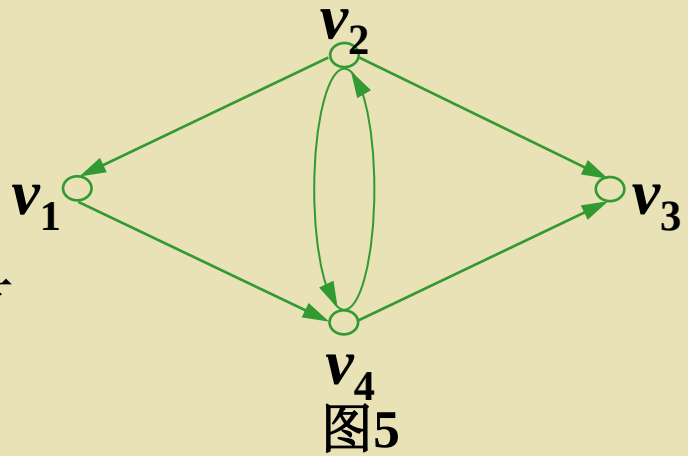


图5

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

$$A=\begin{pmatrix}0&0&0&1\\1&0&1&1\\0&0&0&0\\0&1&1&0\end{pmatrix}\quad A^{(2)}=\begin{pmatrix}0&1&1&0\\0&1&1&1\\0&0&0&0\\1&0&1&1\end{pmatrix}\quad A^{(3)}=\begin{pmatrix}1&0&1&1\\1&1&1&1\\0&0&0&0\\0&1&1&1\end{pmatrix}$$

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

有向图G的可达矩阵

$R=A\vee A^{(2)}\vee A^{(3)}\vee A^{(4)}$

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

- 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

$$r_{ij}^{(k)} := r_{ij}^{(k-1)} \vee (r_{ik}^{(k-1)} \wedge r_{kj}^{(k-1)}) \quad ;$$

•于是 有 $r_{ij}^{(k)} = 1$

⇔从结点v_i到结点v_j有直接边或者有通过结点$v_1, v_2, \ldots, v_k$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、免强的、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

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

$$\boldsymbol{R}^{(3)}=\begin{pmatrix}0&0&0&1\\1&0&1&1\\0&0&0&0\\1&1&1&1\end{pmatrix} \qquad \boldsymbol{R}^{(4)}=\begin{pmatrix}\underline{1}&\underline{1}&\underline{1}&1\\1&\underline{1}&1&1\\0&0&0&0\\1&1&1&1\end{pmatrix}=\boldsymbol{R}$$

- ***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⇔可达矩阵R是全1矩阵；

(2)有向图G单向连通⇔$R \vee R^T$除主对角线外均是1；

(3)有向图G弱连通⇔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⇔可达矩阵R的某些主对角线元素$r_{ii}=1$；

离散数学

§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→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

并称图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。

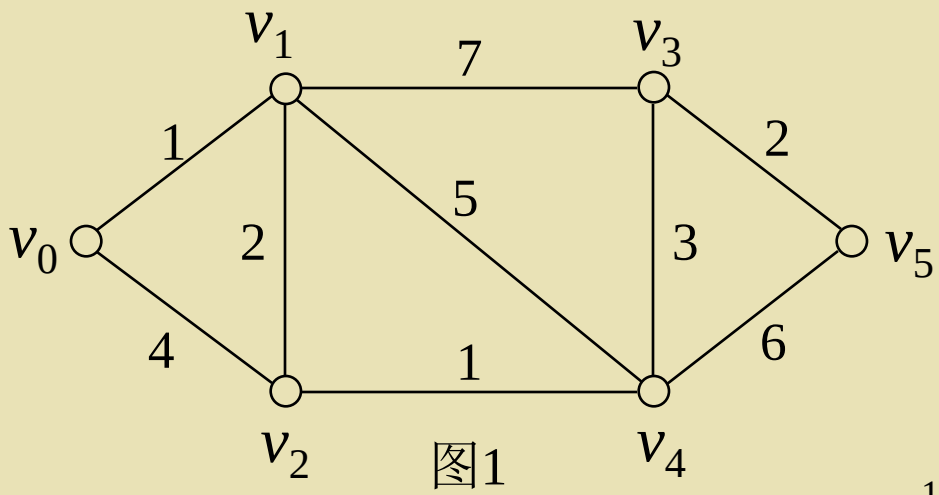


图1

注：带权图有时也称为网络(network)。

离散数学

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

(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。在《数据结构》课程中读者们将会遇到。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},…, 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：

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

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

***E.W.Dijkstra*算法**(1959)：

此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。

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

离散数学

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

$p := v_0$ ； 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)*/

$(\exists t_0\in T)(\forall t\in T)(d(t_0)\le d(t))$ ； /*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\$\{t_0\}$ ； /*修改P-T界面*/

$p := t_0$ ； mark$(t_0) := d(t_0)$ ； /*改圆为方*/

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

else goto No2 ；

◆**算法的基本思想：** 最短路上的任何一段(局部)也都是最短的。即，如果$P_0 =(v_0, v_1, v_2,\ldots, v_n)$是一条最短路，则其

中任一段$(v_i, v_{i+1}, v_{i+2},\ldots, v_j)$ $(0\le i< j\le n)$也是最短的。

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

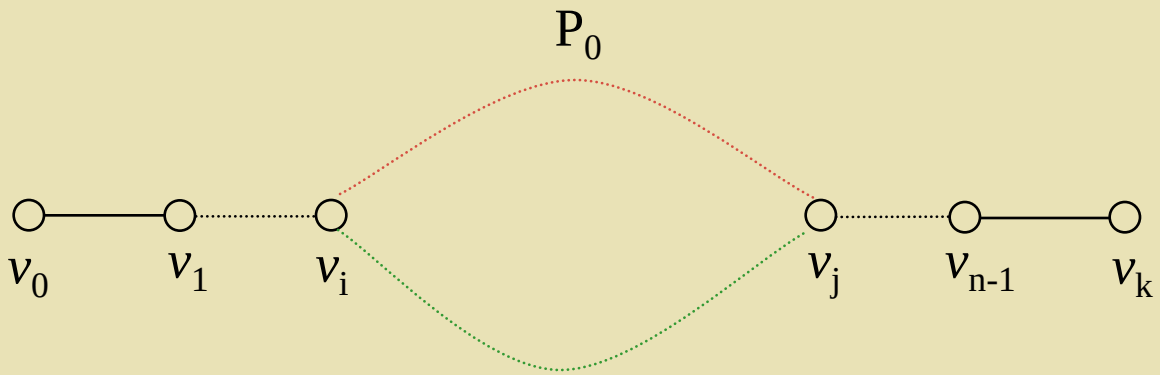


图2

离散数学

- ***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。

(1)图上作业法：

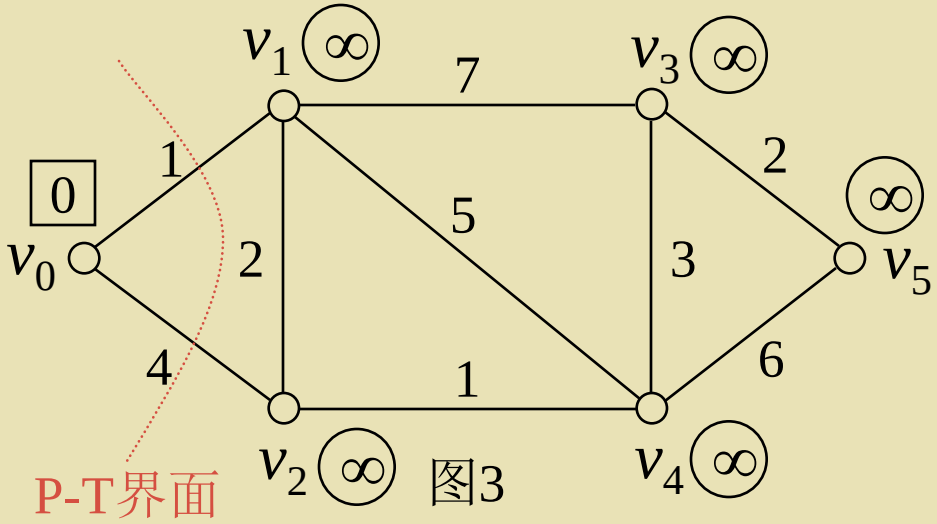


图3

设定P-T界面：

结点v_0 做方标号；

其余结点做圆标号；

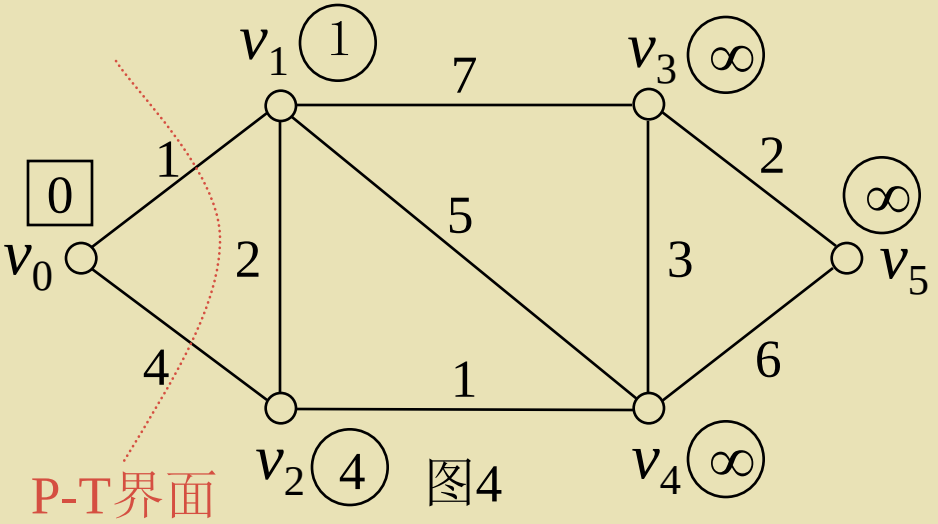


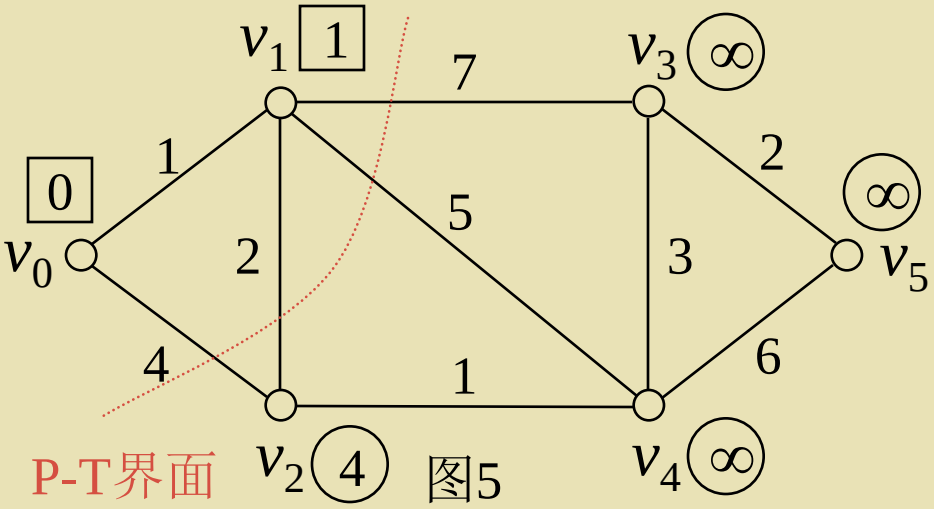
图4

修改圆标号(只修改P-T界面近前的结点v_1，v_2)：

v_1：∞，0+1=1；

v_2：∞，0+4=4；

离散数学



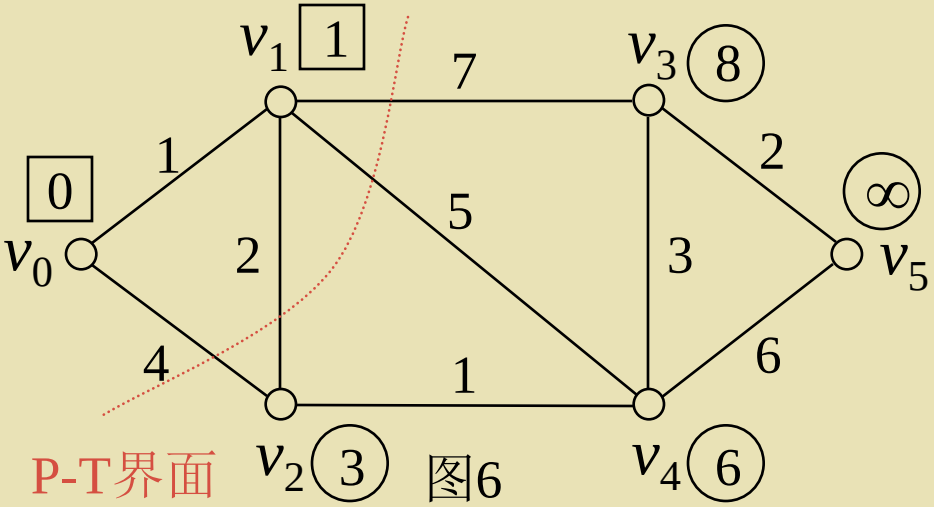
P-T界面 图5

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
1	4	∞

修改P-T界面；



P-T界面 图6

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_2：4,1+2=3；

v_3：∞,1+7=8；

v_4：∞,1+5=6；

离散数学

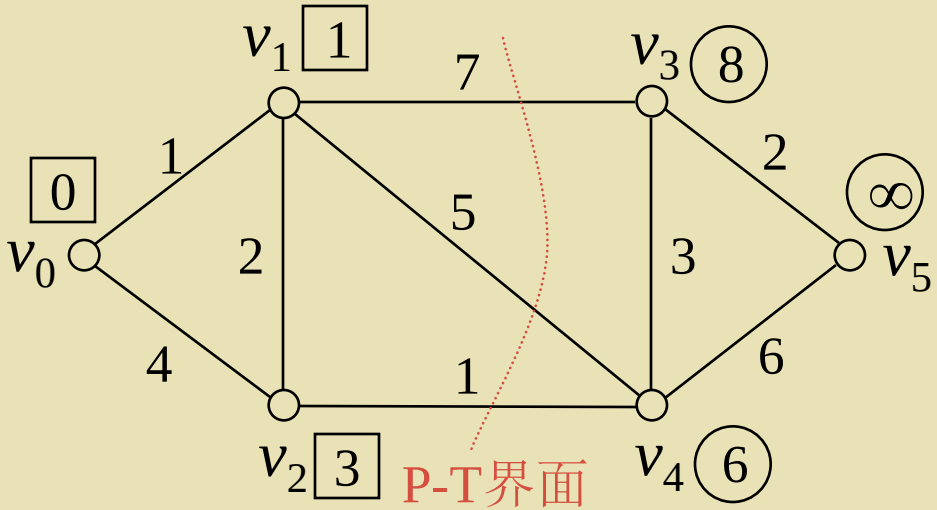


图7

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
3	8	6	∞

修改P-T界面；

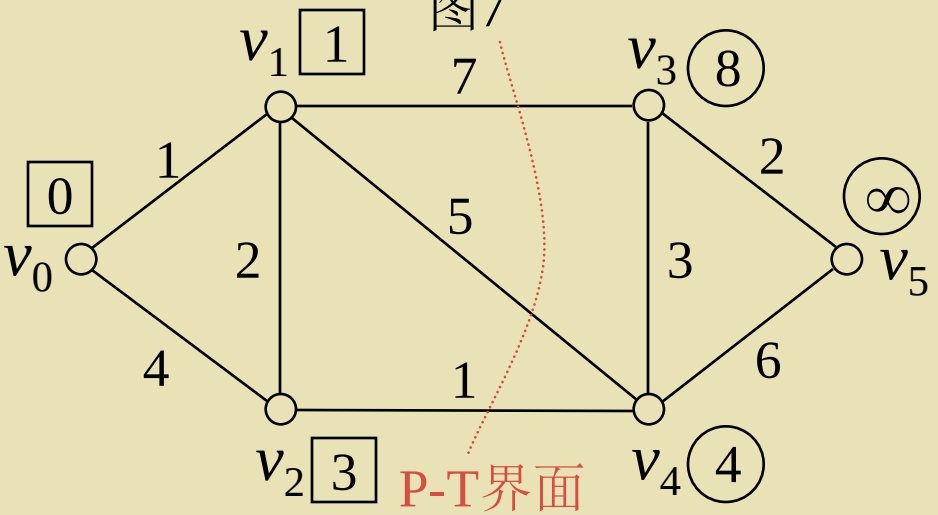


图8

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；

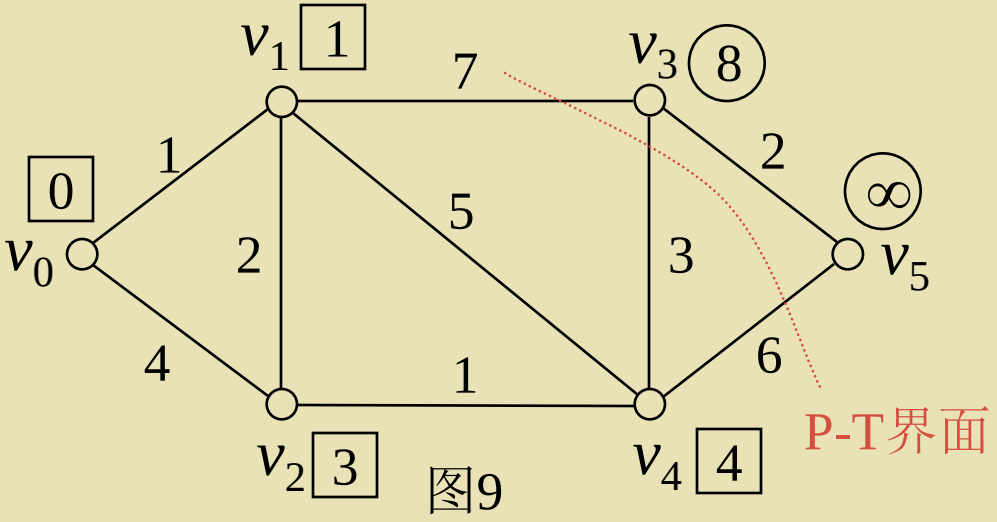


图9

改圆为方：

寻求T中圆标号最小的结点$t_0 = v_4$：

v_3	v_4	v_5
8	4	∞

修改P-T界面；

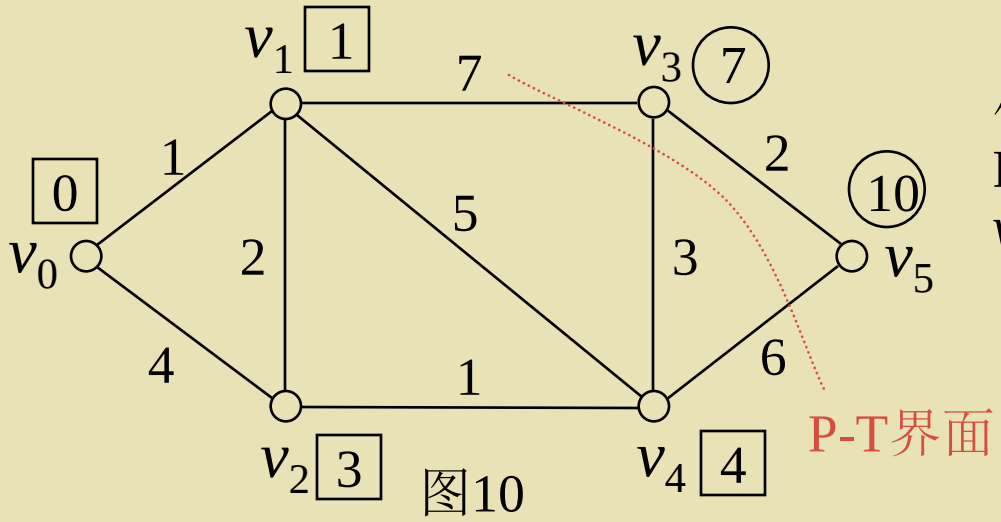


图10

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：∞,4+6=10；

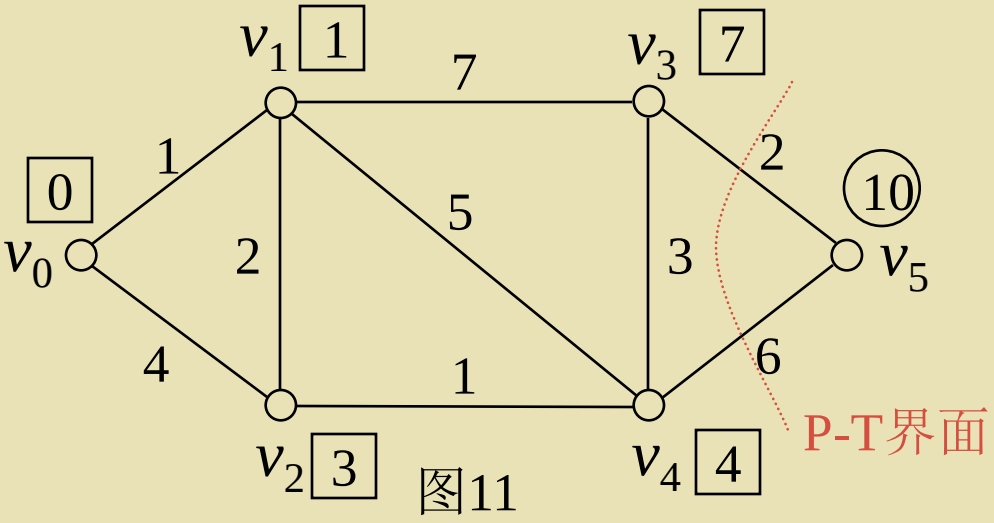


图11

改圆为方：

寻求T中圆标号最小的结点$t_0 = v_3$：

v_3	v_5
7	10

修改P-T界面；

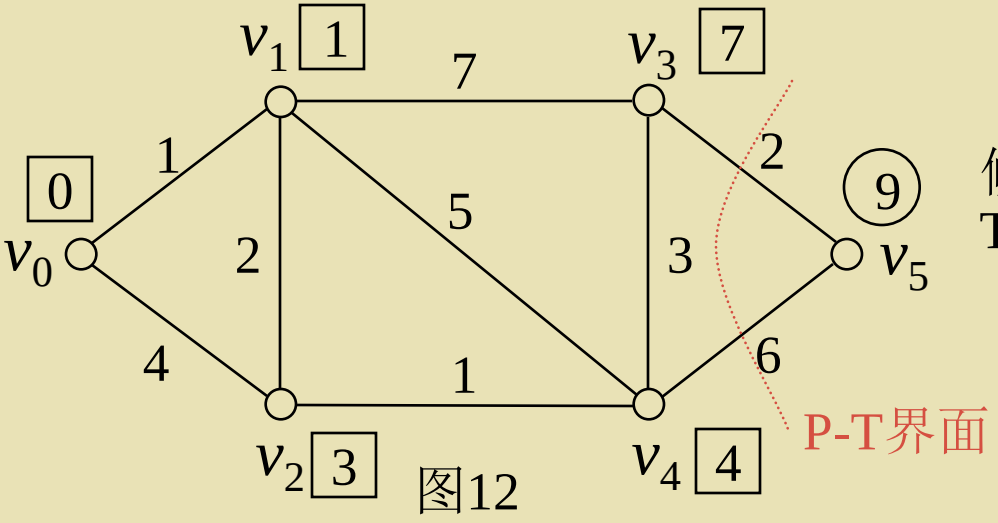


图12

修改圆标号（只修改离P-T界面最近的结点v_5）：

v_5：10,7+2=9；

离散数学

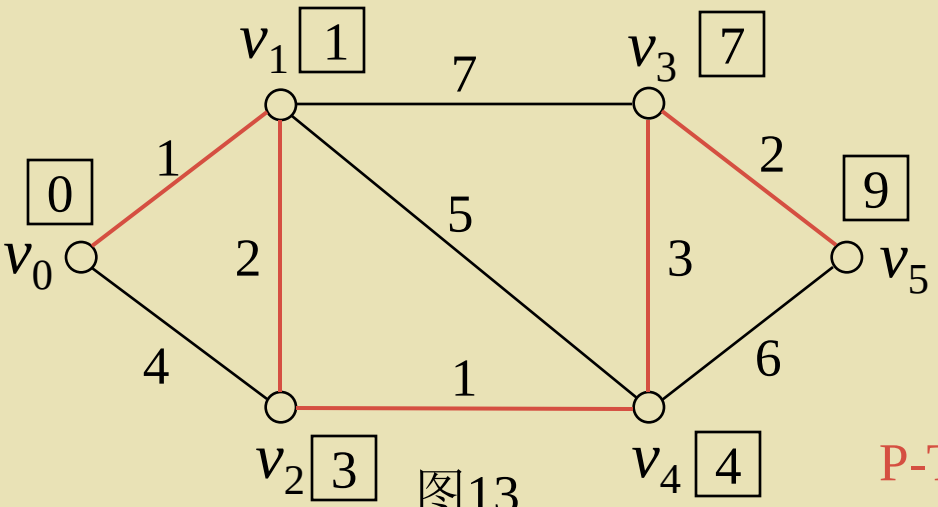


图13

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

P-T界面

于是，从结点v_0到结点v_5间的最短路为

$P_0 = (v_0, v_1, v_2, v_4, v_3, v_5)$　　(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)程序计算法：

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

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。我们可以将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；

•最好是两法结合着用。

◆***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；(参见本章习题17)。

***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$|P|=k$，$|T|=n-k(0\le k\le n-1)$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。

离散数学

- 第八章　图论

  前五节到此

  已经结束！