数据结构与算法

计算机学院

朱晨阳 副教授

zhuchenyang07@nudt.edu.cn

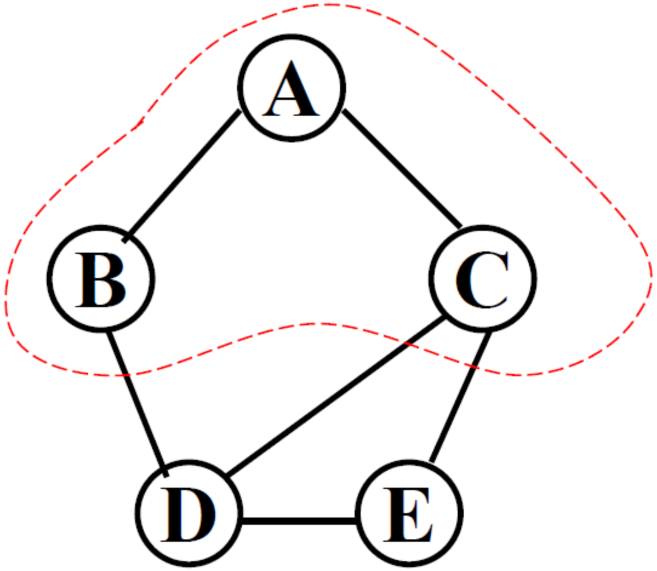
图

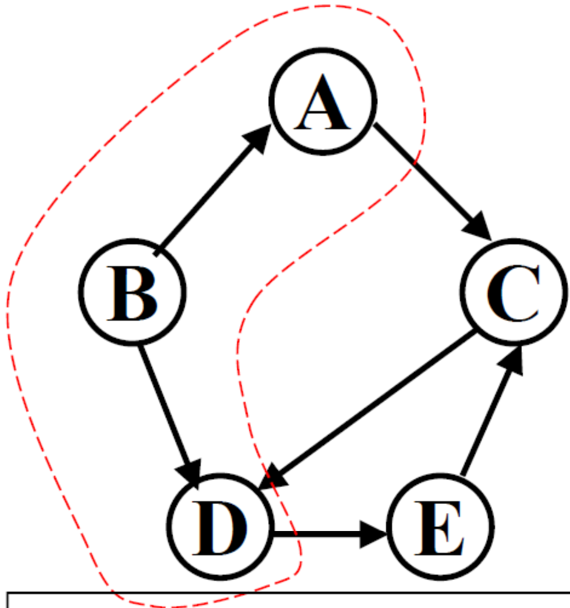
- 存储：邻接矩阵、邻接表、邻接多重表
- 遍历：宽度优先遍历、深度优先遍历
- 最小生成树、Prim算法、Kruskal算法
- 单源最短路径、Dijkstra算法
- 每对顶点间最短路径、Floyd算法
- AOV拓扑排序、AOE关键路径

什么是图?

- 图的任意结点可以有多个前驱和多个后继
- 图G=(V, E)由非空有穷顶点集V和V上的顶点对构成的边集E组成。
  - 如果E中任意顶点对是无序的，则G是无向图
  - 如果E中任意顶点对是有序的，则G是有向图
  - (v1,v2)代表无向边，(v1,v2)和(v2,v1)相同
  - <v1,v2>代表有向边，<v1,v2>和<v2,v1>不同
- 若G1=(V1,E1),G2=(V2,E2)是两个图，且V2⊆V1，E2 ⊆ E1，则称G2是G1的子图。

图





G=(V, E)

V={A,B,C,D,E}

E={(A,B),(A,C),(B,D),(C,D),(C,E),(D,E)}

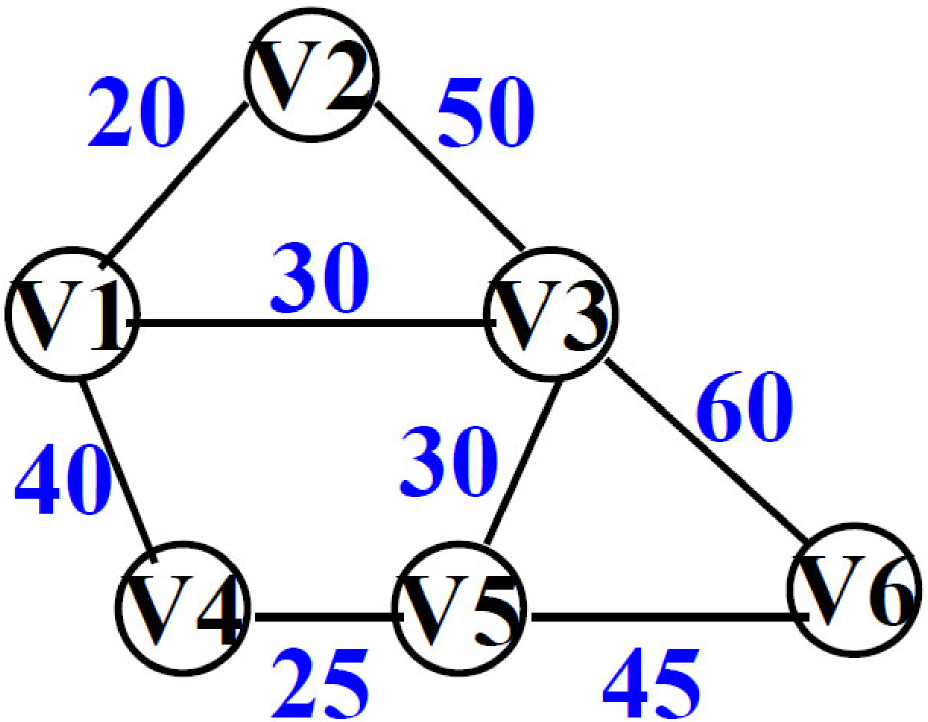
G=(V, E)

V={A,B,C,D,E}

E={<B,A>,<A,C>,<B,D>,<C,D>,<E,C>,<D,E>}

加权图

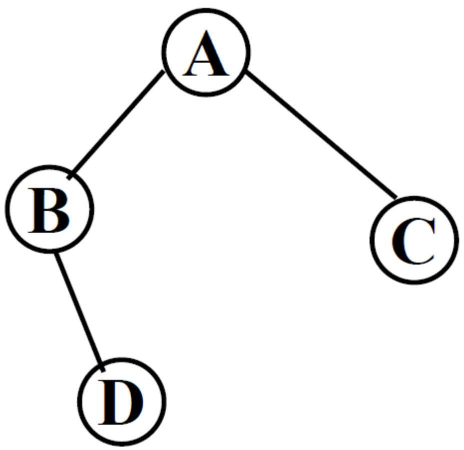
- 加权图：图的每条边对应一个权值。权可以代表费用、时间长度、道路长度等。

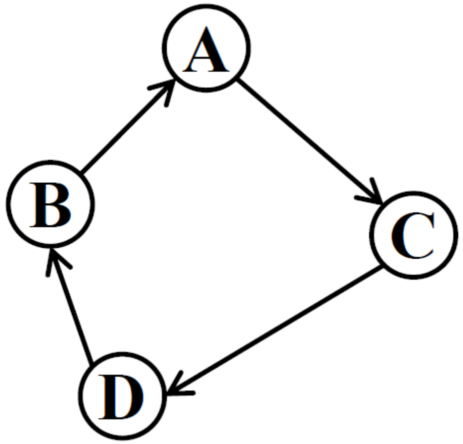


相关联

- 设G=(V, E)是无向图，若边(V1,V2)∈E，则称V1和V2是相邻顶点，边(V1,V2)是与顶点V1和V2相关联的边。

- 设G =(V, E)是有向图，若边<V1,V2>∈E，则称顶点V1邻接到顶点V2，顶点V2邻接于顶点V1，边<V1,V2>是与顶点V1和V2相关联的边。





度

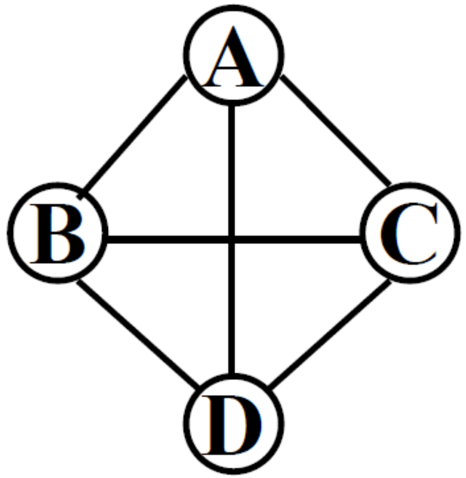
- 无向图中，与顶点V1相关联的边数称为V1的度。
- 有向图中，以顶点V1为始点并与V1相关联的边数，称为V1的出度；以顶点V1为终点并与V1相关联的边数称为V1的入度；V1的出度与入度的和称为V1的度。
- 有向图中出度为0的顶点称为终端顶点(叶子)。
- 设图G有n个顶点，t条边，则所有顶点的度数之和为2t。

简单图

- 简单无向图：任意两个顶点之间最多一条边，且不含自回路(顶点到其自身的边)。
- 简单有向图：任意两个顶点之间最多两条方向相反的边，且不含自回路。
- n个顶点的简单无向图，最大边数为n(n-1)/2。简单有向图最大边数为n(n-1)

完全图

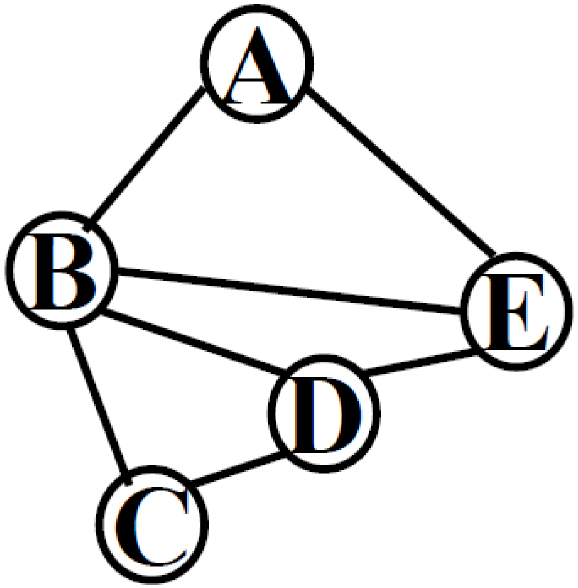
- 任意两个顶点间都有一条边的简单无向图称为无向完全图。含有n个顶点的无向完全图有n(n-1)/2条边。
- 任意两个顶点间都有方向相反的两条边的简单有向图称为有向完全图。含有n个顶点的有向完全图有n(n-1)条边。



路径

- 在有向(或无向)图中，无重复边且相邻边前后衔接的边序列<v1,v2>,<v2,v3>,...,<vn-1,vn>(或(v1,v2),(v2,v3),...,(vn-1,vn))称为从v1到vn的一条路径，序列中的边数称为路径的长度。
- 若除了v1和vn可以相同外，路径上所有顶点各不相同，则称该路径为简单路径，v1=vn的简单路径称为回路或环。

路径



是否是路径？

(A,B)(B,C)(C,D)(D,E)

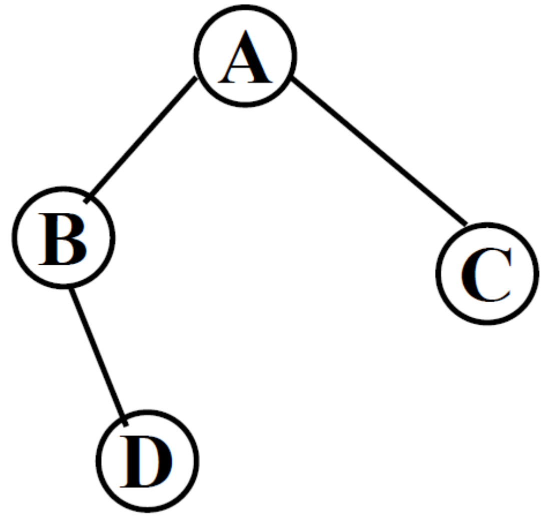
(A,B)(B,C)(C,D)(D,E)(E,D)

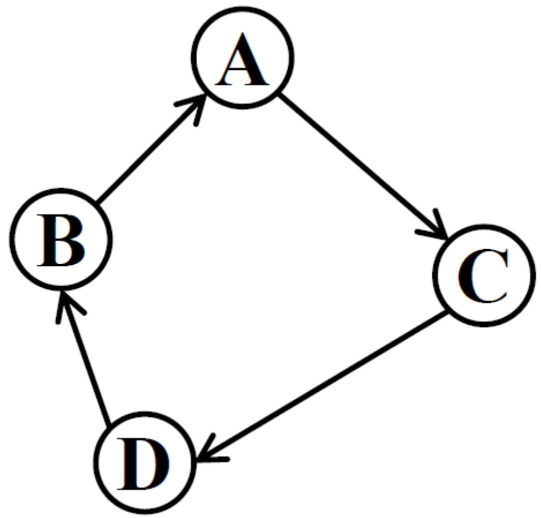
(A,B)(B,C)(C,D)(D,B)(B,C)(C,D)(D,E)

(A,B)(B,C)(C,D)(D,B)(B,E)

连通图/强连通图

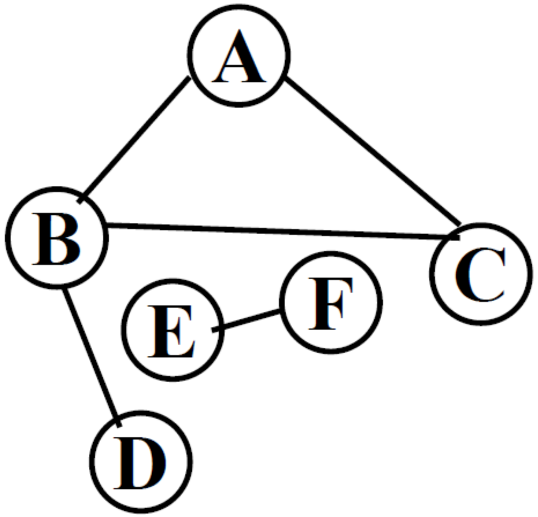
- 在无向图G中，若任意两个顶点Vi和Vj之间有路径，则称G是连通图。
- 在有向图G中，若任意两个顶点Vi和Vj，存在Vi到Vj的路径，以及Vj到Vi的路径，则称G是强连通图。

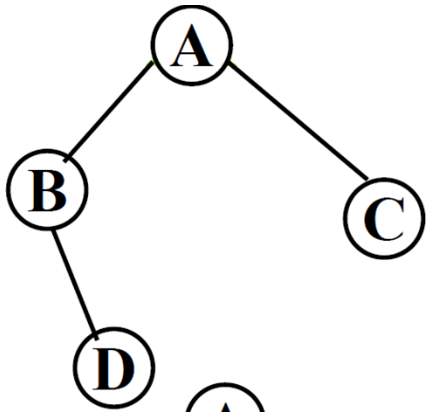




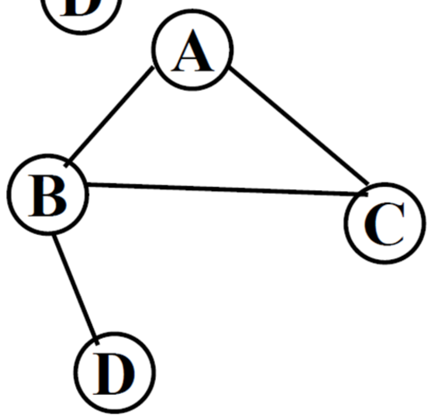
连通分量

- 无向图的极大连通子图称为连通分量

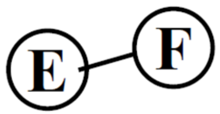




No



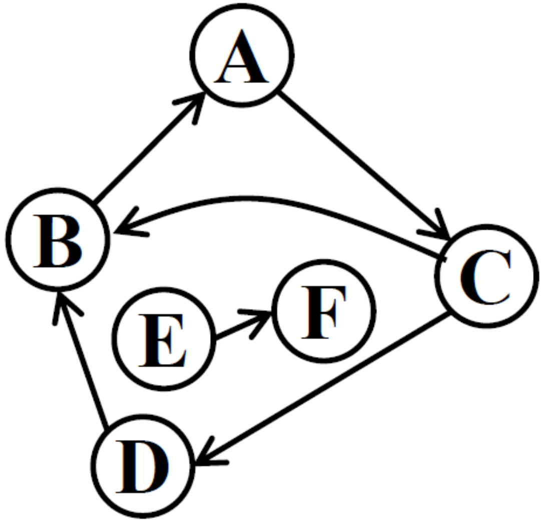
Yes

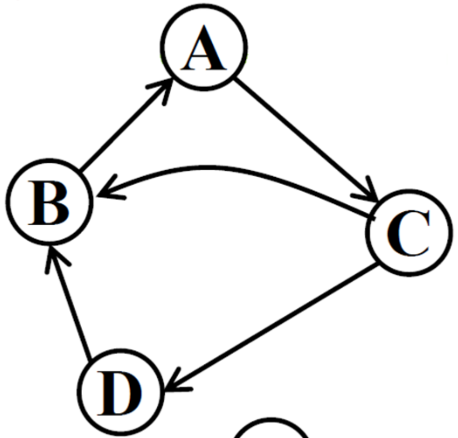


Yes

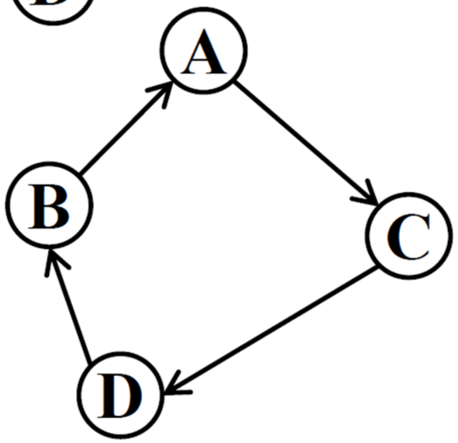
强连通分量

- 有向图的极大强连通子图称为强连通分量

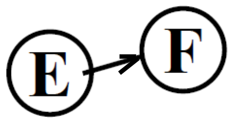




Yes



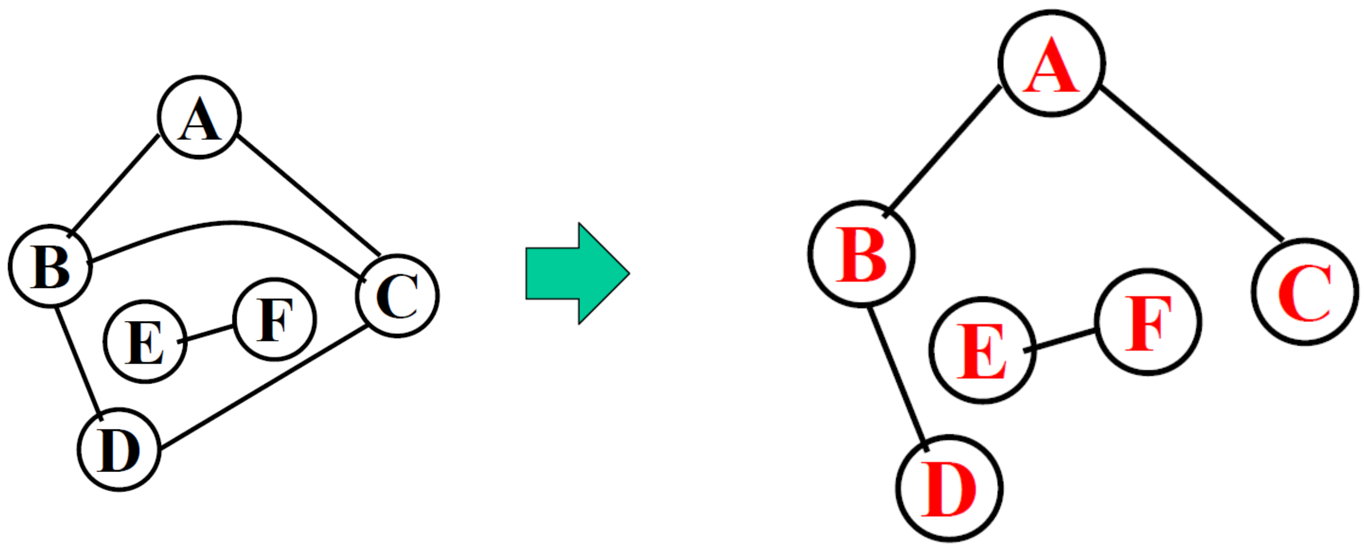
No



No

生成树

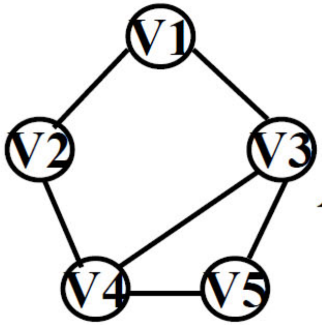
- 含有n个顶点的连通无向图G的生成树是G的一个含有全部n个顶点和n-1条边的连通图。(即含有全部n个顶点的极小连通子图)
- 有m个连通分量的无向图的每个连通分量都有一棵生成树，构成图的生成树林。



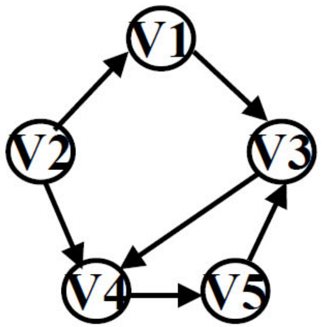
用相邻矩阵表示图

- 设图G = (V,E)有n个顶点和m条边，$V=\{V_1,V_2,\ldots,V_n\}$，则G的相邻矩阵A_{nxn}的元素a_{ij}定义为

$$a_{ij}=\begin{cases}1 & (V_i,V_j)\in \mathbf{E}(\text{或}<V_i,V_j>\in E)\\ 0 & \text{else}\end{cases}$$



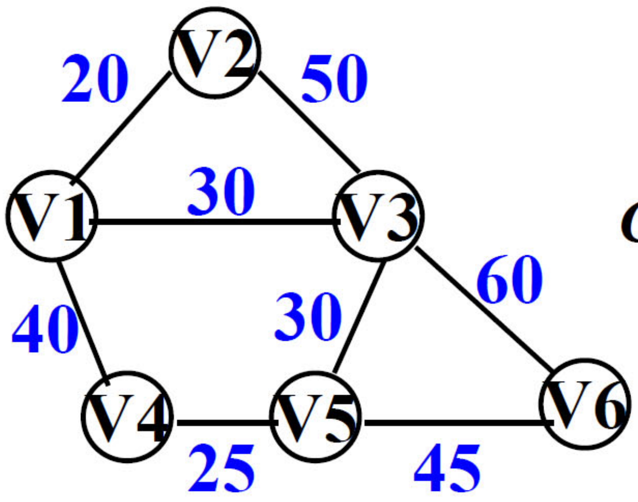
$$A=\begin{bmatrix}0&1&1&0&0\\1&0&0&1&0\\1&0&0&1&1\\0&1&1&0&1\\0&0&1&1&0\end{bmatrix}$$



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

相邻矩阵

- 加权图的相邻矩阵



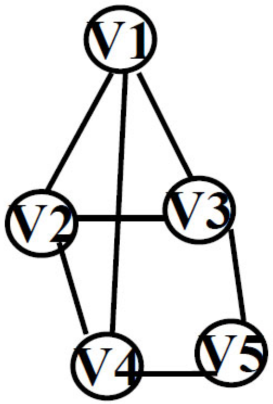
$$
C=\begin{bmatrix}
\infty & 20 & 30 & 40 & \infty & \infty \\
20 & \infty & 50 & \infty & \infty & \infty \\
30 & 50 & \infty & \infty & 30 & 60 \\
40 & \infty & \infty & \infty & 25 & \infty \\
\infty & \infty & 30 & 25 & \infty & 45 \\
\infty & \infty & 60 & \infty & 45 & \infty
\end{bmatrix}
$$

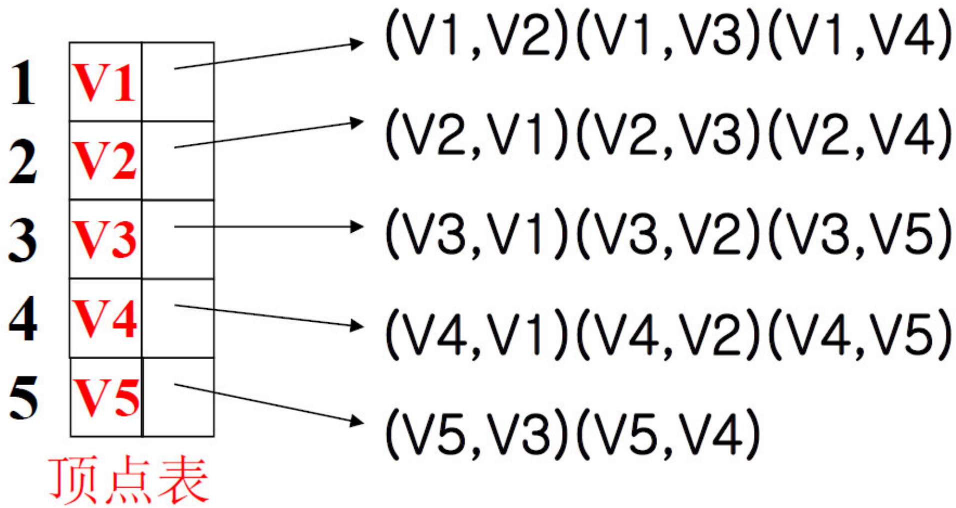
相邻矩阵

- 用相邻矩阵表示图
- 需要一个顺序表存储n个顶点
- 需要一个nxn的矩阵存储边
  - 对于有向图，需要n^2个单元
  - 对于无向图，只需存储矩阵的上三角或下三角,只需n(n-1)/2个单元

邻接表

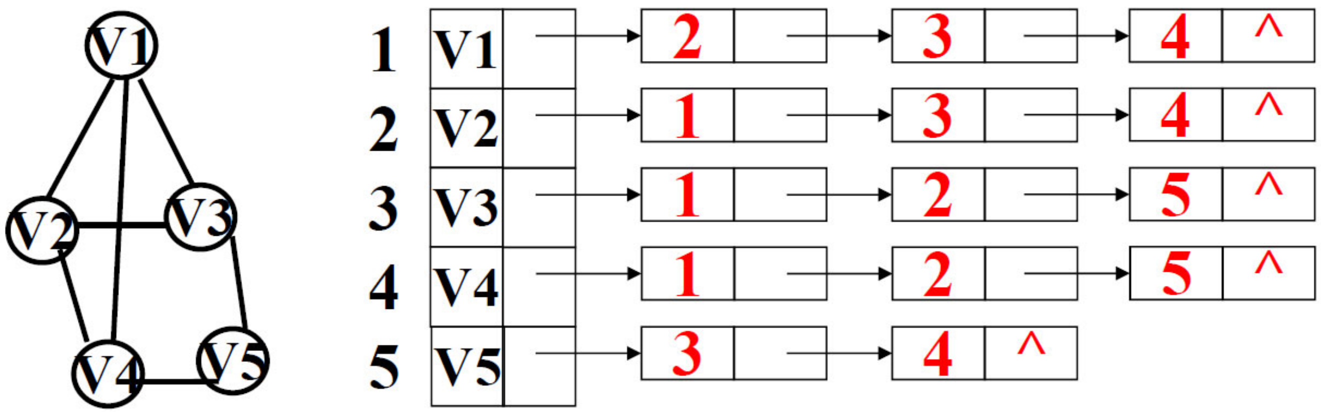
- 邻接表由顶点表和边表组成。
  - 无向图：对于每个顶点，将与顶点相关联的边组织一个链表，称为边表。顶点表的每个表项对应一个顶点，保存与该顶点相关联的边表的表头指针。





邻接表

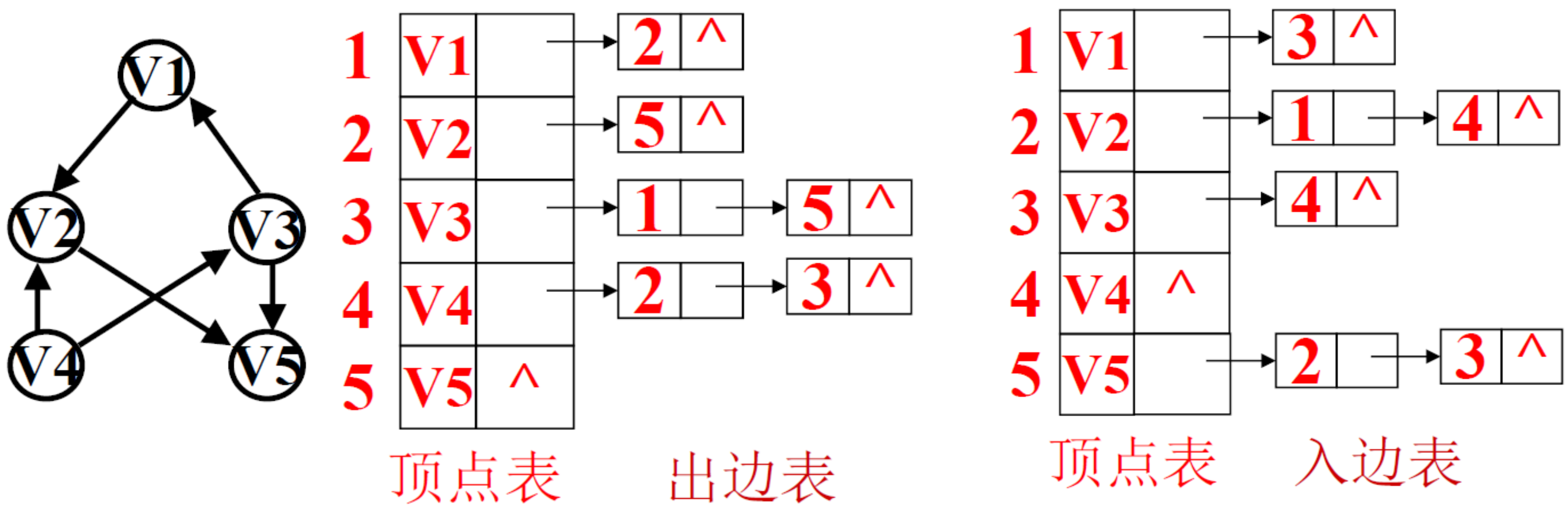
- 邻接表由顶点表和边表组成。
  - 无向图：对于每个顶点，将与顶点相关联的边组织一个链表，称为边表。顶点表的每个表项对应一个顶点，保存与该顶点相关联的边表的表头指针。



顶点表　　边表

邻接表

- 邻接表由顶点表和边表组成。
  - 有向图：可以将与顶点相关联的出边组织成链表，作为与该顶点相关联的边表，称为出边表；或者将所有入边组织成链表，称为入边表。

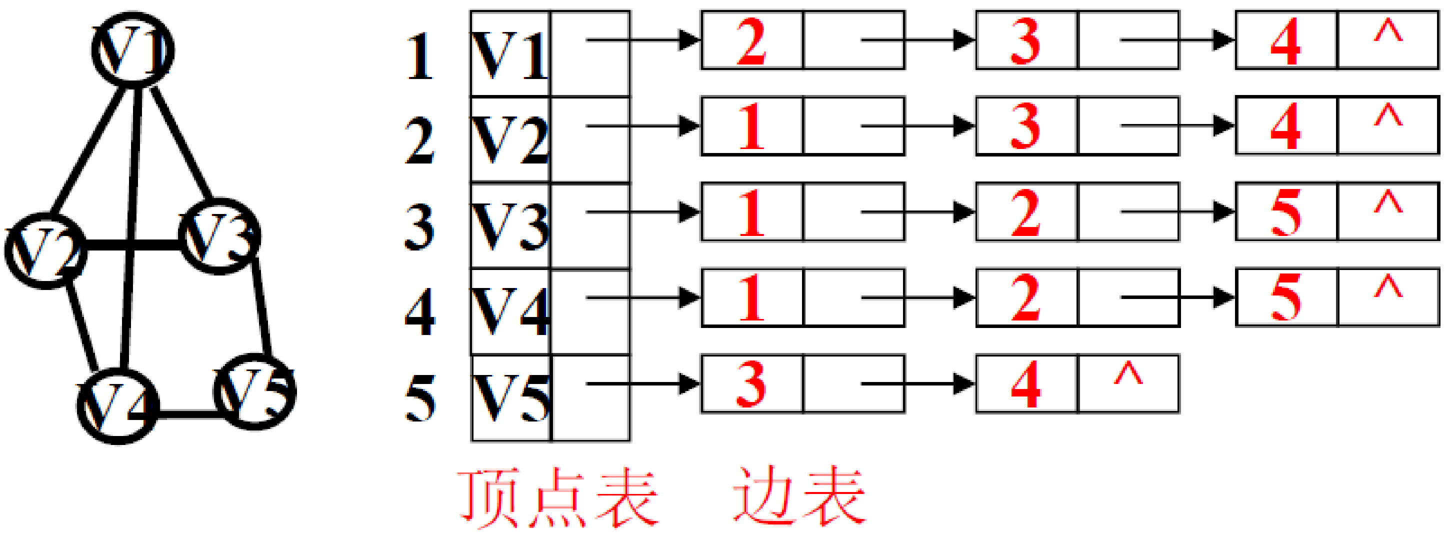


邻接表

- 邻接表表示加权图(网)，只需在边表的每个结点加上一个表示权的字段。
- 邻接表存储具有n个顶点、m条边的图
  - 顶点表n个表项
  - 无向图的所有边表共有2m个表项
  - 有向图的每个边表有m个表项
- 邻接表表示无向图时，每条边对应2个边表结点。

邻接表

- 邻接表表示无向图时，每条边对应两个边表结点，不利于插入、删除等操作。如何修改邻接表，使得每条边只对应一个边表结点？



邻接表多重表

- 邻接多重表由顶点表和边表组成，每条边只对应一个边表结点。
- 顶点表结点包含data域(保存顶点信息)和edge域(指向与该顶点相关联的第一条边)

邻接表多重表

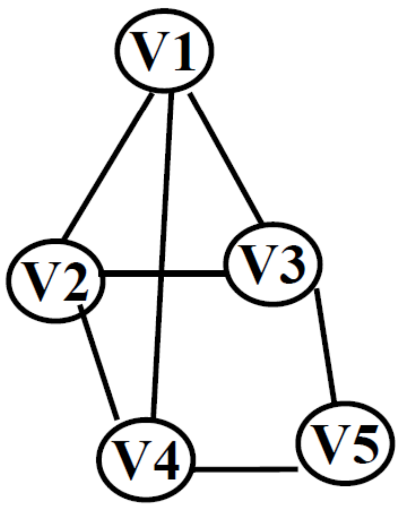
- **边表结点**包含5个域

  mark ：边访问标记

  i,j ： 边(Vi,Vj)的两个顶点的标号

  ilink：指向与Vi相关联的下一条边

  jlink：指向与Vj相关联的下一条边



		edge
1	V1	N1
2	V2	N1
3	V3	N2
4	V4	N3
5	V5	N6

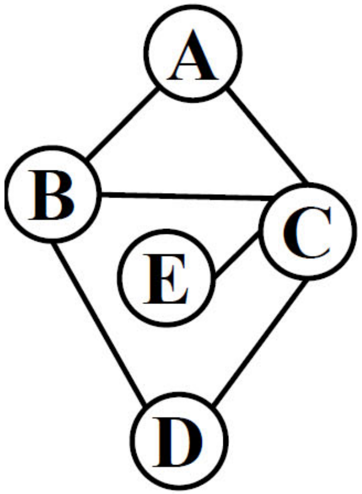
顶点表

		i	ilink	j	jlink
N1		1	N2	2	N4
N2		1	N3	3	N4
N3		1	^	4	N5
N4		2	N5	3	N6
N5		2	^	4	N7
N6		3	^	5	N7
N7		4	^	5	^

边表

邻接表多重表

- 练习



		edge
0	A	
1	B	
2	C	
3	D	
4	E	

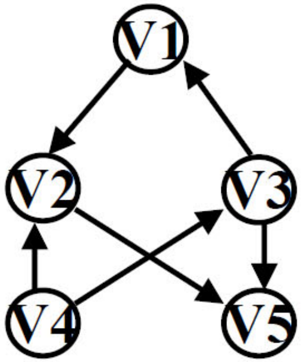
		i	ilink	j	jlink
N1		0		1	
N2		0		2	
N3		2		1	
N4		3		1	
N5		3		2	
N6		4		2	

邻接多重表

- 邻接表表示有向图时只保存顶点的入边表或出边表。有向图的邻接多重表同时能表示顶点的出边和入边，而不增加边表结点数。
- 有向图的邻接多重表包含顶点表和边表，顶点表的每个表项包含3个域
  - data: 表示顶点信息
  - edge1: 指向以该顶点为始点的边表的第一条边
  - edge2: 指向以该顶点为终点的边表的第一条边

邻接多重表

边表结点结构与无向图的邻接多重表相同，但ilink指向以Vi为始点的下一条边，jlink指向以Vj为终点的下一条边。



		edge1	edge2
1	V1	N1	N3
2	V2	N2	N1
3	V3	N3	N6
4	V4	N5	^
5	V5	^	N2

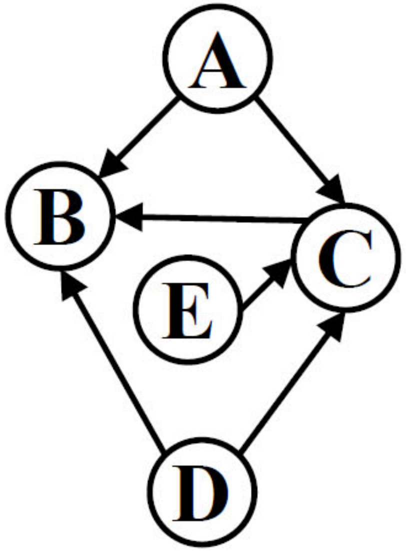
顶点表

		i	ilink	j	jlink
N1		1	^	2	N5
N2		2	^	5	N4
N3		3	N4	1	^
N4		3	^	5	^
N5		4	N6	2	^
N6		4	^	3	^

边表

邻接多重表

- 练习



		edge1	edge2
0	A		
1	B		
2	C		
3	D		
4	E		

		i	ilink	j	jlink
N1		0		1	
N2		0		2	
N3		2		1	
N4		3		1	
N5		3		2	
N6		4		2	