

第5章图及其相关算法

5.1 图的基本概念	5.6 有向图的搜索
5.2 图的存储表示	5.7 强连通图*
5.3 图的搜索(遍历)	5.8 拓扑分类
5.4 图与树的联系	5.9 关键路径
5.5 无向图的双连通性*	5.10 最短路径

图的搜索算法是有关图问题的重要算法！

哈尔滨工业大学 计算机科学与技术学院

5.1 图的基本概念

定义图

图是由顶点集合(**vertex**)及顶点间的关系集合组成的一种数据结构：
 $\text{Graph} = (\mathbf{V}, \mathbf{E})$

其中 $\mathbf{V} = \{x \mid x \in \text{某个数据对象}\}$ 是顶点的有穷非空集合；
 $\mathbf{E} = \{(x, y) \mid x, y \in \mathbf{V}\}$ 是顶点之间关系的有穷集合，也叫做边(**edge**)集合。

有向图 若图G的每条边都有方向，则称G为有向图
(**Digraph**)。在有向图中，有向边(也称弧)都是顶点的有序对 $\langle x, y \rangle$ 。

无向图 若图G的每条边都是没有方向的，则称G为无向图
(**Undigraph**)。在无向图中，每条边都是顶点的无序对 $\{x, y\}$ 。

5.1 图的基本概念

定义图

- 若 (v_i, v_j) 是一条无向边，则称 v_i 和 v_j 互为邻接点 (**Adjacent**)，或称 v_i 与 v_j 相邻接；并称 (v_i, v_j) 依附或关联 (**Incident**) 于 v_i 和 v_j ，或称 (v_i, v_j) 与 v_i 和 v_j 相关联。
- 若 $\langle v_i, v_j \rangle$ 是一条有向边，则称 v_i 邻接到 v_j ，或称 v_j 邻接于 v_i ；并称边 $\langle v_i, v_j \rangle$ 关联于 v_i 和 v_j ，或称 $\langle v_i, v_j \rangle$ 与 v_i 和 v_j 相关联。
- 若将图中的每条边都赋予一个权，则称这种带权的图为网络 (**Network**)。通常权是一个具有某种意义的数（如表示两顶点间的距离、耗费等）。

哈尔滨工业大学 计算机科学与技术学院

5.1 图的基本概念

定义 结点的度

- 无向图中顶点 v 的度 (**Degree**) 是关联于该顶点的边的数目，或与该顶点相邻的顶点数目，记为 $D(v)$ 。
- 若G是有向图，则把邻接到顶点 v 的顶点数目或边数目称为顶点 v 的入度 (**Indegree**)，记为 $ID(v)$ ；把邻接于顶点 v 的顶点数目或边数目称为顶点 v 的出度 (**Outdegree**)，记为 $OD(v)$ ；顶点 v 的度则定义为该顶点的入度和出度之和，即 $D(v) = ID(v) + OD(v)$ 。
- 无论是有向图还是无向图，顶点数 n 、边数 e 和度数之间的关系为：

$$e = \frac{1}{2} \sum_{i=0}^n D(v_i)$$

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 5

5.1 图的基本概念

定义 路（径）、路径长、简单路、简单环路

- 在无向图G中，若存在一个顶点序列 $v_p, v_{i1}, v_{i2}, \dots, v_{im}, v_q$ ，使得 $(v_p, v_{i1}), (v_{i1}, v_{i2}), \dots, (v_{im}, v_q) \in E(G)$ ，则称顶点 v_p 路到 v_q 有一条路径（Path）。
- 在有向图G中，若存在一个顶点序列 $v_p, v_{i1}, v_{i2}, \dots, v_{im}, v_q$ ，使得有向边 $<v_p, v_{i1}>, <v_{i1}, v_{i2}>, \dots, <v_{im}, v_q> \in E(G)$ ，则称顶点 v_p 路到 v_q 有一条有向路径（Path）。
- 非带权图的路径长度是指此路径上边的条数。
- 带权图的路径长度是指路径上各边的权之和。
- 简单路径 若路径上各顶点 v_1, v_2, \dots, v_m 均不互相同，则称这样的路径为简单路径。
- 简单回路 若路径上第一个顶点 v_1 与最后一个顶点 v_m 重合，则称这样的简单路径为回路或环。

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 6

5.1 图的基本概念

定义 图的连通性

连通图与连通分量

- 在无向图中，若从顶点 v_i 到顶点 v_j 有路径，则称顶点 v_i 与 v_j 是连通的。
- 如果图中任意一对顶点都是连通的，则称此图是连通图。
- 非连通图的极大连通子图叫做连通分量。

强连通图与强连通分量

- 在有向图中，若对于每一对顶点 v_i 和 v_j ，都存在一条从 v_i 到 v_j 和从 v_j 到 v_i 的路径，则称此图是强连通图。
- 非强连通图的极大强连通子图叫做强连通分量。

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 7

5.1 图的基本概念

定义 在上的基本操作

设图 $G=(V,E)$ ，图上定义的基本操作如下：

```

node NEWNODE ( G )
void DELNODE ( G, v )
void SETSUCC ( G, v1, v2 )
void DELSUCC ( G, v1, v2 )
listofnode SUCC ( G, v1, v2 )
listofnode PRED ( G, v )
int ISEEDGE ( G, v1, v2 )
node FirstAdjVex( G, v ): 顶点v的第一个邻接顶点
node NextAdjVex( G, v, w ): 顶点v的某个邻接点w的下一个
邻接顶点。

```

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 8

5.2 图的存储表示

5.2.1 邻接矩阵 (Adjacency Matrix)

一、图的邻接矩阵表示

- 在图的邻接矩阵表示中，有一个记录各个顶点信息的顶点表，还有一个表示各个顶点之间关系的邻接矩阵。
- 设图 $A = (V, E)$ 是一个有 n 个顶点的图，图的邻接矩阵是一个二维数组 $A.edge[n][n]$ ，定义：

$$A.edge[i][j] = \begin{cases} 1 & \text{若 } i, j \in E \text{ 或 } (i, j) \in E \\ 0 & \text{否则} \end{cases}$$

图示：

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

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

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 9
<ul style="list-style-type: none"> 无向图的邻接矩阵是对称的； 有向图的邻接矩阵可能是不对称的。 在有向图中，统计第<i>i</i>行1的个数可得顶点<i>i</i>的出度，统计第<i>j</i>列1的个数可得顶点<i>j</i>的入度。 在无向图中，统计第<i>i</i>行(列)1的个数可得顶点<i>i</i>的度。 <p>二、网络的邻接矩阵</p> <p>$A.edge[i][j] = \begin{cases} W(i, j), & \text{若 } i \neq j \text{ 且 } (i, j) \in E \text{ 或 } (j, i) \in E \\ \infty, & \text{若 } i \neq j \text{ 且 } (i, j) \notin E \text{ 或 } (j, i) \notin E \\ 0, & \text{若 } i = j \end{cases}$</p> $A.edge[i][j] = \begin{bmatrix} 0 & 1 & \infty & 4 \\ \infty & 0 & 9 & 2 \\ 3 & 5 & 0 & 8 \\ \infty & \infty & 6 & 0 \end{bmatrix}$		

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 10
<p>三、用邻接矩阵表示的图结构的定义</p> <pre>#define MaxValue Int_Max //在<limits.h>中 #define NumEdges 50; //边条数 #define NumVertices 10; //顶点个数 typedef char VertexData; //顶点数据类型 typedef int EdgeData; //边上权值类型 typedef struct { VertexData vexlist[NumVertices]; //顶点表 EdgeData edge[NumVertices][NumVertices]; //邻接矩阵=边表, 可视为边之间的关系 int n, e; //图中当前的顶点个数与边数 } MTGraph;</pre>		

哈尔滨工业大学 计算机科学与技术学院

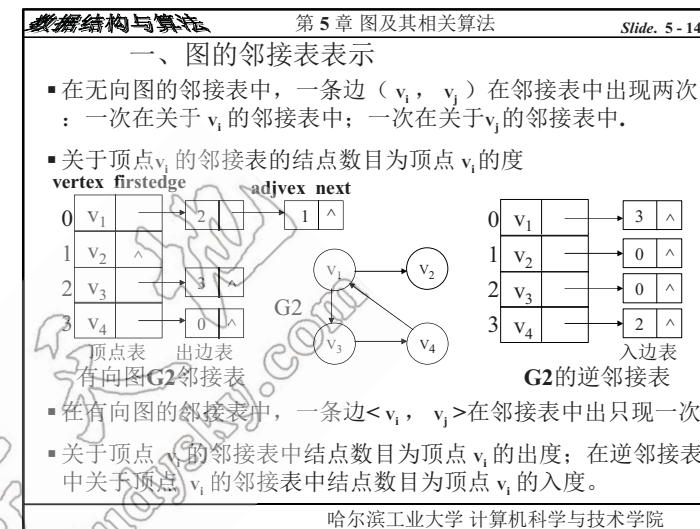
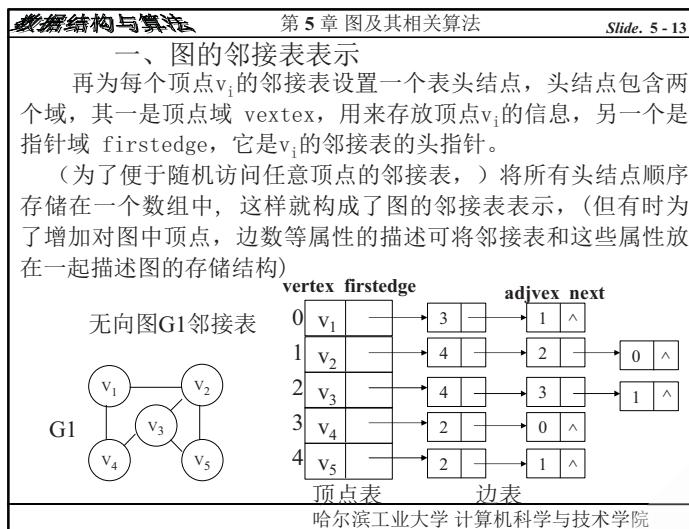
数据结构与算法	第5章 图及其相关算法	Slide. 5 - 11
<p>四、图的邻接矩阵构造算法</p> <pre>void CreateMGraph(MTGraph *G) //建立(无向)图的邻接矩阵 { int i, j, k, w; scanf("%d,%d",&G->n,&G->e); //1.输入顶点数和边数 for (i=0; i<G->n; i++) //2.读入顶点信息, 建立顶点表 G->vexlist[i]=getchar(); for (i=0; i<G->n; i++) for (j=0;j<G->n;j++) G->edge[i][j]=0; //3.邻接矩阵初始化 for (k=0; k<G->e; k++) { //4.读入e条边建立邻接矩阵 scanf("%d %d %d",&i,&j,&w); //输入边 (i,j) 上的权w G->edge[i][j]=w; G->edge[j][i]=w; } //空间复杂性: S=O(n+n^2) } //时间复杂性: T=O(n+n^2+e)。当e < n, T=O(n^2)</pre>		

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 12																								
<h3>5.2.2 邻接表 (Adjacency List)</h3> <p>一、图的邻接表表示</p> <p>对于G中的每个顶点v_i，把所有邻接(于)v_i的顶点v_j连成一个单链表(称为关于v_i的邻接表)。邻接表中每个表结点都有两个域：其一是邻接点域 $adjvex$，用以存放与v_i相邻顶点的序号；其二是链域 $next$，用来将邻接表的所有表点链在一起；另外若要表示边上的信息如(权值)，则在表结点中还应增加一个数据域$cost$。</p> <table border="1"> <thead> <tr> <th>vertex</th> <th>firstedge</th> <th>adjvex</th> <th>next</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>v_1</td> <td>3</td> <td>1 ^</td> </tr> <tr> <td>1</td> <td>v_2</td> <td>4</td> <td>2 ^</td> </tr> <tr> <td>2</td> <td>v_3</td> <td>4</td> <td>3 ^</td> </tr> <tr> <td>3</td> <td>v_4</td> <td>2</td> <td>0 ^</td> </tr> <tr> <td>4</td> <td>v_5</td> <td>2</td> <td>1 ^</td> </tr> </tbody> </table> <p>无向图G1邻接表</p> <p>G1</p> <p>顶点表 边表</p>			vertex	firstedge	adjvex	next	0	v_1	3	1 ^	1	v_2	4	2 ^	2	v_3	4	3 ^	3	v_4	2	0 ^	4	v_5	2	1 ^
vertex	firstedge	adjvex	next																							
0	v_1	3	1 ^																							
1	v_2	4	2 ^																							
2	v_3	4	3 ^																							
3	v_4	2	0 ^																							
4	v_5	2	1 ^																							

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126



数据结构与算法 第5章 图及其相关算法 Slide. 5 - 15

二、用邻接表表示的图结构的定义

```
#define NumVertices 10; //顶点个数
typedef char VertexData; //顶点数据类型
typedef int EdgeData; //边上权值类型
typedef struct node { //边表结点
    int adjvex; //邻接点域 (下标)
    EdgeData cost; //边上的权值
    struct node *next; //下一边链接指针
} EdgeNode;
typedef struct { //顶点表结点
    VertexData vertex; //顶点数据域
    EdgeNode *firstedge; //边链表头指针
} VertexNode;
typedef struct { //图的邻接表
    VertexNode vexlist [NumVertices];
    int n, e; //图中当前的顶点个数与边数
} AdjGraph;
```

表结点
头结点
vertex firstedge

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 16

三、邻接表的构造算法

```
void CreateGraph (AdjGraph G)
{ cin >> G.n >> G.e; //1.输入顶点个数和边数
for ( int i = 0; i < G.n; i++ ) { //2.建立顶点表
    cin >> G.vexlist[i].vertex; //2.1输入顶点信息
    G.vexlist[i].firstedge = NULL; } //2.2边表置为空表
for ( i = 0; i < G.e; i++ ) { //3.逐条边输入,建立边表
    cin >> tail >> head >> weight; //3.1输入(变量说明省了)
    EdgeNode * p = new EdgeNode; //3.2建立边结点
    p->adjvex = head; p->cost = weight; //3.3设置边结点
    p->next = G.vexlist[tail].firstedge; //3.4链入第tail号链表的前端
    G.vexlist[tail].firstedge = p; //空间复杂度: 有向图: S=O(e+n) 无向图: S= O(2e+n)
    p = new EdgeNode; //当e<<n^2时, S= O(n)
    p->adjvex = tail; p->cost = weight;
    p->next = G.vexlist[head].firstedge; //链入第head号链表的前端
    G.vexlist[head].firstedge = p; }
} //时间复杂度: O(2e+n)
```

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 17

四、两种存储结构的比较

- 在邻接矩阵中求边的数目 e ，必须检查整个矩阵，所耗时间是 $O(n^2)$ ，与边的个数 e 无关；而在邻接表中求边的数目 e ，只要对每个边表的结点个数进行计数即可求得 e ，所耗时间是 $O(e+n)$ 。因此当 $e \ll n^2$ 时，采用邻接表更节省空间。
- 在邻接矩阵中，很容易判断 (v_i, v_j) 和 (v_i, v_k) 是否为图的一条边，只要判断矩阵中的第 i 行第 j 列是否为非零元素即可；但在邻接表中，须扫描第 i 个边表，最坏情况下消耗时间 $O(n)$ 。
- 空间复杂度

思考题（算法设计）

- 已知图的邻接矩阵表示，求其邻接表表示；或相反。
- 已知有向图的正邻接表表示，求其逆邻接表表示；或相反。
- 根据邻接表或邻接矩阵，求图的边数、顶点的（入度/出度）度。

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 18

5.3 图的搜索(遍历)

- 图的搜索：从图的某个顶点出发，访遍图中所有的顶点，且使每个顶点被访问一次且仅被访问一次，称为图的搜索（遍历）（Graph Traversal）。
- 有许多方法和策略，不同的方法和策略导致不同的搜索算法，因此图的搜索算法是有关图的算法的基础。

最重要的：深度优先搜索DFS(Depth-First search)
搜索算法：广度优先搜索BFS (Breadth-First search)

- 图中可能存在回路，且图的任一顶点都可能与其它顶点相连，所以在访问完某个顶点之后可能会沿着某些边又回到了曾经访问过的顶点。图的遍历比树形结构复杂。
- 为了避免重复访问，可设置一个标志顶点是否被访问过的辅助数组 `visited[i]`—初始状态为 0，在图的遍历过程中，一旦某一个顶点 i 被访问，就立即让 `visited[i]` 为 1，防止它被多次访问。

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 19

5.3.1 先深搜索与先深编号

一、先深搜索定义——类似于树的先根遍历

- 设给定的（无向、有向）图的初态是所有顶点都未访问过(new)，在 G 中任选一个顶点 v 为初始出发点（源点），则先深搜索可定义为：

- 首先，访问出发点 v ，并将其标记为“访问过”(old)；
- 然后，从 v 出发，依次考察和 v 相邻（邻接于 v ）的顶点 w ；若 w 未访问过(new)，则以 w 为新的出发点递归地进行先深搜索，直到图中所有和源点 v 有路径相通的顶点（亦称从源点可到达的顶点）均被访问为止；
- 若此时图中仍有未被访问过的顶点，则另选一个未访问过的新顶点作为新的搜索起点，重复上述过程，直到图中所有顶点都被访问过为止。

先深搜索示例

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 20

5.3.1 先深搜索与先深编号

无向图 G3
 $V_1, V_2, V_4, V_8, V_5, V_3, V_6, V_7$

有向图 G4
 $V_1, V_2, V_4, V_8, V_5, V_3, V_6, V_7$

- 先深搜索特点：是递归的定义，特点是尽可能对纵深方向上进行搜索，故称先深搜索或深度优先搜索。
- 先深搜索过程中，根据访问顺序给顶点进行的编号，称为先深编号。
- 先深搜索过程中，根据访问顺序得到的顶点序列，称为先深序列或DFS序列。
- 先深搜索结果不唯一，即图的DFS序列和先深编号不唯一。

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 21
5.3.1 先深搜索与先深编号		
二、对先深搜索过程的进一步分析 设 x 是当前被访问的顶点，在对 x 做访问标记后，选择一条从 x 出发的未被检测过的边 (x, y) ；若发现顶点 y 已访问过，则重新选择另一条从 x 出发的未检测过的边，否则沿着边 (x, y) 到达未访问过的顶点 y ，对 y 访问并标记已访问过；然后从 y 开始搜索，直到搜索完从 y 出发的所有路径，即访问完所有从 y 出发的可达的顶点之后，才回溯到顶点 x ，并且再选择一条从 y 出发的未曾检测过的边。上述过程直到从 x 出发的所有边都已检测过为止。此时，若 x 不是源点，则回溯到在 x 之前被访问的顶点；否则图中所有和源点有路径相通的顶点（即从源点可达的所有顶点）都已被访问过，若图是连通的，则遍历过程结束，否则继续选择一个尚未访问过的顶点作为新源点，进行新的搜索过程。		
哈尔滨工业大学 计算机科学与技术学院		

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 22
5.3.1 先深搜索与先深编号 三、先深搜索算法（算法5.1）		
输入： $L[v]$ 表示无向图 G 的关于 v 的邻接表 输出： 每个结点有先深编号的无向图 G 和树边集 T <pre> { T = ∅ ; /* 树边集开始为空 */ count = 1 ; /* 先深编号计数器 */ for (all v ∈ V) while (there exists a vertex v ∈ V marked [new]) search(v); void search(v) / 递归 { dfn[v] = count ; /* 对 v 编号 */ count = count + 1 ; mark v [old] ; /* 访问结点 v */ for (each vertex w ∈ L[v]) if (w is marked [new]) { add (v, w) to T ; /* (v, w) 是树边 */ search (w); /* 递归搜索 */ } } } </pre>		
哈尔滨工业大学 计算机科学与技术学院		

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 23
三、先深搜索算法		
<pre> typedef enum{FALSE,TRUE} Boolean; Boolean visited[NumVertices]; // 访问标记数组是全局变量 int dfn[NumVertices]; // 顶点的先深编号 void DFSTraverse (AdjGraph G) // 先深搜索一邻接表表示的图G；而以邻接矩阵表示G时，算法完全相同。 { int i , count = 1; for (int i = 0; i < G.n; i++) visited [i] = FALSE; // 标志数组初始化 for (int i = 0; i < G.n; i++) if (! visited[i]) DFSX (G, i); // 从顶点 i 出发开始搜索search() BFSX(G, i) } </pre>		
哈尔滨工业大学 计算机科学与技术学院		

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 24
<pre> void DFS1 (AdjGraph* G, int i) // 以 vi 为出发点时对邻接表表示的图 G 进行先深搜索 { EdgeNode *p; cout<<G->vexlist[i].vertex; // 访问顶点 vi; visited[i]=TRUE; // 标记 vi 已访问 dfn[i]=count++; // 对 vi 进行编号 p=G->vexlist[i].firstedge; // 取 vi 边表的头指针 while(p) { // 依次搜索 vi 的邻接点 vj, 这里 j=p->adjvex if (! visit[p->adjvex]) // 若 vj 尚未访问 DFS1(G, p->adjvex); // 则以 vj 为出发点先深搜索 p=p->next; } } // DFS1 </pre>		
哈尔滨工业大学 计算机科学与技术学院		

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 25

```
void DFS1 (AdjGraph* G, int i)
//以vi为出发点时对邻接表表示的图G进行先深搜索
{ EdgeNode *p;
    cout<<G->vexlist[i].vertex;
    visited[i]=TRUE;
    dfn[i]=count++;
    p=G->vexlist[i].firstedge;
    while(p ) {
        if( !visit[p->adjvex] )
            DFS1(G, p->adjvex);
        p=p->next;
    }
} //DFS1
```

顶点表 边表

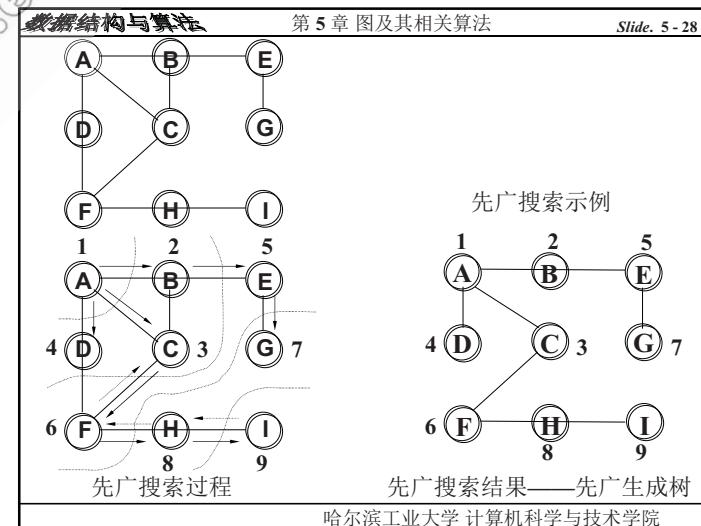
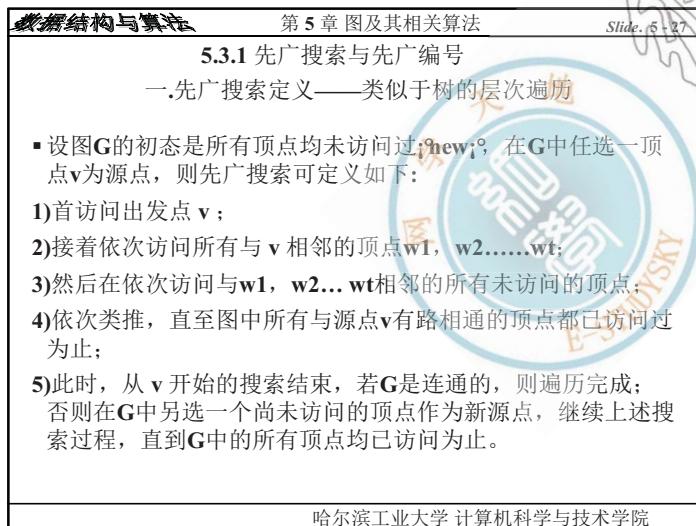
哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 26

```
void DFS2(MTGraph *G, int i)
//以vi为出发点对邻接矩阵(0,1矩阵)表示的图G进行深度优先搜索
{ int j;
    cout<<G->vexlist[i]; //访问定点vi
    visit[i]=TRUE; //标记vi已访问
    dfn[i]=count; //对vi进行编号
    count++;
    for( j=0; j<G->n; j++ ) //依次搜索vi的邻接点
        if((G->edge[i][j] == 1)&&! visited[j] ) //若vj尚未访问
            DFS2( G, j );
} //DFS2
```

$G.edge = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$

哈尔滨工业大学 计算机科学与技术学院



哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 29
<ul style="list-style-type: none"> 先广搜索特点：尽可能横向进行搜索，并使先被访问的顶点的邻接点先于后被访问的顶点的邻接点被访问，故称先广搜索或广度优先搜索。 先广搜索过程中，根据访问顺序给顶点进行的编号，称为先广编号。 先广搜索过程中，根据访问顺序得到的顶点序列，称为先广序列或BFS序列。 先广搜索结果不唯一，即图的BFS序列和先广编号不唯一。 如何保证先访问的顶点其邻接点亦先访问—先进先出原则 <p>设x和y是两个相继被访问的顶点，它们的邻接点分别记为x1,x2...xs,和y1,y2...yt。因为x先于y被访问，故访问x1,x2,...xs中未访问的顶点必先于y1,y2...yt中的未访问者。为了确保这种先访问的顶点其邻接点亦先被访问的先进先出原则，可以使用队列来保存已访问的顶点(或者保存尚未访问的顶点？)。</p>		

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 30
<h3>5.3.2 先广搜索与先广编号</h3> <p>二. 先广搜索算法（算法5.2）</p> <p>输入：L[v]表示无向图G的关于v的邻接表 输出：每个结点有先广编号的无向图G和树边集T</p> <pre> void bsearch(v) { MAKENULL(Q); bfn[v] = count; /*先广编号*/ count = count + 1; mark v old; ENQUEUE(v, Q); /*入队*/ while (!EMPTY(Q)) { v = FRONT(Q); DEQUEUE(Q); for (each w ∈ L[v]) if (w is marked) new; bfn[w] = count; count = count + 1; /*先广编号*/ mark w old; ENQUEUE(w, Q); /*树边入T*/ INSERT((v, w), T); /*树边入T*/ } } </pre>		

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 31
<pre> void BFS1 (AdjGraph *G, int k)//这里没有进行先广编号 { int i; EdgeNode *p; QUEUE Q; MAKENULL(Q); cout << G->vexlist[k].vertex; visited[k] = TRUE; ENQUEUE(k, Q); //进队列 while (!Empty(Q)) { //队空搜索结束 i=DEQUEUE(Q); //vi出队 p=G->vexlist[i].firstedge; //取vi的边表头指针 while (p) { //若vi的邻接点vj(j=p->adjvex)存在,依次搜索 if (!visited[p->adjvex]) { //若vj未访问过 cout << G->vexlist[p->adjvex].vertex; //访问vj visited[p->adjvex]=TRUE; //给vj作访问过标记 ENQUEUE (p->adjvex , Q); //访问过的vj入队 } p = p->next; //找vj的下一个邻接点 } //重复检测vi的所有邻接顶点 } //外层循环, 判队列空否 } //以vk为出发点时对用邻接表表示的图G进行先广搜索 </pre>		

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 32
<pre> void BFS2 (MTGraph *G, int k)//这里没有进行先广编号 { int i, j; QUEUE Q; MAKENULL(Q); cout << G->vexlist[k].vertex; visited[k] = TRUE; ENQUEUE(k, Q); //vk进队列 while (!Empty(Q)) { //队空时搜索结束 i=DEQUEUE(Q); //vi出队 for(j=0; j<G->n; j++) { //依次搜索vi的邻接点vj if (G->edge[i][j] == 1 && !visited[j]) { //若vj未访问过 cout << G->vexlist[j].vertex; //访问vj visited[j]=TRUE; //给vj作访问过标记 ENQUEUE (j , Q); //访问过的vj入队 } } //重复检测 vi的所有邻接顶点 } //外层循环, 判队列空否 } //以vk为出发点时对用邻接矩阵表示的图G进行先广搜索 </pre>		

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 33

无向图（的搜索）及其应用

- 无向图连通性判定
- 不连通：
 - 求连通分量个数；
 - 求出每个连通分量；
- 连通：
 - 判断是否有环路；
 - 求带权连通图的最小生成树；**5.3** 和 **5.4**
 - 判断是否是双连通的；
 - 求关键点和双连通分量。

5.5

The diagram shows two graphs. The left graph has 7 nodes (a-g) and 9 edges, forming 3 connected components: {a,b,c}, {d,e,f}, and {g}. The right graph has 7 nodes and 10 edges, forming 2 connected components: {a,b,c} and {d,e,f,g}.

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 34

5.4 图与树的联系

5.4.1 先深生成森林和先广生成森林

The diagram shows a graph with 7 nodes (a-g) and 10 edges. It illustrates two search results:

- DFS (Left):** Shows a tree where node 'a' is the root. Edges are numbered 1 to 10. Nodes are grouped into three sets based on their discovery order: Tree nodes (1, 2, 3, 4, 5), Non-tree nodes (6, 7), and Root node (a).
- BFS (Right):** Shows a tree where node 'a' is the root. Edges are numbered 1 to 7. Nodes are grouped into three sets: Tree nodes (1, 2, 3, 4, 5), Non-tree nodes (6, 7), and Root node (a).

、搜索的结果
生成森林和（先深或先广）编号—顶点的线性序列。
树边 与 非树边
连通图 一个生成树
非连通图 生成森林 连通子图（连通分量）

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 35

5.4.1 先深生成森林和先广生成森林

二、搜索过程中对边的分类

1. 先深搜索对边的分类

- 两类：树边—在搜索过程中所经过的边；回退边—图中的其它边。
- 特点：树边是从先深编号较小的指向较大的顶点；回退边相反；
- 如何在搜索过程中区分树边和回退边？

设 v 是当前访问过的顶点 old ，而下面搜索到 w , w 有三种情况：

1. w 是 new，则 (v,w) 是树边，将其加入 T ；
2. w 是 old ，且 w 是 v 的父亲，则 (w,v) 是树边，但是第二次遇到，不再加入 T ；
3. w 是 old 且 w 不是 v 的父亲，则 (v,w) 是回退边

■ 结论：若 G 中存在环路，则在先深搜索过程中必遇到回退边；反之亦然

The diagram shows a graph with 7 nodes (a-g) and 10 edges. It highlights edges as tree edges (solid) or back edges (dashed). In the search tree, edges (a,b), (a,c), (b,d), (b,e), (c,f), (c,g) are tree edges, while (d,f), (e,f), (f,g) are back edges.

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 36

5.4.1 先深生成森林和先广生成森林

二、搜索过程中对边的分类

2. 先广搜索对边的分类

- 两类：树边—在搜索过程中所经过的边；横边—图中的其它边。
- 特点：树边是从先广编号较小的指向较大的顶点；而横边不一定与之相反，但可规定：大 \rightarrow 小。

The diagram shows a graph with 7 nodes (a-g) and 10 edges. It highlights edges as tree edges (solid) or horizontal edges (dashed). In the search tree, edges (a,b), (a,c), (b,d), (b,e), (c,f), (c,g) are tree edges, while (d,f), (e,f), (f,g) are horizontal edges.

■ 结论：若 G 中存在环路，则在先广搜索过程中必遇到横边；反之亦然

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 37
■ 一个小附录		
连通分量 (Connected component)		
当无向图为非连通图时，从图中某一顶点出发，利用深度优先搜索算法或广度优先搜索算法不可能遍历到图中的所有顶点，只能访问到该顶点所在的最大连通子图(连通分量)的所有顶点。		
若从无向图的每一个连通分量中的一个顶点出发进行遍历，可求得无向图的所有连通分量。		
■ 求连通分量的算法需要对图的每一个顶点进行检测：若已被访问过，则该顶点一定是落在图中已求得的连通分量上；若还未被访问，则从该顶点出发遍历图，可求得图的另一个连通分量。		
■ 对于非连通的无向图，所有连通分量的生成树组成了非连通图的生成森林。		
哈尔滨工业大学 计算机科学与技术学院		

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 38
■ 5.4.2 无向图与开放树的联系		
定义：连通而无环路的无向图称作开放树 (Free Tree)		
开放树的性质：(证明见教材P154~155)		
(1) 具有 $n \geq 1$ 个顶点的开放树包含 $n-1$ 条边；		
(2) 如果在开放树中任意加上一条边，便得到一条回路。		
联系：如果指定开放树中的一个顶点为根(如搜索起点)，并且把每条边看成是背离根的，则一株开放树变成一株树。		
■ 5.4.3 最小生成树		
设 $G = (V, E)$ 是一个连通图， E 中每一条边 (u, v) 的权为 $c(u, v)$ ，也叫做边长。图 G 的一株生成树 (spanning tree) 是连接 V 中所有结点的一株开放树。将生成树中所有边长之和称为生成树的价 (cost)。使这个价最小的生成树称为图 G 的最小生成树 (minimum-cost spanning tree)。		
哈尔滨工业大学 计算机科学与技术学院		

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 39
■ 5.4.3 最小生成树		
■ 使用不同的遍历图的方法，可以得到不同的生成树；从不同的顶点出发，也可能得到不同的生成树。		
■ 按照生成树的定义， n 个顶点的连通图的生成树有 n 个顶点、 $n-1$ 条边。		
■ 构造最小生成树的准则		
■ 必须使用且仅使用该连通图中的 $n-1$ 条边连接结图中的 n 个顶点；		
■ 不能使用产生回路的边；		
■ 各边上的权值的总和达到最小。		
哈尔滨工业大学 计算机科学与技术学院		

数据结构与算法	第5章 图及其相关算法	Slide. 5 - 40
■ 5.4.3 最小生成树		
MST性质		
描述1：		
设 $G = (V, E)$ 是一个连通图，在 E 上定义一个权函数，且 $\{(V_1, T_1), (V_2, T_2), \dots, (V_k, T_k)\}$ 是 G 的任意生成森林。令		
$T = \bigcup_{i=1}^k T_i \quad (k > 1)$		
又设 $e = (v, w)$ 是 $E - T$ 中这样一条边，其权 $C[v, w]$ 最小，而且 $v \in V_1$ 和 $w \in V_1$ 。则图 G 有一棵包含 $T \cup \{e\}$ 的生成树，其价不大于包含 T 的任何生成树的价。		
描述2：		
假设 $N = (V, \{E\})$ 是一个连通网， U 是顶点 V 的一个非空子集。若 (u, v) 是一条具有最小权值 (代价) 的边，其中 $u \in U, v \in V - U$ ，则必存在一棵包含边 (u, v) 的最小生成树。		
哈尔滨工业大学 计算机科学与技术学院		

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

MST性质证明

假设网N的任何一棵最小生成树都不包含(u, v)，设 T 是连通网上的一棵最小生成树，当将边(u, v)加入到 T 中时，由生成树的定义， T 中必包含一条(u, v)的回路。另一方面，由于 T 是生成树，则在 T 上必存在另一条边(u', v')，且 u 和 u' 、 v 和 v' 之间均有路径相同。删去边(u', v')便可消去上述回路，同时得到另一棵最小生成树 T' 。但因为(u, v)的代价不高于(u', v')，则 T' 的代价亦不高于 T ， T' 是包含(u, v)的一棵最小生成树。

哈尔滨工业大学 计算机科学与技术学院

1、求最小生成树——Prim 算法（算法5.3）

输入：连通的加权无向图（无向网） $G=(V, E)$ ，其中 $V=(1, 2, \dots, n)$ 。
输出： G 的最小生成树

要点：引入集合 U 和 T 。 U 存放生成树的顶点， T 存放生成树的边集。初值 $U=\{1\}$ ， $T=\emptyset$ 。选择有最小权的边(u, v)， $u \in U$, $v \in (V-U)$ ，将 v 加入 U ，(u, v)加入 T 。重复这一过程，直到 $U=V$ 。

```

void Prim(G, T)
{
    T = ∅;
    选项点！ U = { 1 };
    while ((U - V) != ∅) {
        设(u, v)是使u ∈ U与v ∈ (V-U)且权最小的边；
        T = T ∪ {(u, v)};
        U = U ∪ {v};
    }
}
    
```

/* 时间复杂性：O(n^2) 哈尔滨工业大学 计算机科学与技术学院

求最小生成树Prim算法

哈尔滨工业大学 计算机科学与技术学院

```

void Prim(C)
Costtype C[n+1][n+1];
{ costtype LOWCOST[n+1]; int CLOSSET[n+1]; int i,j,k; costtype min;
for( i=2; i<=n; i++ )
    { LOWCOST[i] = C[1][i]; CLOSSET[i] = 1; }
for( i = 2; i <= n; i++ )
    { min = LOWCOST[i];
        k = i;
        for( j = 2; j <= n; j++ )
            if ( LOWCOST[j] < min )
                { min = LOWCOST[j]; k=j; }
        cout << i << k << endl;
        CLOSSET[k] = 1;
        LOWCOST[k] = infinity;
        for (j = 2; j <= n; j++)
            if ( C[k][j] < LOWCOST[j] && LOWCOST[j] < infinity )
                { LOWCOST[j] = C[k][j]; CLOSSET[j]=k; }
    }
}
    
```

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 45

2、求最小生成树——Kruskal 算法（算法）

输入：带权连通图 $G=(V, E)$, 其中 $V=(1, 2, \dots, n)$
输出：G的最小生成树
算法要点：选边！

- 起初，令 $T=(V, \Phi)$ （最小生成树的初态只有 n 个顶点而无边的非连通图）， T 由 G 的 n 个顶点构成，没有任何一条边。这样 T 的每个顶点自身构成一个连通分量。
- 然后按权值不减的顺序，依次考察 E 中的每条边。如果被考察的边连接两个不同分量的两个顶点，则把两个分量合并在一起，构成一个分量；如果被考察的边连接同一分量的两个顶点，则放弃这条边，避免形成环路。
- 于是， T 中的连通分量的个数逐渐减少，当 T 中的连通分量个数为 1 时，说明 V 中的全部顶点，通过 E 中权值最小的边，构成了一个没有环路的连通图 T ，即图 G 的最小生成树。

哈尔滨工业大学 计算机科学与技术学院

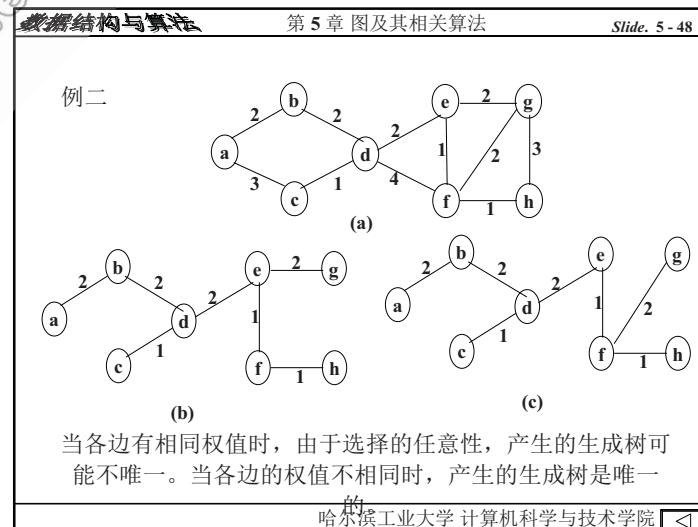
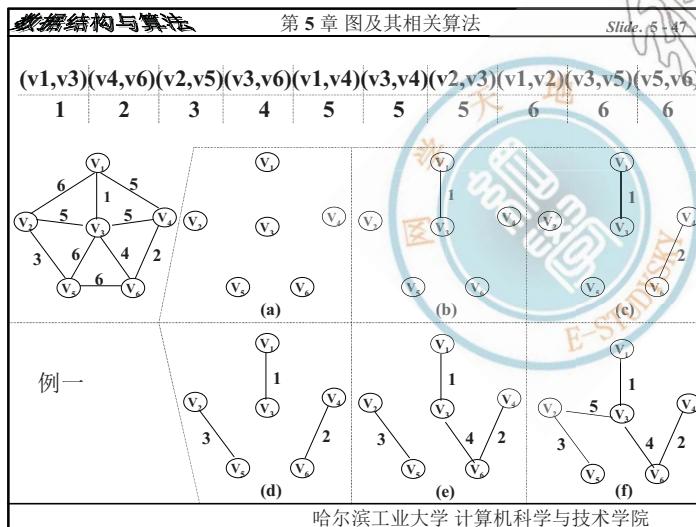
数据结构与算法 第5章 图及其相关算法 Slide. 5 - 46

2、求最小生成树——Kruskal 算法（算法）

输入：带权连通图 $G=(V, E)$, 其中 $V=(1, 2, \dots, n)$
输出：G的最小生成树
算法描述：

```
void Kruskal ( V, T )
{ T = V;
  ncomp = n; /*结点个数*/
  while ( ncomp > 1 ) {
    从 E 中取出权最小的边 (v, u) ;
    if ( v 和 u 属于 T 中不同的连通分量){
      T = T ∪ { (v, u) } ;
      ncomp -- ;
    }
  }
} /* O (|E| * log |E|) */
```

哈尔滨工业大学 计算机科学与技术学院



哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

第5章 图及其相关算法 Slide. 5 - 49

5.5 无向图的双连通性

5.5.1 无向图的双连通分量 (Biconnected Component)

设 $G = (V, E)$ 是一个连通的无向图

定义 称顶点 $a \in V$ 是关节点 (articulation point)，如果存在 $v, w \in V, v \neq w \neq a$ ，使得 v 和 w 之间的每条路都包含 a 。

若在删去顶点 v 以及与之相邻的边之后，将图的一个连通分量分割成两个或两个以上的连通分量，则称该顶点为关节点。

定义 若对 V 中每个不同的三元组 v, w, a ：在 v 和 w 之间都存在一条不包含 a 的路，就说 G 是双连通的 (biconnected)。没有关节点的连通图称为双连通图。

哈尔滨工业大学 计算机科学与技术学院

第5章 图及其相关算法 Slide. 5 - 50

- 双连通的无向图是连通的，但连通的无向图未必双连通。
- 一个连通的无向图是双连通的，当且仅当它没有关节点。
- 在双连通图上，任何一对顶点之间至少存在有两条路径，在删去某个顶点及与该顶点相关联的边时，也不破坏图的连通性。
- 一个连通图 G 如果不是重连通图，那么它可以包括几个双连通分量。

定义 说边 e_1 和 e_2 是等价的，若 $e_1 = e_2$ 或者有一条环路包含 e_1 又包含 e_2 。

定义 设 V_i 是等价边集 E_i 中各边所连接的点集 ($1 \leq i \leq k$)，每个图 $G_i = (V_i, E_i)$ 叫做 G 的一个双连通分量。

双连通图的性质：

性质1 G_i 是双连通的 ($1 \leq i \leq k$)

性质2 对所有的 $i \neq j$, $V_i \cap V_j$ 最多包含一个点

性质3 v 是 G 的关节点，当且仅当 $v \in V_i \cap V_j$ 存在 ($i \neq j$)

哈尔滨工业大学 计算机科学与技术学院

第5章 图及其相关算法 Slide. 5 - 51

5.5.2 求关节点—对图进行一次先深搜索边可求出所有的关节点

由先深生成树可得出两类关节点的特征

- 若生成树的根有两株或两株以上子树，则此根结点必为关节点（第一类关节点）。因为图中不存在连接不同子树中顶点的边，因此，若删去根顶点，生成树变成生成森林。
- 若生成树中非叶顶点 v ，其某株子树的根和子树中的其它结点均没有指向 v 的祖先的回退边，则 v 是关节点（第二类关节点）。因为删去 v ，则其子树和图的其它部分被分割开来。

哈尔滨工业大学 计算机科学与技术学院

第5章 图及其相关算法 Slide. 5 - 52

定义 $low[v]$ 设对连通图 $G = (V, E)$ 进行先深搜索的先深编号为 $dfn[v]$ ，产生的先深生成树为 $S = (V, T)$ ， B 试回退边之集。对每个顶点 v ， $low[v]$ 定义如下：

$$low[v] = \min \left\{ dfn[v], dfn[w], low[y] \mid \begin{array}{l} (y, v) \in B, y \text{ 是顶点 } v \text{ 在先深生成树上有回退边连接的祖先结点;} \\ (v, w) \in T, w \text{ 是顶点 } v \text{ 在先深生成树上的孩子顶点} \end{array} \right\}$$

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

第5章 图及其相关算法 *Slide. 5 - 53*

算法5.5 求无向图的双连通分量

输入：连通的无向图 $G=(V, E)$ 。 $L[v]$ 表示关于 v 的邻接表
输出： G 的所有双连通分量，每个连通分量由一序列的边组成。

算法要点：

- 1.计算先深编号：对图进行先深搜索，计算每个结点 v 的先深编号 $dfn[v]$ ，形成先深生成树 $S=(V, T)$ 。
- 2.计算 $low[v]$ ：在先深生成树上按后根顺序进行计算每个顶点 v 的 $low[v]$ ， $low[v]$ 取下述三个结点中的最小者：
 - (1) $dfn[v]$ ；
 - (2) $dfn[w]$ ，凡是有回退边 (v, w) 的任何结点 w ；
 - (3) $low[y]$ ，对 v 的任何儿子 y 。
- 3.求关节点：
 - (1)树根是关节点，当且仅当它有两个或两个以上的儿子（第一类关节点）；
 - (2)非树根结点 v 是关节点当且仅当 v 有某个儿子 y ，使 $low[y] \geq dfn[v]$ （第二类关节点）。

哈尔滨工业大学 计算机科学与技术学院

第5章 图及其相关算法 *Slide. 5 - 54*

求双连通分量的算法——同先深搜索算法（略）见书上P161

哈尔滨工业大学 计算机科学与技术学院

第5章 图及其相关算法 *Slide. 5 - 55*

5.6 有向图的搜索

一、先深搜索对边的分类

1. 分类四类

- 树边：搜索时所经过的边。
- 向前边：从一个顶点指向其后代的边，如果不是树边，则称为向前边。如 A→D。
- 回退边：从一个顶点到生成森林中该顶点祖先顶点的边，如 C→A。
- 横边：从一个顶点到另一个与之不存在祖先和先代关系的顶点的边称为横边。

哈尔滨工业大学 计算机科学与技术学院

第5章 图及其相关算法 *Slide. 5 - 56*

5.6 有向图的搜索

2. 如何在搜索中区分

其一，根据先深编号，把边分为两类³

- 若 $dfn[v] < dfn[w]$, 则 (v, w) 是树边或向前边；
- 若 $dfn[v] > dfn[w]$, 则 (v, w) 是回退边或横边；

其二，根据 $visit[v]$ 和 $dfn[v]$ 把树边和向前边区分开

- 若 $visit[v] = "old"$, $visit[w] = "new"$, 则 $(v, w) \in T$;
- 若 $visit[v] = "old" \neq visit[w] = "old"$ 且 $dfn[v] < dfn[w]$, 则 $(v, w) \in$ 向前边；

其三，引入 $father[v]$, 把回退边和横边区分开

当 $visit[v] = "old" \neq visit[w] = "old"$ 且 $dfn[v] > dfn[w]$ 时，由沿着 v 的树边向上 ($father[v]$) 查找 w (可能一直到根)。若找到 w , 则 (v, w) 是回退边；否则 (v, w) 是横边。

哈尔滨工业大学 计算机科学与技术学院

5.6 有向图的搜索

二、先广搜索对边的分类

1.分三类：树边、回退边、横边
2.与先广的区别：无向前边、横边可能从先广编号大的结点指向编号小的或相反

哈尔滨工业大学 计算机科学与技术学院

5.7 强连通性

定义设 $G = (V, E)$ 是一个有向图，称顶点 $v, w \in V$ 是等价的，要么 $v = w$ ；要么从顶点 v 到 w 有一条有向路，并且从顶点 w 到 v 也有一条有向路。

上述等价关系将 V 分成若干个等价类 V_1, V_2, \dots, V_r

定义设 $E_i (1 \leq i \leq r)$ 是头、尾均在 V_i 中的边集，则 $G_i = (V_i, E_i)$ 称为 G 的一个强连通分量，简称强分量。

强连通图：只有一个强分量的有向图称为强连通图。
显然，有向图的强连通分量是满足下列要求的最大子集：
对任意两个顶点 x 和 y ，都存在一条有向路从 x 到 y ，也存在一条有向路从 y 到 x 。

哈尔滨工业大学 计算机科学与技术学院

5.7 强连通性

分支横边：不在任何强连通分支（量）中的边，如： $a \rightarrow d, c \rightarrow d$
注：每个结点都是在某个强连通分支中出现，但某些边可能不在任何强分支中。
归约图：用强分量代表顶点，用分支横边代表有向边的图称为原图的归约图。
显然，归约图是一个不存在环路的有向图，它表示了强分量之间的连通性。

哈尔滨工业大学 计算机科学与技术学院

5.7 强连通性

求强连通分量的算法 — 对有向图进行两遍先深搜索

- Step1. 对有向图 G 进行先深搜索，并按逆先根顺序对顶点进行编号；
- Step2. 将图 G 中的每条边取反向，构造新的有向图 G_r ；
- Step3. 根据 Step1 的编号，从编号最大的顶点开始对 G_r 进行一次先深搜索，凡是经过树边（Step1 中形成的横边除外）能到达的所有顶点，形成一株先深生成树；若这次搜索没能达到所有顶点，则下次先深搜索从余下顶点中编号最大的顶点开始。
- Step4. 在 G_r 的先深生成森林中，每株树对应 G 的一个强分量。

哈尔滨工业大学 计算机科学与技术学院

第5章 图及其相关算法 **Slide. 5 - 61**

5.7 强连通性

求强连通分量的算法 — 对有向图进行两遍先深搜索

要注意的主要问题：

- 如何对顶点进行逆先根编号？
- 见P151。出栈时编号
- 如何把有向边取反？
- 逆邻接表当正邻接表/邻接矩阵转置。
- 如何区分分支横边和非分支横边？

引入father[]数组。 (v, w) 是分支横边，如果 v, w 无共同祖先。

哈尔滨工业大学 计算机科学与技术学院

第5章 图及其相关算法 **Slide. 5 - 62**

有向无环图及其应用

5.8 拓扑分类

5.9 关键路径

背景

所有的工程或系统都可以分为若干个称作活动（activity）的子工程，而这些子工程之间，通常存在着一定的制约关系，如其中的某些子工程必须在另外一些子工程完成之后开始。对于整个工程或系统，我们关心的是两个方面的问题：

- 工程能否顺利完成？（AOV网络）
- 工程完成所必需的最短时间？（AOE网络）

哈尔滨工业大学 计算机科学与技术学院

第5章 图及其相关算法 **Slide. 5 - 63**

5.8 拓扑分类

5.8.1 无环路有向图

无环路有向图：

- 不存在环路的有向图简称为无环路有向图。
- 注意：无环路的有向图对应的无向图可能存在环路。
- 无环路的有向图可以描述含有公共子式的表达式(节省空间)。
- 无环路的有向图④用于表示偏序关系。

公共子式
 b
 $(c+d)$
 $(c+d)*e$

哈尔滨工业大学 计算机科学与技术学院

第5章 图及其相关算法 **Slide. 5 - 64**

5.8.1 无环路有向图

偏序关系：若集合X上的关系R是自反的、反对称的和传递的即

- 自反性：任意 $x \in X$, $(x, x) \in R$
- 反对称性：任意 $x, y \in X$, 若 $(x, y) \in R$ 且 $(y, x) \in R$, 则 $x=y$
- 传递性：任意 $x, y, z \in X$, $(x, y) \in R$ 且 $(y, z) \in R$, 则 $(x, z) \in R$

则称R是集合X上的偏序关系。

全序关系：设R是集合X上的偏序，如果对每个 $x, y \in X$, 必有 $(x, y) \in R$ 或 $(y, x) \in R$, 则称R是集合X上的全序关系。

直观上，偏序指集合上只有部分元素之间可比较，而全序是指全体元素均可比较。

无环路的有向图可用于表示偏序关系：

设R是有穷集合上的偏序关系，对X中每个 v , 用一个以 v 为符号的结点表示，由此构成结点集V，对任意 $(a, b) \in R$, $(a \neq b)$, 由对应两个结点建立一条边，由此构成边集E，则 $G = (V, E)$ 是无环路有向图。

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

第 5 章 图及其相关算法 *Slide. 5 - 65*

5.8.2 拓扑分类算法

拓扑分类问题描述：
给定一个无环路有向图 $G=(V,E)$ ，各顶点的编号为 $V=(1,2,\dots,n)$ 。
要求对每一个结点 i 重新进行编号，使得若 i 是 j 的前导，则有 $label[i] < label[j]$ 。即拓扑分类是将无环路有向图排成一个线性序列，使当从结点 i 到结点 j 存在一条边，则在线性序列中，将 i 排在 j 的前面。

简单地说，拓扑分类是由某个集合上的一个偏序的到该集合上的 **AOV网序** (**Activity On Vertex Network**)。

用顶点表示活动，用边表示活动间的优先关系的有向图称为顶点表示活动的网络简称 **AOV网**。

在 **AOV网** 中，若从顶点 i 到 j 有一条有向路，则称 i 为 j 的前驱， j 为 i 的后继。若 $(i, j) \in E$ ，则 i 称为 j 的直接前驱， j 称为 i 的直接后继。

哈尔滨工业大学 计算机科学与技术学院

第 5 章 图及其相关算法 *Slide. 5 - 66*

例：教学计划制定
■ 课程及课程间的先修关系可用 **AOV网** 表示

任何无环路的 **AOV网**，其顶点都可以排成一个拓扑序列，并且其拓扑序列不一定是唯一的。

拓扑分类算法基本步骤：

- 在 **AOV网** 中选择一个入度为0的顶点且输出之；
- 从 **AOV网** 中删除该顶点及其所有出边；
- 重复以上两步，直到全部顶点已输出；或图中没有无前驱的顶点为止（此时有环路）。

哈尔滨工业大学 计算机科学与技术学院

第 5 章 图及其相关算法 *Slide. 5 - 67*

算法5.6 (1e)拓扑分类算法——实质是先广搜索算法

输入：用邻接表表示的有向图
输出：所有顶点组成的拓扑序列

算法要点：（使用排队）
建立入度为零的顶点排队
扫描顶点表，将入度为0的顶点入队；
while（排队不空）{
 输出队头结点；
 记下输出结点的数目；
 删去与之关联的出边；
 若有入度为0的结点，进队；
}
若输出结点个数小于n，则输出有环路；
否则拓扑排序正常结束。

哈尔滨工业大学 计算机科学与技术学院

第 5 章 图及其相关算法 *Slide. 5 - 68*

算法5.6(1e)的求精：

```
void Topologicalsort( AdjGraph G )
{
    QUEUE Q;
    MAKENUILL( Q );
    for( v=1; v<=G.n ; ++v )
        if( indegree[v] ==0 ) ENQUEUE( v, Q );
    nodes = 0 ;
    while ( !EMPTY( Q ) )
    {
        v = FRONT( Q );
        DEQUEUE( Q );
        cout << v ; nodes ++ ;
        for( 邻接于 v 的每个顶点 w )
            if( !(--indegree[w]) ) ENQUEUE(w,Q);
    }
    if( nodes < n ) cout << "图中有环路";
```

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 69

关于先广拓扑分类的几点说明

1. 与先广搜索的差别：
 - 搜索起点是入度为0的顶点；
 - 需判断是否有环路；
 - 需对访问并输出的顶点计数（引入计数器**nodes**）。
 - 需删除邻接于 v 的边（引入数组**indegree[]**或在顶点表中增加一个属性域**indegree**）。
2. 也可以采用栈数据结构进行先广拓扑分类。
3. 上述拓扑分类算法可以称为无前驱顶点优先的拓扑分类算法，亦可采用无后继顶点优先的拓扑分类算法。只不过得到的拓扑序列与前者恰好相反。
4. 也可以利用**DFS**遍历生成拓扑序列。

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 70

算法5.6 (2e)拓扑分类算法——实质是先广搜索算法

输入：用邻接表表示的有向图
输出：所有顶点组成的拓扑序列
算法要点：（使用栈）
建立入度为零的顶点栈
扫描顶点表，将入度为0的顶点入栈；
while（栈不空）{
 弹出栈顶顶点vj 并输出之；
 记下输出顶点的数目；
 检查关于vj的邻接表，将每条出边<vj,vk>的入度减1；
 若有入度为0的结点，压入栈；
}
若输出结点个数小于n，则输出“有环路”；
否则拓扑排序正常结束。

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 71

算法5.6(2e)的求精：

```
void Topologicalsort( AdjGraph G )
{
    MAKENUILL(S);
    for( v=0; v<n ; ++v )
        if( !indegree[v] ) push( v, S );
    count = 0 ;
    while( !EMPTY( S ) ){
        v = pop( S ); printf( v ); ++count ;
        for( 邻接于 v 的每个顶点 w ){
            if( !( -indegree[w] ) )
                push( S, w );
        }
        if( count < n ) cout<<“图中有环路”;
    }
}
```

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 72

利用**DFS**遍历生成拓扑序列

```
void topodfs( v )
{
    PUSH( v ,S );
    mark[v]=TRUE;
    for( L[v] 中的每一个顶点w) do
        if( mark[w] = FALSE )
            topodfs( w );
    printf( top( S ) );
    POP( S );
}

void dfs-topo ( GRAPH L )
{
    MAKENUILL(S);
    for( u=1;u<=n;u++ )
        make[u]=FALSE;
    for( u=1;u<=n;u++ )
        if( !mark[u] )
            topodfs( u );
}
```

思想：借助栈，在DFS中，把第一次遇到的顶点入栈，到达某一顶点递归返回时，从栈中弹出顶点并输出。

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 73

5.9 关键路径——拓扑分类

AOE网（Activity On Edge Network）

在带权的有向图中，用结点表示事件，用边表示活动，边上权表示活动的开销（如持续时间），则称此有向图为边表示活动的网络，简称AOE网。

下图是有11项活动，9个事件的AOE网，每个事件表示在它之前的活动已经完成，在它之后的活动可以开始。

AOE网

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 74

5.9 关键路径——拓扑分类

AOE网的性质

- 只有在某个顶点所代表的事件发生后，从该顶点出发的各边代表的活动才能开始；
- 只有在进入某一顶点的各边代表的活动已经结束，该顶点所代表的事件才能发生；
- 表示实际工程计划的AOE网应该是无环的，并且存在唯一的入度为0的开始顶点（源点）和唯一的出度为0的结束点（汇点）。

AOE网

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 75

5.9 关键路径——拓扑分类

AOE网研究的主要问题：

如果用AOE网表示一项工程，那么仅仅考虑各个子工程之间的优先关系还不够，更多地是关心整个工程完成的最短时间是多少，哪些活动的延迟将影响整个工程进度，而加速这些活动能否提高整个工程的效率，因此AOE网有待研究的问题是：

- (1) 完成整个工程至少需要多少时间？
- (2) 哪些活动是影响工程进度的关键活动？

AOE网

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 76

5.9 关键路径——拓扑分类

路径长度、关键路径、关键活动：

- 路径长度：**是指从源点到汇点路径上所有活动的持续时间之和。
- 关键路径：**在AOE网中，由于有些活动可以并行，所以完成工程的最短时间是从源点到汇点的最大路径长度。因此，把从源点到汇点具有最大长度的路径称为关键路径。
- 一个AOE中，关键路径可能不只一条。**
- 关键活动：**关键路径上的活动称为关键活动。

AOE网

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：[网学天地](http://www.e-studysky.com)（www.e-studysky.com）；咨询QQ：2696670126

第 5 章 图及其相关算法

Slide. 5 - 80

AOE网示意图，展示了顶点、活动、事件和时间参数。

顶点 (V_i): V₁, V₂, V₃, V₄, V₅, V₆, V₇, V₈, V_j, V_k

活动 (a_i): a₁=6, a₂=4, a₃=5, a₄=1, a₅=1, a₆=2, a₇=9, a₈=7, a₉=4, a₁₀=2, a₁₁=4, a_i

事件 (VE[j], VL[k]):

顶点	VE[j]	VL[k]
V ₁	0	0
V ₂	3	4
V ₃	2	2
V ₄	6	6
V ₅	6	7
V ₆	8	8

活动 (E[i], L[i]):

活动	E[i]	L[i]
a ₁	0	1
a ₂	0	0
a ₃	3	4
a ₄	3	4
a ₅	2	2
a ₆	2	5
a ₇	6	6
a ₈	6	7

时差 (E[i] - L[i]):

	E[i] - L[i]
a ₁	1
a ₂	0
a ₃	1
a ₄	1
a ₅	0
a ₆	3
a ₇	0
a ₈	0
a ₉	1
a ₁₀	2
a ₁₁	0
a _i	1

源点 (起始点): V₁

汇点 (结束点): V_k

AOE网中顶点的发生时间和活动的开始时间

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

5.9 关键路径——拓扑分类

利用拓扑分类算法求关键路径和关键活动

- Step1(前进阶段)：从源点 V_1 出发，令 $VE(1) = 0$ ，按拓扑序列次序求出其余各顶点事件的最早发生时间：

$$VE(k) = \max_{j \in T} \{ VE(j) + ACT[j][k] \}$$

其中 T 是以顶点 V_k 为尾的所有边的头顶点的集合($2 \leq k \leq n$)

如果网中有回路，不能求出关键路径则算法中止；否则转

- Step2(回退阶段)：从汇点 V_n 出发，令 $VL(n) = VE(n)$ ，按逆拓扑有序求其余各顶点的最晚发生时间：

$$VL(j) = \min_{k \in S} \{ VL(k) + ACT[j][k] \}$$

其中 S 是以顶点 V_j 为头的所有边的尾顶点的集合($2 \leq j \leq n-1$)

哈尔滨工业大学 计算机科学与技术学院

5.9 关键路径——拓扑分类

利用拓扑分类算法求关键路径和关键活动

- Step3：

求每一项活动 a_i 的最早开始时间：

$$E(i) = VE(j)$$

最晚开始时间：

$$L(i) = VL(k) - ACT[j][k]$$

若某条边满足 $E(i) = L(i)$ ，则它是关键活动。

为了简化算法，可以在求关键路径之前已经对各顶点实现拓扑排序，并按拓扑有序的顺序对各顶点重新进行了编号。

- 不是任意一个关键活动的加速一定能使整个工程提前。
- 想使整个工程提前，要考虑各个关键路径上所有关键活动。

哈尔滨工业大学 计算机科学与技术学院

5.9 关键路径——拓扑分类

利用拓扑分类算法求关键路径和关键活动实现提示：

- 在拓扑分类之前设初值， $VE[i]=0$ ；
- 增加一个计算 Vi 的直接后继 Vk 的最早发生时间操作：若 $VE[j]+ACT[j,k]>VE[k]$ ，则 $VE[k]=VE[j]+ACT[j,k]$ ；
- 为了能按逆拓扑有序序列的顺序计算个顶点的 VL 值，需记下在拓扑排序过程中求得的拓扑有序序列：增加一个栈以记录拓扑有序序列，则在计算各个顶点的 Ve 值之后，从栈顶到栈底便为逆拓扑有序序列。

哈尔滨工业大学 计算机科学与技术学院

5.10 最短路径 (Shortest Path)

- 最短路径问题：如果从图中某一顶点(称为源点)到达另一顶点(称为终点)的路径可能不止一条，如何找到一条路径使得沿此路径上各边上的权值总和达到最小。
- 问题解法
 - ◆ 边上权值非负情形的单源最短路径问题
 - Dijkstra算法 ✓ (讲此算法)
 - ◆ 边上权值为任意值的单源最短路径问题
 - Bellman和Ford算法 × (不讲)
 - ◆ 所有顶点之间的最短路径
 - Floyd算法 ✓ (讲此算法)

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 85

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 86

边上权值非负情形的单源最短路径问题

- 问题的提法：给定一个带权有向图G与源点v，求从v到G中其它顶点的最短路径。限定各边上的权值大于或等于0。
- 为求得这些最短路径，Dijkstra提出按路径长度的递增次序，逐步产生最短路径的算法。首先求出长度最短的一条最短路径，再参照它求出长度次短的一条最短路径，依次类推，直到从顶点v到其它各顶点的最短路径全部求出为止。
- 其它最短路径问题
 - 单目标最短路径问题：找出图中每个顶点v到某个指定结点c最短路径(只需每边取反)
 - 单结点对间最短路径问题：对于某对顶点u和v找出u到v的一条最短路径(以u为源点)
 - 所有顶点间的最短路径问题：对图中每对顶点u和v，找出u到v的最短路径(以每个顶点为源)

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 87

单源最短路径：Dijkstra算法

- 基本思想：是按路径长度递增的次序生成从源点v到其它顶点的最短路径的贪心算法。把图中的所有顶点分成两组，第一组包括已确定最短路径的顶点，第二组包括尚未确定最短路径的顶点，按最短路径长度递增的顺序逐个把第二组的顶点加到第一组中去，直到从源点v出发可以到达的顶点都包含在第一组中。在整个过程中，总保持从v到第一组各顶点的最短距离，都不大于从v到第二组的任何顶点的最短距离。另外，每个顶点对应一个最短距离，第一组的顶点的最短距离值就是从v到此顶点（且只包含第一组顶点为中间顶点）的最短距离长度。

源点S	中间结点	终点	路径长度
1		2	1 0
1		4	3 0
1	4	3	5 0
1	4 3	5	6 0

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法 第5章 图及其相关算法 Slide. 5 - 88

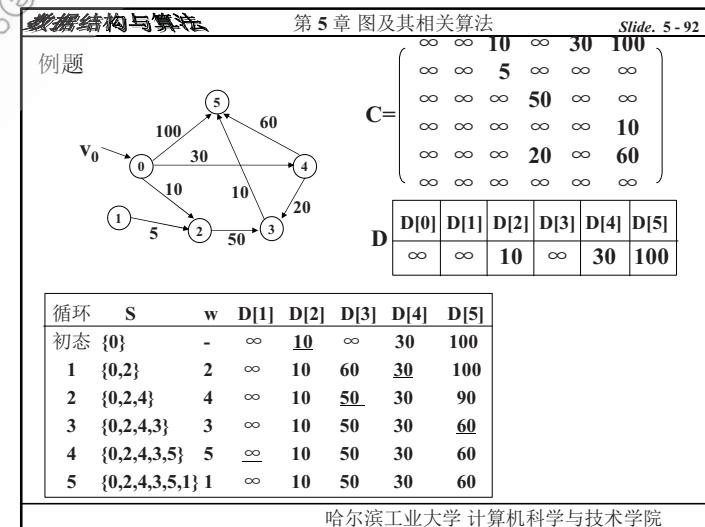
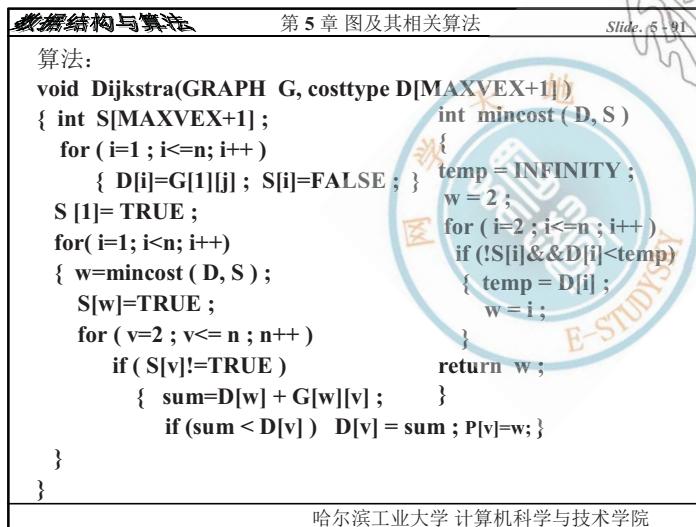
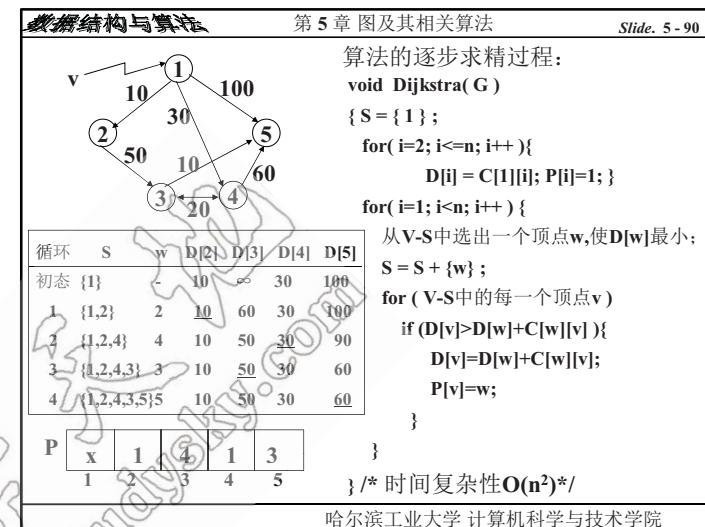
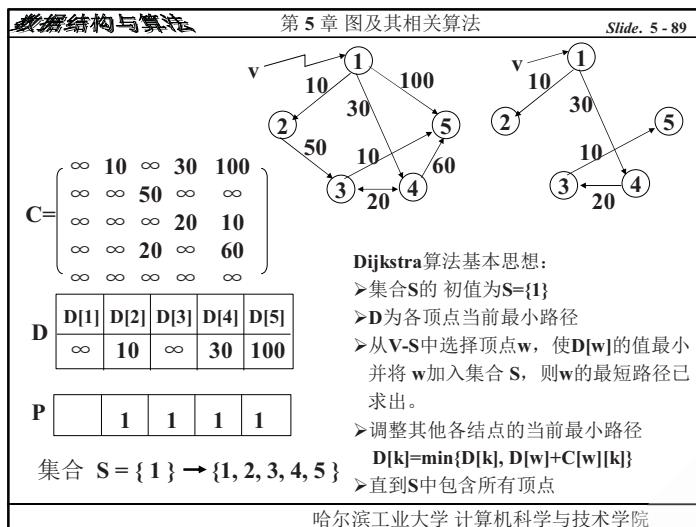
算法5.7 Dijkstra算法

输入：带权有向图G=(V,E) (其中V={1, 2, ..., n})的邻接矩阵C。
输出：从源点1到其余顶点的最短路径长度（及路径）
算法要点：

- 将V分为两个集合S和V-S，其中S是最短路径已经确定的顶点集合，初值为S={1}，V-S是最短路径尚未确定的顶点集合。设数组D记录从源点到其余各结点当前的最短路长，初始为D[i]=C[1][i], i=2,3,...n。设数组P记录从源点到其余顶点最短路径上最后经过的顶点，初始为P[v]=1(源点)，v≠1。
- 从S之外即V-S中选取一个顶点w，使D[w]最小(即选这样的w, D[w]=min{D[i]| i∈V-S})，于是从源点到达w只通过S中的顶点，且是一条最短路径（选定路径），并把w加入集合S。
- 调整D中记录的从源点到V-S中每个顶点的最短距离，即从原来的D[v]和D[w]+C[w][v]中选择最小值作为D[v]的新值(且P[v]=w)。
- 重复上述过程，直到S中包含V的所有顶点为止。结果数组D就记录了从源到V中各顶点的最短距离(数组P记录最短路径)。

哈尔滨工业大学 计算机科学与技术学院

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126



哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：[网学天地](http://www.e-studysky.com)（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法

第5章 图及其相关算法

Slide. 5 - 93

每一对顶点间的最短路径: Floyd算法

问题提法:

已知一个各边权值均大于0的带权有向图, 对每一对顶点 $v_i \neq v_j$, 要求求出 v_i 与 v_j 之间的最短路径和最短路径长度。

基本想法:

- 如果 v_i 与 v_j 之间有一条有向边, 则 v_i 与 v_j 之间有一条路径, 但不一定是最短的;
- 也许经过某些中间会使路径长度更短;
- 经过那些中间点会使路径长度缩短呢? 经过那些中间点会使路径长度最短呢?
- 只需尝试在原路径中间加入其它顶点作为中间顶点。
- 如何尝试? 系统地在原路径中间加入每个顶点, 然后不断地调整当前路径(长度)。

<2,>>5 <2,0><0,1> 4 a[2][1]=a[2][0]+a[0][1] 调整
 注:考虑v0做中间点可能还会改变其它顶点间的
 距离:<-2,0,3>7 <>2,3>8 a[2][3]=a[2][0]+a[0][3]
 <2,>>3 <>2,0><0,3>: <>2,0><0,1><1,3>=<>2,0,1,3>
 a[2][3]=6 调整
 注:有时加入中间顶点后的路径比原路径长,保持

0	1	2	3
0	1	∞	4
∞	0	9	2
3	5	0	8
∞	∞	6	0
3			

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法

第5章 图及其相关算法

Slide. 5-94

每一对顶点间的最短路径: Floyd 算法

基本思想:

假设求顶点 v_i 到顶点 v_j 的最短路径。如果从 v_i 到 v_j 存存在一条长度为 $C[i][j]$ 的路径, 该路径不一定是最短路径, 尚需进行 n 次试探。首先考虑路径 (v_p, v_0, v_j) 是否存在。如果存在, 则比较 (v_p, v_j) 和 (v_p, v_0, v_j) 的路径长度取长度较短者为从 v_i 到 v_j 的中间顶点的序号不大于 0 的最短路径。假设在路径上再增加一个顶点 v_k , 也就是说, 如果 (v_p, \dots, v_k) 和 (v_p, \dots, v_j) 分别是当前找到的中间顶点的序号不大于 0 的最短路径, 那么 $(v_p, \dots, v_k, \dots, v_j)$ 就是有可能是从 v_i 到 v_j 的中间顶点的序号不大于 1 的最短路径。

将它和已经得到的从 v_i 到 v_j 中间顶点序号不大于 0 的最短路径相比较, 从中选出中间顶点的序号不大于 1 的最短路径, 再增加一个顶点 v_{2s} , 继续进行试探。

一般情况下, 若 (v_p, \dots, v_k) 和 (v_k, \dots, v_j) 分别是从 v_i 到 v_k 和从 v_k 到 v_j 的中间顶点序号不大于 $k-1$ 的最短路径, 则将 $(v_p, \dots, v_k, \dots, v_j)$ 和已经得到的从 v_i 到 v_j 且中间顶点序号不大于 $k-1$ 的最短路径相比较, 其长度较短者便是从 v_i 到 v_j 的中间顶点的序号不大于 k 的最短路径。

哈尔滨工业大学 计算机科学与技术学院

数据结构与算法		第5章 图及其相关算法	Slide. 5 - 95
输入:	表示有向图 $G = (V, E)$ 的邻接矩阵 C		
输出:	表示任意两点之间最短路长的矩阵 A		
算法要点:			
void Floyd(A,C,P,n) { for (i = 1; i <= n; i++) for (j = 1; j <= n; j++) { A[i][j] = C[i][j] ; P[i][j] = 0; /* -1 */ } for (k = 1; k <= n; k++) for (i = 1; i <= n; i++) for (j = 1; j <= n; j++) if(A[i][k] + A[k][j] < A[i][j]) { A[i][j] = A[i][k] + A[k][j] ; P[i][j] = k; } } /* 时间复杂性 O(n^3) */	<ul style="list-style-type: none"> 允许有负权，也可以有环路； 但不能有负长度的环路。 	<p>Warshall算法</p> <p>求有向图邻接矩阵C的传递闭包A</p> <p>$A[i][j] = A[i][j] \cup (A[i][k] \cap A[k][j])$;</p> <p>可以判定有向图任意两点间是否存在有向路。</p>	

数据结构与算法

第5章 图及其相关算法

Slide. 5-96

求有向图的中心点

定义：设 $G = (V, E)$ 是一个有向图， $d[i][j]$ 表示从 i 到 j 的最短距离。对任一点 k ，称：

$$E(k) = \max_{i \in V} \{ d[i][k] \}$$

为结点 k 的偏心度，而称具有最小偏心度的结点为图 G 的中心点。

	a	b	c	d	e
a	0	1	3	5	7
b	∞	0	2	4	6
c	∞	3	0	2	4
d	∞	1	3	0	7
e	∞	6	8	5	0

最短路径矩阵 D

算法步骤：

结点	偏心度
a	∞
b	6
c	8
d	5
e	7

第一步：调用 Floyd 算法，求每一对顶点间的最短路径矩阵 D

第二步：对矩阵 D 的每一列求最大值，即为各顶点 j 的偏心

度。

第三步：求具有最小偏心度的顶点。

哈工大计算机考研全套视频和资料，真题、考点、典型题、命题规律独家视频讲解！
详见：网学天地（www.e-studysky.com）；咨询QQ：2696670126

数据结构与算法

第 5 章 图及其相关算法

Slide. 5 - 97

本章小结

- 图的ADT {
 - 数学模型
 - 操作集
- 有向图 无向图 有向网 无向网
- 图的存储结构 {
 - 邻接矩阵
 - 邻接表
- 图的遍历 {
 - 先深搜索(DFS)
 - 先广搜索(BFS)
- 最小生成树 {
 - Prim算法
 - Kruskal算法
- 双连通性 ○强连通性
- 拓扑序列 ○关键路径
- 最短路径 {
 - 单源最短路径
 - 每一对顶点间最短路径

图的搜索算法是有关图问题的重要算法！

哈尔滨工业大学 计算机科学与技术学院



网学天地
www.e-studysky.com