

## 5.2.2图的存储

回忆: 路由协议设问题中, 需要存储路由器和网关的信息, 路由相互之间是否有通路及通路长度三个信息. 即使部分问题不牵涉通路长度(图的权值), 也至少需要存储图的顶点和边的两方面信息, 如何存储?

**仍然有顺序存储和链式存储2种方法!**

# 图的存储1-邻接矩阵

## 一、数组表示法（邻接矩阵）

设图 $G = (V, \{E\})$ 有 $n$ 个顶点，则 $G$ 的邻接矩阵定义为 $n$ 阶方阵 $A$ 。

其中：

$$A[i, j] = \begin{cases} 1 & \text{若}(v_i, v_j) \text{或} \langle v_i, v_j \rangle \text{是图} G \text{的边} \\ 0 & \text{若}(v_i, v_j) \text{或} \langle v_i, v_j \rangle \text{不是图的边} \end{cases}$$

## 图的存储1-邻接矩阵

例如：G1、G2的邻接矩阵

$$A_1 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

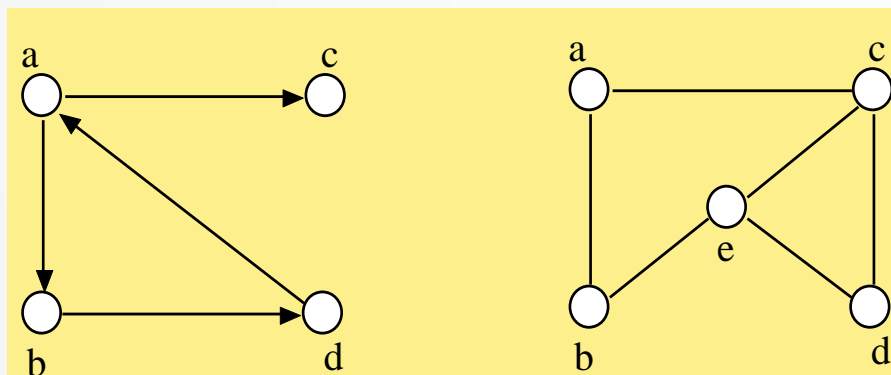


图 4.8 有向图 G1 和无向图 G2

### 邻接矩阵的特点:

1. 判定两个顶点 $V_i$ 与 $V_j$ 是否关联, 只需判 $A[i,j]$ 是否为1;
2. 求顶点的度容易:

• **无向图中:**  $TD(V_i) = \sum_{j=1}^n A[i,j] = \sum A[j,i]$

即顶点 $V_i$ 的度等于邻接矩阵中第 $i$ 行 (或第 $i$ 列) 的元素之和 (非0元素个数)。

• **有向图中:**  $TD(V_i) = OD(V_i) + ID(V_i)$

$$= \sum_{j=1}^n A[i,j] + \sum_{j=1}^n A[j,i]$$

即顶点 $V_i$ 的出度为邻接矩阵中第 $i$ 行元素之和  
顶点 $V_i$ 的入度为邻接矩阵中第 $i$ 列元素之和

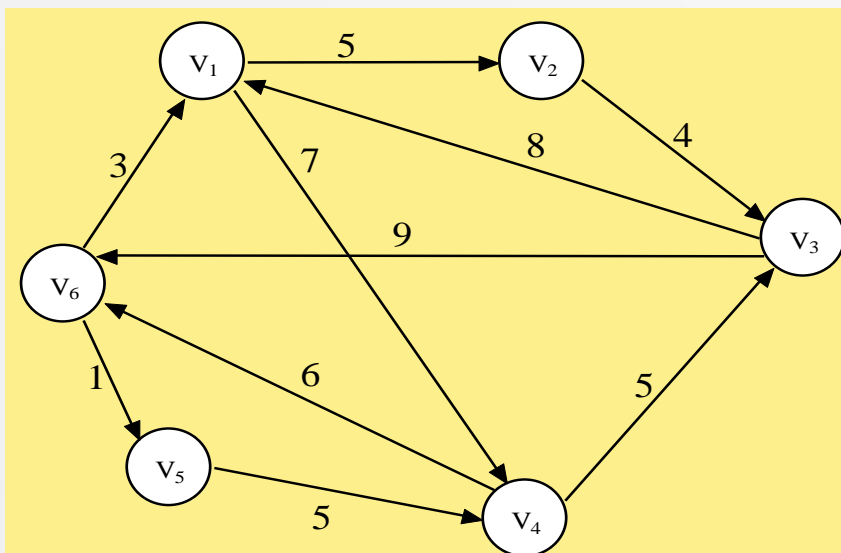
## 图的存储1-邻接矩阵

如果G是带权图， $w_{ij}$ 是边  $(v_i, v_j)$  或  $\langle v_i, v_j \rangle$  的权，则其邻接矩阵定义为：

$$A[i, j] = \begin{cases} w_{ij} & \text{若 } (v_i, v_j) \text{ 或 } \langle v_i, v_j \rangle \text{ 是图G的边 } (i \neq j) \\ \infty & \text{若 } (v_i, v_j) \text{ 或 } \langle v_i, v_j \rangle \text{ 不是图G的边 } (i \neq j) \\ 0 & \text{所有对角线元素 } (i = j) \end{cases}$$

# 图的存储1-邻接矩阵

例如



(a) 网

0	5	$\infty$	7	$\infty$	$\infty$
$\infty$	0	4	$\infty$	$\infty$	$\infty$
8	$\infty$	0	$\infty$	$\infty$	9
$\infty$	$\infty$	5	0	$\infty$	6
$\infty$	$\infty$	$\infty$	5	0	$\infty$
3	$\infty$	$\infty$	$\infty$	1	0

(b) 邻接矩阵

图4.9 网及其邻接矩阵

## 讨论:

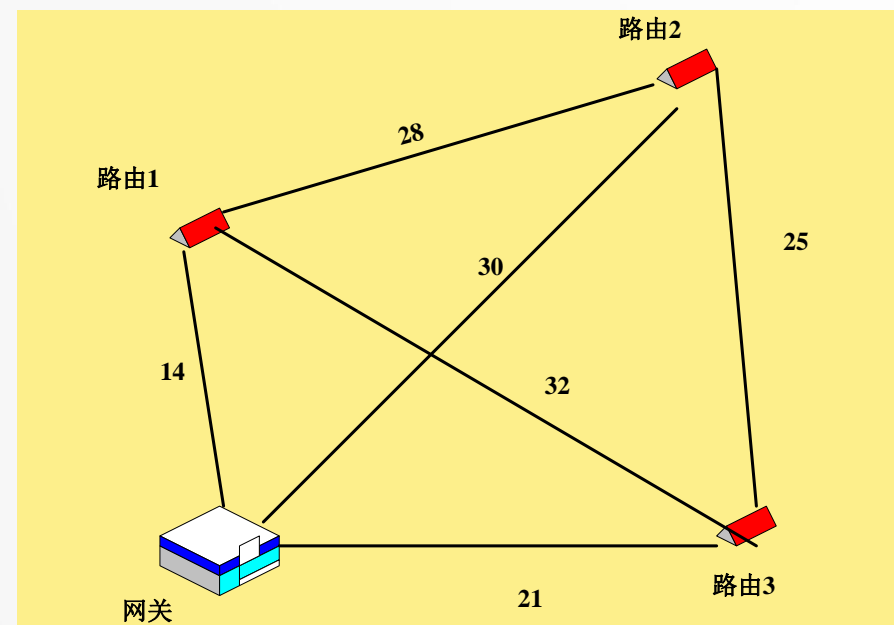
请采用邻接矩阵存储右图路由网络数据

答: 设 $G=\{V,E\}$

$V=\{ \text{'路由1' ; 路由2' ; 路由3' ; 网关'} \}$

4个顶点,则邻接矩阵为

$$A[4][4] = \begin{Bmatrix} 0 & 28 & 32 & 14 \\ 28 & 0 & 25 & 30 \\ 32 & 25 & 0 & 21 \\ 14 & 30 & 21 & 0 \end{Bmatrix}$$





网络的邻接矩阵的定义:

```
#define MaxVerterNum 100
typedef char VerterType;
typedef int EdgeType;
typedef struct{
    VerterType vexs[MaxVerterNum];//存储顶点的一维数组
    EdeType edges[MaxVerterNum][MaxVerterNum];//
    int n,e; //图当前的顶点数和边数
} MGragh;
```

存储邻接矩阵  
的二维数组

建立无向网络邻接矩阵的算法:

```
Viod createMGragh(MGragh *G)
int i,j,k,w;
scanf( "%d%d" ,&G->n,&G->e); //读入顶点数和边数
for(i=0;i<G->n;i++) //读入顶点信息, 建立顶点表
    G->vexs[i]=getchar( );
```



```
for(i=0;i<G->n;i++) //邻接矩阵初始化
    for(j=0;j<G->n;j++)
        G->edges[i][j]=0;
for(k=0;k<G->e;k++){ //读入e条边，建立邻接矩阵
    scanf( "%d%d" ,&i,&j); //读入边<vi,vj>
    G->edges[i][j]=1;
    G->edges[j][i]=1;}
}
```

算法时间复杂度：  $O(n^2)$