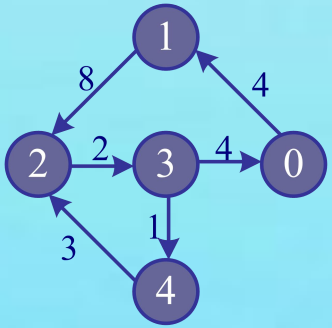
本节主题：

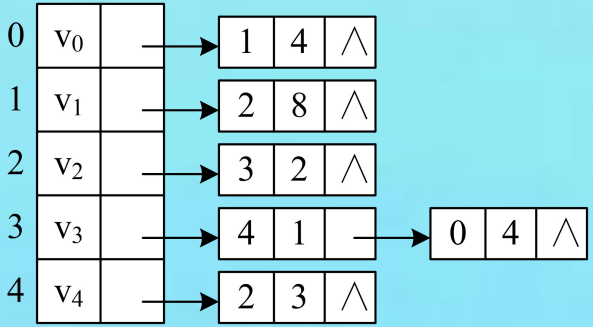
图的邻接矩阵存储结构

图的存储结构

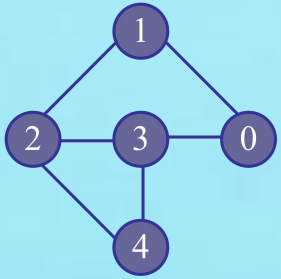
- 邻接矩阵存储方法
- 邻接表存储方法

$$A=\begin{bmatrix} 0 & 4 & \infty & \infty & \infty \\ \infty & 0 & 8 & \infty & \infty \\ \infty & \infty & 0 & 2 & \infty \\ 4 & \infty & \infty & 0 & 1 \\ \infty & \infty & 3 & \infty & 0 \end{bmatrix}$$



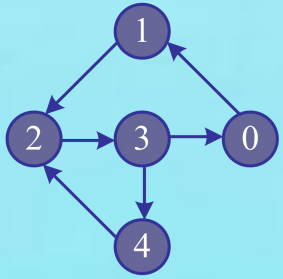


邻接矩阵存储方法



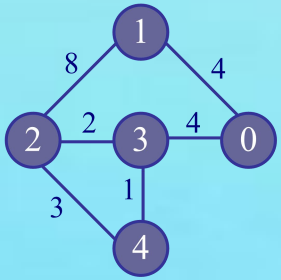
$$A=\begin{bmatrix}0&1&0&1&0\\1&0&1&0&0\\0&1&0&1&1\\1&0&1&0&1\\0&0&1&1&0\end{bmatrix}$$

(1)无向图 $A[i][j]=\begin{cases}1 & (i,j)\in E(G)\\ 0 & others\end{cases}$



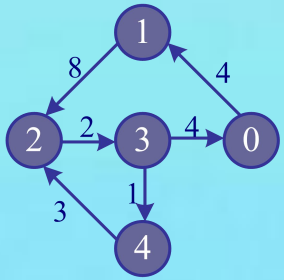
$$A=\begin{bmatrix}0&1&0&0&0\\0&0&1&0&0\\0&0&0&1&0\\1&0&0&0&1\\0&0&1&0&0\end{bmatrix}$$

(2)有向图 $A[i][j]=\begin{cases}1 & <i,j>\in E(G)\\ 0 & others\end{cases}$



$$A=\begin{bmatrix}0&4&\infty&4&\infty\\4&0&8&\infty&\infty\\\infty&8&0&2&3\\4&\infty&2&0&1\\\infty&\infty&3&1&0\end{bmatrix}$$

(3)带权无向图 $A[i][j]=\begin{cases}w_{ij} & i\neq j,(i,j)\in E(G)\\ 0 & i=j\\ \infty & others\end{cases}$



$$A=\begin{bmatrix}0&4&\infty&\infty&\infty\\\infty&0&8&\infty&\infty\\\infty&\infty&0&2&\infty\\4&\infty&\infty&0&1\\\infty&\infty&3&\infty&0\end{bmatrix}$$

(4)带权有向图 $A[i][j]=\begin{cases}w_{ij} & i\neq j,<i,j>\in E(G)\\ 0 & i=j\\ \infty & others\end{cases}$

邻接矩阵的特点

- 图的邻接矩阵表示是唯一的。
- 邻接矩阵的存储
  - 无向图的邻接矩阵一定是一个对称矩阵，可以考虑压缩存储。
  - 不少邻接矩阵是一个稀疏矩阵，当图的顶点较多时，可以采用三元组表的方法存储。
- 顶点的度
  - 对于无向图，邻接矩阵的第i行（或第i列）非零元素（或非∞元素）的个数正好是第i个顶点的度。
  - 对于有向图，邻接矩阵的第i行（或第i列）非零元素（或非∞元素）的个数正好是第i个顶点的出度（或入度）。
- 性能评价
  - 用邻接矩阵方法存储图，很容易确定图中任意两个顶点之间是否有边相连。
  - 要确定图中有多少条边，则必须按行、按列对每个元素进行检测，时间代价大。

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

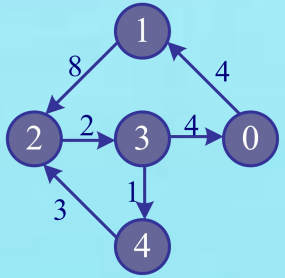
$$A=\begin{bmatrix}0&4&\infty&4&\infty\\4&0&8&\infty&\infty\\\infty&8&0&2&3\\4&\infty&2&0&1\\\infty&\infty&3&1&0\end{bmatrix}$$

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

$$A=\begin{bmatrix}0&4&\infty&\infty&\infty\\\infty&0&8&\infty&\infty\\\infty&\infty&0&2&\infty\\4&\infty&\infty&0&1\\\infty&\infty&3&\infty&0\end{bmatrix}$$

邻接矩阵的数据类型定义

```
#define MAXV   <最大顶点个数>
#define LIMITLESS 9999
typedef struct
{
    int no;                  //顶点编号
    InfoType info;           //顶点其他信息
} VertexType;                //顶点类型
typedef struct   //图的定义
{
    int n , e;                  //顶点数，边数
    int edges[MAXV][MAXV]; //邻接矩阵
    VertexType vexs[MAXV];//存放顶点信息
} MGraph;
```



$$A=\begin{bmatrix} 0 & 4 & \infty & \infty & \infty \\ \infty & 0 & 8 & \infty & \infty \\ \infty & \infty & 0 & 2 & \infty \\ 4 & \infty & \infty & 0 & 1 \\ \infty & \infty & 3 & \infty & 0 \end{bmatrix}$$

MGraph g;

edge

0	4	∞	∞	∞	…
∞	0	8	∞	∞	…
∞	∞	0	2	∞	…
4	∞	∞	0	1	…
∞	∞	3	∞	0	…
…	…	…	…	…	…

vexs

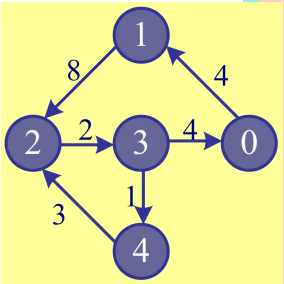
110	v0
114	v1
119	v2
120	v3
122	v4
…	…

n 5　　e 6

操作邻接矩阵

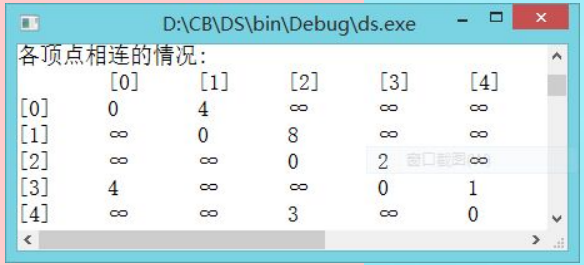
```
void CreateMGraph(MGraph *G)
{
   int i,j,k,w;
   printf("请输入顶点数和边数:");
   scanf("%d %d",&(G->n),&(G->e));
   printf("请输入顶点信息:\n");
   for (i=0; i<G->n; i++)
      scanf("%d %s",&(G->vexs[i].no),G->vexs[i].info);
   for (i=0; i<G->n; i++)
      for (j=0; j<G->n; j++)
      {
         if(i==j)
            G->edges[i][j]=0;
         else
            G->edges[i][j]=LIMITLESS;
      }
   printf("请输入:i j w:\n");
   for (k=0; k<G->e; k++)
   {
      scanf("%d %d %d",&i,&j,&w);
      G->edges[i][j]=w;
   }
}
```

```
#define LIMITLESS 9999
```




```
void DispMGraph(MGraph *G)
{
   int i,j;
   printf("顶点数: %d ，边数: %d\n", G->n, G->e);
   printf("%d 个顶点的信息：:\n", G->n);
   for (i=0; i<G->n; i++) /*输出顶点信息*/
      printf("%5d %5d %s\n", i, G->vexs[i].no, G->vexs[i].info);
   printf("各顶点相连的情况:\n");
   printf("\t");
   for (j=0; j<G->n; j++)
      printf("[%d]\t", j);
   printf("\n");
   for (i=0; i<G->n; i++)
   {
      printf("[%d]\t", i);
      for (j=0; j<G->n; j++)
      {
         if(G->edges[i][j]==LIMITLESS)
            printf("∞\t");
         else
            printf("%d\t", G->edges[i][j]);
      }
      printf("\n");
   }
}
```

$$A=\begin{bmatrix} 0 & 4 & \infty & \infty & \infty \\ \infty & 0 & 8 & \infty & \infty \\ \infty & \infty & 0 & 2 & \infty \\ 4 & \infty & \infty & 0 & 1 \\ \infty & \infty & 3 & \infty & 0 \end{bmatrix}$$




```
int main()
{
   MGraph *g;
   g = (...*)malloc(sizeof(MGraph));
   CreateMGraph(g);
   DispMGraph(g);
   return 0;
}
```

思考题

- （1）对于有n个顶点e条边的无向图，邻接矩阵表示时有多少个元素，多少个非0元素？
- （2）对于有n个顶点e条边的有向图，邻接矩阵表示时有多少个元素，多少个非0元素？