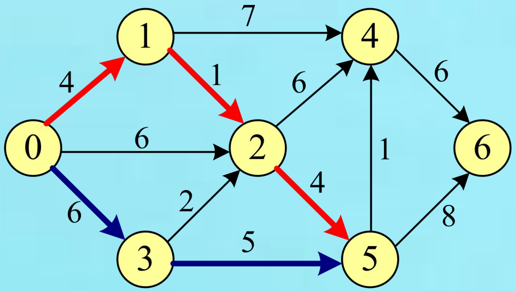
本节主题：

每对顶点之间的最短路径

问题及方案

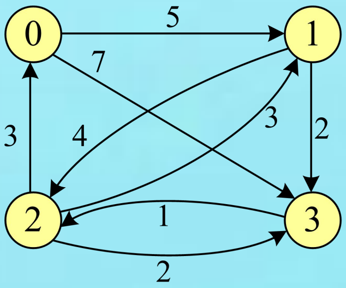
- 问题
  - 对于一个各边权值均大于零的有向图，对每一对顶点i≠j，求出顶点i与顶点j之间的最短路径和最短路径长度。
- 简单分析
  - 以每个顶点作为源点循环求出每对顶点之间的最短路径
  - 采用Dijkstra算法，循环n次
  - 复杂度O(n^3)
- 本讲内容
  - 弗洛伊德（Floyd）算法
  - 依然O(n^3)　，又一种风味



Robert W Floyd

问题描述及算法思想

- 问题描述
  - 有向带权图G=(V，E)
  - 采用邻接矩阵存储
  - 设置二维数组A用于存放当前顶点之间的最短路径长度，分量A[i][j]表示当前顶点i到顶点j的最短路径长度。
- 弗洛伊德算法思想
  - 递推产生一个矩阵序列A_0，A_1，…，A_k，…，A_{n-1}，其中A_k[i][j]表示从顶点i到顶点j的路径上所经过的顶点编号不大于k的最短路径长度。



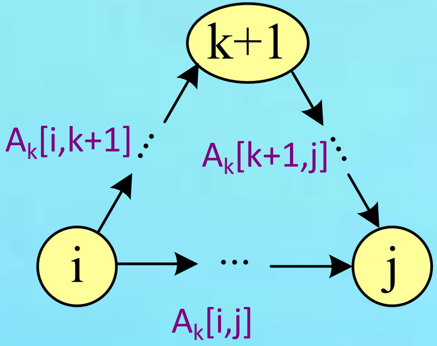
$$A_{-1}=\begin{bmatrix} 0 & 5 & \infty & 7 \\ \infty & 0 & 4 & 2 \\ 3 & 3 & 0 & 2 \\ \infty & \infty & 1 & 0 \end{bmatrix}$$

初始时，A_{-1}[i][j]=g.edges[i][j](cost[i][j])

A_{-1}[i][j]=cost[i][j]

A_{k+1}[i][j]=MIN{A_k[i][j], A_k[i][k+1]+A_k[k+1][j]}　(-1≤k≤n-2)

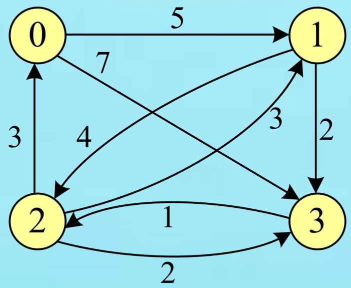
迭代解法：已考虑 0、1、……k 这k+1个顶点的基础上，再考虑顶点k+1，以此得到A_{k+1}



求解过程(0)

$A_{-1}[i][j]=cost[i][j]$

$A_{k+1}[i][j]=MIN\{A_k[i][j], A_k[i][k+1]+A_k[k+1][j]\}$ $(-1 \le k \le n-2)$



cost

$$\begin{bmatrix} 0 & 5 & \infty & 7 \\ \infty & 0 & 4 & 2 \\ 3 & 3 & 0 & 2 \\ \infty & \infty & 1 & 0 \end{bmatrix}$$

$$A_{-1}=\begin{bmatrix} 0 & 5 & \infty & 7 \\ \infty & 0 & 4 & 2 \\ 3 & 3 & 0 & 2 \\ \infty & \infty & 1 & 0 \end{bmatrix} \quad Path_{-1}=\begin{bmatrix} -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \end{bmatrix}$$

path[i][j]=x，表示i点到j要经过x

path[i][j]=-1，表示i点直接到j，没有中间顶点

考虑顶点0，$A_0[i][j]$表示由i到j、经由顶点0的最短路径

$A_0[i][j]=MIN\{A_{-1}[i][j], A_{-1}[i][0]+A_{-1}[0][j]\}$

$$A_0=\begin{bmatrix} 0 & 5 & \infty & 7 \\ \infty & 0 & 4 & 2 \\ 3 & 3 & 0 & 2 \\ \infty & \infty & 1 & 0 \end{bmatrix} \quad Path_0=\begin{bmatrix} -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \end{bmatrix}$$

2-0-1：不改变

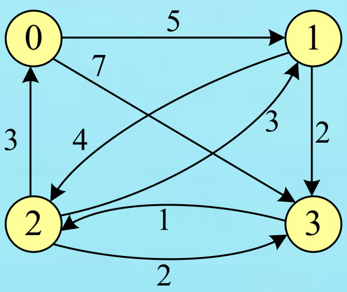
2-0-3：不改变

3-2-0-1：不改变

求解过程(1)

$A_{-1}[i][j]=cost[i][j]$

$A_{k+1}[i][j]=MIN\{A_k[i][j],\ A_k[i][k+1]+A_k[k+1][j]\}\quad(-1\le k\le n-2)$



cost

$$\begin{bmatrix} 0 & 5 & \infty & 7 \\ \infty & 0 & 4 & 2 \\ 3 & 3 & 0 & 2 \\ \infty & \infty & 1 & 0 \end{bmatrix}$$

$$A_0=\begin{bmatrix} 0 & 5 & \infty & 7 \\ \infty & 0 & 4 & 2 \\ 3 & 3 & 0 & 2 \\ \infty & \infty & 1 & 0 \end{bmatrix}\qquad Path_0=\begin{bmatrix} -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \end{bmatrix}$$

考虑顶点1，$A_1[i][j]$表示由i到j、经由顶点1的最短路径

$A_1[i][j]=MIN\{A_0[i][j],\ A_0[i][1]+A_0[1][j]\}$

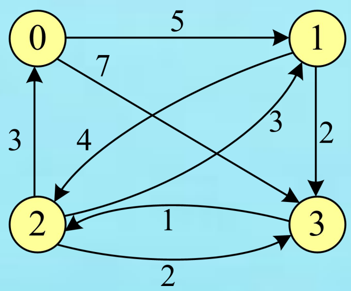
$$A_1=\begin{bmatrix} 0 & 5 & 9 & 7 \\ \infty & 0 & 4 & 2 \\ 3 & 3 & 0 & 2 \\ \infty & \infty & 1 & 0 \end{bmatrix}\qquad Path_1=\begin{bmatrix} -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 \end{bmatrix}$$

0-1-2 路径长度为9，
将A[0][2]改为9，
path[0][2]改为1。

求解过程(2)

$A_{-1}[i][j]=cost[i][j]$

$A_{k+1}[i][j]=MIN\{A_k[i][j],\ A_k[i][k+1]+A_k[k+1][j]\}\quad(-1\le k\le n-2)$



$$A_1=\begin{bmatrix}0&5&9&7\\ \infty&0&4&2\\ 3&3&0&2\\ \infty&\infty&1&0\end{bmatrix}\qquad Path_1=\begin{bmatrix}-1&-1&1&-1\\ -1&-1&-1&-1\\ -1&-1&-1&-1\\ -1&-1&-1&-1\end{bmatrix}$$

cost

$$\begin{bmatrix}0&5&\infty&7\\ \infty&0&4&2\\ 3&3&0&2\\ \infty&\infty&1&0\end{bmatrix}$$

考虑顶点2，$A_2[i][j]$表示由i到j、经由顶点2的最短路径

$A_2[i][j]=MIN\{A_1[i][j],\ A_1[i][2]+A_1[2][j]\}$

$$A_2=\begin{bmatrix}0&5&9&7\\ 7&0&4&2\\ 3&3&0&2\\ 4&4&1&0\end{bmatrix}\qquad Path_2=\begin{bmatrix}-1&-1&1&-1\\ 2&-1&-1&-1\\ -1&-1&-1&-1\\ 2&2&-1&-1\end{bmatrix}$$

3-2-0：长度为4，将A[3][0]改为4；将path[3][0]改为2。

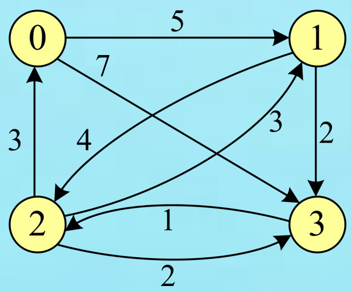
3-2-1：长度为4，将A[3][1]改为4；将path[3][1]改为2。

1-2-0：长度为7，将A[1][0]改为7；将path[1][0]改为2。

求解过程(3)

$A_{-1}[i][j]=cost[i][j]$

$A_{k+1}[i][j]=MIN\{A_k[i][j],\ A_k[i][k+1]+A_k[k+1][j]\}\quad(-1\le k\le n-2)$



$$A_2=\begin{bmatrix}0&5&9&7\\7&0&4&2\\3&3&0&2\\4&4&1&0\end{bmatrix}\qquad Path_2=\begin{bmatrix}-1&-1&1&-1\\2&-1&-1&-1\\-1&-1&-1&-1\\2&2&-1&-1\end{bmatrix}$$

cost

$$\begin{bmatrix}0&5&\infty&7\\\infty&0&4&2\\3&3&0&2\\\infty&\infty&1&0\end{bmatrix}$$

考虑顶点3，$A_3[i][j]$表示由i到j、经由顶点3的最短路径

$A_3[i][j]=MIN\{A_2[i][j],\ A_2[i][3]+A_2[3][j]\}$

$$A_3=\begin{bmatrix}0&5&8&7\\6&0&3&2\\3&3&0&2\\4&4&1&0\end{bmatrix}\qquad Path_3=\begin{bmatrix}-1&-1&3&-1\\3&-1&3&-1\\-1&-1&-1&-1\\2&2&-1&-1\end{bmatrix}$$

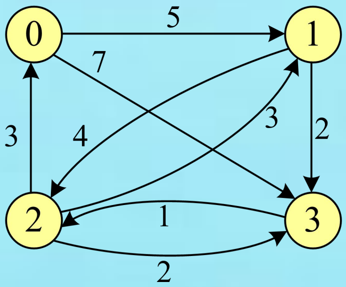
0-3-2：长度为8，
A[0][2]改为8；

1-3-2-0：长度为6，
A[1][0]改为6；

1-3-2：长度为3，
A[1][2]改为3。

将path[0][2],path[1][0]和path[1][2]均改为3。

求解结果

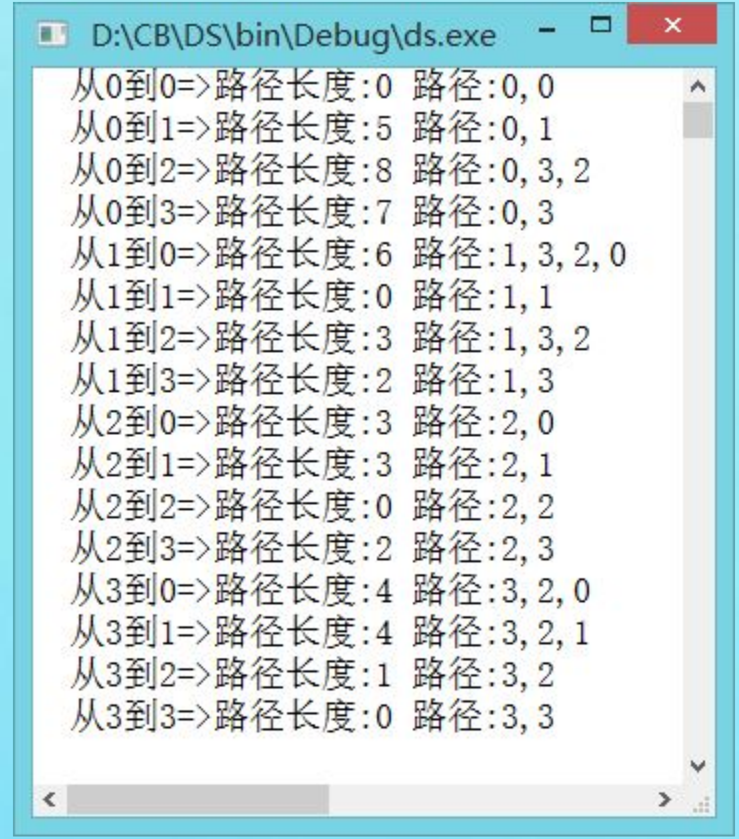


$$A=\begin{bmatrix}0 & 5 & 8 & 7\\ 6 & 0 & 3 & 2\\ 3 & 3 & 0 & 2\\ 4 & 4 & 1 & 0\end{bmatrix}$$

$$Path=\begin{bmatrix}-1 & -1 & 3 & -1\\ 3 & -1 & 3 & -1\\ -1 & -1 & -1 & -1\\ 2 & 2 & -1 & -1\end{bmatrix}$$

path[i][j]=x，表示i点到j要经过x

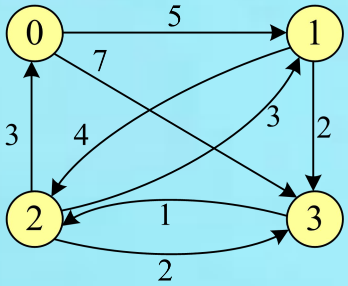
path[i][j]=-1，表示i点直接到j，没有中间顶点



算法实现

```
void Floyd(MGraph g)
{
  //定义辅助存储并赋初值
    int A[MAXV][MAXV],path[MAXV][MAXV];
    int i,j,k;
    for (i=0; i<g.n; i++)
       for (j=0; j<g.n; j++)
       {
          A[i][j]=g.edges[i][j];
          path[i][j]=-1;
       }
  //迭代更新数据
  for (k=0; k<g.n; k++)
  {
     for (i=0; i<g.n; i++)
        for (j=0; j<g.n; j++)
           if (A[i][j]>A[i][k]+A[k][j])
           {
              A[i][j]=A[i][k]+A[k][j];
              path[i][j]=k;
           }
  }
  //输出最短路径
  Dispath(A,path,g.n);
}
```

$A_{k+1}[i][j]=\mathrm{MIN}\{A_k[i][j],\ A_k[i][k+1]+A_k[k+1][j]\}\quad(-1\le k\le n-2)$



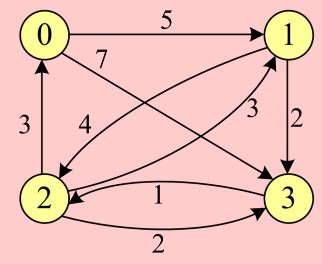
$$A_{-1}=\begin{bmatrix}0 & 5 & \infty & 7\\ \infty & 0 & 4 & 2\\ 3 & 3 & 0 & 2\\ \infty & \infty & 1 & 0\end{bmatrix}$$

$$Path_{-1}=\begin{bmatrix}-1 & -1 & -1 & -1\\ -1 & -1 & -1 & -1\\ -1 & -1 & -1 & -1\\ -1 & -1 & -1 & -1\end{bmatrix}$$

输出结果

```
void Dispath(int A[][MAXV],int path[][MAXV],int n)
{
   int i,j;
   for (i=0; i<n; i++)
      for (j=0; j<n; j++)
      {
         if (A[i][j]==INF)
         {
            if (i!=j)
               printf("从%d到%d没有路径\n",i,j);
         }
         else
         {
            printf("  从%d到%d=>路径长度:%d 路径:",i,j,A[i][j]);
            printf("%d,",i);   //输出路径上的起点
            Ppath(path,i,j);  //输出路径上的中间点
            printf("%d\n",j);//输出路径上的终点
         }
      }
}
```

```
void Ppath(int path[][MAXV],int i,int j)
{
   int k;
   k=path[i][j];
   if (k==-1) return;
   Ppath(path,i,k);
   printf("%d,",k);
   Ppath(path,k,j);
}
```



$$A=\begin{bmatrix}0&5&8&7\\6&0&3&2\\3&3&0&2\\4&4&1&0\end{bmatrix}$$

$$Path=\begin{bmatrix}-1&-1&3&-1\\3&-1&3&-1\\-1&-1&-1&-1\\2&2&-1&-1\end{bmatrix}$$

思考题

- 求一个带权有向图中所有顶点之间的最短路径可以采用Dijkstra算法，循环n次即可，其时间复杂度为O(n3)，而Floyd算法的时间复杂度也为O(n3)。两者有什么不同？