最短路径算法—Floyd(弗洛伊德)算法分析与实现(C/C++)

提醒: 此博客暂时不再更新。

- 新博客:http://blog.tankywoo.com
- 个人 Wiki: http://wiki.tankywoo.com
- 个人主页:http://tankywoo.com
 - 上一篇讲过 Dijkstra 算法:

http://www.wutianqi.com/?p=1890

Floyd-Warshall 算法,简称 Floyd 算法,用于求解任意两点间的最短距离,时间复杂度为 $O(n^3)$ 。 使用条件&范围

通常可以在任何图中使用,包括有向图、带负权边的图。

Floyd-Warshall 算法用来找出每对点之间的最短距离。它需要用邻接矩阵来储存边,这个算法通过考虑最佳子路径来得到最佳路径。

- 1.注意单独一条边的路径也不一定是最佳路径。
- **2**.从任意一条单边路径开始。所有两点之间的距离是边的权,或者无穷大,如果两点之间没有边相连。

对于每一对顶点 \mathbf{u} 和 \mathbf{v} ,看看是否存在一个顶点 \mathbf{w} 使得从 \mathbf{u} 到 \mathbf{w} 再到 \mathbf{v} 比己知的路径更短。如果是更新它。

3.不可思议的是,只要按排适当,就能得到结果。 伪代码:

```
1 // dist(i,j) 为从节点i到节点j的最短距离
2 For i←1 to n do
3    For j←1 to n do
4         dist(i,j) = weight(i,j)
5
6 For k←1 to n do // k 为"媒介节点"
7    For i←1 to n do
8         For j←1 to n do
9         if (dist(i,k) + dist(k,j) < dist(i,j)) then // 是否是更短的路径?
10         dist(i,j) = dist(i,k) + dist(k,j)</pre>
```

我们平时所见的 Floyd 算法的一般形式如下:

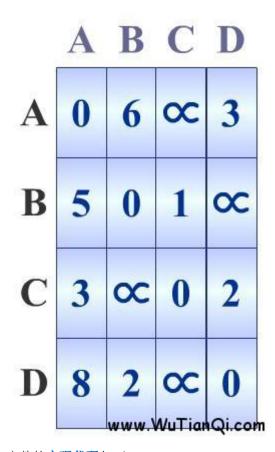
```
1 void Floyd() {
2    int i,j,k;
3    for(k=1;k<=n;k++)
4    for(i=1;i<=n;i++)
5    for(j=1;j<=n;j++)
6        if(dist[i][k]+dist[k][j]<dist[i][j])
7        dist[i][j]=dist[i][k]+dist[k][j];
8 }</pre>
```

注意下第 6 行这个地方,如果 dist[i][k]或者 dist[k][j]不存在,程序中用一个很大的数代替。最好写成 if(dist[i][k]!=INF && dist[k][j]!=INF && dist[i][k]+dist[k][j]< dist[i][j]),从而防止溢出所造成的错误。<p=""></dist[i][j]),从而防止溢出所造成的错误。<>

Floyd 算法的实现以及输出最短路径和最短路径长度,具体过程请看【动画演示 Floyd 算法】。 代码说明几点:

- 1、AIII数组初始化为各顶点间的原本距离,最后存储各顶点间的最短距离。
- 2、path[][]数组保存最短路径,与当前迭代的次数有关。初始化都为-1,表示没有中间顶点。在求 A[i][j]过程中,path[i][j]存放从顶点 vi 到顶点 vj 的中间顶点编号不大于 k 的最短路径上前一个结点的编号。在算法结束时,由二维数组 path 的值回溯,可以得到从顶点 vi 到顶点 vj 的最短路径。

初始化 A[][]数组为如下,即有向图的邻接矩阵。



完整的**实现代码**如下:

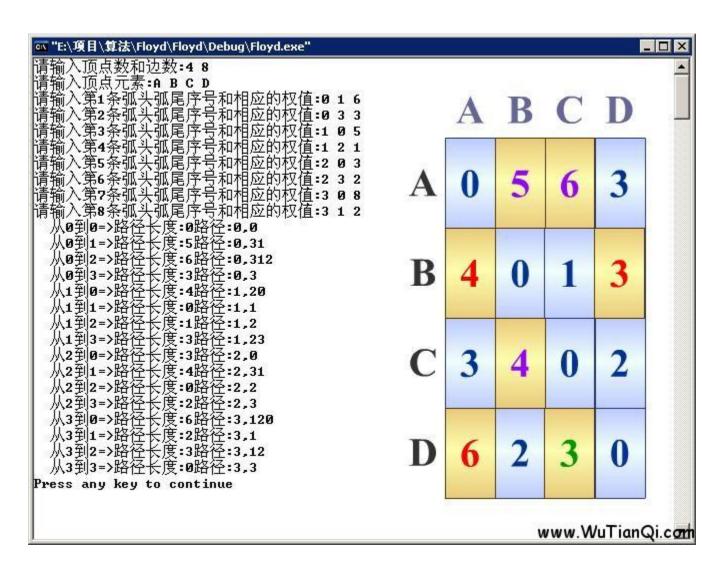
- 1 #include <iostream>
 2 #include <string>
- 3 #include <stdio.h>
- 4 using namespace std;
- 5 #define MaxVertexNum 100
- **6** #define INF 32767
- 7 typedef struct

```
8 {
9
     char vertex[MaxVertexNum];
10
     int edges[MaxVertexNum] [MaxVertexNum];
     int n,e;
11
12 }MGraph;
13
14 void CreateMGraph (MGraph &G)
15 {
16
    int i,j,k,p;
17
     cout<<"请输入顶点数和边数:";
     cin>>G.n>>G.e;
18
19
    cout<<"请输入顶点元素:";
     for (i=0;i<G.n;i++)
20
21
      cin>>G.vertex[i];
22
23
     }
     for (i=0;i<G.n;i++)
24
     {
25
        for (j=0;j<G.n;j++)
26
27
28
            G.edges[i][j]=INF;
29
           if (i==j)
30
           {
               G.edges[i][j]=0;
31
32
       }
33
     }
34
35
    for (k=0; k< G.e; k++)
36
37
         cout<<"请输入第"<<k+1<<"条弧头弧尾序号和相应的权值:";
        cin>>i>>j>>p;
38
        G.edges[i][j]=p;
39
40
     }
41 }
42 void Dispath(int A[][MaxVertexNum],int path[][MaxVertexNum],int n);
43
44 void Floyd (MGraph G)
45 {
47 A[MaxVertexNum] [MaxVertexNum], path [MaxVertexNum] [MaxVertexNum];
          int i,j,k;
48
          for (i=0;i<G.n;i++)
49
           {
50
51
                for (j=0;j<G.n;j++)
```

```
52
53
                               A[i][j]=G.edges[i][j];
54
                               path[i][j]=-1;
55
56
            for (k=0; k<G.n; k++)</pre>
57
58
                      for (i=0;i<G.n;i++)</pre>
59
60
                               for (j=0;j<G.n;j++)</pre>
61
62
63
                                         if (A[i][j]>A[i][k]+A[k][j])
64
                                                  A[i][j]=A[i][k]+A[k][j];
65
66
                                                  path[i][j]=k;
67
68
                     }
69
70
            Dispath(A, path, G.n);
71
72 }
73
74 void Ppath(int path[][MaxVertexNum], int i, int j)
75 {
            int k;
76
            k=path[i][j];
77
            if (k==-1)
78
79
80
                      return;
81
            Ppath(path,i,k);
82
            printf("%d,",k);
83
84
            Ppath(path,k,j);
85 }
87 void Dispath(int A[][MaxVertexNum],int path[][MaxVertexNum],int n)
88 {
            int i,j;
89
            for (i=0;i<n;i++)</pre>
90
91
92
                      for (j=0;j<n;j++)</pre>
93
                               if (A[i][j]==INF)
94
95
```

```
if (i!=j)
 96
 97
                                         {
                                                 printf("从%d 到%d 没有路径
 98
99 \n",i,j);
                                        }
100
101
                               }
102
                               else
103
                               {
104
                                        printf(" 从%d 到%d=>路径长度:%d 路
105 <mark>径:",</mark>i,j,A[i][j]);
106
                                        printf("%d,",i);
107
                                        Ppath(path,i,j);
108
                                        printf("%d\n",j);
109
                               }
110
                      }
111
            }
112 }
113
114 int main()
115 {
             freopen("input2.txt", "r", stdin);
116
117
             MGraph G;
118
             CreateMGraph(G);
             Floyd(G);
             return 0;
```

测试结果如下:



本文主要来至网上资料各种杂糅:

- 1.http://www.cnblogs.com/MiYu/archive/2010/09/25/1834826.html
- 2.http://blog.csdn.net/akof1314/archive/2009/08/09/4424946.aspx
- 3.http://www.nocow.cn/index.php/Floyd-Warshall%E7%AE%97%E6%B3%95