Floyd 算法在物流线路中最短路径的研究

徐建军　王森
南京师范大学泰州学院　江苏泰州　225300

[摘　要] 两点之间的最短路径算法是物流配送系统涉及的最基本算法。基于 Floyd 算法算法的基本原理，提出一种物流配送系统最短路径设计。该设计使用 C 语言程序代码实现，经过测试，使用 Floyd 算法求得最短路径，具有使用便捷、简单迅速的特征。通过该算法设计可以获得任意多个结点之间的最佳路径，从而能有效提高配送效率，降低配送成本。

[关键词]Floyd 算法；物流；最短路径

引言

在物流配送时，配送员总是会希望能早点到达目的地，因此人们会去寻找尽可能短的路线。列举出所有的路线进行规划，计算对比每条路线所经过的长度然后选择最佳路线出行。这种方法看起来可执行，但是对于较多较复杂的地方，即使排除多余不必要的路线，仍然存在百万种可通行的路线，如果这时候存在一种最短路径算法，这类问题便可以得到很好地解决。

对于任意起始点都不固定而随后有可能去其他任一地点，我们在计算路线长短时都可以使用 Floyd 算法。该算法使用了动态规划的思想，将问题逐步解决从而求得最短路径。

Floyd 算法的应用性很强，既可以方便我们的路程规划，我们不必再为出行路线担忧，选择捷径少走弯路；也可以方便我们解决任意两点之间的最短路径，省时省力，使我们的研究变得简单起来。

1 Floyd 算法与算法设计

1.1 算法介绍

1）算法简介：Floyd 算法，也称弗洛伊德算法，或称插点法。该算法可以找出所给有向图或无向图中多源点的最短路径，运用的思想是动态规划。

2）核心思想：在给定的有向图或者无向图中，我们统计顶点数，记录各点与各点之间的路线距离，建立图的权值矩阵，再通过 Floyd 算法求得路线矩阵，从而求得最短路径。由第一个带权邻接矩阵开始更新，每更新一次便会得到一个新的矩阵。假设更新次数为 n，我们可以得到 n 个矩阵，而最后得到的矩阵的行列便是两顶点之间最短的距离长度，该矩阵我们也称之为距离矩阵。对于两点之间最短距离的记录，我们可以使用后继节点。

3）优缺点分析：

优点：Floyd 算法使用三重遍历循环，代码简短有效，理解起来不难。Floyd 算法使用范围很广，其中边权无论正负都可使用。

缺点：时间复杂度比较高。

4）算法描述：

核心代码：

```
For k:=1 to n
For i:=1 to n
For j:=1 to n
If D[i,j]>D[i,k]+D[k,j] Then
D[i,j]:=D[i,k]+D[k,j];
```

1.2 算法设计

1.2.1 算法步骤

1）初始化距离矩阵，修改边的权重。

2）初始化路由矩阵，记录任意两点之间最短路径信息。

3）进行三重循环遍历，更新距离矩阵和路由矩阵。

4）输出结果。

1.2.2 算法流程图与算法模型

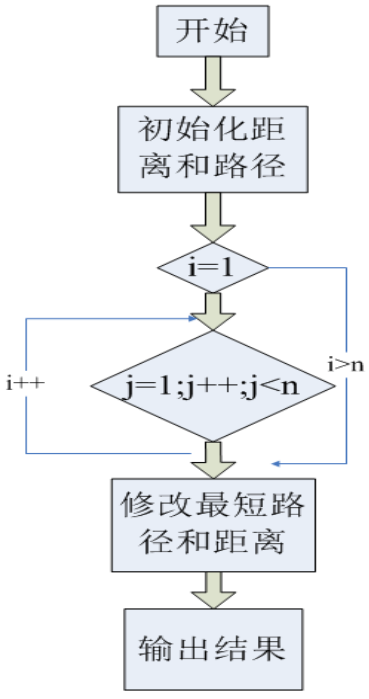

图 1.2.1 算法流图

★基金项目：江苏省高等学校大学生创新创业训练计划项目，项目名称：顾及多路径的订单配送优化模型研究。

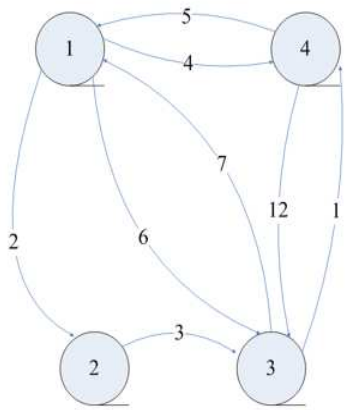

图 1.2.2 算法模型

如图 1.2.2 已知两个城市的四个交通地点以及它们任意两点之间的距离，希望可以找出任意两点之间的最短路径以及途经点。建立二维数组 A 来存储信息。

	1	2	3	4
1	0	2	6	4
2	∞	0	3	∞
3	7	∞	0	1
4	5	∞	12	0

2 Floyd 算法程序实现

2.1 程序伪代码：

```
void Floyd（Graph G）//G-- 图
int A[NUMS][NUMS],path[NUMS][NUMS];
    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;
    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;
            Dispath（A,path,G.n）；
void Ppath（int path[][NUMS],int i,int j）
    int k;
    k=path[i][j];
    if（k==-1）
        return;
    Ppath（path,i,k）；
    printf（“%d,”,k+1）；
    Ppath（path,k,j）；
void Dispath（int A[][NUMS],int path[][NUMS],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+1,j+1);
         else
          printf（“从%d到%d=>最短路径长度为：%d,路径站点为 :”,i+1,j+1,A[i][j]）；
                printf（“%d,”,i+1）；
                Ppath（path,i,j）；
                printf（“%d\n”,j+1）；
```

2.2 程序模拟仿真

通过输入四个地点之间的权值来类比物流货物重量。运行程序，结果如下图：

```
0       2       6       4
65535   0       3       65535
7       65535   0       1
5       65535   12      0
从1到1=>最短路径长度为：0,路径站点为 :1,1
从1到2=>最短路径长度为：2,路径站点为 :1,2
从1到3=>最短路径长度为：5,路径站点为 :1,2,3
从1到4=>最短路径长度为：4,路径站点为 :1,4
从2到1=>最短路径长度为：9,路径站点为 :2,3,4,1
从2到2=>最短路径长度为：0,路径站点为 :2,2
从2到3=>最短路径长度为：3,路径站点为 :2,3
从2到4=>最短路径长度为：4,路径站点为 :2,3,4
从3到1=>最短路径长度为：6,路径站点为 :3,4,1
从3到2=>最短路径长度为：8,路径站点为 :3,4,1,2
从3到3=>最短路径长度为：0,路径站点为 :3,3
从3到4=>最短路径长度为：1,路径站点为 :3,4
从4到1=>最短路径长度为：5,路径站点为 :4,1
从4到2=>最短路径长度为：7,路径站点为 :4,1,2
从4到3=>最短路径长度为：10,路径站点为 :4,1,2,3
从4到4=>最短路径长度为：0,路径站点为 :4,4
```

图 2.2.1 Floyd 物流仿真程序结果图

通过图 2.2.1 可以找出找出各物流点之间的最优路径，从运行结果我们便能找出任意两地点之间的最短距离。假定数据量较大较繁琐，要解决多个复杂点之间的最短距离时，依旧可以使用该程序，使用该算法，以求得最优解。

2.3 算法实例验证

Floyd 算法涉及的领域较广，应用也较为广泛，一般在处理最

短距离问题，极值问题，数学建模问题都非常具有优势。问题千变万化，需求也是各有不同，但是利用 Floyd 算法解决这类问题的原理确是大同小异。

例如下面这个问题，在运输货物过程中，路线及限重值如图所示，要求运载车辆在不超过限重量的情况下，使得运载货物的重量最大。

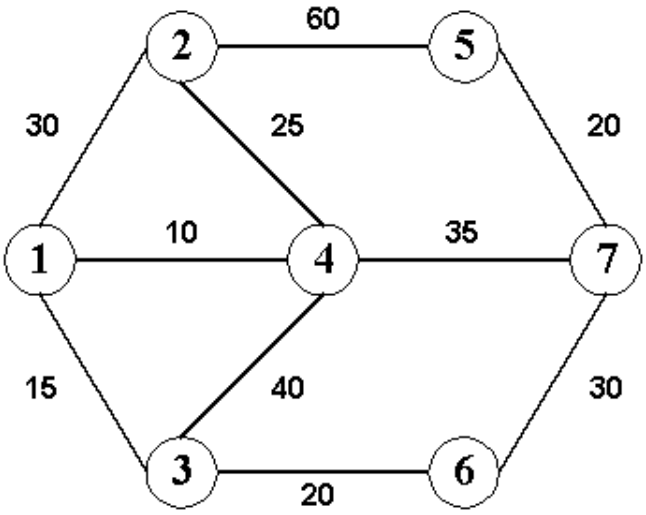

图 2.3.1 物流配送点及配送最大承重

图 2.3.1 中一共有七个结点，列出权重表格，使用 Floyd 算法进行计算。

（1）列表，权重表格如下：

	1	2	3	4	5	6	7
1	0	30	15	10	∞	∞	∞
2	30	0	∞	25	60	∞	∞
3	15	∞	0	40	∞	20	∞
4	10	25	40	0	∞	∞	35
5	∞	60	∞	∞	0	∞	20
6	∞	∞	20	∞	∞	0	30
7	∞	∞	∞	35	20	30	0

（2）建立文本文档：

```
0       30      15      10      65536   65536   65536
30      0       65536   25      60      65536   65536
15      65536   0       40      65536   20      65536
10      25      40      0       65536   65536   35
65536   60      65536   65536   0       65536   20
65536   65536   20      65536   65536   0       30
65536   65536   65536   35      20      30      0
```

（3）运行结果：

通过编译运行，我们可以得到任意两位置之间，货车运载的承重量，找到最大的承载值。

```
0       30      15      10      65536   65536   65536
30      0       65536   25      60      65536   65536
15      65536   0       40      65536   20      65536
10      25      40      0       65536   65536   35
65536   60      65536   65536   0       65536   20
65536   65536   20      65536   65536   0       30
65536   65536   65536   35      20      30      0
从1到1运载的最大重量为：0，路线为:1,1

从1到2运载的最大重量为：30，路线为:1,2

从1到3运载的最大重量为：15，路线为:1,3

从1到4运载的最大重量为：10，路线为:1,4

从1到5运载的最大重量为：65，路线为:1,4,7,5

从1到6运载的最大重量为：35，路线为:1,3,6

从1到7运载的最大重量为：45，路线为:1,4,7

从2到1运载的最大重量为：30，路线为:2,1

从2到2运载的最大重量为：0，路线为:2,2

从2到3运载的最大重量为：45，路线为:2,1,3

从2到4运载的最大重量为：25，路线为:2,4

从2到5运载的最大重量为：60，路线为:2,5
```

图 2.3.2 最大承重量算法结果

3 结论

Floyd 算法在多源路径寻找最短距离的问题中很有优势。对于其他算法搜索速度慢，效率低，时间花费多的缺陷，还无法加载权值。Floyd 算法的优点显得尤为突出，它的核心代码并不长，使用动态规划的思想经过三重遍历循环便能找出任意两点之间的最短路径。本文设计的算法模型是找出四个交通点之间的最短距离，通过改变文件的权值内容，把权值转换为物流配送最大承重量，仿真并实践检验 Floyd 算法，发现该算法在物流配送中优势突出。

参考文献：

[1] 宁正元．算法与数据结构．北京：清华大学出版社 .2006

[2] 邹海明，佘祥宣．计算机算法基础．湖北：华工中学院出版社 .1985

[3] 吕国英．算法设计与分析．北京：清华大学出版社，2006

[4] 郝自军，何尚录．最短路径问题 floyd 算法的若干讨论．重庆：重庆工学院院报 .2008

[5] 谭浩强．C++ 程序设计．北京：清华大学出版社 .2006