(4) 20 — 25

一种新的优化算法——F-D 算法*

0157.5
O224

聂　黎　　俞集辉
（重庆大学电气工程系，重庆，400044；第一作者 26 岁，男，硕士）

摘　要　针对实际工程中的优化问题，将 Floyd 算法和 Dijkstra 算法结合起来，形成一种用以求解无向图中部分顶点间最短路径的新优化算法——F-D 算法，并用算例验证 F-D 算法的计算效率。

关键词　图；优化；算法
中国图书资料分类法分类号　TM11；O224

F-D 算法

0　引　　言

实际工程中存在着大量的优化问题，如电缆敷设设计中为减少电缆用量需选择最短路径；城市交通中恰当地选择交通线路等问题，都可以等价于求一个无向图中部分顶点间的最短路径问题。目前，常采用 Floyd 算法计算图中部分顶点之间的最短路径，并确定优化方案[1]。但由于不可预知的因素或实际条件的限制，可能出现路径必须经过图中的一个或几个顶点，或图中的某一条或某几条边不能用来构成路径等约束条件。这些是采用一次 Floyd 算法的计算结果所不能满足的。

最优方案的难以确定，往往是由于最短路径所经顶点中某一对顶点或某些顶点之间现有的邻接关系被改变，由邻接变为不邻接[2]。此种情况下，在修改邻接主关系改变的顶点对所对应的权矩阵的元素后，如果再利用 Floyd 算法重新寻找新的最短路径，其结果只是对邻接关系改变的顶点对之间的路径作了修改，而大部分原有路径保持不变，浪费了许多时间，作了大量不必要的运算。

Dijkstra 算法适用于计算图中一对顶点之间的最短路径；计算多对顶点之间的最短路径时，采用 Floyd 算法能节省计算时间[3,4]。如果综合 Dijkstra 算法和 Floyd 算法之长，可以形成一种新的算法：用 Floyd 算法求得无向图中多对顶点间的最短路径，再根据最短路径模拟实施优化方案，在方案实施过程中，如果某种因素（如约束条件）发生作用，致使路径中少数顶点之间的邻接关系发生变化，这时只需确定邻接关系被改变的顶点对，用 Dijkstra 算法求得这些顶点对之间的最短路径，再加上其余部分路径，就能得到图中各对顶点间新的最短路径。这就是 F-D 算法的基本思想。

1　F-D 算法

* 收文日期　1997-03-14

1.1　用 Floyd 算法求得所需的最短路径

设 G 是有 n 个顶点无向图，其每条边都被赋予权值。要求计算图中 m 对不相邻接的顶点之间的最短路径。

1) 建立权矩阵 $W=[w_{ij}]$.

2) 在 W 中找出所有大于 $w_{i1}+w_{1j}$ 的元素，将它们用 $w_{i1}+w_{1j}$ 取代，得到新矩阵 $W_1=[w_{ij}^{(1)}]$；在 W_1 中找出所有大于 $w_{i2}^{(1)}+w_{2j}^{(1)}$ 的元素，将它们用 $w_{i2}^{(1)}+w_{2j}^{(1)}$ 取代，得到新的矩阵 $W_1=[w_{ij}^{(2)}]$；按这样的方法一直进行下去，求出 $W_n=[w_{ij}^{(n)}]$，得到图中所有顶点之间的最短路径。

1.2　得到所需的路径并进行检验

假设起点为 p，终点为 q，则最短路径长度为 $w_{pq}^{(n)}$，路径所经顶点为：$p,l,\cdots,i,j,k,\cdots,m,q$，该路径包含的边为：$e(p,l),\cdots,e(i,j),e(j,k),\cdots,e(m,q)$.

根据实际情况模拟实施优化方案，具体做法如下：由实际问题确定检验标准，将优化方案的模拟实施转化为对所得最短路径的检验。检验的方法是：依次对 $e(p,l),\cdots,e(i,j),e(j,k),\cdots,e(m,q)$ 进行检验，如果路径中的每条边均满足实际问题的要求，说明顶点 p 到顶点 q 的最短路径是正确的，算法转向步骤 1.4，否则算法往下进行。

1.3　用 Dijkstra 算法修正路径 $e(i,j)$

如果顶点 p 到顶点 q 的最短路径的某条边 $e(i,j)$ 不能满足实际问题的要求，则令 $w_{ij}^{(0)}=\infty$.

1) 令顶点 i 标号为 0，其它顶点标号为 ∞. 称顶点 i 为定标顶点，其它顶点为未定标顶点。

2) 对所有未定标的顶点给出暂时标号，暂时标号的值由下式确定：暂时标号 $=\min[x$ 的旧标号，(y 的旧标号 $+w_{xy}$)，$(x=1,2,\cdots,n)$，y 是前一步刚被定标的顶点。

3) 找出所有暂时标号的最小值，用它作为相应顶点的定标号。如果存在几个有同一最小标号值的顶点，则可任选一个加以定标。

重复 2)、3)，直至指定的终点 j 被定标为止，得到顶点 i,j 之间的最短路径长度(j 的标号值) 和路径所经顶点 $i,\cdots,r,\cdots,j$.

修正顶上点 p 到顶点 q 的最短路径长度和路径所经顶点，重新对路径进行检验。

1.4　将经过检验的路径作为 p 到 q 的最短路径

重复步骤 1.2 ～ 1.4，直至求出所有 m 对顶点间的最短路径。

F-D 算法的计算流程图如图 1 所示。图中 N 为无向图 G 的顶点数，m 为需要计算的顶点对数，$qd(x),zd(x)(x=1,2,\cdots,N)$ 分别记录所求路径的起点和终点，$LJCD$ 表示所求最短路径的长度，$ZDLJ$ 表示最短路径经过的顶点，9999 代表 ∞.

2　约束条件下的 F-D 算法

2.1　要求最短路径经过一个或几个顶点

假设所求的最短路径起点为 k，终点为 l，要求该路径必须经过顶点 i. 分别求出顶点 k 至顶点 i 以及顶点 i 至顶点 l 之间的最短路径，再将两段路径合并，即得顶点 k 至顶点 l 的最短路径。如果顶点 k 至顶点 l 之间的最短路径有 $n(n>1)$ 个控制点，分别求出起点 k 至控制点 1，

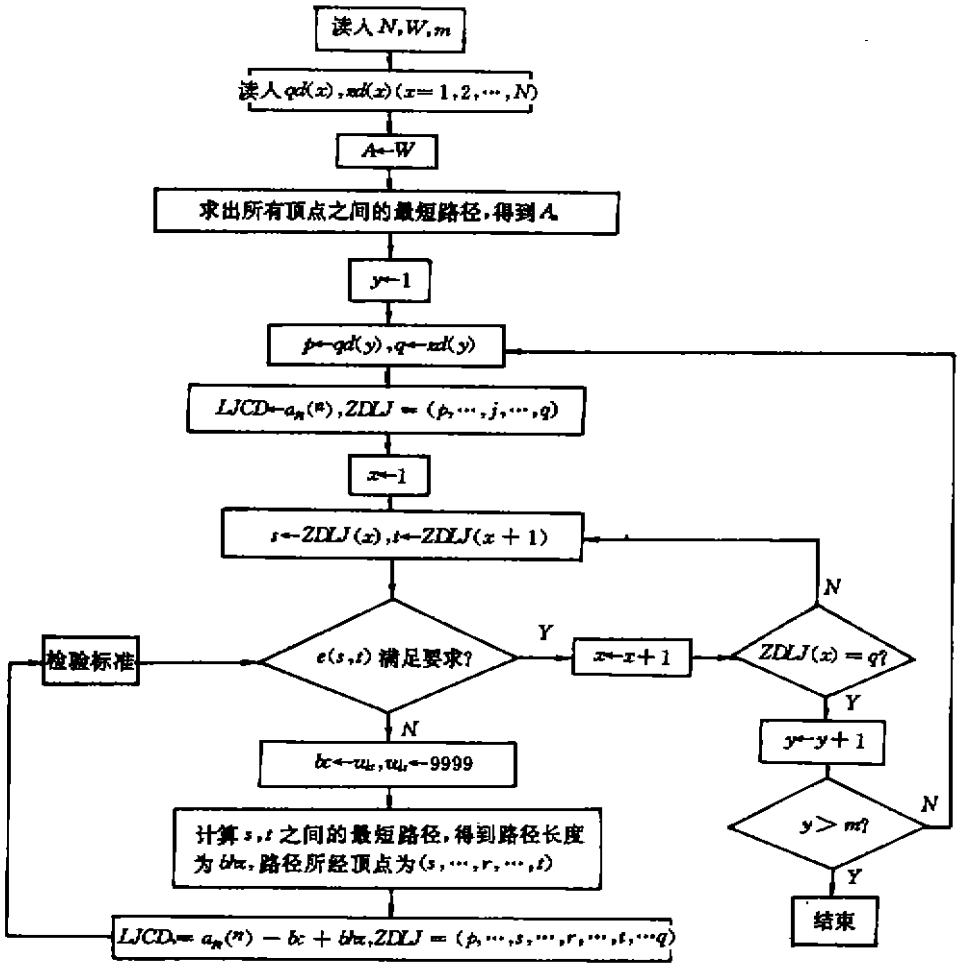

图 1 F-D 算法流程图

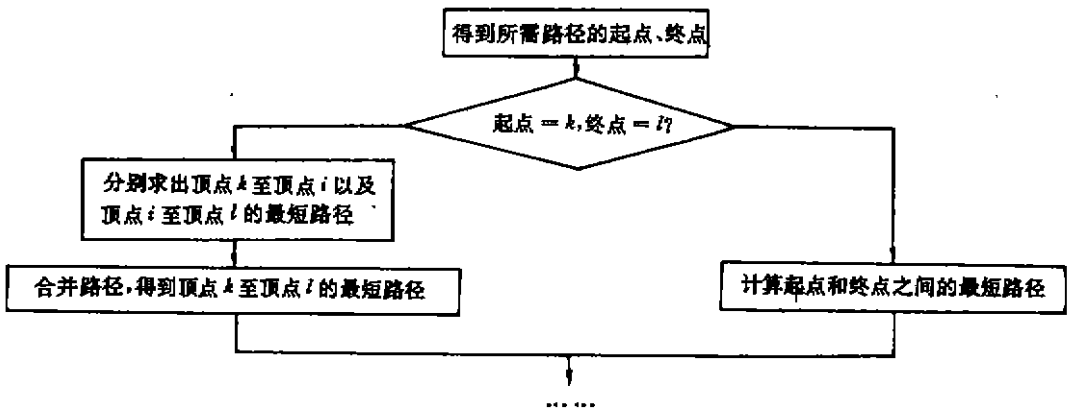

图 2 修改后的 F-D 算法计算流程图(部分)

控制点 1 至控制点 2，…，控制 n 至终点 l 的最短路径，然后将这些路径合并，即可得到顶点 k 至顶点 l 的最短路径。假设顶点 k 至顶点 l 的最短路径中有一个控制点 i，修改后的 F-D 算法计算流程如图 2 所示。

2.2　图中的一条或几条边不能用来构成路径

图中不能构成路径的边统称为“无效边”。从 F-D 算法的计算步骤来看，F-D 算法首先求得所需的最短路径，再对求得的最短路径的每条边进行检验，以确定所得路径是否正确。如果图中有“无效边”，就可能使得某些最短路径不正确。为了检验出不正确的路径，可将“无效边”作为 F-D 算法的检验标准，只要发现求得的路径中出现“无效边”，就按照 F-D 算法重新计算最短路径。

3　算例分析

下面用一算例验证 F-D 算法的正确性，并与 Floyd 算法和 Dijkstra 算法在计算效率上进行比较。

算例　图 3 所示的 Q 是一个无向图，顶点编号为 ①，②，…，⑳，每一条边都被赋予权值。图中共有 158 对不相邻接的顶点，可以形成 158 条路径。从中选取顶点 ①、②、③ 与其余不相邻接的顶点构成 47 个顶点对，分别用 F-D 算法算法、Floyd 算法、Dijkstra 算法计算它们之间的最短路径，并加上如下约束条件：(1) 某路径 $(p_1, p_2, \cdots, p_n)$ 权值为 11 的一条边 $e(p_m, p_k)$，它是不允许作为该路径的一条无效边；(2) 在这 47 对顶点中，有 3 对顶点需要加入控制点，如表 1 所示。

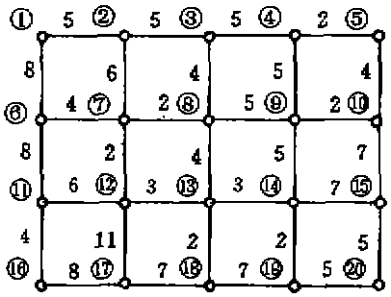

图 3　无向图 Q

表 1　未经检验的含有控制点的三条最短路径

起点	终点	控制点	路径长度	路径所经顶点
1	13	3	18	1,2,3,8,13
2	18	8,9	23	2,7,8,9,14,13,18
3	12	13	11	3,8,13,12

计算在 486/66 微机上进行，F-D 算法的计算步骤如下：1) 根据约束条件(2)，将需要计算的 47 对顶点分为两部分；一部分是没有控制点的控制对，共 44 对；另一部分是有控制点的顶点对，共 3 对。2) 按照图 1、图 2 的计算流程分别计算没有控制点的顶点对和有控制点的顶点对之间的最短路径，计算结果分别如表 1、表 2 所示。3) 根据约束条件(1)对求得的路径进行检验，并对不正确的路径重新计算。表 3 是经检验为不正确的路径，表 4 是对表 3 重新计算的结果。

表 2 未经检验的其它无控制点的最短路径

起点	终点	路径长度	路径所经顶点	起点	终点	路径长度	路径所经顶点
1	3	10	1,2,3	2	11	14	2,7,12,11
1	4	15	1,2,3,4	2	12	8	2,7,12
1	5	17	1,2,3,4,5	2	13	11	2,7,12,13
1	7	11	1,2,7	2	14	14	2,7,12,13,14
1	8	13	1,2,7,8	2	15	21	2,7,12,13,14,15
1	9	18	1,2,7,8,9	2	16	18	2,7,12,11,16
1	10	20	1,2,7,8,9,10	2	17	19	2,7,12,17
1	11	16	1,6,11	2	19	16	2,7,12,13,14,19
1	12	13	1,2,7,12	2	20	21	2,7,12,13,14,19,20
1	14	19	1,2,7,12,13,14	3	5	7	3,4,5
1	15	26	1,2,7,12,13,14,15	3	6	10	3,8,7,6
1	16	20	1,6,11,16	3	7	6	3,8,7
1	17	24	1,2,7,12,17	3	9	9	3,8,9
1	18	18	1,2,7,12,13,18	3	10	11	3,4,5,10
1	19	21	1,2,7,12,13,14,19	3	11	14	3,8,7,12,11
1	20	26	1,2,7,12,13,14,19,20	3	13	8	3,8,13
2	4	10	2,3,4	3	14	11	3,8,13,14
2	5	12	2,3,4,5	3	15	18	3,4,5,10,15
2	6	10	2,7,6	3	16	18	3,8,7,12,11,16
2	8	8	2,7,8	3	17	17	3,8,13,18,17
2	9	13	2,7,8,9	3	18	10	3,8,13,18
2	10	15	2,7,8,9,10	3	19	13	3,8,13,14,19

表 3 经检验不合格的最短路径

起点	终点	路径长度	路径所经顶点
1	17	24	1,2,7,12,17
2	17	19	2,7,12,17

表 4 对表 3 重新计算的结果

起点	终点	路径长度	路径所经顶点
1	17	25	1,2,7,12,13,18,17
2	17	20	2,7,12,13,18,17

用 Floyd 算法计算的步骤如下：1）算出所有顶点对之间的最短路径；2）选取所需的 47 条路径并逐一进行检验；3）发现不合格的路径后，修改权矩阵的相应元素，重新计算所有顶点对之间的最短路径，再从中选取所需路径。用 Dijkstra 算法计算时，通过多次执行 Dijkstra 算法得到所需的最短路径。在得到相同计算结果的前提下，三种算法时间如表 5 所示。

表 5　三种算法的计算时间　s

	计算时间
F-D 算法	20
Floyd 算法	40
Dijkstra 算法	48

上述算例计算结果表明：F-D 算法综合了 Floyd 算法和 Dijkstra 算法的特点，计算量小，有较高的计算效率，再加上约束条件的有效施加，形成了一种有良好应用前景的最短路径算法。

限于篇幅，关于 F-D 算法软件的实际应用，将在下一篇文章中具体介绍。

参 考 文 献

1　陈树柏．网络图论及其应用．北京：科学出版社，1982. 105～106
2　肖金声．关于最短路径算法．中山大学学报，1987，26(3)：42～47
3　米涅卡 E. 网络和图的最优化算法．北京：中国铁道出版社，1984. 43～46
4　吴文沧．图论基础及应用．北京：中国铁道出版社，1982. 97～98

A New Optimization Alogorithm——the F-D Algorithm

Nie Li　　*Yu Jihai*
(Department of Electrical Engineering, Chongqing University)

ABSTRACT　This paper proposes a new algorithm——the F-D algorithm by the combination of the Dijkstra algorithm with the Floyd algorithm to solve the shortest path problem between partial vertexes in undirected graph based on practical optimization problem and the computational efficiency of the F-D algorithm is verified by numerical examples.

KEYWORDS　graph; optimization; algorithm