具有多条最短路径的最短路问题

王志坚，韩伟一，李一军

（哈尔滨工业大学 管理学院，哈尔滨 150001，wyhan@mails.gucas.ac.cn）

摘　要：尽管 Dijkstra 算法是解决正权单源点最短路问题公认的最好算法，但它仅能求得从源点到指定点的一条最短路径，为了给出从源点到指定点的所有最短路径，通过改进临时标号过程，得到了修正的 Dijkstra 算法. 修正后的算法得到的不再是最短路径树，而是最短路径图. 相对于原算法，修正后的算法不仅更加简便，而且应用 Yen 算法能够按照边数由少到多的顺序罗列出所有的最短路径.

关键词：算法；最短路问题；Dijkstra 算法；Yen 算法

中图分类号：O221.4　　**文献标志码**：A　　**文章编号**：0367－6234(2010)09－1428－04

Shortest path problem with multiple shortest paths

WANG Zhi-jian, HAN Wei-yi, LI Yi-jun

(School of Management, Harbin Institute of Technology, 150001, China, wyhan@mails.gucas.ac.cn)

Abstract: Though Dijkstra algorithm is the best known algorithm to solve the shortest path problem with non-negative weight, it can only get one path from the source vertex to a designated vertex. In order to present all shortest paths from the source vertex to a designated vertex, the revised Dijkstra algorithm is obtained by improving the temporary label updating process. As a result, the revised algorithm gives the shortest path graph other than the shortest path tree. Compared with Dijkstra algorithm, the revised algorithm is simple, and all shortest paths can be given according to the number of edge by applying Yen algorithm.

Key words: algorithm; the shortest path problem; Dijkstra algorithm; Yen algorithm

单源点正权最短路问题是最短路问题中最简单的、应用最为广泛的一类问题，在计算机科学、交通科学等工程技术领域具有非常重要的应用. Shimbel[1] 提出了第一种算法——矩阵算法，但计算效率较低，计算复杂性为 $O(n^3\log n)$. 长期以来，Dijkstra 算法一直是解决正权单源点最短路问题的最好算法之一[2]，该算法既可用于有向问题，又可用于无向问题，应用 Fibonacci 堆计算复杂性为 $O(m + n\log n)$[3-7]. 然而，它只能求得从源点到其它点的一条最短路径，最终结果表现为一棵最短路径树. 事实上，从源点到指定点可能存在多条最短路径，Dijkstra 算法无法给出所有的最短路径. 为了方便，本文把这类从源点到指定点存在多条最短路径的问题特称为具有多条最短路径的最短路问题. 如果源点和指定点仅仅存在数量不多的最短路径，获得这些最短路径不存在实质性难度，可以通过枚举列出. 如果源点和指定点间的最短路径数量庞大，则无法一一列出，只能从中选出若干条较满意的最短路径，如所选择的最短路径具有最少的边数等[8-10]. 为此，本文将对 Dijkstra 算法进行修正，修正后的算法得到的不再是最短路径树，而是最短路径图. 在最短路径图中，从源点到指定点的任意两条路径都具有相同的距离. 文中还将基于最短路图，应用 Yen 算法罗列出从源点到指定点的所有最短路径.

收稿日期：2010－01－20.

基金项目：国家自然科学基金资助项目(90924015)；哈尔滨工业大学青年优秀基金资助项目(2009036).

作者简介：王志坚(1972—)，男，博士研究生；
李一军(1956—)，男，教授，博士生导师.

1　修正的 Dijkstra 算法

单源点正权最短路问题可描述为：给定一个具有 n 个点 $x_i(1 \leqslant i \leqslant n)$、$m$ 条边的有向图或无向图 $G(V,E)$，令源点为 v_1，边 (v_i,v_j) 的权 $w(v_i,v_j)$ 为正

实数. 若点 v 为图中任意一点，R 是 $G(V,E)$ 中从 v_1 到 v 的一条路径，定义路径的权是所有边的权之和，记为 $w(R)$. 单源点最短路问题就是在所有从 v_1 到 v 的路 R 中，求一条权最小的路 R_0 使得 $w(R_0) = \{w(R)\}$.

经典 Dijkstra 算法是一种标号算法，其思想为：

1）引入了两种标号，即 T 和 P. T 为从源点 v_1 到 v 的最短路权的上界，称为临时标号，记为 $T(v)$. P 为从源点 v_1 到这一点的最短路权，称为固定标号，记为 $P(v)$. 之所以称为固定标号，是因为当点 v 拥有 P 后，其 T 的值就不再改变. 同时，引入集合 S，用以表示所有具有 P 的点的集合.

2）引入函数 $\lambda(v)$，用以记录路径 R 中点 v 的上一个点.

3）首先令源点 v_1 的 $T(v_1) = 0$，其它点 v 的 $T(v) = +\infty$，并令源点 v_1 首先成为具有 P 的点，同时把点 v_1 加入到集合 S. 算法的每一步总能使一个没有 P 的点转变为新的具有 P 的点，这样一来，经 $n-1$ 步，所有点都将拥有 P，算法终止. 这时候，点 v 的 P 就是从源点 v_1 到 v 的最短路权.

4）根据函数 $\lambda(v)$，反溯得到从源点 v_1 到 v 的最短路径.

Dijkstra 算法的具体步骤为：

Step. 1　初始化.

$T(v_1) = 0, \lambda(v_1) = v_1$; $T(v) = +\infty$,

$\lambda(v) = M$; $S = \varphi$.

Step. 2　选取不拥有 P 但具有最小 T 的点 y，同时令 $P(y) = T(y)$，$S = S \cup \{y\}$.

Step. 3　对所有未进入集合 S 的、点 y 的邻点 x 进行 T 更新过程. 即：

如果 $T(y) + w(y,x) < T(x)$，那么 $T(x) = T(y) + w(y,x)$，同时令 $\lambda(x) = y$.

Step. 4　如果所有点都进入集合 S，那么算法停止，根据函数 $\lambda(v)$ 得到从源点 v_1 到 v 的最短路径，否则转入 Step. 2.

应用 Dijkstra 算法可以求得最短路径树. 在最短路径树中，从源点到任意点仅仅只有一条最短路径，这条路径在原图中则表现为相应两点之间的一条最短路径. 下面给出修正的 Dijkstra 算法为：

Step. 1　初始化.

$T(v_1) = 0$; $\mu(y,x) = 0$; $R = \{v_1\}$; $S = \phi$.

Step. 2　在集合 R 中选取具有最小 T 标号的点 y，则：

1）$S = S \cup \{y\}$，$R = R - \{y\}$；

2）若边 $(y,x) \in E$ 且 $x \notin S$，则 $R = R \cup \{x\}$.

Step. 3　若边 $(y,x) \in E$，则：

情形 1　若 $T(y) + w(y,x) < T(x)$，则 $T(x) = T(y) + w(y,x)$，$\mu(y,x) = 1$；

同时若 $\mu(z,x) = 1$ 且 $z \neq y$，则 $\mu(z,x) = 0$.

情形 2　若 $T(y) + w(y,x) = T(x)$，则 $\mu(y,x) = 1$.

Step. 4　如果所有点都进入集合 S，那么算法停止，把所有 $\mu(y,x) = 0$ 的边从图 $G(V,E)$ 中删去，这样得到的新图就是最短路径图，记为 $G^{\#}$，在 $G^{\#}$ 中从源点 v_1 到 v 的任意一条路径都是原图 $G(V,E)$ 中从源点 v_1 到 v 的最短路径，此时的 $T(v)$ 就是从源点 v_1 到 v 的最短路径的权；否则转入 Step. 2.

修正的 Dijkstra 算法相对于经典的 Dijkstra 算法得到的不再是最短路径树，而是最短路径图. 在最短路径图中，从源点到指定点的任意一条路径在原图中均是最短路径. 修正的 Dijkstra 算法具有：1）本改进算法仅保留了 Dijkstra 算法的 T 标号，简化了 Dijkstra 算法；2）在 Dijkstra 算法的初始化步骤中，令 $T(v) = +\infty$，这给计算机编程实现带来了困难，不得不选定一个足够大的数来代替 $+\infty$，本文的算法可避免这个麻烦.

下面给出算法正确性的证明：

引理 1　在修正的 Dijkstra 算法中，如果 $T(y) + w(y,x) = T(x)$，那么必有 $\mu(y,x) = 1$.

证明　根据算法的 Step. 3，当 $T(y) + w(y,x) < T(x)$ 或 $T(y) + w(y,x) = T(x)$ 时，都将有 $\mu(y,x) = 1$ 且 $T(y) + w(y,x) = T(x)$. 而当 $T(y) + w(y,x) > T(x)$ 时，由算法的 Step. 3，$\mu(y,x)$ 的值不发生变化，由于每个点仅进行一次 T 更新过程且 $\mu(y,x)$ 的初始值为 0，因而 $\mu(y,x)$ 的值仍然为 0. 另一方面，根据算法的 Step. 3，点 y 的 T 过程使得 $T(z) + w(z,x) = T(x)$ 不再成立，情形 1 将令 $\mu(z,x) = 0$，从而保证了只有当 $T(y) + w(y,x) = T(x)$ 时，$\mu(y,x)$ 的值才可能为 1.

综合上述几种情形，当 $\mu(y,x) = 1$ 时，$T(y) + w(y,x) = T(x)$ 时，总有 $\mu(y,x) = 1$.

定理 1　修正的 Dijkstra 算法可获得从源点 v_1 到点 v 的所有最短路径.

证明　假设 $P_0 = (v_1, U_1, \cdots, U_i, \cdots, U_h, v)$ 是得到的从点 v_1 到点 v 的任一条最短路径，而 $P_1 = (v_1, Q_1, \cdots, Q_i, \cdots, Q_r, v)$ 是从点 v_1 到点 v 的任意一条路径.

根据算法的步骤和引理 1，对于路 P_0 有

$T(v_1) = T(v_1) + w(v_1, U_1)$；

$T(U_i) = T(U_{i-1}) + w(U_{i-1}, U_i), (2 \leqslant i \leqslant h)$;

$T(v) = T(U_h) + w(U_h, v)$.

如果 P_1 也是从点 v_1 到点 v 的一条最短路径，那么必有

$$T(v) = T(U_h) + w(U_h, v) = T(Q_g) + w(Q_r, v).$$

根据引理 1，那么总有 $\mu(Q_r, v) = 1$. 这样就保证了边 (Q_r, v) 保留在最短路图中. 同时，也必然有

$T(v_1) = T(v_1) + w(v_1, Q_1)$;

$T(U_i) = T(Q_{i-1}) + w(U_{i-1}, U_i), (2 \leqslant i \leqslant r)$.

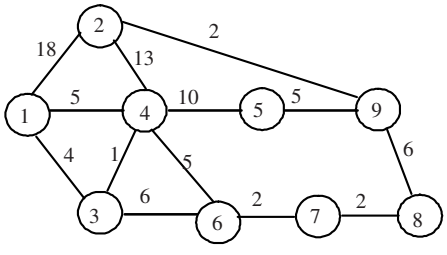
否则 P_1 就不是从点 v_1 到点 v 的最短路径，这样也有 $\mu(v_1, Q_1) = 1$ 且 $\mu(Q_{i-1}, Q_i) = 1 (2 \leqslant i \leqslant r)$，说明 P_1 中的所有边都在最短路图中.

鉴于 P_1 的任意性，因而修正的 Dijkstra 算法总可获得从源点 v_1 到点 v 的所有最短路径，这些最短路径中的任意一边都保留在最短路图中.

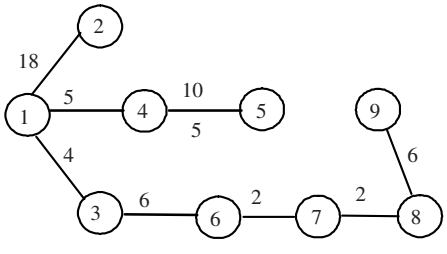
定理 1 保证了算法的正确性. 由引理 1 和定理 1 可直接得到：

推论 1 在最短路图中，从点 v_1 到点 v 的任一条路径都是原图中从点 v_1 到点 v 的最短路径.

例 1 求解下列最短路问题.



在例 1 中，显然从点 1 到点 2 ~ 点 8 和点 9 都存在多条路径，应用经典的 Dijkstra 算法得到的最短路径树为：



而应用修正的 Dijkstra 算法，得到的最短路径图与原图相同. 由最短路径图，从点 1 到点 9 将具有 8 条最短路径. 例 1 的图规模较小，列出两点间的所有最短路径是容易的，而当图的规模庞大且两点间的最短路径的数目可观时，将不得不寻求专门的算法.

2 获得两点间的所有最短路径

应用修正的 Dijkstra 算法得到一个最短路径图. 该算法基于最短路径图可以获得从源点到指定点的所有最短路径. 该算法的思想是，根据各最短路径中边的数目对各最短路径进行排序，边数越少的最短路径对应的级别就越高，算法将按照级别由高到低的顺序依次列出所有的最短路径.

如果令最短路径图中的各边的权均为 1，那么从源点到指定点的路径长度和路径中的边数相等. 这样一来，从源点到指定点的路径长度越短，该路径的级别就越高. 按照级别由高到低依次列出所有的最短路径，可以借助于求解第 k 条最短路径问题的 Yen 算法来解决[11].

设从源点到指定点有多条路径，这些路径具有不同的长度，如果按照长度由小到大进行排序，那么第 k 条最短路径是指排在第 k 个位置的路径. Yen 算法是求解无圈第 k 条最短路径问题当前公认的最好算法. Yen 算法的思想是，首先求得从源点到指定点的第 $1, 2, \cdots, k-1$ 条最短路径，然后在这 $k-1$ 条最短路径的基础上求得第 k 条最短路径. 由于在最短路径图中，从源点到指定点的最短路径是有限的，因而当 k 足够大时，按照 Yen 算法总可以求得从源点到指定点的所有最短路径.

下面给出获得所有最短路径的算法为：

Step. 1 应用修正的 Dijkstra 算法得到最短路径图 $G^{\#}$.

Step. 2 令最短路图 $G^{\#}$ 中所有边的权均为 1，得到新的图 G^*.

Step. 3 应用经典的 Dijkstra 算法得到图 G^* 的从源点到指定点的最短路径，也就是第 1 条最短路径.

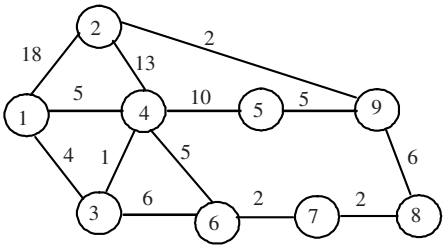
Step. 4 令 $k = 2$.

Step. 5 应用 Yen 算法求得图 G^* 的从源点到指定点的第 k 条最短路径. 如果 Yen 算法无法得到第 k 条最短路径，则算法结束，否则转入 Step. 6.

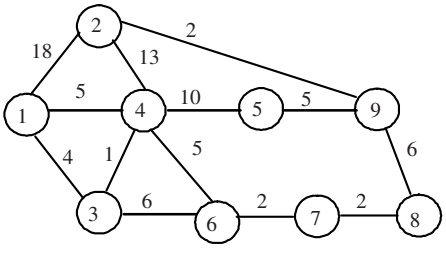
Step. 6 $k = k + 1$，转入 Step. 5.

由于从源点到指定点的最短路径的数目是有限的，因而算法是收敛的. 事实上，在算法的 Step. 3就得到了从源点到指定点的、边数最少的一条最短路径.

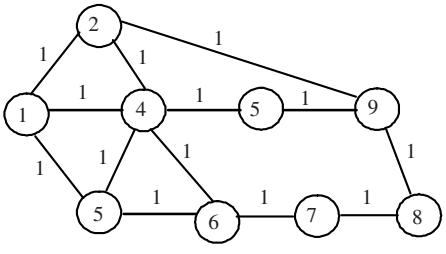
例 2 求解下列最短路问题，求得从点 1 到点 9 的所有最短路径.



解：应用修正的 Dijkstra 算法得到最短路径图.



由最短路径图，可知从点 1 到点 9 具有多条最短路径，最短路径的长度为 20. 接下来，令最短路径图的各边的权均为 1，得到：



在最短路径图上，应用 Yen 算法得到从点 1 到点 9 的所有最短路径为：

第 1 条最短路径：1→2→9；

第 2 条最短路径：1→4→2→9；

第 3 条最短路径：1→4→5→9；

第 4 条最短路径：1→3→4→5→9；

第 5 条最短路径：1→3→4→2→9；

第 6 条最短路径：1→4→6→7→8→9；

第 7 条最短路径：1→3→6→7→8→9；

第 8 条最短路径：1→3→4→6→7→8→9.

3　结　论

1）对 Dijkstra 算法进行了修正，修正后的算法得到的不再是最短路径树，而是最短路径图.

2）对 Dijkstra 算法进行了简化，不再沿用 P 标号仅仅保留了 T 标号，在初始化步骤中也不再令各点的 T 标号的值为无穷大，客观上为算法的计算机实现提供了方便.

3）基于最短路径图，应用求解第 k 条最短路径的 Yen 算法，根据边数由少到多可依次给出从源点到指定点的所有最短路径.

参考文献：

[1] SHIMBEL A. Structure in communication nets [C]// Proceedings of the Symposium on Information Networks. New York: Polytechnic Press of the Polytechnic Institute of Brooklyn, 1955: 199 – 203.

[2] DIJKSTRA E W. A note on two problems in connexion with graphs [J]. Numerische Mathematik, 1959, 1(1): 269 – 271.

[3] FREDMAN M L, TARJAN R E. Fibonacci heaps and their uses in improved network optimization problems [J]. Journal of the ACM, 1987, 34(3): 596 – 615.

[4] NOSHITA K, MASUDA E, MACHIDA H. On the expected behaviors of the Dijkstra's shortest paths algorithm for complete graph [J]. Information Processing Letters, 1978, 7(5): 237 – 243.

[5] NOSHITA K. A theorem on the expected complexity of Dijkstra's shortest paths algorithm[J]. Journal of Algorithms, 1985, 6(3): 400 – 408.

[6] PETTIE S, RAMACHANDRAN V. Computing shortest paths with comparisons and additions [C]//Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms. Philadelphia: Society for Industrial and Applied Mathematics, 2002: 267 – 276.

[7] PETTIE S, RAMACHANDRAN V. A shortest path algorithm for real-weighted undirected graphs [J]. SIAM Journal on Computing, 2005, 34(6): 1398 – 1431.

[8] 孙强，杨宗源. 求受顶点数限制的最短路径问题的一个算法[J]. 计算机工程，2002，28(9)：73 – 74.

[9] 周经纶，吴焕群. 受顶点数限制的最短路径问题及其算法[J]. 系统工程，l996，l4(5)：37 – 44.

[10] MARFINS V E Q. On a multicriteria shortest path problem[J]. European Journal of Operational Research, 1984, l6(2): 236 – 245.

[11] YEN J Y. Finding the K shortest loopless paths in a network[J]. Management Science, 1971, 17(11): 712 – 716.

（编辑　张　红）