一种改进的多播路由算法

刘　姝　　李伟生　　王　涛
（北京交通大学计算机与信息技术学院　北京 100044）

摘　要：低代价最短路径树是一种广泛使用的多播树，它能够在保证传送时延最小的同时尽量降低带宽消耗。DDSP(Destination-Driven Shortest Path) 算法是一个性能较好，计算效率较高的低代价最短路径树算法，在该算法基础上，通过改进结点的搜索过程，提出一种改进的快速低代价最短路径树算法。由算法分析和实验比较得出，改进算法的计算效率高于 DDSP 算法，且算法构造的最短路径树的性能也优于 DDSP 算法构造的树。

关键词：多播，低代价，最短路径树
中图分类号：TP393　　**文献标识码**：A　　**文章编号**：1009-5896(2005)04-0638-04

An Advanced Algorithm for Fast Lower-Cost Shortest Path Tree

Liu Shu　　Li Wei-sheng　　Wang Tao
(*College of Computer and Information Technology, Beijing Jiaotong University, Beijing* 100044, *China*)

Abstract　Lower-cost shortest path tree is a commonly-used multicast tree type, which can minimize end-to-end delay and at the same time reduce bandwidth as possible. This article presents an algorithm for lower cost shortest path tree. The algorithm adjusts the nodes' minimum cost to the current shortest path tree dynamically, and gradually gets shortest path tree with low total cost by selecting the node with minimum cost to current shortest path tree in turn. The algorithm has better performance and lower complexity than Destination-Driven Shortest Path tree (DDSP) algorithm so that is a very fine shortest path tree algorithm by algorithm analysis and simulation.

Key words　Multicast routing, Lower cost, Shortest path tree

1　引言

多播是一种允许向一组目的主机同时传送信息的通信方式。它在电视会议、视频点播、网络电话和基于 web 的应用等领域有着广泛的应用。

当前网络支持多播的有效方法是构建连接源结点与所有目的结点的树型路由，一般有最短路径树(SPT)和共享树两种类型[1]。建立一棵好的 SPT 就是要在保证通信时延最小的情况下尽可能降低传输代价。长期以来，人们提出了许多解决方案。文献[2]提出最优最短路径树(SBPT)算法，在构造 SPT 的过程中采用贪心策略选择最短路径，可以降低所构造的 SPT 总消耗。文献[3]提出目的驱动最短路径树(Destination-Driven Shortest Path tree, DDSP)算法，算法利用 Dijkstra 算法路径递增的基本思想和 DDMC 算法[4]目的结点共享路径的方法，在保证与源结点路径最短的前提下，每次都选择与已计算的目的结点最近的路径，通过目的结点之间共享尽可能长的路径来降低生成树的总消耗。

DDSP 算法在将一个目的结点添加到树上时，只考虑与已计算的目的结点共享路径，但可能存在某非目的结点能与新添目的结点共享更长的路径。为了进一步降低多播路由树的代价，本文在 DDSP 基础上提出快速的低代价最短路径树（Fast Lower-Cost Shortest Path tree，FLCSP）。在构建最短路径树的过程中，确定某结点的父结点时不仅搜索目的结点集，而且考虑非目的结点，从而能够保证新增的路径是所有可能的路径中最短的；同时，在计算结点的可达最短路径长度时保存其最优的父结点，不需重新搜索，从而减少计算量。由算法分析和实验比较得出，本文算法的计算效率高于 DDSP 算法，且算法构造的最短路径树的性能也优于 DDSP 算法构造的树。

2　DDSP 算法构造最短路径树

DDSP 算法在保证与源结点路径最短的前提下，选择与已计算的目的结点最近的路径，通过目的结点之间共享尽可能长的路径来降低生成树的总消耗。DDSP 算法基本思路如下：

（1）引入向量 **CST**，**rCST** 和 ***S***。CST(u)表示结点 u 到源结点 s 的最短路径长度，rCST(u)表示结点 u 到生成树 T 上最近的目的结点的距离。$S(u)$为 1 则表示结点 u 已经计算过，

2004-03-25 收到，2004-08-05 改回

否则表示 u 尚未计算。

（2）初始化向量 **CST**，**rCST** 和 ***S***。对于图 G 中任意结点 u，$S(u)$赋值为 0，如果存在源结点 s 到结点 u 的边，则将 CST(u)和 rCST(u)赋值为边(s, u)的代价 cost[s, u]，否则赋值为∞。初始化生成树 T 以源结点 s 为其根结点，CST(s)和 rCST(s)赋值为 0, $S(s)$赋值为 1。

（3）选择结点 u，使得

$$\mathrm{CST}(u)=\min\{\mathrm{CST}(w)|1\le w\le n, S(w)=0\} \tag{1}$$

令 $S(u)$=1。选择结点 p 作为结点 u 的父结点，使得结点 p 满足

$$\mathrm{CST}(u)=\mathrm{CST}(p)+\mathrm{cost}[p, u]$$

$$\text{且 } \mathrm{rCST}(u)=r\mathrm{CST}(p)+\mathrm{cost}[p, u] \tag{2}$$

如果结点 u 为一个目的结点，则修改 rCST(u)=0，并将结点 u 添加到生成树 T。

（4）对于任意结点 v，如果 $S(v)$=0 且 cost[u,v]<∞，修改此结点的 CST(v)和 rCST(v)值。如果 CST(u)+ cost[u, v]<CST(v)，则令

$$\left.\begin{aligned}\mathrm{CST}(v)&=\mathrm{CST}(u)+\mathrm{cost}[u,v]\\ \mathrm{rCST}(v)&=\mathrm{rCST}(u)+\mathrm{cost}[u,v]\end{aligned}\right\} \tag{3}$$

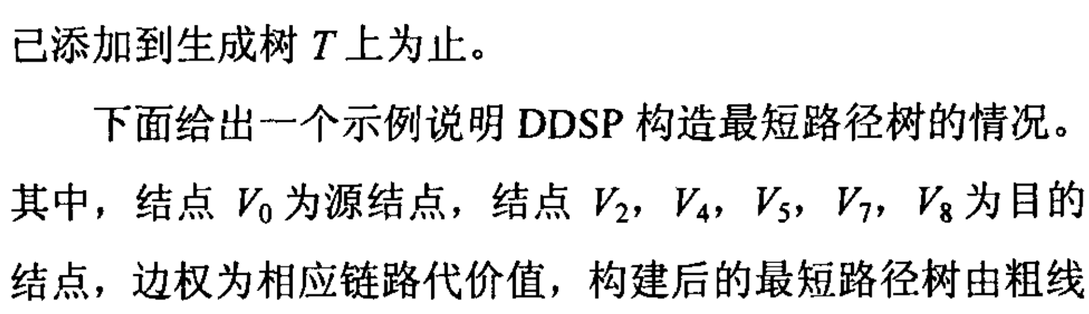
否则，如果 CST(u)+cost[u,v]=CST(v) 且 rCST(v) >rCST(u)+cost[u,v]，则令

$$\mathrm{rCST}(v)=\mathrm{rCST}(u)+\mathrm{cost}[u, v] \tag{4}$$

（5）重复步骤（3）和步骤（4）,直到所有目的结点都已添加到生成树 T 上为止。

下面给出一个示例说明 DDSP 构造最短路径树的情况。其中，结点 V_0 为源结点，结点 V_2，V_4，V_5，V_7，V_8 为目的结点，边权为相应链路代价值，构建后的最短路径树由粗线示出。

图 1（a）是一棵 DDSP 算法构造的 SPT，算法在执行过程中，当目的结点 V_2 加入树 T 后，结点 V_9 与目的结点 V_2 之间最近的路径为 4，共享的路径为 V_0—V_2 段。仔细研究可以发现，欲加入结点 V_9，如果利用已加入树中的非目的结点 V_6，从结点 V_6 到结点 V_9 可以直接构造一条路径，其距离仅为 2，这样利用非目的结点 V_6，可以与新添目的结点 V_9 共享更长的路径（V_0—V_2—V_6 段），所以原图可以构造出另一棵



(a) DDSP 算法（总代价为 14)　　(b) FLCSP 算法（总代价为 12)

图 1　构造最短路径树的示例

SPT(图 1(b))，使得 SPT 总代价比 DDSP 的更小。而且，DDSP 算法在添加结点到生成树上时都需要重新从已计算的结点中搜索其父结点，而本文算法在计算结点的可达最短路径长度时将保存其最优的父结点，不需重新搜索，从而可以减少计算量。这些正是本文提出的算法要达到的目标。

3　快速低代价最短路径树(FLCSP)算法

3.1　FLCSP 算法

FLCSP 算法通过改进结点的搜索过程，依次选择和当前最短路径树有最近路径的结点来逐步生成总体代价小的最短路径树。

算法将通信网络抽象为无向图 G=(V, E), V 为主机或路由器结点集，E 为通信链路集，cost(u, v)为链路(u, v)代价值，假定所有链路的代价值非负，若(u, v)不存在，则置 cost(u, v)为∞（在计算机上可用允许的最大值代替）。T 为从 s 出发的最短路径树结点集合，它的初始状态为空集。向量 CST（u）表示结点 u 到源结点 s 的最短路径长度，RCST (u)表示结点 u 到生成树 T 上最近的目的结点的距离。nState(u)标记结点是否访问过，nState(u) = 1 表示结点 u 已访问过，nState(u) = 0 表示 u 尚未访问。Recal 表示需重新计算相邻结点代价值的结点集合。Pri 用于指向结点在树中的父结点。给定一源结点 $s\in V$，及一组目的结点集 $D\subset V$。算法过程描述如下：

（1）初始化向量 **CST**，**RCST**，**nState**。对于图 G 中任意结点 u，nState(u)赋值为 0；如果存在源结点 s 到结点 u 的边，则将 CST(u)和 RCST (u)赋值为边(s,u)的代价值 cost[s,u]，否则赋值为∞。初始化生成树 T={s}，Recal=Φ，CST(s)= 0，RCST (s)= 0，nState(s)= 1。

（2）选择某一结点 u，使得 CST(u)= min{CST(w)|1≤ w ≤n，nstate(w)=0}。

（3）令 nState(u)= 1，并将 u 加入 Recal。如果属于目的结点集，则将其加入 T，并修改其 RCST 值为 0，记录其父结点 w；否则跳转到第（5）步。

（4）考察 w 的 RCST 值，若不为 0（即不属于目的结点集），则将 w 加入 T，修改其 RCST 为 0，并将 w 加入 Recal，记录父结点编号 Pri[w]，仍记录在 w 中。此步重复执行直至 RCST(w)=0。（此步完成时建立起结点 u 到 T 的路径，且 T 中新添结点的 RCST 均为 0。）

（5）若 Recal≠ Φ，取出首结点 u。对与 u 相邻的每个结点 w，如果 nState(w)=0 且 cost[u, w]<∞，修改此结点的 CST (w)和 RCST (w)值，即

如果 CST (u) + cost[u, w] < CST (w), 则令

CST (w) = CST (u) + cost[u, w],

RCST (w) = RCST (u) + cost[u, w],

Pri[w] = u;

否则如果 CST (u) + cost[u, w] = CST(w)且 RCST (w) > RCST (u) + cost[u, w], 则令

RCST (w) = RCST (u) + cost[u,w] ,

Pri[w] = u。

（6）重复执行算法第(3)~(5)步，直至所有的目的结点都已加入 T 中。

该算法用向量**RCST**保存结点到已计算目的结点的最近距离，在有多条最短路径的情况下总是选择能够使 RCST 分量值最小的路径，最大限度地与其它目的结点共享路径，因而能够降低最短路径树的总消耗。同时，该算法在计算结点的可达最短路径长度时保存其最优的父结点，不需重新搜索，从而可以减少计算量。

3.2 算法时间复杂度分析

用 n 表示图 G 中结点数目，e 表示所有边的数目，$d(u)$ 表示结点 u 的邻接结点数目。

算法的初始化时间复杂度是 $T_1=O(n)$。

算法需要从未访问的结点集中选择最合适的结点进行下一步扩展，所以必须考虑使用何种数据结构。较常用的数据结构有数组、堆栈、队列、单双链表和哈希散列等。将未访问的结点集构造成 Fibonacci 堆，则从该集合中输出最小结点等操作可以在 $O(\log n)$时间内完成。由算法第(3)，(4)步可看出，每一个结点最多往 Recal 中添加两次，且每添加一个新结点 u 到生成树上后需要访问 $d(u)$个 u 的邻接结点。在所构造的 SPT 中最多包含 n 个结点，所以算法构造 SPT 时间复杂度为

$$T_2 = n\log n + 2\sum_{i=1}^{n} d(V_i) = O(n\log n + 2e) \tag{3}$$

算法的总时间复杂度为

$$\begin{aligned} T &= T_1 + T_2 = O(n) + O(n\log n + 2e) \\ &= O(n + n\log n + 2e) = O(n\log n + e) \end{aligned} \tag{4}$$

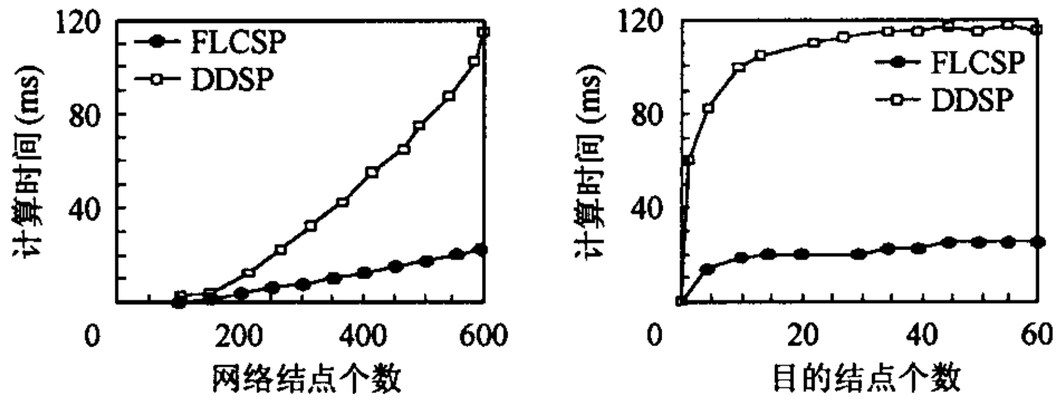
本文算法的时间复杂度的估计比 DDSP 算法所估计的复杂度要低。

4 模拟试验及其分析

为了验证本文提出方法的正确性和有效性，采用了多组随机网络模型进行试验，比较 FLCSP 算法和 DDSP 算法产生最短路径树的总代价，发现在所有的随机网络模型中，FLCSP 算法的计算时间小于 DDSP 算法，且 FLCSP 算法产生的最短路径树的总体代价小于 DDSP 算法。

为了产生具体实际网络特征的随机网络图，本文采用文献[5]中的方法生成随机网络模型。在该模型中，n 个结点随机地分布在直角坐标系内（结点的坐标均为整数），结点之间的直线距离作为其费用。结点之间的连通性由连通概率 $P(u, v)$决定。$P(u, v)$的值由 u, v 之间的距离决定。$P(u,v) = \beta e^{-d(u,v)/L^{\alpha}}$，其中 $d(u, v)$是 u, v 之间的距离，L 是任意两结点间的最大距离，α 和 β 是调节网络图特征的参数。当 α 增加时，长边相对短边的比增加；β 增加，结点的度也随着增加。调整 α 和 β 可以产生不同类型的随机网络图，使之更接近实际网络。根据试验采用 α =0.5, β =0.3。

图 2 (a)是目的结点数固定为 100 时，DDSP 算法和 FLCSP 算法的计算时间随网络结点数目变化的曲线。横坐标为网络结点数目，纵坐标为计算时间(单位：ms)。从图中可见，当目的结点数目固定时，DDSP 算法和 FLCSP 算法的计算时间随着网络结点数目的增加而增加，但 FLCSP 算法的增长幅度较小。图 2(b)是网络结点数固定为 600 时，DDSP 算法和 FLCSP 算法的计算时间随目的结点数目变化的曲线。两种算法的计算时间都随目的结点数目的增加而增加，但 FLCSP 算法的增长幅度较小。



(a) 计算时间随网络结点数目的变化 (b) 计算时间随目的结点数目的变化

图 2 算法计算时间变化曲线

表 1 是目的结点数固定为 20 时，FLCSP 和 DDSP 算法随网络结点数变化的平均性能比较。如表所示，随着结点数的增加，FLCSP 和 DDSP 算法的平均总体代价都是增加的，但是 DDSP 的平均总体代价高出 FLCSP 约 0.9 – 1.2%。

表 2 是网络结点数固定为 200 时，FLCSP 和 DDSP 算法随目的结点数变化的平均性能比较。如表所示，随着目的结点数的增加，FLCSP 和 DDSP 算法的平均总体代价也是增加的，且目的结点数小于 60 时总体代价增长迅速，超过 60 后总体代价增长较缓慢，但是 DDSP 的平均总体代价仍然高出 FLCSP 约 1.5 – 3.5%。

5 结束语

本文提出了一种改进的快速低代价最短路径树算法，该算法通过改进最短路径结点的搜索过程，以较小的存储空间为代价，获得了性能较优的低代价最短路径树。随机网络模型的仿真结果表明，该算法生成最短路径树较 DDSP 算法计

算速度更快，且总体代价相对较小，是一种很有实际应用价值的算法。

表 1　平均总体代价随结点数变化表

网络结点数	50	100	150	200	250
DDSP	99.52	108.81	111.67	121.13	127.48
FLCSP	98.4	106.89	110.6	118.96	126.51
网络结点数	300	350	400	450	500
DDSP	125.67	148.07	169.14	185.23	198.36
FLCSP	125.12	147.98	149.36	181.78	197.01

表 2　平均总体代价随目的结点数变化表

目的结点数	10	20	30	40	50
DDSP	84.92	149.47	229.38	253.08	283.76
FLCSP	83.04	146.95	225.52	248.11	279.02
目的结点数	60	70	80	90	100
DDSP	298.93	336.63	385.76	395.63	398.87
FLCSP	295.12	332.26	379.86	389.19	396.08

参 考 文 献

[1] 张宝贤，等. 多媒体通信中的多点路由问题. 通信学报，1999, 20(5): 63 – 70.

[2] Fujinoki H, Christensen K. The new shortest best path tree (SBPT) algorithm for dynamic multicast tree. Proceedings of the IEEE 24th Conference on Local Computer Networks, Lowell, MA, USA, 1999: 204 – 211.

[3] Zhang B X, Mouftah H T. A destination-driven shortest path tree algorithm. IEEE International Conference on Communications, Kingston, Canada, 2002, 4: 2258 – 2262.

[4] Shaikh A, Shin K G. Destination-driven routing for low-cost multicast. *IEEE J. on Selected Areas in Communications*, 1997, 15(3): 373 – 381.

[5] Waxman B M. Routing of multipoint connections. *IEEE J. on Selected Areas in Communications*, 1988, 6(9): 1617 – 1622.

[6] Maxemchuk N F. Video distribution on multicast networks[J]. *IEEE J. on Selected Areas in Communications*, 1997, 15(2): 357 – 372.

刘　姝：　女，1979年生，硕士生，主要研究方向为计算机网络技术、图论算法的设计与分析.

李伟生：　男，1945年生，教授，主要研究方向为计算机网络技术、图论算法的设计与分析.

王　涛：　男，1980年生，助教，主要研究方向为计算机网络技术、图论算法的设计与分析.