局部搜索最小路径费用算法

李汉兵[1], 喻建平[2], 谢维信[2]
(1. 西安电子科技大学电子工程学院, 西安 710071; 2. 深圳大学信息工程学院, 深圳 518060)

摘 要: 本文在 MPH(Minimum Path Cost Heuristic)的基础上, 改进了端节点的加入过程, 得到了两种改进的 MPH 算法: 局部搜索最小路径费用算法 LSMPH(Locally Searching MPH)和简化的 LSMPH: 最短端节点最小路径费用算法 STMPH(Shortest Terminal MPH). 在随机网络模型的基础上, 我们进一步进行了仿真. 仿真结果表明, LSMPH 以相对较小的费用增加换取更快的计算速度. 如果要求更快的速度, 可以采用 STMPH.

关键词: NP 完全问题; 路由算法; 组播树; 网络
中图分类号: TN393 文献标识码: A 文章编号: 0372-2112 (2000) 05-092-04

Locally Searching Minimum Path Cost Heuristic

LI Harr-bing[1], YU Jiarr-ping[2], XIE Weir-xin[2]
(1. *School of Electronic Engineering, Xidian University, Xi' an 710071, China;*
2. *School of Information Engineering, Shenzhen University, Shenzhen 518060, China*)

Abstract: On the basis of MPH (Minimum Path Cost Heuristic), The method of subpath adding to the partially constructed multicast tree is modified. Thus two algorithms are adopted, one is locally searching minimum path cost heuristic (LSMPH) and the other is shortest terminal minimum path cost heuristic (STMPH), the simplified version of LSMPH. With the random graph model, the simulation shows that LSMPH can compute faster with relatively smaller cost increasing. If much shorter computing time is needed, STMPH is more suitable than LSMPH.

Key words: NP-complete; routing algorithm; multicast tree; networks

1 引言

网络通信一般分为 5 类, 即单播(unicast), 组播(multicast), 汇播(concast), 群播(multipoint to multipoint)和广播(broadcast). 其中, 组播是应用很广的一种功能. 组播是一种从一个源节点(source)向多个端节点(terminal)同时发送相同信息的通信方式. 这里的节点在实际中是指路由器, 源节点是信息的发送者, 端节点是信息的接收者. 随着网络技术的发展, 会议电视等应用已经开始深入我们的生活. 因而, 网络通信中组播功能也越来越重要. 组播路由是指从源节点到各端节点传送信息的路径(path). 组播路由的确定是一个 NP 完全问题[1]. 这类问题是不可能用最优方案来解决的. 我们所寻求的只是一种次优的, 在实际中可以接受的方案. 到目前为止, 人们提出的方案已经有许多种, 其中比较典型的有 ADH, DNH, MPH[2], SCTF[3] 等. MPH 是大家讨论得较多的一种方法[2~5]. 在文[3]中, S. Ramanathan 把 MPH 作为比较组播树算法的一种标准算法. 这些方法都是基于图论的, 在已知网络拓扑结构前提下, 进行路由计算. 一种好的路由算法是在尽可能少的时间内计算出的组播树的总费用尽可能的少. 基于这样一种考虑, 我们提出了 LSMPH 和 STMPH 算法.

本文以下的内容是这样安排的. 首先, 将本文使用的术语和 MPH 算法作了简要介绍. 接着, 详细描述了改进算法 LSMPH 和 STMPH 的基本思想和步骤. 然后, 对改进的算法进行了仿真. 最后, 对本文的工作进行了总结.

2 相关的工作及术语

2.1 相关术语

为了描述的方便, 我们将一个通信网络表示为一个带权有向图 $G=(V, A, C)$. 这里 V 是网络中的节点, 即路由器; A 是一组有向边(directed edge), 用某条边对应的两个节点 u, v 来表示这条边: (u, v); C 是各边对应的费用(cost)的集合. 图 G 中节点 v 的度(degree)是指与 v 关联的边的个数. 在有向图中, 节点 v 的度还有入度(in degree)与出度(out degree)之分. 节点 v 的入度是指与 v 关联的指向 v 的边的个数; 节点 v 的出度是指与 v 关联的背离 v 的边的个数. 本文中, 为了简化对问题的讨论, 我们将网络表示为一个无向图, 一对节点对应的两条边 (u, v) 和 (v, u) 的费用也就相等, 即这两条边对称.

本文要考虑的问题是以源节点(source)为根的组播树的生成, 其定义如下:

定义 1 给定一个有向图 $G=(V, A, C)$, 一组端节点 M,

收稿日期: 1999-01-04; 修订日期: 1999-10-01
基金项目: 广东省自然科学基金(No. 984114)资助课题

一个源节点 s, 从 G 中生成一个有向树 $T=(V_t, A_t, G)$, 其中 $T\subseteq G$, $V_t\subseteq V$, $M\subseteq V_t$, $A_t\subseteq A$, T 中的源节点 s 到每一个端节点都有通路, 并且 s 的入度为 0, 树中其余节点的入度为 1. 端节点的出度大于或等于 0, 其余节点的出度均大于或等于 1. 组播树 T 的总费用是从源节点 s 到所有端节点的各边费用之和. 组播树中源节点 s 和端节点 M 以外的节点称为 Steiner 节点.

定义 **2**　最短路径: 节点 $u, v\in V$ 之间的最短路径是指从 u 到 v 的费用最小的路径, 用 $P(u, v)$ 表示. 树到节点的最短路径是指树的各节点到该节点的最短路径中费用最小的那条.

定义 **3**　卫星节点: 在生成树中, 一个端节点与邻近端节点之间的 Steiner 点称为该端节点的卫星节点.

2.2　MPH 的基本思想

本文的算法涉及到 MPH 算法, 因此有必要简要介绍 MPH 算法, 作为进一步讨论的基础.

(1) 从源点 $s\in M$ 开始, 将单节点 s 作为 T_1. 此时 $T_k=T_1$.

(2) 从余下的端节点 $M-\{s\}$ 中搜索出距离 T_k 最近(即费用最小)的端节点 i, 将从 T_k 到 i 的最短路径加入到 T_k 中, 得到 T_k+1; 令 $k=k+1$.

(3) 重复(2), 直到 M 中所有节点都包含在树 T_k 中. 此时得到的即为组播树.

3　局部搜索 MPH 算法

在 MPH 算法中, 将一个端节点加入到 T_k 中, i 必须是余下的端节点中距离 T_k 最近的, 其搜索范围很大, 搜索时间较长. 假设 T_k 中的节点为 V_k, 余下的端节点为 M_k, T_k 中的 Steiner 节点为 V_s. Steiner 节点是生成树 T_k 中除去端节点以外的节点. 则 MPH 的搜索次数为 $|V_k|\ |M_k|$. 本文提出的搜索方法是: 先将余下的所有端节点与树 T_k 中的端节点的距离一一比较, 搜索出距离最短的一对 (i, j), $i\in V_k$, $j\in M_k$, $i, j\in M$. 进一步, 比较 M_k 与 i 周围的卫星节点 W_k 中距离最短的一对 (i', j'), $i'\in W_k$, $j'\in M_k$. 将从 i' 到 j' 的最短路径加入到树 T_k 中. 如此重复下去, 直到 M_k 变空. 这样每一步的搜索次数为 $(|M|-|M_k|+|W_k|)|M_k|$. 因为 $|M|-|M_k|+|W_k|<|V_k|$, 搜索次数大为减少.

局部搜索法的基本思想是: 先估算全局搜索中最短节点的大概位置, 再在该局部区域逐个搜索.

局部搜索算法—LSMPH (Locally Searching Minimum Path Cost Heuristic)

已知: 有向图 $G=(V, A, C)$, 源节点 $s\in M$, 端节点集合 $M\subset V$. 令 $c(i, j)$ 为从 i 到 j 的最短路径的总费用, $P(i\to j)$ 为从 i 到 j 的最短路径. 待求: 以 s 为根的有向 Steiner 树 $T=(V_t, A_t)$.

算法开始

(1) 初始化

$\{s\}\to Q_!$ (搜索过的端节点),

$M\setminus\{s\}\to Q_?$ (未搜索的端节点),

$\{\}\to Q_s$(端节点的卫星节点),

$\{s\}\to V_t$(生成树中的节点),

$\{\}\to A_t$(生成树中的边);

(2) *WHILE* $Q_?\neq\{\}$ *DO*

(3)　　$\infty\to MIN$;

(4)　　FOR each m in $Q_?$ DO

(5)　　FOR each n in $Q_!$ DO

(6)　　$MIN=\min\{MIN, c(n, m)\}$, $n_{MIN}=n$;

(7)　　n_{MIN} 的卫星节点 $\to Q_s$

(8)　　FOR each m in $Q_?$ DO

(9)　　FOR each p in Q_s DO

(10)　　$MIN=\min\{MIN, c(p, m)\}$,

(11)　　$m_{MIN}=m$, $p_{MIN}=p$;

(12)　　$Q_!\cup\{m_{MIN}\}\to Q_!$,

$Q_?\setminus\{m_{MIN}\}\to Q_?$,

$V_t\cup\{P(p_{MIN}\to m_{MIN})$ 包含的节点$\}\to V_t$,

$A_t\cup\{P(p_{MIN}\to m_{MIN})$ 包含的边$\}\to A_t$

(13) 输出最后的组播树 $T=(V_t, A_t)$

算法结束

STMPH 是 LSMPH 的简化算法. 按照上面的算法, LSMPH 算法省略(7), (8), (9) 和(10) 后就是 STMPH.

假设每一个节点到每一个端节点的最短路径已知, 总节点数为 n, 端节点数为 m, 总边数为 e, 则 MPH 算法的计算时间复杂度为 $O(m^2n+e)$[3]. LSMPH 算法第(4) 步到第(12) 步的最大运行次数为 mn, WHILE 循环中的最大运行次数为 m. 考虑算法的第(12) 步中的节点和边的加入次数最大不超过网络的总边数 e, 因此 LSMPH 算法的计算时间复杂度为 $O(m^2n+e)$. 实际中, 第(4) 步到第(12) 步的运行 1 遍的次数远远小于 n, 因此实际的时间远比上述结果小. 后面的实验验证了这个分析. 类似地, STMPH 算法的运行时间为 $O(m^3+e)$.

4　仿真及分析

本文着重于实际的性能分析, 而不是图论中的性能理论上界. 本文提出的算法的最差性能上界正在研究, 但我们更感兴趣的是算法在实际中的平均性能. 实际中, 性能最差的情况极少发生. 下面的实验研究算法的平均性能. 对于节点很多的网络图, 求最优组播树在时间上是不允许的. 因此, 本文中将新算法与 MPH 算法进行比较. 为了更进一步分析 LSMPH 和 STMPH, 我们采用多组随机网络模型进行实验, 求统计平均, 比较 MPH 与 LSMPH 和 STMPH 的费用, 运行时间.

4.1　随机网络模型

为了产生随机网络图, 我们采用文[4] 中的方法. 节点随机地分布在直角坐标系内. 为了讨论的方便, 我们将有向图简化为无向图. 边 (u, v) 分布的概率为:

$$P(u, v)=\beta e^{-d(u, v)/L\alpha} \tag{1}$$

这里 $d(u, v)$ 是 u 和 v 之间的 Euclid 距离, L 是任意两节点间的最大距离, α 和 β 是调节网络图特征的参数. α 增加, 长边相对短边的比也增加; β 增加, 节点的度也随着增加. 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. 为了随机地产生边的费用分布, 我们将边的费用限制在

R_1 和 R_2 之间 ($R_1 \leqslant R_2$), $c(u, v) = \mathrm{rand}(R_1, R_2)$. 本文中将文献[4]中的边$(u, v)$分布的概率调整为:

$$P(u, v) = \beta M_1/(n\log_{10} n) \cdot \exp[-d(u, v)/(L\alpha M_2/\log_2 n)] \quad (2)$$

实验表明, 根据公式(2)产生的网络图的点度一致性很好. 如果要求所产生的网络的平均点度为 μ, 就需要对网络的点度进行限制. 本文采用的方法是, 在网络的产生过程中对每一个点的度进行累加, 当和超过 μ+ 2 时不再与该点连接, 也就是该点的度不再增加. 同时对平均点度进行计算, 当平均点度达到或超过 μ 时网络便生成了.

影响网络组播树生成算法的参数包括: (1) 端节点数, (2) 网络总节点数, (3) 非对称性, (4) 网络密度, (5) 长边对短边的比. 本文中的仿真实验只研究参数(1)和(2)对算法的影响.

4.2 仿真及分析

本文仿真不考虑网络的不对称性. 仿真所用的参数为: L = 100, α = 0.4, β = 0.9, μ = 4.00, M_1 = 25, M_2 = 4, R_1 = 10, R_2 = 100. 网络节点数以 n 表示, 端节点数用 m 表示. 每次计算前, 按公式(2)产生一个随机网络图, 然后从中随机选取源节点 s 和端节点. 每种算法都对每个 m 分别进行 100 次实验. 图中的每个点都是 100 次实验的平均结果. 仿真中涉及到的两个概念定义如下: 设 T_A 是算法 A 生成的组播树, $c(T_A)$ 是组播树 T_A 的费用, 则算法 A 相对于算法 MPH 的费用超出量 δ_A 为: $\delta_A = [c(T_A) - c(T_{\mathrm{MPH}})]/c(T_{\mathrm{MPH}}) * 100$. 算法 A 的运行时间与 MPH 算法运行时间的百分比为 $\varepsilon_A = t_A/t_{\mathrm{MPH}} * 100$. 运行时间指的是计算机 CPU 在完成算法时的消耗时间. 文中用 δ_A 代表算法 A 的费用性能, 用 ε_A 代表算法 A 的时间性能(也称为计算效率).

第 1 组仿真中, 网络节点数 n = 100, 端节点数 m 在 10 到 55 之间变化. 仿真的目的是对算法的费用性能和算法复杂度进行实验, 讨论算法受端节点数影响的程度. 如图 1 和图 2 所示. 从图 1 可以看出, n = 100 时, LSMPH 算法的费用稍微超过 MPH 算法, 超过量最大约为 1.9%; STMPH 算法的总费用超过量最大不超过 3.0%. 由于 STMPH 是 LSMPH 的简化形式, 这两者的性能变化是相似的. 从图 1 可以看出, 当网络规模一定时, LSMPH 算法和 STMPH 算法的费用性能波动趋势基本相同; 从整体来讲, 这两种算法的费用性能随端节点数增大而提高. 图 1 中的局部波动的产生原因尚不清楚. 从图 2 可以得出, LSMPH 和 STMPH 的算法计算效率高于 MPH 算法. 例如当 m = 40 时, LSMPH 算法的相对计算时间为 MPH 算法的 82%, 而 STMPH 则占 79%. LSMPH 算法和 STMPH 算法的相对计算时间最大分别为 MPH 算法的 87% 和 85%. 端节点数较小时, 端节点数和一个端节点的卫星节点数量相当, 搜索卫星节点的时间占有较大比例. 当端节点数较大时, 计算时间主要取决于端节点. 因此, 图 2 中 LSMPH 算法的相对计算时间在 m 为 10 到 20 时缓慢降低, m > 20 时才缓慢上升.

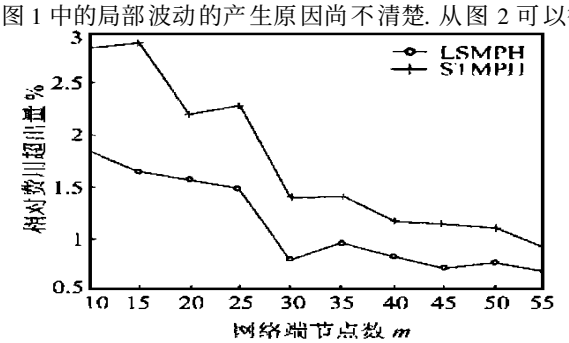

图 1 端节点数对组播树费用的影响

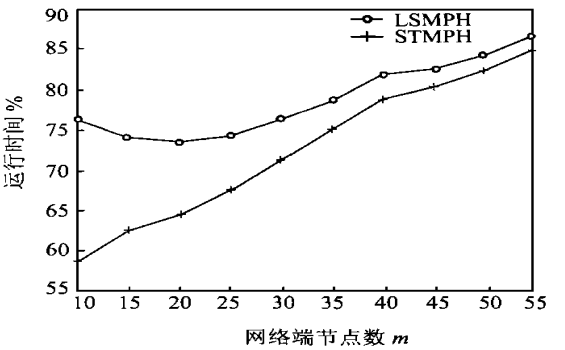

图 2 端节点数对算法时间的影响

在第 2 组仿真中, 端节点数 m = 20, 网络节点数 n 在 120 到 300 之间变化. 仿真的目的是研究算法性能随网络规模的变化情况. 如图 3 和图 4 所示. 从图 3 可以看出, LSMPH 算法的费用稍微超过 MPH 算法, δ 最大不大于 2.65%; STMPH 算法的 δ 最大不超过 3.55%. 随着网络规模的增大, δ 也随之增大. 图 3 中两种算法的变化趋势几乎相同. 图 3 中的小起伏与算法的搜索深度有关. LSMPH 比 STMPH 的搜索深度大, δ 的起伏就小. 从图 4 可知, LSMPH 和 STMPH 的计算时间小于 MPH 算法. 例如当 n = 300 时, $time_{\mathrm{LSMPH}}$ 为 62%, 而 $time_{\mathrm{STMPH}}$ 为 49%. 端节点数一定时, 这两种算法的计算效率随网络规模的增大而提高. 这是因为这两种算法的计算效率主要与端节点数有关. 端节点数一定时, 这两种算法比较的次数也是一定的. 而 MPH 算法的比较次数主要取决于要比较的 Steiner 节点, 随网络规模的增大而增大. 因而新算法的相对计算时间随网络规模的增大而降低, STMPH 算法比 LSMPH 算法降低得稍快. 因为随着网络规模的增大, 端节点的卫星节点数增加, 在端节点与某个端节点的卫星节点之间进行比较的次数就增加, 对 LSMPH 算法来说, 相对计算时间也就稍微增加了.

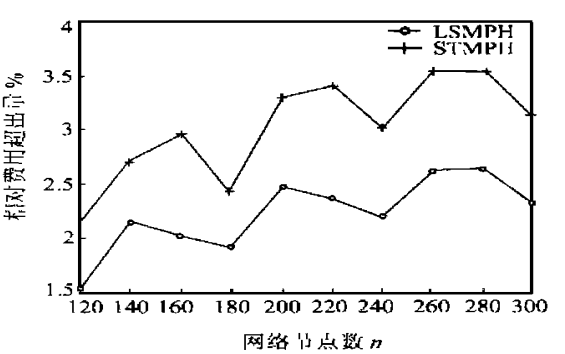

图 3 网络节点数对组播树费用的影响

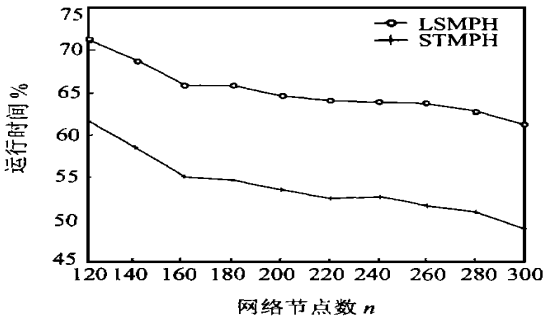

图 4　网络节点数对算法时间的影响

5　结论

本文对边对称情形下的组播树生成算法的实验性能进行了仿真. 从仿真结果看, 在网络节点数和端节点数变化的条件下, LSMPH 和 STMPH 算法性能稳定, 比 MPH 算法更适合于更大的网络. LSMPH 和 STMPH 算法是紧密相关的, 在实际中可以根据需要决定选用哪一种.

参考文献

[1] 卢开澄, 卢华明著. 图论及其应用. 北京: 清华大学出版社, 1995, 8

[2] Pawel Winter. Steiner problem in networks: A survey. Networks, 1987, 17: 129~ 167

[3] S. Ramanathan. Multicast tree generation in networks with asymmetric links. IEEE/ ACM Trans. On Networking, August 1996, 4(4)

[4] Bernard M. Waxman. Routing of multipoint connections. IEEE Journal on Selected Areas in Communications, December 1988, 6(9)

[5] Anees Shaikh, and Kang Shin. Destinatiorr driven routing for low cost multicast. IEEE Journal on Selected Areas in Communications, April 1997, 15(3)

李汉兵　1970 年出生, 博士生, 1997 年 3 月于西安电子科技大学获得通信与电子系统工学硕士学位. 现正在攻读信号与信息处理博士学位. 主要研究方向有卫星通信, 计算机通信, 网络路由算法, 高速路由器技术等.

喻建平　1968 年出生, 1995 年获西安电子科技大学通信与电子学博士学位, 深圳大学副教授. 从事信息安全保密技术, 通信与计算机信息系统的研究.

谢维信　1941 年出生, 西安电子科技大学博士生导师, 深圳大学校长兼信息工程学院院长, 从事信号与信息处理和计算机通信的研究工作.

(上接第 91 页)

[6] 于宗光, 许居衍, 魏同立等. 应用科学学报, 1997, 15(1): 82~ 87

[7] Nozawa H, Matsukawa N, Morita S. IEEE Trans on Electron Device, 1986, 33(1): 275~ 280

[8] Atsumi S, Kuriyama M, Umezawa A, et al. IEEE J Solid-State Circuits, 1994, 29(4): 461~ 469

[9] Kawahara T, Kobayashi T, Jyouno Y, et al. IEEE J Solid-State Circuits, 1996, 31(11): 1590~ 1600

[10] Kim J_ K, Sakui, K, Lee, S_ S, et al. IEEE J Solid-State Circuits, 1997, 32(5): 670~ 678