

# 一种基于策略函数的应用层组播路由算法

潘国庆,李陶深

(广西大学 计算机与电子信息学院,广西 南宁 530004)

**摘要:**由于IP组播存在可扩展性差、难以管理等方面的缺陷,研究人员提出了应用层组播。实时传输是应用层组播技术的一个主要应用领域,对网络延迟有严格的限制。文中着重研究构建最小延迟应用层组播树的算法,提出一种基于策略函数构造应用层最小直径组播树的启发式算法 BCT-H。该算法采用策略函数迭代的选择使生成树直径最短的路径,从而有效地减少了网络中的转发时延和同一条链路的重复分组数量。模拟实验表明该算法能够有效地降低链路强度,减少组播树的时延。

**关键词:**应用层组播;带宽;时延约束;策略函数;组播树

**中图分类号:**TP313

**文献标识码:**A

**文章编号:**1673-629X(2008)05-0138-03

## An Application Layer Multicast Routing Algorithm Based on Strategy Function

PAN Guo-qing, LI Tao-shen

(School of Computer, Electronics and Information, Guangxi University, Nanning 530004, China)

**Abstract:** Due to sparse deployment of IP multicast in the Internet, application-layer multicast (ALM) has been proposed. Real time transmission, which is delay sensitive, is an important aspect of application-layer multicast. It is crucial to build an efficient multicast tree to guarantee the lower delay. Focuses on the algorithms of the minimum-delay spanning tree for the application-layer multicast, and proposes an improved BCT-H (balance compact tree-heuristic) algorithms based on strategy function. By using strategy function, this algorithm iteratively chooses the path which makes diameter of the spanning tree the shortest, thus decreases the transmitting latency of the network and the number of repeated grouping in the same link path. The simulation results show that the algorithms can effectively reduce link stress and decrease multicast tree latency.

**Key words:** application layer multicast; bandwidth; delay limited; strategy function; multicast tree

### 0 引言

由于许多技术和商业原因,IP组播在Internet范围内没有得到广泛部署和应用。一些研究者开始反思IP组播体系本身存在的问题,并提出了应用层组播(Application Layer Multicast)作为替代方案。应用层组播自2000年提出后,马上成为国内外研究的重点。

目前比较著名的应用层组播协议有 Narada<sup>[1]</sup>, NICE<sup>[2]</sup>, Yoid<sup>[3]</sup>, ALMI<sup>[4]</sup>和 Scribe<sup>[5]</sup>等,这些协议本身都有各自的优缺点。应用层组播的主要追求目标是:如何运用一个好的算法形成一棵最优组播树,保证组播参与者尽快收到组播包。对于如何在覆盖网(over-

lay network)上,运用组播算法形成一棵最优组播树可以归结为两个问题:一是如何抽象问题模型,即怎样将覆盖网映射成图;二是在映射图中如何生成最小延迟生成树。

求解应用层组播树算法的主要差异是研究者根据不同的应用需求提出的不同的问题模型。然后根据提出的模型,设计出不同的求解算法。如何设计良好的组播路由算法来满足实时的多媒体应用是当前应用层组播研究的热点和难点。在构造一棵组播树的时候,有多种因素要被考虑,比如带宽、时延、抖动和开销。而基于多约束来构造一棵组播树的问题已经被证明是一个NP完全问题<sup>[6]</sup>。对于这类问题不存在多项式的时间解,只能寻求最优解。

文中针对 MDLL (Minimum-Diameter Degree-Limited spanning tree problem)<sup>[7]</sup>模型中的 CT (Compact Tree) 算法存在的问题,提出了一种改进的基于策略函数的应用层组播路由算法 BCT-H (Balance Compact

收稿日期:2007-08-19

基金项目:广西自然科学基金资助项目(桂科自0640026)

作者简介:潘国庆(1976-),男,广西人,硕士研究生,研究方向为组播技术、网络路由;李陶深,教授,CCF会员,研究方向为分布式数据库、网络信息安全、网络路由算法。

Tree-Heuristic)。实验表明该算法能平衡带宽和时延,在降低链路强度和减少时延方面较CT算法有明显的提高。

## 1 CT 算法

为了叙述问题方便,先将应用层网络映射成无向完全图  $G = \langle V, E \rangle$ ,其中  $V$  是顶点集,代表网络中的端系统。 $E = V \times V$  是边集,表示节点之间的逻辑信道。对于  $e \in E, C(e) \in Z^+$  为其时延属性,表示边  $e(e \in E)$  上的时延。对于  $v \in V, dmax(v) \in Z^+$  表示节点的度数(等同于节点  $V$  的带宽),表示一个端系统最多能向其它端系统复制和转发分组的数目。

CT 算法是 Shi 等人在文献[8]中提出的一个集中式的贪心路由算法。其主要思想是:在输入为  $G = (V, E)$ ,链路代价为  $C(u, v), u, v \in V$ ,节点最大度为  $dmax(v)$  的情况下,选择度最大的节点(假设为  $A$ ),使得当前树直径最小,并考虑度限制最大限度地与其他节点建立路径。反复上述一过程直到构造出一棵生成树。

从 CT 算法可以看出,虽然 CT 算法在节点加入过程中考虑了节点度,但并没有考虑节点度的平衡分配问题,也就是说,网络中的负载分布不均衡。对于应用层网络来说,每个端主机的转发能力和处理能力不同且都受到限制,这就有可能造成了有些处理能力较弱的主机因为任务太多,处理较慢,而处理能力强的主机反而空闲。显然这种算法存在极大的缺陷。

用图说明 CT 算法存在的不足。以图 1(a)所示一个五个节点的完全图为例,  $A, B, C, D, E$  分别代表端主机,边上的数字表示时延,节点括号中的数字表示该节点的最大度限制。根据 CT 算法的思想,得到图 1(b)所示的最小直径生成树,生成树的直径 = 8,路径为  $C-B-E$ 。可以看出节点  $B$  的度为 4,也就是说不管以任意节点为源节点发送数据,节点  $B$  至少需要处理和转发 3 份数据。而具有同样处理能力的节点  $D$  的度数为 1,则不用转发数据。造成负载分布极不均衡。节点  $B$  由于任务多而处理转发时延增长,同时节点  $D$  却处于空闲状态。这样既增加了生成树的时延,又浪费了节点  $D$  的资源。该算法缺乏均衡流量负载的能力,结果造成了系统的利用率较低和会话拒绝率高。

## 2 BCT-H 算法

CT 算法是在 MDDL 模型上提出的一种度约束的最小延迟的组播树生成算法,它没有考虑主机的处理能力,也没有全面考虑度约束和延迟的平衡问题,从而导致网络资源利用率低,主机负载极不均衡。生成的

组播树看上去很粗壮,但是极不稳定。为解决上述问题,提出一种改进的 BCT-H 算法。

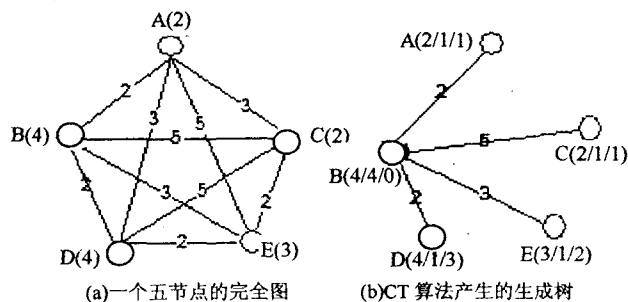


图 1 CT 算法的实例

BCT-H 算法描述如下:首先选定节点度最大的节点为源节点,接着选择  $m$  个使当前生成树直径最小的节点作为待选节点;然后在每一个节点处迭代地选择下一步的待选节点,直到迭代深度为  $n$ ,这时将产生  $mn$  条待选路径;再根据设计的策略函数从待选路径中选择策略函数最大的一条路径加入生成树中。重复上述过程,直到全部的节点都在生成树中为止。如果存在策略函数相同的路径,则选择包含离源节点最远的节点的路径,因为这样可以使生成树中的单播路径能够较多复用,减少了分组复制和降低链路的数据冗余度。

当  $m = 1, n = 1$  时, BCT-H 算法为 CT 算法。当  $m =$  节点的度数,  $n =$  节点数目 - 1 时, BCT-H 算法相当于洪泛算法,即选择所有可能的路径。因此,  $m$  和  $n$  的取值直接影响到构建的生成树。为了寻找剩余带宽平衡且最小化直径的生成树,综合考虑了度约束和最小延迟,提出了以下的策略函数:

$$F(u, v) = a \times 1/n \sum CB(i) + b \times CD(u, v) \\ 1 \leq i \leq n; a, b \in (0, 1) \text{ 且 } a + b = 1$$

其中  $CB(i) = (B - B0)/B, B0 \in (0, B), CB(u) \in [0, 1], B$  表示节点  $i$  的最大度约束,  $B0$  表示当前树上节点  $i$  分配的度,  $P(u, v)$  表示部分生成树中任意两节点最长距离所在的路径,  $CB(i)$  越大说明节点度分配越均匀;  $CD(u, v) = D/D0, D \leq D0, CD(u, v) \in (0, 1), D$  表示从节点  $u$  到节点  $v$  的最短单播时延,  $D0$  表示当前生成树上通过节点  $v$  的最大时延,  $CD(u, v)$  越大说明生成树的直径越小。

从上式可以看出,  $F(u, v)$  越大,对于生成树来说,节点的负载分布越均衡,节点的时延越小。式中的系数  $a, b$  分别表示用户对带宽和时延的关心程度。

BCT-H 算法的具体步骤描述如下:

输入:图  $G(V, E)$ ,图中节点的  $dmax(v)$ ,边权值  $c(e)$ ,策略函数  $F(u, v)$ 。

输出:树  $T$ 。

Step1: 初始化待求树  $T =$ , 将度最大的节点纳入到树  $T$  中。  $T = \{W = \{r\}, L = \{\}\}$

Step2: 从  $G$  中选择  $m$  个满足带宽限制且使树直径最小的节点纳入集合  $J$  中;

Step3: 对集合  $J$  中的任一节点重复(1) $n$  次或直到邻居节点为空或已经加入;

Step4: 从集合  $J$  中的  $mn$  条路径中选择一条根据策略函数  $F(u, v)$  值最大的路径;

Step5: 将路径  $P(V, E)$  纳入到树  $T$  中, 即  $W = W + [V], L = L + [E]$ ; 更改节点的  $dmax$ ;

Step6: 重复 Step2 至 Step5, 直到所有的节点都纳入到  $T$  中。

### 3 模型的性能分析

为了评估 BHT-H 算法的优化效果, 采用 GT-ITM<sup>[9]</sup> 网络拓扑生成器来产生 Transit-Sub<sup>[9]</sup> 网络拓扑, 并用 NS-2.27 进行了仿真实验, 通过实验用节点的平均时延和链路强度等两个指标对 BCT-H 算法和 CT 算法进行了比较分析。实验的过程如下:

(1) 随机生成 100 个节点, 从中任意选取 6 个节点进行 10 次实验, 然后求每个节点的时延, 最后取平均值, 得到如图 2 的结果。

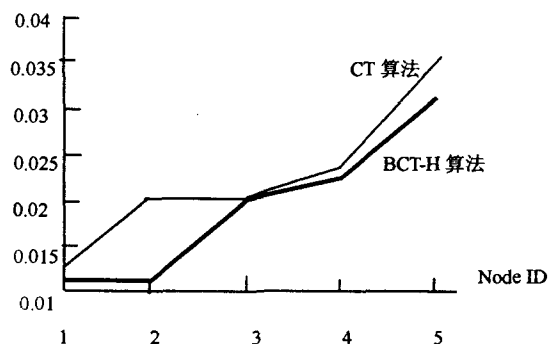


图 2 平均时延的实验结果

(2) 对平均链路强度的实验, 用 GT-ITM 网络拓扑生成器随机生成节点为 16, 32, 64, 128, 256 的网络拓扑, 在每个拓扑图中随机选取 6 个节点进行实验, 最后取平均值, 得到如图 3 的结果。

通过仿真实验可以看出, 与 CT 算法相比, 平衡了节点剩余度, 有效地减少了链路强度, 降低了时延。

### 4 结束语

针对 MDDL 模型中的 CT 算法存在的问题, 提出了一种改进的基于策略函数的 BCT-H 算法, 并通过仿真实验对 BCT-H 算法和 CT 算法进行了分析比较, 说明了 BCT-H 算法能有效地降低链路强度和节点的平均时延。从理论上讲, 因为 BCT-H 算法采用

了策略函数, 该策略函数同时兼顾了深度和广度优先搜索, 因此在性能是优于 CT 算法的。实验结果与理论分析是吻合的。

文中的算法还没有考虑端系统自身所产生的时延, 如何将建立端系统的时延模型并加入到 BCT-H 算法将是下一步要解决的问题。

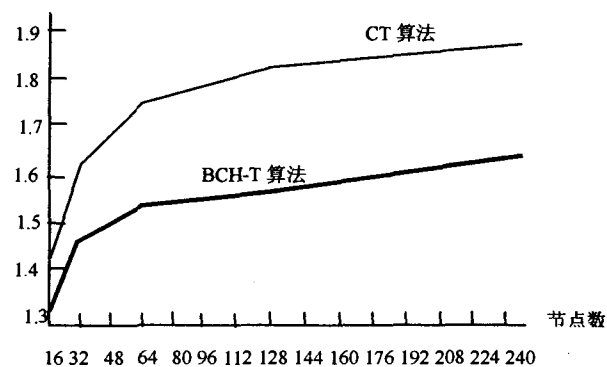


图 3 平均链路强度的实验结果

### 参考文献:

- [1] Chu Y-H, Ran S G, Zhang H. A case for end system multicast[C]// Proceedings of the ACM SIGMETRICS. Santa Clara: ACM Press, 2000: 1-12.
- [2] Bannerjee S, Bhattacharjee B, Kommareddy C. Scalable application layer multicast[J]. ACM SIGCOMM Computer Communication Review, 2002, 32(4): 205-217.
- [3] Francis P. Yoid: Extending the Multicast Internet Architecture[EB/OL]. 1999. <http://www.aciri.org/yoid/>.
- [4] Pendarakis D, Shi S, Verma D, et al. ALMI: An application level multicast infrastructure[C]// In: 3rd Usenix Symposium on Internet Technologies and Systems (USITS). San Francisco: USENIX Press, 2001: 49-60.
- [5] Castro M, Druschel P, Kermarrec A M, et al. SCRIBE: A large-scale and decentralised application-level multicast infrastructure[J]. IEEE Journal on Selected Areas in Communications, 2002, 20(8): 100-110.
- [6] Wang Z, Crowcroft J. Bandwidth-delay based routing algorithms[C]// IEEE Proceedings of Global Telecommunications Conference. Singapore: IEEE Press, 1995: 2129-2133.
- [7] Shi S Y, Turner J S. Multicast routing and bandwidth dimensioning in overlay networks[J]. IEEE Journal on Selected Areas in Communications, 2002, 20(8): 1444-1455.
- [8] Shi S Y, Turner J S. Routing in overlay multicast networks [C]// Proceedings of IEEE INFOCOM. New York: IEEE Press, 2002: 1200-1208.
- [9] Zegura E W, Calvert K, Bhattacharjee S. How to model an internetwork[C]// Proceedings of the 15th Annual Joint Conf of the IEEE Computer and Communications Societies. New York, USA: [s.n.], 1996: 594-602.