文章编号:1001 - 9081(2006)06 - 1292 - 03

一种快速低代价延迟受限组播路由算法

孙丽霞,李仁发
(湖南大学 计算机与通信学院, 湖南 长沙 410082)
(slx5460@126.com)

摘 要:为满足实时业务的 QoS 要求,在非延迟受限组播路由算法(Fast Low-cost Shortest Path Tree,FLSPT)的基础上添加了延迟约束,使得生成的组播树上,每条从源到目的地的路径都满足给定的延迟限制,同时保持了原算法计算复杂度低,代价性能优越的特点。仿真结果表明,本文算法的代价和时间性能均优于延迟受限最短路径(Delay-Constrained Shortest Path, DCSP),且更适合用于目的节点分布集中的密集模式下。

关键词:快速低代价;延迟受限;组播路由
中图分类号: TP393.04 **文献标识码**:A

Delay-constrained fast low-cost multicast routing algorithm

SUN Li-xia, LI Ren-fa
(*School of Computer and Communication, Hunan University, Changsha Hunan* 410082, *China*)

Abstract: In order to satisfy the QoS requirements of real-time applications, the algorithm added delay constrains upon a non-delay-constrained multicast routing algorithm FLSPT was proposed to ensure that the accumulated delay from source to any destination along the tree does not exceed the pre - specified bound, as well as keep the outstanding performance on time complexity and cost of it. Simulation results show that, our algorithm outperforms DCSP in terms of cost and time complexity, and is more suitable for the dense mode, that is, multicast destination nodes are densely distributed in the area.

Key words: fast Low-cost; delay-constrained; multicast routing

组播是将信息从源节点同时发送到网络中多个目的节点的通信方式,它构建树形路由,只在树的分叉点进行信息复制,共享路径上只传输信息的一份拷贝。相比于单播通信方式,组播可以节省网络资源。将组播技术应用于实时多媒体业务中,可以显著提高网络传输效率,节省带宽,但同时也面临较大的挑战,因为这一类业务对信息传递时延有严格的要求,因此,延迟受限组播路由算法显得日益重要。延迟受限组播路由问题可以归结为:组播树的总代价最小;同时每条路径都满足给定的延迟限制的延迟受限 Steiner 树问题,它是一个 NP 完全问题,一般采用启发式算法[1~3,9]。

本文基于非延迟受限组播路由算法 FLSPT 给出了具有时延限制的快速低代价组播路由算法(Delay-Constrained fast low-cost multicast routing algorithm)。算法构造的组播树具有较低的总代价,保证树上的每条路径都能满足给定的延迟约束,是一种快速有效的算法。

1 问题描述及相关工作

1.1 问题描述

用有向图 $G = (V,E)$ 来表示网络,其中 V 是所有节点的集合,每个节点代表路由器或者交换机,E 是边的集合,每条边代表节点间逻辑的或者物理的连接。$|V|$ 和 $|E|$ 代表节点和边的数量。每条链路都是双向的,也就是说,存在一条从 u 到 $v(u,v \in V)$ 的链路 $e = (u,v)$,就意味着,另一条链路 $e' = (u,v)$ 也存在。每条链路 $e \in E$ 有一个代价 $C(e): E \rightarrow R^+$ 和一个延迟 $D(e): E \rightarrow R^+$ 与之关联。函数 $C(.)$ 定义了要优化(最小化)的度量。函数 $D(.)$ 定义了要约束(限制)的度量。因为计算机网络中的不对称性,可能会出现 $C(e) \neq C(e')$ 和 $D(e) \neq D(e')$。方便起见,本文假定网络是对称的,且链路代价和链路延迟都为正整数。

用集合 $M = \{\{s\} \cup R\} \subseteq V$ 表示组播组,其中 s 是一个源节点,它发送数据给一系列接收者,这些接收者的集合用 R 表示。同样,$|M|$ 和 $|R|$ 分别代表组播节点数量和接收节点数量。组播树 $T(s,R) \subseteq E$,以 s 为根且包含 R 中所有节点。我们用 $P_T(r_i) \subseteq T$ 表示 T 中组成从 s 到 $r_i \in R$ 的路径的所有链路的集合。从 s 到 r_i 的总路径代价是 $P_T(r_i)$ 中所有链路的代价之和,用 $Cost[r_i]$ 表示:

$$Cost[r_i] = \sum_{e \in P_T(r_i)} C(e) \quad (1)$$

从 s 到 r_i 的总路径延迟是 $P_T(r_i)$ 中所有链路的延迟之和,用 $Delay[r_i]$ 表示:

$$Delay[r_i] = \sum_{e \in P_T(r_i)} C(e) \quad (2)$$

树的总代价是树上所有链路的代价之和,用 $Cost(T)$ 表示:

$$Cost(T) = \sum_{e \in T} C(e) \quad (3)$$

设实时业务中的端到端延迟限制为 Δ。所以延迟受限 Steiner 树问题可以描述为,找到一棵以 s 为根,包含集合 M 中的所有节点,且满足下面两个要求的树:

1) Minimum $Cost[T]$

2) $Delay[r_i] < \Delta, \forall r_i \in R$

收稿日期:2005 - 12 - 29;**修订日期**:2006 - 02 - 27 **基金项目**:中国网上教育平台(计高技 2034)

作者简介:孙丽霞(1981 -),女,湖南临武人,硕士研究生,主要研究方向:无线网络与通信; 李仁发(1957 -),男,湖南长沙人,教授,博士生导师,博士,主要研究方向:嵌入式计算、无线网络、SOC 设计、网络安全与对抗、数字化实验与仿真技术.

1.2 相关工作

延迟受限 Steiner 树问题是一个 NP 完全问题,文献[2]给出了一个该问题的最优算法,计算时间非常长,所以一般采用启发式算法[1~3,9,10]。文献[3]提出的 BSMA 算法平均性能仅与最优解相差 7%,是目前最好的延迟受限组播路由算法,但它的时间复杂度($O(k|V|3\log|V|)$)太高,其中 k 是指搜索到一条延迟受限最短路径的收敛次数),不适合求解大型网络。并且,该算法是一个源路由算法,要求源节点存储全网拓扑信息[4]。

从实现方式上来说,有一类组播路由算法是通过在代价和延迟之间进行折中来构建组播树的,首先基于代价构建初始树,然后将不满足延迟约束的路径用最小延迟路径代替。代表算法有 SL[9],CDKS[10],DCSP[1],计算复杂度 SL > CDKS = DCSP,而代价性能 DCSP 优于 CDKS。这一类算法的特点是实现简单,计算复杂度低,但代价性能相对来说优势较小。本文提出的算法属于这一类。

文献[5]提出的 DDMC(Destination-Driven Multicasting)算法提出了路径共享的思想。文献[6]提出了目的驱动最短路径树算法 DDSP(Destination-Driven Shortest Path)。采用 Dijkstra 算法路径递增的基本思想,并结合 DDMC 算法的目的节点共享路径的方法,从而降低所构造 SPT 的总消耗。文献[7]提出的低代价最短路径树的快速算法(FLSPT),通过改变 DDSP 算法中确定节点的父节点的方法,提高了计算效率。但这些算法都没有考虑路径的延迟限制,不能用于实时业务中。

本文在 FLSPT 的基础上添加了延迟限制,将它扩展为延迟受限组播路由算法,同时保持其良好的代价性能和计算优势。算法首先使用受限 FLSPT 算法构造组播树,得到 $T1$,如果 $T1$ 中没有包含所有的目的节点,则使用改进的 Dijkstra 算法对其余目的节点构造最短延迟路径树 $T2$,合并 $T1$ 和 $T2$ 得到结果树。通过与文献[1]给出的延迟受限组播路由算法 DCSP 进行性能比较,表明本文提出的算法具有更优的代价性能,而计算时间也更短。

2 具有时延限制的快速低代价组播路由算法

2.1 原始算法

因为组播树上共享的路径只要计算一次代价,所以在保证与源节点路径最短的前提下,选择与已计算的目的节点最近的路径,通过目的节点之间共享尽可能长的路径可以降低生成树的总代价,DDSP 算法就是基于该思想的,它采用 Dijkstra 算法路径递增的基本思想,并结合 DDMC 算法的目的节点共享路径的方法,通过用 $rCST$ 向量保存节点到已计算目的节点的最近距离,在有多条最短路径时总是选择能够使 $rCST$ 分量最小的路径,最大限度地与其他目的节点共享路径,因而能够降低最短路径树的总消耗。算法时间复杂度较低,而且每个节点只需知道其邻接节点的信息,无需保存全网络状态。FLSPT 则通过改变确定父节点的方式,提高了 DDSP 的计算效率。算法描述参见文献[7]。

2.2 算法改进

本文提出的算法,首先将 FLSPT 修改为延迟受限组播路由算法,构造一棵包含尽可能多的目的节点的子树;然后对没有包含在这棵树中的目的节点利用改进后的 Dijkstra 算法求得从源节点到这些目的节点的最短延迟路径树;最后将这两棵子树合并,去除可能出现的回路,得到结果树。

第一步:

1) 引入向量 CST,$rCST$,DST,$\triangle$ 和 S. $CST(u)$ 表示节点 u 到源节点 s 的最短路径长度,$rCST(u)$ 表示节点 u 到生成树 T 上最近的目的节点的距离。$DST(u)$ 表示节点 u 到源节点 s 的最短路径的延迟值。$\triangle$ 表示源到目的节点的延迟限制。$S(u)$ 为 1 则表示节点 u 已经计算过,否则表示尚未计算。

2) 初始化向量 CST,$rCST$,DST,$\triangle$ 和 S,对于图 G 中任意节点 u,$S(u)$ 赋值为 0,如果存在源节点 s 到节点 u 的边,则将 $CST(u)$ 和 $rCST(u)$ 赋值为边 (s,u) 的权值 $cost[s,u]$,否则赋值为 ∞。如果存在源节点 s 到节点 u 的边,则将 $DST(u)$ 赋值为边 (s,u) 的权值 $delay[s,u]$,否则赋值为 ∞。初始化生成树 T 以源节点 s 为其根节点,$CST(s)$ 和 $rCST(s)$ 为 0,$S(s)$ 赋值为 1。

3) 选择节点 u,使得:

$$CST(u) = \min\{CST(w) \mid 1 \leqslant w \leqslant n,\ S(w) = 0, DST(w) < \triangle\}$$

如果节点 u 为一个目的节点,则修改 $rCST(u) = 0$,并将节点 u 添加到生成树 T 上。

4) 对于任意节点 v,如果 $S(v) = 0$,且 $cost[u,v] < \infty$,修改此节点的 $CST(v)$ 和 $rCST(v)$ 值。如果:

$$CST(u) + cost[u,v] < CST(v) \text{ 且 } DST(u) + delay[u,v] < \triangle$$

则令:

$$CST(v) = CST(u) + cost[u,v],$$
$$rCST(v) = rCST(u) + cost[u,v],$$
$$DST(v) = DST(u) + delay[u,v]$$

将 v 的父节点设置为 u。

否则,如果:

$$CST(v) = CST(u) + cost[u,v] \text{ 且 } rCST(v) > rCST(u) + cost[u,v] \text{ 且 } DST(u) + delay[u,v] < \triangle$$

则令:

$$rCST(v) = rCST(u) + cost[u,v],$$
$$DST(v) = DST(u) + delay[u,v]。$$

将 v 的父节点设置为 u。

5) 重复执行步骤 3) 和步骤 4),如果未计算的节点中没有满足延迟约束的,T 为中间结果 $T1$,进入算法第二步;否则(所有目的节点都已添加到 T 上),算法终止,T 即为结果树。

第二步:

对于未包含进组播树的目的节点,使用改进后的 Dijkstra 最短路径算法(每个节点只使用了其相邻节点的信息,并且区别目的节点和一般节点)基于延迟计算最短路径树 $T2$。

第三步:

将 $T1$ 和 $T2$ 合并,删除可能出现的回路。为了满足延迟约束,需要保留 $T2$ 中的路径。具体实现为,如果两棵树共有的一个节点 m 在两棵树中具有不同的父节点,那么形成回路。删除 $T1$ 中此节点 m 与其父节点相连的边,然后判断其父节点是否为叶子节点,如果是叶子节点但同时又是目的节点,删边过程结束;如果此父节点是叶子节点但不是目的节点,则将此父节点删除,并同时删除与之相连的边,然后再判断被删除节点的父节点是否为叶子目的节点,依照上面做法直到遇到一个目的节点或非叶子节点为止。上述过程一直重复,直到不存在在两棵树中具有不同父节点的节点。

经过以上步骤,最终得到一棵组播树,且所有的目的节点满足延迟限制。

2.3 算法讨论

定理 1 总能找到满足延迟限制的组播树，只要它存在。

证明 假定对于某个组播源和目的节点集合以及延迟约束，存在满足条件的延迟受限组播树，那么最短延迟路径一定满足延迟约束。设原始图为 $G=(V,E)$，其中源节点为 $s\in V$，组播目的节点为 $M\subseteq V$。

1）设算法第一步得到的结果为 $T1=(V_{T1},E_{T1})$，$V_{T1}\subseteq V$，$E_{T1}\subseteq E$。其中包含的组播节点为 $M_{T1}\subseteq M$。显然 $T1$ 是一棵树，且树上从 s 到 M_{T1} 中所有节点的路径符合延迟约束。如果 $M_{T1}=M$，算法终止，$T1$ 即为结果，满足定理。

2）否则，进入算法第二步，设得到的组播树为 $T2=(V_{T2},E_{T2})$，$V_{T2}\subseteq V$，$E_{T2}\subseteq E$。其中包含的组播节点为 $M_{T2}=M-M_{T1}$。因为 $T2$ 是基于延迟构建的最短路径树，所以 $T2$ 上从 s 到 M_{T2} 中所有节点的路径都满足延迟约束。将 $T1$ 和 $T2$ 合并得到 $T=(V_T,E_T)$，T 包含 M 中的所有节点，下面对两种可能出现的情况，分别进行证明。

第一种情况是合并后没有出现回路，那么 T 即为结果。因为在 T 中，M_{T1} 依然使用 $T1$ 中的路径，M_{T2} 依然使用 $T2$ 中的路径，所以 T 为一棵包含 s 和 M 中所有节点的树，且从 s 到 M 中所有节点的路径都满足延迟约束。满足定理。

第二种情况是合并后出现回路，如图 1。目的节点 $M1$ 在 $T1$ 中的路径为 $s-N1-N2-M1$，$delay(M1)=(Delay1+Delay2)<\triangle$，目的节点 $M2$ 在 $T2$ 中的路径为 $s-N2-M2$，$delay(M2)=Delay3+Delay4<\triangle$，节点 $N2$ 在 $T1$ 和 $T2$ 中具有不同的父节点。根据算法，去除回路以后得到的组播树为 T，在 T 上，$M2$ 的路径保持不变，$M1$ 的路径更改为 $s-N2-M1$，$delay(M1)=Delay3+Delay2$。因为 $Delay1+Delay4>\triangle$，否则 $T1$ 将包含 $M2$，所以 $Delay3<Delay1$，所以 $Delay3+Delay2<\triangle$。所以 T 为一棵包含 s 和 M 中所有节点的树，且从 s 到 M 中所有节点的路径都满足延迟约束。满足定理。

证毕

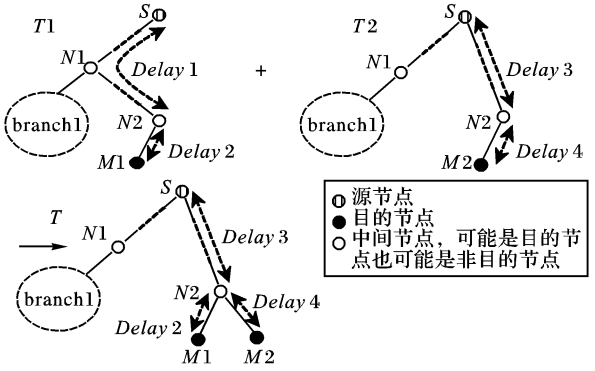

图 1 算法讨论

3 仿真验证

为验证本文提出的算法的正确性和有效性，采用多组随机网络模型进行实验，比较 DCSP 和本文算法产生的结果树的代价及计算时间。

为了产生具有实际特征的随机网络图，我们采用文献[8]中的方法。节点随机分布在直角坐标系内，按均匀分布产生节点的 x 和 y 坐标值。节点 u 和 v 之间存在一条边的概率为：

$$P(u,v)=\beta e^{-d(u,v)/L\alpha} \tag{4}$$

其中，$d(u,v)$ 是节点 u 和 v 之间的 Euclidean 距离，L 是两点之间的最大距离，α 和 β 是两个调节网络特征的参数。如果 α 增加，长边相对短边的比例就增加；如果 β 增加，节点的度就随之增加。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。本文采用与文献[1]相同的参数：$\alpha=0.2$，$\beta=0.65$。边代价与边长度成正比；边延迟与边长度正比，并附加一个随机值相当于链路建立延迟，使之更加符合实际网络特性。延迟限制 $\triangle$ 的值为从源到所有目的节点的最短路径的延迟的最大值。

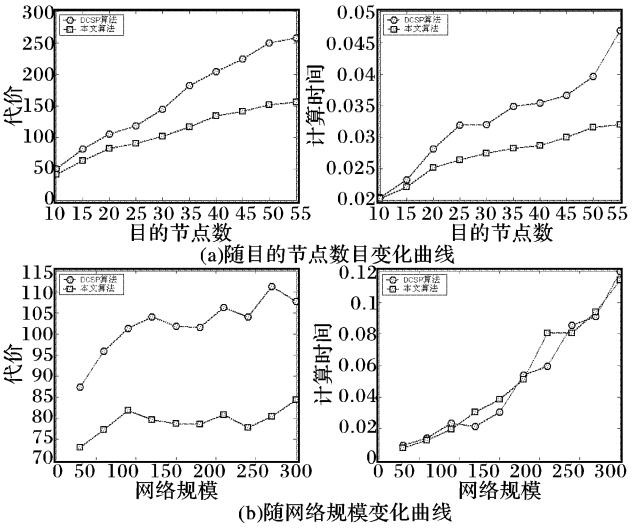

图 2 仿真结果比较

首先固定网络规模，增加目的节点数量，对两个算法进行仿真。对每组数据，产生 100 个随机网络，取平均值为最后结果。图 2(a)分别是节点数固定为 100 时，两个算法的代价和计算时间随网络中目的节点数量变化的曲线。从图中可以看出，本文提出的算法代价和计算性能都优于 DCSP 算法，且随着网络中目的节点比例的增大，代价和计算时间的增长幅度也小于 DCSP，说明算法适合用于目的节点分布密集的情形下。

然后固定网络中的目的节点数量，增大网络规模，对两个算法进行仿真，同样取平均值为最后结果。图 2(b)分别是目的节点数固定为 20，两个算法的代价和计算时间随网络规模变化的曲线。从图中可以看出，本文提出的算法代价性能优于 DCSP，而随着网络规模的增大，计算时间与 DCSP 基本持平。

4 结语

本文在 FLSPT 算法的基础上增加了延迟限制，使构建的组播树能满足实时业务对于信息传输时延的要求。通过与 DCSP 算法进行仿真比较，表明本文提出的算法保持了 FLSPT 算法的代价和计算优势，性能优于 DCSP 算法。

下一步工作是考虑组播会话中节点的动态加入问题，开发快速有效的算法将新组播节点添加到已有的组播树上，将本文提出的算法扩展为动态延迟受限组播路由算法。

参考文献：

[1] FENG G, TAK SHING PETER YUM. Efficient multicast routing with delay constraints[J]. International Journal of Communication Systems. 1999, 12(3): 181 – 195.

[2] WIDYONO R. The design and evaluation of routing algorithms for real – time channels[R]. Technical Report, ICSI TR – 94 – 024, University of California at Berkeley: International Computer Science Institute, June 1994.

[3] PARSA M, QING ZHU, GARCIA-LUNA-ACEVES JJ. An iterative algorithm for delay-constrained minimum-cost multicasting [J]. IEEE/ACM Trans on Networking. 1998, 6(4): 461 – 474.

（下转第 1300 页）

CSMA/CA 机制的吞吐率比 RTS/CTS 的吞吐率高,这也是因为没有考虑隐藏节点问题。节点数量影响着网络负载,节点数量越多,那么网络的负载越重。另外,节点数量越多,无线信道的竞争者就越多,那么信道中的通信量就越多,信道利用率也就越低。所以随着节点数量的增加,网络的吞吐率下降,传输延迟也会增加。信道中的通信量越多就意味着在仿真中产生和处理的消息也就越多。这样一来,仿真的速度就会降低。

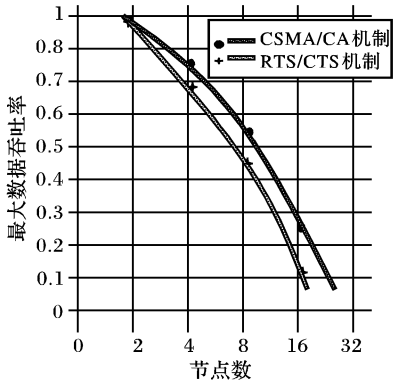

图3 节点数量和吞吐率的关系

4.2 网络负载对仿真结果的影响

在正常情况下,MAC 层对移动 Ad Hoc 网络吞吐率的影响是:点对点机制的吞吐率随网络负载线性增长直到饱和。CMSA/CA 机制和 RTS/CTS 机制在网络负载较轻的时候也随网络负载呈线形增长。但是因为移动 Ad Hoc 层网络中存在隐藏接点问题,当网络负载继续增加时,使用 CSMA/CA 机制,网络开始出现阻塞,吞吐率逐渐趋近于 0。而使用 RTS/CTS 机制,网络并不出现阻塞,而是逐渐趋近于饱和。

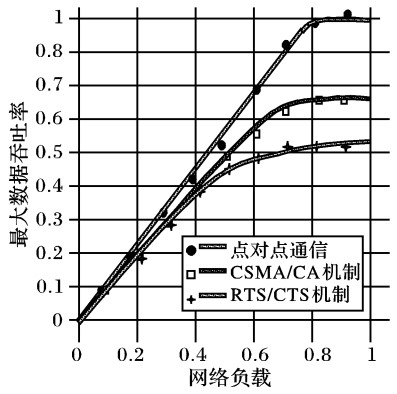

图4 网络负载与数据吞吐率的关系

仿真得出的实验数据如图 4 所示。点对点机制的数据是基本与正常情况下的结果一致,但是 CSMA/CA 机制和 RTS/CTS 机制却有明显的不同。一是 CSMA/CA 机制没有出现阻塞,二是CSMA/CA机制的吞吐率比RTS/CTS机制的吞吐率高。这些不同是由于在仿真时没有考虑隐藏节点问题。在这种情况下,对于 CMSA/CA 机制,即使出现隐藏节点,数据照样可传输,所以网络不会出现阻塞。如果考虑隐藏节点的话,当网络负载继续增加时,使用 CSMA/CA 机制,网络开始出现阻塞,吞吐率逐渐趋近于 0。而使用 RTS/CTS 机制,网络并不出现阻塞,而是逐渐趋近于饱和。

4.3 解决隐藏节点和暴露节点问题的策略

采用双信道的方法,即利用数据信道收发数据,利用控制信道收发控制信号和 802.11b 中的 MAC 协议的 RTS/CTS 握手机制来解决,隐藏节点和暴露节点的问题基本得以解决。

但由于需要发送 RTS 和 CTS 帧,所以 RTS/CTS 机制会给无线局域网带来很多额外的开销。当发送短数据包时,RTS/CTS 机制增加了许多开销,所以在发送短数据包时,可以不使用 RTS/CTS 机制。不用 RTS/CTS 机制发送数据包时,需要定义出数据的最小长度,这个长度叫做 RTS 阈值。只有数据包的长度大于 RTS 阈值时,才可以使用 RTS/CTS 机制发送数据。当帧长度超过 RTS/CTS 阈值时,工作节点就采用 RTC/CTS,若没有超过就采用 DCF,这是一种混合模式。用户也可以关闭 RTS/CTS 机制,这样帧长度即使超过 RTS/CTS 阈值,工作节点仍然使用 DCF 来传输数据。

5 结语

网络的路由和协议是当前移动 Ad Hoc 网络的研究重点,而且也是仿真测试的难点,为了让这些协议被有效测试,仿真工具应当能提供逼近实际网络的仿真环境,得出准确的仿真数据供协议研究者使用。本文提出的针对移动 Ad Hoc 网络的特殊性的仿真测试系统,能够在有线网络环境中模拟出无线移动的网络环境来。为了更加逼真地仿真真实的网络环境,应进一步完善 MAC 层仿真,提供上层的接口,使那些用到了 MAC 层功能的上层协议也能仿真,以支持安全性,服务质量等问题的研究。

参考文献:

[1] SUN J-Z. Mobile Ad Hoc networking: an essential technology for pervasive computing[Z], 0-7803-7010-4/01/2001 IEEE.

[2] LUNDGREN H, *et al.* A large-scale tested for reproducible Ad Hoc protocol evaluations[A]. In Proceedings of IEEE Wireless Communications and Networking Conference (WCNC 2002)[C]. Orlando, Florida, Mar. 2002.

[3] KE Q, MALTZ DA, JOHNSON DB. Emulation of multi-hop wireless Ad Hoc networks[A]. In Proceedings of 7th International Workshop on Mobile Multimedia. Communications (MoMuC'00)[C]. Oct. 2000.

[4] http://www.computeruser.com/resources/dictionary/popup_definition.php? Lookup = 8410[EB/OL], 2005.

(上接第 1294 页)

[4] 刘莹,吴建平,王颖. 延迟受限启发式组播路由算法[J]. 高技术通讯, 2003,13(9):1-4.

[5] SHAIKH A, SHIN KG. Destination-driven routing for low-cost multicast[J]. IEEE Journal of Selected Areas in Communications, 1997,15(3):373-381.

[6] ZHANG BX, MOUFTAH HT. A destination-driven shortest path tree algorithm[A]. IEEE International Conference on Communications[C]. Canada(Kingston): 2002. 2258-2262.

[7] 王涛,李伟生. 低代价最短路径树的快速算法[J]. 软件学报, 2004,15(5):660-665.

[8] WAXMAN BM. Routing of multipoint connections[J]. IEEE Journal on Selected Areas in Communications. 1988,6(9):1617-1622.

[9] SUN Q, LANGERDORFER H. A new distributed routing algorithm with end-to-end delay guarantee[A]. In Proc. IFIP Fifth International Workshop on Quality of Service[C]. New York: Columbia University, 1997.

[10] SUN Q, LANGENDORER H. Efficient multicast routing for delay-sensitive applications[A]. In proceedings of Second International Workshop on Protocols for Multimedia Systems (PROMS'95)[C]. Austria (Salzburg): Mozart on Multimedia Highways, 1995. 452-458.