一种新的固定速率分层组播拥塞控制协议*)

张 冰 原 冰 刘增基

(西安电子科技大学综合业务网国家重点实验室 西安 710071)

摘 要 提出了一种新的固定速率分层组播拥塞控制算法 FLMCC。组播会话中的每层按照固定速率发送数据包。各接收端根据估计的期望速率累计订购数目不等的层,从而获得不同的吞吐量。为准确估计期望速率并实现 TCP 友好性,各接收端采用在接收端实现的窗口机制,即在每层独立维护拥塞窗口,利用 GAIMD 算法调整窗口,并根据窗口值计算期望速率。为测量 RTT,采用了一种精确测量和粗略测量相结合的策略;为避免 RTT 精确测量时产生的反馈内爆问题,采用了基于随机定时器机制的反馈抑制策略。协议实现简单。仿真表明,算法具有良好的 TCP 友好性、响应性和协议内公平性,且链路利用率高。

关键词 分层组播,拥塞控制,TCP 友好,GAIMD

Fixed-rate Layered Multicast Congestion Control

ZHANG Bing YUAN Bing LIU Zeng-Ji

(State Key Lab. of Integrated Services Networks, Xidian University, Xi'an 710071)

Abstract A new fixed-rate layered multicast congestion control algorithm called FLMCC is proposed. The sender of a multicast session transmits data packets at fixed rate on each layer, while each receivers obtain different throughput by cumulatively subscribing to deferent number of layers based on their expected rates. In order to provide TCP-friendliness and estimate the expected rate accurately, a window-based mechanism implemented at receivers is presented. To achieve this, each receiver maintains a congestion window on each layer, adjusts it based on the GAIMD algorithm, and an expected rate from the congestion window calculats. To measure RTT, a new method is presented, which combines an accurate measurement with a rough estimation. A feedback suppression based on a random timer mechanism is given to avoid feedback implosions during the accurate measurement. The protocol is simple in its implementation. Simulations indicate that FLMCC shows good TCP-friendliness, responsiveness as well as intra-protocol fairness, and provides high link utilizations.

Keywords Layered multicast, Congestion control, TCP-friendliness, GAIMD

组播拥塞控制是拥塞控制研究的难点。设计组播拥塞控制协议需要解决的主要问题是可扩展性和公平性。可扩展性主要表现在需要解决大量接收端反馈引起的反馈内爆[1]和丢失路径多样性[2]。公平性问题表现在要保证 TCP 友好性(TCP friendliness),以及协议内公平性(intra-protocol fairness)。组播拥塞控制协议设计的困难之处还在于,不同的组播应用有不同的需求,因此很难设计出一种适合各种应用的组播传输协议和拥塞控制协议,这就使得组播拥塞控制面临的问题变得更为复杂多变。

在各种组播拥塞控制机制中,分层组播很有代表性并受到研究者的关注。分层组播的特点是:组播会话内建立多个不同的组播组(称为层),每层以一定的速率发送数据。要发送的数据流也相应地被分解为若干组,分别通过不同的层发送。这些层之间预先规定了上下逻辑关系,订购(subscribe)时要从最低层开始依次订购。各接收端根据其拥塞状况,可以依次订购从最低层开始的若干层;订购某一层后,就可接收该层内发送的数据包。累计订购的层数越多,接收端获得的吞吐量越高。当网络条件变化时,接收端可以通过加入/退出层而增加/减小其接收速率。分层组播可以为同一组播会话内的不同接收端提供不同的接收速率,较好地适应了各接收端网络条件的异构性,同时可更有效地利用网络资源。

分层组播拥塞控制协议可以分为固定速率分层和动态速率分层两种类型。总体而言,已提出的各种协议在性能和实现复杂度方面尚有问题,目前都未得到广泛认同。动态速率分层组播协议[3~4]具有灵活的适应性,但实现机制过于复杂。和动态速率分层算法相比,层速率固定的分层组播算法[5~7]实现简单,无需进行信息反馈或反馈信息量较小,具有较好的可扩展性。但现有算法的缺点是不具备或不能很好实现 TCP 友好性,且在响应网络拥塞时往往导致速率剧烈抖动。

本文提出了一种基于窗口的固定速率分层组播拥塞控制算法 FLMCC(Fixed-rate Layered Multicast Congestion Control)。为保证 TCP 友好性,算法采用了 TCP 友好的 GAIMD 算法,实现拥塞窗口调整。GAIMD 算法和快速变化的 TCP 算法相比,窗口变化更为平缓,因此较好地解决了频繁加入/退出各层引起的速率剧烈抖动问题,选择平滑的 GAIMD 算法同时还提供了很高的链路利用率。为获得 RTT 测量值,采用了一种精确测量和粗略测量相结合的策略;为避免 RTT 精确测量时产生的反馈内爆问题,采用了基于随机定时器机制的反馈抑制策略。仿真表明,本算法可为异构接收端有效地提供多速率服务,具有良好的协议内公平性、TCP 友好性和响应性,并能提供很高的链路利用率。

1 协议描述

FLMCC 协议的设计要点是:发送端的每层按照固定速

*)国家自然科学基金项目资助(90104012)。张 冰 副教授,在职博士生。

率发送数据包;各接收端采用基于窗口的机制判断网络拥塞状况并估计期望速率;根据期望速率,各接收端累计订购若干层,以获得与期望速率相匹配的吞吐量;为保证 TCP 友好性,以及避免接收端频繁的层加入和层退出引起的速率剧烈振荡,协议采用 TCP 友好而平滑的 GAIMD 算法实现窗口调整,GAIMD 算法较好地实现了平滑性和响应性之间的折衷;为保证 TCP 友好性,还需要各接收端测量 RTT,为此采用一种精确测量和粗略测量相结合的策略。

1.1 层设置

分层组播采用累计分层策略。协议首先确定分层配置方案,通常可采用加性分层或指数性分层方案。采用加性分层方案时,各层速率或速率的差值相等;采用指数性分层方案时,从基层(第 0 层)到第 n 层(n 为层序号)的累计速率之和通常具有 a^n(a 为常数,一般选为 2)的形式。本文采用加性分层方案。

为叙述方便,做如下定义:B_i 为从基层到第 i 层的最大累计发送速率,按照加性分层方案,$B_i=(i+1)\times B_0$;R_i 为第 i 层的实际发送速率;TR_i 为从基层(第 0 层)到第 i 层的实际累计发送速率,$TR_i=\sum_{i=0}^{i}R_i$;X_j 为接收端 j 的期望速率。各参数的关系如图 1 所示。由于协议中各层速率固定不变,因此 R_i 和 TR_i 均不随时间变化。

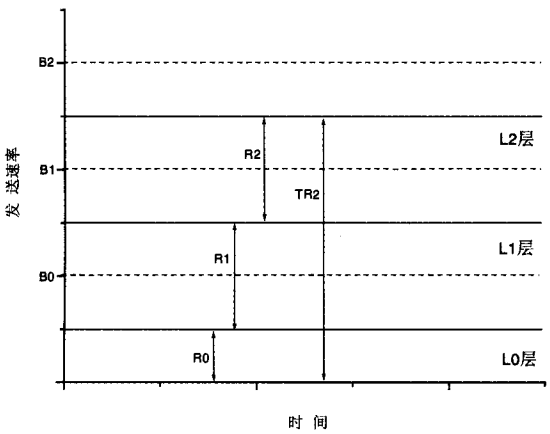


图 1 协议说明

协议规定,每个接收端根据其计算的期望速率从基层开始依次订购一组彼此相邻的层。若期望速率在$(0,B_0]$时,接收端订购第 0 层,此时其接收速率 $TR_0=R_0$;期望速率在$(B_0,B_1]$时,接收端订购第 0 层和第 1 层,此时其接收速率 $TR_1=R_0+R_1$;依此类推。如何确定各层的实际发送速率取决于具体的设计策略。为更好地保证和 TCP 流的友好性,各层的累计发送速率 TR_i 可选取该层最大和最小累计发送速率的中值。因此,若期望速率在$(0,B_0]$时,$TR_0=R_0=B_0/2$,接收端以 $B_0/2$ 接收数据包;期望速率在$(B_0,B_1]$时,$TR_1=(B_1+B_0)/2$,$R_0=B_0/2$,$R_1=B_1/2=B_0$;依此类推。$i>0$ 时,$TR_i=(B_i+B_{i-1})/2$,$R_i=B_0$。当接收端估计的期望速率小于 TR_i 但却获得 TR_i 的吞吐量时,理论上会对 TCP 流造成一定的不公平。但由于接收端的期望速率是时刻变化的,当各层速率间隔不大时,在某些时刻期望速率会小于实际接收速率,而另一些时刻则会大于实际接收速率,因此总体上采用这种策略不会对 TCP 流造成实质上的不利影响。相反,能更好地保证组播流的 TCP 友好性。当然,各层累计发送速率 TR_i 还可以采用保守策略 $TR_i=B_{i-1}(i>0)$。此时组播流获得的累计速率略低于 TCP 流,这可以更严格地保证组播流不会侵害 TCP 流的吞吐量,但却可能造成组播流吞吐量不适当地低于其公平速率。本文中各层发送速率采用前一种方案,即各层累计发送速率 TR_i 为该层最大和最小累计发送速率的中值。

1.2 发送端行为

发送端的行为主要由两部分构成:在各层按照固定速率发送数据包,以及协助各接收端精确测量往返时延 RTT。由于各层发送速率是固定的,无需接收端反馈有关的速率信息,因此和单速率组播或自适应分层组播中需要各接收端反馈速率信息以调整发送速率的方式相比,固定速率分层组播避免了因速率通告而产生的大量反馈。发送端协助接收端进行 RTT 测量的机制在接收端机制中讨论。

1.3 各接收端行为

各接收端估计期望速率,并根据期望速率订购适当的层,从而获得相应的吞吐量。为估计期望速率,协议采用基于窗口的机制,各接收端需要独立地维护并调整拥塞窗口,并估计 RTT。

1.3.1 窗口调整策略

各接收端在其加入的每层都独立维护一个拥塞窗口,并按照 GAIMD 算法[8] 调整窗口的增减。通过选取合适的参数,GAIMD 算法可实现 TCP 友好性,以及平滑性和响应性之间的折衷。GAIMD 算法如式(1)所示:

$$\begin{aligned} &I:\ cwnd_{t+R}\leftarrow cwnd_t+\alpha, && \alpha>0\\ &D:\ cwnd_{t+\delta_t}\leftarrow(1-\beta)cwnd_t, && 0<\beta<1 \end{aligned} \tag{1}$$

式中 I 和 D 分别表示拥塞窗口的增、减;$cwnd_t$ 为 t 时刻的拥塞窗口值(单位为包);R 代表收到一个“窗口”的数据包的持续时间(本文称为“轮”);δ_t 为一足够短的时间;α,β 为窗口调整参数,为满足 TCP 友好性,二者应满足一定关系[8],并综合考虑响应性和平滑性等因素,本文选取 $\alpha=0.20$,$\beta=0.125$。按照(1)式,若在一轮内没有出现丢包,则增大窗口;若发生丢包,则减小窗口。该式提供了一种 TCP 友好的窗口调整策略,和 TCP 相比,窗口的调整要平滑得多,这是产生期望速率平滑变化,并避免频繁层加入/退出的基础。

为在接收端实现窗口机制,协议模拟 TCP 协议,在每层设置了独立的状态转移图,如图 2 所示。设置 4 个状态:慢启动(SS)、拥塞避免(CA)、快速恢复(FR)和超时(TO)。简单工作过程为:最初 *cwnd* 设为 1,当收到第一个数据包时,进入 SS 状态,此时 *cwnd* 按指数方式增加(每隔一轮,*cwnd* 增加 1 倍),以尽快探测网络可用带宽。该过程一直持续至检测到丢包,或 *cwnd* 超过设定的慢启动阈值(ssthresh)为止,并分别转入 FR 和 CA 阶段。发生丢包时,进入 FR 阶段,根据(1)式减小算法,立即减小 *cwnd* 并计算期望速率,等待一个 RTT 后转入 CA 阶段。在 CA 阶段,若一轮内没有丢包,按照(1)式增加算法,增大 *cwnd* 并计算期望速率。当持续拥塞以致接收定时器超时时,转入 SS 状态,并重置 *cwnd* 为 1。

为实现式(1)的窗口调整算法,在接收端引入了“轮”机制来模拟发送端的窗口策略。接收端将连续接收的数据流按照拥塞窗口的大小划分为轮。如图 3 所示,设当前拥塞窗口值为 *cwnd*,在正常无拥塞情况下,每收到序号连续的 *cwnd* 个数据包,就认为经历了一轮。每轮用唯一整数(称为轮号)标识,相邻轮的轮号是连续的,图 3 中 *cwnd*(n)代表第 n 轮的拥塞窗口值。在该轮结束时,按(1)式更新 *cwnd*,并按新 *cwnd* 值开始新一轮。这样,接收端将收到的各个数据包划分为不

同的轮。当检测到拥塞时，当前轮立即结束，按公式减小 *cwnd*，等待一个 RTT 后重新开始新一轮。任何情况下，在每轮结束时，都要更新拥塞窗口，并计算期望速率。状态转移机制和轮机制相互配合，实现了完整的接收端窗口调整功能。

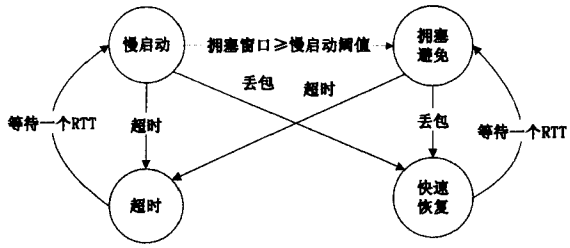


图 2　接收端的状态转移图

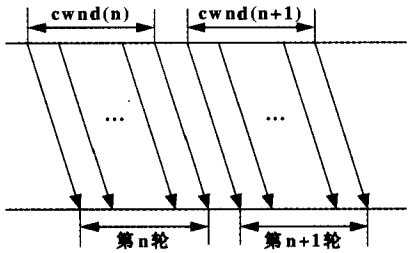


图 3　轮的概念

接收端根据丢包来判断拥塞(事实上也可采用如 ECN 的机制，利用标记来检测拥塞)。在每层内数据包是按顺序发送的，接收端保存着各层每轮内接收数据包的历史记录。当网络由于拥塞产生丢包时，历史记录就会出现间断。若在该间断发生之后累计收到 3 个大于该间断序号的数据包，就认为该数据包丢失，网络发生拥塞，于是结束当前轮，并立即进入 FR 状态。若某层在间断发生后收到的包数小于 3 个但判断发送端已发出 $cwnd-1$ 个数据包，或在一段时间内(一个 RTO)没有收到任何数据包，则认为发生严重拥塞，结束当前轮并转入 TO 状态。

1.3.2　RTT 测量

要获得期望速率，接收端还需要测量到发送端的往返时延 RTT。协议采用精确测量和粗略测量两种相互补充的 RTT 测量机制。精确测量 RTT 采用时间戳机制：某接收端向发送端发送带有时间戳及其自身标识的反馈包，若发送端在其后的数据包中返回该时间戳和接收端标识，该接收端即可计算 RTT。由于组播组可能存在大量接收端，为避免此方法测量 RTT 引起的反馈内爆，协议参考 TFMCC 协议[9]，采用随机反馈定时器和反馈轮(feedback round)机制。这需要发送端和各接收端之间的配合。其简单原理为：在发送端将时间分为连续的时间段，每段称为一个反馈轮，用唯一整数(称为反馈轮号)标识，相邻反馈轮的轮号是连续的；每个反馈轮持续一定时间，其时长为从反馈轮开始到收到当前轮内的第一个反馈报告为止；发送端在其发送的数据包中包含当前反馈轮号。接收端在收到一个标识新反馈轮的数据包后，设置新的随机反馈定时器，其时长 t 设为：

$$t=\max(T\cdot(1+\log(x)/\log(N)),0) \tag{2}$$

式中，x 是(0,1)之间服从均匀分布的随机变量。T 取值 $b\times R\_\max$，$R\_\max$ 为发送端收到的各接收端报告 RTT 的最大值(在发送数据包中携带；为此，发送端利用接收端的反馈报告测量到各接收端的 RTT，并将最大值保存在 $R\_\max$ 中)，b 的取值决定着接收端的平均反馈迟延大小，本文取为 4。N 是组播接收端数目上限的估计值，协议中设为 10,000。

当定时器超时后，接收端发送反馈报告，请求测量 RTT。在收到接收端反馈报告后，发送端通过返回时间戳以协助该接收端测量 RTT；若发送端在连续发送的两个数据包之间收到多个接收端的反馈报告，则先响应以前未获得 RTT 测量值的接收端，其次才响应已获得 RTT 值的接收端；若有多个同类的接收端，则按照先后次序，先到者先响应。在当前反馈轮内，发送端收到任何接收端的反馈报告时，终止当前反馈轮并开始一个新反馈轮(反馈轮号加 1)。接收端在收到带有新反馈轮号的数据包后，取消旧反馈轮内设置的反馈定时器(若有)并重新启动新定时器。利用这一机制，在每一个反馈轮内，大部分接收端的定时器在超时前都被取消，从而不能发送反馈。只有少数接收端在定时器超时后才能发送反馈，因此有效解决了反馈内爆问题。

当组播组规模庞大时，利用以上反馈轮和随机定时器机制，所有接收端获得精确的 RTT 测量值需要相当长的时间。为解决这一问题，协议还规定了粗略的 RTT 测量策略，当接收端尚未获得精确测量 RTT 的机会时，接收端把从发送组播组加入(IGMP Join)请求到收到第一个数据包时的时间间隔，作为接收端计算期望速率时的 RTT 值。

为避免单个 RTT 测量值的随机抖动产生的不良影响，协议还采用了指数加权移动平均(EWMA)算法对 RTT 测量值进行平滑：$t\_rtt=(1-\gamma)\cdot t\_rtt+\gamma\cdot t\_inst$，其中，$t\_rtt$ 为平滑后的 RTT 值，t_inst 为当前 RTT 测量值，γ 为权值因子，取值为 0.1。

1.3.3　期望速率估计

在获得当前拥塞窗口值 *cwnd* 和往返时延 RTT 之后，各层就可以计算出当前期望速率样本值 r：

$$r=cwnd\cdot packet\_size/t\_rtt \tag{3}$$

式中 $packet\_size$ 为数据包大小。为进一步减小速率抖动，利用以下平滑算法，最终得到该层期望接收速率 E_Rate：

$$E\_Rate=\frac{\sum_{i=1}^{n}w(i)\cdot r(i)}{\sum_{i=1}^{n}w(i)}$$

$$w(i)=\begin{cases}1, & 1\leqslant i\leqslant n/2\\ 1-\dfrac{i-n/2}{n/2+1}, & n/2<i\leqslant n\end{cases} \tag{4}$$

式中 n 表示平滑深度，协议取为 8，$r(i)$ 为最近第 i 轮利用(3)式获得的速率样本值；$w(i)$ 为加权系数。接收端 j 根据该层速率更新其期望速率，即 $X_j=E\_Rate$。

1.3.4　层加入和退出

当某个接收端计算的期望速率值增大，以致超过当前层 i 的最大累计速率 B_i 时，该接收端应加入 $i+1$ 层。此时接收端应向 $i+1$ 层发送 IGMP 加入(Join)消息。当估计的期望速率在相邻两层的交界处附近上下变化时，有可能导致接收端频繁地加入和退出某一层，从而引起吞吐量的振荡。为尽量避免这一情况，协议采用了保守的方法，规定只有当计算速率在 $(dB_i, B_{i+1}]$ 范围内时，接收端才能加入高一层，这里 d 是阻尼因子，本文取 $d=1.1$。当某个接收端计算的期望速率值减小，以致低于当前层 i 的最小速率 B_{i-1} 时，应发送 IGMP 离开(Leave)消息，退出当前层。

2　性能仿真

笔者利用 NS2 仿真平台，从协议的有效性、TCP 友好性、

响应性、协议内公平性等方面对 FLMCC 的性能进行仿真分析。实验中 TCP 采用 TCP Reno，各结点采用 RED 队列方式，最大队长设为带宽迟延积的 2 倍，最小阈值 minthresh 和最大阈值 maxthresh 分别设为最大队长的 10%和 40%，gentle 参数设置为 On。基层最大速率 B_0 为 0.5Mbps，基层发送速率为 0.25Mbps，其他层发送速率为 0.5Mbps。

2.1 有效性、TCP 友好性

选择如图 4 所示的具有 3 个瓶颈链路的拓扑。各链路传播迟延均为 4ms，各流的 RTT 为 32ms。FLMCC 组播会话包含 3 个接收端 F_1、F_2、F_3，各自分别与 4 个 TCP 流共享同一瓶颈链路。3 个瓶颈链路带宽分别为 1.25Mbps、3.75Mbps、6.25Mbps，因此理论上各瓶颈支路上的公平带宽分别为 0.25Mbps、0.75Mbps 和 1.25Mbps。各流在 0～2s 内先后启动。图 5 为仿真结果。图 5(a)给出 FLMCC 各接收端的吞吐量(累计接收速率)，各接收端最初加入基层，F_2 在 6.9s 时加入第 1 层，F_3 在 22.8s 时加入第 2 层。F_1 始终只订购基层，因此获得的吞吐量和基层的发送速率一致，一直稳定在 0.25Mbps。F_2 和 F_3 获得的吞吐量在整个仿真期间出现了少许波动，但总体上稳定在其公平带宽上。经历不同网络状况的接收端通过加入不同的层，获得了不同的吞吐量，体现了分层组播提供多速率服务的有效性。其次，图 5(b,c,d)分别比较了各接收端与同一瓶颈支路上的任一条 TCP 流的吞吐量，各接收端都能在各自瓶颈链路上与 TCP 流公平地分享瓶颈带宽，体现了 FLMCC 的 TCP 友好性。由于采用了平滑的窗口调整算法以及固定的分层速率，各接收端的吞吐量相对于 TCP 要平滑得多。

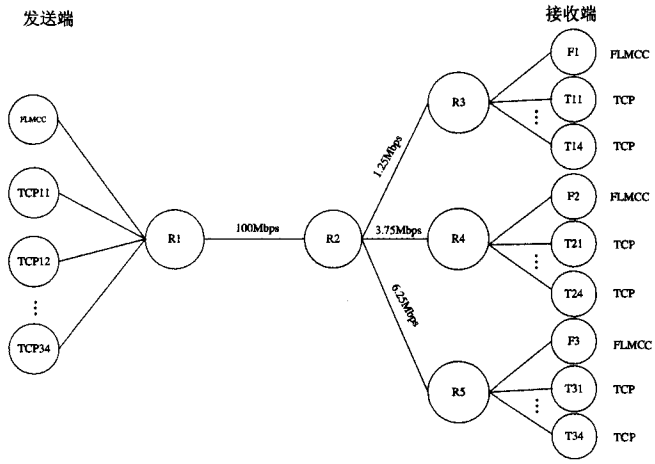
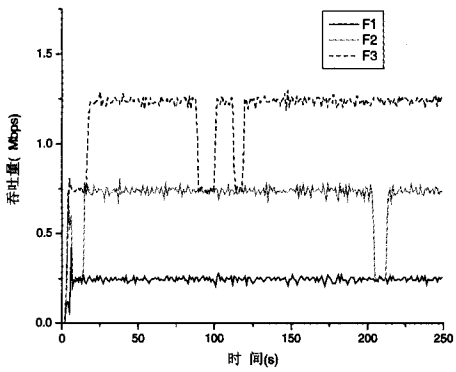
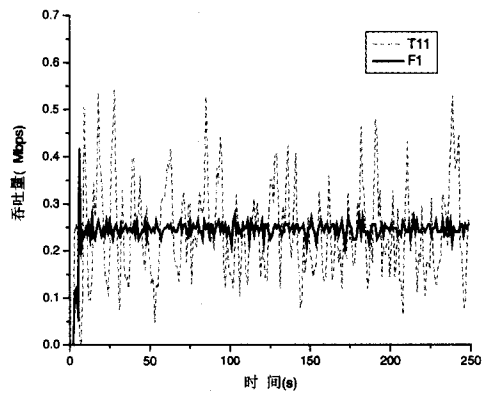


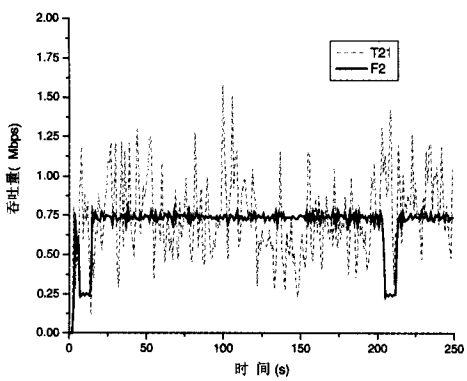
图 4 有 3 个瓶颈链路的仿真拓扑



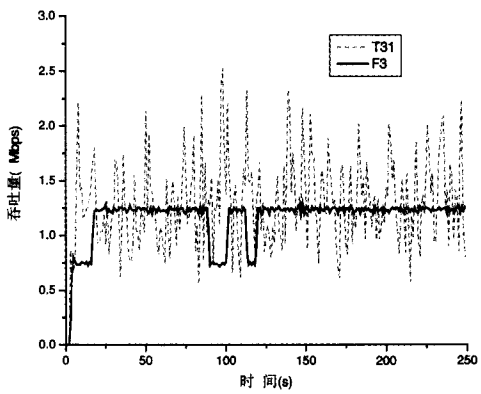
(a) 各 FLMCC 接收端



(b) T11 和 F1



(c) T21 和 F2

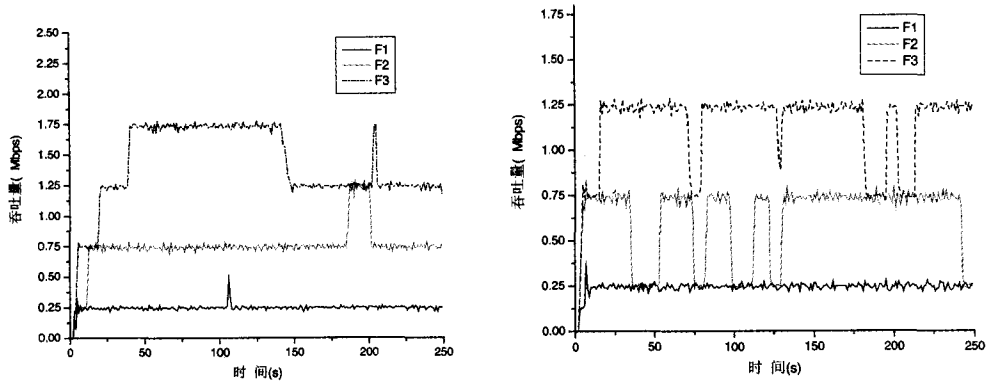


(d) T31 和 F3

图 5 FLMCC 接收端及竞争 TCP 流的吞吐量

图 6 分别给出了另外两组瓶颈链路带宽下各接收端的吞吐量。仿真拓扑与图 5 情况完全相同。图 6(a)为 3 个瓶颈链路带宽分别为 1.5Mbps、4Mbps 和 7Mbps 时 3 个组播接收端的吞吐量(公平带宽分别为 0.3、0.8、1.4Mbps)，而图 6(b)中各瓶颈链路的带宽分别为 1Mbps、3Mbps 和 6Mbps(公平带宽分别为 0.2、0.6、1.2Mbps)。两种情况下各接收端的理论公平带宽与相应各层的发送速率并不吻合。从图中可以看到，此时各接收端总体上仍然获得了有效的吞吐量，其实际获得的吞吐量接近于其应当获得的公平值，并保持了较好的 TCP 友好性(图中省略了 TCP 的吞吐量)。另外，吞吐量围绕其公平值呈现上下波动，但波动范围有限，维持在相邻上下层的范围内。这是协议适应网络状况的动态变化，采用基于

窗口的机制进行调整的正常结果。



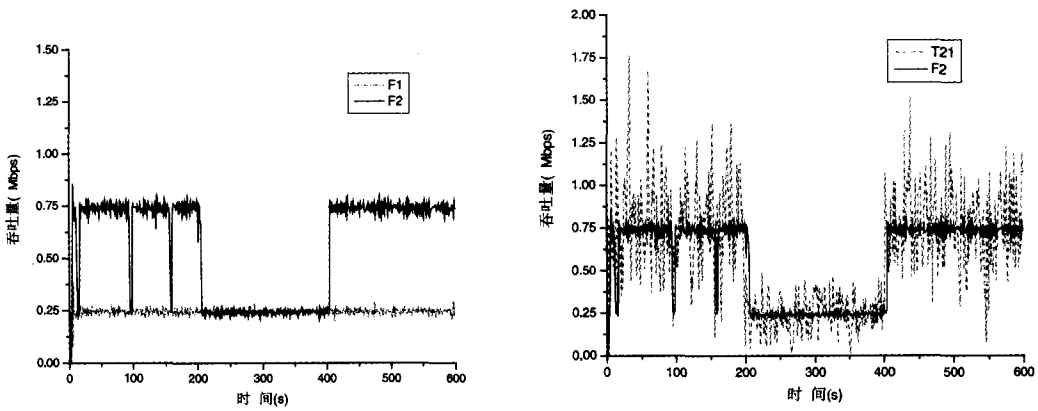
(a) 带宽: 1.5M、4M 和 7Mbps　　(b) 带宽: 1M、3M 和 6Mbps

图 6　两组不同瓶颈带宽下各接收端的吞吐量

2.2　响应性

响应性实验用于考察协议在网络拥塞状况发生变化时的反应速度。实验仍采用类似图 4 的拓扑，仿真条件有所变化：组播会话仅包含 F1 和 F2 两个接收端，以及 R2-R3 和 R2-R4 两条瓶颈链路；F1，F2 最初分别与 4 个 TCP 流竞争瓶颈链路；200s 时，F2 所在的瓶颈链路上加入 10 条新的 TCP 流，引起该支路业务量的变化，15 条流共享 3. 75Mbps 瓶颈带宽，F2 的公平速率从 0. 75Mbps 下降到约 0. 25Mbps；400s 时，后加入的 10 条 TCP 流同时退出。仿真运行 600s。图 7(a)显示 2 个 FLMCC 接收端的吞吐量。F1 的吞吐量仿真期间一直保持在 0. 25Mbps。200s 前，F2 的吞吐量基本维持在 0. 75Mbps，偶有波动。200s 后，由于新业务流的加入，F2 很快做出了响应，其吞吐量迅速下降，在 205. 8s 时，F2 退出第 1 层，之后以 0. 25Mbps 的速率接收数据包。400s 时，后加入的 TCP 流退出，F2 在 405. 1s 重新加入第 1 层，其吞吐量迅速增大并恢复到 200s 前的取值。图 7(b)显示了 F2 与同一瓶颈链路上的一条 TCP 流的吞吐量，在整个仿真过程中，F2 在体现响应性的同时，仍保持了良好的 TCP 友好性。

以上仿真说明，FLMCC 具有良好的响应性，可以对网络状况的变化做出及时的反应。



(a) 各接收端的吞吐量　　(b) T21 和 F2 的吞吐量

图 7　FLMCC 的响应性

2.3　协议内公平性

协议内公平性是指执行同一协议的多个流间的公平性，其评价参数通常采用公平性指数(Fairness Index，FI)[10]。公平性指数定义为 $f_i=(\sum_{i=0}^{N-1}T(i))^2/(N\cdot\sum_{i=0}^{N-1}T(i)^2)$，其中 $T(i)$为第 i 条流的吞吐量。f_i 取值在$[1/N,1]$之间，f_i 越大，各流间的公平性越好。

仿真拓扑采用哑铃型结构，如图 8 所示。仿真中分别选择不同数目的 FLMCC 组播流，瓶颈链路 R1-R2 的带宽根据流数目不同在 10～24Mbps 之间变化，测量不同情况下各流的吞吐量和瓶颈链路利用率。表 1 给出了仿真结果。可以看出，不同流数目情况下，公平性指数均大于 0. 99，说明协议具备很好的协议内公平性；瓶颈链路利用率均大于 0. 98，非常接近于 1，表明协议可以有效地利用链路带宽。

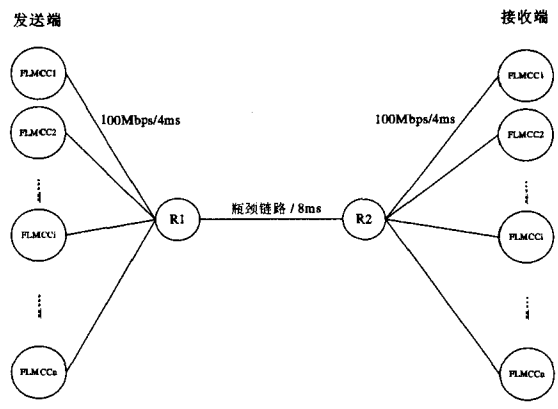


图 8　协议内公平性仿真拓扑

结论 本文提出了一种新的固定速率分层组播拥塞控制算法FLMCC。组播会话中的每层按照固定速率发送数据包。各接收端采用窗口机制，利用TCP友好而平滑的GAIMD算法完成窗口调整，并根据当前窗口值计算期望速率；根据期望速率订购数目不等的层，从而获得不同的吞吐量。为测量RTT，采用了一种精确测量和粗略测量相结合的策略；精确测量时，为避免大量反馈产生反馈内爆，采用了随机定时器策略。仿真表明，算法具有良好的TCP友好性、响应性和协议内公平性，链路利用率高。算法可为异构的组播成员有效提供多速率服务，增强了组播拥塞控制的可扩展性，并且实现简单。

表1 多个FLMCC组播流的公平性指数及链路利用率

参数	组播流数目									
	2	4	6	8	10	12	14	16	18	20
公平性指数	0.9972	0.9986	0.9970	0.9984	0.9976	0.9982	0.9973	0.9991	0.9971	0.9986
链路利用率	0.9818	0.9857	0.9833	0.9923	0.9953	0.9915	0.9938	0.9974	0.9960	0.9973

参考文献

1 Nonnenmacher J, Biersack E W. Scalable feedback for large groups. IEEE/ACM Trans on Networking, 1999,7(3): 375～386

2 Bhattacharyya S, Towsley D, Kurose J. The Loss Path Multiplicity Problem in Multicast Congestion Control. In:Doshi B, ed. Proc. of IEEE INFOCOM. New York, USA, 1999

3 Kwon G, Byers J W. Smooth Multirate Multicast Congestion Control. In: Proc. of IEEE INFOCOM 2003, San Francisco, USA, 2003

4 Byers J, Horn G, Luby M, et al. FLID-DL: Congestion Control for Layered Multicast. IEEE JSAC Special Issue on Network Support for Multicast Communication, 2002, 20(8): 1558～1570

5 McCanne S, Jacobson V, Vetterli M. Receiver-Driven Layered Multicast. In:Proc. of ACM SIGCOMM '96, 1996

6 Vicisano L, Rizzo L, Crowcroft J. TCP-like Congestion Control for Layered Multicast Data Transfer. In: Proc. of IEEE INFOCOM '98, 1998

7 Legout A, Biersack E W. PLM: Fast convergence for cumulative layered multicast transmission schemes. In: Proc. of ACM Sigmetrics, 2000

8 Yang R Y, Simon S L. General AIMD Congestion Control. In: Proc. of the 8th International Conference on Network Protocols. Osaka, Japan, 2000

9 Widmer J, Handley M. Extending Equation-based Congestion Control to Multicast Applications. In:Floyd S,ed. Proc of ACM SIGCOMM. San Diego, USA. 2001

10 Chiu D-M, Jain R. Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks. Computer Networks and ISDN Systems, 1989, 17(1): 1～14

(上接第4页)

26 Watts D J, Dodds P S, Newman M E J. Identity and search in social networks. Science, 2002, 296: 1302～1305

27 Karypis G, Kumar V. Multilevel Algorithms for Multi-Constraint Graph Partitioning. In: Proceedings of the 1998 ACM/IEEE Conference on Supercomputing(CDROM), San Jose, CA, November, 1998. 7～13

28 Fiedler M. Algebraic connectivity of graphs. Czech Math J, 1973, 23: 298～305

29 Pothen A, Simon H, Liou K P. Partitioning sparse matrices with eigenvectors of graphs. SIAM J Matrix Anal Appl, 1990, 11: 430～452

30 Mohar B. The Laplacian spectrum of graphs. Graph Theory, Combinatorics, and Applications, 1991, 2: 871～898

31 Kernighan W, Lin S. An efficient heuristic procedure for partitioning graphs. Bell System Technical Journal, 1970, 49: 291～307

32 Scott J. Social Network Analysis: A Handbook. 2nd edition. Sage, London, 2000

33 Golub G H, Van Loan C F. Matrix computations. Baltimore, MD: Johns Hopkins University Press, 1989

34 Burt R S. Position in networks. Social Forces, 1976, 55: 93～122

35 Scott J. Social Network Analysis: A handbook. 2nd edition. London: Sage Publication,2000

36 Breiger R L, Boorman S A, Arabie P. An al-gorithm for clustering relations data with applications to social network analysis and comparison with multidi-mensional scaling. Journal of Mathematical Psychology, 1975, 12: 328～383

37 Amaral LA N, Scala A, Barthelemy M, et al. Classes of small-world networks. Proc Natl Acad Sci USA, 2000, 97: 11149～11152

38 Marchiori M, Latora V. Harmony in the small-world. Physica A, 2000, 285: 539～546

39 Newman M E J, Girvan M. Finding and evaluating community structure in networks. Phys Rev, 2004, E 69: 026113

40 Pallal G, Derenyi I, Farkasl I, et al. Uncovering the overlapping community structure of complex networks in nature and society. Nature,2005, 435: 814～817

41 Newman M E J. Scientific collaboration networks: II. Shortest paths, weighted networks, and centrality, Phys Rev, 2001, E 64:016132

42 Girvan M, Newman M E J. Community structure in social and biological networks. Proc Natl Acad Sci USA, 2002, 99: 7821～7826

43 Holme P. Subnetwork hierarchies of biochemical pathways Bioinformatics. 2003, 19: 532～538

44 Wilkinson DM, Huberman BA. A Method for Finding Communities of Related Genes. Proc Natl Acad Sci USA, 2004,101 (Suppl 1):5241～5248

45 Gleiser P, Danon L. Community structure in jazz. 2003, 0307434

46 Newman M E J. Mixing patterns in networks. Phys Rev, 2003, E 67:026～126

47 Radicchi F, Castellano C, Cecconi F, et al. Defining and identifying communities in networks. Proc Natl Acad Sci USA ,2004, 101: 2658～2663

48 Derenyi I, Palla G . Clique Percolation in Random Networks. Phy Rev Lett PRL, 2005, 94: 160～202

49 Milo R, et al. Network Motifs: Simple Building Blocks of Complex Network. Science, 2002, 298:824

50 Shen-Orr S, Milo R, Mangan S, et al. Network motifs in the transcriptional regulation network of Escherichia coli. Nature Genet, 2002, 31: 64

51 Everett M G, Borgatti S P. Analyzing clique overlap. Connections, 1998, 21: 49～61

52 Shiffrin R M, Börner K. Mapping knowledge domains. Proc Natl Acad Sci USA, 2004, 101: 5183～5185