有线无线网络 TCP 友好流媒体拥塞控制机制①

Wire and Wireless Network TCP Friendly Streaming Media Congestion Control Mechanism

何建新　杨格兰　（湖南城市学院 计算机科学系 湖南益阳 413000）

摘　要： 在 TCP 友好拥塞控制方法比较基础上，为了实现流媒体平滑传输以及保证 TCP 流友好性，提出了 TCP 友好速率控制算法 WTFCC。该算法能够在接收端区分网络拥塞丢包和链路错误随机丢包，准确判断网络拥塞状况；结合接收端缓存区占用程度，自适应实施多级速率调节。仿真实验结果表明，该机制对 TCP 流是友好的，并且保障了媒体播放质量，在有线无线混和网络中具有很好的性能。

关键词： TCP 友好性　流媒体　拥塞控制　服务质量（Qos）WTFCC

随着无线网络发展，在无线移动终端上对基于语音和视频的流媒体服务有很大需求。流媒体数据传输具有高网络带宽、低传输延迟、低延迟抖动及对传输可靠性要求相对较低的特点，给网络传输带来了巨大挑战，流媒体网络传输已成为网络研究的热点之一。流媒体应用对 QoS 有一定要求，同时还要保持对因特网中 TCP 流的友好性。基于 UDP 协议的流媒体传输由于没有拥塞控制机制，对分组超时或丢弃不敏感，相对 TCP 流在带宽资源竞争中占优势，甚至导致 TCP 流“饿死”。无线网络中链路随机出错率高，使 TCP 流的传输性能进一步降低，因此实现流媒体平滑传输以及保证 TCP 流友好性已成为流媒体传输控制中需要考虑的重要因素。

1　TCP 友好拥塞控制方法

1.1　基于速率控制方法

根据网络丢包率来判断网络拥塞程度，作为调节网络传输速率的依据。该类方法的主要代表有 TFRC。

TFRC 是基于速率控制的 TCP 友好协议[1]，根据式(1)所示流量方程实时调整网络发送速率 T：

$$T=\frac{S}{RTT\sqrt{\frac{2P}{3}}+t_{RTO}(3\sqrt{\frac{3p}{8}})p(1+32p^2)} \quad (1)$$

其中 S 表示数据包大小，RTT 为往返时延，P 为网络稳定状态下的丢包率，t_{RTO}为重传计时器值。基于此式速率控制方案的带宽占用显然能够保证对 TCP 友好性，满足流媒体服务对速率变化的平滑要求，不过其对单个丢包事件不敏感，不利于拥塞的解除。

1.2　基于窗口控制方法

该类方法的主要代表有 TEAR。采用 TCP 协议窗口加性增加乘性减少（AIMD）方法调节发送速率，其 cwnd 变化规律如式(2)、(3)所示。

$$AI: w_{t+1}=w_t+\alpha w_t^k,\ \alpha>0 \quad (2)$$

$$MD: w_{t+1}=w_t-\beta w_t^l,\ 0<\beta<1 \quad (3)$$

其中 w_{t+1}，w_t 分别表示间隔 RTT 时间 cwnd 大小；α，β，k 和 l 是用来调整速度的参数。TCP 中 AIMD 参数组为 $\alpha=1$，$\beta=0.5$，$k=0$，$l=1$。虽然基于窗口控制机制保证了 TCP 友好性，但遭遇分组丢弃时将窗口减半，导致速率突变而严重影响流媒体 QoS。

TEAR（TCP Emulation at Receivers）在接收端模拟 TCP 协议[2]，接收端并不把探测到的拥塞信号反馈回发送端，而是自己马上利用这些信号来修改发送速率并报告发送方，适合于组播和在不对称网络上传输实时多媒体。

上述算法都没有考虑有线/无线环境对控制算法

① 基金项目：湖南省教育厅科研基金项目（07c665）；湖南城市学院校级科研项目（07c004）

性能的影响。本文将结合上述算法优点,提出一种新的适用于有线/无线混合网络 TCP 友好和动态 QoS 调整的流媒体拥塞控制算法 WTFCC(Wireless TCP – friendly congestion control)。[3]

2 无线网络中 TCP 友好速率控制算法 WTFCC

WTFCC 算法在带宽资源富余情况下,能保证 TCP 友好性,同时有效改善流媒体业务 QoS;在带宽资源紧张时,能够自适应实现 TCP 友好性和流媒体业务 QoS 折中。其基本思想包括以下几个方面:

2.1 区分拥塞与误码丢包

在无线网络中,丢包并不完全由网络拥塞产生,甚至主要是由链路随机比特出错导致。有效区分拥塞/误码丢包准确获得网络状态是实施拥塞控制的前提。有线无线混合网络链路丢包率总和 $P = P_c + P_w$,其中 P_c 为拥塞丢包率,P_w 为无线链路误码率。正确区分拥塞丢包和无线链路误码丢包,将正确的网络拥塞状态反馈给源节点成为解决无线网络流媒体传输的关键。

采用从接收端判断,假定第 k 个分组从源节点发送的时间为 t_{ts}^k,以分组头时间戳方式随数据一同发送到目的节点,到达时间为 t_r^k,考虑发送端、接收端时钟不同步,时钟偏差为 δt,则单向延迟有 $t_d^k = t_r^k - t_{ts}^k + \delta_t$,对于第 k+1 个分组,有 $t_d^{k+1} = t_r^{k+1} - t_{ts}^{k+1} + \delta_t$,第 k,k+1 号分组的单向传输延时之差为 Δt,有

$\Delta_t^k = t_r^{k+1} - t_r^k = (t_r^{k+1} - t_r^k) - (t_{ts}^{k+1} - t_{ts}^k)$,且有

$$\eta_k = \frac{\Delta_t^{k+1}}{\Delta_t^k} = \frac{(t_r^{k+2} - t_r^{k+1}) - (t_{ts}^{t+2} - t_{ts}^{k+1})}{(t_r^{k+1} - t_r^k) - (t_{ts}^{k+1} - t_{ts}^k)}$$

当 $\Delta_t^k > 0$ 或者 $\eta_k > 1$,表明网络趋向于拥塞。当网络出现分组丢弃时,若 $\eta_k > 1$,可以推断网络发生拥塞应降低网络发送速率,否则分组丢弃是由无线链路错误导致。

引入拥塞指示因子 ς 并通过滤波器来平滑网络测量所带来的噪音,令 $\varsigma_k = \alpha\varsigma_{k-1} + (1-\alpha)\eta_k$,其中 ς_{k-1} 为上次测量值,η_k 为当前分组单向延时比值,α($0 < \alpha < 1$)为获取权值。

2.2 实现 TCP 友好性与流媒体 QoS 折中

流媒体拥塞控制必须保证对 TCP 友好性,并尽可能保证流媒体业务 QoS。流媒体回放要求播放速率变化非常平滑,在网络传输系统中,不论是基于速率还是基于窗口的控制机制所提供的媒体流在传输速率上均无法达到此要求。为了获得平滑的播放质量在接收端将数据缓存,然后以流媒体业务要求的速率从缓存区读取数据,从而保证其 QoS。WTFCC 算法在正确区分拥塞/误码丢包基础上,结合缓存区资源占用情况,设计了一种动态速率反馈策略。缓存区结构如图 1 所示。[4]

T_{in} 为缓存区分组输入速率,T_{out} 为缓存区流媒体播放输出速率,缓存区容量为 Q_b,Q_{cur} 为缓存区当前队列长度测量值,缓存区高低门限阈值 Q_{max}、Q_{min} 将缓存区分成三段:

(1) $Q_b \sim Q_{max}$ 为 higher 段,队列长度接近缓存区容量,容易导致缓存区溢出,可以适当降低输入速率,避免接收端分组丢失,影响流媒体播放质量。

(2) $Q_{max} \sim Q_{min}$ 为 normal 段,缓存区队列长度适中,既能保证流媒体平滑播放,同时也不容易产生缓存区溢出丢包,定义最优分组队列长度 Q_{opt} 为 $(Q_{max} + Q_{min})/2$。

(3) $Q_{min} \sim 0$ 为 lower 段,队列长度不够长,当网络遭遇暂时拥塞,缓存数据往往不能满足流媒体平滑播放要求,缓存区空将导致播放出现停顿。

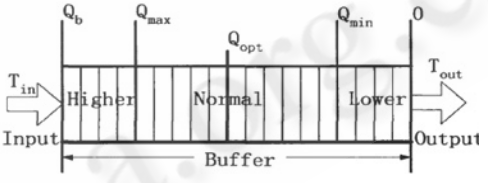

图 1　接收端缓存区结构

保证 TCP 友好性和流媒体 QoS 必须考虑网络拥塞状况兼顾接收端缓存区占用状况,对三个区段分别采用不同的速率调节。针对丢包情况和接收端队列长度,发送速率调节方案如下:

(1) 流媒体业务允许少量丢包,接收端探测到丢包率小于设定值 p_{min},Q_{cur} 位于 normal 段,说明系统正处于平稳运行状态,当连接刚启动时,缓存区容量必须达到 Q_{min} 才开始播放。

(2) 当丢包率小于 p_{min},Q_{cur} 位于 lower 段,若 $T_{in} > T_{out}$,Q_{cur} 将不断增加,而进入(1),保持发送速率不变;若 $T_{in} < T_{out}$,缓存区很快将会清空导致播放停顿,应增加发送速率。

(3) 当丢包率小于 p_{min},Q_{cur} 位于 higher 段,若 $T_{in} >$

T_{out}，表明网络带宽富余，可以适当增加 T_{out} 提高播放质量，若 $T_{in} < T_{out}$，不做任何操作 Q_{cur} 将回落到 normal 段。

(4) 当丢包率大于 p_{min}，网络处于拥塞状态，若 Q_{cur} 位于 lower 段，此时应适当降低 T_{in} 和 T_{out}，以避免播放停顿，不过牺牲了播放质量。这样可以保证流媒体业务对 TCP 友好性，以牺牲部分 QoS 为代价。

(5) 当丢包率大于 p_{min}，若 Q_{cur} 位于 higher、normal 段时，此时应适当降低 T_{in} 但不降低 T_{out}，这样可以对拥塞做出与其他 TCP 流相同的操作维持 TCP 友好性，同时也保障了播放质量，等待网络退出拥塞。

综上所述速率调节策略如式(4)所示：

$$T_{in}(t+1)=\begin{cases} Tin(t)+\alpha, \text{ if } p(t)\le Pmin \,\&\, Qcur \le Qmin \\ Tin(t), \text{ if } p(t)\le Pmin \,\&\, Qmax < Qcur \le Qb \,\&\, Tin(t) \le Tout \\ Tin(t), Tout = (Tout + Tin(t))/2, \text{ if } p(t)\le Pmin \,\&\, \\ \qquad Qmax < Qcur \le Qb \,\&\, Tin(t) > Tout \\ Tin(t) - \beta Tin(t), Tout = Tin(t), \text{if } p(t) > Pmin \,\&\, Qcur < Qmin \\ Tin(t) - \beta Tin(t), \text{if } p(t) > Pmin \,\&\, Qmin < Qcur < Qb \end{cases} \quad (4)$$

其中发送端的速率为：

$$T(t) = \frac{T_{in}(t)}{1-p} = \frac{T_{in}(t)}{1-p_c-p_w} \quad (5)$$

基于上述速率调整方案，WTFCC 算法伪代码描述如下：

初始化 T_{out}、p_{min}、Q_{max}、Q_{min}、Q_b 等参数；

While（数据包传输未完成）

{测量 P_c、P_w、η_k 判断网络拥塞丢包与无线链路误码丢包；

if 数据包 P_i 被接受 then 更新 ξ_k、P_c、Q_b；

if 速率控制计时器超时 then

接收端根据式(4)调节速率 T(t)，采用 TCP 向源端反馈速率调节控制信息，保证可靠传输；

if 控制信息被源端接受 then

更新速率控制计时器；

}

3 实验仿真与分析

通过仿真实验验证算法有效性。在 ns－2.28 的 TCP 部分加入了 WTFCC 模块，重新编译仿真软件。N 台源主机 S_i (i=1,2...N) 与路由器 R1 相连；R1 与无线接入点 R2 连接；R2 通过无线链路与移动主机 H_i 连接，网络拓扑结构、链路带宽与单向延迟如图 2 所示。无线链路产生随机丢包通过调用 NS2 中随机错误模块实现，分组大小为 1000 字节。

比较不同无线链路丢包率环境下，基于不同拥塞控制算法 TCP NewReno，TFRC 和 WTFCC 吞吐量和端到端丢包率比较，将链路丢包率从 0.01 ~0.1 变化，间隔为 0.01 秒，从 S_i 到 H_i 的连接数分别为 2，32，使网络处于轻载和拥塞状态，连接持续时间为 1000 秒。

图 2　网络仿真模型

实验结果如图 3、图 4 所示当链路丢包率为零时，吞吐量基本能够达到链路容量，随着丢包率增加吞吐量均下降。当 n=2 时，TCP 和 TFRC 吞吐量显著下降，最后维持在一个非常低的水平，而 WTFCC 由于接收端能够区分链路错误和网络拥塞，网络吞吐量能够维持在一个较高水平；当 n=32 时，网络处于拥塞状态，在链路误码率比较低的情况下，网络吞吐量基本接近网络瓶颈链路容量，随着链路误码率增加，TCP 和 TFRC 发送速率不断降低，网络吞吐量不断下降，而 WTFCC 表现对链路错误的免疫力，吞吐量一直维持在网络容量附近。

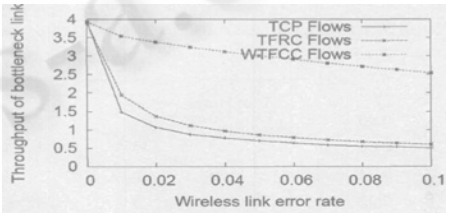

图 3　吞吐量与链路丢包率关系(n=2)

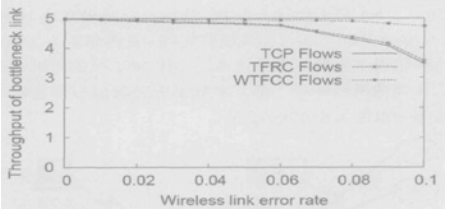

图 4　吞吐量与链路丢包率关系(n=32)

4 结论

深入探讨了流媒体传输对 TCP 友好性的影响，指出一些主要流媒体传输方法在无线网络中存在的问题，改进 RTP 和 TFRC 算法，提出了一种适用于无线网络流媒体传输算法。仿真实验表明网络处于轻载和拥塞状态，不同无线链路丢包率情况下该算法较好地实现了流媒体平滑传输、维持了 TCP 友好性，同时也保障了媒体播放质量。将 TCP 友好性流量控制算法扩展到流媒体的组播传输中具有重要的意义。

参考文献

1 柳建国. 无线网络中多媒体传输拥塞控制机制的研究. 计算机工程与应用，2007，43(7)：160－162.

2 Rhee I. TEAR: TCP emulation at receivers flow control for multimedia streaming. North Carolina State University, 2000. 186－190.

3 胡严，张光昭，张国清. 多媒体流在 Internet 上传输的一种 TCP－Friendly 拥塞控制机制. 计算机学报，2003，4：201－205.

4 Ishibashi Y, Tasaka S. A synchronization mechanism for continuous media in multimedia communications. In Proc IEEE Infocom95, 1995. 4. 1010－1019.