

# EHSTCP: 改进的高速 TCP 算法

龙承念<sup>1)</sup> 杨会龙<sup>1)</sup> 李 欣<sup>1)</sup> 关新平<sup>2)</sup>

<sup>1)</sup>(燕山大学电气工程学院网络控制与生物信息研究中心 河北 秦皇岛 066004)

<sup>2)</sup>(上海交通大学电子信息与电气工程学院自动化系 上海 200240)

**摘 要** TCP 在高带宽时延积网络中不能获得良好的性能,主要表现为低的吞吐量和大的窗口震荡. HSTCP 算法解决了传统 TCP 算法在高带宽时延积网络下的性能瓶颈,但 HSTCP 在拥塞点时会产生大量的数据包丢失,同时当队列管理为去尾算法时,存在着严重的 RTT 不公平性问题. 针对 HSTCP 算法的性能缺陷,该文提出一种在拥塞避免阶段进行拥塞避免模式切换的改进算法,称为 EHSTCP. 基于拥塞窗口历史值的端到端可用带宽预测方法,利用拥塞窗口历史信息来判断拥塞避免切换点. 同时引入 RTT 公平因子,消除了 HSTCP 的 RTT 不公平性问题. NS2 仿真实验验证了算法的有效性.

**关键词** HSTCP; 拥塞控制; TCP; 包丢失; RTT 不公平性

**中图法分类号** TP393

## EHSTCP: Enhanced TCP in High-Speed Networks

LONG Cheng-Nian<sup>1)</sup> YANG Hui-Long<sup>1)</sup> LI Xin<sup>1)</sup> GUAN Xin-Ping<sup>2)</sup>

<sup>1)</sup>(Centre for Networking Control and Bioinformatics (CNCB), Institute of Electrical Engineering,  
Yanshan University, Qinhuangdao, Hebei 066004)

<sup>2)</sup>(Department of Automation, School of Electronic Information and Electrical Engineering,  
Shanghai Jiaotong University, Shanghai 200240)

**Abstract** TCP doesn't perform well in high bandwidth-delay product networks, and its main limitations are low throughput and large oscillation of congestion window. HSTCP solves the primary limitations of regular TCP, but there exists large number of packet loss upon congestion epoch and serious RTT unfairness with drop-tail gateway. To overcome the above deficiency, this paper proposes an improved algorithm, called EHSTCP, which switches between two congestion avoidance modes in congestion avoidance phase. To locate the switch point, this paper presents a new prediction method for end-to-end available bandwidth using history congestion window dynamics. Simultaneously, it introduces the RTT fairness factor to eliminate the serious RTT unfairness in HSTCP. NS2 simulation results validate the effectiveness of the proposed scheme.

**Keywords** HSTCP; congestion control; TCP; packet loss; RTT unfairness

## 1 引 言

随着因特网的迅猛发展,网络规模、用户和应用

急剧增加,随之而来的是日益突出的网络拥塞问题. 拥塞控制是确保因特网鲁棒性的关键因素,也是各种管理控制机制和应用的基础,因此成为当前国内外计算机网络和控制理论交叉领域的一个热点课

题<sup>[1]</sup>. 下一代因特网具有高带宽时延积特性, 给现有的研究提出了新的挑战. 该问题主要体现在如下两个方面: 第一, 现有的 TCP/AQM 协议框架在包层次上的算法性能降低. 主要原因在于高的带宽时延积会带来非常大的 TCP 拥塞窗口 (数量级为  $10^4 \sim 10^5$ ). 但传统的 TCP 拥塞控制机制主要采用加式递增乘式递减 (AIMD) 形式的拥塞避免机制<sup>[2-5]</sup>, 在 AIMD 调整过程中, 如此大的拥塞窗口必然会导致“过缓”或“过激”的效应. 如加式提升, 每一往返时延 (RTT) 仅增加一个包, 这使得用户达到稳态吞吐量的时间过长. 若可用带宽为 1 Gbps, 一个包长为 1500 Bytes, RTT 为 200 ms 的 TCP 链接需要 27 min 才能从包丢失的状态重新获得可用带宽. 当发生包丢失和 ECN 标记时, 将当前拥塞窗口 (cwnd) 减半, 这种行为又显得过激, 带来大的拥塞窗口振荡, 导致低的网络吞吐量. 第二, 在流量层次上现有算法的鲁棒性和稳定性难以保证. 保持大的平衡窗口需要极小的包丢失率 (数量级为  $10^{-8} \sim 10^{-10}$ ), 它们的关系如式 (1) 所示.

$$w_0 = \frac{\alpha}{\sqrt{p_0}} \quad (1)$$

其中,  $w_0$  为用户的平衡窗口大小,  $p_0$  为端到端的包丢失概率的平衡值,  $\alpha$  为比例因子. 很显然随着带宽时延积的增加, 在实际网络中该平衡状态将难以保证.

当前国际上对于高带宽时延积网络拥塞控制算法的研究主要分为 3 类, 一类基于包丢失机制, 主要算法包括 HSTCP<sup>①</sup>、STCP<sup>[6]</sup>、BIC<sup>[7]</sup> 和 L-TCP<sup>[8]</sup>; 第二类基于端到端队列时延变化, 如 FAST TCP<sup>[9]</sup>; 第三类基于多比特的全信息反馈控制机制, 主要算法有 XCP<sup>[10]</sup>. 还有一些其它的高速 TCP 算法, 如 H-TCP<sup>[11]</sup>、SQRT<sup>[12]</sup>、CUBIC<sup>[13]</sup> 和 AntiECN<sup>[14]</sup>. HSTCP 算法在高带宽网络环境中解决了标准 TCP 算法的主要性能缺陷, 并且实现简单, 便于在实际网络中推广, 该方案已经被 IETF 所采用.

由于 HSTCP 在拥塞避免阶段采用非线性的增长模式, 当产生突发流时, 在拥塞点会造成大量的数据包丢失<sup>①</sup>. 文献 [15] 提出采用 Block-pacing 机制来减轻这一问题, Block 机制将窗口的增长部分划分为两个阶段: 快速增长阶段和线性增长阶段, 即当 HSTCP 流没有获得可用带宽时采用快速的增长方式, 当接近可用带宽时转换为标准 TCP 的增长方式, 这样可以避免过多的包丢失. 文献 [17] 同样采用了双模态的窗口增长方式. 该文献指出, 采用两种模态相互转换的增长方式可以降低包丢失事件的发生

频率, 从而降低包丢失率. 该机制运用于与传统 TCP 共同存在的混合网络中能提高传统 TCP 的吞吐量, 改善 HSTCP 的 TCP 友好性, 类似的工作还有文献 [18]. 但上述方法均是根据队列时延的变化来预测网络拥塞的, Biaz 在文献 [19] 指出运用往返时延的测量信息不能充分地反映路由器队列长度的变化, 不能被用来作为拥塞发生的指示, 因此运用队列时延难以准确地判断窗口增长模式的切换点, 类似的工作还有文献 [20] 等. 在高速网络环境下较小的缓冲区更加适合市场的需求, 因此要求路由器缓冲区要小于带宽时延积, 这使得运用队列时延变化进行可用带宽估计更难以实现. 为解决这一难题, 本文提出了一种新的切换点判断方法. 该方法基于窗口历史动态来预测网络的可用带宽. 根据估计的端到端可用带宽, 我们可以设定一个适当的切换阈值. 当网络的瓶颈链路利用率超过切换阈值时, 拥塞避免阶段从高速的增长模式切换为 TCP 的低速增长模式. 在去尾算法下, HSTCP 存在严重的 RTT 不公平性, 即长时延的 HSTCP 流与短时延的 HSTCP 流相互竞争时带宽分配极其不公平<sup>[7]</sup>. 文献 [15] 通过选取 RTT 公平因子  $C * RTT$ , 使窗口的增加量与往返时延无关, 但网络吞吐量与 RTT 成反比例关系, 因此没有完全解决 HSTCP 存在的 RTT 不公平性问题. 文献 [16] 选取的 RTT 公平因子为  $(RTT/t_p)^2$ , 消除了 RTT 对吞吐量的影响. 本文仍采用引入 RTT 公平因子方法解决 HSTCP 算法的 RTT 不公平性, 不同之处是我们把 RTT 公平因子扩展到了更为一般的形式, 选取 RTT 公平因子为  $(RTT/t_p)^\alpha$ , 根据不同算法的响应函数, 合理地选取参数  $\alpha$  能够消除 RTT 对吞吐量性能的影响. 该方法不但适应于 HSTCP 算法, 同样适应于其它的高速 TCP 算法.

## 2 HSTCP 算法性能分析和相关的改进工作

### 2.1 HSTCP

Floyd 针对 TCP 算法在高带宽时延积网络环境下存在的扩展性问题和响应缓慢的缺点, 提出了一种基于包丢失的 AIMD 窗口调整算法 HSTCP<sup>①</sup>. 它与传统的算法设计思路相反, 首先从流量层次着手来

① Floyd S, Ratnasamy S, Shenker S. Modifying the TCP's congestion control for high speeds. <http://www.icir.org/floyd/hstep.html>, May 2002

设计合适的响应函数,然后根据所设计的响应函数给出包层次上的拥塞避免算法.为了在低速网络环境下保持 TCP 友好性,HSTCP 首先确定出一窗口阈值  $w_{low}$ ,当  $cwnd < w_{low}$  时,HSTCP 保留了传统的 TCP 窗口调整机制,根据 TCP 的响应函数可以计算出当  $w_{low} = 31$  时  $p_{low} = 0.0015$ .为了保证在传输媒介允许的包丢失率的范围内维持大的平衡窗口,HSTCP 选取在包丢失概率为  $10^{-7}$  时窗口为 83000,即  $w_{high} = 83000, p_{high} = 10^{-7}$ .然后选取  $w$  和  $p$ ,它们之间成如下的对数线性关系

$$w = 10^{s(\log(p) - \log(p_{low})) + \log(w_{low})} \quad (2)$$

其中,  $s = (\log(w_{high}) - \log(w_{low})) / (\log(p_{high}) - \log(p_{low}))$ ,代入  $w_{low}, w_{high}, p_{low}, p_{high}$  可得  $s = -0.82$ .由式(1)可得出 HSTCP 的响应函数

$$w = \frac{0.15}{p^{0.82}} \quad (3)$$

HSTCP 根据上述响应函数,设计包层次上的拥塞避免算法.它仍采用 AIMD 形式的窗口调整机制,但窗口的变化量  $a(w), b(w)$  是随着窗口的大小自适应变化的.该算法在包层次上窗口动态调整如下:

On ACK:  $w \leftarrow w + a(w)/w$ ,

On Loss:  $w \leftarrow w - b(w) \times w$ ,

其中加式增加因子  $a(w)$  和乘式下降因子  $b(w)$  的选取直接影响到该算法的响应性能. HSTCP 首先根据当前窗口大小来计算出  $b(w)$ ,具体如下式:

$$b(w) = (B - 0.5) \times \frac{\log(w) - \log(w_{low})}{\log(w_{high}) - \log(w_{low})} + 0.5 \quad (4)$$

其中,  $B = 0.1$ .由上式可知  $b(w)$  是随着窗口的增加而减少的,且在  $[0.1 \sim 0.5]$  之间变化.该设计确保了在大的窗口时,算法下降的调整量不会太大,避免了过大的窗口抖动,从而获得高的吞吐量.对于 AIMD 形式的算法,存在如下的窗口平衡状态

$$w^* = \frac{\sqrt{a(w)(2 - b(w))/2b(w)}}{\sqrt{p^*}} \quad (5)$$

由上式可知:

$$a(w) = \frac{w^2 \times 2.0 \times b(w) \times p(w)}{2.0 - b(w)} \quad (6)$$

HSTCP 通过在流量层次上设计合适的响应函数,在包层次上设计自适应变化的窗口调整量  $a(w), b(w)$ ,保证了该算法的可扩展性.

## 2.2 HSTCP 的 RTT 不公平性和 TCP 非友好性

本节主要提出 HSTCP 存在的两个主要缺点: DropTail 队列管理算法下 RTT 不公平性以及 HSTCP 流与传统的 TCP 流在网络中共同存在时

的 TCP 非友好性.

文献[7]在同步反馈的模型基础上给出了不同时延的 HSTCP 流互相竞争时的吞吐量关系,如下式(7)所示

$$\frac{w_1/RTT_1}{w_2/RTT_2} = \left(\frac{RTT_2}{RTT_1}\right)^{\frac{1}{1-d}} \quad (7)$$

其中对于 HSTCP,  $d = 0.82$ .从式(7)可以看出,当两条流的往返时延之比为  $1:2$  时,其吞吐量之比为  $76.11:1$ ,可见 HSTCP 具有较高的 RTT 不公平性.

文献[15,17-18]指出当 HSTCP 流与标准 TCP 流互相竞争可用带宽时,HSTCP 流会过分地掠夺标准 TCP 流的资源,造成标准 TCP 流极低的吞吐量.其根本原因在于 HSTCP 采用相对于 TCP Sack 过于侵略的增长方式和过于保守的下降方式造成.

除了上述两个缺陷之外,HSTCP 还存在另外一个重要的缺陷,即突发的包丢失和由于其可扩展性带来的过多的包丢失.可扩展性带来的包丢失主要是由于 HSTCP 采取了过于侵略的窗口增加机制造成的. HSTCP 算法在一个 RTT 内窗口的增加量为  $a(w)$ .当路由器发生拥塞时,源端要经过一个 RTT 的时间才能获得包丢失信息,在这段时间内,源端窗口又增加了  $a(w)$ ,此时这些新增加的数据包都要被丢掉.文献[15]中对该现象作了详细的分析和仿真验证,这里不再详细叙述.

## 2.3 相关的改进工作

针对上述 HSTCP 存在的性能缺陷,一些学者提出了改进算法,主要有 TCP Africa<sup>[17]</sup>、CB-HSTCP<sup>[15]</sup> 和 gHSTCP<sup>[18]</sup>.为了减少 HSTCP 可扩展性带来的过多的包丢失和改善 HSTCP 同标准 TCP 之间的友好性,这 3 种算法均采用了两种拥塞避免模式相互转换的窗口增加模式.当 HSTCP 流没有获得可用带宽时,采取 HSTCP 的快速的窗口增长方式去探测可用带宽,当网络即将发生拥塞时,转换为标准 TCP 的低速的窗口增加方式,这样可以避免 HSTCP 可扩展性带来的过多的包丢失.同时可以改善同标准 TCP 流之间的友好性.下面我们简单介绍 TCP Africa 和 CB-HSTCP 算法的切换点判断机制.

TCP Africa 的切换点的判断机制为

```
if (W × (RTT - RTTmin) / RTT < α) {
    W = W + fast_increase(W) / W;
} else {
    W = W + 1 / W;
}
```

其中  $W/RTT$  可以视为当前源端的发送速率,  $RTT - RTT_{min}$  为源端对链路队列时延的估计值,这样  $W \times$

$((RTT - RTT_{\min})/RTT)$  为源端对路由器缓冲区内的该条流的数据包的估计值. 当该估计值小于预先设定的阈值时, 进行增长方式的切换. 但  $\alpha$  究竟应该怎样设置是一个问题:  $\alpha$  设得过大造成切换点转换得过晚, 甚至不能转换;  $\alpha$  设得过小会造成切换点的过早转换, 造成低的吞吐量. 理想的设置应该是  $\alpha = \text{buffer size} / N$ , 其中  $N$  为用户数. 但是对于源端来说缓冲区大小是未知的, 网络中的用户数也是时变的, 这成为限制 TCP-Africa 难以准确判断切换点的重要原因.

CB-HSTCP 的切换点判断机制为

```

if  $((R_C - RTT_{\min}) / (RTT_{\max} - RTT_{\min}) < \beta)$  {
     $W += \text{increment}$ ;
} else {
     $W += 1/W$ ;
}

```

即 CB-HSTCP 是根据当前的队列时延占队列时延最大值的比值来判断切换点的, 当该比值小于一个预先设定的阈值  $\beta$  时, 进行增长方式的转换, 在实际网络中队列时延的变化是随机的, 在每次发生包丢失时获得的队列时延最大值也是随机变化的. 为说明此问题, 我们给出在 NS2 中进行的仿真实验结果来说明队列时延的动态变化情况, 拓扑结构如图 1 所示, 瓶颈链路  $R_1 \sim R_2$  之间的带宽为 1000 MB, 时延为 10 ms, 其它链路时延均为 20 ms, 其余参数见图 1, 两个源端均采用 HSTCP 代理, 5% 的 UDP 背景流, 仿真时间为 300 s. 图 2 给出了源端  $S_1$  跟踪的队列时延变化情况.

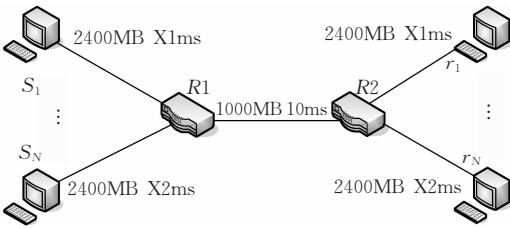


图 1 仿真拓扑结构图

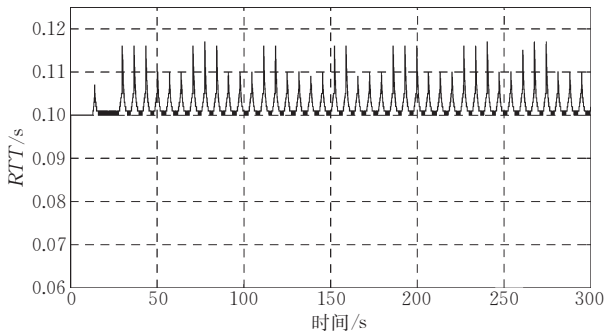


图 2 源端跟踪 RTT 动态变化图

由图 2 可以看出, 队列时延的变化是随机的. 包丢失发生时获得的队列时延最大值也是随机变化的, 跟踪出的队列时延最大值为 16 ms, 在 120 ms ~ 150 ms 之间运用此最大值会造成 HSTCP 模式不能转换为标准 TCP 的增长模式, 带来不必要的包丢失.

综上所述, 用时延作为切换点的判断依据不能完全准确地判断出切换点. 如果将 HSTCP 增长方式误判为标准 TCP 的增长方式会造成低的响应速度, 而将标准 TCP 的增长方式误判为 HSTCP 的增长方式会造成较多的包丢失. 另外这些方法中均用源端估计到的最小时延作为链路的物理时延, 在实际网络中准确地估计链路的物理时延是十分困难的, 例如当网络中已经有数据包排队时, 新加入的数据流会过高地估计链路物理时延.

### 3 EHSTCP 算法

针对 HSTCP 算法存在的性能问题, 我们主要做了两个方面的改进. 为了避免突发包丢失和改善 HSTCP 流和标准 TCP 流之间的友好性, 提出了在稳态阶段运用窗口历史值来预测当前可用带宽, 并运用窗口历史信息来估计两种模式 (HSTCP 模式和标准 TCP 模式) 相互转换的切换点, 我们称其为 EHSTCP 模式, 该模式在起始阶段拥塞窗口快速地收敛到拥塞窗口增加模式的切换点 ( $\gamma \times W_{\text{fullavg}}$ ), 然后转为标准 TCP 的低速增加模式, 该类型的窗口增加模式类似于文献[21]中给出的窗口增加模式, 保证了网络的稳定性和效率. 另外 EHSTCP 针对 HSTCP 算法在 DropTail 算法管理下存在的 RTT 不公平性问题, 通过引入 RTT 公平因子, 解决了 HSTCP 的 RTT 不公平性问题.

#### 3.1 基于历史窗口的切换点判断方法

HSTCP 窗口增长分为两个阶段, 即暂态阶段和稳态阶段. EHSTCP 算法只在稳态阶段运用窗口的历史信息对当前可用带宽进行估计, 并进行窗口增长模式的转换. 图 3 中给出了理想的 EHSTCP 算法的窗口随时间的变化图. EHSTCP 在暂态阶段保留 HSTCP 算法的窗口调整机制, 不作任何变化. 定义  $W_{\text{full}}$  为路由器丢失或标记数据包时源端的窗口大小,  $W_{\text{max}}$  为源端窗口的最大值. 因为当路由器发生拥塞时, 源端要经过 1 个 RTT 的时间才能判断出发生了拥塞, 在这段时间内源端窗口增加了  $a(w)$ , 因此  $W_{\text{full}} \approx W_{\text{max}} - a(w)$ .

暂态阶段主要分为两部分: 第一部分为 HSTCP

流刚刚启动或由于重传超时重新启动时, HSTCP 采取快速的窗口递增加模式去探测可用带宽, 此时  $W_{\max}$  呈现出连续递增的趋势. 第二部分为当网络中的某些数据流已经获得了可用带宽, 此时有新的数据流进入网络时, 原有的数据流会释放已经占据了的可用带宽, 此时  $W_{\max}$  呈现出连续下降的趋势. 则 EHSTCP 对暂态的判断准则为, 若某段时间内  $W_{\max}$  呈现出连续  $S\_num$  次上升或连续下降的趋势, 且每次上升或下降的值超过  $S\_value$  时, EHSTCP 进入暂态阶段. 根据 HSTCP 的窗口调整机制综合考虑对暂态阶段判断的准确性和快速性, 通过仿真实验, 选取  $S\_num$  和  $S\_value$  默认值分别为 3 和  $W_{\max}/32$ . 定义  $w_{prev}$  为前一时刻发生包丢失时的窗口大小. 暂态阶段判断的伪码为

```
On packet loss
if (( $w_{prev} > cwnd$ ) && ( $w_{prev} - cwnd > S\_value$ )) {
    numdec++;
    if (numdec  $\geq S\_num$ ) {
        Enter transient mode;
    }
} else {
    numdec = 0;
}
if (( $w_{prev} < cwnd$ ) && ( $cwnd - w_{prev} > S\_value$ )) {
    numinc++;
    if (numinc  $\geq S\_num$ ) {
        Enter transient mode;
    }
} else {
    numinc = 0;
}
 $w_{prev} = cwnd$ ;
```

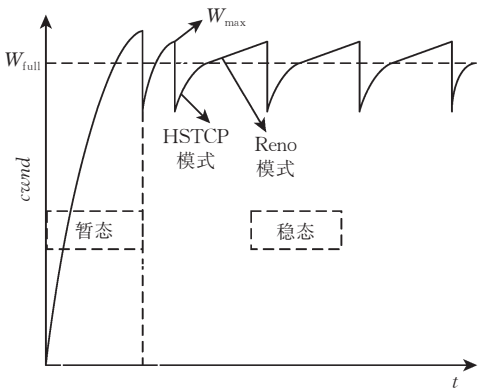


图 3 理想的 EHSTCP 窗口变化图

在稳态阶段, 源端已经获得了可用带宽, 此时  $W_{full}$  的波动是较小的. 在稳态阶段的判断准则为: 若

某段时间内, 窗口呈现出连续  $M\_mun$  次小范围内的波动, 且波动范围小于  $M\_value$  时, EHSTCP 进入稳态阶段.  $M\_mun$  和  $M\_value$  的默认值分别为 2 和  $W_{\max}/32$ . 稳态阶段的判断伪码为

```
On packet loss
if (abs( $w_{prev} - cwnd$ )  $< M\_value$ ) {
    numstab++;
    if (numstab  $\geq M\_mun$ ) {
        Enter stable mode;
    }
} else {
    numstab = 0;
}
 $w_{prev} = cwnd$ ;
```

EHSTCP 只在稳态阶段对当前可用带宽进行估计, 并根据估计结果进行窗口增长模式的转换. 在估计可用带宽时, 如果单纯利用上一次拥塞时的  $W_{full}$  作为路由器发生拥塞的判断依据是不准确的, EHSTCP 采用在稳态阶段对所得到的  $W_{full}$  进行滤波的方法得到平滑的  $W_{fullavg}$ , 并把  $W_{fullavg}$  作为当路由器发生拥塞时的源端窗口大小的预测值, 这样可以避免稳态阶段窗口的较大的波动对判断切换点造成的影响. 具体的滤波算法为  $W_{fullavg} = \beta W_{fullavg} + (1 - \beta) W_{full}$ , 其中选取  $\beta = 0.8$ , 这样比较尊重历史值, 同时可以消除小范围内窗口较大的波动对算法性能的影响. 针对  $W_{fullavg}$ , EHSTCP 设置了一个目标利用率  $\gamma$ , 当窗口大于  $\gamma W_{fullavg}$  时, 窗口的增长模式转换为标准 TCP 的增长方式.  $\gamma$  取得过小会造成 EHSTCP 由 HSTCP 的窗口增长模式过早地转换为标准 TCP 的增长模式, 造成低的吞吐量;  $\gamma$  取得过大会造成过晚的增长模式的转换, 使单位时间内的平均拥塞次数增加, 造成多的包丢失. 在这里选取  $\gamma = 0.95$  可以达到吞吐量和包丢失之间的性能折中. 该类型的窗口增加模式类似于文献[21]中给出的窗口增加模式, 保证了网络的稳定性和效率. 同文献[21]相比, 两种算法的可用带宽预测的方法都是基于历史窗口值, 不同处是文献[21]中采用的 PIMD 机制仅基于前一个历史窗口值, 而 EHSTCP 则基于前几个历史窗口的值, 先判断是否进入稳态阶段, 之后再行带宽的预测, 这样更加准确. EHSTCP 进入稳态后执行的伪码为

```
if (Enter stable mode) {
     $W_{full} = W_{max} - a(w)$ ;
     $W_{fullavg} = \beta W_{fullavg} + (1 - \beta) W_{full}$ ;
} else {
```

$$W_{\text{fullavg}} = W_{\text{full}} = W_{\text{max}} = 0;$$

Enter transient mode;

}

当窗口大于  $W_{\text{fullavg}}$  时,若还没有发生包丢失,此时可用带宽很可能已经发生了变化,EHSTCP 和 BIC 算法一样,需要重新探测可用带宽,EHSTCP 相对于 BIC 算法的主要优点在于 EHSTCP 只是在稳态阶段进行增长方式的转换,避免了在暂态阶段转换为慢速的增长方式造成的低的收敛速度.文献[22]中提到的提高 HSTCP 算法收敛速度的方法可以加入暂态阶段用以提高收敛速度.EHSTCP 窗口转换模式的伪码为

On ACK

$$a(w) = \text{pow}(RTT/t_p, \alpha) \times a(w);$$

if(( $cwnd < \gamma W_{\text{fullavg}}$ ) && (Stable mode)) {

$cwnd += a(w)/cwnd$ ; HSTCP mode

} else if( $cwnd \leq W_{\text{fullavg}}$ ) {

$cwnd += 1/cwnd$ ; Reno mode

} else {

$cwnd += a(w)/cwnd$ ; 重新探测可用带宽

}

### 3.2 RTT 公平因子

CB-HSTCP 中通过添加 RTT 公平因子  $C * RTT$ ,改善了 RTT 不公平性,添加该公平因子的效果使 HSTCP 的窗口增长与 RTT 无关,但该公平因子没有做到使吞吐量与 RTT 无关.文献[16]选取的 RTT 公平因子为  $(RTT/t_p)^2$ ,消除了 RTT 对吞吐量的影响.但此 RTT 公平因子仅针对文献[16]算法中的 AI 阶段设计,不具有一般性.我们在这里选取的 RTT 公平因子为  $(RTT/t_p)^\alpha$ ,通过合理地选取参数  $\alpha$  可以完全消除 RTT 对吞吐量的影响,并且该方法不但适应于 HSTCP 算法,而且还适应于其它的高速 TCP 算法.首先给出如下定理.

**定理 1.** 当 RTT 公平因子为  $(RTT/t_p)^\alpha$  时,两条具有不同往返时延的 HSTCP 流(其时延分别为  $RTT_1$  和  $RTT_2$ )相互竞争同一瓶颈链路带宽时,其平均的吞吐量之比为

$$\frac{T_1}{T_2} = \left( \frac{RTT_2}{RTT_1} \right)^{\frac{1-\alpha d}{1-d}} \quad (8)$$

证明. 假定流  $i$  的丢失事件服从丢包率为  $p_i$  的同一分布,则流  $i$  在两次丢包事件之间总共发送的数据包为  $1/p_i$ ,设时间为  $t$ ,则往返时延的数目为  $t/RTT_i$ ,因此平均的窗口大小为

$$w_i = \frac{1/p_i}{t/RTT_i} = \frac{RTT_i}{t p_i} \quad (9)$$

HSTCP 的吞吐量的响应函数为

$$\frac{w_i}{RTT_i} = T = \frac{1}{RTT_i} \frac{c}{p_i^d} \quad (10)$$

引入公平性因子后可以得到

$$a'(w_i) = a(w_i) \times (RTT/t_p)^\alpha \quad (11)$$

对比式(6)可得

$$p'(w_i) = p(w_i) \times (RTT/t_p)^\alpha \quad (12)$$

由式(9),(10)和式(11)可知

$$p'(w_i) = \sqrt[d]{\frac{c \times \left( \frac{RTT}{t_p} \right)^{\alpha d}}{w_i}} \quad (13)$$

令  $\beta = c \times (1/t_p)^\alpha$ ,将式(13)代入式(9)中可以得到

$$w = \left[ \frac{RTT_i^{1-\alpha}}{t \times \sqrt[d]{\beta}} \right]^{\frac{d}{d-1}} \quad (14)$$

因此其吞吐量之比为

$$\begin{aligned} \frac{T_1}{T_2} &= \frac{(w_1/RTT_1) \times (1-p_1)}{(w_2/RTT_2) \times (1-p_2)} \approx \frac{w_1/RTT_1}{w_2/RTT_2} \\ &= \left[ \frac{RTT_2}{RTT_1} \right]^{\frac{(1-\alpha)d}{1-d}} \times \frac{RTT_2}{RTT_1} = \left[ \frac{RTT_2}{RTT_1} \right]^{\frac{1-\alpha d}{1-d}} \end{aligned} \quad (15)$$

当  $\alpha$  取 1 时,吞吐量之比与往返时延成反比;当  $\alpha=1/d$  时,吞吐量之比与往返时延无关.  $\alpha$  取  $2-1/d$  时,吞吐量之比同标准 TCP 算法.在这里我们取  $\alpha=1/d$ ,此时改进后的 HSTCP 算法的吞吐量与往返时延无关,而 CB-HSTCP 中只做到了算法的窗口增加量与往返时延无关,其吞吐量之比与往返时延成反比(相当于  $\alpha=1$ ).  $t_p$  的选取将直接关系到算法的性能,  $t_p$  选取得过大,将会造成低的收敛速度,  $t_p$  选取得过小会带来多的包丢失,仿真验证选取  $t_p=100$  ms 能够获得良好的性能.为了验证引入 RTT 公平因子的有效性,在 NS2 中进行了仿真,采用图 1 所示的仿真拓扑结构,使用其中的两条数据流,源  $S_1$  到路由器  $R_1$  的时延为 20 ms,路由器  $R_2$  到接收端  $r_1$  的时延为 20 ms;源  $S_2$  到路由器的时延为 7.5 ms,路由器  $R_2$  到接收端  $r_2$  的时延为 7.5 ms,这样源  $S_1$  的往返链路时延为 100 ms,源  $S_2$  的往返链路时延为 50 ms,其余参数见图 1.两个源端均采用 HSTCP 代理,5% 的 UDP 作为背景流.分别对没有加入 RTT 公平因子的 HSTCP 以及加入 RTT 公平因子后,  $\alpha$  分别取  $2-1/d$ , 1 和  $1/d$  作了仿真,仿真结果如图 4 所示.图 4 表明,未引入 RTT 公平因子时, HSTCP 具有十分严重的 RTT 不公平性.  $\alpha=1/d$  时,长时延的 HSTCP 流的窗口增加量要大于短时延 HSTCP 流的窗口增加量,其比值接近于 2:1.仿真结果验证了引入 RTT 公平因子的有效性并验证了我们给出的解析结果.



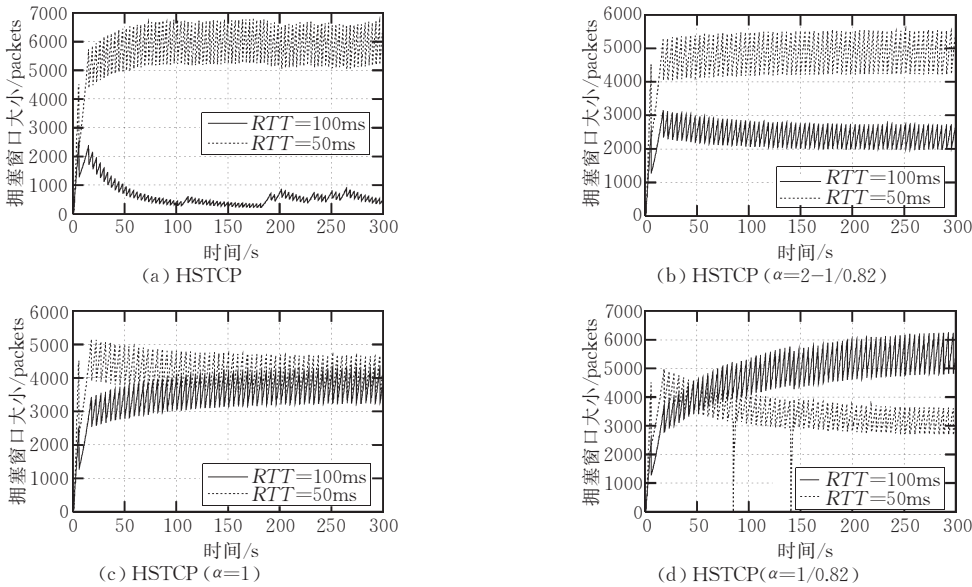


图 4 HSTCP 和引入 RTT 公平因子后的 HSTCP 的 RTT 公平性比较

4 仿真分析

为了验证 EHSTCP 算法的有效性,我们在 NS2 中实现了 EHSTCP 算法,主要对 EHSTCP 算法的 RTT 公平性,改善由于 HSTCP 可扩展性带来的过多的包丢失,同标准 TCP 之间的友好性,以及对带宽变化时的自适应能力进行仿真验证.

4.1 包丢失分析及 RTT 公平性分析

首先来研究 EHSTCP 算法的 RTT 公平性和降低由于 HSTCP 可扩展性带来的过多的包丢失.采用图 1 所示的仿真拓扑结构,通过改变该图中的参数,固定其中一条流的时延为 50 ms,另一条流的时延分别取 100 ms、150 ms、200 ms、250 ms 和 300 ms,共 6 组实验.在每次实验中均统计 3 个性能指标,即吞吐量之比、包丢失率和链路利用率.以下的仿真实验中缓冲区大小均设为 1000 packets(相对于带宽时延积是很小的).表 1 列出了  $\alpha=1$  时,EHSTCP 算法的性能;表 2 列出了  $\alpha=1/0.82$  时,EHSTCP 算法的性能;表 3 在同样的环境中列出了 HSTCP 算法的性能指标.

可见 EHSTCP 通过引入 RTT 公平因子,极大地改善了 HSTCP 算法的 RTT 不公平性,通过在稳态阶段利用历史窗口信息来判断网络拥塞,降低了包丢失率,其包丢失率相对于 HSTCP 降低了一个数量级左右.此外当  $\alpha=1$  时,EHSTCP 的链路利用率也要比 HSTCP 高出 1%.随着 RTT 比值的增

高,EHSTCP 的 RTT 不公平性也会提高,当  $\alpha=1/0.82$  时,EHSTCP 的吞吐量之比基本维持在 1 : 2 左右,与链路的往返时延无关,这和我们给出的理论分析结果有一定的出入,这是因为在实际网络中包丢失并不是服从同一分布的,而是随机的,此外我们是根据 HSTCP 的响应函数给出的解析结果,实际的仿真结果并不是严格服从响应函数.仿真实验证明了 EHSTCP 算法的有效性.此外文献[15]中用到的 Block-pacing 方法可以作为本文研究的补充,进一步避免 HSTCP 存在的突发的包丢失,我们并没有加入 Pacing 机制的原因在于,现有的研究<sup>[23]</sup>表明光学突发交换技术(OBS)最有希望成为将来 Internet 骨干网的交换技术,研究表明在使用 OBS 时,使用 pacing 技术时会造成吞吐量性能上的下降.

表 1 EHSTCP 有效性及 RTT 不公平性测试 $\alpha=1$			
RTT 比率	吞吐量比率	包丢失率	链路利用率/%
1 : 2	2.582	$3.75 \times 10^{-6}$	98.06
1 : 3	5.363	$3.86 \times 10^{-6}$	98.00
1 : 4	8.237	$2.56 \times 10^{-6}$	98.10
1 : 5	10.109	$3.20 \times 10^{-6}$	97.63
1 : 6	17.357	$5.96 \times 10^{-6}$	98.21

表 2 EHSTCP 有效性及 RTT 不公平性测试 ( $\alpha=1/0.82$ )			
RTT 比率	吞吐量比率	包丢失率	链路利用率/%
1 : 2	1.319	$3.78 \times 10^{-6}$	97.83
1 : 3	2.604	$6.22 \times 10^{-6}$	97.20
1 : 4	2.445	$6.64 \times 10^{-6}$	97.05
1 : 5	2.048	$7.34 \times 10^{-6}$	96.39
1 : 6	2.737	$1.02 \times 10^{-5}$	96.67

表 3 HSTCP 有效性及 RTT 不公平性测试

RTT 比率	吞吐量比率	包丢失率	链路利用率/%
1 : 2	17.433	$6.73 \times 10^{-5}$	96.72
1 : 3	101.960	$7.71 \times 10^{-5}$	96.93
1 : 4	210.946	$8.84 \times 10^{-5}$	96.98
1 : 5	305.852	$7.06 \times 10^{-5}$	96.69
1 : 6	662.161	$1.04 \times 10^{-5}$	97.23

4.2 EHSTCP VS TCP Sack

本节将通过仿真分析来验证 EHSTCP 算法与标准 TCP 算法之间的友好性,采用图 1 所示的拓扑结构,源  $S_1$  采用 EHSTCP 代理,源  $S_2$  采用 TCP Sack 代理,重复 2.2.2 节中的仿真实验,仿真结果如图 5 所示。

图 5 中共给出了  $\gamma=0.94, \gamma=0.95, \gamma=0.96$  和  $\gamma=0.97$  四组参数时 EHSTCP 和 TCP Sack 的窗口变化。仿真实验结果表明 EHSTCP 算法在与 TCP Sack 算法共同存在于网络中时, EHSTCP 不会过

分地掠夺 TCP Sack 算法的资源,造成其过低的吞吐量。其中 TCP Sack 所获得的吞吐量可以近似由下式计算

$$T_{\text{Sack}} = \frac{C \times (1 - \gamma)}{N}$$

(16)

其中  $N$  为连接数,因为在暂态阶段 EHSTCP 采取 HSTCP 的窗口增长方式,造成了 TCP Sack 较低的吞吐量,而在稳态阶段, EHSTCP 要和 TCP Sack 共同竞争  $C \times (1 - \gamma)$  可用带宽,由于 EHSTCP 采用了标准 TCP 的窗口增长方式,因此在稳态阶段 EHSTCP 将和 TCP Sack 均分此部分带宽。图 5 表明随着  $\gamma$  的增加, TCP Sack 获得的吞吐量逐渐降低,当  $\gamma=0.95$  时,达到令人满意的效果,相对于 HSTCP (图 3), EHSTCP 在与 TCP Sack 共同存在的复合网络环境中具有良好的 TCP 友好性。

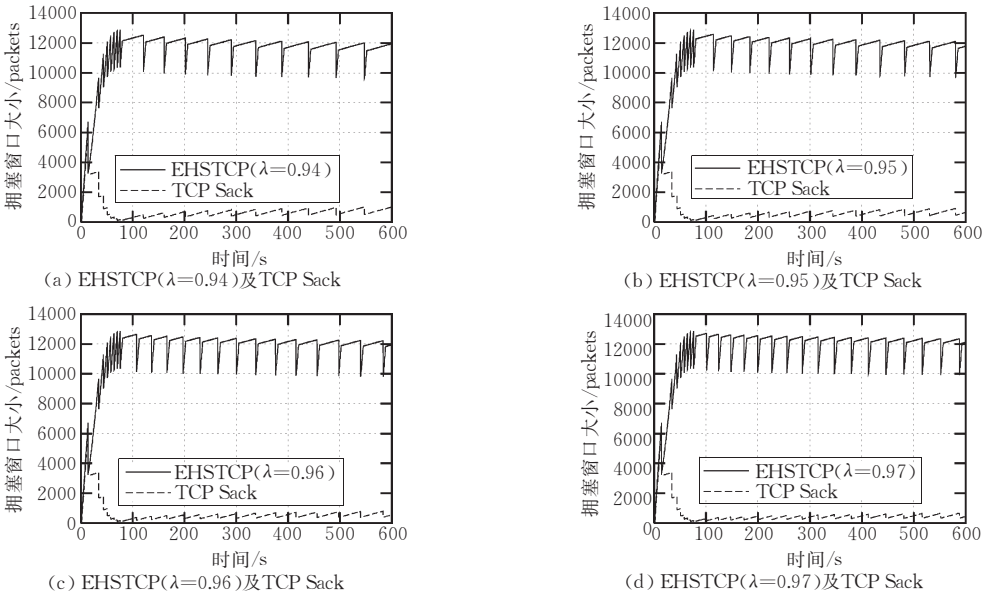


图 5 EHSTCP 和 TCP Sack 窗口变化

接下来验证多条流时 EHSTCP 与标准 TCP 之间的友好性问题,采用图 1 所示的网络仿真环境,将流的数增加到 10 条,瓶颈链路的带宽增加到 2500 Mbps,时延为 10 ms。源  $S_1 \sim S_5$  到路由器  $R_1$  之间的带宽为 100 Mbps,源  $S_6 \sim S_{10}$  到路由器  $R_2$  之间的带宽为 2400 Mbps,仿真时实验共分 3 组,其中在任何一组实验中源  $S_6 \sim S_{10}$  均采用 TCP Sack 代理,源  $S_1 \sim S_5$  分别采用 TCP Sack 代理, HSTCP 代理和 EHSTCP 代理。设置图 1 中的参数  $X1=20\text{ ms}$ ,  $X2=20\text{ ms}$ ,使其往返链路时延为 100 ms,仿真时间为 150 s,在每组实验中均统计源  $S_1 \sim S_5$  和源  $S_6 \sim S_{10}$  的总的吞吐量之比,仿真结果如表 4 所示。

表 4 EHSTCP 友好性测试(链路带宽为 2500 Mbps,流的数目为 10)

源算法	链路利用率/%			
	Sack, HSTCP, EHSTCP	Sack	UDP	总计
Sack 与 Sack	65.86	14.34	5.10	85.3
HSTCP 与 Sack	87.56	3.39	4.85	95.8
EHSTCP 与 Sack	81.32	9.92	4.96	96.2

从表 4 中可以看出当源  $S_6 \sim S_{10}$  采用 HSTCP 代理时,由于 HSTCP 算法采取过于侵略的增长方式,过分地掠夺了 TCP Sack 所使用的资源,而 EHSTCP 在保证瓶颈链路利用率的同时,没有过分剥夺 TCP Sack 的资源,各业务流之间资源分配较为均衡,仿真结果验证了 EHSTCP 算法与标准 TCP 算法之间的友好性。



### 4.3 响应性能分析

响应性能是评价一个源端算法好坏的重要标准,一个好的算法必须能够对带宽的变化迅速做出响应,接下来研究 EHSTCP 算法的响应性能问题.采用图 1 所示的拓扑结构,其中源  $S_1$  采用 EHSTCP 代理,源  $S_2$  采用 UDP 代理,在 200 s 时加入 UDP 流,其发送速率为 500 Mbps,在仿真时间为 400 s 时,撤去 UDP 流,仿真时间为 600 s.图 6 给出了 EHSTCP 的窗口变化图,可见当加入 UDP 流时,EHSTCP 能够对可用带宽的变化迅速做出响应,获得新的可用带宽.

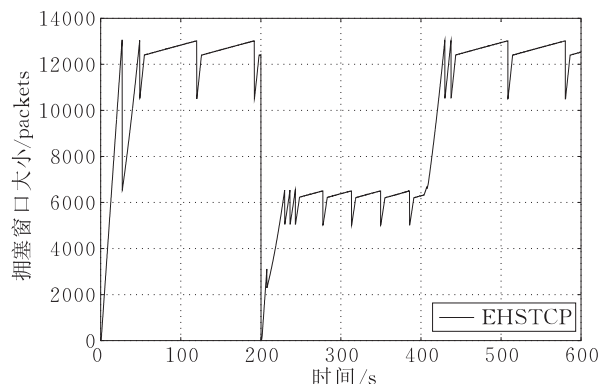


图 6 EHSTCP 响应性能(加入 UDP 流)

## 5 结论和研究展望

本文详细分析了 HSTCP 算法存在的缺陷,通过仿真证实了 HSTCP 算法在去尾算法下存在严重的 RTT 不公平性以及同标准 TCP 算法共同存在的网络环境中会过分地掠夺标准 TCP 算法的资源,此外 HSTCP 还存在由于其可扩展性带来的过多的包丢失等缺陷.针对这些缺陷,我们提出了 EHSTCP 算法,该算法通过引入新的 RTT 公平因子,可以极大地改善 RTT 不公平性,通过选取公平因子中的参数  $\alpha$ ,可以完全消除 RTT 不公平性现象( $\alpha=1/d$ );提出了在稳态阶段运用窗口历史值来预测网络拥塞,并利用窗口历史信息来获得窗口增长模式的切换点.理论分析和 NS2 仿真实验均表明 EHSTCP 在保留了 HSTCP 算法的优点的基础上,降低了包丢失率、在与标准 TCP 流共同存在的复杂网络环境中,不会过分地掠夺标准 TCP 流的资源,保证了算法的安全性能.

### 参 考 文 献

[1] Zhang Miao, Wu Jian-Ping, Lin Chuang. Survey on Internet

end-to-end congestion control. Journal of Software, 2002, 13(3): 354-363(in Chinese)

(章森,吴建平,林闯.互联网端到端拥塞控制研究综述.软件学报, 2002, 13(3): 354-363)

- [2] Jacobson V. Modified TCP congestion avoidance algorithm. End2end-interest mailing list, April 30, 1990
- [3] Floyd S. The NewReno Modification to TCP's Fast recovery algorithm. IETF RFC 2582, April 1999
- [4] Mathis M, Mahdavi J, Floyd S. TCP selective acknowledgment options. IETF RFC 2018, October 1996
- [5] Brakmo L S, Peterson L L. TCP Vegas: End-to-end congestion avoidance on a global Internet. IEEE Journal on Selected Areas in Communications, 1995, 13(8): 1465-1480
- [6] Kelly T. Scalable TCP: Improving performance in high speed wide area networks. ACM Computer Communications Review, 2003, 33(2): 83-91
- [7] Xu L, Harfoush K, Rhee I. Binary increase congestion control (BIC) for fast Long-distance networks//Proceedings of the IEEE INFOCOM 2002. Hong Kong, China, 2004: 2514-2524
- [8] Bhandarkar S, Jain S, Reddy A L N. LTCP: Improving the performance of TCP in highspeed networks. ACM SIGCOMM Computer Communication Review, 2006, 36(1): 41-50
- [9] Jin C, Wei D X, Low S H. FAST TCP: Motivation, architecture, algorithms performance//Proceedings of the IEEE INFOCOM 2004. Hong Kong, China, 2004: 2490-2501
- [10] Katabi D, Handley M, Rohrs C. Congestion control for high bandwidth-delay product networks//Proceedings of the ACM SIGCOMM 2002. Pittsburgh, Pennsylvania, USA, 2002: 89-102
- [11] Shorten R N, Leith D J. H-TCP: TCP for high-speed and long-distance networks//Proceedings of the PFLDnet. Argonne, Illinois, 2004
- [12] Hatano T, Fukuhara M, Shigeno H, Okada K. TCP-friendly SQRT TCP for high-speed networks//Proceedings of the APSITT. Noumea New Caledonia, 2003: 455-460
- [13] Rhee Injong, Xu Lisong. CUBIC: A new TCP-friendly high-speed TCP variant//Proceedings of the 3rd International Workshop on Protocols for Fast Long-Distance Networks. Lyon, France, 2005
- [14] Kunniyur S. AntiECN Marking: A marking scheme for high bandwidth delay connections//Proceedings of the IEEE ICC. Anchorage, USA, 2003: 647-651
- [15] Su Fan-Jun, Pan Xue-Zeng, Cai Liang, Xu Jian. CB-HSTCP: Fair TCP in high speed network. Chinese Journal of Electronics, 2005, 11(33): 2084-2089(in Chinese)  
(苏凡军,潘雪增,蔡亮,徐建. CB-HSTCP: 高速网络中的公平 TCP 算法. 电子学报, 2005, 11(33): 2084-2089)
- [16] Xia Y, Subramanian L, Stoica I, Kalyanaraman S. One more bit is enough//Proceedings of the ACM SIGCOMM'05. Philadelphia, Pennsylvania, USA, 2005: 37-48

- [17] King R, Baraniuk R, Riedi R. TCP-Africa: An adaptive and fair rapid increase rule for scalable TCP//Proceedings of the IEEE INFOCOM 2005. Miami, USA, 2005; 1838-1848
- [18] Zhang Zong-Sheng, Hasegawa G, Murata M. Performance analysis and improvement of high-speed TCP with TailDrop/RED routers. IEICE Transactions on Communication, 2005, E88-B(6): 2495-2507
- [19] Biaz S, Vaidya N H. Is the round-trip time correlated with the number of packets in flight? Texas A&M University, Laredo of USA: Technical Report TR99-006, 1999
- [20] Martin J, Nilsson A, Rhee I. Delay-based congestion avoidance for TCP. IEEE/ACM Transactions on Networking, 2003, 11(3): 356-369
- [21] Shao Li-Song, Zhang He-Ying, Dou Wen-Hua. Window-Based end-to-end congestion control: Network stability and efficiency. Chinese Journal of Computers, 2005, 29(3): 353-360(in Chinese)  
(邵立松, 张鹤颖, 窦文化. 基于窗口的端到端拥塞控制: 网络稳定性与效率. 计算机学报, 2005, 29(3): 353-360)
- [22] Nabeshima M, Yata K. Improving the convergence time of high-speed TCP//Proceedings of the IEEE International Conference on Networks (ICON2004). Singapore, 2004; 19-23
- [23] Zhu L, Ansari N, Liu J. Throughput of high-speed TCP in optical burst switching networks. IEE Proceedings-Communications, 2005, 152(3): 349-352



**LONG Cheng-Nian**, born in 1977, Ph. D., associate professor. His current research interests include congestion control for high-speed Internet, noncooperative behavior and incentive mechanism design in wireless multi-hop networks, and energy-efficiency protocol design in wireless sensor networking.

**YANG Hui-Long**, born in 1981, master. His main re-

search interest is congestion control of high-speed networks.

**LI Xin**, born in 1981, master candidate. His main research interest is congestion control of high-speed networks.

**GUAN Xin-Ping**, born in 1963, Ph. D., professor. His current research interests include wireless sensor networks, congestion control of networks, robust control and intelligent control for nonlinear systems, chaos control and synchronization.

## Background

As computing, communications and storage technology of rapid development, the global grid system has provided the enough capacity and the high speed effective hardware environment for the computation and the scientific research. Development of the new generation network, the challenge is the existing network control algorithm and the resource allocation algorithm can not be extended to the next generation of high-bandwidth communications network. The main evidence is that, in the high-speed network environment, the existing protocol algorithm can not guarantee the required QoS (Quality of Service), for example low packet loss rate, low delay and delay jitter. Due to the inefficient congestion control algorithm which led to the uncooperative users competition and greed occupied bandwidth, it will even leads to instability in the entire network operation, significantly reducing the QoS. In high-bandwidth network environment, HSTCP algorithm solves the major performance deficiencies of standard TCP algorithm. The proposed solution is simple to implement, which has been adopted by the IETF. However,

HSTCP adopts non-linear growth pattern in congestion avoidance. There will be a large amount of data packet loss and RTT unfairness when the sudden flow generated at the point of congestion. To resolve these issues, the authors propose an improved HSTCP algorithm in this paper. The basic idea of the proposed algorithm is to choose different congestion avoidance manners under different network congestion statuses. In this paper, the authors present a new end-to-end available bandwidth prediction method based on the history values of congestion window. Based on the prediction information, it can achieve a suitable switch point for different congestion avoidance stages. Simultaneously, the authors introduce the RTT fairness factor to eliminate the serious RTT unfairness in HSTCP. NS2 simulation results validate the effectiveness of the proposed scheme. This work was supported in part by the National Science Fund for Distinguished Young Scholars of China under grant No. 60525303 and National Natural Science Foundation of China under grant No. 60404022.