

一种协同改进 FAST TCP 公平性算法

陈晓龙^{1,2}, 章 云¹, 刘 治¹

(1. 广东工业大学 自动化学院, 广东 广州 510006; 2. 茂名学院 计算机与电子信息学院, 广东 茂名 525000)

摘要: 针对同时到达多个 FAST TCP 连接时, 新连接存在难以估测的传播延时而导致不公平性的问题, 提出一种协同改进算法. 该算法通过各连接充分利用源端获得的局部信息, 协同合作确定同步回退时钟和最小回退因子. 在无法直接通信的情况下, 实现短暂同步回退清空缓冲队列长度, 确保快速获得准确的传播延时和 FAST TCP 协议的公平性. NS-2 仿真实验表明, 该算法与其它相关改进算法相比, 在失去了较小稳定性的情况下确保了各连接良好的公平性.

关键词: 公平性; 传播延时; 局部信息; 协同; 同步回退时钟

中图分类号: TP393

文献标识码: A

文章编号: 1007-7162(2010)04-0004-05

FAST TCP^[1-5] 是加州理工学院网络实验室 Steven H. Low 等人针对高速、长延迟、大容量的下一代高性能网络提出的一种新型传输控制协议. 在高速网络环境中, 和基于分组丢失作为拥塞反馈信号的改进 TCP 协议相比, FAST TCP 采用排队延时信息估计网络拥塞状态, 确定窗口调整策略, 取得了更好的稳定性和更充分的链路使用效率^[1]. 但其也存在难以正确估测传播延时 (BaseRTT) 这个公开问题^[1,4-5].

由文献[6-7] 知, FAST TCP 连接 (以下简称连接) 将源端获得的最小 RTT 作为传播延时. 因此, 当网络中已存在处于平衡状态下 FAST TCP 连接 (以下简称旧连接) 时, 新建的 FAST TCP 连接 (以下简称新连接) 将难以快速获得准确的传播延时, 从而导致新、旧连接无法公平地分配链路带宽^[6-7].

为了使各连接能够快速获得准确的传播延时, 公平地分配链路带宽, 有学者作了下面的研究工作. 谭连生等人^[6] 提出一种改进每个连接第一个分组排队优先级的算法. 该算法将每个连接发送的第一个分组设置高优先级, 使得该分组经过链路时不需要参与排队直接发送, 从而根据第一个分组的 RTT 得到该连接的准确的传播延时. 该算法确实能获得准确的传播延时和保证各连接公平地分配链路带宽, 但需要修改每个连接第一个分组头部的内容和链路端队列调度算法的支持, 存在部署方面的问题. Tony

Cui 等人^[8] 指出了在没有路由器等网络设备的帮助下, 短时清空链路端缓冲空间的排队队列是 FAST TCP 源端获得准确传播延时的唯一方法. 因此, 提出了对每个新建的连接, 当到达平衡点后采取暂时的回退策略, 但该策略要求新连接清空队列的时间必须小于旧连接的传播延时, 否则当旧连接发现新连接回退时会自发增加发送速率, 使得链路缓冲区无法清空, 新连接无法得到准确的传播延时. Miguel 等人^[9] 提出了一种利用源端局部信息估计传播延时的改进算法. 该算法对每个新建的连接, 当其到达平衡点后, 记录该链接的 RTT 和发送速率, 然后以一定比例提高新连接的发送速率, 在规定间隔下, 记录其 RTT 的变化情况; 根据其源端能够获得的局部信息 (调整比例、RTT 变化量、记录间隔) 估计旧连接数和链路带宽, 计算其传播延时. 该方法弥补了前两种改进策略的不足之处, 但该方法只对新建立一个连接时有效, 若同时到达多个新连接, 就无法利用局部信息计算旧连接数和链路传输流量, 无法估计传播延时.

针对上述改进算法的局限性, 本文假设链路上已存在多个已知其准确的传播延时且处于平衡态的旧连接时 (由文献[6,8-9] 知, 当有连接释放时, 队列将会呈现短暂清空现象, 因而任何长持续流将有机会获得准确的传播延时, 因而该假设是合理的. 但获得准确的传播延时的时间是不可控的, 特别是出

收稿日期: 2010-04-06

基金项目: 国家和广东省联合基金资助项目 (U0735003); 国家自然科学基金资助项目 (60604006); 教育部博士点基金资助项目 (20070562005)

作者简介: 陈晓龙 (1971-), 男, 副教授, 博士研究生, 主要研究方向为网络拥塞控制、非线性控制.

现文献[7-8]描述的持续拥塞(Persistent Congestion)情况,可能会造成长时间的不公平性),又有多个新连接同时到达的情况,提出一种协同式改进 FAST TCP 窗口调整策略.该策略充分利用各连接能够获得的局部信息,协同确定同步回退时间和合理的回退幅度,当有新连接到达时,新、旧连接能够快速同步回退清空队列,使得新连接能够快速获得准确传播延时,从而使得新、旧连接能够公平地分配链路带宽.

1 基本概念和平衡态特性比值

网络拓扑采用哑铃型网络拓扑,链路带宽为 c (packets/s),各源端采用 FAST TCP 协议,取相同的协议参数 α .假设链路上已建立了 κ 个已知准确传播延时,且处于平衡态(以下称前平衡态)的持续的旧连接时,又到达了 η 个持续的新连接.由文献[6]知每个旧连接留 α 个分组在链路缓冲区.当 η 个新连接建立时,系统经过短暂调整后又会到达新的平衡态(以下简称后平衡态).由文献[8]知,每个旧连接在后平衡态留在链路缓冲区分组的个数仍为 α ,但新连接个数无法准确确定,假设每个新连接到达后平衡态时留在链路缓冲区个数相同,总个数为 B ,都获得相同的传输流量.

根据 FAST TCP 平衡点^[1-2]性质,当新连接到达后,新、旧连接经过短暂调整到达后平衡态时,对每一个旧连接(下标取 old)有如下关系成立:

排队延时

$$q_{old} = (\kappa\alpha + B) \quad (s). \quad (1)$$

前平衡态传输流量

$$x_{old}^f = c/\kappa \quad (\text{packets/s}). \quad (2)$$

后平衡态时传输流量

$$x_{old}^b = \alpha c/(\kappa\alpha + B) \quad (\text{packets/s}). \quad (3)$$

后、前平衡态传输流量比值

$$z = x_{old}^b/x_{old}^f. \quad (4)$$

后、前平衡态传播延时不变,为 $d(s)$.

同样对每一个新连接(下标取 new)有:

θ ,预先设置的第一次回退采用的回退因子.

传播延时

$$d + \kappa\alpha/c \quad (s).$$

排队延时

$$q_{new} = B/c \quad (s). \quad (5)$$

传输流量

$$x_{new} = \alpha c/B \quad (\text{packets/s}). \quad (6)$$

定义 j_t 为每个连接预先设置的时间间隔.

每个旧连接在平衡态可探测到的已知的信息包含本连接的协议参数 α ,排队延时 q ,传输流量 x ,预设新连接回退因子 θ ,旧连接后、前平衡态传输流量比 $z = x_{old}^b/x_{old}^f$.虽然每个旧连接无法获得网络参数 η, κ, B, c ,但旧连接可根据 FAST TCP 协议平衡态性质和在源端能够获得的局部信息,求解新、旧连接数比值 η/κ .

定理 1 每个旧连接在进入后平衡态时,新、旧连接数比值与后、前平衡态传输流量比值满足 $\eta/\kappa = (1 - z)^2/z$.

证明

新连接到达后平衡态时有

$$\kappa x_{old}^b + \eta x_{new} = c, \quad (7)$$

将式(3)和式(6)代入式(7)得

$$\kappa\alpha/(\kappa\alpha + B) + \eta\alpha/B = 1, \quad (8)$$

$$B = \alpha(\eta + \sqrt{\eta^2 + 4\eta\kappa})/2. \quad (9)$$

将式(2)和式(3)代入式(4)得

$$z = x_{old}^b/x_{old}^f = \kappa\alpha/(\kappa\alpha + B). \quad (10)$$

将式(9)代入式(10)得

$$z = 2\kappa/(2\kappa + \eta + \sqrt{\eta^2 + 4\eta\kappa}). \quad (11)$$

变换式(11)得

$$z = 2/(2 + (\eta/\kappa) + \sqrt{(\eta/\kappa)^2 + 4(\eta/\kappa)}), \quad (12)$$

由式(12)知, z 越小,新连接数所占比例越大.

证毕.

2 协同同步回退策略

1) 协同同步回退策略思路分析.

由文献[6,8]知,只要链路存在处于平衡态的旧连接,新建的连接便难以快速获得准确的传播延时.而要想新连接快速获得准确的传播延时,最好的办法就是到达平衡态后短暂清空缓冲区队列^[8].文献[8]只是针对新建立连接采取回退策略,存在一定缺陷.若所有新、旧连接能够同步进行回退,不但可以避免文献[8]过高的限制条件和回退时激烈的振荡,而且突破了文献[9]所提方法的局限之处,即每次只允许建立一个新连接.虽然,各 TCP 连接都是一个单独执行的进程,无法直接通信实现同步回退策略,但由于它们共存于一链路,当有新连接建立时,该新连接会通过影响链路的状态,间接通知旧连接,因此,旧连接会同时表现出一些特有的相同的特征.本节正是通过综合分析各连接出现的历史特征信息,协同合作找到同步时钟,估算最小回退因子,实现同步回退清空队列.具体解决思路如下:

(1) 旧连接同步时钟.

在平衡状态下,建立新连接或释放旧连接时,其所经过链路状态信息会发生变化,此时各旧连接在经过一个 RTT 后,会同时得到链路变化的反馈信息,并立即重新调整窗口,进入暂态.因此,旧连接可记录下进入暂态的时间 zt ,并根据暂态时间 zt ,计算新连接到达或旧连接释放时间:

$$\text{syntime} = zt - \text{RTT}. \quad (13)$$

从而所有旧连接获得了同步时钟.

(2) 旧连接确定同步回退时间.

旧连接在得到同步时间后,并不马上进行回退,而是经过一短暂的时间调整后,各连接又进入平衡态.若该平衡态发送数率 $x_{\text{old}}^b < x_{\text{old}}^f$,两个平衡态之间没有执行回退策略,传播延时没有减小,则旧连接可准确判定,该平衡态是因为有新连接达到而重新调整达到的.在各连接预先设置足够大的时间间隔 jt 的基础上,确定旧连接同步回退时间 $\text{backtime} = \text{syntime} + jt$,在该时刻执行同步回退策略.

(3) 新连接同步回退.

当旧连接开始回退时,新连接发送窗口会因为它们的回退而迅速增大,若不同时采取回退措施,将无法清空队列和无法探测到准确的传播延时.因此新建立的连接在首次到达平衡态,若发送窗口变大、RTT 变小,说明旧连接已开始回退,为达到清空队列的目的,新连接采取同步回退策略.

(4) 新、旧连接最小回退因子.

在实现同步回退后,要考虑选择最小回退因子,确保链路快速同步回退清空队列,获得准确传播延时和公平分配链路带宽的同时,最小幅度地降低系统的稳定性.

由文献[9]知当 FAST TCP 协议处于平衡态时有

$$\alpha_i = x_i q_i = \frac{w_i}{R_i} (R_i - d_i) = w_i - x_i d_i, \quad (14)$$

其中 w_i 表示连接 i 发送窗口,一个 RTT 内需要发送分组的个数; $x_i d_i$ 表示一个 RTT 内连接 i 中处于传播线路上正在发送的分组个数 (packets).

由式(14)知协议参数 α_i 可表示连接 i 留在缓冲区的个数.说明当新连接到达平衡态,链路在一个 RTT 内可以传输 $\sum w - \kappa\alpha - B$ 个分组.因而至少要将全部连接的发送窗口减小 $\kappa\alpha + B$ 个分组,可保证在一个 RTT 内清空队列.因假设同时建立的新连接获得相同的传输流量,由式(9)得每个新连接留在缓冲区分组的个数

$$b = B/\eta = \alpha(1 + \sqrt{1 + 4\kappa/\eta})/2. \quad (15)$$

由式(12)、式(15)知 $b > \alpha$,且 z 越大, b 越大.

式(15)虽理论分析了 $b > \alpha$,由于新连接无法获得 η/κ ,对每一个新连接并不能根据式(15)计算出来,因而只能将每个新连接窗口回退因子预先设置 $\text{backoff}_{\text{new}} = \theta\alpha$ (θ 预先设置,取 $\theta > 1$),则在回退阶段,将窗口设置为

$$w_{\text{new}} = w_{\text{new}} - \theta\alpha. \quad (16)$$

在预设新连接回退因子情况下,可设置每个旧连接最小回退因子为

$$(\kappa\alpha + B - \eta\theta\alpha)/\kappa. \quad (17)$$

由式(17)知,虽然旧连接无法获得具体的 κ, η ,但将式(9)、式(12)代入式(17)后得

$$\begin{aligned} \text{backoff}_{\text{old}} &= (\kappa\alpha + \alpha(\eta + \sqrt{\eta^2 + 4\eta\kappa}) / \\ &2 - \eta\theta\alpha) / \kappa = \alpha(1 + (1 - z) \times \\ &(\sqrt{((1 - z)/z)^2 + 4/z}) / 2 + ((1 - z)^2 / z) \times \\ &(0.5 - \theta)). \end{aligned} \quad (18)$$

由式(18)知旧连接可通过后、前平衡态传输流量比、协议参数 α 和新连接回退因子计算出旧连接最小回退因子.

旧连接在回退阶段发送窗口调整策略为

$$w_{\text{old}} = w_{\text{old}} - \text{backoff}_{\text{old}} w_{\text{old}}. \quad (19)$$

综上所述,新、旧连接能够在没有直接的通信前提下,利用自己获得的局部历史变化信息获得了回退同步时钟,并根据新连接到达前、后旧连接平衡态的传输流量比值,定量分析了每个新连接实际留在链路缓冲区的分组的个数和旧连接最小的回退因子.无论是预设的新连接回退因子还是式(18)给出的最小的回退因子都远小于文献[8]给出的回退因子.

2) 协同同步回退算法状态图描述.

综上所述,建立具有同步回退策略的改进 FAST TCP 窗口调整策略状态图模型,该模型包括 4 种状态、3 个事件、2 个条件和 3 个动作.

(1) 状态

初态:TCP 连接建立,开始传输数据,Flag 为真.

终态:数据传输结束,TCP 连接释放.

暂态^[10]:非平衡态,即发送窗口变化的绝对值不小于 $z\text{value}^{[10-11]}$ (很小的常数).

平衡态^[10]:连续发送窗口变化的绝对值恒小于 $p\text{value}^{[10-11]}$ (很小的常数).

(2) 事件

事件 1:FAST TCP 源端发送窗口变化的绝对值首次出现连续 $z\text{num}^{[10-11]}$ 次大于 $z\text{value}$.

事件2:FAST TCP 源端发送窗口变化的绝对值首次连续 $pnum^{[10-11]}$ 次小于 $pvalue$.

事件3:TCP 连接数据传输完毕.

(3) 条件

条件1同时满足以下4点:

(a) Flag 为 Flase,本次平衡态传输流量小于上次平衡态传输流量,表明因新建连接引起的平衡态转移.

(b) 两次平衡态之间没有通过旧连接回退状态,避免连续回退情况.

(c) 传播延时没有减小,排除新连接回退到平衡态后又继续回退.

(d) 系统运行到达同步时间.

条件2:Flag 为真,该连接发送窗口连续 $znum$ 次变大,RTT 连续 $znum$ 次变小.

(4) 动作

动作1:在指定的回退时间(RTT的整数倍)内,按式(19)计算的回退窗口大小发送数据分组.

动作2:在指定的回退时间(RTT的整数倍)内,按式(16)计算的回退窗口大小发送数据分组,设置Flag为假.

动作3:记录进入暂态的时间 zt .

(5) 状态事件转换图(见图1).

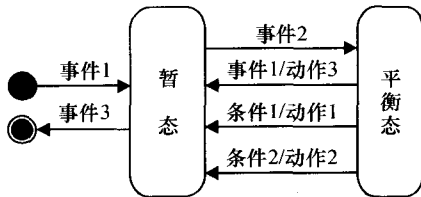


图1 同步回退策略状态事件转换图

3 仿真分析

为了验证上述算法的有效性,将以上描述的同步回退策略融入到 FAST TCP 代码(<http://www.cubinlab.ee.Unimelb.edu.au/ns2fasttcp>)的窗口控制模块.仿真环境为 UBUNTU7.04 + NS2.31,仿真采用如图2所示的哑铃型网络拓扑结构.

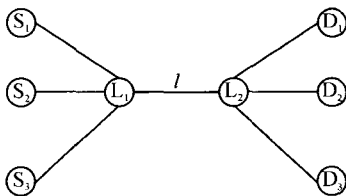


图2 哑铃型网络拓扑

网络参数设置如下:

链路带宽 c 为 96 Mb/s (即 $c = 12\,000$ packet/s),缓冲区大小为 20 000 packet/s,链路采用丢尾算法(DropTail),仿真时间 1 000 s.链路包含 3 条路径.

路径1: S_1-D_1 在仿真开始时建立 4 个 FAST TCP 连接,进行 FTP 传输,运行时间区间为 0 ~ 1 000 s,传播延时为 0.1 s.

路径2: S_2-D_2 在 200 s 开始建立 6 个 FAST TCP 连接,进行 FTP 传输,运行时间区间为 200 ~ 800 s,传播延时为 0.14 s.

路径3: S_3-D_3 在 400 s 开始建立 4 个 FAST TCP 连接,进行 FTP 传输,运行时间区间为 400 ~ 600 s,传播延时为 0.12 s.

本改进策略参数设置:

$\alpha = 50$, $\theta = 1.5$, $znum = 3$, $zvalue = w_{max}/32^{[11]}$ (w_{max} 为连接记录的最大窗口取值), $pnum = 2$, $pvalue = w_{max}/500^{[11]}$.

仿真结果如图3、图4所示,刚启动时,链路存在4个已处于平衡态且已知其准确传播延时的旧连接.当仿真运行到200 s时,有6个新连接同时建立,经过短暂时间调整后进入平衡态.由图3可看出,当新连接刚建立时,新、旧连接表现出不公平性.新、旧连接采用本文提出的同步回退策略经过短暂时间回退后,由图4可看出缓冲队列在短暂时间内被清空.由图3知,经过源端窗口回退、链路清空队列后,新

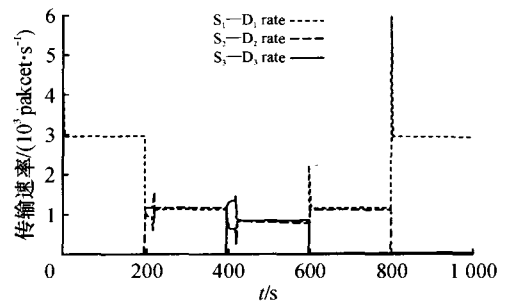


图3 各路径传输速率

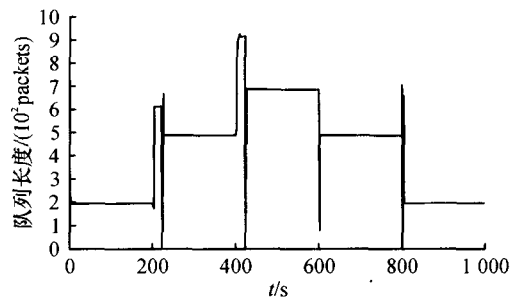


图4 链路队列长度

连接获得准确的传播延时后,新、旧连接公平地分配了链路带宽. 同样当运行到 400 s 时,又有 4 个新连接同时建立. 由图 3 知,经过本策略短暂调整后,新、旧连接又能够公平地分配了链路带宽. 而当运行到 600 s、800 s 时,由于释放了连接,不需要运行本文的回退策略.

由图 3 和文献[7]中图 1 相比,由于本文给出的合理的回退因子,系统表现出较好的稳定性.

4 结论

虽然网络中 FAST TCP 连接不断地建立和释放,使得任何长持续流有机会获得准确的传播延时,但是它们获得准确的传播延时和完全的公平性的时间是不可控的,特别是出现文献[7,9]描述的持续拥塞情况,将会造成长时间的不公平性. 因而本文提出的改进算法,一旦发现有新连接建立,在不需要路由支持和相互通信,只是完全根据各连接自己能够获得局部信息,协同合作快速实现了所有连接的最小同步回退、清空队列,使得各连接短时间内获得准确的传播延时和完全的公平性. 与其它相关改进算法相比,本算法具有不存在部署问题、回退幅度小、允许同时建立多个新连接和实现了各连接完全公平性等优点.

参考文献:

[1] David X Wei, Cheng Jin, Low S H. FAST TCP: Motivation,

Architecture, Algorithms, Performance[J]. IEEE TRANS on NETWORKING, 2006, 14(6): 1246-1259.

[2] Wang JianTao, David X Wei, Low S H. Modeling and Stability of FAST TCP[C]//USA: Proc. IEEE INFOCOM Miami, FL, 2005: 938-948.

[3] Krister Jacobsson, Lachlan L Steven H Low. An Improved Link Model for Window Flow Control and Its Application to FAST TCP[J]. IEEE Transactions on Automatic Control, 2009, 54(3): 551-564.

[4] 黄小猛, 林闯, 任丰源. 高速传输协议研究进展[J]. 计算机学报, 2006, 29(11): 111-120.

[5] 宋丽华, 陈鸣, 张睿. 一种基于测量的 FAST TCP 改进方案[J]. 北京邮电大学学报, 2005, 28(4): 232.

[6] Liansheng Tan, Cao Yuan, Mosh Z. FAST TCP: Fairness and Queueing Issues[J]. IEEE Communications Letters, 2005, 9(8): 762-764.

[7] Low S H, Peterson L, Wang L. Understanding Vegas: a duality model[J]. ACM, 2002, 49(2): 207-235.

[8] Tony Cui, Lachlan, Liansheng Tan. Improving the Fairness of FAST TCP to New Flows[J]. IEEE Communications Letters, 2006, 10(5): 414-416.

[9] Migule R, Sergio H. Achieving Fair Network Equilibria with Delay-based Congestion Control Algorithms[J]. IEEE Communications Letters, 2008, 12(7): 535-53.

[10] 龙承念, 杨会龙, 关新平. EHSTCP: 改进的高速 TCP 算法[J]. 计算机学报, 2008, 31(3): 143-152.

[11] 王建新, 陈建二. 高带宽延时网络中一种协同时拥塞控制协议[J]. 软件学报, 2008, 19(1): 125-135.

A Cooperant Improved FAST TCP Fairness Algorithm

Chen Xiao-long^{1,2}, Zhang Yun¹, Liu Zhi¹

(1. Faculty of Automation, Guangdong University of Technology, Guangzhou 51006, China;

2. Faculty of Computer Science, Electronics and Information, Maoming College, Maoming 525000, China)

Abstract: A cooperant improved algorithm is proposed, aiming at the problem of unfairness caused by the inaccurate estimation of the round trip propagation delay at the time of simultaneous connections of multiple FAST TCP. The synchronization back-off clock and the minimum back-off factor were determined via the full use of the locally available information cooperantly. Without direct communication of the FAST connections, the true propagation delay was accurately and quickly estimated, and the fairness was achieved for the buffer queue to empty occasionally. The NS-2 simulation results demonstrate that compared with other related algorithms, this improved algorithm can achieve fairness and higher stability.

Key words: fairness; propagation delay; local information; cooperant; synchronization back-off clock