利用时延特性的模糊TCP拥塞控制算法

易发胜，赵继东

(电子科技大学计算机科学与工程学院 成都 610054)

【摘要】提出了一种基于时延的模糊控制TCP拥塞控制算法(DFCC)， 采用当前RTT变化和历史变化相结合的方法，通过一个模糊控制器调节CWND，从而对网络拥塞情况有了更好的估计和反应，有效解决了突发拥塞降低吞吐量的问题。仿真试验表明，改进的算法更加准确地监测到网络拥塞，提高了网络的吞吐量。

关 键 词 拥塞控制； 模糊控制； 网络时延； TCP Vegas

中图分类号 TP393 **文献标识码** A doi:10.3969/j.issn.1001-0548.2010.02.023

Fuzzy TCP Congestion Control Algorithm Using Delay

YI Fa-sheng and ZHAO Ji-dong

(School of Computer Science and Engineering, University of Electronic Science and Technology of China Chengdu 610054)

Abstract A fuzzy TCP congestion control algorithm is proposed. The algorithm achieves better evaluation on network congestion by the way of combining the current and historical RTT changes which are used to adjust congestion window (CWND) with fuzzy controller, so that the problem of low throughput under abrupt congestion situations is solved effectively. The performance of the algorithm has been tested and evaluated on NS simulator and the simulation results demonstrate that the algorithm inspects network congestion more exactly and accordingly increases network throughput.

Key words congestion control; fuzzy control; network delay; TCP Vegas

因特网在主机端利用TCP协议处理拥塞问题。TCP通过检测每个发送出去的数据报文的应答了解网络的拥塞情况。发送方每发送一个数据报文，启动定时器，如果在超时限之前没有收到确认(ACK)，发送方认为丢包，网络发生拥塞，则减小拥塞窗口大小(CWND)，降低发送速度，从而缓解拥塞。虽然TCP的拥塞控制机制有效地避免了网络拥塞崩溃，但是多项研究表明TCP的拥塞控制策略趋于保守，限制了网络的吞吐量。为了提高TCP协议的吞吐量，有针对TCP拥塞控制的大量研究，也出现了各种TCP版本，如Tahoe、Reno和Vegas等算法。在各种TCP的实现方案中，TCP Vegas算法[1]达到了更好的吞吐量。

TCP Vegas算法重点改进了“拥塞避免”阶段的CWND调整算法。它通过观测TCP连接中往返时延(RTT)的变化，而不是用分组的丢失探测网络可能出现的拥塞，并相应地调节CWND的大小，是一种基于端到端延迟的拥塞避免(delay-based congestion avoidance，DCA)算法。算法认为RTT的变化是由网络的拥塞造成的，当RTT增加时，表示网络拥塞，按照一定的百分比降低发送方CWND大小；而RTT减小时，表示网络负载减轻，按照一定的比例增加发送方CWND大小。与目前广泛使用的TCP Reno相比，TCP Vegas可增加40%～70%的吞吐量[1]。但是实际的RTT测量值太粗糙，往往导致TCP Vegas不适当地调节拥塞窗口大小，从而导致TCP吞吐量降低。利用RTT指示拥塞反而会降低TCP吞吐量。大量测试分析发现某些情况下仅有7%～18%的报文丢失与RTT增加有关[2]。在此情况下，如果利用TCP Vegas算法避免拥塞，会降低TCP吞吐量达7%～58%。实际上，文献[2]指出单纯利用一个RTT作为网络是否拥塞的指示太粗糙，既不能及时反映高速网络的突发拥塞，也不能精确判断网络拥塞情况。

为进一步提高TCP Vegas吞吐量，近年来有大量的研究工作[3-10]。TCP New Vegas[4]研究了TCP Vegas在长延时链路的性能下降问题；TCP Vegas-A[5]研究了TCP Vegas适用于卫星链路的RTT波动情况；而文献[6]根据延迟抖动和丢失率提出了使用模糊控制保

收稿日期： 2008 – 09 – 12；修回日期：2009 – 02 – 07
基金项目：国家863高技术研究基金(2007AA01Z443)
作者简介：易发胜(1968 –)，男，博士，主要从事计算机网络和体系结构方面的研究.

持发送速率稳定。这些方法虽然提高了某些情况下TCP Vegas的性能，但是并没有通用性，在很多情况下还是不能指示真正的拥塞状况。为了提高辨别拥塞的能力，一种方法是通过时间戳选项，得到比较精确的RTT时间，并消除回程ACK的延迟影响[7]，但由于TCP时间戳选项并没有考虑延迟ACK的影响以及不同主机的时钟差异，结果并不准确；另一种方法是通过接收方和路由器合作，提供更加精确的拥塞指示和控制，提高TCP Vegas吞吐量和带宽公平利用[8]，但该方法太复杂，还需要路由器协作处理。此外，一些算法修改AIMD的工作方式[9-10]，拥塞时改变CWND增加或者减少的速度，使TCP吞吐量有了更好的改进。

本文针对RTT作为拥塞控制指示进行了认真分析，认为单纯利用一个RTT作为拥塞指示具有两个主要缺陷：(1) 没有考虑RTT中非数据拥塞的影响。实际上，在RTT的组成中，ACK的回程拥塞以及延迟ACK也是RTT产生变化的重要原因，并影响了对拥塞的指示。(2) 高速网络突发拥塞的出现时间很短，传统的基于延迟的算法不能及时作出正确的判断。基于这些分析，本文首先通过改进TCP时间戳算法消除RTT中影响拥塞指示的因素，然后运用模糊理论对RTT的变化这一模糊问题进行描述，并采用当前RTT变化和历史变化相结合的方法，通过一个模糊控制器调节CWND，从而对网络拥塞情况有更好的估计和反应，有效解决了突发拥塞降低吞吐量的问题，大大提高了TCP在拥塞避免阶段的吞吐量。仿真试验表明，该算法显著提高了TCP吞吐量，改善了网络性能。

1　拥塞避免算法的模糊控制模型

TCP Vegas在拥塞避免阶段并不持续增加拥塞窗口，而是估计网络中的额外数据量合理控制CWND。首先它记录最小的往返延迟BaseRTT=min(RTT$_i$)，然后计算期望的发送速率Expected=CWND/BaseRTT，当收到一个新的RTT采样值NewRTT时，计算实际的传输速率Actual=CWND/NewRTT。比较Expected和Actual，得到速率差值Diff=Expected–Actual。根据定义，Diff应大于0。如果Diff<0，表示BaseRTT太大，需要调整为NewRTT。然后定义两个阈值α、β。如果Diff<α，Vegas线性增加CWND；如果Diff > β，Vegas线性减小CWND，而α<Diff<β 时CWND保持不变。

实际上，Diff=Expected–Actual=(NewRTT–BaseRTT)∗CWND/(BaseRTT∗NewRTT)，即：

$$\mathrm{Diff}=\Delta\mathrm{RTT}*\mathrm{CWND}/(\mathrm{BaseRTT}*\mathrm{NewRTT}) \quad (1)$$

式中　ΔRTT是最小的RTT和当前RTT的差值，是影响DCA算法效果的重要因素。影响RTT的因素主要有：数据传播时间T_p(分组在所有链路的信号传播时间)、处理时间T_d(分组在各个节点处理时间)、传输时间T_t(分组从网络节点送到链路时间)和排队时间T_q(分组在网络节点排队等候时间)，其中只有T_q是网络拥塞的有效指示。

真正指示数据传输拥塞的，应该是数据传输方向的排队时间，而ACK在传输时的排队时间可能对网络拥塞状况给出错误的指示[7]。为了减少单独地确认报文数量，TCP普遍采用延迟ACK的策略，便于捎带确认，但导致有些ACK可能会延迟0～500 ms，也会对整个RTT产生较大的影响。延迟ACK对TCP拥塞控制具有很大影响[11]，同样也严重影响TCP Vegas的吞吐量。

在高速网络中，突发拥塞的时间往往持续很短。这是因为网络波动导致的缓冲区排队现象会由于高速链路的迅速传输，在很短时间内结束。表现为偶尔会收到具有很大延迟的RTT，如图1所示。这时如果按照Vegas算法，线性运算大量减少CWND，以避免拥塞的发生，从而降低TCP的吞吐量。

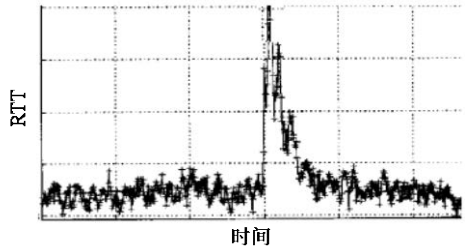

图1　突发拥塞时的RTT示意图

因此，为了解决目前TCP Vegas在ACK回程拥塞、延迟ACK的干扰和大带宽延迟积中突发拥塞等原因导致的性能下降问题，宜采用一种模糊控制器进行拥塞控制。模糊控制由于其非线性控制的特点，非常适合处理不精确信息，且便于处理其他影响TCP吞吐量的因素[12]。如图2所示，拥塞控制模型包含一个修正器和模糊控制器，修正器根据当前和历史的延迟为模糊控制器提供拥塞控制所需要的准确信息，模糊控制器则综合各种信息，对拥塞窗口作出适当调整。

模糊控制器的输入采用3个输入量，它们分别是最大的排队延迟差值ΔT_{fq}、当前的排队延迟变化量ΔT_{now}以及最近的排队延迟变化量ΔT_{old}。本文将详细

讨论这些值的计算以及用于拥塞控制的方法。

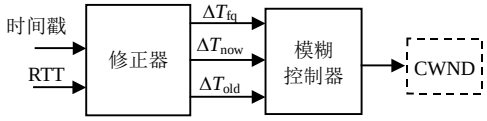

图2 一种模糊TCP拥塞控制模型

2 基于模糊控制的拥塞控制算法

2.1 修正器的算法

通过前面的分析知道，在TCP的RTT中，能够有效体现网络拥塞情况的组成部分是数据传输的排队延迟时间T_{fq}。但是在整个RTT的组成中，T_{fq}只占其中的一部分。在基于延迟的拥塞避免算法中，必须克服延迟ACK和ACK回程拥塞带来的影响，才能进一步提高预测拥塞的精确度。如前所述，一个RTT时间表示如下：

$$RTT=T_p+T_t+T_d+T_q \tag{2}$$

在式(2)中，由于TCP数据流在一个相对稳定的时间内保持不变的路径[13]，因此T_t、T_p可以认为保持相对不变；根据前面的分析，$T_d=T_d'+T_{delay}$，其中T_{delay}表示延迟ACK可能耽误的时间；而$T_q=T_{fq}+T_{bq}$，其中T_{fq}表示数据传输的排队延迟；T_{bq}表示确认传输过程中的排队时间。将上述值代入式(2)得：

$$RTT=T_p+T_t+T_d'+T_{delay}+T_{fq}+T_{bq} \tag{3}$$

式中 $T_p+T_t+T_d'$相对固定，其他的几项都会变化，但是真正能够让TCP Vegas预测网络拥塞状况的是T_{fq}。因此要能够准确预测网络的拥塞状况，必须剔除T_{delay}和T_{bq}的干扰。为了达到这个目的，根据文献[7]的思路，改进TCP的时间戳选项，可以获得更加精确的T_{fq}。

文献[7]使用RFC1323定义的时间戳选项只是定义了发送时间以及要确认的时间戳，虽然可以使发送方得到比较准确的RTT，但由于延迟ACK的存在，以及不同主机时钟的差异，并不能很好估计数据传输方向的时延。因此，为提高T_{fq}的估计值，对TCP的时间戳选项进行了扩展，如图3所示。

kind	type	T_{Sval}	T_{Secr}	T_{Secv}
1B	1B	4B	4B	4B

图3 扩展的TCP时间戳选项

在标准时间戳选项上增加了一个T_{Secv}项，该选项表示当前确认的数据报文接收到的时间值。同时规定，携带ACK的时间戳选项总是回应最近收到的数据段的时间戳值T_{Secr}和T_{Secv}。发送方收到确认时，根据式(4)可以计算：

$$T_{fp}+T_{ft}+T_{fd}+T_{fq}=T_{Secv}-T_{Secr}-\Delta T_{clock} \tag{4}$$

式中 ΔT_{clock}是收发双方主机的时钟差值，T_{fp}、T_{ft}、T_{fd}分别表示数据传输方向的传输时间、传播时间和处理时间，都是相对固定的，可以得到：

$$T_{fq}=T_{Secv}-T_{Secr}-(\Delta T_{clock}+T_{fp}+T_{ft}+T_{fd}) \tag{5}$$

即：

$$T_{fq}=T_{Secv}-T_{Secr}-\Delta T \tag{6}$$

式中 ΔT表示固定的所有非排队延迟时间。修改TCP Vegas，利用ΔRTT调节CWND的方法，用ΔT_{fq}替换ΔRTT调节CWND，将会大大提高拥塞控制的精度。对式(1)进行修改得到：

$$Diff=\Delta T_{fq}*CWND/(BaseRTT*NewRTT) \tag{7}$$

式中 ΔT_{fq}表示当前收到的确认中的T_{fq}与最小的T_{fq}之差，即：

$$\Delta T_{fq}=(T_{Secv}-T_{Secr}-\Delta T)_{cur}-(T_{Secv}-T_{Secr}-\Delta T)_{min}=$$
$$(T_{Secv}-T_{Secr})_{cur}-(T_{Secv}-T_{Secr})_{min}=\Delta TS_{cur}-\Delta TS_{min} \tag{8}$$

式中 ΔTS表示时间戳中$T_{Secv}-T_{Secr}$的值；而ΔTS_{cur}和ΔTS_{min}分别表示当前报文时间的ΔTS和最小RTT时的ΔTS。ΔT_{fq}很容易根据扩展的时间戳计算出来。相比ΔRTT，ΔT_{fq}具有更加准确指示拥塞的能力，将极大改善TCP Vegas的效率。同时，为了对近期的拥塞情况有所反映，修正器使用连续两次的T_{fq}值，即T_{fq_1}、T_{fq_2}表示当前网络拥塞的变化情况，用ΔT_{now}表示为：

$$\Delta T_{now}=T_{fq_2}-T_{fq_1}=\Delta TS_2-\Delta TS_1 \tag{9}$$

下标1和2分别表示最近两次的报文序号，其中2是当前报文，1是最近一次报文。此外，修正器将保留最近计算的ΔT_{now}。根据RTT的大小，决定保存多少个ΔT_{now}，每次将计算：

$$\Delta T_{old}=\sum_{i=1}^{n}\Delta T_{now_i} \tag{10}$$

通过每次计算式(10)，修正器能得到近期的数据排队时间变化趋势值ΔT_{old}，连同当前计算的ΔT_{now}送给模糊控制器进行处理。此外，修正器还记录每次得到的ΔTS，如果有最小的ΔTS则及时修改ΔTS_{min}，如果相当长时间ΔTS偏离最小值，需要重新设定ΔTS_{min}，以对网络路径变化作出反应。

2.2 基于延迟时间差值的拥塞评判

由于拥塞这一概念本身的模糊性，不可能用精确的排队时间描述网络的拥塞状态。虽然用修正器处理可得到比较准确的数据排队时间差值ΔT_{fq}，但是仍然只能说这些数据报文处于某种程度的拥塞。如某一时刻收到RTT后，ΔT_{fq}和ΔT_{now}值可能很大，说明当前网络存在拥塞的可能，但是ΔT_{old}很小甚至为负数，则表示当前收到的很可能是突发拥塞现象。

因此，引入模糊理论描述这一模糊性过程，充分利用模糊理论在处理非线性和不确定性问题上的优越性，以达到优化的拥塞控制性能和提高整个网络吞吐量的目的。

ΔT_{fq}、ΔT_{now}和ΔT_{old}表示值的范围不一样，为了简单处理，首先对这三个参数归一化处理为闭区间[0,1]上的时延差值比率。将TCP计算的timeout值作为最大的RTT，然后计算：

$$\Delta RTT_{max}=timeout-BaseRTT \tag{11}$$

设ΔT_{fq}、ΔT_{now}和ΔT_{old}归一化处理结果分别是$\Delta T'_{fq}$、$\Delta T'_{now}$、$\Delta T'_{old}$，则：

$$\Delta T'_{fq}=\Delta T_{fq}/\Delta RTT_{max} \tag{12}$$

$$\Delta T'_{now}=(\Delta RTT_{max}+\Delta T_{now})/2\Delta RTT_{max} \tag{13}$$

$$\Delta T'_{old}=(\Delta RTT_{max}+\Delta T_{old})/2\Delta RTT_{max} \tag{14}$$

时延差值比率被划分为3个语言变量{空闲, 正常, 拥塞}。划分具体意义为空闲(简写为F)，表示网络拥塞程度轻，比较空闲；正常(简写为N)，表示网络处于轻度拥塞状态；拥塞(简写为C)，表示处于拥塞状态。根据Zadeh对模糊子集的定义，定义论域U为时延差值比率(当前时延差值在最大时延差值的比例)。取U到闭区间[0,1]上的3个映射μF：$U\rightarrow[0,1]$，$u\rightarrow\mu F(u)$；μN：$U\rightarrow[0,1]$，$u\rightarrow\mu N(u)$；μC：$U\rightarrow[0,1]$，$u\rightarrow\mu C(u)$确定出U的3个模糊子集F、N、C。$\mu F(u)$、$\mu N(u)$、$\mu C(u)$分别称为对于F、N、C的隶属函数。同时，用决策集$V=\{F, N, C\}$描述整个网络的当前拥塞状态。

隶属函数的选择是进行模糊综合分析的关键因素，常用隶属函数形式有三角函数、梯形函数、样条函数等。根据拥塞程度的等级描述各个延迟时间差值的语言模糊集个数，每个语言模糊集对应着不同的拥塞等级和不同的隶属函数。用三角函数确定时延差值比率论域U上的3个隶属函数$\mu F(u)$、$\mu N(u)$、$\mu C(u)$，如图4所示。

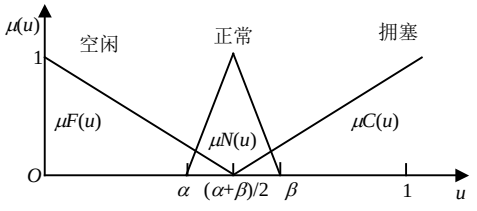

图4　隶属函数

图中，正常区间由α和β确定，其作用类似于TCP Vegas的两个阈值α、β，以确定系统保持稳定时的网络状态。

2.3　基于三个时延差值比率的拥塞处理

为了得到比较准确的当前网络拥塞状况，需要对3个时延差值比率进行综合评判。用因素集U表示3个模糊变量$U=\{u_1,u_2,u_3\}$，其中，u_1、u_2和u_3分别对应于3个时延差值比率$\Delta T'_{fq}$、$\Delta T'_{now}$和$\Delta T'_{old}$，从因素集U到决策集V存在模糊关系，用模糊矩阵$\boldsymbol{R}$表示，记为：

$$\boldsymbol{R}=\begin{pmatrix}\mu F(u_1) & \mu N(u_1) & \mu C(u_1)\\ \mu F(u_2) & \mu N(u_2) & \mu C(u_2)\\ \mu F(u_3) & \mu N(u_3) & \mu C(u_3)\end{pmatrix}$$

考虑到各个时延差值比率在综合评判中的重要程度，权重分配为$w=(w_1, w_2, w_3)$，权系数w_k表示对第k个时延差值比率的重视程度，且满足$0\leqslant w_k\leqslant 1$，$k=1,2,3$；$w_1+w_2+w_3=1$。很明显当$w_1=1$，$w_2=w_3=0$时，其效果接近于TCP Vegas。

设综合评判结果为$p=(p_1,p_2,p_3)$，则它们有如下关系[14]：

$$p=w\circ B \tag{15}$$

式中 $\circ$为常用模糊矩阵Max-min合成运算，p_j表示综合考虑3个时延差值比率后，当前网络拥塞状况属于V中不同状态的隶属程度，比较$p_i(i=1,2,3)$，存在一个k，使得：

$$p_k=\max_j\{p_j\} \tag{16}$$

根据最大隶属度原理，可以判断网络处于p_k所确定的状态，从而智能控制CWND。模糊拥塞处理规则如表1所示。

表1　模糊拥塞处理规则

状态	处理方法
空闲	线性增加 CWND
正常	CWND保持不变
拥塞	线性减少 CWND

2.4　算法处理步骤

(1) 对接收到的数据报文进行修正处理，根据式(8)、(9)和(10)分别计算ΔT_{fq}、ΔT_{now}、ΔT_{old}。

(2) 根据式(12)～式(14)对ΔT_{fq}、ΔT_{now}和ΔT_{old}进行归一化处理，得到$\Delta T'_{fq}$、$\Delta T'_{now}$和$\Delta T'_{old}$。

(3) 根据归一化处理结果填写矩阵$\boldsymbol{R}$。

(4) 根据确定权重w，由式(15)计算p。

(5) 由式(16)计算p_k得到当前网络状态。

(6) 根据表1计算最新的CWND值。

(7) 结束。

3　性能仿真分析

NS2是比较成熟的网络仿真环境，本文对NS2进行进一步扩展，实现了本文所提出的DFCC算法，并与TCP Vegas算法进行了性能比较。仿真采用的网

络拓扑结构如图5所示。

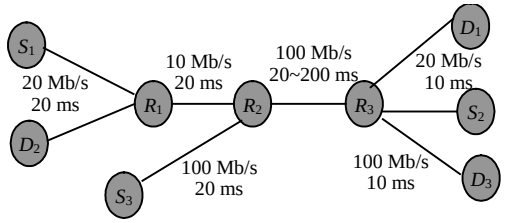

图5 网络拓扑结构

R_1、R_2和R_3作为路由器，其他节点作为主机。其中，D_1端接收S_1端的数据，D_2端接收S_2端的数据。S_3和D_3作为背景流量在R_2产生拥塞。各链路都是双向的，其具体带宽和延迟可根据仿真要求进行调整。定义各链路数据传输与接收的成功率为100%，以免影响各节点算法性能评价。

仿真时，各路由器使用FIFO和DropTail队列管理算法，缓冲区最大队列长度为100 KB大小，接收方通知窗口大小设为20个报文段。最大数据包长度设为1 400 B。S_1～D_1运行TCP，分别采用纯TCP Vegas和拥塞避免阶段DFCC(下面称DFCC TCP)比较DFCC算法对拥塞避免的改进。仿真方案如下：

(1) 启动延迟ACK，比较传输效果。

(2) 比较有ACK拥塞时的传输效果。

(3) 比较网络有突发拥塞时的传输效果。

(4) 使用不同权重系数分析性能影响。

在性能评价过程中，默认情况下设定各个模糊量隶属函数的参数α、β分别为0.4和0.6，权重分配为w=(0.4,0.3,0.3)，检查不同条件下DFCC TCP和纯TCP Vegas的吞吐量。

3.1 具有延迟ACK时的传输效果对比

为了说明DFCC对延迟ACK传输延迟的影响具有比较好的适应性，在S_1和D_1之间构造一个8 Mb/s的Telnet流，设置D_1采用延迟ACK策略。为了进行对比，对DFCC TCP和TCP Vegas的所有其他设置都一样。每次仿真60 s，在不同的情况下对比检查S_1的吞吐量，结果如图6所示。

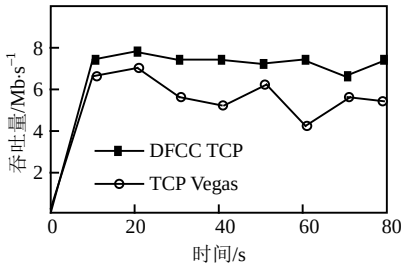

图6 延迟ACK对吞吐量的影响

图6显示在具有延迟ACK时，DFCC TCP具有明显比TCP Vegas高的吞吐量。这是因为DFCC TCP采用扩展的时间戳选项，完全免除了延迟ACK的干扰。TCP Vegas的吞吐量变化幅度大，说明延迟ACK具有不稳定性和随机性。

3.2 具有ACK回程拥塞时的性能比较

由于网络TCP流量的不对称性，有时某个数据的传输方向没有拥塞，而ACK回程通道上却出现拥塞。在前面测试的基础上，让S_2发送ON-OFF类型的VBR流，设定在ON期间S_2发送20 Mb/s、平均9 Mb/s的流量。从20 s处开始S_2的VBR流量，测量S_1的TCP吞吐量，如图7所示。

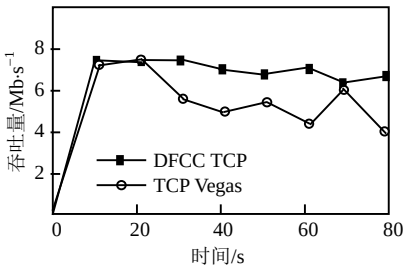

图7 具有ACK回程拥塞时的比较

从图7可以明显看出，当ACK回程拥塞开始出现时，TCP Vegas的吞吐量明显减少，而DFCC TCP的吞吐量减少较小。这是因为采用可扩展时间戳的DFCC算法很好地消除了回程ACK延迟的干扰。当然，由于某些ACK的丢失，DFCC TCP的吞吐量也有所降低。

3.3 具有突发拥塞时的性能比较

为了说明DFCC对突发拥塞的影响具有比较好的适应性，在第一个测试的基础上，S_3发送背景流量为ON-OFF的流量到D_3，平均每秒钟有连续50 ms的100 Mb/s的发送。为了突出突发拥塞的效果，设定R_2的发送缓冲区为50 KB。突发拥塞流量从20 s开始注入，R_2～R_3的链路时延改为200 ms，检查R_3的吞吐量，结果如图8所示。

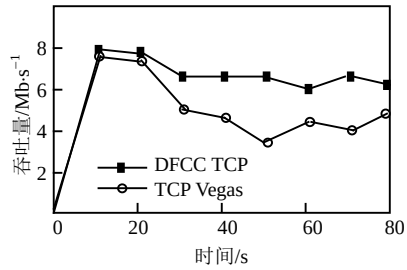

图8 突发拥塞时的吞吐量对比

从图8可以看出，DFCC TCP明显提高了整个系统的吞吐量。在有突发拥塞时，TCP Vegas很不稳定，但是DFCC TCP的吞吐量摆动幅度不大，主要是由于DFCC TCP根据历史时延差值平滑了突发拥塞带来的吞吐量大幅度变化。

3.4 权重系数对DFCC的影响

模糊控制器中的权重系数w决定了何种因素对网络拥塞估计的影响最大。在前面的仿真实验中，都是假定使用比较均衡的W_0=(0.4,0.3,0.3)。为了估计不同权重系数的影响，分别设定3个不同的W值W_1=[2,1,1]/4，W_2=[1,2,1]/4，W_3=[1,1,2]/4，重新对DFCC TCP进行具有突发拥塞时的仿真试验。环境设置与上述试验设置完全类似，比较不同权重系数对S_1的DFCC TCP吞吐量的影响，如图9所示。

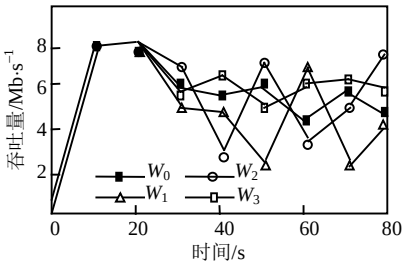

图9　不同权重系数下突发拥塞的吞吐量

根据图9的对比可发现，对于W_1和W_2两种权重系数，吞吐量的波动比较大，而W_3的波动比较小。这主要是由于W_3对ΔT_{old}的权重大，当前的CWND的大小同历史拥塞记录关系大，而W_1和W_2两种情况及时体现了当前的拥塞变化情况。另外，W_3具有最好的总吞吐量，W_2次之，W_1最差，显示在突发拥塞的情况下，需要根据历史信息调节拥塞窗口才能够提高总吞吐量。在突发拥塞的情况下，由于拥塞控制往往不能及时反映网络的当前状况，所以有必要提高ΔT_{old}的权重增加整体吞吐量；但同时也需要足够的ΔT_{fq}和ΔT_{now}的权重，让系统具有足够快的反应能力。通过对比试验显示，W_0具有反应灵敏且相对较高的吞吐量和较好的总体性能。

上述仿真中，无论哪种权重系数，相对于TCP Vegas，DFCC TCP都极大改善了TCP的吞吐量，相比而言，前面仿真使用的(0.4,0.3,0.3)权重系数具有比较好的综合效果。

4 结 束 语

在DFCC算法中考虑了确认排队迟延和延迟ACK的影响，采用改进的时间戳选项，结合当前和历史的拥塞控制信息，显著提高了预测网络拥塞的准确性。仿真试验表明，在具有ACK延迟比重大、有确认报文拥塞和具有突发拥塞的传输流中，本文算法表现出很大的性能改进，可明显提高网络的总吞吐量，增加IP网络的资源利用率。

本文的研究工作得到了电子科技大学青年科学基金(JX05031)的资助，在此表示感谢。

参 考 文 献

[1] LAWRENCE S, BRAKMO, L. TCP Vegas: end to end congestion avoidance on a global internet[J]. IEEE Journal on Selected Areas in Communications, 1995, 13(8): 1465-1480.

[2] MARTIN J, NILSSON A, RHEE I. Delay-based congestion avoidance for TCP[J]. IEEE/ACM Transactions on Networking, 2003, 11(3): 356-369.

[3] KYUNGSUP K, Chong-Ho Choi. Queue delay estimation and its application to TCP Vegas[J]. Computer Networks, 2003, 43: 619-631.

[4] SING J, SOH B. TCP new vegas: improving the performance of TCP vegas over high latency links[C]// Network Computing and Applications, Fourth IEEE International Symposium. Cambridge: IEEE Press, 2005: 73-82.

[5] SRIJITH K N, JACOBL L, ANANDA A L. TCP vegas-a: improving the performance of TCP vegas[J]. Computer Communications, 2005, 28(4): 429-440.

[6] YAN Li, QIU Bin. An intelligent algorithm for improving QOS in a delay-based end-to-end congestion avoidance schem[C]//Information Technology and Applications, ICITA. Sydney: IEEE Press, 2005: 653- 658.

[7] CHAN Yi-cheng, CHAN Chia-tai, CHEN Yaw-chung. An enhanced congestion avoidance mechanism for TCP vegas[J]. IEEE Communications Letters, 2003, 7(7): 343-345.

[8] YONG Xia, HARRISON D, KALYANARAMAN S et al. Accumulation-based congestion control[J]. IEEE/ACM Transactions on Networking, 2005, 13(1): 69-80.

[9] LEITH D J, SHORTEN R N, MCCULLAGH G, et al. Making available base-RTT for use in congestion control applications[J]. IEEE Communications Letters, 2008, 12(6): 429-431.

[10] SING J, BEN S. Improving congestion window growth in large bandwidth delay product networks[C]//15th IEEE International Conference on Network, Adelaide: IEEE Press, 2007: 382-387.

[11] PAXSON V, ALLMAN M. Known TCP implentation problems[S]. RFC 2525, 1999.

[12] OLIVEIRA R D, BRAUN T. A delay-based approach using fuzzy logic to improve TCP error detection in ad hoc networks[C]//IEEE Wireless Communications and Networking Conference. Atlanta: IEEE Press, 2004: 1666-1671.

[13] PAXSON V. Measurements and analysis of end-to-end internet dynamics[D]. UC: Berkeley, University, 1996.

[14] ZIMMERMANN H J. Fuzzy set theory and its application[M]. New York: Springer Publishing , 2001.

编 辑 蒋 晓