ITC19/ Performance Challenges for Efficient Next Generation Networks 1375-1384
LIANG X.J. and XIN Z.H.(Editors)
V.B. IVERSEN and KUO G.S.(Editors)
Beijing University of Posts and Telecommunications Press

Fairness Comparison of FAST TCP and TCP Vegas

Lachlan L. H. Andrew[1], Liansheng Tan[2], Tony Cui[1], and Moshe Zukerman[1]

[1] ARC Special Research Centre for Ultra-Broadband Information Networks (CUBIN),*
an affiliated programme of the National ICT Australia
Department of Electrical and Electronic Engineering,
The University of Melbourne, Victoria 3010, Australia
(e-mail: l.andrew,t.cui,mzu@unimelb.edu.au)

[2] Computer Science Department of Central China Normal University,
Wuhan 430079, PR. China
(e-mail: l.tan@mail.ccnu.edu.cn)

Abstract. This paper compares the equilibrium properties of FAST TCP and TCP Vegas. Although the two have the same equilibrium point when all sources know their true propagation delays, FAST is fairer when there are estimation errors. The performance of Vegas approaches that of FAST when the queueing delay is very much less than the propagation delay.

1 Introduction

Given the importance of controlling congestion and stability of the Internet, there have been many proposals aiming to improve the well known TCP Reno protocol [4]. Two such popular proposals that have received significant attention in recent years are TCP Vegas [1, 2] and FAST TCP [6–8]. This paper compares these two TCP versions from the point of view of fairness in equilibrium.

TCP Vegas and FAST TCP aim to improve throughput and fairness over their predecessors, the most popular of which is TCP Reno, by using queueing delay as a congestion signal because queueing delay provides a finer measure of congestion and scales more naturally with network capacity than packet loss probability does [7].

Because both FAST TCP and TCP Vegas use queueing delay as a congestion signal, their window updating algorithms rely on knowing the propagation delay, which is estimated by a measure called *baseRTT*. This measure is defined as the minimum round-trip time (RTT) observed so far. Because sometimes the routers' queues never become empty, the actual propagation delay may be inaccurately estimated by *baseRTT* which results in unfairness [7, 11], and excessive variations of routers' queues.

Considering steady state (equilibrium) conditions, the aim of this paper is to analyse and compare the fairness of FAST TCP versus TCP Vegas. However, it is important to clarify at the

* This work was supported by the Australian Research Council

outset a certain confusion that one may encounter considering the definition of TCP Vegas. A certain ambiguity exists in this definition and there are two interpretations of TCP Vegas, one of which makes TCP Vegas very similar to FAST TCP. In particular, TCP Vegas uses queueing delay as the congestion signal which is different from TCP Reno. It is parameterised by a rate α. As pointed out in [10], the original description of Vegas [2] was ambiguous about whether α measures the rate per second, or per round trip time. In this paper, the term "Vegas" will be used for the form analysed in [10] (specifically, Section 4.2 of [10]). This matches the prose description in [2], in which α is the rate per second. Where ambiguity may arise, it will be called "the prose version of Vegas". The term "FAST" will be used for the the form adopted by FAST TCP [7], which matches the implementation of Vegas, in which α is the rate per round trip time. As we consider equilibrium conditions, we ignore the slow start phase for both Vegas and FAST TCP in this paper.

The structure of this paper is as follows. In Section II, we define the notation for this paper, in Sections III, IV and V, we introduce the TCP Vegas and FAST TCP algorithms in detail. In Section VI, we analyse the fairness of the two TCP protocols and in Section VII, we use simulation to verify the analysis.

2 Notation

Let $L(i)$ denote the set of links used by flow i. Let d_i [seconds] be the true propagation delay of flow i, let $\hat{d}_i = d_i + \delta_i$ be the estimated propagation delay, and let $D_i(t) = d_i + q_i(t)$ be the round trip time of flow i, including queueing delay of $q_i(t)$. Let $w_i(t)$ [packets] and $x_i(t)$ [packets/s] be the window size and rate for flow i, which are related by

$$w_i(t) = x_i(t)D_i(t). \tag{1}$$

Let c_l [packets/s] be the capacity of link l, and $b_l(t)$ [packets] be the backlog at link l.

Quantities without explicit time dependence are either constants or equilibrium values; for example, b_l is the equilibrium backlog at link l.

Both FAST and Vegas use a parameter called α, although the meaning of each is subtly different, as alluded to in the introduction. Flows using FAST and Vegas aim to keep a fixed number of packets in queues throughout the network. Under FAST, flow i aims to keep α_i packets, while under Vegas it aims to keep $\alpha_i d_i$. To avoid confusion, the alpha values for Vegas will be denoted α^+.

Comparisons in this paper will use the following scenario, called Persistent Congestion in [10]. All flows share a single bottleneck link of capacity c [seconds], have equal α [packets] (or α^+ [packets/s]), and have equal propagation delays, d [seconds]. Flows arrive consecutively, spaced far enough apart for the system to reach equilibrium between arrivals, and keep transmitting greedily and persistently. When the ith flow arrives, it causes the queue size at the bottleneck link to increase by $B(i)$ [packets]. If d were known exactly, then $B(i)$ would be α under FAST, or $\alpha^+ d$ under Vegas. However, the estimate $\hat{d}$ will be assumed to be the RTT seen when the flow first arrives, given by $d(i) = d + p(i-1)$ [seconds], and $B(i)$ will consequently be larger. Here $p(i) = \sum_{j=1}^{i} B(j)/c$ [seconds] is the total queueing delay after the arrival of flow i.

To distinguish between the equilibrium of FAST and of Vegas, quantities pertaining to Vegas will have a superscript $+$.

By arguments analogous to [10], it can be shown that the FAST equilibrium maximises the sum of flows' utilities, where the utilities are now given by

$$U_i(x_i) = \alpha_i \log x_i + \delta_i x_i. \tag{10}$$

The core argument is to show that the derivative of the utility of flow i is the sum of suitable Lagrange multipliers, p_l, corresponding to the price of each link, evaluated at the equilibrium. That is,

$$U_i'(x_i) \equiv \frac{\alpha_i}{x_i} + \delta_i = \sum_{l \in L(i)} p_l. \tag{11}$$

That can be seen as follows. The number of packets queued by flow i at link l is $b_l x_i / c_l$. The total number of packets queued by flow i in equilibrium is

$$\sum_{l \in L(i)} \frac{b_l x_i}{c_l}. \tag{12}$$

Since the total number of packets from flow i in flight, w_i, is equal to the sum of those in propagation, $x_i d_i$, and those queued, in equilibrium

$$w_i - x_i d_i = \sum_{l \in L(i)} \frac{b_l x_i}{c_l}. \tag{13}$$

Combining (1), (9) and (13) gives

$$\alpha_i = w_i \left(1 - \frac{\hat{d}_i}{D_i}\right) = w_i - x_i(d_i + \delta_i) = \sum_{l \in L(i)} \frac{b_l x_i}{c_l} - \delta_i x_i. \tag{14}$$

Rearranging and setting $p_l = b_l / c_l$ yields (11), as required.

5 FAST under Persistent Congestion

On the surface, it seems that (4) and (10) are equivalent under the substitution $\alpha_i = \alpha_i^+ \hat{d}_i$. However, it turns out to be very significant that the number of packets that Vegas attempts to maintain in the queue, $\alpha_i^+ \hat{d}_i$, depends on the error in the estimate of the propagation delay. To see that, consider the equations for persistent congestion analogous to (5) and (7), which will now be derived, following [11].

The equilibrium rates satisfy

$$x_1 + x_2 + \ldots + x_i = c, \tag{15}$$

and

$$x_j = \alpha / q(j), \tag{16}$$

where $q(j)$ is the queueing delay as observed by flow j. When there is only one flow in the link, this yields $x_1 = c$, and $q(1) = B(1)/c$. Using (15) and (16) we have $B(1) = \alpha$ and $p(1) = \alpha/c$. When the second flow enters the link, it estimates $\hat{d}(2) = d + B(1)/c$, and perceives a queueing delay of $q(2) = B(2)/C$, while the first flow sees the true queueing delay of $q(1) =$

where the second inequality uses $p(1) = p^+(1)$. This contradiction establishes $p^+(i) > p(i)$. Since $p(0) = 0$, (7) and (18) show that, for this α, $x_1^+(i) < x_1(i)$. However, $x_1(i)$ is independent of α by (18), since $p(i) \propto \alpha$. Thus $x_1^+(i) < x_1(i)$ for all $i > 1$ in general.

This finding has two implications. On one hand, it shows that the prose version of Vegas is less fair than FAST and the implemented form of Vegas. On the other, it shows that persistent congestion is less of a problem than is predicted by the analysis of [10].

The authors of Vegas suggested using small values of α. However, the optimal value is difficult to set in practice. If it is too small, then the queueing it induces in a high speed link may be small compared to the jitter of round trip times, making measurement inaccurate. Hence, a large value must often be used, as is proposed for FAST [6]. However, if such an α is used over a low capacity link, it is possible for α^+/c to be large. As α^+/c becomes large, Vegas becomes arbitrarily unfair. In the case of two sources,

$$x_1^+(2) < \frac{c}{2 + \alpha^+/c} \to 0. \tag{23}$$

Consider also the inductive hypothesis (in i) that $p^+(j) >> p^+(k) >> d$ for all $k < j \le i$. This is true for $i = 1$ by (6), when $\alpha^+/c >> 1$. Then (5) becomes

$$\frac{p^+(i-1)}{p^+(i)} \approx \frac{c}{\alpha^+} << 1, \tag{24}$$

showing that $p^+(i) \approx d(\alpha^+/c)^i$ for all i. Thus, by (7), the most recently arriving flow obtains almost all of the capacity. Although this case is pathological, it is in principle possible under Vegas. However, it cannot occur under FAST, since $p(i)$ is independent of d.

This scaling of Vegas is in contrast to that of FAST. Although $B(k) = (p(k) - p(k-1))c$ are known to diverge [11], they diverge slower than any power of k. To see that, assume instead that there exist an $\eta > 0$ and $\gamma > 0$ such that $B(k) > \eta k^\gamma$ for all k. Approximating sums by integrals in (17) gives

$$\begin{aligned}
\frac{1}{\alpha} &= \sum_{j=1}^{i} \frac{1}{\sum_{k=j}^{i} \eta k^\gamma} \\
&\approx \frac{\gamma+1}{\eta} \sum_{j=1}^{i} \frac{1}{i^{\gamma+1} - (j-1)^{\gamma+1}} \\
&\approx \frac{\gamma+1}{\eta} \left(\frac{1}{i^{\gamma+1}} + \sum_{j=2}^{i} \frac{1}{i^\gamma (i-j+1)(\gamma+1)} \right) \\
&= O\left(\frac{\log(i)}{i^\gamma} \right) \to 0.
\end{aligned}$$

However, α is a constant, which is a contraction and shows that there is no positive power of k that $B(k)$ consistently grows faster than.

Empirically, it appears that $B(i)/B(1) = \log(i) + o(1)$. This can be seen in Figure 1, which plots $B(i)/B(1) - \log(i)$. If this is indeed the case, then the lowest throughput of any source is $B(1)/(B(1) + \ldots + B(i)) = O(1/i \log(i))$.

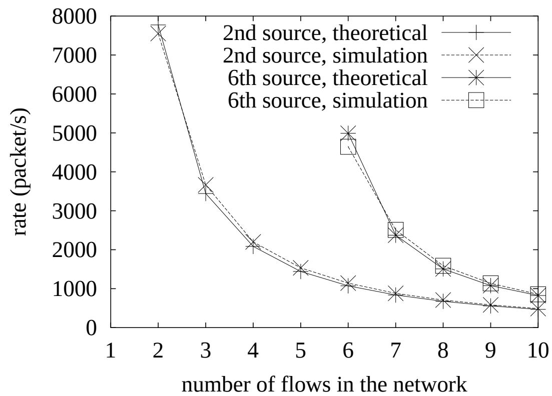

Fig. 3. Second and sixth sources rate for Vegas ($\alpha^+ = 250$ packet/s and $\beta^+ = 275$ packet/s)

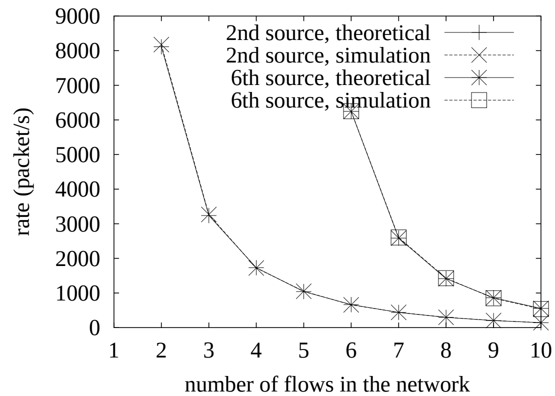

Fig. 4. Second and sixth sources rate for Vegas ($\alpha^+ = 2475$ packet/s and $\beta^+ = 2500$ packet/s)

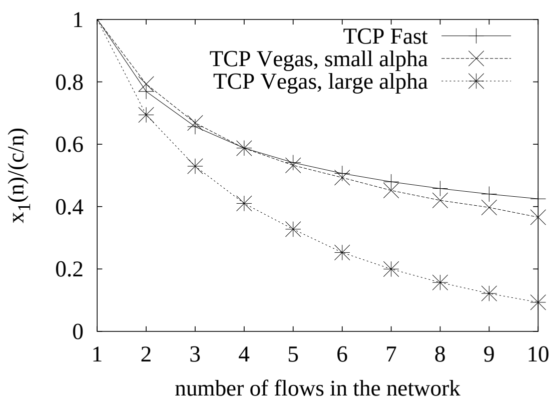

Fig. 5. Fairness to the first flow

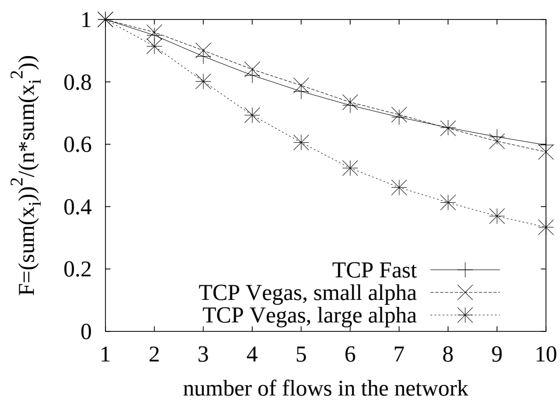

Fig. 6. Overall fairness comparison

the second criteria are plotted in Figure 6. A very consistent message emerges from Figures 5 and 6. Fairness is adversely affected by the increase in the number of sources. Also, the larger α^+ is, the more unfairly the first Vegas flow is treated, which is consistent with our analysis in the previous section. This is illustrated for a wider range of α^+ in Figure 7.

Another disadvantage of Vegas is that it generally requires more buffer space than FAST. Moreover, the rate of increase of the required buffer size will be greater for larger α^+, as is shown in Figure 8.

For completeness, we also simulated the pathological case of $\alpha^+ >> c$. We consider 10 flows with $\alpha^+ = 2500$ packet/s sharing a 1Mbps bandwidth link with round trip propagation delay 40 ms. This gives $\alpha^+ = 20c$. The analysis shows that $p(i)$ should be roughly equal to $d(\alpha^+/c)^i$. Table 1 compares this approximation with both the simulated and the theoretical $p(i)$. Simulation results were only obtained for $i \leq 3$, due to the very large queue sizes involved. The 20% discrepancy between the theoretical and simulation results for the first source is because the measured *baseRTT* includes the 8 ms packetisation delay, which is 20% of the 40 ms propagation delay.

October 1994.

2. L. S. Brakmo and L. L. Peterson, “TCP Vegas: End to end congestion avoidance on a global Internet” *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1465–1480, October 1995.
3. T. Cui and L. Andrew, “FAST TCP simulator module for ns-2, version 1.1”. [online]. Available: http://www.cubinlab.ee.mu.oz.au/ns2fasttcp/.
4. V. Jacobson, “Congestion avoidance and control,” in *Proceedings of ACM SIGCOMM ’88*, pp. 314–332, August 1988. See updated version in ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z
5. R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modeling*, John Wiley and Sons, Inc., 1991.
6. C. Jin, D. Wei, and S. H. Low, “FAST TCP for high-speed long-distance networks,” Internet draft draft-jwl-tcp-fast-01.txt, [Online]. Available: http://netlab.caltech.edu/pub/papers/draft-jwl-tcp-fast-01.txt.
7. C. Jin, D. Wei, and S. H. Low, “FAST TCP: Motivation, architecture, algorithms, performance,” in *Proceedings of IEEE INFOCOM 2004*, pp. 2490–2501, Hong Kong, March 2004.
8. C. Jin *et al.*, “FAST TCP: From theory to experiments,” *IEEE Network*, vol. 19, no. 1, pp. 4–11, Jan.-Feb. 2005.
9. USC/ISI, Los Angeles, CA. The NS simulator and the documentation. [Online]. Available: http://www.isi.edu/nsnam/ns/.
10. S. H. Low, L. L. Peterson and L. Wang, “Understanding Vegas: A duality model”, *Journal of the ACM*, vol. 49, no. 2, March 2002, pp. 207–235.
11. L. Tan, C. Yuan and M. Zukerman, “FAST TCP: Fairness and queuing issues”, accepted for publication in *IEEE Communications Letters*.
12. M. Zukerman, L. Tan, H. Wang and I. Ouveysi, “Efficiency-Fairness Tradeoff in Telecommunications Networks”, *IEEE Communications Letters*, vol. 9, no. 7, July 2005.