Far East Journal of Electronics and Communications
Volume 3, Number 1, 2009, Pages 1-16
Published Online: June 29, 2009
This paper is available online at http://www.pphmj.com

TAIMD: A TCP-FRIENDLY RATE CONTROLLING ALGORITHM USING TWO-STEP AIMD

G. G. DUMINDA NISHANTHA and YUKUO HAYASHIDA

ICT Institute
Ritsumeikan Asia Pacific University
Jumonjibaru 1-1, Beppu City 874-8577 Oita
Japan
e-mail: gamagelk@apu.ac.jp

Faculty of Science and Engineering
Saga University
Honjo, Saga-shi, Saga, 840-8502
Japan
e-mail: hayashida@is.saga-u.ac.jp

Abstract

This paper presents a novel end-to-end TCP-Friendly algorithm TAIMD that features high smoothness and fast convergence to fairness. TAIMD uses Two-step Additive Increase and Multiplicative Decrease algorithm to control the congestion window using a history-based fair-share estimation. Analytically, we derive parametric relationships for TCP-Friendliness and fairness convergence of TAIMD. Through simulations conducted using the NS-2 network simulator, performance of TAIMD is observed and compared with the performance of TCP-Friendly Rate Controlling Algorithm (TFRC).

1. Introduction

TCP (Transmission Control Protocol) is the dominant transport protocol in the

Keywords and phrases: AIMD rate control, TCP-Friendliness, fair-share estimation, fairness convergence, smoothness.

Received January 27, 2009

Internet [13] that achieves its operational stability through a powerful congestion control mechanism based on the principle of additive increase and multiplicative decrease (AIMD) [17]. However, TCP is not well-suited [2] for several emerging applications including streaming and real-time audio/video, because its abrupt rate halving, reliability and ordering semantics increase end-to-end delay and delay jitter. One solution approach of answering the above problem is to entice the applications to use reservations or differential services that provide QoS (Quality of Service) guarantee [16]. A second approach is to promote the use of end-to-end congestion controlling for the best effort traffic and deploy incentives for its use [3]. For implementing fairness to TCP through end-to-end controlling [6] two approaches are possible: (a) window based AIMD congestion, and (b) an equation based rate controlling governed by a fair-share estimation using loss rate.

In this paper, we present the design rationale and simulation based performance analysis of a novel TCP-Friendly algorithm TAIMD (Two-step Additive Increase and Multiplicative Decrease) that adopts a modified version of General AIMD mechanism to control the congestion window. Our work is motivated by the need for new controls that give high smoothness in steady state and fast convergence to fairness, when the traffic composition changes. We argue that the conventional window based algorithms, i.e., binomial (α, β, l, k) [8, 17], are not capable of optimally trade offing rate fluctuations due to static parameters they use and inherent short-term fluctuations due to the bursty nature of packet scheduling. On the other hand, the equation based algorithms like TFRC (TCP-Friendly Rate Control) are slowly changing algorithms resulting unacceptable rate variations in longer time scales especially when the frequency of the packet losses goes down [15]. Towards an optimal TCP-Friendly controlling, hybrid form of window based [7] and equation based [4] algorithms are emerging [10, 14] and customized to various network situations [11]. Latest Linux OS versions provides a trial implementation of a TCP-Friendly protocol DCCP [9], which can select its congestion control to either mimics TCP (CCID 2) or TFRC (CCID 3).

We contribute to this paper by introducing an end-to-end congestion control algorithm, TAIMD, that uses a history-based fair-share estimation (W_F) and two-step increase function, to address the above problems. We present a simple deterministic model for TAIMD and analytically derive a condition for TCP friendliness and justify it through simulations. This contribution is an enhanced version of TAIMD algorithm in [12] and presents a base for parametric decision

criteria as well as algorithmic performance against *TCP mice*. We show that the two-step increase and decreasing from the fair-share algorithms significantly improve the smoothness of the flow rate for a wide range of loss conditions over drop-tail gateways, which is the dominant prevalent form of service discipline in the current Internet. We also show that the smoothness and convergence trade-off could be decoupled to a certain extent by using a second estimation of fair-share based on loss interval, to guide the fast convergence algorithm. Finally, we suggest that the adverse effect of short-TCP flows on TAIMD performance could be avoided by restricting TCP's slow-start spike with rough awareness of the available bandwidth.

The rest of this paper is organized as follows: In Section 2, we present an overview of the TAIMD algorithm. In Section 3, the design rationale is presented in detail. Section 4 presents performance evaluations of TAIMD based on simulation results and comparisons with TFRC. Finally, conclusions appear in Section 5.

2. An Overview of TAIMD

2.1. Evolution of congestion window

TAIMD adopts a two-step additive increase function and a single multiplicative decrease function for controlling its congestion window as shown by equations (2.1) and (2.2):

$$\begin{cases} w_{i,k+1} = w_{i,k} + \alpha_1 \ (\text{if } \ w_{i,k} \le W_{F(i)}) \\ w_{i,k+1} = w_{i,k} + \alpha_2 \ (\text{if } \ w_{i,k} > W_{F(i)}), \end{cases} \tag{2.1}$$

$$w_0 = W_{F(i+1)}(1-\beta), \tag{2.2}$$

where $0 < \beta < 1$, $\alpha_1 \ge \alpha_2$ and w denotes the congestion window. The suffix i denotes the loss epoch and the suffix k denotes the kth window in the ith loss epoch. This control method features high aggressive increasing when operating below the estimated fair-share W_F, and low aggressive increasing when operating above the fair-share compared to the GAIMD [1, 17] algorithm that has a continuous increasing function. Our control model of the congestion window differs to the GAIMD protocol in three aspects: (a) use of an estimated fair-share W_F, (b) the increase function is discontinuous above and below the estimated fair-share (two-step increase) and (c) multiplicative decrease occurs from the estimated fair-share of the window size but not from the maximum window size.

2.2. Window based non-reliable data transmission

TAIMD is window based, i.e., packets are transmitted by a timer based mechanism that mimics ACK clocking. Our model takes into account both the possibility of packet reordering at the receiver and occurrence of packet losses in the feedback channel. The TAIMD source assigns a unique sequence number on each transmitted packet and uses corresponding ACKs for round-trip-time (*rtt*) estimations. The *rtt* values estimated at the source are conveyed to the sink in the forward packet header. The packet loss detection is performed at the TAIMD sink that identifies the gaps in the receiving sequence. When a gap in the sequence numbers is detected, the receiver emulates correct reception up to the 2nd out of order packet and then enters into loss-mode, where it starts sending NACK packets as illustrated in the state diagrams of Figure 1. After an rtt_{short} delay, the sink resets back to normal-mode and sends an ACK packet corresponding to the latest packet it has received.

The source is informed of packet losses through a train of NACK packets. When the first NACK is detected, the source enters into the loss-mode by reducing its congestion window to w_0 (using equation (2.2)). In the loss-mode, TAIMD transmits not more than w_0 count of packets within one round. This operation closely mimics the fast recovery of TCP. When the NACK sequence is over, the source switches back to the normal-mode and sets the number of outstanding packets to $w_0 - 1$.

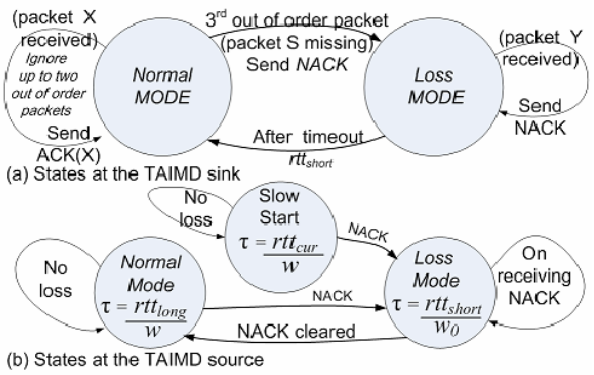

Figure 1. Operating states at TAIMD source and sink.

2.3. Controlling of transmit packet interval

As the conventional window mechanism is triggered by ACK clocking, an inherent burstiness is introduced to the sending rate. To avoid this bursty behavior,

inter-packet duration (τ) is evenly spaced along the corresponding round-trip-time duration by using a transmission timer (TX). At the TAIMD source, three *rtt* estimations are made to represent *rtt* in different time scales as shown in Table 1. For estimating moving averages, the filter coefficients are selected on heuristic observations.

Table 1. Inter-packet duration of TAIMD

State (mode)	Timer interval	RTT calculation
Slow-start	$\tau = rtt_{cur} / w$	$rtt_{cur} = t_{echo} - t_{send}$
Loss-mode	$\tau = rtt_{short} / w_0$	$rtt_{short} \leftarrow 0.8 \times rtt_{short} + 0.2 \times rtt_{cur}$
Normal-mode	$\tau = rtt_{long} / w$	$rtt_{long} \leftarrow 0.8 \times rtt_{long} + 0.2 \times rtt_{short}$

3. TAIMD Rationale

3.1. Transmission model and notations

To support the rest of the discussion, we formalize the transmission model and the notations used. For simplicity, a transmission model with *deterministic packet loss* and *binary feedback* is used as shown in Figure 2. At steady state, each TAIMD flow starts at window w_0 and increases up to w_m according to (2.1). We model this behavior in terms of rounds. A round starts with transmission of *w* contiguous packets, where *w* is the current window size. Once all these packets are transmitted and the ACK for the first packet is received by the sender, current round finishes and the next round begins. We denote *j*th round counted from the last window decrease using the suffix *j*. At the last round $j = L$, a window of w_m packets is fully transmitted, where the last packet is dropped causing a window reduction according to (2.2). The duration between two loss events is termed as a *loss epoch*, and the *i*th loss epoch is denoted by the suffix *i*. At the end of each loss epoch *i* an estimated fair-share $W_{F(i+1)}$ is calculated using the minimum and maximum window values. Let the packet loss rate at *i*th loss epoch be p_i. For our deterministic model, all loss epochs are homogeneous at steady state, hence we use the parameters w, w_0, w_m and W_F with and without the suffixes *i*, *j* as and when necessary. A constant packet size *S* and equal round trip delay *R* for all the competing flows are assumed.

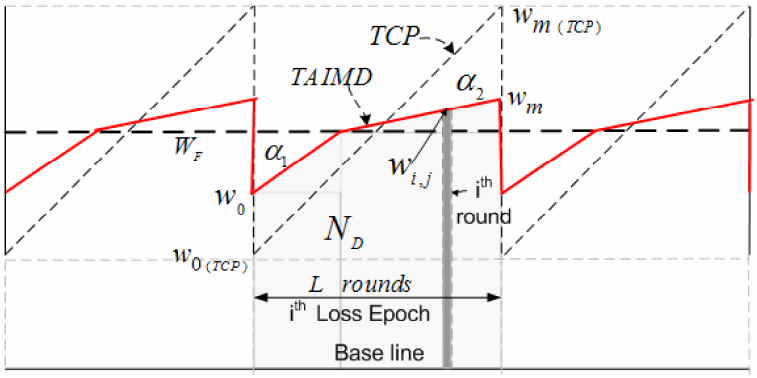

Figure 2. TAIMD transmission model.

3.2. TCP-Friendliness

In order to derive a condition for TCP-Friendliness, we consider a conformant TCP flow operating with TAIMD simultaneously through a single bottleneck link. Let the TCP loss epoch begin from $w_{0(TCP)}$ and end at $w_{m(TCP)}$ as shown in dashed lines in Figure 2.

Then the average throughput of TCP

$$\lambda_{TCP} = \left\{\frac{(w_{0(TCP)} + w_{m(TCP)}) \cdot L}{2}\right\}\frac{S}{R} = \left\{\frac{3w_m^2}{8}\right\} \cdot \frac{S}{R}. \tag{3.1}$$

From the periodic and deterministic loss assumption:

$$p_{TCP} = \frac{8}{3w_m^2}, \tag{3.2}$$

$$\lambda_{TCP} = \sqrt{\frac{3}{2}} \cdot \frac{S}{R} \cdot \frac{1}{\sqrt{p_{TCP}}}. \tag{3.3}$$

For TCP-Friendliness, the throughput of TAIMD should be less than or equal to that of the TCP. Hence

$$\lambda_{TAIMD} = W_F \cdot \frac{S}{R} \le \lambda_{TCP} \rightarrow W_F \le \sqrt{\frac{3}{2}} \cdot \frac{1}{\sqrt{p}}, \tag{3.4}$$

where

$$p = p_{TAIMD} = p_{TCP}.$$

Let N_D denote the number of packets transmitted by TAIMD in one loss epoch. Then TCP-Friendly inequality is rewritten as in (3.7) by substituting (3.5) and (3.6) in (3.4)

$$W_F = \frac{N_D}{L}, \tag{3.5}$$

$$p = \frac{1}{N_D}, \tag{3.6}$$

$$\frac{W_F}{L} \le \frac{3}{2}. \tag{3.7}$$

But we have from the model in Figure 2:

$$L = \left(\frac{W_F - w_0}{\alpha_1}\right) + \left(\frac{w_m - W_F}{\alpha_2}\right), \tag{3.8}$$

$$N_D = \left(\frac{w_0 + W_F}{2}\right)\left(\frac{W_F - w_0}{\alpha_1}\right) + \left(\frac{w_m + W_F}{2}\right)\left(\frac{w_m - W_F}{\alpha_2}\right). \tag{3.9}$$

From (3.5), (3.8) and (3.9)

$$W_F \le \frac{W_F^2[\alpha_2 - \alpha_1 - (1-\beta)^2\alpha_2 + \alpha_1 w_m^2]}{2W_F(\alpha_2\beta - \alpha_1) + 2w_m\alpha_1}$$

$$\rightarrow W_F \le \frac{w_m\sqrt{\alpha_1}}{\sqrt{\alpha_1} \pm \beta\alpha_2} \text{ (solution for quadratic equation).} \tag{3.10}$$

Substituting (3.8) and (3.10) in (3.7) and using $\beta > 0$ and $\alpha_1 \ge \alpha_2$ conditions:

$$\beta \ge \frac{2\alpha_1\sqrt{\alpha_2}}{3(\sqrt{\alpha_1} + \sqrt{\alpha_2})}. \tag{3.11}$$

An interesting observation of equation (3.11) is that the above parameters are independent of the maximum window size w_m as well as the estimated fair-share W_F.

3.3. Fairness convergence

It is clear that TCP-Friendliness is achieved by satisfying equation (3.11) under the condition that the value W_F represents the correct fair-share of the congestion

window. At steady state operation, the value W_F is a constant as no flows will enter or leave the system. However, during transients, the value W_F is to be evolved towards the actual fair-share. Using the steady state model in Figure 2, $W_{F(i+1)}$ is estimated as in (3.12). Thus, W_F will follow the ascending or descending transmission rates gradually. Using phase plots, we can show that TAIMD converges to fair-share

$$\frac{w_m - W_F}{W_F - w_0} = \frac{\sqrt{\alpha_2}}{\sqrt{\alpha_1}} \rightarrow W_{F(i+1)} = \frac{\sqrt{\alpha_1} \cdot w_{m(i)} + \sqrt{\alpha_2} \cdot w_{0(i)}}{\sqrt{\alpha_1} + \sqrt{\alpha_2}}. \tag{3.12}$$

3.4. Fast convergence

To address the problem of slow aggressiveness of TAIMD at higher congestion window values, which results in longer convergence time, W_F is updated different to that shown in equation (3.12) using a loss event interval based fair-share estimation. At steady state, it can be shown that

$$\text{TCP equivalent fair-share } W_S = \sqrt{\frac{3}{2}} \cdot \sqrt{L}, \tag{3.13}$$

where L is the average rounds in the loss epoch. As fair-share estimation done on the basis of loss event only gives a long term prediction [15] of the average fair-share, direct substitution of the result of (3.15) in (2.1) will not only introduce heavy fluctuations but also cause slow convergence to fair-share. TAIMD overcomes this problem by being less sensitive to short-term fluctuations in the loss behavior, yet using the loss-based fair-share estimate W_S, as a target to change its congestion window faster than (2.1) as shown below:

Let γ be the number of rounds required to reach W_S from the current window size using equation (1.1). If W_S could not be reached in the current loss interval L, then an external bias is applied such that W_F and w_0 are incremented in each round. Thus, if $W_S \geq W_F$ and $\{(\gamma > L) \text{ and } (L > 1)\}$, then

$$W_F \leftarrow W_F + \alpha_1,\ w_0 \leftarrow w_0 + \alpha_1. \tag{3.14}$$

3.5. TAIMD parameter space

TAIMD parameter space (α_1, α_2 and β) is only bounded by the relationships $\alpha_1 \geq \alpha_2$ and $1 > \beta > 0$. For TCP-Friendliness an additional restriction is imposed by

equation (3.11). Without loss of generality, let us assume that $\alpha_1 = k\alpha_2$ for $k \geq 1$. Consider a TAIMD flow running at steady state between minimum and maximum window values of w_0 and w_m. Let δ be the ratio of peak-to-peak window variation to the average window size. Thus,

$$\delta = \frac{w_m - w_0}{W_F} = \beta\left(1 + \frac{1}{\sqrt{k}}\right). \tag{3.15}$$

Then $\beta = \frac{2 \cdot k \cdot \alpha_2}{3(\sqrt{k}+1)}$, $\alpha_2 = \frac{3 \cdot \delta}{2 \cdot \sqrt{k}}$ and $\alpha_1 = \frac{3 \cdot \delta \cdot \sqrt{k}}{2}$.

Hence, each ordered pair (δ, k) maps to a tuple $(\alpha_1, \alpha_2, \beta)$ as illustrated in Figure 3. Note that $k = 1$ is the GAIMD control while $k > 1$ exhibits less aggressiveness and high responsiveness than GAIMD for corresponding α and β. The larger the value of k two-step increase characteristics dominates but extra controlling is necessary to ensure convergence. On the other hand, the value of δ should be sufficiently large to be adequately responsive to congestion, but should not be too large as it deteriorates smoothness of the throughput. Considering these trade-offs, on a heuristic approach, we selected the optimal values as $\delta = 0.1$ and $k = 3$, for all simulations that appear in the next section.

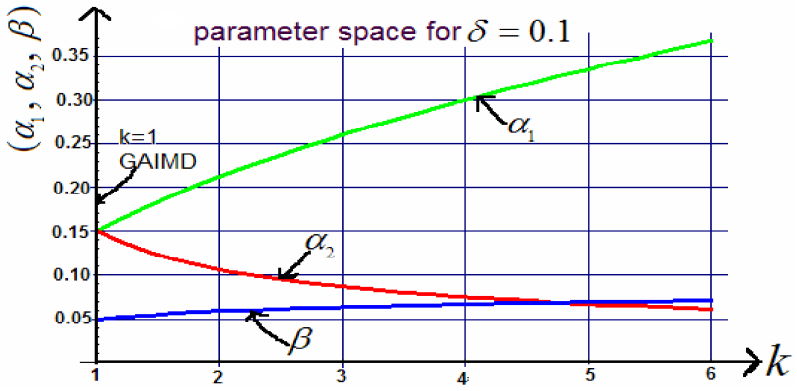

Figure 3. TAIMD parameter space for $\delta = 0.1$.

4. TAIMD Performance (Simulation based)

In order to validate our theoretical arguments on TAIMD, simulations were carried out on Network simulator (NS-2).

For better understanding the level of performance of TAIMD, we have made comparisons with TFRC which has been currently proposed [5] as a congestion control protocol for servicing multimedia traffic. For performance comparisons, we use three performance matrices: the ratio of aggregates throughput of TCP and TCP-Friendly flows as a measure of TCP-Friendliness, average coefficient of variation of the throughput of individual flows (COV) as a measure of smoothness, and packet loss rate of individual flows to understand the efficiency of the non-reliable transmission. In addition, we use the time domain behavior of the throughput of individual flows, measured in 2RTT time interval (i.e., granularity = 2RTT).

4.1. Simulation environment

Simulations were carried out for a typical dumbbell network with one bottleneck link as shown in Figure 4. The narrow link has a bandwidth *B* and one way delay (i.e., transmission and buffering) RTT. Queue discipline is drop-tail and the queue length at router R_1 is set to be equal to bandwidth-roundtrip delay product of the narrow link. All other connection links have a higher channel bandwidth *b* to ensure no packet drops occur expect at router R_1. Traffic sources and sinks are connected to router R_1 and R_2, respectively. Phase effects are tried to be minimized by using lightweight backward traffic, shifts in flow start timing and RTT delays. Unless otherwise mentioned, for all simulations we have used b = 100Mbps, RTT = 50ms.

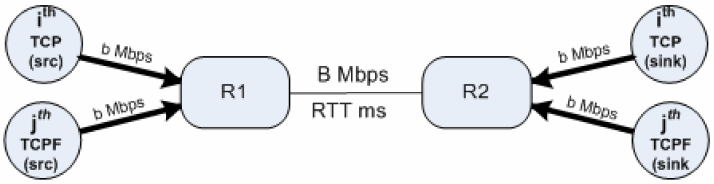

Figure 4. Simulation environment.

4.2. Convergence of TAIMD

Figure 5(a) illustrates the convergence performance of 5 TAIMD flows through 10Mbps bottleneck link. It can be seen that the basic TAIMD algorithm takes more than 200s to converge to fair-share while the algorithm with fast convergence achieves this in 50s. However, through a bottleneck link with capacity 2Mbps as shown in Figure 5(b), the convergence performance is acceptable even without the fast convergence. This behavior could be reasoned by the fact that number of rounds required to reach the estimated fair-share is not larger than the size of the loss epoch.

TFRC on the other hand does not employ any additional convergence algorithm but only follows the loss based fair-share estimation. Hence as shown in Figure 5(c), TFRC converges rapidly towards the fair-share but shows significant fluctuations about the fair-share in steady state, due to short-term errors in the estimation. When the loss rate is small (i.e., 10Mbps), TFRC does not encounter an adequate number of loss events to make a good estimate of the fair-share, thus slowly fluctuates with large amplitudes about the actual fair-share dominantly at low loss rates.

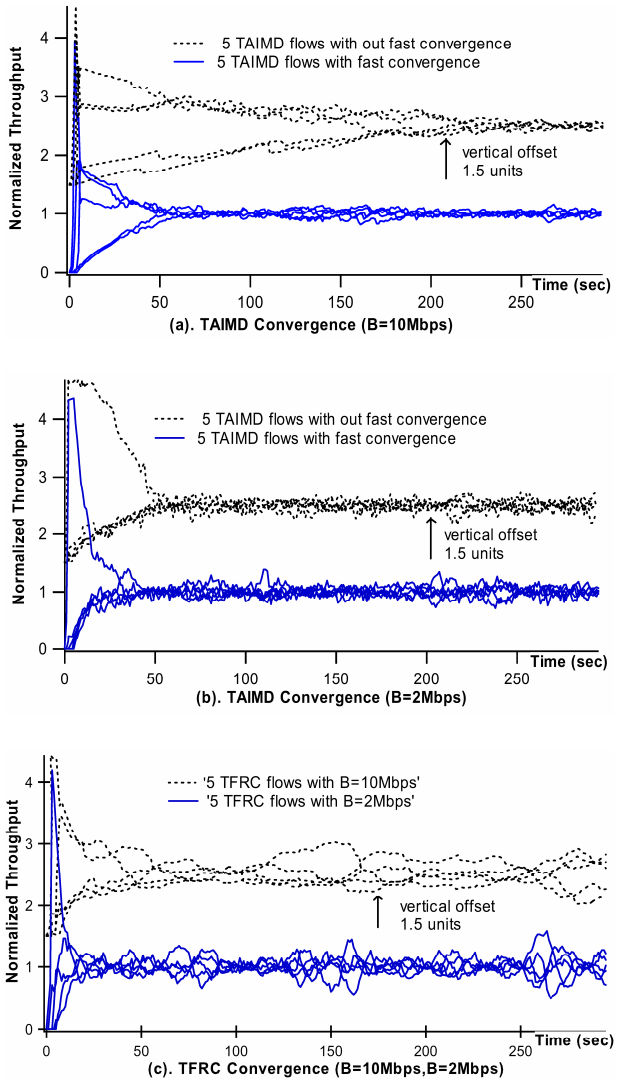

Figure 5. Convergence of TAIMD and TFRC.

Figure 6 shows the simulation results of average COV, average loss rate and throughput ratio of 5 TCP-Friendly flows competing with 5 TCP flows. Due to the existence of TCP that rapidly fluctuates, both TAIMD and TFRC experience higher loss rates and COV compared to the case without TCP. In this case, TAIMD also perform very similar to TFRC. For low loss rates, both algorithms achieve an excellent fairness, but with the increase of packet loss rate, unfairness against TCP is gradually increased. Authors believe that this behavior is due to inaccurate throughput model that both TAIMD and TFRC use (for drop-tail queuing), especially in cases that involve a large number of re-transmit and recovery events in TCP.

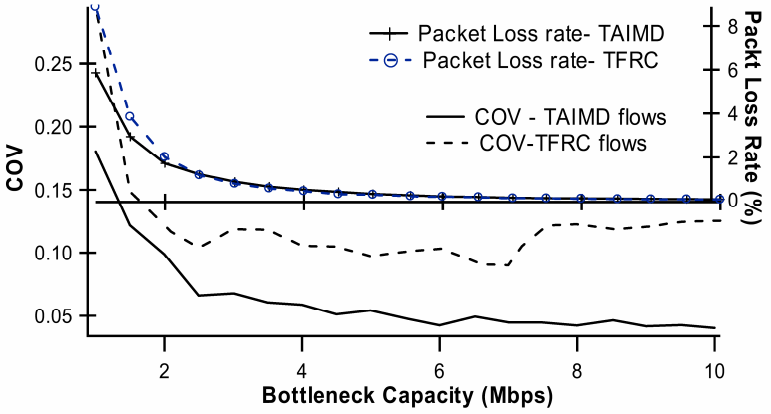

Figure 6. Intra protocol performance.

4.3. Smoothness and loss performance of TAIMD

Enhanced smoothness in absence of TCP is an important feature of TAIMD. Figure 7 shows the simulation results of average COV and average loss rate of 5 TAIMD flows compared with 5 TFRC flows. It can be seen that TAIMD has similar loss behavior to TFRC at higher link capacities but slightly improved loss performance at lower link capacities. This behavior is attributable to fact that TAIMD's use of loss rate based estimation only as a guidance for convergence, but not as a metric to directly control the sending rate. Due to the same consequence, TAIMD features a gradually decreasing COV with increasing link capacity but TFRC features considerable fluctuations due to shortage of loss events to frequently update its estimation accurately.

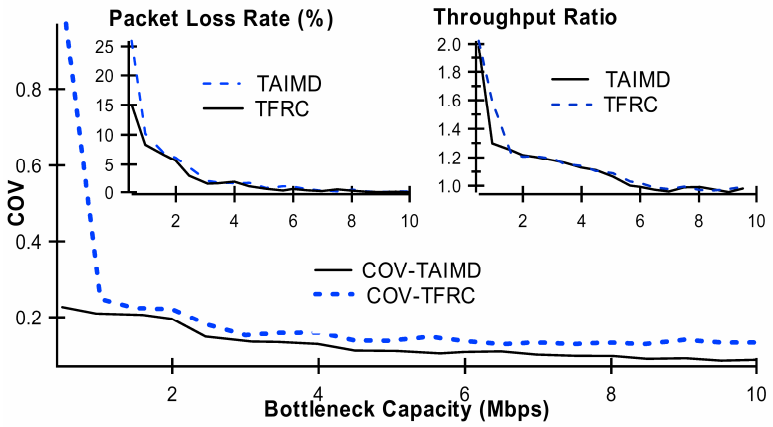

Figure 7. Inter protocol performance with TCP.

4.4. TAIMD interaction with short-TCP flows

In practice, with the increasing load of web based access, the interaction between short-TCP flows (*TCP mice*) and the TCP-Friendly protocols is expected to be more common than the interaction with long existing TCP flows (*TCP elephants*). Figure 8(a) illustrates the simulation results for this situation in which 5 TCP-Friendly flows were carrying traffic for 600s through a 7Mbps link under two operating conditions: four bandwidth non-restricted flows and one bandwidth restricted flow (at 1Mbps). A TCP elephant was introduced from 200s to 350s. From 400s to 600s, 30 TCP mice were (5s lifetime) introduced. From the simulation results, it can be seen that TAIMD interacts well in both bandwidth restricted and non-restricted cases with one long lasting TCP flow, while TFRC suffers by its inherent large amplitude fluctuations. However, in bandwidth restricted case, TAIMD experience a drop in throughput when competing with short-TCP flows whereas TFRC regains stability after a short glitch.

In practice, multimedia streams are expected to be always bandwidth restricted by some means (e.g., QoS restrictions, etc.) and need to compete with TCP mice. We propose restricting the slow-start spike of TCP as a simple solution for this. With a rough estimate of the available bandwidth and knowing the average RTT, TCP's maximum window size for slow-start could be decided. Figure 8(c) shows almost similar results indicating that a rough estimation using known bandwidth estimation tools would be satisfactory.

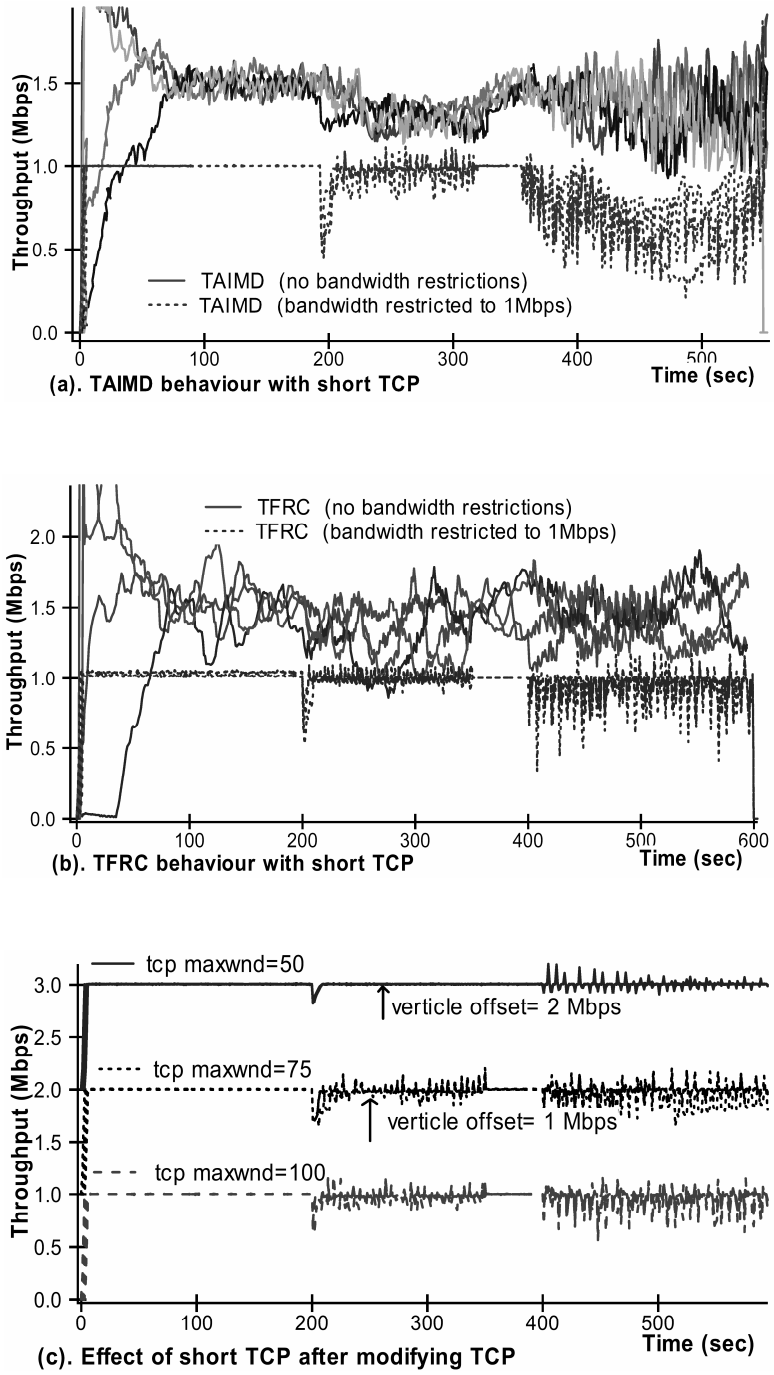

Figure 8. Interaction with short-TCP flows.

5. Conclusions

We believe that the results presented in this paper lead to a deep understanding of the TAIMD algorithm and its performance over drop-tail gateways. The simulations results reveal that TAIMD outperforms TFRC for drop-tail gateways, with significant improvement of smoothness when TCP is not present, while showing improved performance even with TCP cross traffic. This makes TAIMD appealingly applicable for future DifServ channels over low-cost drop-tail gateways as well as for general Internet with TCP cross traffic.

References

[1] D. Bansal and H. Balakrishnan, Binomial congestion control algorithms, Proc. of IEEE INFOCOM 2 (2001), 631-640.

[2] H. ElAarag, A. Moedinger and Chris Hogg, TCP friendly protocols for media streams over heterogeneous wired-wireless networks, Computer Comm. 31(10) (2008), 2242-2256.

[3] S. Floyd and K. Fall, Promoting the use of end-to-end congestion control in the Internet, IEEE TON 7(4) (1999), 458-472.

[4] S. Floyd, M. Handley, J. Padhye and J. Widmer, Equation based congestion control for unicast applications, Proc. of ACM SIGCOMM, August 2000, pp. 43-56.

[5] S. Floyd, M. Handley, J. Pahdye and J. Widmer, RFC 5348 TCP-Friendly Rate Control (TFRC) Protocol Specification, RFC 5348 Proposed Standard, 2008.

[6] J. Golestani, A class of end-to-end congestion control algorithms for the Internet, Proc. of ICNP, 1998, pp. 137-150.

[7] S. Jin, L. Guo, I. Matta and A. Bestavros, A spectrum of TCP-friendly window-based congestion control algorithms, IEEE Transactions on Networking 11 (2003), 341-355.

[8] C. M. Kellett, R. H. Middleton and R. N. Shorten, On AIMD congestion control in multiple bottleneck networks, Communications Letters, IEEE 11(7) (2007), 631-633.

[9] E. Kohler, M. Handley and S. Floyd, Datagram Congestion Control Protocol (DCCP), RFC 4340, 2006.

[10] Yuan-Cheng Lai, TCP-friendly congestion control to guarantee smoothness by Slack Term, Computer Communications 30(2) (2007), 341-350.

[11] S. W. Ng and Edward Chan, Equation-based TCP-friendly congestion control under lossy environment, J. Systems Architecture 51(09) (2005), 542-569.

[12] Duminda Nishantha and Yukuo Hayashida, Rate smoothness and TCP-Friendliness through two-step AIMD (TAIMD), Proceeding of 1st IEEE Inter. Conf. Industrial and Information Systems, Colombo, Sri Lanka, August, 2006, pp. 402-407.

[13] J. Padhye, V. Firoiu, F. Towsley and J. Kurose, Modeling TCP Reno performance: A simple model and its empirical validation, IEEE TON 8 (2000), 133-145.

[14] Injong Rhee and Lisong Xu, CUBIC a New TCP-Friendly High-Speed TCP Variants, PFLDnet, Lyon, France, 2005.

[15] Injong Rhee and Lisong Xu, Limitations of equation-based congestion control, IEEE/ACM Transactions on Networking, 2007.

[16] S. Tsao, Y. Lai and Y. Lin, Taxonomy and evaluation of TCP-Friendly congestion-control schemes on fairness, aggressiveness and responsiveness, IEEE Network, 2007.

[17] Y. Richard Yang and Simon S. Lam, General AIMD congestion control, Proceedings of ICNP, November 2000.