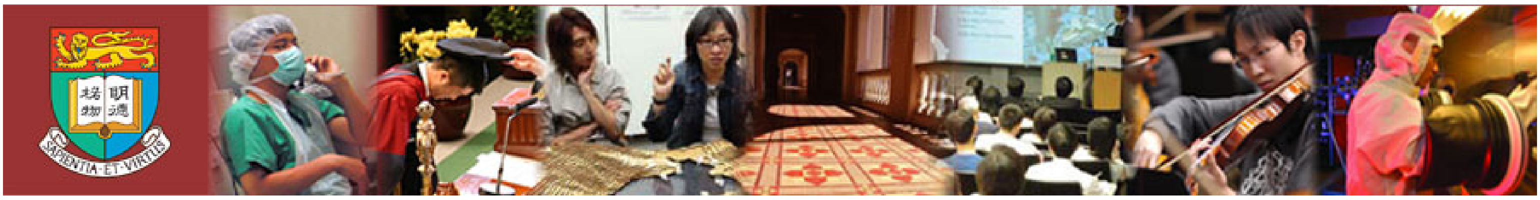


Title	Design and analysis of TCP AIMD in wireless networks
Author(s)	Lai, C; Leung, KC; Li, VOK
Citation	The 2013 IEEE Wireless Communications and Networking Conference (WCNC 2013), Shanghai, China, 7-10 April 2013. In IEEE Wireless Communications and Networking Conference Proceedings, 2013, p. 1422-1427
Issued Date	2013
URL	http://hdl.handle.net/10722/184991
Rights	IEEE Wireless Communications and Networking Conference. Proceedings. Copyright © IEEE Communications Society.

Design and Analysis of TCP AIMD in Wireless Networks

Chengdi Lai, Ka-Cheong Leung, and Victor O.K. Li
Department of Electrical and Electronic Engineering
The University of Hong Kong
Pokfulam Road, Hong Kong, China
E-mail: {laichengdi, kcleung, vli}@eee.hku.hk

Abstract—**The class of additive-increase/multiplicative-decrease (AIMD) algorithms constitutes a key mechanism for congestion control in modern communication networks, like the current Internet. The algorithmic behaviour may, however, be distorted when wireless links are present. Specifically, spurious window reductions may be triggered due to packet reordering and non-congestive loss. In this paper, we develop a framework for AIMD in TCP to analyze the aforementioned problem. The framework enables a systematic analysis of the existing AIMD-based TCP variants and assists in the design of new TCP variants. It classifies the existing AIMD-based TCP variants into two main streams, known as compensators and differentiators, and develops a generic expression that covers the rate adaptation processes of both approaches. It further identifies a new approach in enhancing the performance of TCP, known as the compensation scheme. A tax-rebate approach is proposed as an approximation of the compensation scheme, and used to enhance the AIMD-based TCP variants to offer unified solutions for effective congestion control, sequencing control, and error control. In traditional wired networks, the new family of TCP variants with the proposed enhancements automatically preserves the same inter-flow fairness and TCP friendliness. We have conducted a series of simulations to examine their performance under various network scenarios. In most scenarios, significant performance gains are attained.**

I. Introduction

Transmission Control Protocol (TCP) [1] is the *de facto* standard transport layer protocol in the Internet, and accounts for the majority of Internet data traffic. It performs congestion avoidance so that end systems back off their transmission upon the incipience of network congestion. This aims to keep the Internet operating at the optimal operational region of low delay and high throughput [6]. The additive-increase/multiplicative-decrease (AIMD) algorithm is the central mechanism in many popular TCP variants for performing congestion avoidance.

TCP uses the size of the congestion window (*cwnd*) to regulate the amount of outstanding data (i.e. data sent but not yet acknowledged), thereby controlling the load offered to the network. AIMD adjusts *cwnd* in a step-wise manner based on the estimated network load. Additive increase (AI) is carried out when no network congestion is detected, so that *cwnd* is incremented by one packet size per round-trip time (RTT). Multiplicative decrease (MD) takes over when network congestion is inferred, and *cwnd* is reduced to a fraction of its current size. The AIMD-based TCP variants essentially differ in what constitutes the signal of network congestion, and/or by how much the window should be increased/decreased in AI/MD.

The standardized TCP variants, TCP Reno and TCP NewReno (TCP-NR), are AIMD-based. A TCP receiver expects all the data packets received to be consecutively ordered. If an out-of-order packet is received, the TCP receiver will send back a duplicate acknowledgement (ACK) to its corresponding TCP sender. At the sender side, when the number of duplicate ACKs reaches a certain threshold value, say, three, fast retransmit and fast recovery will be activated. Fast retransmit treats the packet expected by the receiver as being lost since (presumably) three subsequently transmitted packets have already arrived at the receiver. It thus retransmits the packet. Fast recovery further assumes that the packet loss is due to congestion, and reduces *cwnd* by half.

Wireless networks present a great challenge in the design of the AIMD algorithms. Specifically, packet reordering and non-congestive packet loss are common over wireless networks [14]. They constitute additional causes of triple duplicate acknowledgments and trigger unnecessary reductions of *cwnd*, thereby disturbing the proper functioning of the AIMD algorithms in TCP. Packet reordering is common in modern networks due to the increased level of parallelism in various network components [11], [17]. For example, a high-speed interconnection between routers is sometimes realized via multiple physical links. Packets belonging to the same flow may thus traverse different physical paths even if they have identical routes, thereby disrupting the order of packets.

Non-congestive losses occur frequently in wireless networks because signals propagating over wireless links are subject to severe interference, noise, and propagation loss. Packets may be damaged to an extent beyond the recovery capability of error correction codes and are thus discarded. While we can recover packet losses over a wireless link locally via link-layer retransmissions, this cannot fully guarantee reliable packet delivery due to the heterogeneity of wireless networks. For example, packet losses in a WiFi-based long distance network, which is promising due to low equipment cost, have been observed to be as high as 80% [18].

Numerous AIMD-based TCP variants have attempted to adapt TCP to wireless networks. Please refer to [14] for a survey.

A. Our Contributions

The focus of this work is to propose a theoretical framework for TCP AIMD in wireless networks, which enables a *systematic* analysis of the existing AIMD-based TCP variants and assists in the design of new TCP variants. We formally classify the existing AIMD-based TCP variants into two main streams, and propose a new feasible direction for enhancing wireless TCP, known as the compensation scheme.

A tax-rebate approach has then been developed as a practical approximation of the compensation scheme. It enhances the existing TCP variants to provide unified solutions for performing effective congestion control and sequencing control over reordering, error-prone communication networks, such as wireless networks. In the traditional wired networks, the new family of TCP variants with the proposed enhancements automatically preserves the same inter-flow fairness and TCP friendliness. We have conducted a

978-1-4673-5939-9/13/$31.00 ©2013 IEEE

series of simulations to examine their performance under various network scenarios. In most scenarios, significant performance gains are attained.

In Section II, we review some representative TCP variants in the literature and, on this basis, develop the framework for TCP AIMD that accounts for non-congestive loss and packet reordering in addition to congestive loss. Section III devises the tax-rebate approach for enhancing TCP. The throughput bound for the enhanced TCP is also derived. Section IV presents the simulation results. Section V concludes and discusses some possible extensions of our work.

II. TCP AIMD Model

In this section, we develop the framework for TCP AIMD that accounts for packet reordering and non-congestive loss in addition to congestive loss. Section II-A reviews some representative TCP variants in the literature and classifies these variants into two main streams, namely *compensators* and *differentiators*. On this basis, Section II-B derives an analytical expression of the rate adaptation process of generic TCP AIMD algorithms, and formally defines a compensator and a differentiator. It goes on to motivate a new feasible direction for enhancing wireless TCP.

A. AIMD-Based TCP Variants

Blanton and Allman [3], RR-TCP [21], TCP-DCR [2], and TCP-PR [4] propose to abandon the fixed triple duplicate ACKs as a signal of a congestive loss. Instead, they proactively postpone a window reduction until a corresponding timer expires (TCP-PR), or when the number of duplicate ACKs received reaches an adaptively evolved threshold value (the other three variants). The latter event, which acts as an indication of a packet loss, is used as a signal of congestive loss, with the premise that non-congestive loss is rare.

The Eifel algorithm [16] and SACK TCP [9] try to detect any false window reduction after a reduction on the window size upon receiving triple duplicate ACKs. Packet retransmission is activated along with window reduction. Thus, a false packet retransmission implies a false window reduction. Upon successful detection, the congestion window is restored to the original size before reduction, possibly via a slow start process. However, such detection scheme will not work properly if the occurrences of non-congestive loss are non-negligible.

All these aforementioned variants try to differentiate packet reordering and/or non-congestive loss from congestive loss. We thus refer to them as *differentiators*. Differentiators focus on constructing a prompt, reliable signal of congestive loss over wireless networks. However, the construction is exceptionally challenging when both packet reordering and non-congestive loss are taken into account. Moreover, a comprehensive analysis on the inter-flow fairness of the AIMD algorithms employing different binary signals is still lacking. However, such analysis is crucial to ensure that such enhancement would not degrade inter-flow fairness.

Some other TCP variants leave the signalling mechanism intact and explore adaptive settings of the increment and/or decrement parameters. JTCP [20], TCP-FIT [19], TCP Veno [10], and TCP Westwood (TCP-W) [5] propose to estimate the network load dynamically over a TCP session. If the estimated network load is light, the reduction in the window size upon the arrival of the triple duplicate ACKs will be smaller so as to facilitate *faster recovery*. These variants essentially attempt to compensate the excessive number of *cwnd* reductions by making the reduction less abrupt. We thus refer to them as *compensators*.

Currently, there are two major problems that haunt the performance of *compensators* in the face of non-congestive loss and packet reordering. First, load estimation assumes that a longer RTT suggests a heavier network load. This is problematic over reordering channels since packet reordering always contributes to RTT variation. Second, how to set the increment and decrement parameters optimally remains conceptually unclear.

B. Framework for TCP AIMD in Wireless Networks

We now consider a generic TCP AIMD algorithm that covers both approaches of differentiators and compensators.

Denote the average size (in segments or packets) of the congestion window during $[t, t+1)$ as $W(t)$. Without loss of generality, we assume that the interval $[t, t+1)$ consists of many RTT rounds. We further assume that the equilibrium RTT is a constant D, as is customary in the literature. In each RTT round, the size of the congestion window is incremented by $(1+\gamma)$ if none of the (approximately) $W(t)$ packets sent in the previous RTT round is marked as lost due to congestion. Otherwise, the congestion window will be reduced to $\frac{1+\epsilon}{2}$ of its current size. Furthermore, a single packet is marked for triggering an MD with probability $q(t)$. Following a derivation similar to [15], we can show that the generic TCP AIMD adapts the source rate based on:

$$\begin{aligned} x(t+1) &= x(t) + (1+\gamma)\frac{(1-q(t))^{Dx(t)}}{D^2} \\ &\quad - \frac{2(1-\epsilon)}{(3+\epsilon)D} \cdot (1-(1-q(t))^{Dx(t)})x(t) \quad (1) \end{aligned}$$

where $x(t) = \frac{W(t)}{D}$.

Denote the probability of a lost packet due to congestion as $q^c(t)$. (1) with $q(t) = q^c(t)$ and $\epsilon = \gamma = 0$ corresponds to the well-behaved rate adaptation process of TCP-NR over reliable, in-order transmission channels. We refer to this rate adaptation process as the *ideal rate adaptation process* and will use it as a benchmark.

In wireless networks, AIMD-based TCP variants differ in their settings of γ and ϵ, and the way they process feedback signals from the network. The latter in turn determines $q(t)$. We discuss these in more details for different classes of the TCP variants in the following.

1) TCP-NR: For TCP-NR, ϵ and γ are zero. $q(t)$ assumes $q^a(t)$, which is defined as:

$$q^a(t) \approx q^r + q^w + q^c(t) \quad (2)$$

where q^r and q^w denote the probabilities of a packet being reordered (by more than three packets) and being lost due to transmission errors, respectively.

2) Differentiators: Per our discussion in Section II-A, differentiators set ϵ and γ to zero, and work on improving $q(t)$. An ideal differentiator can fully and promptly differentiate among congestive loss, non-congestive loss, and packet reordering. It restores $q(t)$ to be $q^c(t)$, thereby attaining the ideal rate adaptation process.

In reality, however, it is hard for TCP to make such differentiation in real time. For example, some of the existing differentiators, including RR-TCP, TCP-DCR, TCP-PR, and TCP-SACK, essentially assume that all packet losses are due to network congestion. Thus, they manage to reduce $q(t)$ to $q^d(t)$, which is defined as:

$$q^d(t) = \delta q^r + q^w + q^c(t) \quad (3)$$

where $\delta \in [0, 1)$ is the portion of the reordered packets misinterpreted as packet losses. Obviously, a smaller value of δ implies a better robustness against reordering.

3) Compensators: Per our discussion in Section II-A, compensators adopt $q(t)$ as $q^a(t)$ defined in (2), and work on adapting ϵ and γ.

For example, TCP-W estimates the available bandwidth, $B(t)$, based on the inter-arrival time of consecutive ACKs. Upon the arrival of the triple duplicate ACKs, $cwnd$ is reduced to $B(t)\tilde{D}$, where $\tilde{D}$ is the minimum measured value of RTT. We can show that this in effect sets $\gamma = 0$ and $\epsilon = \frac{3B(t)\tilde{D} - 2x(t)D}{2x(t)D - B(t)\tilde{D}}$.

We define an ideal compensator as a compensator that attains the ideal rate adaptation process. With $q(t) = q^a(t)$, this is attained by adapting γ and ϵ as:

$$\gamma^{*c} = \left(\frac{1 - q^c(t)}{1 - q^a(t)}\right)^{Dx(t)} - 1 \tag{4}$$

$$\epsilon^{*c} \approx 1 - \frac{1 - (1 - q^c(t))^{Dx(t)}}{1 - (1 - q^a(t))^{Dx(t)}} \tag{5}$$

γ^{*c} and ϵ^{*c} thus provide an answer to the problem of finding the optimal AIMD parameters raised in Section II-A. Furthermore, TCP-NR can thus become an ideal compensator by changing its increment and decrement parameters to $(1 + \gamma^{*c})$ and $\frac{1+\epsilon^{*c}}{2}$, respectively.

4) Compensated Differentiators: Compensated differentiators work on both partly improving $q(t)$ and partly adapting ϵ and γ.

We define an ideal compensated differentiator as one that attains the ideal rate adaptation process. Suppose an ideal compensated differentiator manages to restore $q(t)$ to $q^d(t)$ as defined in (3). The corresponding settings of γ and ϵ (denoted as γ^{*d} and ϵ^{*d}, respectively) can be obtained via (4) and (5) with $q^a(t)$ replaced by $q^d(t)$.

The non-ideal differentiators, RR-TCP, TCP-DCR, TCP-PR, and TCP-SACK, can thus become the ideal compensated differentiators by setting their increment and decrement parameters to $(1 + \gamma^{*d})$ and $\frac{1+\epsilon^{*d}}{2}$, respectively.

III. A Tax-Rebate Approach

In Section II, we have presented approaches that upgrade TCP-NR and the differentiators to become an ideal compensator and ideal compensated differentiators, respectively. Collectively, we refer to these approaches as the *compensation scheme*. The compensation scheme requires knowledge of various network information, such as the average window size and congestive loss rate, to set the AIMD parameters. Such knowledge is generally not readily available. In this section, we propose a tax-rebate approach as a close approximation of the compensation scheme. The approach provides a mechanism to dynamically determine the values of γ and ϵ in real time.

Tax, denoted as $T(t)$, refers to the rate of non-congestive loss and/or packet reordering that causes the reduction of *cwnd*. Specifically, if an AIMD algorithm adapts its rate based on (1), the corresponding *tax* is:

$$T(t) = q(t) - q^c(t) \tag{6}$$

In the real world, the imposition of tax discourages the purchase of a product and leads to a suboptimal equilibrium of a perfectly competitive market, if the product concerned does not incur an external cost. In our scenario, $T(t)$ is imposed upon the link prices (i.e. congestive loss rates) [15]. It discourages the AIMD algorithms to fully exploit the network capacity. Our approach essentially rebates some *tax* to a source by compensating its rate adaptation process based on $T(t)$.

In the following discussion, we will refer to the enhancements via tax-rebates for TCP-NR and TCP-PR as TCP-NR+ and TCP-PR+, respectively.[1] Section III-A describes the tax-rebate algorithm. Section III-B discusses a scheme for estimating $T(t)$. Some performance bounds have been derived in Section III-C.

A. Tax-Rebate Algorithm

The tax-rebate algorithm consists of three parts, namely, the generalized AIMD algorithm, the algorithm for the computation of γ, and the algorithm for the computation of ϵ.

1) Generalized AIMD algorithm: The AIMD algorithm can be modified in a straightforward manner by simply replacing the increment and decrement parameters. We pay special attention to the following two issues:

(1) To apply the tax-rebate approach, we do not need to modify the slow start stage ($cwnd < ssthresh$) and the initial probing of the available bandwidth by a TCP session ($first\_decrease$).

(2) The computations of γ and ϵ are both performed in the MD algorithm, which is far less frequently activated than the AI algorithm. This reduces the computational expense of the tax-rebate algorithm. Moreover, it also facilitates the computation of the two parameters, as will be elaborated later.

The pseudocode for the generalized AI and MD are shown in the following, respectively.

Algorithm 1 Procedure Generalized-AI

if *cwnd* $<$ *ssthresh* **then**
  cwnd $\leftarrow$ *cwnd* $+ 1$
else
  cnt $\leftarrow$ *cnt* $+ \frac{1+\gamma}{cwnd}$
  if *cnt* ≥ 1 **then**
    cnt $\leftarrow 0$
    cwnd $\leftarrow$ *cwnd* $+ 1$
  end if
end if

Algorithm 2 Procedure Generalized-MD

if *first_decrease* **then**
  ssthresh $\leftarrow \frac{cwnd}{2}$
  first_decrease $\leftarrow 0$
else
  compute γ
  compute ϵ
  ssthresh $\leftarrow \frac{cwnd}{2} \cdot (1 + \epsilon)$
end if
cwnd $\leftarrow$ *ssthresh*

2) Computation of γ: Rearranging (4) yields:

$$\gamma = \left(1 + \frac{T(t)}{1 - q(t)}\right)^{W(t)} - 1 \approx W(t)T(t) \tag{7}$$

[1] We have applied the tax-rebate approach to extend RR-TCP and TCP-DCR, too. The related discussion and results are similar to TCP-PR+ and thus omitted due to constraints in space.

$W(t)$ corresponds to the average window size. We note that the process for updating γ is invoked when the MD algorithm is to be activated. The average window size for the period between the current and the last invocations of the MD algorithm can be estimated as:

$$SampleW \leftarrow \frac{cwnd + ssthresh}{2} \tag{8}$$

We thus estimate $W(t)$ by computing an exponentially weighted moving average of $SampleW$ as:

$$W \leftarrow (1-\alpha)\cdot W + \alpha \cdot SampleW \tag{9}$$

where $\alpha \in (0,1)$ is the weight assigned to the instantaneous sample. In general, increasing α makes the algorithm more responsive against changes in the network environment but may undermine its robustness against random fluctuations. We recommend setting α to be between 0.2 and 0.3 as a good tradeoff. The simulation results presented in Section IV correspond to $\alpha = 0.25$. The results are similar when α is set to other values in the recommended range.

It follows that γ can be estimated as:

$$\gamma \leftarrow W \cdot T \tag{10}$$

Thus, γ can be computed by (8)-(10).

3) Computation of ϵ: By rearranging (5), we can show that:

$$\epsilon \approx T(t)\cdot\left(\frac{W(t)}{1-(1-q(t))^{W(t)}} - W(t)\right) \tag{11}$$

where $\frac{W(t)}{1-(1-q(t))^{W(t)}}$ corresponds to the average number of packets transmitted between two consecutive invocations of MD, *Np*. We make use of an existing variable in TCP, *npack*, which keeps track of the total number of data packets sent so far within the session, and introduce an extra variable, *npackLM*, which records the number of data packets sent up to the previous MD. Thus, the number of packets sent between the current MD and the previous MD can be estimated as:

$$SampleNp \leftarrow npack - npackLM \tag{12}$$

We then update *npackLM* as:

$$\textit{npackLM} \leftarrow \textit{npack} \tag{13}$$

We thus estimate Np as:

$$\textit{Np} \leftarrow (1-\alpha)\cdot \textit{Np} + \alpha \cdot \textit{SampleNp} \tag{14}$$

It follows that ϵ can be estimated as:

$$\epsilon \leftarrow T \cdot (\textit{Np} - W) \tag{15}$$

Thus, ϵ can be computed by (12)-(15).

B. Tax Estimation

The tax-rebate approach is based on the knowledge of $T(t)$. For TCP-NR, $T(t) = q^r(t) + q^w$, and for TCP-PR, $T(t) = (1-\delta)q^r(t) + q^w \approx q^w$.

An estimation of $T(t)$ requires knowledge of the non-congestive loss rate q^w. For a wireless link l, non-congestive loss rate can be obtained at the link layer of its receiving end. Suppose that the number of damaged packets received is N_l^d and the number of total packets received is N_l^p. The non-congestive loss rate over Link l, p_l^w, can be estimated as $\frac{N_l^d}{N_l^p}$. To make p_l^w up-to-date, N_l^d and N_l^p can be computed via a running average or reset periodically. p_l^w can then be communicated to the TCP sender via the control messages. Finally, q_w is computed as the sum of p_l^w of all the links traversed by the TCP connection.

For TCP-NR, the knowledge of the packet reordering rate is also needed. We are mainly interested in the forward-path reordering, which is the major reason for triggering duplicate ACKs. A TCP receiver essentially has the full information regarding the forward-path reordering. Whenever a TCP receiver detects an arrived packet reordered by more than three packets, it infers that such packet reordering will lead to a window reduction at the sender side, and thus alarms its TCP sender via a TCP option. The TCP sender can then compute the intensity of packet reordering via the number of alarms received and the number of total packets sent. Nevertheless, we note that this incurs large communication overhead, and seems to be less efficient than modifying TCP to be reordering-robust (so that the knowledge of the non-congestive loss rate suffices to estimate $T(t)$). Thus, from the implementation perspective, TCP-PR+ is preferred over TCP-NR+.

C. Performance Bounds

The following theorem estimates the throughput of TCP-NR+. The derivation is similar to that in [7].

Theorem 1: The throughput of TCP-NR+ is approximately:

$$\frac{1}{D}\cdot\frac{2}{\sqrt{(q^r+q^w)^2+\frac{8qq^c}{3q+q^r+q^w}} - (q^r+q^w)} \tag{16}$$

It is worth noting that (16) reduces to $\frac{1}{D}\cdot\sqrt{\frac{1.5}{q^c}}$, the throughput of a TCP-friendly flow defined in [7], when $q^w = q^r = 0$. This reaffirms that the tax-rebate approach is TCP-friendly over the traditional wired networks, where non-congestive loss and packet reordering are rare.

The tax-rebate approach requires an additional maintenance of a few variables (W, T, *Np*, and *npackLM*). The computations of γ and ϵ are invoked only upon the invocation of MD and involve some simple operations of addition, subtraction, multiplication, and division. Thus, the use of tax rebates induces minimal extra cost to TCP in terms of the memory requirements and computational overhead.

IV. Performance Evaluation

In this section, we present our simulation results. Section IV-A examines the effectiveness of the tax-rebate approach in boosting the performance of TCP under non-congestive loss and packet reordering. An infrastructure-based wireless network serves as the simulation topology.[2]

Section IV-B studies the inter-flow fairness property of the TCP variants enhanced with the tax-rebate approach. In the conventional wired networks, the study is trivial since the tax-rebate approach is effectively disabled in this scenario. Thus, we focus on the inter-flow fairness over wireless networks. A heterogeneous wired/wireless network with a dumbbell topology serves as the simulation topology.

[2]We have also simulated the multi-hop ad hoc wireless networks, where packet reordering can be significant due to link-layer retransmission over multiple wireless links. The performance of the tax-rebate approach is found to be similar to its performance under packet reordering in infrastructure-based networks reported in this paper.

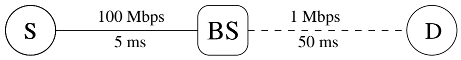


Fig. 1. An infrastructure-based wireless network.

Section IV-C discusses the implication of the tax under-estimation in some practical scenarios.

All simulation experiments have been performed using Network Simulator Version 2.29. In each test, a total of 20 runs, each lasting 2000 seconds and using different random seeds, have been performed to compute an average value and a 95% confidence interval of the performance metric of interest. In order to remove the effect of the transient states, only the statistics in the last 1000 seconds in each run are collected for the computation. In Sections IV-A and IV-C, the connection goodput is selected as the performance metric. In Section IV-B, a metric that quantifies the inter-flow fairness is used.

A. Non-Congestive Loss and Packet Reordering

We compare the performance of TCP variants enhanced with the tax-rebate approach (TCP-NR+ and TCP-PR+) with their counterparts without the enhancements in wireless networks. We further measure the performance of TCP-W, a typical compensator per our previous discussion, and use it as a benchmark for evaluating the performance gains attained by the tax-rebate approach.

In the infrastructure-based wireless network as illustrated in Fig. 1, a TCP sender (S) is connected to a TCP receiver (D) via a base station (BS). S and BS are directly connected by an in-order, error-free wired link. Packet errors are introduced randomly with probability from zero to 12% into the wireless link between BS and D. Link-layer retransmission (LLRTX) is installed over the wireless link. The enhanced TCP variants are assumed to have perfect knowledge of *tax* in these experiments. Thus, $T(t)$ is set as the aggregate rate of non-congestive loss and packet ordering for TCP-NR+, and as the non-congestive loss rate for others.

We conduct two sets of experiments. In the first set, LLRTX is disabled to simulate non-congestive loss due to packet errors. The simulation results are exhibited in Fig. 2(a)-(b). The goodput performance of TCP-NR, TCP-W, and TCP-NR+ are plotted against the packet error rate in Fig. 2(a). TCP-NR suffers severe performance deterioration as the packet error rate increases. TCP-NR+ is observed to attain significant performance improvement over TCP-NR by up to 100%. The performance gain is more significant than that of TCP-W, especially for the packet error rate lower than 5%.

TCP-PR+ is compared with TCP-W and TCP-PR in Fig. 2(b). We can derive similar observations from these plots. TCP-PR+ is observed to attain better performance than TCP-NR+ due to its proactive postponement of any window reductions.

In the second set of experiments, LLRTX is enabled with the local retransmission limit being set to three. Non-congestive loss can thus be locally recovered at the wireless link. Yet, packet reordering will be introduced since a locally retransmitted packet is intermingled with later packets. Packet reordering occurs at approximately the packet error rate. $T(t)$ is thus set as the packet error rate for TCP-NR+. $T(t)$ is set to zero for TCP-PR+, which effectively reduces the latter to TCP-PR. We thus use TCP-PR(+) to denote both TCP-PR and TCP-PR+.

The connection goodputs are plotted against the packet error rate in Fig. 2(c). TCP-PR(+) attains the best performance. This is due to its accuracy in differentiating between packet reordering and packet loss, making it a very close approximation of the ideal differentiator in this scenario. On the other hand, TCP-NR+ and TCP-W tend to trigger packet retransmissions spuriously with packet reordering. This leads to their comparatively poorer performance. However, TCP-NR+ exhibits a performance gain over TCP-NR of up to 130%, which is greater than that achieved by TCP-W.

To summarize, TCP-NR+ attains a significant performance gain over TCP-NR in both sets of experiments, but it is relatively less robust against packet reordering. TCP-PR+ proves to be an effective unified solution for both packet reordering and random loss.

B. Inter-Flow Fairness

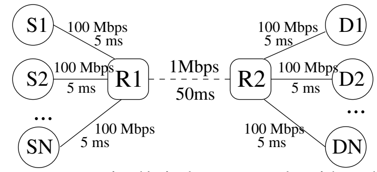


Fig. 3. A heterogeneous wired/wireless network with a dumbbell topology.

In the heterogeneous wired/wireless network with a dumbbell topology as exhibited in Fig. 3, 12 pairs of TCP senders and receivers share a wireless bottleneck link, R1-R2. We introduce the random packet loss from zero to 12% into R1-R2.

We select a metric to measure the inter-flow fairness of the 12 TCP flows. Denote the set of these 12 flows as F and the set of connection goodputs produced by them as $\mathbf{x} = (x_f : f \in F)$. $\mathcal{F}(\mathbf{x})$ is defined as [6]:

$$\mathcal{F}(\mathbf{x}) = \frac{(\sum_{f:f\in F} x_f)^2}{|F| \sum_{f:f\in F} {x_f}^2} \tag{17}$$

where $\mathcal{F}(\mathbf{x})$ varies between $\frac{1}{|F|}$ and one, with the former representing extreme unfairness (where a single flow consumes all the available bandwidth) and the latter representing perfect fairness (where all flows share the bandwidth equally). $\mathcal{F}(\mathbf{x})$ in the vicinity of one generally suggests good inter-flow fairness. It is independent of the goodput size, and thus comparable across groups of TCP flows having different aggregate goodputs.

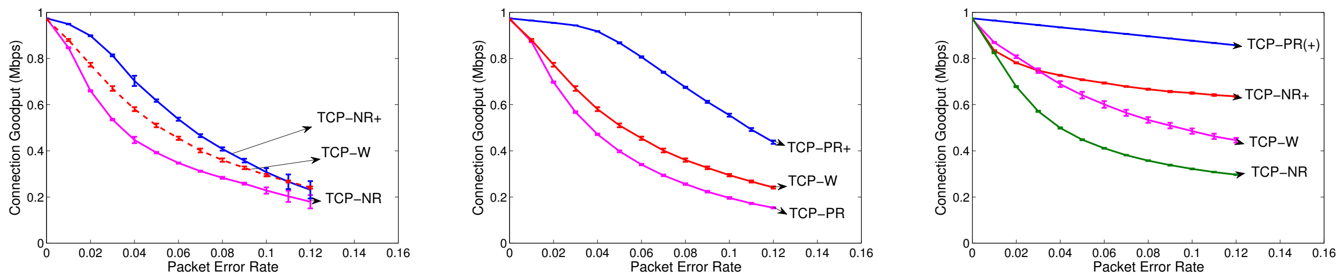
$\mathcal{F}(\mathbf{x})$ attained by flows running TCP-NR and those running TCP-NR+ are plotted against the packet error rate in Fig. 4(a). TCP-NR+ is assumed to have the perfect knowledge of *tax* (equal to the packet error rate). All the test cases can be observed to achieve $\mathcal{F}(\mathbf{x})$ beyond 0.9, thereby demonstrating good inter-flow fairness. TCP-NR+ essentially attains similar inter-flow fairness as that of TCP-NR.

Similar results are reported for TCP-PR+. Thus, the tax-rebate approach can maintain the inter-flow fairness of a TCP variant as demonstrated in our simulation results.

C. Discussion: Under-Estimation of Tax

We recommend the conservative estimation of *tax* in the tax-rebate approach so as to minimize any disruptions to the coexisting flows not enhanced by the tax rebates. In particular, if the rate of non-congestive loss or packet reordering over a link traversed by a TCP connection is unknown and not lower bounded, we assume the rate to be zero when estimating *tax*. Thus, under-estimation of *tax* may sometimes occur.

We have rerun our previous simulated cases in the infrastructure-based wireless network with LLRTX disabled when *tax* is severely under-estimated (by 50%). We have observed that tax under-estimation can lead to performance degradation of



(a) TCP-NR+ under non-congestive loss. (b) TCP-PR+ under non-congestive loss. (c) TCP-NR+ and TCP-PR+ under packet reordering.

Fig. 2. Connection goodput performance under non-congestive loss and packet reordering.

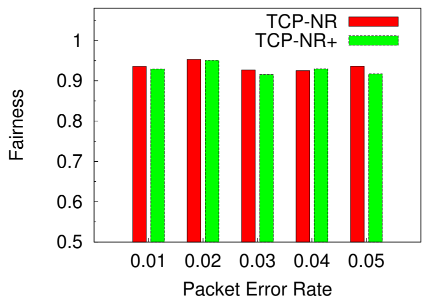


Fig. 4. Fairness performance under non-congestive loss.

TCP-NR+ (up to 22%) and TCP-PR+ (up to 52%). Nevertheless, they still maintain better performance than TCP-W under low packet error rate (up to 2% for TCP-NR+ and 7% for TCP-PR+). Moreover, they attain non-trivial performance gains over their counterparts without enhancements under the range of the packet error rates simulated.

V. Conclusions

In this paper, we have developed a framework for studying the AIMD-based TCP variants in wireless networks. We categorize TCP variants into differentiators and compensators, and further study the feasibility of a compensated differentiator. A compensation scheme has been developed based on the concepts of compensator and compensated differentiator, providing an idealized scheme for enhancing TCP NewReno and various differentiators as unified solutions for packet reordering and non-congestive loss. A tax-rebate approach has been constructed as an excellent approximation of the compensation schemes. The average throughput for the approach has been derived. Our simulation results show that the tax-rebate approach offers significant performance gains in a variety of scenarios while preserving inter-flow fairness.

There are several possible extensions to our work, including: 1) incorporating techniques for measuring non-congestive loss and packet reordering with the tax-rebate approach, and studying their interactions, and 2) extending the framework to analyze some popular non-AIMD-based TCP variants, such as CUBIC TCP [12].

Acknowledgement

This research is supported in part by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Grant No. HKU 714510E.

References

[1] M. Allman, V. Paxson, and E. Blanton. TCP Congestion Control. *IETF RFC 5681*, Apr. 2009.

[2] S. Bhandarkar and A.L.N. Reddy. TCP-DCR: Making TCP Robust to Non-Congestion Events. *Lecture Notes in Computer Science*, Vol. 3042, pp. 712-724, May 2004.

[3] E. Blanton and M. Allman. On Making TCP More Robust to Packet Reordering. *Computer Comm. Rev.*, Vol. 32, No. 1, pp. 20-30, Jan. 2002.

[4] S. Bohacek, J.P. Hespanha, J. Lee, C. Lim, and K. Obraczka. A New TCP for Persistent Packet Reordering. *IEEE/ACM Trans. Networking*, Vol. 14, No. 2, pp. 369-382, Apr. 2006.

[5] C. Casetti, M. Gerla, S. Mascolo, M.Y. Sanadidi, and R. Wang. TCP Westwood: End-to-End Congestion Control for Wired/Wireless Networks. *Wireless Networks*, Vol. 8, No. 5, pp. 467-479, 2002.

[6] D. Chiu and R. Jain. Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks. *Computer Networks and ISDN Systems*, Vol. 17, No. 1, pp. 1-14, Jun. 1989.

[7] S. Floyd. Connections with Multiple Congested Gateways in Packet-Switched Networks, Part 1: One-way Traffic. *ACM SIGCOMM Computer Comm. Rev.*, Vol. 21, No. 5, pp. 30-47, Oct. 1991.

[8] S. Floyd and T. Henderson. The NewReno Modification to TCP's Fast Recovery Algorithm. *IETF RFC 2883*, Apr. 1999.

[9] S. Floyd, J. Mahdavi, M. Mathis, and M. Podolsky. An Extension to the Selective Acknowledgment (SACK) Option for TCP. *IETF RFC 2883*, Jul. 2000.

[10] C.P. Fu and S.C. Liew. TCP Veno: TCP Enhancement for Transmission over Wireless Access Networks. *IEEE J. Selected Areas in Comm.*, Vol. 21, No. 2, pp. 216-228, Feb. 2003.

[11] L. Gharai, C. Perkins, and T. Lehman. Packet Reordering, High Speed Networks and Transport Protocol Performance. *Proceedings of IEEE ICCCN 2004*, pp. 73-78, Oct. 2004.

[12] S. Ha, I. Rhee and L. Xu. CUBIC: A New TCP-Friendly High-Speed TCP Variant. *ACM SIGOPS Operating System Review*, Vol. 42, No. 5, pp. 64-74, Jul. 2008.

[13] V. Jacobson. Congestion Avoidance and Control. *Comp. Comm. Rev.*, Vol. 18, No. 4, pp. 314-329, Aug. 1988.

[14] K.-C. Leung and V.O.K. Li. Transmission Control Protocol (TCP) in Wireless Networks: Issues, Approaches, and Challenges. *IEEE Comm. Surveys and Tutorials*, Vol. 8, No. 4, pp. 64-79, Fourth Quarter 2006.

[15] S.H. Low. A Duality Model of TCP and Queue Management Algorithms. *IEEE/ACM Trans. Networking*, Vol. 11, No. 4, pp. 525-536, Aug. 2003.

[16] R. Ludwig and R. Katz. The Eifel Algorithm: Making TCP Robust Against Spurious Retransmissions. *ACM SIGCOMM Computer Comm. Rev.*, Vol. 30, No. 1, pp. 30-36, Jan. 2000.

[17] M. Mellian, M. Meo, L. Muscariello, and D. Rossi. Passive Analysis of TCP Anomalies. *Computer Networks*, Vol. 52, No. 14, pp. 2663-2676, Oct. 2008.

[18] A. Sheth, S. Nedevschi, R. Patra, S. Surana, L. Subramanian, and E. Brewer. Packet loss characterization in Wifi-based Long Distance Networks. *Proceedings of IEEE INFOCOM*, pp. 312-320, May 2007.

[19] J. Wang, J. Wen, J. Zhang, and Y. Han. TCP-FIT: An Improved TCP Congestion Control Algorithm and Its Performance. *Proceedings of IEEE INFOCOM*, pp. 2894-2902, Oct. 2011.

[20] E. H.-K. Wu and M.-Z. Chen. JTCP: Jitter-Based TCP for Heterogeneous Wireless Network. *IEEE J. Selected Areas Comm.*, Vol. 22, No. 4, pp. 757-766, May 2004.

[21] M. Zhang, B. Karp, S. Floyd, and L. Peterson. RR-TCP: A Reordering-Robust TCP with DSACK. *Proceedings of IEEE ICNP*, pp. 95-106, Nov. 2003.