Regular Paper

Compound TCP+: A Solution for Compound TCP Unfairness in Wireless LAN

Hiroki Oda[1,a) Hiroyuki Hisamatsu[2,b) Hiroshi Noborio[2,c)

Received: March 30, 2012, Accepted: October 10, 2012

Abstract: In high-speed and long-distance networks, TCP NewReno, the most popular version of Transmission Control Protocol (TCP), cannot achieve sufficient throughput owing to the inherent nature of the congestion control mechanism of TCP. Therefore, in order to overcome this limitation, Compound TCP was proposed. Compound TCP can achieve a considerably higher throughput than TCP NewReno in high-speed and long-distance networks. The congestion control mechanism of Compound TCP consists of loss-based and delay-based congestion controls. However, in wireless LAN, the media access control used causes unfairness in the throughput among TCP connections. Compound TCP has the same type of congestion control as TCP NewReno; hence, it is expected that the problem will occur among Compound TCP connections. In this study, we evaluate the performance of Compound TCP for wireless LAN, and demonstrate that the throughput among Compound TCP connections becomes unfair. Then, we propose Compound TCP+, which implements a finer congestion control by detecting a state of slight congestion. Using simulation, we show that in wireless LAN, Compound TCP+ connections achieve fairness and share the bandwidth equally. We also demonstrate through simulation that Compound TCP+ achieves high throughput in a high-speed wired network.

Keywords: compound TCP, fairness, wireless LAN, congestion control

1. Introduction

Transmission Control Protocol (TCP) is a transport-layer protocol that is currently used as a standard for the Internet. However, the congestion control mechanism of TCP NewReno (hereafter, unless otherwise stated, "TCP" denotes "TCP NewReno"), the most popular version of TCP, cannot use the network bandwidth efficiently in high-speed and long-distance networks [1]. Several congestion control mechanisms have been proposed to solve this problem [2], [3], [4], [5], [6], [7]. These proposed mechanisms can be divided into two types—loss-based [2], [3], [4], [5] and delay-based [6], [7] congestion control.

HighSpeed TCP [2], Scalable TCP [3], BIC TCP [4], and CUBIC TCP [5] are classified as loss-based congestion control protocols that use packet losses as the index of congestion. Compared with TCP, these new transport-layer protocols enlarge the increase size of the congestion window at the time of receiving ACK packets and reduce the decrease size of the congestion window at the time of packet losses, and utilize the bandwidth effectively in a high-speed and long-distance network. This method can also be called extremely greedy congestion control because it continues to increase the congestion window until a packet loss occurs.

FAST TCP [6] and transport-layer protocol [7] are classified as delay-based congestion control protocols that use the network delay as the index of congestion. FAST TCP increases its congestion window exponentially when a network has vacant bandwidth, and the bandwidth is effectively utilized in a high-speed and long-distance network. Moreover, by measuring the available bandwidth and predicting the round-trip time, the transport-layer protocol [7] utilizes the bandwidth effectively by conducting congestion control so that the queuing delay is reduced. However, the throughput of the TCP in delay-based congestion control significantly decreases when it competes with the TCP in loss-based congestion control [8].

As described, loss-based congestion control suffers from the problem that a congestion window continues to increase in size until a packet loss occurs. Moreover, delay-based congestion control has the problem that the throughput of a connection that uses delay-based congestion control degrades significantly when it competes with a connection that uses loss-based congestion control. The new congestion control is therefore proposed [9], [10], [11], [12], [13], [14], [15], [16], [17]. According to the available bandwidth, which is estimated from the receiving interval of an ACK packet, TCP Westwood [9] determines the decrease in the width of the congestion window at the time of a packet loss. Moreover, TCP Westwood+ [10] estimates the available bandwidth using the round-trip time and the decrease in the width of the congestion window at the time of a packet loss. Furthermore, TSUNAMI [11] and UDT [12] are protocols that combine TCP and UDP.

Furthermore, mechanisms that combine loss-based and delay-based congestion controls [13], [14], [15], [16], [17] have been

1 Graduate School of Information Science and Arts, Osaka Electro-Communication University, Shijonawate, Osaka 575–0063, Japan
2 Department of Computer Science, Osaka Electro-Communication University, Shijonawate, Osaka 575–0063, Japan
a) dt11a001@oecu.jp
b) hisamatu@isc.osakac.ac.jp
c) nobori@isc.osakac.ac.jp

proposed. When a network is not in a congestion state, these mechanisms increase the congestion window size, and utilize the vacant bandwidth that TCP cannot use fully. Moreover, when a network is in a congestion state, these mechanisms can match the fairness of TCP. Owing to the installation of Compound TCP in MS Windows since Windows Vista SP1, it is expected that Compound TCP will be widely used henceforth. Therefore, in this paper, we focus on Compound TCP.

In wireless LAN, fairness in throughput is not achieved when multiple TCP connections are involved in data transfer [18], [19], [20]. This problem originates in the wireless LAN access control and the TCP congestion control. In wireless LAN, when a terminal starts communication, it first determines whether the other terminals are communicating. Then, after acquiring a transmitting right, the terminal transmits packets. All wireless terminals, including access points, obtain this transmitting right in a fair manner. However, owing to this fairness, buffer overflow occurs at an access point, and many ACK packets are discarded. Hence, TCP connections with small throughput and TCP connections with large throughput coexist in wireless LAN.

Similar to TCP, Compound TCP uses loss-based congestion control. Therefore, when Compound TCP is used in wireless LAN, the throughput among connections may become unfair. In this study, we evaluate the performance of Compound TCP in wireless LAN by simulation, and show that the throughput of Compound TCP becomes unfair among connections similar to TCP. Further, we propose Compound TCP+, which implements a finer congestion control by detecting a state of slight congestion. In addition, we use simulation to demonstrate that Compound TCP+ shares the bandwidth fairly in wireless LAN. Finally, we show that Compound TCP+ can obtain a high throughput in a high-speed and long-distance network.

The rest of this paper is organized as follows. First, in Section 2, we explain the problem of unfair throughput of TCP in wireless LAN. In Section 3, we discuss the algorithm of the congestion control mechanism of Compound TCP, and show that the unfair throughput problem also occurs among Compound TCP connections in wireless LAN. In Section 4, we propose Compound TCP+ and evaluate its performance. Finally, in Section 5, we state the conclusion, and discuss future work.

2. Unfairness in TCP Throughput in Wireless LAN

In wireless LAN, TCP has two fairness problems: (1) the throughput between up and down flows becomes unfair [18], [19], [20] and (2) the throughput between up flows becomes unfair [21], [22]. In this paper, we deal with the problem of the unfairness of throughput between up flows. Below we discuss the latter problem, past researches performed to addresses the unfairness issue, and the disadvantages of the current methods.

2.1 Cause of Unfairness

In wireless LAN, when multiple TCP connections perform a data transfer, this results in unfairness among the TCP connections [18], [19], [20]. This problem originates in Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA), which is the access control used in wireless LAN, and the congestion control of TCP. In CSMA/CA, a wireless terminal first verifies the existence of communication between other wireless terminals. If no other wireless terminals are transmitting packets, this wireless terminal acquires a transmitting right, and then, begins transmission. Each wireless terminal and an access point can obtain a transmitting right fairly. Here, we consider the case where multiple wireless terminals perform data transfer to hosts using TCP via an access point. Let n be the number of wireless terminals connected to an access point The wireless terminals and the access point obtain a transmitting right fairly; the probability that each wireless terminal and an access point obtain a transmitting right is $1/(n + 1)$. Next, we consider the up and down flows in wireless LAN. During up flows, wireless terminals transmit data packets, and during down flows, an access point transmits TCP ACK packets. Since n wireless terminals in up flows require a transmitting right, the probability that up flows obtain a transmitting right is $n/(n + 1)$. On the contrary, only one access point in down flows requires a transmitting right; therefore, the probability that down flows obtain a transmitting right is $1/(n + 1)$. Thus, the probability that down flows obtain a transmitting right decreases with an increase in the number of wireless terminals. Consequently, at an access point, ACK packets continue to be accumulated at the buffer, thus resulting in the dropping of ACK packets due to the buffer overflow.

When three duplicate ACK packets are received, TCP concludes that packet loss has occurred, and initiates “fast retransmit”. In fast retransmit, the discarded packet is retransmitted without waiting for the retransmission timeout. When buffer overflow has occurred at an access point, it is difficult for a connection with a small congestion window to receive the three duplicate ACK packets required for fast retransmit. When a packet loss occurs, and three duplicate ACK packets cannot be received, this results in a retransmission timeout.

TCP decreases the congestion window to one when a timeout occurs. Owing to the extremely small size of the congestion window of the TCP connection involved in the timeout, the possibility of another retransmission timeout is high. Therefore, the TCP connection in which the retransmission timeout occurred once can rarely increase the congestion window. Thus, the throughput of such a connection becomes extremely low. On the contrary, a connection with a large congestion window has a high probability of successful retransmits in the fast retransmit when a packet loss occurs. Therefore, the difference in the size of the congestion windows becomes large, thus causing a large difference in the throughput. Hence, throughput fairness among TCP connections is not achieved.

2.2 Related Work

Various solutions for this problem have been proposed and studied [21], [22], [23], [24], [25]. For instance, TCP throughput fairness is improved by changing the MAC layer protocol of an access point [23]. In Ref. [23], contention window (CW), a range of random numbers used in calculating the waiting time for collision avoidance, is determined so that an access point may obtain the predominancy of the packet transmission. Another method proposes that the data packet of up flows should be discarded probabilistically based on the number of packets in an access point buffer [21]. The congestion control of TCP decreases the congestion window when a packet loss is detected. The transmission rate of the connection whose congestion windows are decreased becomes small. Therefore, this method prevents the discarding of ACK packets at an access point by intentionally discarding the data packet of up flows, thus causing degradation in the transmission rate of up flows. Consequently, the fairness among TCP connections improves. Changes in the minimum and maximum values of CW are proposed according to the type of traffic [24]. This improves the fairness among connections in wireless LAN using IEEE 802.11e Enhanced Distributed Channel Access (EDCA). Further, the buffer management mechanism ensures that the number of packets of each flow that exists in an access point buffer have approximately similar values [25].

As described above, there has been significant research in the area of improving throughput fairness of TCP in wireless LAN. However, most proposed approaches require a change in an access point. This infringes the end-to-end principle of not implementing complicated control in a network. In addition, it is difficult to add the modification to an access point because the control mechanism at the access points is implemented in its hardware. Therefore, most methods that have been proposed are not feasible for implementation. A new congestion control mechanism that uses the dropping of ACK packets as an indication of congestion was proposed [22]. Although this adheres to the end-to-end principle, and only requires a change in the TCP on the sender side, it results in a decrease in throughput. In this study, we propose a new method that complies with the end-to-end principle, and does not require modification of an access point.

Some studies on Compound TCP have been pursued in the past [16], [17], [26], [27]. For example, it was shown by simulations and experiments in a real network that Compound TCP achieves a high link utilization and shares the bandwidth fairly with TCP Reno in a high-speed and long-distance network [16], [17], [26]. Further, a method for automatically adjusting γ, the threshold of the delay window of Compound TCP, was proposed, and its effectiveness was demonstrated by simulations and experiments in a real network [27]. Moreover, in Ref. [28], the authors evaluated Compound TCP in the wireless LAN environment using simulations. Their results show that, when Compound TCP is used in wireless LAN, throughput unfairness between up flows occurs. The throughput unfairness is explained based on the sending-out window of Compound TCP and loss of ACK packets. However, no study has been performed from the viewpoint of the Compound TCP on conducting a hybrid congestion control that combines loss-based and delay-based congestion control. In contrast, in this paper, which focuses on the characteristics of Compound TCP, we show that there is no increase/decrease in a delay window in wireless LAN, and as a result, the same control as TCP is conducted, causing throughput unfairness.

3. Compound TCP Throughput in Wireless LAN

In this section, we first explain the congestion control mechanism in Compound TCP. Then, we evaluate the performance of Compound TCP in wireless LAN, and show that, as seen in TCP, throughput fairness among connections is not achieved.

3.1 Congestion Control in Compound TCP

Compound TCP implements a window-based congestion control, and adjusts the number of packets sent to a network on the basis of the sending-out window. The sending-out window, *swnd* of Compound TCP is given by

$$swnd = cwnd + dwnd \tag{1}$$

where *cwnd* is the loss window used in loss-based congestion control, and *dwnd* is the delay window used in delay-based congestion control.

Compound TCP consists of two operational phases called the slow start phase and the congestion avoidance phase. Compound TCP increases the loss window size with every receipt of an appropriate ACK, and the increase in the loss window size depends on the phase. The loss window in each phase is given by

$$cwnd = \begin{cases} cwnd + 1 \\ \quad (slow\ start\ phase) \\ cwnd + \dfrac{1}{swnd} \\ \quad (congestion\ avoidance\ phase) \end{cases} \tag{2}$$

Compound TCP increases the size of the loss window exponentially and linearly in the slow start phase and the congestion avoidance phase, respectively. Moreover, when a packet loss is detected, the decrease in the size of the loss window differs depending on the packet loss detection method. The decrease in the size of the loss window is given by

$$cwnd = \begin{cases} \dfrac{cwnd}{2} & (duplicate\ ACKs) \\ 1. & (timeout) \end{cases} \tag{3}$$

Compound TCP has two types of detection methods for packet loss. When a packet loss is detected owing to the receipt of three duplicate ACKs, it is concluded that slight congestion has occurred in the network, and Compound TCP decreases the loss window size to half of the current value. In contrast, when a packet loss is detected owing to a timeout, it is concluded that serious congestion has occurred in the network, and Compound TCP decreases the size of loss window to one.

The delay window remains unchanged in the slow start phase, but it changes in the congestion avoidance phase. The increase or decrease in the size of delay window in the congestion avoidance phase is calculated as follows: First, the number of packets queued in the entire network, *Diff*, is estimated by

$$Diff = \left(\frac{swnd}{baseRTT} - \frac{swnd}{RTT}\right) \cdot baseRTT \tag{4}$$

where $baseRTT$ is the minimum value of the round-trip time and RTT is the current round-trip time. When packet loss does not

Table 1 Definition of notations.

swnd	sending-out window
cwnd	loss window
dwnd	delay window
Diff	number of packets queued in the network
basRTT	minimum round-trip time
RTT	current round-trip time
α, β, k, ζ	control parameters
γ	threshold value for *Diff*

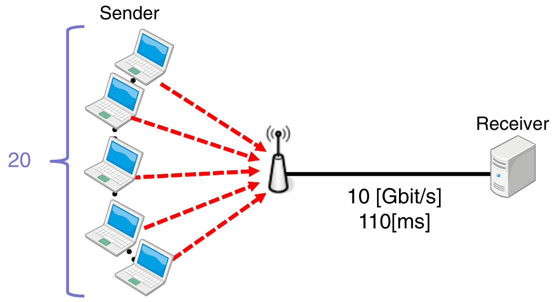


Fig. 1 Simulation model for wireless LAN.

occur, Compound TCP changes the size of the delay window for every round-trip time, and the size of delay window is calculated by the following equation that is based on *Diff*.

$$dwnd = \begin{cases} dwnd + (\alpha \cdot swnd^{k} - 1)^{+} & (5) \\ \quad (Diff < \gamma) & \\ (dwnd - \zeta \cdot Diff)^{+} & (6) \\ \quad (Diff \geq \gamma) & \end{cases}$$

where $(x)^{+}$ is defined as max (x, 0). When *Diff* is smaller than the threshold γ, it is determined that a network is not in the congested state, and the delay window increases. On the other hand, when *Diff* is larger than the threshold γ, it is determined that congestion has occurred in the network, and the delay window decreases.

Compound TCP decreases the delay window when three duplicate ACKs are received, or a retransmission timeout occurs. In this case, the size of delay window is given by

$$dwnd = \begin{cases} (swnd * (1 - \beta) - cwnd/2)^{+} \\ \quad (duplicate\ ACKs) \\ 0 \\ \quad (timeout) \end{cases}$$

where β is the control parameter for Compound TCP. The notations used in this section are summarized in **Table 1**.

3.2 Performance Evaluation of Compound TCP

We investigate the fairness among Compound TCP connections in a wireless LAN, by using the ns-2 [29] simulator. In the simulation model shown in **Fig. 1**, 20 wireless terminals transmit data packets through a single access point. The link bandwidth and delay between the access point and the receiver host are 10 [Gbit/s] and 110 [ms], respectively. The packet size is 1,500 [byte]. The wireless LAN standard, IEEE 802.11g, is used. The Slot Time, Short Inter Frame Spacing (SIFS), Distributed Inter Frame Spacing (DIFS), CW_{min}, CW_{max}, and Data Rate are 9 [μs], 16 [μs], 34 [μs], 15, 1,023 and 54 [Mbit/s], respectively. The simulations run for 500 [s], and the simulation results from the last 400 [s] are used for the measurements. The network parameters and wireless LAN parameters used in the simulations are summarized in **Table 2**. The parameters for Compound TCP, α, β, k, ζ, and γ are 1/8, 1/2, 1, 0.8 and 30, respectively. Note that we used the same values for parameters of Compound TCP as those used in Ref. [17], in which Compound TCP is proposed. The Compound TCP parameters used in the simulations are summarized in **Table 3**.

Table 2 Parameters used in simulations for wireless LAN.

Network Parameters	
Number of wireless terminals	20
Two-way propagation delay	220 [ms]
Link bandwidth	10 [Gbit/s]
Packet length	1,500 [byte]
Wireless LAN Parameters	
Wireless LAN Standard	IEEE 802.11g
Slot time	9 [μs]
SIFS	16 [μs]
DIFS	34 [μs]
CW_{min}	15
CW_{max}	1,023
Data rate	54 [Mbit/s]

Table 3 Parameters used in simulations for Compound TCP.

α	1/8
β	1/2
ζ	1
k	0.8
γ	30

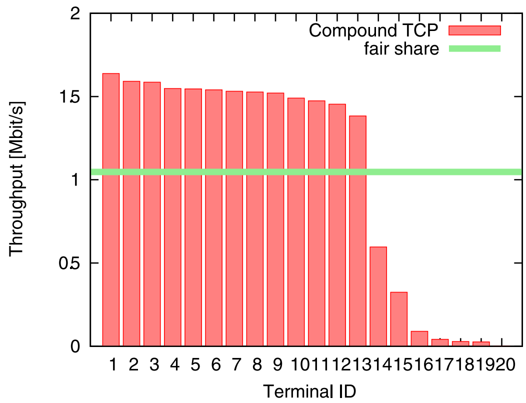


Fig. 2 Compound TCP throughput in wireless LAN.

The loss-based congestion control of Compound TCP implemented in the current version of ns-2 has a problem in the timer implementation; hence, it cannot determine the progress of the round-trip time accurately. We resolve this problem and run simulations.

Figure 2 shows the average throughput of Compound TCP connections in decreasing order of values. The value when the bandwidth is shared fairly is also plotted in the figure. From Fig. 2, it can be observed that connections with a high throughput and connections with an extremely low throughput are present. Therefore, it can be concluded that the problem of unfair throughput, existing in TCP, also exists in the case of Compound TCP.

Figure 3 shows the loss, delay and sending-out windows of the connection that achieved the highest throughput in Fig. 2. From Fig. 3, it can be observed that there is no increase or decrease in the delay window; further, only the loss window changes. This means that the congestion control of Compound TCP in wireless LAN performs in the same manner as TCP does. Therefore, when Compound TCP is used in wireless LAN, it is thought that the

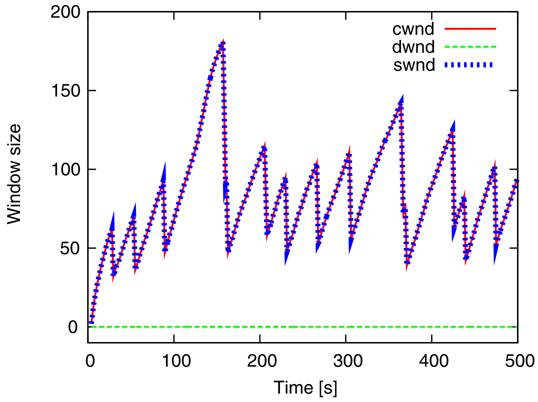


Fig. 3 Size of the loss, delay, and sending-out windows of Compound TCP.

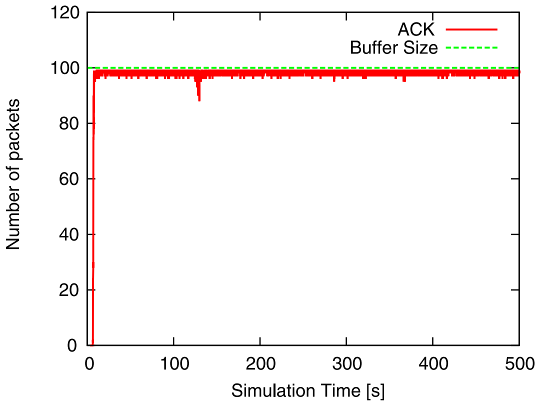


Fig. 4 Number of packets in the access point buffer.

throughput unfairness problem between the connections occurs in a similar way as in TCP.

Figure 4 shows the number of packets in the access point buffer. From Fig. 4, it can be observed that the access point buffer is always full. Many packets are accumulated at the access point buffer, and the round-trip time increases. Delay-based congestion control judges from the increase in the round-trip time that the network is in a slightly congested state, and continues decreasing its delay window. Therefore, as shown in Fig. 3, the delay window size is always zero. In contrast, loss-based congestion control is not concerned with the accumulation status of the access point buffer but makes increases in its loss window until it detects a packet loss. Therefore, in wireless LAN, Compound TCP conducts the same congestion control as TCP does.

In Fig. 3 and Fig. 4, it can be observed that the delay window is zero when many packets are buffered at the access point and buffer overflow occurs. At an access point, an increase in the number of buffered packets increases the queuing delay. When loss-based congestion control is implemented, the number of buffered packets continues to increase until buffer overflow occurs. Moreover, a delay-based congestion control decreases its delay window if the queuing delay increases. Therefore, before buffer overflow occurs, the delay window decreases to zero. **Figure 5** shows the simulation results when the number of wireless terminals in the network model shown in Fig. 1 is two. This figure shows the loss, delay, sending-out window, and the time of buffer overflow occurring at an access point. From Fig. 5, we can find that the delay window of Compound TCP is set to zero before

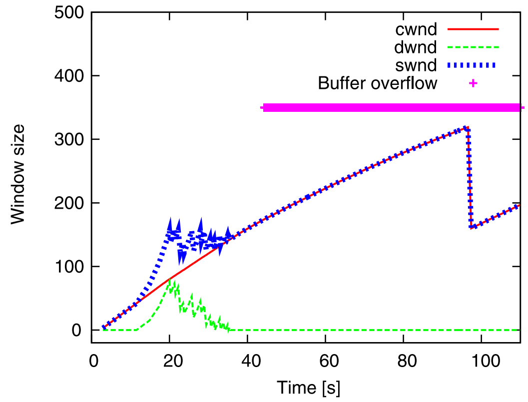


Fig. 5 Size of the loss, delay, sending-out windows of Compound TCP, and time of buffer overflow.

buffer overflow occurs.

4. Compound TCP+

In Section 3, it was seen that when Compound TCP is used in wireless LAN, the problem of unfair throughput among connections exists, similar to the case of TCP. In this section, we propose Compound TCP+, which solves this problem. Then, we demonstrate the effectiveness of Compound TCP+ by simulations.

4.1 Congestion Control in Compound TCP+

In wireless LAN, the mechanism of loss-based congestion control and absence of delay-based congestion control lead to unfairness in the throughput among Compound TCP connections. Loss-based congestion control is a greedy control mechanism that fills an access point buffer. Therefore, we propose a method utilizing delay-based congestion control to eliminate the greed of a loss-based congestion control mechanism when the network is in a state of congestion just prior to the occurrence of a packet loss.

The delay window of Compound TCP+ changes in a manner similar to the case of Compound TCP, and is based on the estimated value, *Diff*, of the number of packets queued in the entire network. *Diff* is obtained by Eq. (4). The delay window is determined on the basis of *Diff*. When *Diff* is less than the threshold γ, it is concluded that the network is not in a congested state, and Compound TCP+ increases the delay window. When the network is not congested, the delay window of Compound TCP+ is specified by Eq. (5). When *Diff* exceeds the threshold γ, it is concluded that the network is in a congested state, and the delay window is decreased. In this case, the delay window is specified by Eq. (6).

The change in the loss window size depends on the delay window. When the size of delay window is greater than zero, it is concluded that the network is not in a congested state. At this time, Compound TCP+ increases the loss window similar to the case of Compound TCP. The loss window in Compound TCP+ is calculated using Eq. (2).

When an appropriate ACK is received, and the delay window size is equal to zero, the network is considered to be in a lightly congested state. If the loss window increases as in the case of Compound TCP, the congestion will worsen, and lead to buffer overflow at the access points. Therefore, when the delay window size is zero, Compound TCP+ changes the loss window at every round-trip time according to the following equation:

$$cwnd = f(cwnd)$$

where $f(cwnd)$ is a function that returns a value less than or equal to $cwnd$. We examine the function $f(cwnd)$ in the following section. Delay-based congestion control decreases its delay window size using Eq. (6), when *Diff* exceeds γ. When *Diff* continues to exceed γ, the delay window size continues to decrease, until it reaches the minimum value of zero. A network is considered to be close to the state of a packet loss occurrence when the delay window attains its minimum value. However, in spite of the delay window size being zero, Compound TCP continues to increase its loss window, thus, causing buffer overflow at an access point. Hence, fairness among connections is not achieved. Therefore, Compound TCP+ does not increase the loss window size based on the function $f(cwnd)$ for every round-trip time when the delay window size is zero. It must be noted that, when packet loss occurs, there is no change in the Compound TCP+ operations compared to Compound TCP. The sending-out window is obtained by Eq. (1). The sending-out window is the sum of the loss and delay windows, similar to the case of Compound TCP.

4.2 Performance Evaluation of Compound TCP+

Next, we evaluate Compound TCP+ by simulation. The simulation environment and the parameters of IEEE 802.11g are the same as described in Section 3. By simulation, we investigate the following functions to determine which function we should use in Compound TCP+, respectively.

$$f(cwnd) = \lceil cwnd \times 2/3 \rceil \quad (7)$$

$$f(cwnd) = \lceil cwnd - 1/2 \rceil \quad (8)$$

$$f(cwnd) = cwnd \quad (9)$$

where $\lceil x \rceil$ is the minimum integer value that is greater than or equal to x. When the delay window size is zero, the network is considered to be in a state close to the occurrence of a packet loss. Thus, when the delay window size is zero, the loss window size should not be increased. In this study, we investigate the following cases: (1) multiplicative decrease in the loss window size (Eq. (7)), (2) linear decrease in the loss window size (Eq. (8)), and (3) no change in the loss window size (Eq. (9)). When an appropriate ACK is received, and the delay window size is zero, the network is not in a serious congestion state that would cause a packet loss. Therefore, we determine the function $f(cwnd)$ such that the loss window in this case is greater than the loss window in the case where a packet loss occurs (in the case of packet loss, the loss window is specified by Eq. (3)). The simulations are executed 10 times. The time period for each simulation is 500 [s] and the last 400 [s] are used for calculating simulation results — the fairness index, the round-trip time, and the total throughput.

We evaluate the fairness among the Compound TCP connections, and the Compound TCP+ connections. In this study, Fairness Index [30] represents the index of fairness. Let n be the number of samples and $x_i (1 \leq i \leq n)$ be the value of a sample. Fairness Index f, which represents the fairness of the sample values, is given by

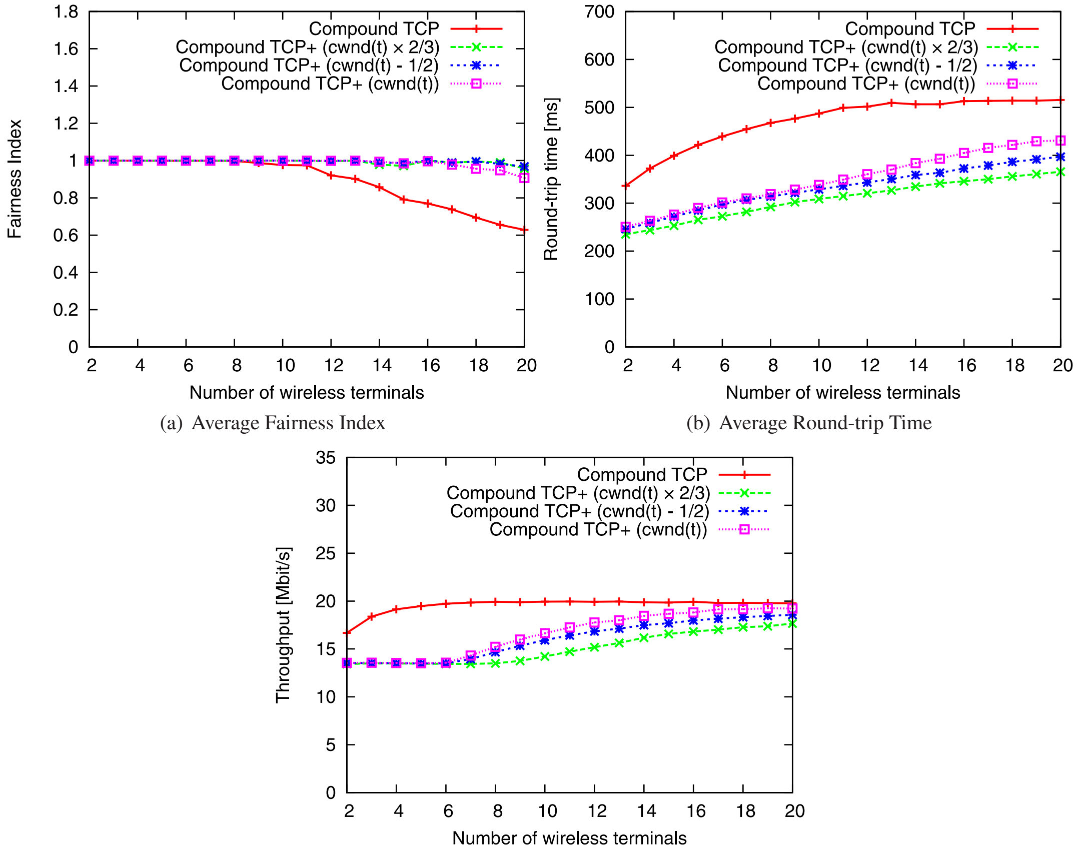
$$f = \frac{(\sum_{n=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2}. \quad (1 \leq i \leq n)$$

Fairness Index f is greater than or equal to zero and less than or equal to one. A higher value of fairness is indicated by a value of f closer to one.

Figure 6 shows simulation results for different values of the number of Compound TCP and Compound TCP+ connections. Figure 6 (a)–(c) show (a) the average fairness index, (b) the average round-trip time, and (c) the average total throughput, respectively. From Fig. 6 (a), it can be observed that when fewer wireless terminals are connected to the access point, the fairness of Compound TCP and Compound TCP+ are approximately equal. However, with an increase in the number of wireless terminals, the fairness of Compound TCP degrades. On the contrary, except in the case where Compound TCP+ uses Eq. (9), Compound TCP+ achieves high fairness. This is because when the delay window size is zero, the Compound TCP+ decreases the loss window size without a packet loss occurrence. This means the sending-out window size becomes small, and Compound TCP+ stops filling the buffer at the access point. As a result, Compound TCP+ achieves high fairness. Further, when Eq. (9) is used, it is found that fairness slightly worsens with increase in the number of wireless terminals. When the number of connections in a network increases, the state of the network approaches state of a packet loss occurrence. In such a case, Compound TCP+ with Eq. (9) continues to maintain the loss windows, thus leading to worse fairness. This result shows that although a packet loss does not occur in such a network state, the loss window should be decreased in order to improve the fairness.

Figure 6 (b) shows that the average round-trip time of Compound TCP+ is very small when compared with the average round-trip time of Compound TCP. Compound TCP continues to increase the loss window until it detects a packet loss. An increase in the loss window causes many packets to be stored in an access point buffer. Therefore, the average round-trip time of Compound TCP becomes extremely high. This high round-trip time can be a serious problem for interactive applications, but it is insignificant for applications such as a file transfer. In contrast, when a network is considered to be close to the state where a packet loss occurs, congestion control of Compound TCP+ avoids filling the buffer at an access point. Hence, the average round-trip time of Compound TCP+ becomes low.

Figure 6 (c) shows that the total throughput of Compound TCP+ is lower than the throughput of Compound TCP. Compound TCP increases the loss window until a packet loss occurs. On the contrary, when the delay window size is zero, Compound TCP+ decreases the loss window or maintains it at a constant size. Therefore, Compound TCP+ has a lower throughput value. In particular, when Eq. (7) is used, the total throughput is lower than that in the other cases. This is because Compound TCP with Eq. (7) decreases the loss window size multiplicatively. The rapid decrease in the size of the loss window when the size of the delay window is zero is good for fairness. However, in compensation for the fairness, the total throughput is decreased.



(a) Average Fairness Index

(b) Average Round-trip Time

(c) Average Total Throughput

Fig. 6 Simulation results for Compound TCP+ in wireless LAN.

From the above result, we conclude that Eq. (8), which achieves high throughput and high fairness levels, is appropriate as the function $f(cwnd)$ in wireless LAN with a large delay. However, depending on the network or the congestion state, the method for decreasing a loss window and the parameter of each method may not be appropriate. We will investigate methods for decreasing the loss window and the parameter value of each method in many network environments in a future work.

Next, we qualitatively examine fairness in the case where TCP and Compound TCP+ compete. TCP uses packet loss as the index of congestion, and it increases the congestion window size until a packet loss occurs. In contrast, Compound TCP+ decreases the loss window size when the network is considered to be in the state where packet loss can occur, even if a packet loss does not actually occur. When Compound TCP+ competes with TCP, packets are accumulated at the router or access point buffer by the TCP connection, and the round-trip time increases. As for a Compound TCP+ connection, the delay window decreases with the increase in the round-trip time. Moreover, as a result of the continuing decrease in the delay window to zero, the loss window decreases. Therefore, since Compound TCP+ does not take the bandwidth of the existing TCP connection that is competing with it, it can be said that Compound TCP+ implements TCP-friendly congestion control. The same is true when Compound TCP+ and Compound TCP compete. It should be noted that we intend to evaluate the fairness between Compound TCP+ and other transport-layer protocols quantitatively in our future work.

Table 4 Parameters used in simulations for a high-speed network.

Number of end-hosts	2
Two-way propagation delay	220 [ms]
Link bandwidth	10 [Gbit/s]
Packet length	1,500 [byte]

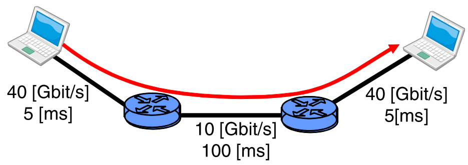


Fig. 7 Simulation model for a high-speed network.

Finally, using simulations, we show that in a high-speed network, Compound TCP+ can obtain a throughput similar to the throughput in Compound TCP. The network model of the simulation is shown in **Fig. 7**. In this model, one Compound TCP or Compound TCP+ connection performs file transfer. The bandwidth of all links is 10 [Gbit/s]. The two-way propagation delay between the sender and receiver hosts is 220 [ms]. The simulation runs for 100 [s]. The network parameters used are summarized in **Table 4**.

Figure 8 shows the throughput of Compound TCP and Compound TCP+. For both protocols, it can be seen that the throughput increases to 10 [Gbit/s] in the slow-start phase. The slow-start phase is completed when a packet loss occurs, and the throughput promptly decreases. However, the throughput of Compound TCP and Compound TCP+ increases gradually and saturates at 10 [Gbit/s]. From this simulation result, it can be concluded that

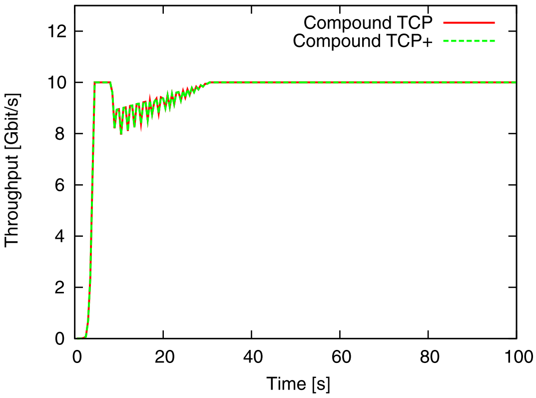


Fig. 8 Throughputs of Compound TCP and Compound TCP+ in a high-speed network.

similar to Compound TCP, Compound TCP+ can achieve high throughput in a high-speed wired network.

5. Conclusion and Future Work

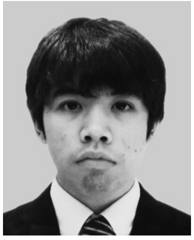
In this paper, we proposed Compound TCP+, a protocol that improves the throughput fairness of Compound TCP connections in wireless LAN, and we demonstrated its effectiveness. First, we evaluated the performance of Compound TCP in wireless LAN by simulation. It was observed that the loss-based congestion control mechanism (same mechanism as used by TCP) performed by Compound TCP does not achieve fairness in throughput. In order to solve this problem, Compound TCP+ was proposed. It decreases the loss window when it anticipates a state where a packet loss may occur, without actual packet loss occurrence. Using simulation, we showed that Compound TCP+ connections have high fairness in wireless LAN. Further, we also demonstrated that Compound TCP+ achieves high throughput in a high-speed wired network.

Compound TCP+ proposed in this paper offers a solution to the problem of throughput unfairness between up flows in wireless LAN. Therefore, it would be interesting to extend Compound TCP+ for solving the problem of throughput unfairness when up and down flows coexist in wireless LAN. Moreover, in this paper, only a qualitative evaluation was conducted for a network where TCP and Compound TCP+ or Compound TCP and Compound TCP+ competed. It is important to quantitatively evaluate the fairness between Compound TCP+ and other transport-layer protocols. It is also important to conduct a performance evaluation of Compound TCP+ in various network environments.

Acknowledgments This work was partly supported by Dayz Inc.

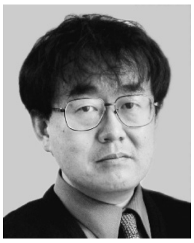
References

[1] Katabi, D., Handley, M. and Rohrs, C.: Congestion Control for High Bandwidth-Delay Product Networks, *Proc. ACM SIGCOMM 2002*, pp.89–102 (2002).
[2] Floyd, S.: HighSpeed TCP for Large Congestion Windows, RFC3649 (2003).
[3] Kelly, T.: Scalable TCP: Improving Performance in Highspeed Wide Area Networks, *Proc. PFLDnet 2003*, pp.82–91 (2003).
[4] Xu, L., Harfoush, K. and Rhee, I.: Binary Increase Congestion Control (BIC) for Fast Long-Distance Networks, *Proc. IEEE INFOCOM 2004*, pp.2514–2524 (2004).
[5] Ha, S., Rhee, I. and Xu, L.: CUBIC: A New TCP-friendly High-Speed TCP Variant, *ACM SIGOPS Operating ASystems Review*, Vol.42, pp.64–74 (2008).
[6] Wei, D.X., Jin, C., Low, S.H. and Hegde, S.: FAST TCP: Motivation, architecture, algorithms, performance, *IEEE/ACM Trans. Networking*, Vol.14, pp.1246–1259 (2006).
[7] Shimonishi, H., Higuchi, J., Yoshikawa, T. and Iwata, A.: A Congestion Control Algorithm for Data Center Area Communications, *Proc. IEEE 2008 International Communication Quality and Reliability Workshop* (2008).
[8] Hasegawa, G., Kurata, K. and Murata, M.: Analysis and Improvement of Fairness between TCP Reno and Vegas for Deployment of TCP Vegas to the Internet, *Proc. ACM SIGMETRICS 2000*, pp.134–144 (2000).
[9] Mascolo, S., Casetti, C., Gela, M., Sanadidi, M.Y. and Wang, R.: TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links, *Proc. ACM MobiCom 2001*, pp.287–297 (2001).
[10] Grieco, L.A. and Mascolo, S.: Performance Evaluation and Comparison of Westwood+, New Reno, and Vegas TCP Congestion Control, *ACM Computer Communication Review*, Vol.3, pp.25–38 (2004).
[11] Meiss, M.R.: Tsunami: UDP Protocol, available from ⟨http://tsunami-udp.sourceforge.net/⟩.
[12] Gu, Y. and Grossman, R.L.: UDT: UDP-Based Data Transfer for High-Speed Wide Area Networks, *Computer Networks*, Vol.51, pp.1777–1799 (2007).
[13] Baiocchi, A., Castellani, A.P. and Vacirca, F.: YeAH-TCP: Yet Another Highspeed TCP, *Proc. PFLDnet 2007*, pp.37–42 (2007).
[14] Kaneko, K., Fujikawa, T., Su, Z. and Katto, J.: TCP-Fusion: A Hybrid Congestion Control Algorithm for High-Speed Networks, *Proc. PFLDnet 2007*, pp.31–36 (2007).
[15] Shimonishi, H., Hama, T. and Murase, T.: TCP-Adaptive Reno for Improving Efficiency-Friendliness Tradeoffs of TCP Congestion Control Algorithm, *Proc. PFLDnet 2006*, pp.87–91 (2006).
[16] Tan, K., Song, J., Zhang, Q. and Sridharan, M.: Compound TCP: A Scalable and TCP-Friendly Congestion Control for High-speed Networks, *Proc. PFLDnet 2006* (2006).
[17] Tan, K., Song, J. and Zhang, Q.: A Compound TCP Approach for High-speed and Long Distance Networks, *Proc. IEEE INFOCOM 2006*, pp.1–12 (2006).
[18] Pilosof, S., Ramjee, R., Raz, D., Shavitt, Y. and Sinha, P.: Understanding TCP Fairness over Wireless LAN, *Proc. IEEE INFOCOM 2003*, pp.863–872 (2003).
[19] Wang, C. and Tang, W.: A Probability-Based Algorithm to Adjust Contention Window in IEEE 802.11 DCF, *Proc. ICCCAS 2004*, pp.418–422 (2004).
[20] Yang, Y., Wang, J. and Kravets, R.: Distributed Optimal Contention Window Control for Elastic Traffic in Wireless LANs, *Proc. IEEE INFOCOM 2005*, pp.35–46 (2005).
[21] Hirano, Y. and Murase, T.: Uplink TCP Traffic Control with Monitoring Downlink Buffer for Throughput Fairness over Wireless LANs, *Proc. PIMRC 2009*, pp.737–741 (2009).
[22] Hashimoto, M., Hasegawa, G. and Murata, M.: A Transport-Layer Solution for Alleviating TCP Unfairness in a Wireless LAN Environment, *IEICE Trans. Comm.*, Vol.E94-B, pp.765–776 (2011).
[23] Abeysekera, B.A.H.S., Matsuda, T. and Takine, T.: Dynamic Contention Window Control to Achieve Fairness Between Uplink and Downlink Flows in IEEE 802.11 WLANs, *Proc. IEEE WCNC'07*, pp.2109–2114 (2007).
[24] Abeysekera, B.A.H.S., Matsuda, T. and Takine, T.: Dynamic Contention Window Control Scheme in IEEE 802.11e EDCA-Based Wireless LANs, *IEICE Trans. Comm.*, Vol.E93-B, pp.56–64 (2010).
[25] Wu, Q., Gong, M. and Williamson, C.: TCP Fairness Issues in IEEE 802.11 wireless LANs, *Journal Computer Communications*, Vol.31, pp.2150–2161 (2008).
[26] Leith, D.J., Andrew, L.L.H., Quetchenbach, T. and Shorten, R.N.: Experimental Evaluation of Delay/Loss-based TCP Congestion Control Algorithms, *Proc. PFLDnet 2008* (2008).
[27] Tan, K., Song, J., Zhang, Q. and Sridharan, M.: CTCP-TUBE: Improving TCP-Friendliness Over Low-Buffered Network Links, *Proc. PFLDnet 2008* (2008).
[28] Hashimoto, M., Hasegawa, G. and Murata, M.: Performance Evaluation and Improvement of Hybrid TCP Congestion Control Mechanisms in Wireless LAN Environment, *Proc. ATNAC 2008*, pp.367–372 (2008).
[29] SAMAN, CONSER and ACIRI: The network simulator – ns2, available from ⟨http://www.isi.edu/nsnam/ns/⟩.
[30] Jain, R.: Throughput Fairness Index: An Explanation, *ATM Forum Contribution 99-0045* (1999).



Hiroki Oda received his B.E and M.E. degrees from Osaka Electro-Communication University, Japan, in 2009 and 2011, respectively. He is currently a doctoral student at the Graduate School of Information Science and Arts, Osaka Electro-Communication University. His research work is in the area of transport architecture. He is a student member of IPSJ, IEEE and IEICE.

Hiroyuki Hisamatsu received his M.E. and Ph.D. degrees from Osaka University, Japan, in 2003 and 2006, respectively. He is currently an Associate Professor of Department of Computer Science, Osaka Electro-Communication University. His research work is in the area of performance evaluation of TCP/IP networks. He is a member of IEEE and IEICE.



Hiroshi Noborio was born in Osaka Prefecture, Japan, in 1958. He graduated in computer science from Shizuoka University in 1982 and received his Dr. Eng. degree (equivalent of Ph.D.) in mechanical engineering from Osaka University in 1987. From 1987 to 1988, he worked as Assistant Professor in the Department of Mechanical Engineering, Faculty of Engineering Science, Osaka University. He was a Humboldt Scholar and a visiting researcher in the Department of Electrical Engineering, Technical University of Munich. He is currently the Dean of Faculty of Information Science and Arts and also a Professor in the Department of Computer Science, Osaka Electro-Communication University. His research interests span robotics, computer graphics, virtual reality, and medical engineering. He has published over 100 scientific papers and papers of conference proceedings.