IEICE TRANS. COMMUN., VOL.E93–B, NO.5 MAY 2010

PAPER Special Section on Technology and Architecture for Sustainable Growth of the Internet

Kyushu-TCP: Improving Fairness of High-Speed Transport Protocols*

Suguru YOSHIMIZU[†a)], ***Student Member***, **Hiroyuki KOGA**[†b)], ***Member***, **Katsushi KOUYAMA**[††c)], ***Nonmember***, **Masayoshi SHIMAMURA**[†††d)], **Kazumi KUMAZOE**[†††e)], ***and*** **Masato TSURU**[††††f)], ***Members***

SUMMARY With the emergence of bandwidth-greedy application services, high-speed transport protocols are expected to effectively and aggressively use large amounts of bandwidth in current broadband and multimedia networks. However, when high-speed transport protocols compete with other standard TCP flows, they can occupy most of the available bandwidth leading to disruption of service. To deploy high-speed transport protocols on the Internet, such unfair situations must be improved. In this paper, therefore, we propose a method to improve fairness, called Kyushu-TCP (KTCP), which introduces a non-aggressive period in the congestion avoidance phase to give other standard TCP flows more chances of increasing their transmission rates. This method improves fairness in terms of the throughput by estimating the stably available bandwidth-delay product and adjusting its transmission rate based on this estimation. We show the effectiveness of the proposed method through simulations.

key words: *high-speed transmission control protocol, fairness, congestion control, and elephant flow*

1. Introduction

Recently, the Internet has rapidly evolved into broadband and multimedia networks. Despite growth, standard Transmission Control Protocol (TCP) [1] is still employed as the standard transport protocol. However, as the capacity of network links has grown and bandwidth-greedy application services have emerged, such as peer-to-peer (P2P) file sharing, grid computing, and network storage systems, standard TCP cannot perform well within the Internet. In particular, standard TCP cannot effectively use the large bandwidth available in such networks because of its congestion avoidance algorithm. Therefore, a wide variety of high-speed transport protocols designed to aggressively increase their throughput have been developed and evaluated through computer simulations and test-bed experiments [2]–[4].

Manuscript received September 15, 2009.
Manuscript revised December 28, 2009.

[†]The authors are with the Dept. of Information and Media Engineering, University of Kitakyushu, Kitakyushu-shi, 808–0135 Japan.
[††]The author is with Kyushu Electric Power Company, Inc., Fukuoka-shi, 810-8720 Japan.
[†††]The authors are with Network Design Research Center, Kyushu Institute of Technology, Iizuka-shi, 820-8502 Japan.

*An earlier version of this work was presented at PacRim2009, Aug. 2009.

a) E-mail: coolwind@net.is.env.kitakyu-u.ac.jp
b) E-mail: koga@net.is.env.kitakyu-u.ac.jp
c) E-mail: katsushi_kouyama@kyuden.co.jp
d) E-mail: shimamura@ndrc.kyutech.ac.jp
e) E-mail: kuma@ndrc.kyutech.ac.jp
f) E-mail: tsuru@ndrc.kyutech.ac.jp

DOI: 10.1587/transcom.E93.B.1104

However, high-speed transport protocols also still have significant problems regarding the unfairness of throughput performance with competing flows. When high-speed transport protocols compete with other standard TCP flows, they occupy most of the available bandwidth because they aggressively increase their congestion window size (cwnd) and non-aggressively decrease it. Furthermore, if these aggressive protocols are applied to bandwidth-greedy applications such as P2P file sharing applications, they significantly degrade the quality of services. In particular, versus aggressive protocols, standard TCP will be prevented from increasing its cwnd, and this problem leads to the disruption of service.

To deploy high-speed transport protocols on the Internet, the unfair bandwidth usage mentioned above must be improved. In particular, we focus on the unfairness problem between high-speed and standard TCP flows with same round-trip time (RTT) rather than RTT-fairness issues that short RTT flows receive more bandwidth than longer ones [4]. To alleviate such unfairness, standard TCP must achieve intrinsic performance even when standard TCP flows compete with other high-speed transport protocol flows. In other words, high-speed transport protocols must enable competing standard TCP flows to obtain the same throughput performance as they compete only with standard TCP flows. In this paper, we use such features as *fairness*. To improve fairness, high-speed transport protocols should make the non-congested period longer and give other standard TCP flows more chances of increasing their cwnd. Namely, high-speed transport protocols should restrain the increase in their cwnd during a period of congestion without entirely losing their aggressiveness. We believe this change should ensure that other standard TCP flows obtain a large bandwidth.

In this paper, we propose a novel congestion avoidance algorithm called Kyushu-TCP (KTCP) to improve fairness in terms of throughput and maintain high throughput performance among flows of high-speed transport protocols and other standard TCP flows. During its congestion avoidance phase, KTCP estimates the stably available bandwidth-delay product and then adjusts its cwnd based on this estimation. Using this behavior, high-speed transport protocols applying KTCP allow other standard TCP flows to increase their throughput, and the frequency of congestion events decreases. In addition, standard TCP flows constantly attain high throughput performance even when they compete with other, more aggressive protocols. Note that KTCP is an add-on feature of congestion avoidance mechanisms, meaning

that KTCP can be combined with any window-based TCP congestion control variants. We evaluate the effectiveness of KTCP in terms of fairness through simulations.

The rest of the paper is organized as follows. Section 2 introduces related work on high-speed transport protocols. Section 3 proposes KTCP and describes its behavior. Section 4 presents our simulation model and simulation scenarios, and Sect. 5 shows and discusses the simulation results. Section 6 provides a conclusion by briefly summarizing the main points of the paper.

2. Related Work

As described in Sect. 1, various high-speed transport protocols with an improved congestion control algorithm have been proposed. They can aggressively increase their cwnd and improve their throughput performance. High-speed transport protocols are classified into loss-based protocols, delay-based protocols, and hybrid protocols. Loss-based protocols such as HSTCP [5], Scalable-TCP [6], H-TCP [7], and CUBIC [8] adjust its cwnd according to packet loss events similar to standard TCP. Delay-based protocols such as FAST [9] make use of RTT as a network congestion estimator. Hybrid protocols such as Compound TCP [10] and TCP-AReno [11] can adaptively switch their congestion control mode or TCP response function according to the congestion level measurement estimated from RTT. In this research, we employ HSTCP to apply the KTCP algorithm, since HSTCP uses simple Additive-Increase and Multiplicative-Decrease (AIMD) algorithm. It helps us to investigate the impact of KTCP algorithm plainly. CUBIC is also employed to evaluate the performance of KTCP, which is one of the protocols focus on the fairness performance. In this section, we describe a congestion control algorithm and the aggressiveness of HSTCP which is one of protocols KTCP is applied to.

HSTCP calculates cwnd differently from the way standard TCP does. In standard TCP, when a host receives an acknowledgment (ACK) packet, cwnd is increased by

$$w \leftarrow w + 1/w,$$

where w represents the current cwnd in the congestion avoidance phase. When packet drops occur, cwnd is decreased by half, given by

$$w \leftarrow 0.5w.$$

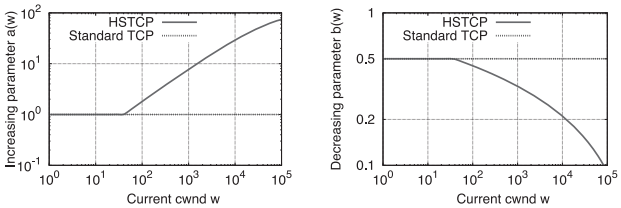
This AIMD strategy constantly increases and decreases cwnd, hence standard TCP cannot rapidly increase the cwnd up to a sufficient size on high bandwidth-delay product links.

To address this issue, two congestion control parameters are introduced in HSTCP to reconsider the simple and traditional AIMD strategy. One is an increasing parameter $a(w)$ and the other is a decreasing parameter $b(w)$, which are determined by the current cwnd w, respectively. In response to a single ACK packet, HSTCP adjusts its cwnd using these two parameters. HSTCP increases its cwnd, given by

$$w \leftarrow w + a(w)/w,$$

and decreases its cwnd, given by

$$w \leftarrow (1 - b(w))\, w.$$



(a) $a(w)$ (b) $b(w)$

Fig. 1 Two congestion control parameters in HSTCP.

Figures 1(a) and 1(b) show a relation between w and $a(w)$ or $b(w)$, respectively. Note that invariables for determining $a(w)$ and $b(w)$ are set to their default values described in Ref. [5]. If w is lower than a threshold W (a default value of 83), HSTCP uses the standard TCP congestion control algorithm. Namely, HSTCP sets $a(w)$ and $b(w)$ to 1 and 0.5, respectively. If w is higher than W, HSTCP uses a specialized algorithm for broadband networks. In this case, as w increases, HSTCP increases $a(w)$ for increasing its cwnd aggressively and decreases $b(w)$ for decreasing its cwnd non-aggressively.

The above modification of the congestion avoidance algorithm enables HSTCP to aggressively utilize network resources in broadband networks [12]. However, it causes unfairness in terms of throughput performance of HSTCP and other standard TCP flows because HSTCP suppresses the increase in throughput of other standard TCP flows [13].

3. Fairness Improvement Method

In this section, we propose KTCP to improve fairness in terms of throughput of high-speed and standard transport protocols. The basic idea of the proposed algorithm is to introduce a non-aggressive state in the congestion avoidance phase to give other standard TCP flows more chances of increasing their cwnd. If high-speed transport protocols continue to aggressively increase their cwnd based on their congestion avoidance algorithm, a burst of packet drops occurs in a short period, and hence other standard transport protocols cannot increase their cwnd sufficiently. The non-aggressive state in KTCP enables high-speed transport protocols to prevent significant fluctuation of the transmission rate derived from the feature of window-based flow control. This leads to a steady and fair bandwidth allocation in the networks. Note that KTCP can be applied to any kind of high-speed transport protocols employing window-based flow control.

We next explain the fundamental behavior of KTCP. Figure 2 illustrates the time-series behavior of the cwnd in

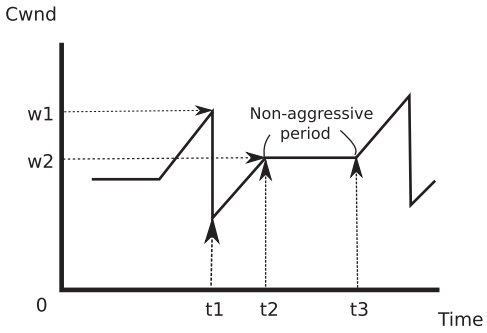


Fig. 2 Time-series behavior of cwnd in KTCP.

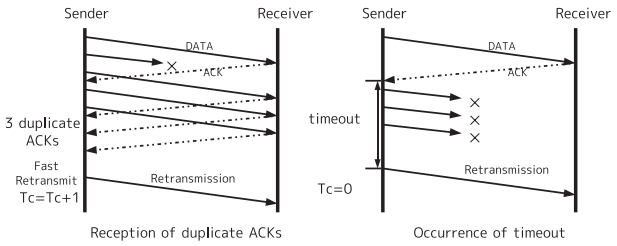


Fig. 3 Change of T_c due to retransmission.

KTCP. After a congestion event occurs at time $t1$, KTCP decreases its cwnd based on the original congestion avoidance algorithm and enters the congestion avoidance phase. In this phase, KTCP increases its cwnd up to the limit of $w2$, which is the estimated stably available bandwidth-delay product in the algorithm described later, and maintains that value during the non-aggressive period, which starts at time $t2$ and ends at time $t3$ in Fig. 2. The non-aggressive period in the congestion avoidance phase introduced by KTCP can prevent fluctuation of the transmission rate. This results in steady throughput and stimulates the throughput performance of other standard TCP flows. After that, KTCP controls its cwnd based on the original congestion avoidance algorithm. KTCP introduces the non-aggressive period only in the congestion avoidance phase. To achieve such congestion control, KTCP needs functions to estimate the stably available bandwidth-delay product and to manage the non-aggressive state.

In the proposed algorithm, we use a reduction ratio α, ranging from 0.0 to 1.0, to estimate the stably available bandwidth-delay product, $w2$, which is given by

$$w2 = w1 \times \alpha, \tag{1}$$

where $w1$ is the cwnd just before packet drops occur. Since the transmission rate determined by cwnd $w1$ exceeds the available bandwidth, we use $w2$ adjusted by α as the estimation of the stably available bandwidth-delay product. It is important to determine an appropriate value of α. If α is too small, KTCP cannot sufficiently increase its cwnd. On the other hand, if α is too large, KTCP cannot leave adequate bandwidth for other standard TCP flows. Therefore, we need to determine the appropriate value of α, which is left as a simulation parameter in this research.

Next, we need to determine the conditions for terminating the non-aggressive period. The non-aggressive period affects the throughput performance of high-speed transport protocols. If the non-aggressive period is too long, high-speed transport protocols cannot achieve sufficient throughput performance; if it is too short, the protocols may degrade the effectiveness of the proposed algorithm. Therefore, KTCP needs to maintain the non-aggressive period for an appropriate length of time based on network conditions. In this study, the non-aggressive period is terminated when the number of transmitted bytes exceeds a threshold without packet drops. The threshold B of termination is given by

$$B = T_c^2 \times w2, \tag{2}$$

where the variable T_c represents the number of times the fast retransmission is performed, caused by the reception of three duplicate ACKs without a timeout. Change of T_c is shown in Fig. 3. That is, T_c indicates the number of light congestion events. Equation (2) depends on the intensity of competition among high-speed and other transport protocols. Since KTCP employs a power estimation in the threshold B, although it keeps the first non-aggressive period short, corresponding to one RTT, KTCP lengthens the non-aggressive period depending on the increase in the number of congestion events and keeps it constant.

If congestion occurs again, KTCP should lengthen the non-aggressive period to restrain its transmission rate; if no congestion occurs, it should exit the non-aggressive period and increase its transmission rate to exploit the available bandwidth. Consequently, the non-aggressive period can be adjusted on the basis of network conditions. Moreover, KTCP also terminates the non-aggressive period when packet drops occur. The occurrence of packet drops means that its current transmission rate exceeds actual available bandwidth, so that KTCP should immediately decrease its transmission rate based on the original congestion avoidance algorithm. Then, KTCP increases its cwnd up to the limit of the $w2$ value updated by this packet drop. At that time, if KTCP enters the slow-start phase, it resets T_c, and then the original congestion control algorithm is performed.

4. Simulation Environment

In this section, we show the simulation environment for evaluating the performance of the proposed method. We used VINT network simulator NS version 2 [14] after adding the proposed method. Figure 4 shows our simulation model. In this simulation, one or two HSTCP flows compete with single or multiple SACK flows. To evaluate the fairness, we compare the throughput performance of an HSTCP flow and the total throughput performance of 20 TCP with selective acknowledgment option (SACK) [15] flows which is approximately the same throughput performance as that of one HSTCP flow as shown in Fig. 5. In addition, CUBIC is employed as high-speed transport protocols. Note that HSTCPs and CUBICs, used in our simulations, also employ

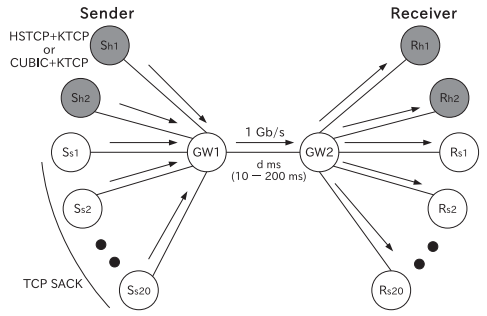


Fig. 4 Simulation topology.

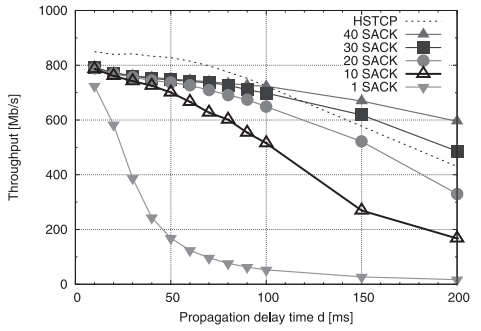


Fig. 5 Total throughput of multiple SACK flows.

SACK option.

Sender nodes S_{h1} and S_{h2} employing HSTCP, CUBIC, HSTCP+KTCP or CUBIC+KTCP, which means KTCP is applied, transmit data to corresponding receiver nodes R_{h1} and R_{h2} through gateway nodes GW_1 and GW_2, respectively. Similarly, sender nodes S_{s1} to S_{s20} employing TCP SACK transmit data to corresponding receiver nodes R_{s1} to R_{s20}, respectively. The bottleneck link between gateway nodes GW_1 and GW_2 has a bandwidth of 1 Gb/s and a propagation delay time d between gateway nodes GW_1 and GW_2 ranging from 10 to 200 ms. All other links have a bandwidth of 10 Gb/s and a propagation delay time of 3 ms. Each node except the end nodes employs a buffer of 200 packets. Senders and receivers employ a buffer with infinite capacity to avoid buffer overflows at end nodes. The TCP packet size is set to 1500 bytes. In addition, the parameter α in KTCP is variably set to 0.70, 0.80, 0.90, and 0.95. KTCP enters the non-aggressive state when HSTCP uses the specialized algorithm (i.e., the threshold of HSTCP exceeds W) described in Sect. 2. In this simulation, HSTCP can perform the specialized algorithm in all ranges of the propagation delay time.

Individual simulation experiments run for 300 seconds each. Note that, for performance evaluation of each flow, we use "throughput performance" as TCP goodput averaged over time (i.e., from the start time of the flow to the end of simulation run) and over 10 runs with different random seeds. In addition, we use "normalized throughput" of SACK flows as another evaluation index. Normalized throughput is calculated by the ratio of the throughput performance of x SACK flows competing with y high-speed transport protocol flows and the throughput performance of x SACK flows competing with other y SACK flows. The value of 1 means that high-speed transport protocols achieve good fairness. Each flow randomly starts in the range from 0 to 5 seconds to prevent any global synchronization phenomena of TCP.

5. Simulation Results

In this section, we examine the performance of the proposed method, KTCP, using the following two evaluation indices. One is fairness in terms of throughput of SACK and HSTCP or HSTCP+KTCP flows. We investigate the effect of the non-aggressive state introduced by KTCP. The other is throughput performance in a situation where multiple HSTCP or HSTCP+KTCP flows compete with each other. We investigate the impact of the non-aggressive state in KTCP on the throughput performance.

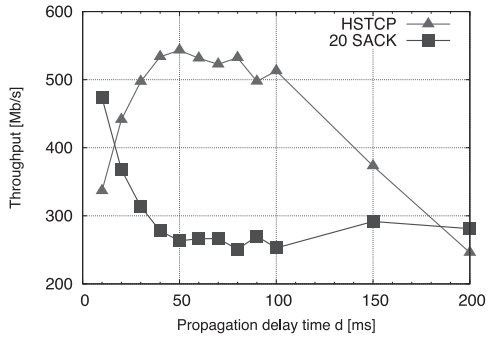
5.1 Fairness

In the following subsections, we discuss the impact of the proposed method on fairness in terms of throughput between HSTCP and SACK flows as well as CUBIC and SACK flows. Simulation parameters used in this simulation are the reduction ratio α and the number of HSTCP flows.

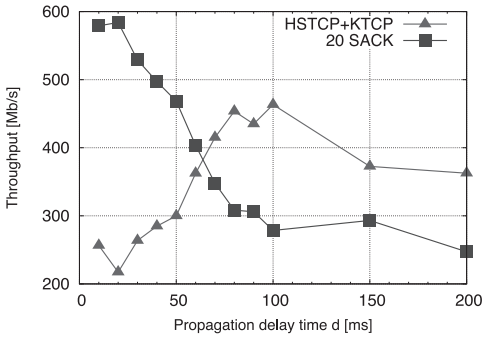
5.1.1 Impact of Parameter α

In this subsection, we show results in case that an HSTCP or HSTCP+KTCP flow competes with 20 SACK flows. First, in Fig. 6, we compare the throughput performance of an HSTCP flow and the total throughput of 20 SACK flows when d ranges from 10 to 200 ms. Figure 6(a) shows the throughput performance of SACK flows and HSTCP without KTCP, while Figs. 6(b), 6(c), and 6(d) show that of SACK flows and HSTCP+KTCP with the parameter α set to 0.80, 0.90, and 0.95, respectively. From Fig. 6(a), the original HSTCP without KTCP achieves excellent throughput, whereas SACK attains poor throughput in a wide range of d (especially, approximately from 50 to 100 ms). However, in Figs. 6(b), 6(c), and 6(d), HSTCP+KTCP restrains its throughput, so that SACK can attain a better throughput than that in Fig. 6(a). In particular, when the parameter α is 0.90, the difference of throughput performance between HSTCP and SACK flows decreases compared to the result of the original HSTCP. In this case, the throughput performance of SACK flows is approximately equal to that of HSTCP+KTCP in short propagation delay time d.

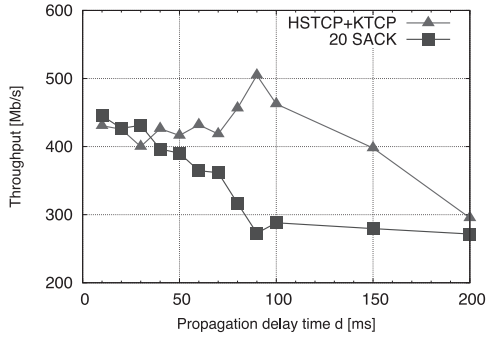
To investigate fairness between HSTCP+KTCP and SACK flows, in Fig. 7 we show the normalized throughput, calculated by the ratio of the throughput performance of HSTCP+KTCP flows and the total throughput performance of all flows including HSTCP+KTCP flows. The value 0.5 means each type of flow had a fair share of the available bandwidth. If α is set to a small value, HSTCP+KTCP cannot attain enough throughput because it underestimates



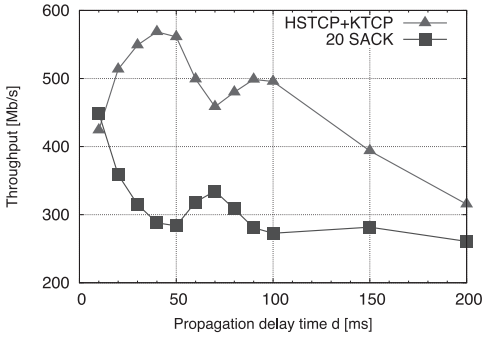
(a) Original HSTCP



(b) HSTCP+KTCP: α=0.80



(c) HSTCP+KTCP: α=0.90



(d) HSTCP+KTCP: α=0.95

Fig. 6 Throughput performance for an HSTCP flow and SACK flows.

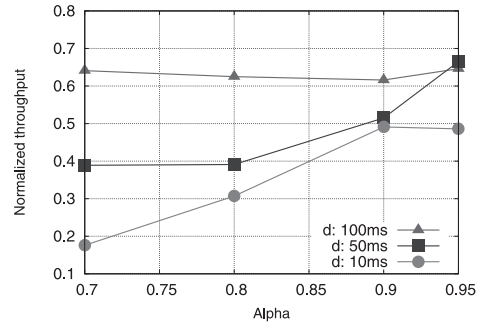


Fig. 7 Normalized throughput of an HSTCP+KTCP flow.

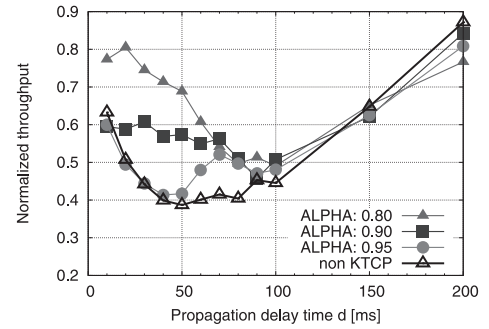
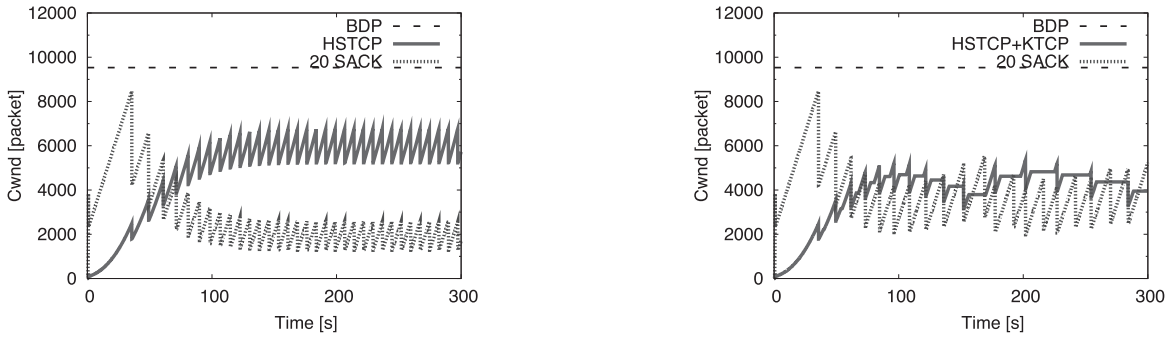


Fig. 8 Normalized throughput of SACK flows.

the available bandwidth. On the other hand, if α is set to a large value, HSTCP+KTCP can attain a high throughput because SACK cannot work well in high bandwidth-delay product links. In addition, if d is short such as 10 ms, HSTCP+KTCP with a small value of α attains poor throughput because congestion events frequently occur and HSTCP+KTCP lengthens the non-aggressive period based on large value of T_c and small value of $w2$. On the other hand, if d is long such as 100 ms, HSTCP+KTCP with any values of α attains high throughput because congestion events rarely occur, hence it decreases the number of times the non-aggressive state is entered.

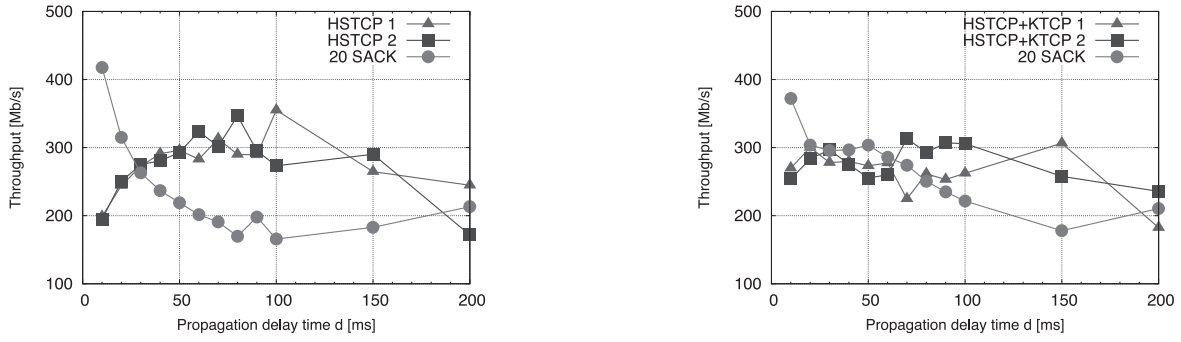
Next, we examine fairness of KTCP from the viewpoint of SACK flows. In Fig. 8, we show the normalized throughput calculated by the ratio of the intrinsic throughput performance of 20 SACK flows competing with an HSTCP or HSTCP+KTCP flow and the throughput performance of 20 SACK flows competing with another SACK flow. The higher normalized throughput in case of HSTCP+KTCP compared to original HSTCP implies that KTCP improves the throuput performance of SACK flows. When the parameter α is set to 0.80 or 0.90 in short d, the normalized throughput of SACK flows becomes high because KTCP frequently mitigates unnecessary competition. On the other hand, in long d, the normalized throughput does not increase compared to case of the original HSTCP because SACK flows cannot work well in high bandwidth-delay product links. From Figs. 7 and 8, considering the throughput performance of both HSTCP and SACK flows, the appropriate value of the parameter α is 0.90 in this simulation environment.

Finally, to investigate how KTCP improves fairness be-



(a) Original HSTCP (b) HSTCP+KTCP: α=0.90

Fig. 9 Time-series behavior of cwnd. (d: 50 ms)



(a) Original HSTCP (b) HSTCP+KTCP: α=0.90

Fig. 10 Throughput performance of multiple HSTCP flows and SACK flows.

tween HSTCP and SACK flows, we present the behavior of HSTCP+KTCP when the parameter α is appropriately set to 0.90. Figures 9(a) and 9(b) shows the time-series behavior of the total cwnd of 20 SACK flows and the cwnd of HSTCP or HSTCP flows, respectively. To provide a deeper understanding, we plot the upper bound of cwnd derived from a bandwidth-delay product (BDP) and a buffer size of routers. In Fig. 9(b), after both HSTCP+KTCP and SACK start the congestion avoidance phase, HSTCP+KTCP enters the non-aggressive period. Then, KTCP lengthens the non-aggressive period every time packet drops occur. Based on this behavior, KTCP restrains the aggressive increase of cwnd in HSTCP. Therefore, SACKs can increase their cwnd sufficiently so that they can attain a better throughput as results shown in Figs. 6(b) and 6(c) even if they compete with HSTCP flows when KTCP is applied.

5.1.2 Impact of Multiple HSTCP Flows

In this subsection, we show the effectiveness of KTCP in case that multiple HSTCP with or without KTCP flows compete with SACK flows. First, we compare the throuput performance of each of two HSTCP (with or without KTCP) flows and the total throughput performance of 20 SACK flows in Fig. 10. The propagation delay time d is varied from 10 to 200 ms.

Figure 10(a) shows the throughput performance of 20 SACK flows and two HSTCP flows without KTCP,

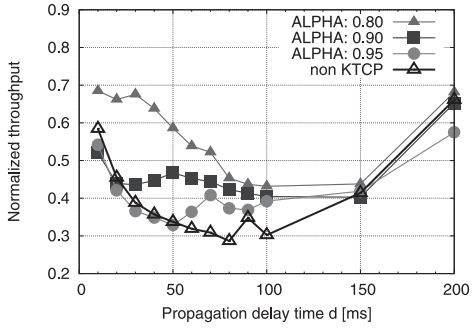


Fig. 11 Normalized throughput of SACK flows competing with multiple HSTCP flows.

while Fig. 10(b) shows that of 20 SACK flows and two HSTCP+KTCP flows with α set to 0.90. From Fig. 10(a), each original HSTCP flow attains a high throughput, whereas SACK flows attain a poor throughput. Moreover, HSTCP flows obtain too much of the bandwidth, particularly in the range of d from 50 to 150 ms. However, in Fig. 10(b), SACK flows improve in throughput over a wide range of d. Consequently, KTCP can work well even if multiple HSTCP flows compete with other standard TCP flows.

Next, we focus on the throughput performance of SACK flows. In Fig. 11, we show the normalized throughput obtained by calculating the ratio of the throughput of SACK flows competing with multiple HSTCP flows and in-

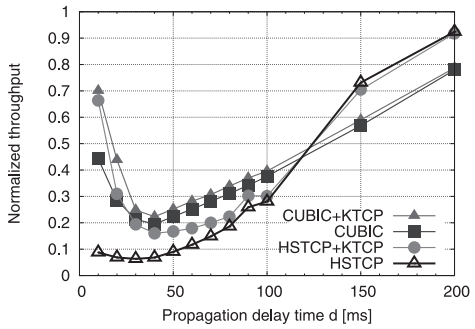


Fig. 12 Normalized throughput of a SACK flow.

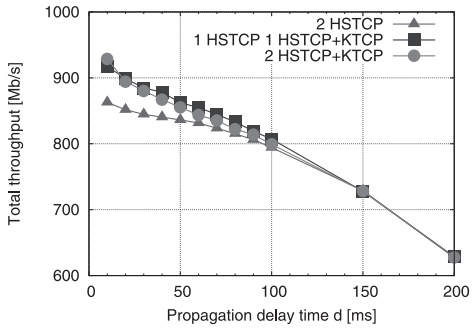


Fig. 13 Total throughput of multiple HSTCP flows.

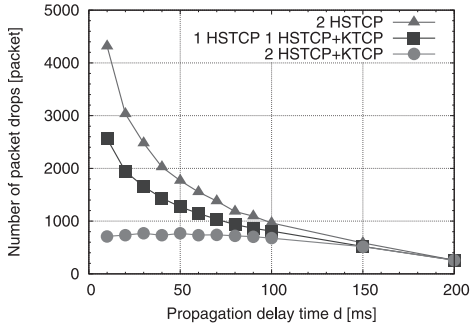


Fig. 14 Number of packet drops.

trinsic throughput of SACK flows competing with another two SACK flows. When the parameter α is 0.80 or 0.90, the normalized throughput becomes high compared with that of SACK flows competing with original HSTCP flows. From the above results, we show that KTCP can improve fair usage among multiple HSTCP and SACK flows.

5.1.3 Impact on Single SACK Flow

In this subsection, we focus on the effectiveness of KTCP on a high-speed transport flow competing with single SACK flow. Figure 12 shows the normalized throughput of a SACK flow calculated by the ratio of the throughput performance of a SACK flow competing with an HSTCP, CUBIC, HSTCP+KTCP or CUBIC+KTCP flow with α of 0.90 and the throughput performance of a SACK flow competing with another SACK flow. From Fig. 12, KTCP can improve fairness of HSTCP as similar to the case where HSTCP flows compete with multiple SACK flows. Moreover, KTCP can improve fairness of CUBIC in a wide range of d. In particular, the improvement of fairness becomes larger in shorter link delay time. Consequently, KTCP can improve fairness between a high-speed transport protocol flow and a SACK flow.

5.2 Throughput

In this section, we show results in case that two HSTCP or HSTCP+KTCP flows compete each other without SACK flows. First, we investigate the impact of using KTCP to the throughput performance of competition between multiple HSTCP flows. KTCP can suppress its own transmission rate (cwnd) to stimulate other flows by its non-aggressive period, but it likely degrades its throughput performance. In this section, to evaluate the throughput performance of KTCP flows, we consider the following three scenarios: competition between (1) two HSTCP flows, (2) one HSTCP and one HSTCP+KTCP flows, and (3) two HSTCP+KTCP flows. The evaluation index used in this section is the total throughput and the number of packet drops obtained by two flows.

Figure 13 shows the total throughput performance of two HSTCP or HSTCP+KTCP flows. When KTCP is applied to HSTCP, the total throughput performance can be improved as compared with HSTCP flows without KTCP in a wide range of d. This is because, HSTCP+KTCP can use the available bandwidth more effectively than original HSTCP by eliminating the fluctuation of cwnd with the introduction of the non-aggressive state. To investigate this phenomenon, we show the total number of packet drops in the same three scenarios in Fig. 14. KTCP can prevent a burst of packet drops over a wide range of d. Consequently, KTCP can also improve HSTCPs' throughput performance.

From the above results, we show that KTCP can maintain aggressiveness of HSTCP and use the large available bandwidth effectively even if two HSTCP flows compete each other.

6. Conclusion

In this paper, we proposed a novel add-on congestion avoidance mechanism, KTCP, which is intended to apply to high-speed transport protocols' congestion avoidance algorithms. KTCP can adaptively avoid aggressive increments of cwnd of high-speed transport protocol flows in order to improve the throughput fairness between those flows and co-existing standard TCP flows. In addition, KTCP can also mitigate unnecessary competition between co-existing high-speed transport protocols and thus decrease unnecessary packet drops, resulting in an efficient use of a large bandwidth of the high-speed networks. The main idea of the proposed algorithm is that KTCP restrains the cwnd of high-speed transport protocols based on network conditions. On the basis of this idea, KTCP introduces an non-aggressive state

into the congsetion avoidance algorithm, and this function aims to give other standard TCP flows more chances of increasing their cwnd and use much more bandwidth.

Through simulations, we confirmed the effect of the non-aggressive state on fairness in terms of throughput performance. Specifically, KTCP achieved good fairness regardless of the number of HSTCP flows, especially when employing the parameter $\alpha = 0.90$ in this simulation environment. Furthermore, KTCP maintains the aggressiveness of HSTCP and attains high throughput performance when HSTCP flows compete with each other.

In future work, we will evaluate the performance of other high-speed transport protocols when applying KTCP and will consider a more effective algorithm focused on appropriate determination of the parameters α and B to improve the response time to reach steady state.

Acknowledgment

This work was supported in part by the Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research (B) (21300024).

References

[1] J. Postel, "Transmission control protocol," IETF RFC793, Sept. 1981.
[2] S. Ha, L. Le, I. Rhee, and L. Xu, "Impact of background traffic on performance of high-speed TCP variant protocols," Elsevier Computer Networks, vol.51, no.7, pp.1748–1762, May 2007.
[3] K. Kumazoe, K. Kouyama, Y. Hori, M. Tsuru, and Y. Oie, "Transport protocol for fast long-distance networks: Evaluation of penetration and robustness on JGN II," Proc. International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet2005), Feb. 2005.
[4] S. Ha, Y. Kim, L. Le, I. Rhee, and L. Xu, "A step toward realistic evaluation of high-speed TCP protocols," Proc. International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet2006), Feb. 2006.
[5] S. Floyd, "HighSpeed TCP for large congestion windows," IETF RFC3649, Dec. 2003.
[6] T. Kelly, "Scalable TCP: Improving performance in highspeed wide area networks," Proc. International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet2003), Feb. 2003.
[7] D. Leith and R. Shorten, "H-TCP protocol for high-speed long distance networks," Proc. International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet2004), Feb. 2004.
[8] I. Rhee and L. Xu, "CUBIC: A new TCP-friendly high-speed TCP variant," Proc. International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet2005), Feb. 2005.
[9] D.X. Wei, C. Jin, S.H. Low, and S. Hegde, "FAST TCP: Motivation, architecture, algorithms, performance," IEEE/ACM Trans. Netw., vol.14, no.6, pp.1246–1259, Dec. 2006.
[10] K. Tan, J. Song, Q. Zhang, and M. Sridharan, "Compound TCP: A Scalable and TCP-friendly congestion control for high-speed networks," Proc. International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet2006), Feb. 2006.
[11] H. Shimonishi, T. Hama, and T. Murase, "TCP-adaptive reno for improving efficiency-friendliness tradeoffs of TCP congestion control algorithm," Proc. International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet2006), Feb. 2006.
[12] M. Hassan and R. Jain, High Performance TCP/IP Networking: Concepts, Issues, and Solutions, Prentice Hall, New Jersey, 2003.
[13] B. Even, Y. Li, and D. Leith, "Evaluating the performance of TCP stacks for high-speed networks," Proc. International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet2006), Feb. 2006.
[14] VINT Project, Network Simulator ns-2, Information available at http://www.isi.edu/nsnam/ns/
[15] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgment options," IETF RFC2018, Feb. 2006.

Suguru Yoshimizu received the B.E. and M.E. degrees in Information and Media Engineering from University of Kitakyushu, Fukuoka, Japan, in 2008 and 2010, respectively. Since 2010 in April, he has worked at Fujitsu Kyushu Systems Limited. His research interests include performance evaluation of transport protocols.

Hiroyuki Koga received the B.E., M.E. and D.E. degrees in computer science and electronics from Kyushu Institute of Technology, Japan, in 1998, 2000, and 2003, respectively. From 2003 to 2004, he was a postdoctoral researcher in the Graduate School of Information Science, Nara Institute of Science and Technology. From 2004 to 2006, he was a researcher in the Kitakyushu JGN2 Research Center, National Institute of Information and Communications Technology. From 2006 to 2009, he was an assistant professor in the Department of Information and Media Engineering, Faculty of Environmental Engineering, University of Kitakyushu, and then has been an associate professor in the same department since April 2009. His research interests include performance evaluation of computer networks, mobile networks, and communication protocols. He is a member of the IEEE.

Katsushi Kouyama received B.E. and M.E. degrees in Computer Science and Communication Engineering from Kyushu University in 1992 and 1994, respectively. He Joined KEPCO (Kyushu Electric Power Co., Inc.) in 1994, and transferred temporarily to Kyushu Telecommunication Network Co., Inc. in 1998, where he was involved in planning and designing FTTH services and networks. From 2004, he was engaged in research into end-to-end communication control technology as a guest researcher of NICT (National Institute of Information and Communications Technology). After returning to KEPCO in 2007, he is now mainly engaged in managing optical fiber leasing services.

Masayoshi Shimamura received the B.E. degree from the Faculty of Engineering of Chiba Institute of Technology, Chiba, Japan, in 2003, and the M.E. and Ph.D. degrees from the Graduate School of Information Science of Nara Institute of Science and Technology (NAIST), Nara, Japan, in 2005 and 2009, respectively. From 2005 to 2007, he was an encouragement researcher in the 21st Century COE Program "Ubiquitous Networked Media Computing" in the Graduate School of Information Science, NAIST. Since 2008, he has been a postdoctoral researcher at the Network Design Research Center of Kyushu Institute of Technology (KIT). His research interests include performance evaluation of networking systems. He is a member of the IEEE.

Kazumi Kumazoe received a B.E., M.E. and D.E. degrees in computer science from Kyushu Institute of Technology, Iizuka, Japan, in 1993, 1995 and 2007, respectively. She worked in Telecommunications at NEC Corperation (1995–2000), Japan Telecom Information Service Corperation (2000–2007) and Human Media Creation Center/KYUSHU (2007–2009). Since April 2009, she has been a post doctoral researcher at Network Design Research Center, Kyushu Institute of Technology. Her research interests are in various areas of computer networks, including transport layer protocols and network management.

Masato Tsuru received B.E. and M.E. degrees from Kyoto University, Japan in 1983 and 1985, respectively, and then received his D.E. degree from Kyushu Institute of Technology, Japan in 2002. He worked at Oki Electric Industry Co., Ltd. (1985–1990), Information Science Center, Nagasaki University (1990–2000), and Japan Telecom Information Service Co., Ltd./Telecommunications Advancement Organization of Japan (2000–2003). In 2003, he moved to the Department of Computer Science and Electronics, Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology as an Associate Professor, and then has been a Professor in the same department since April 2006. His research interests include performance measurement, modeling, and management of computer communication networks. He is a member of the ACM, IEEE, IPSJ, and JSSST.