IEICE TRANS. COMMUN., VOL.E89–B, NO.6 JUNE 2006

PAPER

TCP-STAR: TCP Congestion Control Method for Satellite Internet

Hiroyasu OBATA[†a)], **Kenji ISHIDA**[†], ***Members***, **Satoru TAKEUCHI**[††*], ***Nonmember***, ***and*** **Shouta HANASAKI**[††], ***Student Member***

SUMMARY Satellite Internet is one of the most important networks for emergency communications because of its tolerant of disasters such as earthquake. Therefore, satellite Internet has received considerable attention over recent years. However, most standard implementations of TCP congestion control method perform poorly in satellite Internet due to its high bit error rate and long propagation delay. This paper proposes a new TCP congestion control method called TCP-STAR to improve the throughput over satellite Internet. TCP-STAR has three new mechanisms, namely Congestion Window Setting (CWS) based on available bandwidth, Lift Window Control (LWC), and Acknowledgment Error Notification (AEN). CWS can resist the reduction of the transmission rate when data losses are caused by bit error. LWC is able to increase the congestion window quickly based on the estimated available bandwidth. AEN can avoid the reduction of the throughput by mis-retransmission of data. The mis-retransmission is caused by ack losses or delay. Simulations show that TCP-STAR can obtain the best throughput comparing with other TCP variants (TCP-J and TCP-WestwoodBR). Furthermore, we found that the fairness of TCP-STAR is a little lower than that of TCP-WestwoodBR. However, the fairness of TCP-STAR is equal to TCP-J.

key words: *TCP congestion control method, satellite Internet, throughput*

1. Introduction

Wireless access to the Internet is becoming extremely popular with the growth of the Internet. Additionally, the bandwidth of wireless networks has become broader in recent year. Thus, wireless networks have become an important infrastructure as the backup of optical networks in the case of disasters. In particular, satellite communication systems are expected as high-speed wireless infrastructures for the next generation global information networks, since the systems can provide fairly large capacity global networks rather simply and they are in nature tolerant against large-scale disasters on the earth such as earthquakes [1]. For example, WINDS (Wideband InterNetworking engineering test and Demonstration Satellite) [2] is developed in Japan as a gigabit wireless network.

TCP (Transmission Control Protocol) [3] is a reliable data transfer protocol used widely over the Internet for many applications such as FTP and HTTP. Furthermore, TCP is used in the various networks (e.g. optical networks, satellite networks, cellular networks, and so on) because TCP has been implemented in many information devices such as computers, PDAs, and cellular phones. However, most standard TCP implementations perform poorly in satellite Internet due to its high bit error rate and long propagation delay [4].

Several researches have proposed the congestion control methods to improve TCP performance in wireless networks [5]–[10]. In [10], the authors proposed the congestion control method for satellite Internet. However, the method requires modification at a sender node, a receiver node, and all intermediate routers. Thus, the method in [10] is not realistic. On the other hand, our proposal only requires modification of the sender-side TCP module. Therefore, it is realistic.

This paper proposes a new TCP congestion control method called TCP-STAR to improve the throughput over satellite Internet. TCP-STAR has three new mechanisms, namely (1) Congestion Window Setting (CWS) based on available bandwidth, (2) Lift Window Control (LWC), and (3) Acknowledgment Error Notification (AEN). CWS and LWC use the estimated available bandwidth for increasing the transmission rate. In order to avoid the mis-retransmission of data, AEN detects ack losses or delay using a probe segment. In CWS and LWC, we use ABE (Available Bandwidth Estimation) in TCP-J [5] as the bandwidth estimation method. We have also implemented TCP-STAR in the network simulator NS2 [11] and evaluated the performance of TCP-STAR. In the simulation, we have compared the performance of TCP-STAR with the one of TCP-WestwoodBR [7] and TCP-J. Simulations show that TCP-STAR improves the throughput comparing with other TCP variants. Furthermore, we found that the fairness of TCP-STAR is equal to TCP-WestwoodBR and TCP-J in the homogeneous environment (i.e. all connections use the same TCP version). However, in case of the heterogeneous environment (i.e. the different TCP versions coexist), the fairness of TCP-STAR is a little lower than that of TCP-WestwoodBR.

The rest of this paper is constructed as follows. We describe drawbacks of the congestion control method over satellite Internet and related works in Sect. 2. In Sect. 3, we propose a new TCP congestion control method. Section 4 includes evaluations and discussions on proposal based on the results from simulations. We summarize this paper and show future works at Sect. 5.

Manuscript received August 5, 2005.
Manuscript revised December 9, 2005.

[†]The authors are with the Faculty of Information Sciences, Hiroshima City University, Hiroshima-shi, 731-3194 Japan.
[††]The authors are with the Graduate School of Information Sciences, Hiroshima City University, Hiroshima-shi, 731-3194 Japan.
[*]Presently, with NTT DoCoMo Chugoku, Inc.
a) E-mail: obata@ce.hiroshima-cu.ac.jp

DOI: 10.1093/ietcom/e89–b.6.1766

2. Preliminary

2.1 Drawbacks of Congestion Window Control Method of Standard TCP

In satellite Internet, the congestion control method of standard TCP (i.e. Reno and NewReno[†]) has several drawbacks as the follows.

(a) The sender cannot enough use the available bandwidth, because the rate of the congestion window increment in Slow Start phase is very slow by the long propagation delay [4]. If a delayed ack mechanism is implemented in the receiver, the duration of Slow Start phase becomes more longer. Because an ack is sent after receiving two data in the delayed ack mechanism.

(b) Data losses often occur due to link errors in satellite links (i.e. congestion does not occur). However, the standard congestion control method decides that data losses are caused by the congestion. Therefore, the standard TCP decreases the congestion window unnecessarily. As a result, the throughput decreases drastically.

(c) In satellite links, RTT (Round Trip Time) is almost stable. Therefore, RTO (Retransmission TimeOut) tends to get up close the RTT (see Eq. (1)) [12], [13].

$$\lim_{t\to\infty} RTO_t = RTT \tag{1}$$

So, if RTT spikes up (this phenomenon is caused by the queuing delay or the overload of the sender and the receiver), the arrival time of ack delays and RTT exceeds RTO easily. Therefore, the sender retransmits data needlessly. After the mis-retransmission of data, the sender sets the congestion window as one data segment size and executes Slow Start phase. As a result, the throughput decreases drastically in spite of no data losses.

2.2 Related Works

Several articles modified the congestion control method in order to solve the above drawbacks [5]–[7], [10], [13].

TCP-J [5] and TCP-Westwood [6]: TCP-J and TCP-Westwood use estimation techniques of end-to-end bandwidth. They estimate the available bandwidth by using the received acks and adaptively set *cwnd* (congestion window size) and *ssthresh* (Slow Start threshold) after data losses by taking into account the estimated bandwidth. In [5], the authors propose ABE (Available Bandwidth Estimation) as the bandwidth estimation method. The performance of TCP-J is better than that of TCP-Westwood by adopting ABE. However, these two methods cannot respond the drawback (a)(c).

TCP-WestwoodBR [7]: TCP-WestwoodBR is an extension of TCP-Westwood and addresses performance problems in heavy error environments such as satellite networks. TCP-WestwoodBR has three sender-side modifications: Bulk Repeat (retransmitting all outstanding packets immediately when possible multiple losses are detected in the same window), Fixed Retransmission Timeout (using a fixed timeout value instead of exponential backoff when consecutive losses occur), and Intelligent Window Adjustment (keeping *cwnd* in spite of losses). However, this method cannot also respond the drawback (c).

TCP-Peach [10]: TCP-Peach uses two new methods, namely Sudden Start and Rapid Recovery, which replace Slow Start and Fast Recovery, respectively. These new methods are based on the use of dummy segments that are low-priority segments generated by the sender. If network is congested, the dummy segments are discarded at the router. The purpose of dummy segment is probing available bandwidth. When data losses occur, TCP-Peach sets the congestion window based on the estimated bandwidth. However, TCP-Peach requires modification of all nodes including the intermediate nodes. If satellite Internet includes the terrestrial networks (the Internet), all routers in the terrestrial networks are also required the modification. Since it is difficult to modify all routers in the networks in order to deal with the dummy segment, the method in [10] is not realistic.

Enhanced TCP [13]: The authors propose a congestion control method taking into account the mis-retransmission of data by the ack delay. When RTO timer expires, the method measures RTT of retransmitted data in order to judge whether ack is delayed. If the timer expiration is caused by the ack delay, Slow Start threshold is set to the values before RTO timer expiration. As a result, the congestion control method can minimize the decrease of throughput by mis-retransmission of data. However, this method cannot respond the drawback (a)(b) and ack losses case of (c).

Therefore, the congestion control method which solves the drawbacks (a)(b)(c) of TCP in satellite Internet and requires only modification of the sender node does not existing yet [14], [15].

3. TCP-STAR

To cope with the drawbacks (a)(b)(c) of the existing methods, we propose a new congestion control method TCP-STAR. The proposed method only requires sender-side modification.

TCP-STAR consists of the following three new mechanisms; Congestion Window Setting (CWS) based on avail-

[†]TCP-Reno and TCP-NewReno are widely used in the current Internet.

able bandwidth, Lift Window Control (LWC), and Acknowledgment Error Notification (AEN). In CWS and LWC, TCP-STAR uses ABE (Available Bandwidth Estimation) in TCP-J as the available bandwidth estimation method, because the authors in [5] show that ABE estimates the available bandwidth accurately.

Here, the bandwidth estimation method is one of the TCP-STAR's modules and it is separated from CWS, LWC, and AEN. Therefore, if there is a bandwidth estimation method which accuracy is better than ABE, TCP-STAR can use it instead of ABE.

We describe the three mechanisms in the following subsections in more detail.

3.1 CWS: Congestion Window Setting Based on Available Bandwidth Estimation

CWS avoids the reduction of the transmission rate using estimated bandwidth when data losses are caused by bit error.

To begin with, we explain ABE (Available Bandwidth Estimation) [5]. In ABE, the bandwidth is obtained by two information, which are the ack arrival times and the increment of data delivered to the destination. Let assume that an ack is received at the sender at time t_k, notifying that d_k bytes have been received at the receiver. The sample bandwidth ($BW_{sample}(k)$) at time t_k is calculated by Eq. (2).

$$BW_{sample}(k) = \frac{d_k}{t_k - t_{k-1}} \tag{2}$$

The bandwidth (BW_k) is obtained by smoothing BW_{sample}. BW_k is calculated by Eq. (3). In Eq. (3), α means a smoothing factor. In [5], α is set equal to 0.9.

$$BW_k = \alpha \times BW_{k-1} + (1-\alpha) \times \frac{BW_{sample}(k) + BW_{sample}(k-1)}{2} \tag{3}$$

Next, when data losses occur (i.e. the reception of three duplicate acks), the sender sets *cwnd* and *ssthresh* based on the estimated bandwidth ($BW_k (= BW)$) and the minimum round trip time (RTT_{min}). The sender sets *cwnd* and *ssthresh* as Fig. 1.

3.2 LWC: Lift Window Control

LWC increases the congestion window quickly using both the window control of TCP-Reno and the values based on the results of ABE. Thus, the congestion window becomes larger quickly than the TCP-Reno (see Fig. 2).

To begin with, we define $cwnd_{reno}$ as the congestion window of TCP-Reno. TCP-Reno sets the congestion window as Fig. 3.

$cwnd_{reno}$ captures the basic Reno behavior (i.e. Fast Retransmit) when data losses occur.

Next, the congestion window *cwnd* is calculated by sum of $cwnd_{reno}$ and the window size based on the results of ABE $cwnd_{abe}$. The congestion window is set as follows:

$$cwnd = cwnd_{reno} + cwnd_{abe} \tag{4}$$

$$cwnd_{abe} = \frac{BW \times RTT_{min}}{data\ segment\ size} \tag{5}$$

In Eq. (4), the congestion window size *cwnd* may become larger than the network capacity (i.e. bandwidth product delay). However, the sending window size (means the amount of sending data) is defined by the minimum of the congestion window size and advertised window size *rwnd* from the receiver. *rwnd* is often set equal to the bandwidth delay product when the client wants to obtain the maximum performance. Thus, if *cwnd* becomes larger than network capacity (i.e. *rwnd*), the sending window size does not become larger than the network capacity.

Next, we explain the reason why $cwnd_{reno}$ is added to $cwnd_{abe}$. If *cwnd* is only decided by $cwnd_{abe}$, TCP-STAR will send few data when the estimated bandwidth ($cwnd_{abe}$) is very small (e.g. $cwnd_{abe}$ is equal to 1 or 2). That is, if the condition of low bandwidth continues, the congestion window size does not increase. On the other hand, when the sender of TCP-Reno receives the ack segment, TCP-Reno increases the congestion window size even if the available bandwidth is small. As a result, the amount of sending data by TCP-STAR becomes smaller than that of TCP-Reno

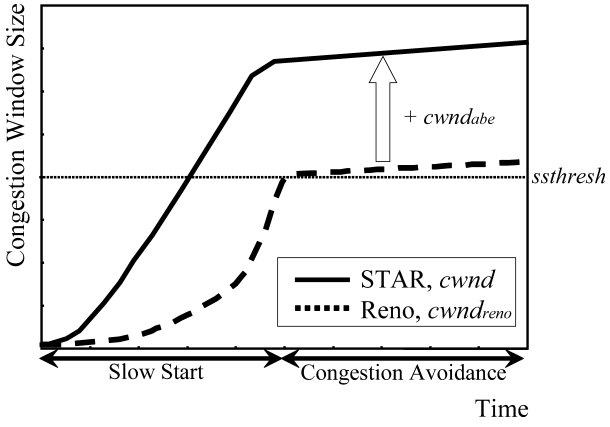

Fig. 2 Congestion window of TCP-STAR (LWC) and TCP-Reno.

if ($cwnd < ssthresh$)
  $ssthresh = \frac{BW \times RTT_{min}}{data\ segment\ size}$
  $cwnd \Longrightarrow$ keep recent value
else if ($cwnd \geq ssthresh$)
  $ssthresh = \frac{BW \times RTT_{min}}{data\ segment\ size}$
  $cwnd = ssthresh$

Fig. 1 CWS mechanism.

if ($cwnd_{reno} < ssthresh$)
  $cwnd_{reno} = cwnd_{reno} + 1$
else if ($cwnd_{reno} \geq ssthresh$)
  $cwnd_{reno} = cwnd_{reno} + \frac{1}{cwnd_{reno}}$

Fig. 3 TCP-Reno's congestion window behavior.

when the available bandwidth becomes small. Thus, we add $cwnd_{reno}$ to $cwnd_{abe}$ in order to keep the fairness between TCP-Reno and TCP-STAR.

3.3 AEN: Acknowledgment Error Notification

AEN can detect the unnecessary timeout which is caused by ack losses or delay and avoid mis-retransmission of data. Thus, TCP-STAR can avoid the reduction of throughput by mis-retransmission of data.

We explain the behavior of AEN in detail. Figure 4 shows typical behavior of AEN when the timeout occurs which is caused by data losses (Fig. 4(a)) and ack losses (or delay) (Fig. 4(b)). In Fig. 4, the timeout timer for a data segment which is sent at time t_0 expires at time t_1. Then, the sender sends an AEN-probe segment (i.e. retransmits a data segment which is sent at time t_0) in order to detect whether the timeout is caused by data losses or ack losses (or delay). The receiver of the AEN-probe segment sends an ack (AEN-ACK) segment which includes the latest sequence number of the received data (i.e. it is the normal action of TCP-Reno receiver). After the sender received the AEN-ACK segment at time t_2, the sender judges the reason of the timeout by comparing the sequence number of the AEN-probe segment (Pro_{seq}) with the one of the AEN-ACK segment (ACK_{seq}). The behaviors after the detection of data losses and ack losses (or delay) are summarized in (I) and (II), respectively.

(I) If $Pro_{seq} > ACK_{seq}$, AEN judges that the timeout is caused by data losses. Then, the sender retransmits the lost data segment (at time t_3 in Fig. 4(a)) and sets the congestion window as one data segment size, because we assume that the network is congested. Also, $ssthresh$ is set equal to $cwnd_{abe}$.

(II) Else if $Pro_{seq} \leq ACK_{seq}$, AEN judges that the timeout is caused by ack losses or delay. Thus, AEN can detect the unnecessary timeout. The sender sends new data with the congestion window size, which is the values before the timeout occurs at time t_1, after the judgment (at time t_3 in Fig. 4(b)). Thus, $cwnd$ and $ssthresh$ keep recent values.

If the timeout of the AEN-probe segment occurs, TCP-STAR executes Slow Start.

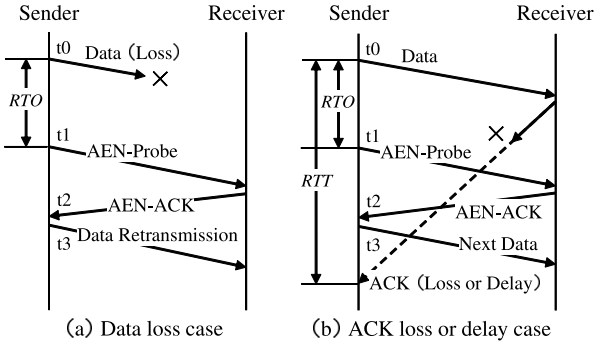

Fig. 4 Typical AEN behavior.

4. Simulations

This section shows that the proposed method improves the throughput by simulation as compared with other TCP variants. We also show the results of fairness evaluation. The proposed method is implemented on the NS2 (Network Simulator Version 2.27) [11].

4.1 Simulation Setting

Table 1 indicates the simulation parameters. In simulation, we use the window scale option [16] in order to evaluate the maximum performance in satellite Internet. Figure 5 represents a simulation topology. The simulation topology consists of a sender, a receiver, and a satellite. The supposed application in this simulation is FTP which has N connections.

4.2 Evaluation of Throughput

This section evaluates the throughput of TCP-STAR and other TCP variants (TCP-J, TCP-WestwoodBR, and TCP-NewReno). In order to verify the new congestion control method, we consider two cases in the simulation: data losses case and ack losses case. Furthermore, we set the number of connection N as one in order to evaluate the maximum performance (throughput) of TCP-STAR.

4.2.1 Data Losses Case

We assume that the buffer size of the sender and receiver are 432 segments (equals to the bandwidth delay product size), because the user can obtain the maximum throughput.

Figure 6 shows the throughput of TCP-STAR, TCP-J, TCP-WestwoodBR, and TCP-NewReno when data losses occur, where the BER (Bit Error Rate) increases from 0 (error-free) to 10^{-5}. Here, the typical BER is from 10^{-8} to 10^{-6} in satellite links [17]. However, in case of heavy rain such as the typhoon, BER may become 10^{-5}.

Table 1 Simulation parameters.

Bandwidth (Mbps)	10
Propagation delay (msec)	250
Buffer size (segments)	64, 432
Data segment size (byte)	1448
Ack segment size (byte)	52

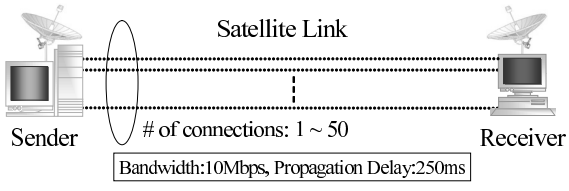

Fig. 5 Simulation topology.

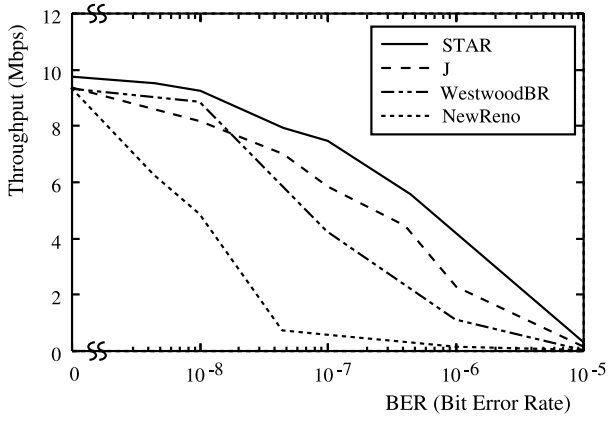

Fig. 6 Throughput: Data losses case.

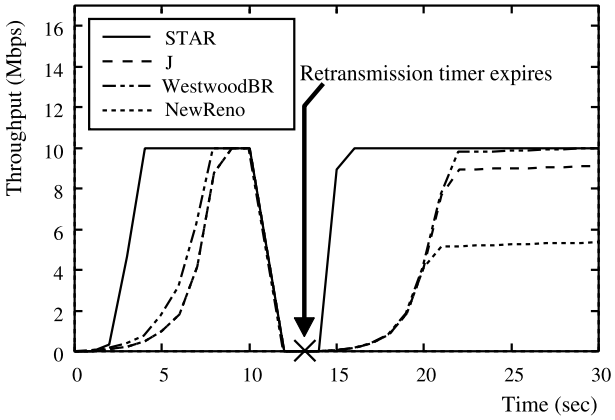

Fig. 8 Throughput: Ack losses case.

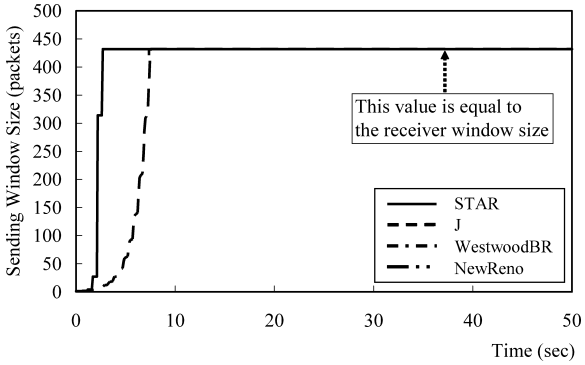

Fig. 7 Typical sending window size in case of BER = 0 (TCP-J, WestwoodBR, and NewReno are overlapping).

From Fig. 6, TCP-STAR improves the throughput comparing to other TCP variants when BER $< 10^{-5}$. It is because TCP-STAR can increase the congestion window size quickly by using LWC and CWS when data losses occur.

Here, note that there is a difference between TCP-STAR and other TCP variants in spite of no packet losses when BER = 0. The reason why the throughput of TCP-STAR is higher than other TCP variants is that TCP-STAR can increase the congestion window size quickly at the beginning of data transmission (i.e. slow start phase) by using LWC. Figure 7 indicates the typical sending window size of each TCP version in case of BER = 0 in Fig. 6. Now, the sending window size is obtained by the minimum of the congestion window size and the receiver (advertised) window size. Therefore, the sending window size is equal to the congestion window size until it reaches the receiver window size. And then, since packet losses do not occur in case of BER = 0, the sending window size is equal to the receiver window size. From Fig. 7, the sending window size of TCP-J, TCP-WestwoodBR, and TCP-NewReno increases slowly at times from 0 sec to 8 sec, because these TCP versions execute the slow start. On the other hand, the sending window size of TCP-STAR increases quickly because it is able to increase the congestion window size immediately by LWC. As a result, in case of no packet losses (i.e. BER = 0), TCP-STAR can obtain higher throughput comparing with other three TCP variants due to the difference of the sending window size at the beginning of data transmission.

In the case of BER = 10^{-5}, the throughput of TCP-STAR is almost equal to other TCP variants. Because BER is very large, the retransmission timeout often occurs by data losses. As a result, TCP-STAR retransmits the lost data and sets the congestion window size as one data segment size. Thus, the difference of the throughput between TCP-STAR and other TCP variants is small. However, TCP-STAR can obtain the best performance in such case.

4.2.2 Ack Losses Case

Figure 8 indicates the throughput of TCP-STAR, TCP-J, TCP-WestwoodBR, and TCP-NewReno when burst losses of ack segments occur at times from 10 sec to 12 sec, where the buffer size is 432 segments. Thus, the timeout timer expiration occurs by ack losses in the simulation. From Fig. 8, we found that TCP-STAR can recover the throughput immediately (from 13 sec to 16 sec) comparing with other TCP variants. It is because TCP-STAR restarts data transmission at time 13 sec with the large window size by using AEN. However, in other TCP variants, the mis-retransmission of data occurs and the Slow Start phase is executed. As a result, it takes long time to increase the throughput. Therefore, we found that AEN operates effectively. When the timeout occurs by ack delay, we confirmed that AEN improves the throughput.

From the results of Sect. 4.2.1 and this subsection, it is clear that TCP-STAR improves the adaptability of the network conditions that there are not only data losses but also ack losses/delay.

4.3 Evaluation of Fairness

This section evaluates the fairness of TCP-STAR by comparing with TCP-J and TCP-WestwoodBR. In the simulation, we set the parameters as follows; the number of con-

nection $N = \{10, 20, 30, 40, 50\}$, BER = $\{10^{-8}, 10^{-6}\}^{\dagger}$, and buffer size = 64 segments. In order to evaluate the fairness, we use Jain's fairness index [18]. The fairness index is obtained by Eq. (6).

$$Fairness\ \ Index = \frac{(\sum_{i=1}^{N} x_i)^2}{N \sum_{i=1}^{N} x_i^2} \tag{6}$$

where N is the number of connection and x_i denotes the throughput of the i-th connection. The fairness index is bounded between 0 and 1. When the fairness index gets close to 1, it indicates the throughput of all connection is fair.

To begin with, we show the results when the different TCP versions coexist (heterogeneous environment). Figures 9 and 10 indicate the fairness index when BER is set equal to 10^{-8} and 10^{-6}, respectively. In Figs. 9 and 10, $N/2$ TCP-STAR's (or TCP-WestwoodBR, TCP-J) connections coexist with $N/2$ TCP-NewReno's connections.

From Fig. 9, the fairness index of each TCP version is better when the number of connection is 10. Here, we found that there is no congestion in case of $N = 10$ from the simulation results. However, the fairness index decreases as the number of connection increases. Because each connection's traffic is competing and packet losses by the congestion occur. Therefore, TCP-NewReno reduces the congestion window to half of its previous values after packet losses. On the other hand, since the congestion window is set to the size based on the available bandwidth in TCP-STAR, TCP-WestwoodBR, and TCP-J, these congestion window tend to become larger than that of TCP-NewReno. As a result, the throughput of TCP-NewReno becomes smaller than that of coexisting TCP versions (TCP-STAR, TCP-WestwoodBR, and TCP-J) and the fairness index decreases.

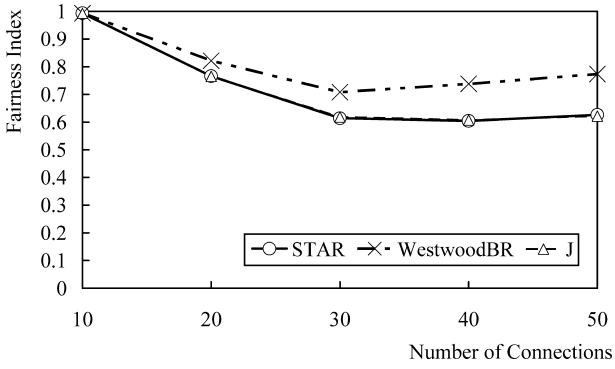

Fig. 9 Fairness index in heterogeneous environment: TCP-NewReno coexists, BER = 10^{-8} (The result of STAR overlaps the one of J).

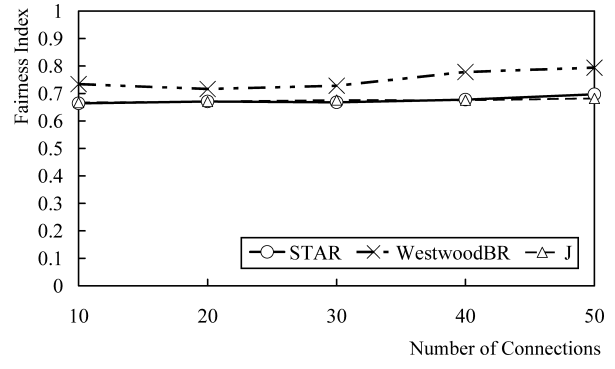

Fig. 10 Fairness index in heterogeneous environment: TCP-NewReno coexists, BER = 10^{-6} (The result of STAR overlaps the one of J).

From Fig. 10, when BER is 10^{-6}, the fairness index in case of $N = 10$ decreases comparing with the one of BER = 10^{-8}. However, as mentioned above, there is no congestion in case of $N = 10$. It is because packet losses often occur due to a high bit error rate and TCP-NewReno decreases the congestion window size unnecessary. As a result, the throughput of TCP-NewReno only decreases and the fairness index decreases in spite of no congestion.

In Figs. 9 and 10, we found that the fairness index of TCP-STAR and TCP-J is a little lower than that of TCP-WestwoodBR excepting $N = 10$ in case of BER = 10^{-8}. Here, TCP-WestwoodBR uses a mechanism LDA (Loss Discrimination Algorithm) [7] which can determine whether packet losses are due to congestion or bit error. If TCP-WestwoodBR judges the packet losses due to the congestion by using LDA, it halves the congestion window as well as TCP-NewReno. Therefore, the fairness of TCP-WestwoodBR is better than that of TCP-STAR and TCP-J when the congestion occurs. However, the LDA mechanism cannot judge perfectly whether the packet losses due to the congestion or bit error. Therefore, TCP-WestwoodBR pillages the available bandwidth from TCP-NewReno by misjudging the reason of packet losses and the fairness index of TCP-WestwoodBR decreases when the congestion occurs.

Next, we present the results when all connections use the same TCP version (homogeneous environment). Figure 11 shows the fairness index in case of BER = 10^{-6}. From Fig. 11, it is clear that all TCP versions keep the better fairness even if the number of connection increases. Furthermore, the fairness of TCP-STAR is almost equal to the ones of TCP-WestwoodBR and TCP-J.

In summary, when the heterogeneous TCP versions co-

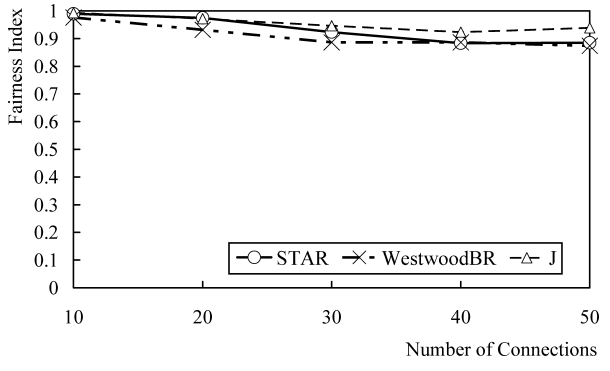

Fig. 11 Fairness index in homogeneous environment.

†In case of BER = 10^{-7}, we found that the result of fairness evaluation has intermediate values between the result of 10^{-8} and that of 10^{-6}.

exist, we observed that the fairness of TCP-STAR is a little lower than that of TCP-WestwoodBR. However, we have shown that TCP-STAR can obtain the best throughput comparing with TCP-WestwoodBR and TCP-J. It is because TCP-STAR sends data aggressively. Furthermore, TCP-STAR's fairness is equal to TCP-WestwoodBR and TCP-J in the homogeneous environment.

5. Conclusion

In this paper, we presented a new congestion control method (TCP-STAR) for satellite Internet. TCP-STAR only requires modification of the sender-side TCP module. TCP-STAR has three new mechanisms (CWS, LWC, and AEN). CWS can resist the reduction of the transmission rate when data losses are caused by bit error. LWC improves the increasing rate of the congestion window by using the available bandwidth. AEN avoids the reduction of the throughput by mis-retransmission of data which is caused by the acknowledgment losses or delay.

Through performance evaluation, TCP-STAR improves the throughput comparing with TCP-WestwoodBR and TCP-J. In the fairness evaluation, the fairness of TCP-STAR is equal to TCP-WestwoodBR and TCP-J in the homogeneous environment. However, in case of the heterogeneous environment, we found that the fairness of TCP-STAR is a little lower than that of TCP-WestwoodBR.

Our future research directions are summarized as follows:

- We consider adopting the mechanism which can detect the reason of packet losses such as LDA of TCP-WestwoodBR and improve the accuracy of the available bandwidth estimation method. Because, there is still room for improvement the fairness of TCP-STAR in the heavy congestion environment.
- We will plan to evaluate of other application protocols (e.g. HTTP) over TCP-STAR.

Acknowledgments

This research was partly supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (C) 16560345, and Hiroshima City University Grant for Special Academic Research 3105.

References

[1] L. Wood, G. Pavlou, and B. Evans, "Effects on TCP of routing strategies in satellite constellations," IEEE Commun. Mag., vol.39, no.3, pp.172–181, 2001.
[2] Japan Aerospace Exploration Agency (JAXA), "WINDS (Wideband InterNetworking engineering test and Demonstration Satellite)," http://www.jaxa.go.jp/missions/projects/sat/tsushin/winds/index_e.html, 2003.
[3] J. Postel, "Transmission control protocol—TCP," RFC 793, 1981.
[4] C. Partridge and T. Shepard, "TCP performance over satellite links," IEEE Netw., vol.11, no.5, pp.44–49, 1997.
[5] N. Sato, M. Kunishi, and F. Teraoka, "TCP-J: New transport protocol for wireless network environments," J. IPSJ, vol.43, no.12, pp.3848–3858, 2002.
[6] C. Casetti, M. Gerla, S. Mascolo, M.Y. Sanadidi, and R. Wang, "TCP Westwood: Bandwidth estimation for enhanced transport over wireless link," Proc. International Conference on Mobile Computing and Networking (Mobicom 2001), pp.287–297, 2001.
[7] G. Yang, R. Wang, Y. Sanadidi, and M. Gerla, "TCPW with bulk repeat in next generation wireless networks," Proc. IEEE International Conference on Communications (ICC 2003), pp.674–678, 2003.
[8] K. Ratnam and I. Matta, "WTCP: An efficient mechanism for improving TCP performance over wireless links," IEEE Symposium on Computer and Communications (ISCC 1998), pp.74–78, 1998.
[9] H. Balakrishnan, V.N. Padmanabhan, and S. Seshan, "A comparison of mechanisms for improving TCP performance over wireless links," IEEE/ACM Trans. Netw., vol.5, no.6, pp.756–769, 1997.
[10] I. Akyildiz, G. Morabito, and S. Palazzo, "TCP-Peach: A new congestion control scheme for satellite IP networks," IEEE/ACM Trans. Netw., vol.9, no.3, pp.307–321, 2001.
[11] UCB Multicast Network Research Group, "UCB/LBNL/VINT network simulator NS (ver.2)," http://www.isi.edu/nsnam/ns/, 2004.
[12] M. Allman, J. Griner, and A. Richard, "TCP behavior in networks with dynamic propagation delay," Proc. IEEE Global Communication Conference (Globecom2000), vol.2, pp.1103–1108, 2000.
[13] G. Hasegawa, M. Murata, and H. Miyahara, "Improvement of the congestion control mechanism of TCP to avoid mis-retransmission," IEICE Trans. Commun. (Japanese Edition), vol.J82-B, no.11, pp.2074–2084, Nov. 1999.
[14] S. Takeuchi, H. Obata, and K. Ishida, "A congestion control method of TCP for satellite Internet," IEICE Technical Report, IN2004-143, 2004.
[15] H. Obata, S. Takeuchi, and K. Ishida, "A new TCP congestion control method considering adaptability over satellite Internet," Proc. 4th International Workshop on Assurance in Distributed Systems and Networks (ADSN2005), pp.75–81, 2005.
[16] V. Jacobson, R. Braden, and D. Borman, "TCP extensions for high performance," RFC 1323, 1992.
[17] M. Allman, D. Glover, and L. Sanchez, "Enhancing TCP over satellite channels using standard mechanisms," RFC 2488, 1999.
[18] 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, pp.1–14, 1989.

Hiroyasu Obata received the B.E. and M.E. degrees in Information Engineering from Hiroshima City University, Japan, in 2000 and 2002. From 2002 to 2003, he was with KDDI Co., Ltd. He is currently a Research Associate in the Department of Computer Engineering, Faculty of Information Sciences, Hiroshima City University, since 2003. His interests include computer communications on wireless networks such as satellite links and high speed transport protocols. Mr. Obata is a member of IPSJ (Japan).

Kenji Ishida received the B.E., M.Sc., and Ph.D. degrees from Hiroshima University, Japan, in 1984, 1986 and 1989, respectively. He joined Hiroshima Prefectural University from 1989 to 1997. From 1997 to 2003, he was an Associate Professor at Hiroshima City University. Since 2003, he has been a Professor in the Department of Computer Engineering, Faculty of Information Sciences, Hiroshima City University. His interests include distributed computing systems and design of control procedures for computer networks and assurance systems. Dr. Ishida is a member of IEEE (USA), ACM (USA), IPSJ (Japan).

Satoru Takeuchi received the B.E. and M.E. degrees in Information Engineering from Hiroshima City University, Japan, in 2003 and 2005. He is currently in NTT DoCoMo Chugoku, Inc. His interests include computer communications on satellite networks.

Shouta Hanasaki received the B.E. degree in Information Engineering from Hiroshima City University, Japan, in 2005. He is currently in the Graduate School of Information Sciences, Hiroshima City University. His interests include transport protocols on wireless networks.