PAPER

TCP Using Adaptive FEC to Improve Throughput Performance in High-Latency Environments*

Yurino SATO[†a)], **Hiroyuki KOGA**[††b)], ***and*** **Takeshi IKENAGA**[†††c)], ***Members***

SUMMARY Packet losses significantly degrade TCP performance in high-latency environments. This is because TCP needs at least one round-trip time (RTT) to recover lost packets. The recovery time will grow longer, especially in high-latency environments. TCP keeps transmission rate low while lost packets are recovered, thereby degrading throughput. To prevent this performance degradation, the number of retransmissions must be kept as low as possible. Therefore, we propose a scheme to apply a technology called "forward error correction" (FEC) to the entire TCP operation in order to improve throughput. Since simply applying FEC might not work effectively, three function, namely, controlling redundancy level and transmission rate, suppressing the return of duplicate ACKs, interleaving redundant packets, were devised. The effectiveness of the proposed scheme was demonstrated by simulation evaluations in high-latency environments.

key words: *TCP, FEC, redundancy control, congestion control, interleave control*

1. Introduction

Communication over global and broadband networks has become widespread; consequently the exchange of large amounts of data, especially in high-latency environments, is increasing [3]. Although network environments for communication have changed significantly, TCP is still commonly used as a reliable data-transmission protocol [4]. It controls congestion by adjusting transmission rates according to network conditions. However, it cannot be utilized for such global and broadband networks, especially in high-latency environments. This is because TCP generally estimates the available bandwidth of networks on the basis of packet losses due to congestion and keeps the transmission rate low while lost packets are recovered. It needs at least "one round-trip time" (RTT) to recover lost packets. Since the recovery time grows longer, especially in high-latency environments, TCP throughput is degraded. To prevent this problem, the number of retransmissions must be kept as low as possible.

One efficient way to prevent packet losses is to apply a technology called "forward error correction" (FEC). FEC enables the sender to transmit packets with redundant information so that lost packets can be recovered (from the redundant information) at the receiver. The success rate of recovery depends on the amount of redundant information; however, redundant information places an additional load on the network. To use FEC effectively, the amount of redundant information must be appropriately determined according to network conditions. Therefore, it is typically used for UDP communication, which has a constant transmission rate, while it is difficult to adapt to TCP communication, in which transmission rate changes often, because it is harder to select an appropriate redundancy level. For that reason, although there have been few studies on generally applying FEC to TCP operations, there have been several studies on restrictively applying it to TCP operations.

In this study, aiming to improve throughput, a scheme to apply FEC to the entire TCP operation is proposed. However, a simple application of FEC to the entire TCP operation might not work effectively. If the redundancy is too low, lost packets might not be recovered effectively. Moreover, unnecessary retransmissions and timeouts are possibly caused, due to the reception of duplicate ACKs and lack of congestion avoidance, respectively, even if recovery is successful. In addition, FEC cannot recover lost packets if both of original and redundant packets are "burstily" lost in a network. Therefore, a scheme to control redundancy level according to transmission rate, suppress the return of duplicate ACKs, control transmission rates when recovery is successful, and interleave redundant packets from original packets is proposed hereafter. The effectiveness of this scheme is demonstrated through simulation evaluations in high-latency environments.

The rest of this paper is organized as follows: Section 2 reviews related work on FEC; Section 3 explains the proposed scheme; Section 4 describes the simulation environment; Section 5 presents the simulation results and discusses them with respect to the effectiveness of the proposed scheme; and Sect 6 concludes the paper and looks at future work.

2. Related Work

Related work that applies FEC to the entire TCP or UDP operation is described below.

Manuscript received March 28, 2018.
Manuscript revised July 21, 2018.
Manuscript publicized September 6, 2018.

[†]The author is with the Department of Control Engineering, National Institute of Technology, Sasebo College, Sasebo-shi, 857-1193 Japan.
[††]The author is with the Graduate School of Environmental Engineering, The University of Kitakyushu, Kitakyushu-shi, 808-0135 Japan.
[†††]The author is with the Graduate School of Engineering, Kyushu Institute of Technology, Kitakyushu-shi, 804-8550 Japan.

*The earlier versions of this work were presented at IEEE ICNP2013 [1] and published in IEICE ComEx [2].

a) E-mail: yuri@net.is.env.kitakyu-u.ac.jp
b) E-mail: h.koga@kitakyu-u.ac.jp
c) E-mail: ike@ecs.kyutech.ac.jp

DOI: 10.1587/transcom.2018EBP3091

FEC allows the receiver to recover lost packets with redundant information appended to packets at the sender [5]. As mentioned above, the success rate of recovery depends on the redundancy level. As ways to control redundancy, the proposed FEC algorithms are categorized into two groups: those keeping constant redundancy level independently of network conditions [6]–[9] and those adjusting redundancy level according to network conditions [10]–[17].

A method for applying FEC to UDP communication, which has a constant transmission rate, was proposed [6]. AL-FEC [7] utilizes FEC for UDP to improve throughput in high-loss-rate wireless environments. These schemes improve throughput performance of UDP communication, but the present study focuses on a scheme that applies FEC to TCP communication, in which transmission rate dynamically changes.

The problem of consecutive reductions in transmission rate caused by congestion control when multiple packets are lost consecutively is focused on in Refs. [10] and [11]. To resolve this problem, a method for applying FEC to TCP only when TCP detects packet losses was proposed [10]. This method attaches redundant packets during a recovery phase, namely, when transmission rate is low. It thus suppresses reduction of transmission rate as well as the amount of redundant information. FEC-ARQ [11] combines FEC with ARQ mechanisms to keep the quality of streaming services in a low-latency environment. This method is based on a packet streaming code well suited to sequential decoding and improves the total delay caused by retransmission. The problem that it takes a long time to recover lost packets, even when short-term burst packet losses occur in a high-latency broadband environment was focused on in Ref. [12]. This problem arises because TCP cannot discriminate between short-term congestion and long-term congestion when burst packet losses occur. To solve that problem, a method of controlling congestion by using "explicit congestion notification" (ECN) [18] to identify short-term congestion was proposed. This method uses adaptive FEC with redundancy proportional to the number of lost packets only when it detects packet losses by ECN. It attains high TCP throughput by quickly responding to burst packet losses in high-latency broadband networks. TCP-AFEC [13] uses FEC for streaming services, and it does not greatly change transmission rate. It controls FEC redundancy by detecting packet loss and maintains the appropriate transmission rate to preserve quality of service. It suppresses temporal reduction of transmission rate by changing redundancy to maintain the transmission rate needed for streaming delivery services. LT-TCP [14] uses ECN and FEC to mitigate the effects of random packet losses over lossy wireless networks. A method for improving fairness between TCP and media flows, specifically, multimedia delivery flows for mobile devices in the next-generation of wireless technology was proposed [15]. When TCP and media flows coexist in a network, media flows can easily cause packet losses and drastically reduce transmission rate due to TCP congestion control. This method prevents media-flow packet losses due to adaptive FEC to maintain fairness between TCP and media flows. It dynamically determines FEC redundancy by predicting loss events on the basis of transmission delay time. Moreover, TCP-IR [16] applies FEC to the TCP operation. It focuses on the problem that TCP needs at least 1 RTT when it recovers lost packets. It injects redundant packets within TCP streams, so it reduces latency of web transactions. This work enhances TCP throughput performance in low latency environments. It was extended to improve TCP throughput in high-latency environments [17]. TCP-IR encodes up to 16 packets. These works focus on restrictively applying FEC to the TCP operations.

3. Proposed Scheme

A scheme to apply FEC to the entire TCP operation, to improve throughput, is proposed. In the following, proposed scheme is overviewed, and each function of the scheme is explained in detail.

As shown in Fig. 1, the proposed scheme provides end-to-end communication based on FEC. The sender creates a redundant packet from original packets and transmits the redundant packet. The receiver can recover lost packets from the redundant packet. Note that recovery of lost packets is focused on; in other words, redundant information is not added to an original packet, and redundant packets encoded from original packets are interjected within TCP streams.

As for the proposed scheme, the sender encodes a redundant packet by using exclusive OR (XOR) with the payload field of all packets in a congestion window ($cwnd$) and transmits the redundant packet after the original packets. Namely, the proposed scheme constantly creates redundant packets according to network conditions. It thus needs only a few memory resources for creating redundant packets, and it can calculate an encoded bit-stream using simple bitwise operations. It has lower overhead than other coding schemes, like Reed-Solomon codes [19]. In the case of the proposed scheme, FEC group is defined as a block of one redundant packet with corresponding original packets, and group size is defined as the number of original packets.

The receiver uses XOR to recover a lost packet within the group when it receives a redundant packet. The proposed scheme cannot recover lost packets when two or more packet

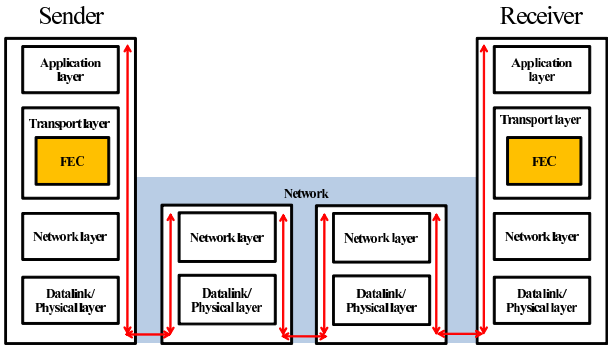


Fig. 1 Applying FEC to TCP operation.

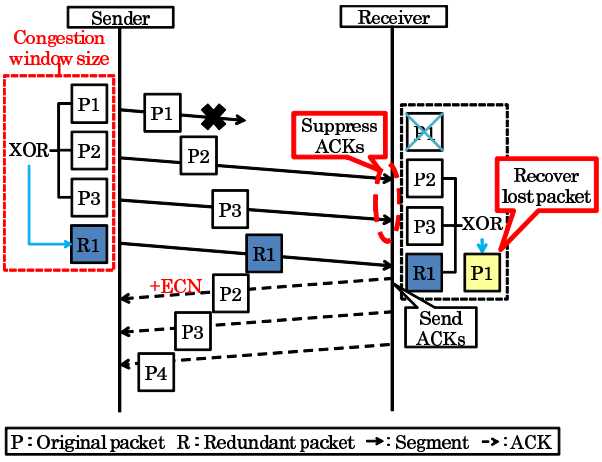


Fig. 2 Proposed scheme.

losses occur within a group. In that case, the sender retransmits the lost packets through the original TCP operation. When packet losses do not occur within a group, a received redundant packet is simply discarded.

The proposed scheme is overviewed in Fig. 2. In the following subsections, each function, namely, redundancy control, duplicate ACK suppression, congestion control, and interleave control, are explained in detail.

3.1 Redundancy Control

As mentioned above, the success rate of recovery depends on the amount of redundant packets, while redundant packets place an additional load on the network. Therefore, the amount of redundant packets must be appropriately determined according to network conditions. In the proposed scheme, redundancy decreases as transmission rate increases. This means that redundancy is high when packet losses have a significant impact — i.e., at low transmission rate — and remains low otherwise. Specifically, group size is controlled according to $cwnd$. Redundancy is defined as the ratio of a redundant packet to the corresponding original packets; i.e., $1/(cwnd - 1)$. For example, redundancy is 1 when $cwnd$ is 2. FEC group size is updated when the sender sends a redundant packet, i.e., at the end of an FEC group. If redundancy is too low, lost packets might not be recovered effectively. Therefore, in the proposed scheme, an upper limit to group size, "l", is introduced to prevent inefficient operation. Redundancy is then expressed as $1/\min(cwnd-1, l)$. The effect of l was evaluated through simulations.

3.2 Suppression of Duplicate ACKs

The proposed scheme can recover a lost packet from a redundant packet within a group. FEC mechanisms might cause unnecessary retransmissions due to the reception of duplicate ACKs even if recovery succeeds. Therefore, the proposed scheme suppresses returning duplicate ACKs until it is determined that lost packets cannot be recovered; namely, when another original packet or a redundant packet in a group is lost.

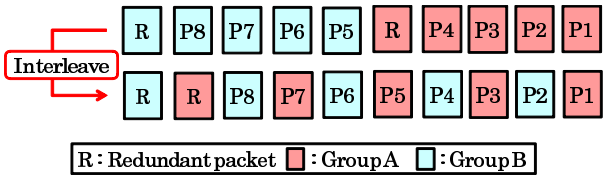


Fig. 3 Interleave control.

3.3 Congestion Control

TCP generally controls congestion by detecting packet losses. Since FEC mechanisms can recover lost packets, congestion does not be controlled through original TCP operations in spite of excessive transmission rate. To avoid congestion, the proposed scheme uses ECN to notify the sender that lost packets have been successfully recovered. Specifically, the receiver adds ECN information to a returning ACK when it recovers a lost packet. The sender decreases transmission rate when it receives the ACK with ECN. Transmission rate, i.e., $cwnd$, is calculated by using a reduction factor, "r" ($0 < r \leq 1$), as

$$cwnd = r * cwnd. \quad (1)$$

For example, the reduction factor of TCP NewReno [20] is 0.5, and that of CUBIC [21] is 0.8. The effect of the reduction factor on throughput performance was evaluated through simulations.

3.4 Interleave Control

On the Internet, packet losses commonly occur "burstily" rather than randomly. Transmitting a redundant packet following original packets might cause unsuccessful recovery due to burst packet losses. Therefore, as shown in Fig. 3, the proposed scheme interleaves each packet of a group with packets of other groups to prevent multiple packet losses belonging to the same group. The number of groups within a $cwnd$ is defined as "g," and each group consists of the same number of packets to be interleaved. Namely, with the proposed scheme, number of groups is constant, and group size is dynamically controlled. Group size is calculated as shown in Fig. 4. When $cwnd$ is less than $2*g$, the number of groups is set to 1; i.e., the packets cannot be interleaved. The large number of groups will have high tolerance to burst packet losses. However, timeouts will be caused by duplicate ACK suppression because interleaving elongates maximum ACK suppression time. The effect of the number of groups was evaluated through simulations as described in the following.

4. Simulation Model

The effectiveness of the proposed scheme in the case of high-latency environments was evaluated through simulation

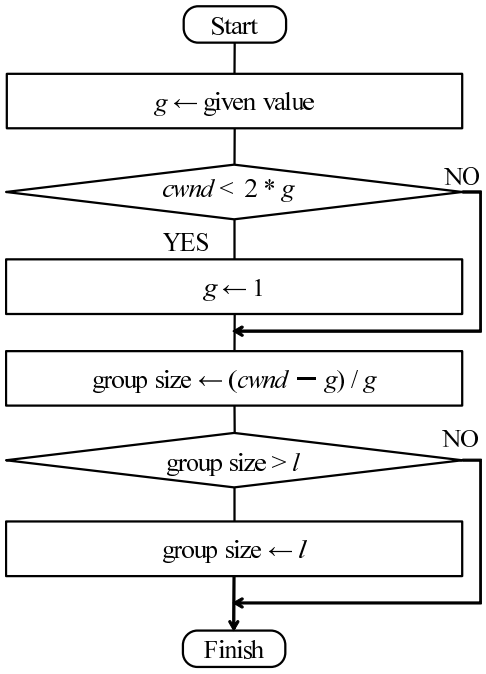


Fig. 4 Calculation of group size.

Table 1 Simulation parameters.

Simulation time	60 [s]
Buffer size on routers	300 [packet]
Buffer size on end nodes	∞
Number of flows	16
Segment size	1000 [Byte]
Number of trials	10
Upper limit of group size (l)	0, 10, 20, 30, 40
Reduction factor (r)	0.5, 0.7, 1.0
Number of groups (g)	1, 2, 3, 4, 5
TCP algorithm	TCP NewReno

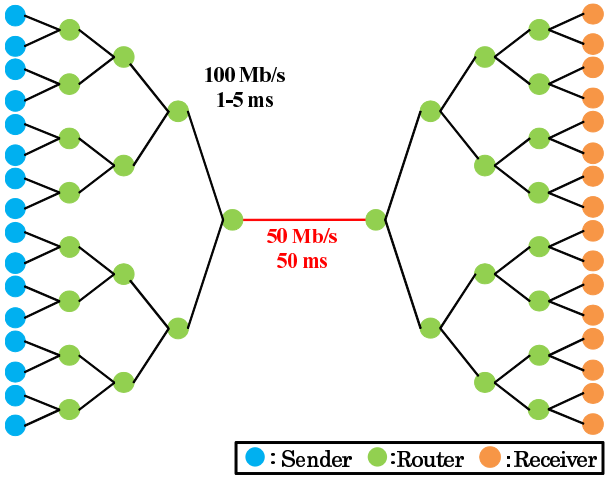


Fig. 5 Simulation model.

using Network Simulator ns-3 [22] after its implementation.

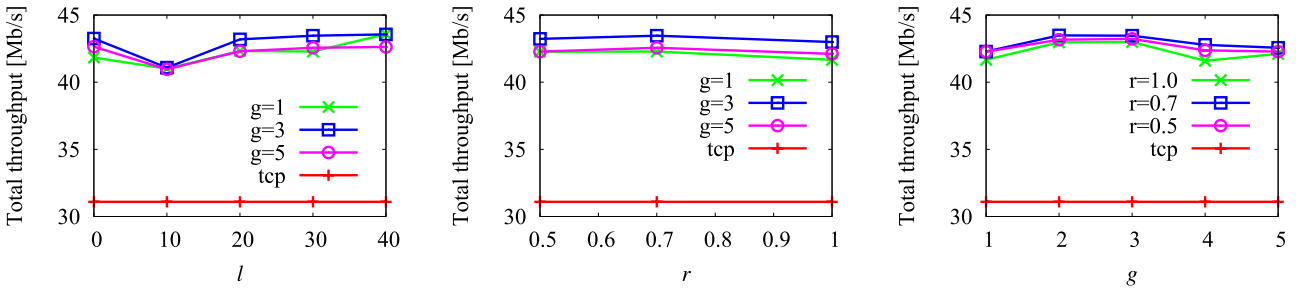
The parameters used in the simulation are summarized in Table 1. As shown in Fig. 5, a sender communicates with the corresponding receiver; that is, the sender transmits continuous data packets to the receiver. Since TCP throughput in high-latency environments is focused on in this study, it is assumed that the bottleneck link has a bandwidth of 50 Mb/s and a delay time of 50 ms. Other links have a bandwidth of 100 Mb/s and a delay time of 1 to 5 ms. In this simulation, the upper limit of group size (l) was varied from 0 to 40. l of "0" means that the proposed scheme does not limit group size. The reduction factor of transmission rates (r) was varied from 0.5 to 1.0. r of 1.0 means that the proposed scheme does not decrease transmission rate when it recovers a lost packet. Number of groups (g) was varied from 1 to 5. g of "1" means that the proposed scheme does not interleave redundant packets.

Performance of the proposed scheme, focusing on total throughput, compared with that of the conventional TCP, was evaluated. Moreover, the characteristics of the proposed scheme were analyzed in terms of number of TCP fast recoveries and timeouts, number of recovery packets, average redundancy rate, and effective recovery rate. Effective recovery rate is defined as the ratio of recovered packets to redundant packets; for example, a rate of 10% means that 90% of redundant packets are not utilized effectively. Furthermore, the characteristics of the proposed and conventional schemes in transient and steady states were analyzed. Transient and steady states are defined as a simulation period from 0 to 20 s and that from 20 to 60 s, respectively.

5. Simulation Results

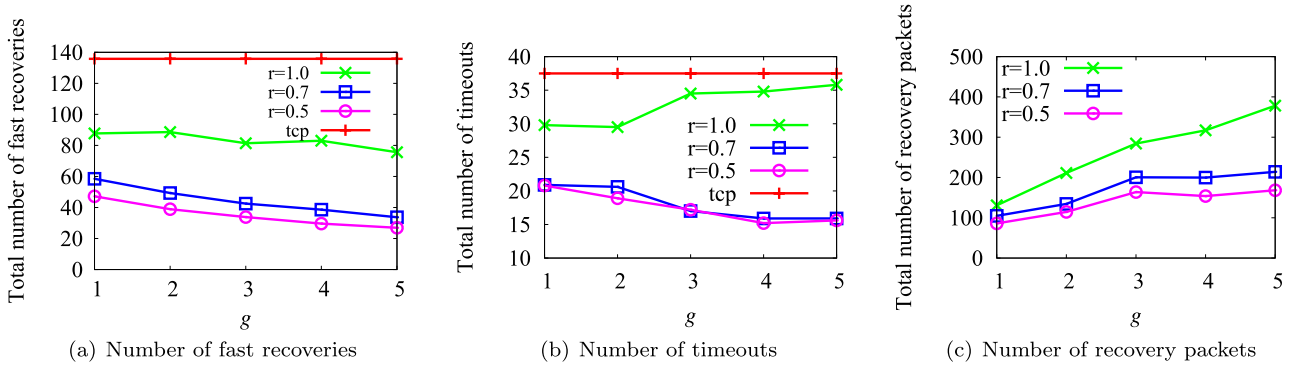
The simulation results are presented in the following, and the effectiveness of the proposed scheme compared with the conventional TCP is discussed. The effect of upper limit of group size (l), reduction factor of transmission rate (r), and number of groups (g) were investigated first. Next, the characteristics of proposed and conventional schemes were then analyzed.

Total throughputs when the proposed and conventional schemes were applied over the whole simulation time are shown in Fig. 6. According to Fig. 6(a), the proposed scheme achieves higher throughput than that of the conventional scheme, namely, "tcp." In the case of the proposed scheme, g is set to 1, 3, and 5, and r is set to 0.7. In particular, the proposed scheme with l of 30–40 attains the highest throughput in this simulation. Small l will cause transmissions of wasteful redundant packets due to excessive redundancy, while large l will cause unsuccessful recovery due to insufficient redundancy. Namely, an appropriate redundancy should be determined according to network conditions. That issue will be addressed in future work. In the following simulation, l is set to 30. Total throughputs of the proposed and conventional schemes when r was varied from 0.5 to 1.0 and g was varied from 1 to 5 are plotted in Figs. 6(b) and 6(c), respectively. These figures indicate that the proposed scheme achieves higher throughput than that achieved by the conventional scheme regardless of the parameters evaluated. In particular, the proposed scheme attains the highest throughput when

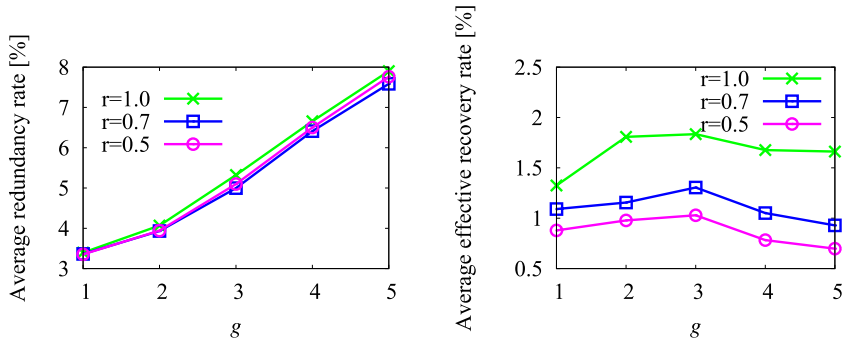


(a) Effect of upper limit of group size ($r = 0.7$) (b) Effect of reduction factor ($l = 30$) (c) Effect of number of groups ($l = 30$)

Fig. 6 Throughput performance (0–60 s).



(a) Number of fast recoveries (b) Number of timeouts (c) Number of recovery packets



(d) Redundancy rate (e) Effective recovery rate

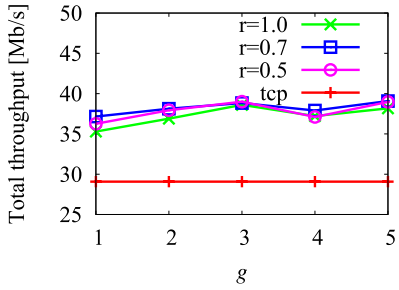
Fig. 7 Analysis of the characteristics (0–60 s).

r and g are 0.7 and 2–3, respectively. Consequently, the proposed scheme can improve TCP throughput performance by applying FEC to TCP operation to recover lost packets effectively. In the following subsections, the characteristics of proposed scheme are discussed in detail.

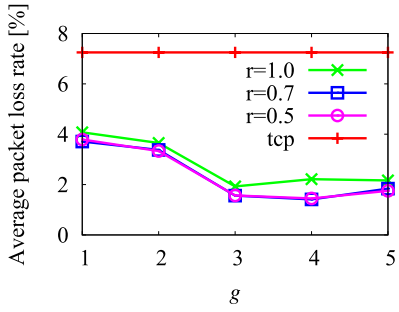
5.1 Analysis of Characteristics of Proposed Scheme

The reason that throughput was improved was investigated as described in the following. Number of TCP fast recoveries and timeouts, number of recovery packets, redundancy rate, and effective recovery rate for the proposed and conventional schemes, when l is set to 30, are shown in Fig. 7. According to Fig. 7(a), the proposed scheme significantly reduces the number of TCP fast recoveries, although the conventional scheme causes a large number of them. In the case of the proposed scheme, the number of fast recoveries decreases as number of groups increases, because a larger number of groups can help to recover lost packets effectively by interleaving. On the other hand, the proposed scheme with the large reduction factor ($r = 1.0$) causes a large number of fast recoveries. This is because the proposed scheme does not avoid congestion, although it can recover lost packets effectively when the reduction factor is large. Number of TCP timeouts, shown in Fig. 7(b), shows a similar trend to that of number of fast recoveries. However, the proposed scheme with r of 1.0 and g of 3–5 increases the number of timeouts. This is because a large number of groups increases maximum duplicate ACK suppression time, so timeouts will easily occur in a congested situation ($r = 1.0$).

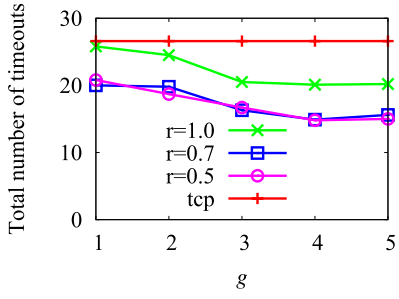
The effect of FEC on the recovery of lost packets was investigated as follows. As shown in Fig. 7(c), number of recovery packets increases as number of groups due to the effect of interleaving. Namely, a large number of groups can



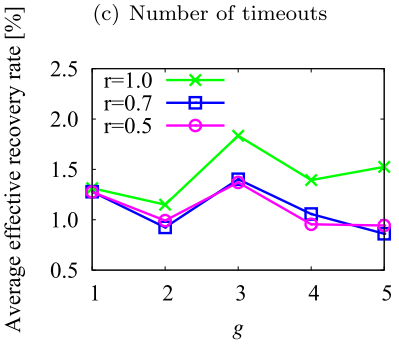
(a) Total throughput



(b) Data-packet loss rate

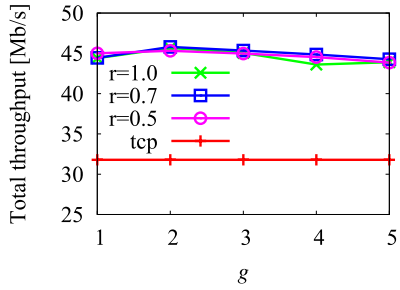


(c) Number of timeouts

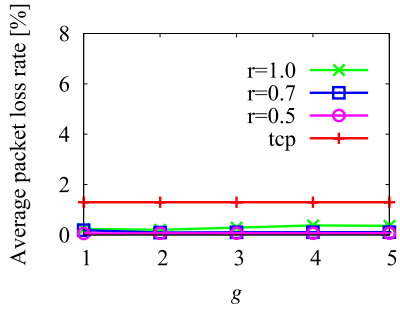


(d) Effective recovery rate

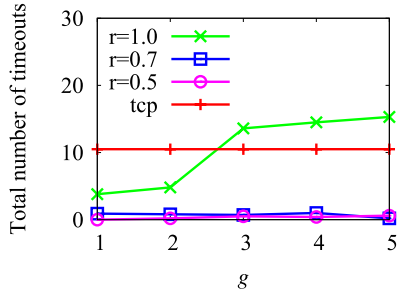
Fig. 8 Transient state (0–20 s).



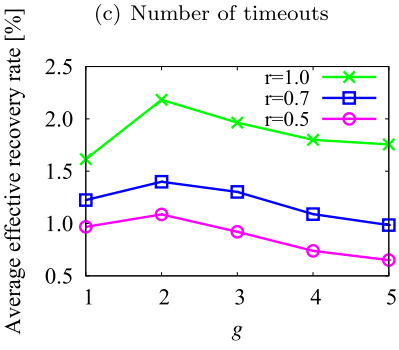
(a) Total throughput



(b) Data-packet loss rate



(c) Number of timeouts



(d) Effective recovery rate

Fig. 9 Steady state (20–60 s).

help to recover lost packets effectively, while it also causes high redundancy, as shown in Fig. 7(d). If redundancy is too high, redundant packets will be wastefully transmitted; consequently, most redundant packets will not be used to recover lost packets. This outcome can be confirmed in terms of effective recovery rate, shown in Fig. 7(e). It is clear from the figure that the proposed scheme achieves the most-efficient recovery of lost packets when number of groups is 3.

The above simulation results demonstrate the proposed scheme achieves the highest throughput performance by recovering lost packets effectively when r and g are set to 0.7 and 3, respectively.

5.2 Analysis Characteristics of Proposed and Conventional Schemes in Each State

The characteristics of the proposed and conventional schemes in transient and steady states were investigate as follows. Total throughput, data-packet loss rate, number of

TCP timeouts, and effective recovery rate for the proposed and conventional schemes in transient and steady states, are shown in Figs. 8 and 9, respectively. According to Figs. 8(a) and 9(a), the proposed scheme achieves higher throughput than the conventional scheme by recovering lost packets effectively in both states. This result can be confirmed by data-packet loss rate shown in Figs. 8(b) and 9(b). The proposed scheme can drastically reduce the packet loss rate in both states. The packet loss rate in transient state is relatively larger than that in steady state. This is because, in transient state, transmission rate significantly changes (due to TCP's slow-start mode) and packet losses can easily occur. In particular, the proposed scheme with large g (= 3–5) severely reduces packet loss rate in transient state for the same reason described above. As shown in Figs. 8(c) and 9(c), the proposed scheme also reduces number of TCP timeouts in both states. In the case of the proposed scheme, number of timeouts decreases as number of groups increases, in a similar manner to the results for packet loss rate in transient state, because a larger number of groups can help to recover lost packets effectively by interleaving. However, in steady state, it increases when the reduction factor is 1.0 and number of groups is larger than 2. In steady state, since transmission rate, i.e., $cwnd$, is relatively large, maximum duplicate ACK suppression time becomes likely long, and it will easily cause timeouts, particularly in congested situations. In transient and steady states, the proposed scheme with g of 3 and 2, respectively, attains the highest throughput. As shown in Figs. 8(d) and 9(d), this is because the lost packets can be recovered most effectively in each state when the number of groups is set to these values. In other words, bursty packet losses can more easily occur, and larger number of groups is needed to effectively interleave in transient state than in steady state. The proposed scheme can therefore be further improved by dynamically adjusting the number of groups according to network conditions such as packet loss rate, which will be considered in future work, although it achieves higher throughput by applying FEC with static parameter setting of appropriate values than the conventional scheme.

6. Conclusion

Packet losses significantly degrade TCP performance in high-latency environments. To improve TCP throughput in such networks, a scheme to apply FEC to the entire TCP operation was proposed. This scheme consists of functions for controlling redundancy level and transmission rate, suppressing the return of duplicate ACKs, and interleaving redundant packets. Evaluations of various characteristics by simulation show that the proposed scheme enables higher TCP throughput performance than the conventional scheme by recovering lost packets effectively. In future work, we aim to devise a scheme to determine the appropriate values of each parameter according to network conditions and to more effectively recover lost packets.

Acknowledgments

This work was supported in part by JSPS KAKENHI Grant-in-Aid for Scientific Research (B) Number 16H02806.

References

[1] Y. Sato, H. Koga, and T. Ikenaga, "Redundancy control and duplicate ACK suppression methods for TCP with FEC," Proc. IEEE ICNP2013, 2 pages, Oct. 2013. DOI: 10.1109/ICNP.2013.6733623
[2] Y. Sato, H. Koga, and T. Ikenaga, "Improving TCP throughput using forward error correction," IEICE Communications Express, vol.6, no.1, pp.28–33, Jan. 2017. DOI: 10.1587/COMEX.2016XBL0158
[3] Cisco Systems, Inc., "Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021," Feb. 2017. Information available at https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.pdf [Retrieved: Feb. 2017].
[4] R. Meza, "A survey on congestion control," European Journal of Computer Science and Engineering, vol.10, 5 pages, 2013.
[5] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft, "Forward error correction (FEC) building block," IETF RFC3542, Dec. 2002.
[6] Y.W. Kwon, H. Chang, and J. Kim, "Adaptive FEC control for reliable high-speed UDP-based media transport," Proc. IEEE PCM2004, pp.364–372, Dec. 2004. DOI: 10.1007/978-3-540-30542-2_45
[7] C. Bouras, N. Kanakis, V. Kokkinos, and A. Papazois, "Application layer forward error correction for multicast streaming over LTE networks," Int. J. Commun. Syst., vol.26, no.11, pp.1459–1474, Nov. 2013. DOI: 10.1002/dac.2321
[8] A. Vishwanath, V. Sivaraman, and M. Thottan, "Enabling a bufferless core optical network using edge-to-edge packet-level FEC," IEEE Trans. Commun., vol.61, no.2, pp.690–699, March 2013. DOI: 10.1109/TCOMM.2012.022513.110788
[9] Y. Sohn, J. Hwang, and S. Kang, "Adaptive packet-level FEC algorithm for improving the video quality over IEEE 802.11 networks," SERSC IJSEIA, vol.6, no.3, pp.27–34, July 2012.
[10] M. Sakakibara, Y. Tanigawa, and H. Tode, "Highly reliable TCP transfer method with error correction technology," Proc. MITA2009, pp.321–322, Aug. 2009.
[11] Y. Huang, S. Mehrotra, and J. Li, "A hybrid FEC-ARQ protocol for low-delay lossless sequential data streaming," Proc. IEEE ICME2009, pp.718–725, June/July 2009. DOI: 10.1109/ICME.2009.5202596
[12] V. Sharma, K. Ramakrishnan, K. Kar, and S. Kalyanaraman, "Complementing TCP congestion control with forward error correction," Proc. IFIP Networking2009, pp.378–391, May 2009. DOI: 10.1007/978-3-642-01399-7_30
[13] T. Tsugawa, N. Fujita, T. Hama, H. Shimonishi, and T. Murase, "TCP-AFEC: An adaptive FEC code control for end-to-end bandwidth guarantee," Proc. IEEE PV2007, pp.294–301, Nov. 2007. DOI: 10.1109/PACKET.2007.4397053
[14] B. Ganguly, B. Holzbauer, and K. Kar, "Loss-tolerant TCP (LT-TCP): Implementation and experimental evaluation," Proc. IEEE MILCOM2012, pp.1–6, Oct./Nov. 2012. DOI: 10.1109/MILCOM.2012.6415694
[15] H. Seferoglu, A. Markopoulou, U.C. Kozat, M.R. Civanlar, and J. Kempf, "Dynamic FEC algorithms for TFRC flows," IEEE Trans. Multimedia, vol.12, no.8, pp.869–885, June 2010. DOI: 10.1109/TMM.2010.2053840
[16] T. Flach, N. Dukkipati, A. Terzis, B. Raghavan, N. Cardwell, Y. Cheng, A. Jain, S. Hao, E. Katz-Bassett, and R. Govindan, "Reducing web latency: The virtue of gentle aggression," Proc. ACM SIGCOMM2013, pp.159–170, Aug. 2013. DOI: 10.1145/2486001.2486014

[17] S. Ferlin and O. Alay, "TCP with dynamic FEC for high delay and lossy networks," Proc. ACM CoNEXT Student Workshop, 2 pages, Dec. 2016.
[18] K. Ramakrishnan, S. Floyd, and D. Black, "The addition of explicit congestion control notification (ECN) to IP," IETF RFC3168, Sept. 2001.
[19] I. Reed and G. Solomon, "Polynomial codes over certain finite fields," J. Soc. Ind. Appl. Math., vol.8, no.2, pp.300–304, June 1960.
[20] L.A. Griece and S. Mascolo, "Performance evaluation and comparison of Westwood+, New Reno, and Vegas TCP congestion control," ACM Comput. Commun. Rev., vol.34, no.2, pp.25–38, April 2004. DOI: 10.1145/997.150.997.155
[21] S. Ha, I. Rhee, and L. Xu, "CUBIC: A new TCP-Friendly high-speed TCP variant," ACM Operating Systems Review, vol.42, no.5, pp.64–74, 2008. DOI: 10.1145/1400097.1400105
[22] The ns-3 network simulator, http://www.nsnam.org/

Yurino Sato received the B.E. and M.E. degrees in Information and Media Engineering from the University of Kitakyushu, Japan in 2012 and 2014, respectively. From 2014 to 2018, she was a postdoctoral student in The University of Kitakyushu. Presently, she has been an assistant professor since April 2018 in the Department of Control Engineering, National Institute of Technology, Sasebo College, Japan. Her research interests include network architecture, transport protocol, and forward error correction.

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, The 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 ACM and IEEE.

Takeshi Ikenaga received the B.E., M.E., and D.E. degrees in computer science from Kyushu Institute of Technology, Japan in 1992, 1994 and 2003, respectively. From 1994 to 1996, he worked at NEC Corporation. From 1996 to 1999, he was an Assistant Professor in the Information Science Center, Nagasaki University. From 1999 to 2004, he was an Assistant Professor in the Department of Computer Science and Electronics, Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology. He was an assistant professor from 2004 to 2011 and then has been a professor since September 2011 in the Department of Electrical, Electronic and Computer Engineering, Faculty of Engineering, Kyushu Institute of Technology. His research interests include performance evaluation of computer networks and QoS routing. He is a member of IEEE.