Compressing Packets Adaptively Inside Networks

PAPER Special Section on New Generation Network towards Innovative Future Society

Compressing Packets Adaptively Inside Networks*

Masayoshi SHIMAMURA[†a)], **Hiroyuki KOGA**[††b)], **Takeshi IKENAGA**[†c)], ***and*** **Masato TSURU**[†d)], ***Members***

SUMMARY Introducing adaptive online data compression at network-internal nodes is considered for alleviating traffic congestion on the network. In this paper, we assume that advanced relay nodes, which possess both a relay function (network resource) and a processing function (computational and storage resources), are placed inside the network, and we propose an adaptive online lossless packet compression scheme utilized at these nodes. This scheme selectively compresses a packet according to its waiting time in the queue during congestion. Through preliminary investigation using actual traffic datasets, we investigate the compression ratio and processing time of packet-by-packet compression in actual network environments. Then, by means of computer simulations, we show that the proposed scheme reduces the packet delay time and discard rate and investigate factors necessary in achieving efficient packet relay.

key words: *advanced relay node, adaptive online lossless packet compression scheme, network resource, computational resource*

1. Introduction

Due to continual growth in the number of users and applications, Internet traffic is increasing explosively [2]. Thus, traffic congestion is not unusual today and is expected to be more serious in the near future [3]. One well-known approach to reduce traffic is data compression [4], [5]. Although the effectiveness of the compression depends strongly on the type of data content, several studies have proven that lossless data compression of network traffic is effective if adopted in adaptive and appropriate ways. Recently, several online adaptive compression schemes have been proposed, which were all assumed to be performed at data senders (i.e., end hosts). For example, the adaptive compression environment (ACE) improves the throughput of end-to-end data transmission considerably [6]; another example reduces the data size of actual traffic by approximately 70% [7].

On the other hand, in response to increased demand for a more dependable information infrastructure for social and economic activities, the Internet is thought to be approaching a turning point. Thus, the need to rethink and redesign the Internet has been arising. In particular, introducing network-supported architectures (e.g., Refs. [8], [9]) inside a network is attracting more attention recently than the conventional end-to-end argument that has led Internet design philosophy so far. For example, in Ref. [8], the provider-managed P4P mechanism (i.e., network side) that coordinates peer-to-peer applications (i.e., user side) achieves improved bandwidth efficiency.

To achieve dependable future networks, e.g., refer to Refs. [10]–[12], we consider that conventional simple packet relay processing at intermediate nodes must be reconsidered, and advanced relay processing that adapts to the network's status is needed. Future networks will contain various types of heterogeneous networks including virtual overlay networks using network virtualization technologies [13], wireless mesh backbone networks using a variety of high-speed wireless technologies [14], [15], and wireless sensor networks [16]. Therefore, relay processing needs to adapt to such networks. With this motivation, we consider that combining the two research directions enables efficient alleviation of network congestion. In general, traffic control inside networks can be more timely, fine-grained, and globally optimal than traffic control on an end-to-end basis, although network-internal control may be costly to implement. In our case, because compression reduces the data to be transmitted at the cost of compression processing time before transmission, data compression in noncongested networks may result only in degradation of data throughput. Thus, a compression strategy adaptive to a network's internal status is essential. If adaptive compression is performed inside networks rather than at end nodes, congestion can be alleviated effectively and overall network performance can be improved by compression of overall network traffic based on instantaneous information on the network's status.

Therefore, in this paper, we attempt to combine these two directions, focusing on adaptive lossless packet compression inside a network and showing its strong potential. The objective is to alleviate network traffic congestion by network providers rather than to improve the performance of specific application flows by users. The main assumption here is that intermediate nodes (e.g., routers) inside the network can perform both conventional packet relay (forwarding) and additional advanced functions using computational and/or storage resources at the nodes. We call such an intermediate node an advanced relay node. In our network-side compression approach, which is enabled by advanced

Manuscript received July 31, 2009.
Manuscript revised November 5, 2009.

[†]The authors are with Network Design Research Center, Kyushu Institute of Technology, Kitakyushu-shi, 804-8550 Japan.
[††]The author is with the Department of Information and Media Engineering, University of Kitakyushu, Kitakyushu-shi, 808-0135 Japan.

*An earlier version of this work has been presented in SAINT2009, July 2009 [1].

a) E-mail: shimamura@ndrc.kyutech.ac.jp
b) E-mail: h.koga@env.kitakyu-u.ac.jp
c) E-mail: ike@ecs.kyutech.ac.jp
d) E-mail: tsuru@ndrc.kyutech.ac.jp

DOI: 10.1587/transcom.E93.B.501

relay nodes, the assumption differs from that of the ACE [6], which is categorized as an end-to-end compression approach. However, both the ACE and our proposal exploit adaptive online data compression instead of compressing all the data offline. The ACE adjusts the compression processing frequency based on estimations of end-to-end available bandwidth. Since the ACE is assumed to be applied end to end for improving the throughput of one or more data flows from the data source (one node), it can control the sending data rate based mainly on the estimated end-to-end available bandwidth, and compression is performed per data block. On the other hand, network-side compression schemes cannot control incoming traffic rates and should avoid the risk of network congestion caused by low compression throughput itself.

As for handling incoming traffic in a network-side approach, one important point must be considered. Advanced relay processing involves additional costs compared with conventional simple relay processing, including I/O processing for using computational and/or storage resources, cross-layer processing for obtaining network information, and data compression. Note that, even in case that the computational processing speed is high, compressing all incoming packets may become inefficient, depending on network and computational resources. If the compression nodes inevitably compress all incoming traffic and then these nodes relay the compressed traffic, online compression may become a bottleneck when the compression nodes have a heavy computational workload or poor computational resources. In particular, compression can cause congestion with large packet delays and a high packet discard rate, when large amounts of traffic pass through the network (i.e., the compression nodes must be able to handle such traffic). Therefore, we assume the following three conditions, under which we design an effective compression approach.

1. The proposed scheme is applied to all incoming packets (except for those already compressed at previous nodes), not to specific packet flows,
2. but it adaptively and selectively compresses some of these packets, not all,
3. and the compression is performed packet by packet, not on an aggregated set of packets (a block).

Using instantaneous queue-length information, our proposed scheme adaptively compresses an incoming packet according to its waiting time in the queue. In other words, our proposed scheme exploits the potential positive effect of compression while avoiding the negative effect. Therefore, the proposed scheme can be effective in various types of network environments mainly depending on the packet compression speed and the packet transmission speed. We will investigate and clarify the effective area of the proposed scheme in Sect. 5.3. In particular, if a network frequently suffers from congestion due to insufficient bandwidth (e.g., wireless mesh backbone networks), introducing this scheme is expected to drastically improve its performance. On the other hand, even in high-speed networks, the proposed scheme can be effective for performance improvement in congested period by adaptively adjusting the compression frequency based on computational workload and congestion.

The effectiveness of the proposed scheme depends on compression performance, such as the data reduction achieved by packet compression and the ratio of packet compression processing time to packet transmission processing time in individual network environments. Furthermore, the proposed scheme may be unable to reduce packet size remarkably because its packet compression strategy handles small amounts of data compared with block compression. In this paper, we preliminarily examine the effectiveness and potential of packet compression using an actual traffic dataset. Then, we show the potential of adaptive packet compression inside networks, assuming that advanced relay nodes perform per-packet compression, and we evaluate our proposed scheme over a wide range of parameters for compression performance. Note that since the decompression processing speed is generally much faster than the compression processing speed [17], [18], we assume the decompression processing time can be ignored, as described in Sect. 5.1. This assumption allows us to focus on the performance effects of the proposed scheme.

The remainder of this paper is organized as follows. Section 2 reviews related studies of online lossless compression schemes. Section 3 describes the design of an adaptive online lossless packet compression scheme. Section 4 examines the possibility of packet compression preliminarily using actual traffic datasets and open benchmark information. Section 5 demonstrates the characteristics and effectiveness of our proposed scheme and clarifies factors in achieving efficient packet relay using computer simulation combined with a preliminary investigation of actual environment parameters. Section 6 concludes by briefly summarizing the main points and proposing future work based on the findings of this study.

2. Online Lossless Compression Schemes

In this section, we review two existing online lossless compression schemes and the problems that arise if they are simply applied to data compression processing inside networks.

The first scheme, the ACE [6], achieves effective compression processing by using a forecasting toolkit, the Network Weather System (NWS). Using end-to-end path information obtained by the NWS, the ACE adopts compression algorithms and adjusts the compression processing frequency. However, there are some concerns about using the ACE directly for network-side compression. First, it is assumed that the ACE can control the sending (i.e., incoming) traffic rate or block sending traffic because it works on an end-to-end basis to improve throughput of data flows from the data source. In contrast, it may degrade the delay performance and is infeasible at advanced relay nodes, which should forward all traffic inside the network. Second, path information management by the NWS is inapplicable to

network-side compression. The ACE uses path information between two endpoints in end-to-end compression, whereas the advanced relay nodes must obtain and manage a massive amount of path information in network-side compression. Finally, the ACE compresses a block, not a packet, so it must aggregate packets to generate a block[†]. This also is impractical at advanced relay nodes inside the network.

The second scheme, IPzip [7], effectively compresses multiple packets by exploiting intra-packet and inter-packet correlation properties. In Ref. [7], the authors define the compression ratio as $\frac{\text{Compressed data size}}{\text{Uncompressed data size}}$, and we adopt this definition in this paper rather than an alternative definition of $\frac{\text{Uncompressed data size}}{\text{Compressed data size}}$ given in Ref. [19]. The compression ratio range is greater than 0, and the value 1 means that the packet cannot be compressed. The compression ratio of actual payloads in IPzip is approximately 0.3 (i.e., 70% data size reduction), whereas that in the conventional algorithms Gzip and Bzip is approximately 0.5. IPzip aggregates similar packets into a block based on flow information and then compresses the block. In general, a massive amount of flow information can be managed in the end nodes, whereas it cannot be managed in the relay nodes, as mentioned in Ref. [20]. Therefore, flow information management should be avoided in advanced relay nodes. In addition, IPzip compresses all blocks. This property causes congestion if the compression processing speed is slower than the relay processing speed (i.e., link bandwidth) in the advanced relay nodes. In this case, packets should be compressed adaptively and selectively depending on the relay and compression processing speeds.

We summarize the previously mentioned issues for online lossless compression in networks in the following three points: (1) the ability to handle a massive amount of various independent incoming traffic, (2) compression processing frequency, and (3) flow information management. For these reasons, none of the conventional end-to-end compression schemes can be directly applied to advanced relay nodes.

3. Adaptive Packet Compression Scheme

As mentioned in Sects. 1 and 2, we assume the following three conditions in designing online lossless compression in advanced relay nodes:

1. Advanced relay nodes need to handle all incoming packets from numerous source hosts and sites (i.e., users).
2. These nodes should compress some packets adaptively, rather than all packets inevitably, to prevent compression processing from becoming a bottleneck.
3. Regarding the compression unit, these nodes should compress packets rather than blocks to avoid blocking packets and flow information management.

Based on these conditions, we construct a model of an adaptive online lossless compression scheme in advanced relay nodes. In Fig. 1, an advanced relay node has two logical queues, the CPU queue and the relay queue. When the node receives a packet, it queues the packet into the CPU queue. Then, when it dequeues the packet from the CPU queue, it determines whether packet compression is effective at that time based on

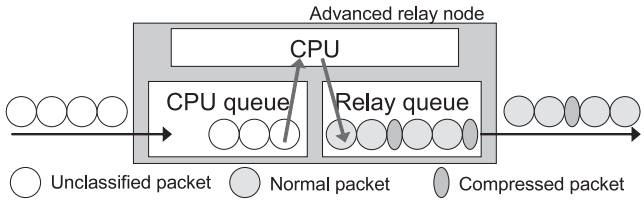


Fig. 1 Model of adaptive compression in advanced relay node.

$$W + \frac{S}{B} \geq \max(W, C) + R \cdot \frac{S}{B}, \tag{1}$$

where W, C, S, B, and R represent the waiting time in the relay queue, compression processing time, packet size, output link bandwidth, and compression ratio, respectively. Eq. (1) is satisfied only if R ranges from 0 to 1 because the compression processing increases the packet size if R is larger than 1. Furthermore, under the assumption that $0 < R \leq 1$, Eq. (1) can be equivalently transformed to

$$W \geq C - (1 - R) \cdot \frac{S}{B}. \tag{2}$$

If Eq. (2) is satisfied, the advanced relay node compresses the packet using the CPU when it moves the packet to the relay queue from the CPU queue. Since it can compress the packet in parallel with relay processing of previously queued packets, the compression processing time becomes approximately zero in practical terms. Otherwise, it moves the packet from the CPU queue to the relay queue without any computational processing at the CPU. Even if it compresses other packets at the CPU, the packet in the CPU queue experiences no additional delay time because the relay queue does not become empty during compression. In other words, when the packet exits the CPU queue, it has an opportunity to be compressed based on Eq. (2). Note that all packets enter and exit both the CPU and relay queues in sequential order to prevent out-of-order packet delivery.

From Eq. (2), in a congestion period, the proposed scheme likely has a chance to compress an arrival packet while the packet awaits departure. Consequently, the proposed scheme simultaneously uses network and computational resources effectively. When a packet passes through multiple congested advanced relay nodes, it has more chances to be compressed at any node. On the other hand, compressing a packet at a node helps mitigate congestion at all succeeding downstream nodes. In other words, the proposed scheme exploits the potential of the positive effect of compression on all advanced relay nodes. We call such a compression strategy an adaptive packet compression strategy. Regarding decompression of the compressed packets, advanced relay nodes located on the egress side decompress these packets, or certain nodes can be chosen for

[†]For example, in Ref. [6], 32 KB is assumed to be one block.

decompression processing based on routing information exchanged among them.

Compared with the related scheme [6] described in Sect. 2, our proposed scheme uses instantaneous local queue information in advanced relay nodes rather than time-averaged end-to-end global bandwidth information obtained from network management services. Note that the proposed scheme is independent of transport layer protocols such as TCP and UDP because it can be applied to all incoming packets independently, not to specific packet flows. Therefore, the proposed scheme does not need to manage flow information, which often hinders scalability.

4. Preliminary Investigation

In this section, to confirm the feasibility of the adaptive packet compression strategy, we preliminarily investigate the packet compression ratio and compression processing time in an actual environment. This investigation helps to determine a meaningful range of important parameters in evaluating the proposed scheme in the following sections.

4.1 Actual Compression Ratio

First, we preliminarily investigate the packet compression ratio using actual network data to show the effectiveness of packet (i.e., small data) compression. In our experiments, we used two datasets: (1) a dataset for a network covering a broad swath of Asia [21], which was captured in December 2006, and (2) a dataset for a network at a university with approximately 6,000 students, which was captured by our experiments in May 2009. We attempt to compress each captured packet offline using the LZO compression algorithm [17] with the fastest compression level (i.e., the weakest compression). The average compression ratio of all packets in both datasets is 0.945. In general, compressing small data may not be remarkably effective, but this experiment shows that the packet size can be reduced about 5% on average by packet compression in an actual environment. Note that the average compression ratio is defined as $\frac{\text{The sum of compressed packets in bytes}}{\text{The sum of uncompressed original packets in bytes}}$.

Furthermore, we consider the possibility of eliminating incompressible packets such as already-compressed or encrypted packets (taking into account a non-expansion policy described in Ref. [4]) based on a simple port-filtering approach. In our experiment, we first classify all packets into groups (classes) identified by protocol and port numbers. Each packet usually belongs to two different packet classes according to its source port number and destination port number. Then, we examine the average compression ratio of each packet class and register incompressible packet classes (i.e., those with an average compression ratio of more than 1.0) to a compression exception list. In this case, 22.2% of all packets belong to incompressible classes, so the remaining 77.8% of all packets can be compressed. Note that application classification based on port numbers is one of the simplest approaches but is not expected to be precise. Other more sophisticated but costly approaches have been studied in intrusion detection research [22].

To investigate the effectiveness of the exception list, we compress all packets except incompressible packets (those belonging to classes in the exception list). The average compression ratio of both datasets is improved to 0.929. That is, with the compression exception list, we have a strong potential to eliminate incompressible packets and apply packet compression effectively in an actual environment, although we need to construct and maintain the exception list in an online, lightweight manner.

For more detailed analysis, we show a preliminary result of packet compression in each packet class (protocol) identified by the Internet Assigned Numbers Authority (IANA) [23]. Table 1 shows the average compression ratio of compressible packet classes. From each compressible class, we extract a "filtered compressible class" by eliminating all the incompressible packets (i.e., those that also belong to other incompressible classes) from the original compressible class. Then, the "Original traffic" field indicates the ratio of all packets in each original compressible class to all packets in the entire datasets (in bytes), while "Compressed traffic" indicates the proportion of all packets in each filtered compressible class. "Average compression ratio" indicates the average compression ratio of each filtered compressible class. We chose the top 20 IANA-identified classes in descending order of the amount of traffic, and we list them in ascending order of the average compression ratio. Notably, some already-encrypted packets (TCP/22) can be compressed, and some packets of major applications (TCP/25 and UDP/53) can be remarkably compressed.

Through the above experiments, we show that packet-by-packet compression can reduce packet data size to some extent even though the compression ratio of packet compression is not as high as that of block compression, such as IPzip. Furthermore, the compression exception list improves the compression ratio even if it uses a simple port-filtering approach that excepts incompressible packets as much as possible.

4.2 Actual Compression Processing Time

Next, we preliminarily examine the compression processing time, which, in addition to the compression ratio, affects the overall performance of the proposed scheme. We use open benchmark information regarding data compression algorithms and calculate an approximation of the compression processing time for one packet.

According to benchmark results [17], the compression processing time per 500-byte packet is approximately 100 μs in an Intel Pentium 133 MHz processor. This compression processing time is about that needed to transfer one packet over a link of 40 Mb/s. Therefore, if the link bandwidth increases, compression processing time can become a bottleneck. Note that although such a processor has already become legacy hardware, it may be used in small devices, rather than personal computers.

Table 1 Compression ratio of actual traffic.

Protocol	Port number	Average compression ratio	Compressible traffic	Original traffic	IANA description [23]
UDP	0	0.046	0.9%	0.9%	Reserved
TCP	0	0.721	0.3%	0.3%	Reserved
TCP	25	0.820	1.9%	2.0%	Simple Mail Transfer
UDP	53	0.876	0.5%	0.6%	Domain Name Server
TCP	5050	0.880	0.1%	0.1%	multimedia conference control tool
TCP	80	0.922	43.0%	49.1%	World Wide Web HTTP
TCP	3128	0.949	0.3%	0.3%	Active API Server Port
UDP	3130	0.959	0.2%	0.2%	ICPv2
TCP	8080	0.962	14.7%	15.7%	HTTP Alternate (see port 80)
TCP	8083	0.972	0.3%	0.3%	Utilistor (Server)
TCP	5002	0.977	0.2%	0.2%	radio free ethernet
TCP	873	0.978	1.1%	1.1%	rsync
TCP	21	0.981	0.8%	1.1%	File Transfer (Control)
TCP	5004	0.988	0.1%	0.1%	RTP media data (RFC 3551)
TCP	20	0.992	5.8%	7.6%	File Transfer (Default Data)
TCP	4000	0.994	0.4%	0.4%	Terabase
TCP	22	0.996	1.2%	1.7%	The Secure Shell (SSH) Protocol
TCP	23	0.996	0.3%	0.5%	Telnet
TCP	4001	0.997	0.2%	0.2%	NewOak
TCP	4662	0.998	2.7%	2.9%	OrbitNet Message Service

According to other information [18] obtained using high-performance CPUs (Intel Core 2 Duo 2.6 GHz and AMD Athlon64 6400+ processors), the compression processing time per 500-byte packet is approximately 5 μs, which is 20 times faster than the previous result, although the result depends on a wide variety of compression algorithms. Such recent processors may be adaptable for high-speed links such as 1.0 Gb/s links.

In addition to analyzing the benchmark information, we performed an experiment for compression processing. Using a medium-performance CPU (Intel Pentium 4 3.20 GHz), we found that the compression processing time of the LZO compression algorithm per 500-byte packet is approximately 20 μs.

These results suggest that the compression processing time depends strongly on CPU hardware, and so does the applicable range of link speeds of every compression scheme, including the proposed scheme.

5. Evaluation

In previous sections, we mentioned the rationale and issues regarding data compression inside the network. In this section, to prove its potential, we evaluate the proposed adaptive packet compression scheme with multiple advanced relay nodes through computer simulation. Note that rigorous analytical approaches based on queueing theory are difficult to apply due to a state explosion caused by the dynamic selection of compression depending on the current queue length. In Sect. 5.1, we show our simulation model. In Sect. 5.2, we demonstrate the characteristics and effectiveness of the proposed scheme by comparison with other schemes. Then, through performance evaluations with changing simulation parameters in Sect. 5.3, we present its adaptability to various networks and analyze significant factors necessary to achieve efficient packet relay. Finally, in Sect. 5.4, we summarize the simulation results.

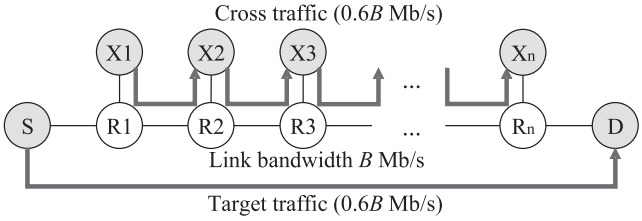


Fig. 2 Simulation topology.

5.1 Simulation Model

In this section, we first present a simulation model to compare three schemes. The first is a no-compression scheme (simply relaying all packets without packet compression), the second is an all-compression scheme (compressing all packets relayed), and the third is the proposed scheme (compressing some packets adaptively selected during relay). In this evaluation, we use the network simulator ns-2 (version 2.31) [24].

Figure 2 depicts the simulated network topology, the so-called parking-lot topology, where there are n advanced relay nodes; we assume that congestion occurs in links between advanced relay nodes R_i and R_{i+1} ($1 \leq i \leq n-1$). The bandwidth of all links is B Mb/s, and the link propagation delay time is zero (i.e., we focus on the transmission delay time). We set the packet size to 500 bytes and use UDP packets for simplicity. Note that the important value is the ratio of the relay processing speed to the compression processing speed, rather than the actual value of the relay processing speed. We show the effect of the compression processing speed later, in Sect. 5.3.4. Target traffic moves from source node S toward destination node D; cross traffic moves from X_i to X_{i+1} and passes through the link between

R_i and R_{i+1}. In this simulation, we use two traffic streams (target and cross traffic) and examine the packet delay time and packet discard rate of target traffic as performance indices representing the degree of congestion inside the network.

Except for the no-compression scheme, incoming packets can be compressed at relay nodes, R_1, R_2, ..., and R_{n-1}. In the all-compression scheme, only R_1 compresses all incoming packets of target traffic from S, while only R_i compresses all incoming packets from X_i. On the other hand, in the proposed scheme, each relay node compresses any incoming packet if and only if the packet is not compressed at a previous node and waits for a while in the output queue on the present node.

In this simulation, we make two assumptions regarding compression processing. The first is that the advanced relay nodes can compress each packet with a constant compression ratio, although the actual compression ratio varies packet by packet depending on the type of data content. For example, if the compression ratio is set to 0.7, any uncompressed packet of 500 bytes can be compressed to 350 bytes when compression is applied. The pattern (distribution) of the packet compression ratio in time and space should be thoroughly investigated using actual network traffic in the future; however, it is beyond the scope of this paper, which focuses on analyzing the basic properties of adaptive packet compression inside the network. The second assumption is that the decompression processing speed is sufficiently faster than the compression processing speed. Since the decompression processing speed is generally much faster than the compression processing speed, we also assume that the decompression processing time can be ignored. For example, according to Refs. [17], [18], the decompression processing speed is 3 to 120 times faster or 2 to 24 times faster than the compression processing speed, depending on the compression level or type of data in the LZO compression algorithm [17] adopted in the open-source virtual private network (VPN) tool OpenVPN [25]. Moreover, the compression processing time varies depending on the compression level, whereas the decompression processing time is relatively constant. In addition, a novel technique to decrease the decompression speed has been developed in Ref. [26]. Therefore, we ignore the decompression processing time and focus only on the compression processing time for simplicity.

Table 2 shows default parameters used in Sects. 5.2 and 5.3 for the simulation. We set the link bandwidth B to 100 Mb/s. The source nodes S and X periodically start and stop to transmit packets with average burst-on and burst-off intervals of 250 μs and 50 μs, respectively, using an ns-2 traffic generator, `Application/Traffic/Exponential`. The packet arrival interval in the burst-on period is 67 μs (60 Mb/s). Therefore, the total peak (short-term) transmission rate from nodes S and X is 120 Mb/s, while the total average (long-term) transmission rate including the burst-off period is 100 Mb/s. Since these traffic streams from nodes S and X attempt to pass through one 100 Mb/s output link, this incoming traffic causes congestion at the advanced relay nodes. In addition, we set the number of advanced relay nodes to two and the buffer size of one advanced relay node to 1,000 packets. The compression processing time is 200 μs, five times the packet transmission time of 40 μs on a 100 Mb/s link, considering other overheads, e.g., obtaining the queue length and determining compression processing packet by packet, taking into account the results in Sect. 4.2. All simulation evaluations run 10 times with different random seeds. Note that we evaluate the proposed scheme by changing simulation parameters for performance evaluation in Sect. 5.3.

Table 2 Default simulation parameters used in Sects. 5.2 and 5.3.

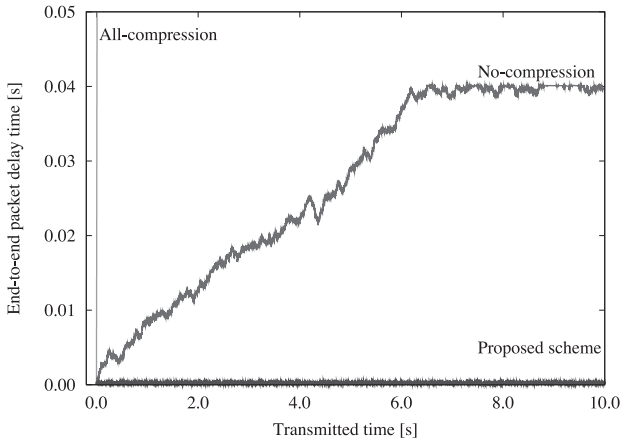
Simulation parameter	Default value
Link bandwidth	100 Mb/s
Average burst-on interval	250 μs
Average burst-off interval	50 μs
Packet length	500 Byte
Packet arrival interval	67 μs/packet
(Packet arrival rate from one node)	(60 Mb/s)
Buffer size	1,000 packet
Number of advanced relay nodes	2
Compression processing time	200 μs/packet

5.2 Behavior Analysis

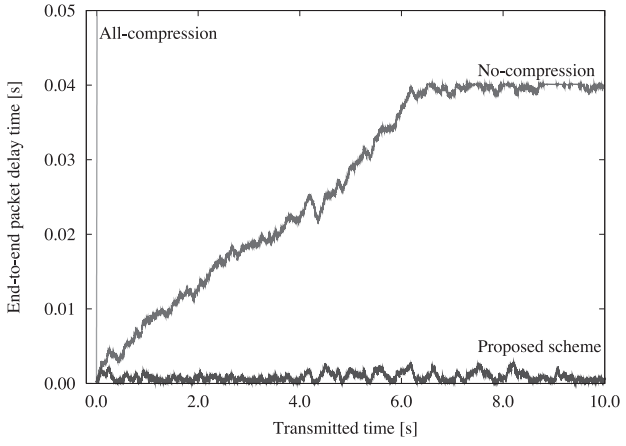
First, we compare the three schemes using burst traffic to demonstrate the effectiveness of the proposed scheme and to show difference of behaviors among the three schemes. In this section, we show how the proposed scheme improve the performance. Note that, to analyze the basic properties of the proposed scheme, we do not use TCP traffic in this simple simulation scenario, while the evaluation for burst traffic may suggest the effectiveness of the schemes on traffic including burst TCP flows. In the future work, we will try a more realistic performance evaluation in dynamic and complex environments that include a number of TCP flows.

Figure 3 shows time-series results for the three schemes. We consider two scenarios: the compression ratio is set to 0.70 (strong compression) in the first and to 0.95 (weak compression) in the second. In Figs. 3(a) and 3(b), the x-axes represent the packet transmission time from S; the y-axes represent the packet delay time from S to destination node D. Both figures show an increase in the delay time for all the schemes. Simulation parameters are set as described in Table 2.

In the first scenario, depicted in Fig. 3(a), the packet delay time of the all-compression scheme is larger than that of the other schemes because compression processing produces a persistent bottleneck. The delay time of the all-compression scheme drastically increases, approaching a maximum of 0.2005 second at around 0.1 second. Similarly, the delay time of the no-compression scheme gradually increases and approaches a maximum of 0.0401 second at around 6.0 seconds. On the other hand, the pro-



(a) Compression ratio: 0.70



(b) Compression ratio: 0.95

Fig. 3 Comparison of behavior in congestion period.

posed scheme can suppress the increase in the delay time to less than 0.0008 second. To compare all the schemes without transient characteristics, we define a steady state as a 20-second period from 10 to 30 seconds into the simulation. Hereafter, we evaluate the performance of all schemes during the steady state. The packet delay time of the proposed scheme is also the smallest among the three schemes, and the average delay times in the all-compression, no-compression, and proposed schemes are 0.2000, 0.0392, and 0.0003 second, respectively. The packet discard rates in the all-compression and no-compression schemes are 0.800 and 0.004, respectively, during the steady state. In contrast, the packet discard rate of the proposed scheme is the smallest at 0.000. Regarding adaptive packet compression, the number of packets compressed at R_1 over the steady state is 14,760 out of 250,913 packets for target traffic and 14,523 out of 251,339 packets for cross traffic in the steady state; i.e., R_1 selectively compresses 5.8% of incoming packets.

Regarding the second compression ratio scenario, shown in Fig. 3(b), the overall trend in the results is similar to that in Fig. 3(a). The delay performance of the all-compression scheme is almost the same as in the first scenario. The maximum delay times of the all-compression, no-compression, and proposed schemes are 0.2005, 0.0401, and 0.0044 second, respectively; the average delay times are 0.2005, 0.0392, and 0.0009 second, respectively. Although the delay time of the proposed scheme is higher than in the first scenario, the proposed scheme still achieves the best performance. The packet discard rate is the same as in the first scenario; i.e., that in the all-compression, no-compression, and proposed schemes is 0.800, 0.004, and 0.000, respectively, during the steady state. As for adaptive packet compression, the number of compressed packets at R_1 over the steady state is 39,852 out of 250,913 packets for target traffic and 39,557 out of 251,339 packets for cross traffic; i.e., R_1 selectively compresses 15.8% of incoming packets.

To clarify why the proposed scheme in the second scenario (i.e., poor compression ratio) needs to compress incoming packets more aggressively than the first scenario, we confirm the queue length at R_1 in both scenarios. The average queue length at R_1 in the first and second scenarios is 3 and 17, respectively, in the proposed scheme; the maximum queue length at R_1 in the first and second scenarios is 17 and 69, respectively. R_1 can compress incoming packets when more than five packets remain in the queue, because the compression processing speed is five times slower than the transmission speed. Therefore, to alleviate traffic congestion inside the network, the proposed scheme in the second scenario needs to compress many packets.

As these results show, the proposed scheme can adaptively compress some but not all incoming packets. Here, we confirm how many incoming packets can be compressed in the proposed scheme. First, to calculate the maximum number of compressed packets, we assume that the queue is filled with incoming packets; i.e., it contains 1,000 packets based on the simulation parameters in this evaluation. In this case, first, 5 packets out of the 1,000 in the CPU queue move to the relay queue with no additional delay time, as depicted in Fig. 1 in Sect. 3 because Eq. (2) cannot be satisfied. Then, the next packet can be compressed in parallel with the relay processing of the previous five packets. Next, the compressed packet moves to the head of the relay queue because the previous five packets were already transferred during compression processing. Similarly, the subsequent four packets cannot be compressed and move to the relay queue, and then the next packet can be compressed. In fact, when heavy congestion occurs continually, one packet out of N_R packets can be compressed if the following equation is satisfied.

$$(N_R - 1) \cdot \frac{S}{B} + R \cdot \frac{S}{B} = C.$$

Therefore, N_R can be calculated as

$$N_R = C \cdot \frac{B}{S} + (1 - R).$$

Consequently, the ratio of the maximum number of com-

pressed packets to the number of incoming packets is given by

$$P_R = \frac{1}{N_R} = \frac{1}{C \cdot \frac{B}{S} + (1 - R)}. \quad (3)$$

Based on Eq. (3), $P_{0.70}$ and $P_{0.95}$ can be calculated at 18.9% and 19.8%, respectively. On the other hand, the minimum number of compressed packets is 0.0% in both cases because the averaged incoming traffic rate is equal to the output speed, such as 100 Mb/s in this evaluation. The two simulation results are smaller than $P_{0.70}$ and $P_{0.95}$, respectively. In other words, both results are within the scope between the theoretical lower and upper bounds.

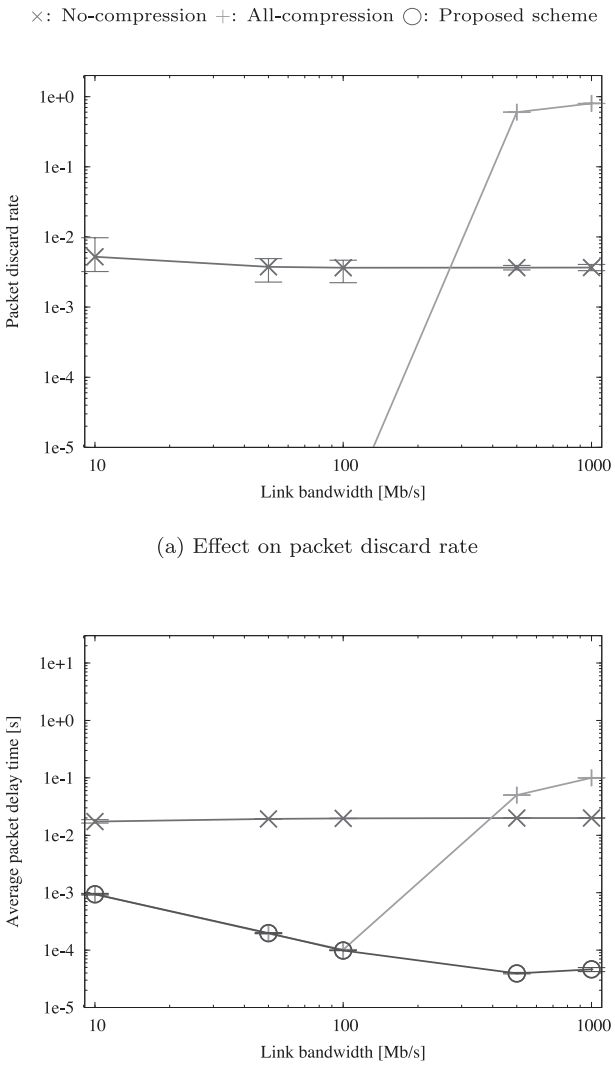
The above-mentioned results show that the proposed scheme determines whether a packet can be compressed by evaluating the effectiveness of packet compression at a given time. As a consequence, the proposed scheme achieves a lower packet delay time and packet discard rate than the others.

5.3 Performance Analysis

Next, we examine the effect of five factors on the performance of the proposed scheme: (1) link bandwidth, (2) the number of congestion points (any advanced relay node R_i may be a congestion point in the simulated network topology depicted in Fig. 2), (3) the compression ratio, (4) the compression processing time, and (5) burst traffic. Through these evaluations, we show that the proposed scheme can be adapted to various network environments, and we analyze the factors necessary to achieve better performance than the other schemes. In Figs. 4–10, the y-axis in (a) represents the minimum, average, and maximum values of the packet discard rate during the steady state as described in Sect. 5.2; the y-axis in (b) represents the minimum, average, and maximum values of the packet delay time per advanced relay node (here, we define the average packet delay time as this value) during the steady state in the simulation. Since the link propagation delay time is set to zero, as described in Sect. 5.1, the average packet delay time means the waiting time spent in the queue in this simulation. In all figures, the cutoff at 10^{-5} represents the value of 0.0 because it cannot be plotted in log-scale figures. We describe the the x-axes in each case individually because they differ.

5.3.1 Link Bandwidth

In this section, we show the adaptability of the proposed scheme in various network environments. To show the effective area of the proposed scheme, we vary the link bandwidth from 10 to 1,000 Mb/s, assuming various compression speeds as investigated in Sect. 4.2. We show three results, for high-speed, medium-speed, and low-speed compression processing times (2, 20, and 200 μs) in Figs. 4, 5, and 6, respectively. On the other hand, the packet transmission time at 10 to 1,000 Mb/s varies from 400 to 0.4 μs. In



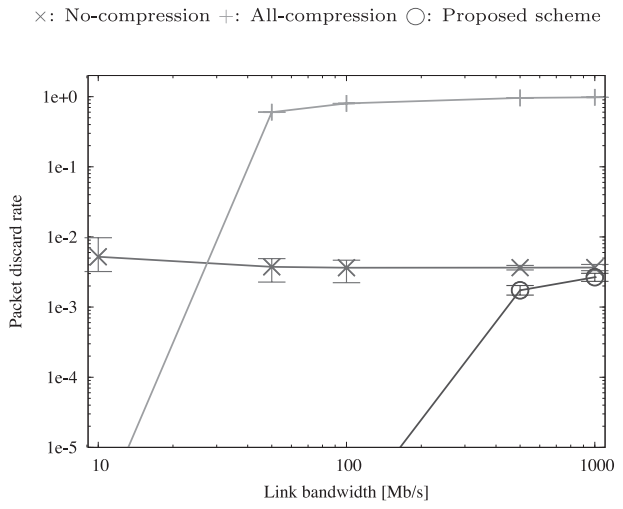
(a) Effect on packet discard rate

(b) Effect on average packet delay time

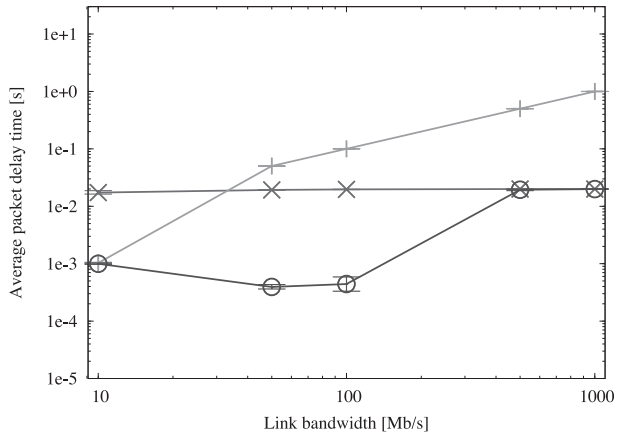
Fig. 4 Effective area of high-speed compression processing time (2 μs/packet).

this simulation, we change the total incoming traffic rate at a rate proportional to the link bandwidth (1.2B Mb/s incoming traffic rate vs. 1.0B Mb/s output link bandwidth). Similarly, we also change the buffer size on all output links (a 10B-packet buffer on a B Mb/s link); i.e., the waiting time of a packet at the end of a queue is the same in all the simulation topologies. All the simulation parameters except the link bandwidth and buffer size are set as described in Table 2.

Across all the results, the proposed scheme can improve performance in narrow-bandwidth environments because the compression processing time is sufficiently faster than the packet transfer time. On the other hand, the performance of the all-compression scheme drastically degrades as the link bandwidth increases. With the no-compression scheme, the performance cannot be improved because the packet relay becomes a persistent bottleneck. In contrast, the performance of the proposed scheme is nearly the same

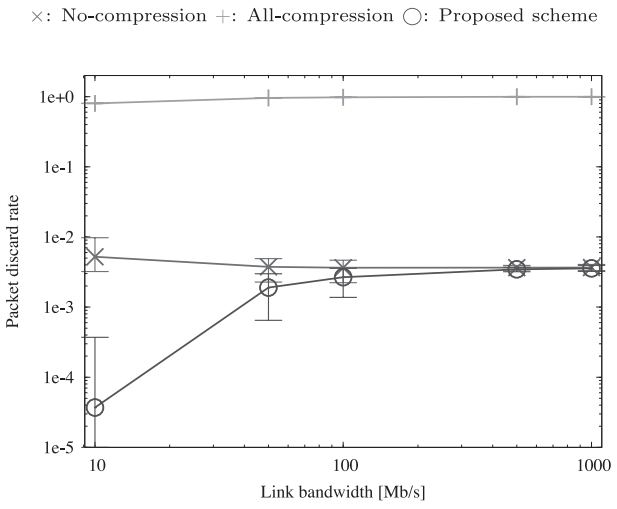


(a) Effect on packet discard rate

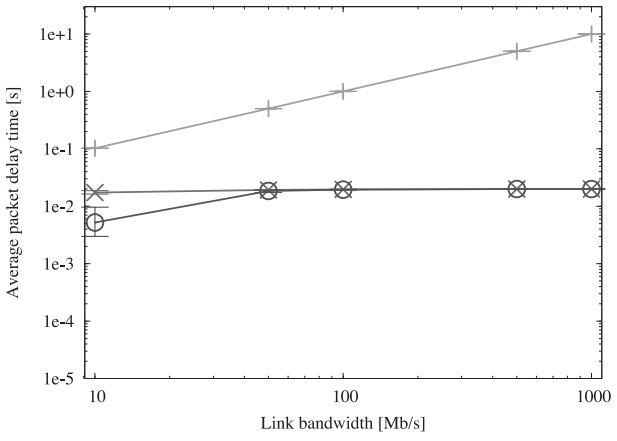


(b) Effect on average packet delay time

Fig. 5 Effective area of medium-speed compression processing time (20 μs/packet).



(a) Effect on packet discard rate



(b) Effect on average packet delay time

Fig. 6 Effective area of low-speed compression processing time (200 μs/packet).

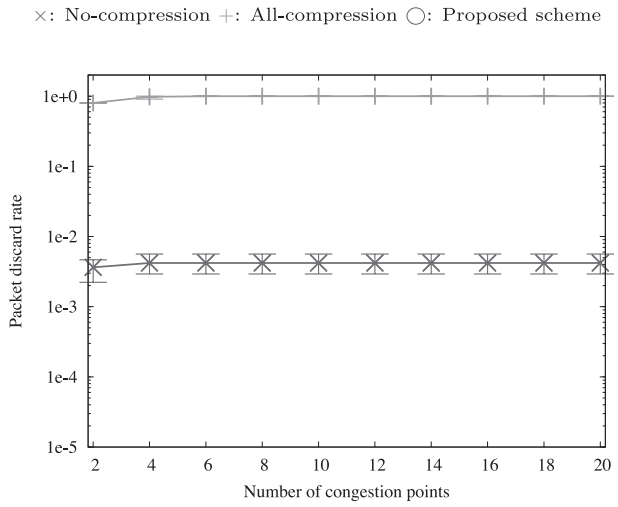
as or better than that of the others despite its performance degradation. This is because the proposed scheme changes its compression strategy into all-compression, selective-compression, and no-compression depending on the situation. For example, in a situation with a 1,000 Mb/s link, the performance of the proposed scheme comes close to that of the no-compression scheme because the proposed scheme reduces its compression frequency. Nevertheless, in this situation, the proposed scheme achieves the best performance; i.e., the packet discard rate and average packet delay time decrease by 100.0% and 99.8%, respectively, in Fig. 4, 27.1% and 0.2% in Fig. 5, and 2.5% and 0.0% in Fig. 6, compared with the no-compression scheme.

These results show that adaptive use of multiple resources (computational and network resources in this case) can be an effective strategy to avoid producing a persistent bottleneck resource that leads to traffic congestion inside networks. In particular, the proposed scheme can achieve efficient packet relay over a wide range of link bandwidths even with low-speed compression processing.

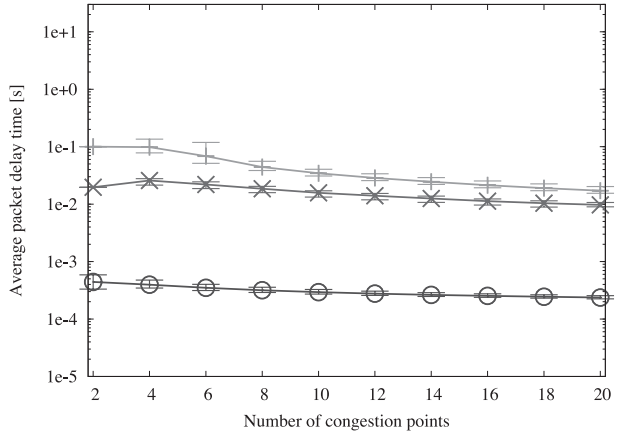
5.3.2 Number of Congestion Points

Next, we analyze how the network's scale affects performance, in particular when multiple advanced relay nodes become congestion points. The effect of the number of congestion points is shown in Figs. 7(a) and 7(b). The *x*-axes indicate the number of congestion points, which is set from 2 to 20 at intervals of 2. All the simulation parameters except the number of congestion points (advanced relay nodes) are set as described in Table 2.

Figure 7(a) shows that the packet discard rate increases slightly in the no-compression and all-compression schemes, but its increment becomes zero as the number of



(a) Effect on packet discard rate



(b) Effect on average packet delay time

Fig. 7 Effect of number of congestion points.

congestion points increases. This is due to heavy congestion at certain relay nodes, described as follows. In this simulation scenario, packet discard occurs many more times at ingress-side nodes than at egress-side nodes. Target traffic first passes through R_1, then passes through congestion points a number of times, and finally passes through R_n. Therefore, if i is smaller than j, the degree of congestion at R_i is larger than that at R_j, which reduces the packet discard rate at the egress side.

Figure 7(b) shows that the average packet delay time decreases as the number of congestion points increases. Since the all-compression scheme compresses all packets, the gradient of the decrement in that case is greater than those of the other two schemes. The proposed scheme slightly decreases the average packet delay, even with the lower decrement. With no compression, because the number of congestion points increases and packet discard occurs many times, the average packet delay time of the surviving packets decreases.

Finally, we present the number of packet compression counts in the proposed scheme. We consider 10 advanced relay nodes as an example; in this case, 9 nodes (R_1 to R_9) can compress incoming packets. The number of packets compressed in each node is shown in Table 3. The first column represents advanced relay nodes. The second and third columns represent the number of compressed packets of target and cross traffic, respectively. The fourth and fifth columns represent the number of incoming packets of each type of traffic. The sixth column gives the ratio of compressed packets to incoming packets. Many packets transmitted from node S are compressed not only at R_1 but also at every advanced relay node (R_2 to R_9). Furthermore, the number of packets compressed at R_i decreases as i increases because the congestion at egress-side nodes can be alleviated by adaptive compression compared with that at ingress-side nodes. This result shows that the proposed scheme attempts to exploit the positive effect of compression in all advanced relay nodes, even if one node determines that compression is not effective.

5.3.3 Compression Ratio

Next, we present the effect of the compression ratio. Our preliminary investigation has already shown the actual compression ratio obtained by packet compression in Sect. 4.1. Nevertheless, we think we should analyze the effect of the compression ratio for the following reason. In recent years, many researchers have been developing novel compression algorithms to improve the compression ratio to adapt to future network environments [27], [28]; hence, we assume that the compression ratio of novel algorithms in future networks can be improved and varied over a wide range.

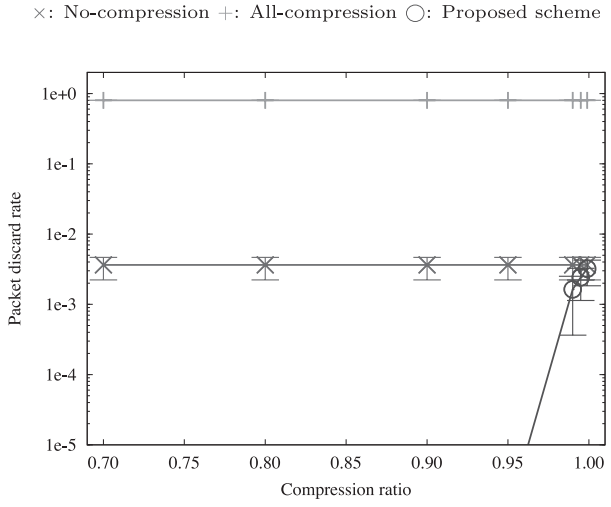
The effect of the compression ratio is shown in Figs. 8(a) and 8(b). The x-axes indicate the compression ratio; these values vary from 0.7 to 0.9 at intervals of 0.1, and then include 0.95, 0.99, 0.995, and 0.999. With no compression, these values do not affect the performance. All the simulation parameters except the compression ratio are set as described in Table 2.

To explain these results, we first focus on the all-compression scheme. As the compression ratio decreases, the performance of this scheme would be expected to improve. However, the packet discard rate is precisely the same in all ranges in Fig. 8(a), and the average packet delay time is approximately the same in all ranges in Fig. 8(b).

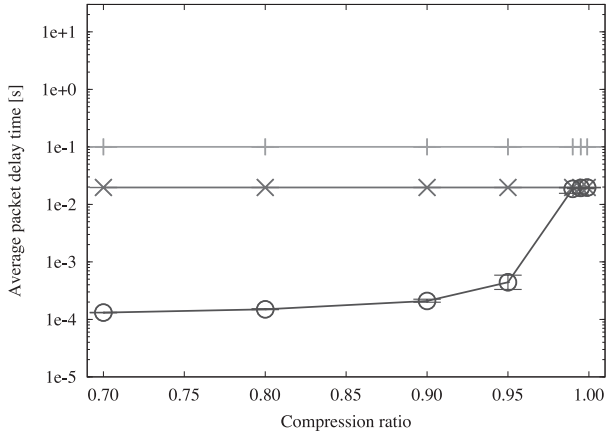
Next, we focus on the proposed scheme. Below 0.80 on the x-axis (i.e., more than 20% data size reduction), the average packet delay time of the proposed scheme is approximately the same. In contrast, above 0.80 (i.e., less than 20% data size reduction), the average packet delay time of the proposed scheme increases. However, the average packet delay time of the proposed scheme decreases by 5.3% and 97.8% in comparison with that of the no-compression scheme when the compression ratio is 0.99 and 0.95, respectively.

Table 3 Number of compressed packets at R_1–R_9.

Relay node R_i	Number of compressed packets (N_c)		Number of incoming packets (N_i)		Ratio (N_c/N_i)
	Target ($S \rightarrow R_i$)	Cross ($X_i \rightarrow R_i$)	Target ($S \rightarrow R_i$)	Cross ($X_i \rightarrow R_i$)	
R_1	39,852	39,557	250,913	251,339	15.8%
R_2	25,477	33,385	250,913	251,058	11.7%
R_3	19,531	31,669	250,913	251,397	10.2%
R_4	15,120	28,057	250,913	251,604	8.6%
R_5	12,209	25,566	250,913	251,474	7.5%
R_6	9,368	22,388	250,913	251,279	6.3%
R_7	8,333	21,606	250,913	251,574	6.0%
R_8	7,360	20,734	250,913	251,191	5.6%
R_9	6,140	18,770	250,913	251,117	5.0%



(a) Effect on packet discard rate



(b) Effect on average packet delay time

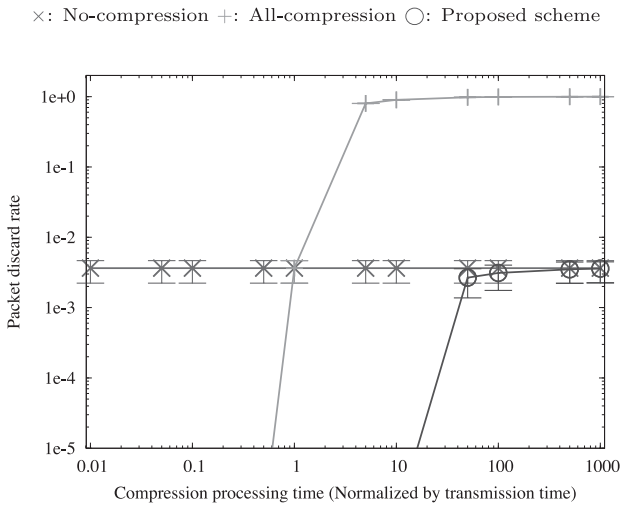
Fig. 8 Effect of compression ratio.

Based on these results, we show that the all-compression scheme cannot improve performance due to the computational resource bottleneck even in the case of strong compression. On the other hand, the proposed scheme with strong compression can remarkably improve performance. Surprisingly, the proposed scheme has an advantage even if the effect of compression is small, i.e., the compression ratio is high. For example, in Fig. 8(a), the packet discard rate was reduced by 11.9% and 33.5% even when the compression ratios were 0.999 and 0.995.

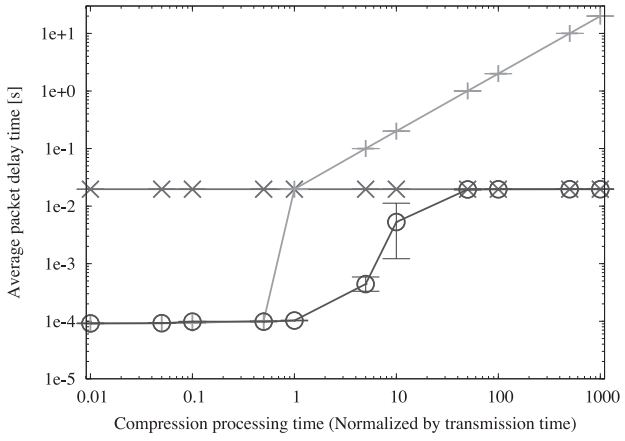
5.3.4 Compression Processing Time

Next, assuming various network environments where the advanced relay nodes have various types of network and computational resources, we show the effect of compression processing time in Figs. 9(a)and 9(b). The x-axes indicate the compression processing time normalized by packet transmission time (40 μs, i.e., 500-byte transmission on a 100 Mb/s link); we vary the values from 0.01 to 1,000. With no compression, these values do not affect the performance. All the simulation parameters except the compression processing time are set as described in Table 2. Since we focus on the ratio of compression processing time to transmission time, this evaluation can be expected to apply not only to current network environments but also to those of the future.

Here, we classify all the results into three areas. The first comprises results in the range from 0.01 to 0.5 on the x-axis. In this case, the bottleneck is relay processing; therefore, the result of the proposed scheme is almost the same as that of the all-compression scheme in terms of the packet discard rate and average packet delay time. In other words, the proposed scheme compresses almost all the incoming packets before forwarding them. The second area comprises results ranging from 1 to 100 on the x-axis. In this range, selective packet compression is most effective, because the bottleneck cannot be determined to be either relay processing or compression processing. In particular, in the range from 1 to 10, the packet discard rate becomes zero. Even when the normalized compression processing time is 100, the packet discard rate and average packet delay time of the proposed scheme decrease by 14.5% and 0.2%, respectively, compared with the no-compression scheme. The third area comprises results ranging from 500 to 1,000; here, the bottleneck is compression processing. In this case, the packet discard rate and average packet delay time of the proposed scheme are almost the same as those of the no-compression scheme. In contrast to the first area, the proposed scheme only relays most incoming packets without compression.

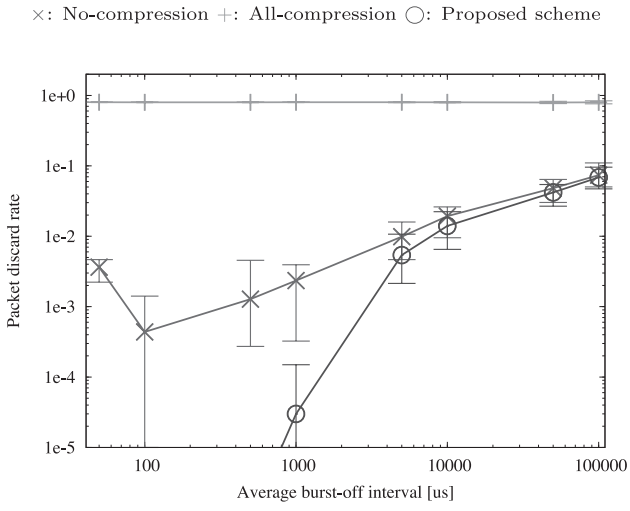


(a) Effect on packet discard rate

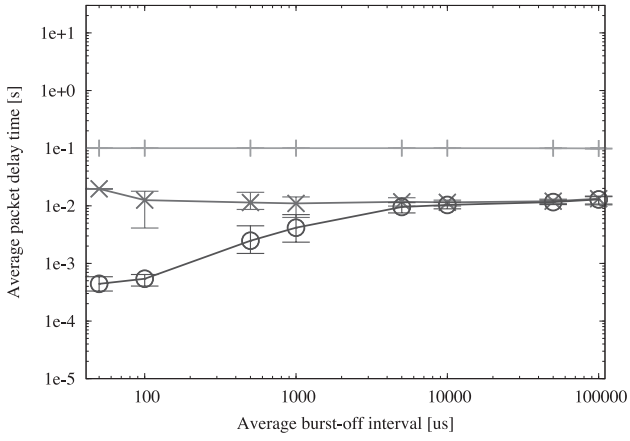


(b) Effect on average packet delay time

Fig. 9 Effect of compression processing time.



(a) Effect on packet discard rate



(b) Effect on average packet delay time

Fig. 10 Effect of burst traffic.

These results demonstrate that the adaptive online lossless packet compression of the proposed scheme adequately controls the packet compression frequency based on relay and compression processing time during congestion periods. The proposed scheme can deliver no compression, selective packet compression, or compression of all packets according to the instantaneous conditions on each node in various types of network environments.

5.3.5 Burst Traffic

Finally, we show the effect of burst traffic in Figs. 10(a) and 10(b). The x-axes indicate the average burst-off interval; the average burst-on interval is five times the burst-off interval. All the simulation parameters except the average burst-on and burst-off intervals are set as described in Table 2.

In these results, the performance of the no-compression and proposed schemes degrades as the average burst-on and burst-off interval increases, because this increase leads to heavy congestion at the output link. However, compared with the no-compression scheme, the proposed scheme shows better performance in all ranges. When the burst-off interval is 1,000 μs, the packet discard rate and average packet delay time of the proposed scheme decrease by 27.4% and 9.6%, respectively, compared with the no-compression scheme. Furthermore, when the burst-off interval is 100,000 μs, the packet discard rate and average packet delay time decrease by 7.0% and 2.6%, respectively. These results show that the proposed scheme can be effective even when heavy congestion occurs.

5.4 Simulation Summary

In Sect. 5, we have presented simulation results to evaluate the adaptive compression scheme. In this section, we summarize the results before presenting concluding remarks.

First, in Sect. 5.1, we showed the simulation model for comparing the proposed scheme to the all-compression and no-compression schemes. In this model, the all-compression and no-compression schemes had persistent bottlenecks in terms of computational and network resources, respectively. Hence, the adaptive compression strategy is essential to avoid persistent bottleneck resources.

In the first simulation results, shown in Sect. 5.2, we demonstrated the characteristics and effectiveness of the proposed scheme. These results showed that the proposed scheme could adaptively determine whether to compress incoming packets, and it alleviated network congestion, so that it could decrease the packet delay time and packet discard rate.

In the next simulation results for the link bandwidth and number of congestion points, shown in Sects. 5.3.1 and 5.3.2, respectively, we showed the adaptability of the proposed scheme to various networks including a range of network and computational resources. Furthermore, even if congestion points increased, the proposed scheme outperformed the other schemes.

Then, we analyzed the significant factors necessary to improve the performance in Sects. 5.3.3 and 5.3.4, and we showed the effect of the compression ratio and compression processing time, respectively. The proposed scheme performed better independently of the compression ratio; i.e., it could decrease the packet discard rate even under poor compression ratios. Furthermore, we showed that the proposed scheme could switch its behavior based on relay and compression processing time. Hence, it showed improved performance in various network environments broadly.

Finally, we analyzed the performance improvement of the proposed scheme in heavy congestion scenarios in Sect. 5.3.5. Even when heavy congestion occurred, the proposed scheme performed better. Notwithstanding performance degradation, the proposed scheme outperformed the all-compression and no-compression schemes.

Through simulation evaluations, we showed that the proposed scheme can improve the packet discard rate and packet delay time with a wide range of simulation parameters. In particular, we evaluated the effectiveness of the proposed scheme in networks with various link bandwidth from 10 Mb/s to 1,000 Mb/s. These results suggest that the proposed scheme can be effective in various network environments, e.g., wireless mesh backbone networks, campus networks, and corporate networks. For future networks aiming to be dependable information infrastructures, both network performance and adaptability to various network environments are essential. Therefore, we conclude that all the simulation results suggest a significant benefit to future networks from the proposed scheme.

6. Concluding Remarks

In this paper, we have proposed an adaptive online lossless compression scheme enabled by advanced relay nodes located inside a network. Our proposed scheme uses the positive effect of compression and eliminates its negative effect. Specifically, our proposed scheme employing adaptive packet compression selectively compresses a packet based on its waiting time in the queue during a congestion period.

To show the effectiveness of packet compression rather than block compression, we have preliminarily examined the packet compression ratio using two actual traffic datasets. Then, using computer simulations, we have shown that the proposed scheme can reduce the packet delay time and discard rate. In particular, the benefit of the proposed scheme was seen even if the compression ratio was high (for example, only 0.1% and 0.5% data size reduction), especially for the packet discard rate. Moreover, even when heavy congestion occurred, the proposed scheme performed better than the all-compression and no-compression schemes, e.g., showing an 7.0% decrease in the packet discard rate. Through our simulations, we have shown that the ratio between relay and compression processing time, as well as the compression ratio, is an important factor in improving performance. Even with undesirable values (i.e., large compression processing time and/or high compression ratio), the proposed scheme can outperform the other schemes. These simulation results show that the proposed scheme can be effective in a wide parameter range. Furthermore, the basic feature of the proposed scheme, i.e., adaptive computational processing during waiting time at advanced relay nodes, is expected to be significantly useful in a variety of aspects in future networks.

In this paper, to clarify the effectiveness and strong potential of this new approach, we evaluated the proposed scheme using a simple and controllable model rather than through detailed implementation in realistic network environments. We plan to the proposed scheme in a large-scale topology and in scenarios including TCP traffic. Moreover, we will construct a model for variation of the compression ratio and processing time.

Acknowledgment

This work was partly supported by the National Institute of Information and Communications Technology, and the Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research (B) (No. 21300024). We thank Asian Internet Interconnection Initiatives (AI3).

References

[1] M. Shimamura, T. Ikenaga, and M. Tsuru, "Compressing packets adaptively inside networks," Proc. IEEE/IPSJ 9th Annual International Symposium on Applications and the Internet (SAINT2009), pp.92–99, July 2009.

[2] Cisco Systems, Inc., "Cisco visual networking index: Forecast and methodology, 2008–2013," June 2009. Information available at http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf

[3] L. Qiu, Y.R. Yang, Y. Zhang, and S. Shenker, "On selfish routing in Internet-like environments," IEEE/ACM Trans. Netw., vol.14, no.4, pp.725–738, Aug. 2006.

[4] A. Shacham, B. Monsour, R. Pereira, and M. Thomas, "IP payload

compression protocol (IPComp)," RFC 3173 (Standards Track), Sept. 2001. Information available at http://www.ietf.org/rfc/rfc3173.txt

[5] M. Bassiouni, "Data compression in scientific and statistical databases," IEEE Trans. Softw. Eng., vol.11, no.10, pp.1047–1058, Oct. 1985.

[6] C. Krintz and S. Sucu, "Adaptive on-the-fly compression," IEEE Trans. Parallel Distrib. Syst., vol.17, no.1, pp.15–24, Jan. 2006.

[7] S. Chen, S. Ranjan, and A. Nucci, "IPzip: A stream-aware IP compression algorithm," Proc. Data Compression Conference 2008, pp.182–191, March 2008.

[8] H. Xie, Y.R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz, "P4P: Provider portal for applications," Proc. ACM SIGCOMM08, pp.351–362, Aug. 2008.

[9] S. Seetharaman and M. Ammar, "Overlay-friendly native network: A contradiction in terms?," Proc. ACM Fourth Workshop on Hot Topics in Networks (HotNets-IV), Nov. 2005.

[10] National Institute of Information and Communications Technology (NICT), "AKARI: Architecture Design Project for New Generation Network," Oct. 2008. Information available at http://akari-project.nict.go.jp/eng/conceptdesign.htm

[11] National Science Foundation (NSF), "FIND: NSF NeTS Future Internet Design Initiative," Information available at http://www.nets-find.net/

[12] Community Research and Development Information Service (CORDIS), "FIRE: Future Internet Research and Experimentation," Information available at http://cordis.europa.eu/fp7/ict/fire/

[13] N. Chowdhury and R. Boutaba, "Network virtualization: State of the art and research challenges," IEEE Commun. Mag., vol.47, no.7, pp.20–26, July 2009.

[14] I. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," Comput. Netw., vol.47, no.4, pp.445–487, March 2005.

[15] K. Letaief and W. Zhang, "Cooperative communications for cognitive radio networks," Proc. IEEE, vol.97, no.5, pp.878–893, May 2009.

[16] O. Akan, O. Karli, and O. Ergul, "Cognitive radio sensor networks," IEEE Netw., vol.23, no.4, pp.34–40, July 2009.

[17] M. Oberhumer, "LZO — A real-time data compression library," April 2008. Information available at http://www.oberhumer.com/opensource/lzo/

[18] L. Reinhold, "QuickLZ," April 2009. Information available at http://www.quicklz.com/

[19] R. Friend and R. Monsour, "IP payload compression using LZS," RFC 2395 (Informational), Dec. 1998. Information available at http://www.ietf.org/rfc/rfc2395.txt

[20] I. Stoica, S. Shenker, and H. Zhang, "Core-stateless fair queueing: A scalable architecture to approximate fair bandwidth allocations in high-speed networks," IEEE/ACM Trans. Netw., vol.11, no.1, pp.33–46, Feb. 2003.

[21] Asian Internet Interconnection Initiatives, "AI3 project," Information available at http://www.ai3.net/

[22] Y. Lin, C. Lu, Y. Lai, W. Peng, and P. Lin, "Application classification using packet size distribution and port association," J. Network and Computer Applications, vol.32, no.5, pp.1023–1030, Sept. 2009.

[23] Internet Assigned Numbers Authority, "Port numbers," June 2009. Information available at http://www.iana.org/assignments/port-numbers

[24] VINT Project, "The network simulator — Ns-2," Information available at http://www.isi.edu/nsnam/ns/

[25] OpenVPN Technologies, Inc., "OpenVPN," Information available at http://openvpn.net/

[26] Y. Matias and R. Refua, "Delayed-dictionary compression for packet networks," Proc. 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM2005), pp.1443–1454, March 2005.

[27] D. Klinc, C. Hazay, A. Jagmohan, H. Krawczyk, and T. Rabin, "On compression of data encrypted with block ciphers," Proc. Data Compression Conference 2009, pp.213–222, March 2009.

[28] M. Banikazemi, "LZB: Data compression with bounded references," Proc. Data Compression Conference 2009, p.436, March 2009.

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.

Hiroyuki Koga received the B.E., M.E. and Ph.D. 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, Japan. From August 2004 to 2006 he was a researcher in the Kitakyushu JGN2 Research Center, National Institute of Information and Communications Technology, Japan. Since April 2006, he has been an assistant professor in the Department of Information and Media Sciences, Faculty of Environmental Engineering, University of Kitakyushu, Japan. Since April 2009, he has been an associate professor in the Department of Information and Media Engineering, Faculty of Environmental Engineering, University of Kitakyushu, Japan. His research interests include performance evaluation of computer networks, mobile networks, and communication protocols. He is a member of the IEEE.

Takeshi Ikenaga received B.E., M.E. and D.E. degrees in computer science from Kyushu Institute of Technology, Iizuka, 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, Since March 2004, he has been an Associate Professor 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 the IEEE.

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 IEEE, ACM, IPSJ, and JSSST.