PAPER

Adaptive Early Packet Discarding Scheme to Improve Network Delay Characteristics of Real-Time Flows

Kazumi KUMAZOE[†a)], **Masato TSURU**[††], ***Members***, ***and*** **Yuji OIE**[††], ***Fellow***

SUMMARY The performance of a real-time networked application can be drastically affected by delays in packets traversing the network. Some real-time applications impose limits for acceptable network delay, and so a packet which is delayed longer than the limit before arriving at its destination is worthless to the flow to which the packet belongs. Not only that, but the rejected packet is also damaging to the quality of other flows in the network, because it may increase the queuing delay for other packets. Therefore, this paper proposes an adaptive scheme using two mechanisms, in which packets experiencing too great a delay are discarded at intermediate nodes based on the delay limit for the application and the delay experienced by each packet. This earlier discarding of packets is expected to improve the overall delay performance of real-time flows competing for network resources when the network is congested. An extensive simulation is conducted, and the results show that the scheme has great potential in improving the delay performance of real-time traffic in both homogeneous and heterogeneous environments in terms of traffic volume and application delay requirements.

key words: *queuing delay, early packet discarding, real-time application, MTQ, QTL*

1. Introduction

The performance of real-time network applications is greatly affected by delay and loss experienced by packets as they traverse a network. Some real-time applications set limits for acceptable network delay. For example, VoIP defines service classes based on an end-to-end packet delay limit and the rate of packet loss for flows in a network [1]. In such applications, packets delayed longer than an acceptable limit are invalidated by their applications when they reach their destinations, even though they have successfully arrived at the receiver. These packets are useless for their applications, and thus they just impose an excess load on the network. In general, if network resources are exhausted by traffic that will eventually be discarded, significant performance deterioration will be seen even though those resources are fully utilized.

In this paper therefore, by extending our preliminary work [2], an early packet discarding scheme is proposed and its effectiveness in congested networks is demonstrated through simulation. In the scheme, those packets not contributing to the performance of real-time applications are discarded in advance at intermediate nodes. The proposed scheme includes two kinds of router-supported mechanisms: one is called Maximum Transmission Queue Delay (MTQ), which resembles the concept of a Maximum Transmission Unit (MTU), and the other is Queue Delay To Live (QTL), which is similar to the mechanism for Time To Live (TTL). Both of these require the use of an additional header field in IP or UDP packets (e.g. an IPv6 optional header) to convey the queuing delay information for each packet.

It is assumed that a fixed amount of minimum delay (e.g. propagation and transmission delays) can be known or predicted for packets traversing from the sender to the receiver before they are sent, and thus only the variable part of the delay (mainly queuing delays) has been taken into account in this scheme. For example, if the acceptable total network delay for an application is 50 [ms] and the fixed delay on an end-to-end path is 30 [ms], the acceptable total queuing delay is 20 [ms]. It is further assumed that real-time application flows are treated separately at network nodes to other elastic traffic such as TCP flows, in terms of the bandwidth (i.e. the queuing buffer) shared by the flows. Thus, this paper will focus only on real-time flows and their queuing delay in a network. As noted later in Sect. 3, the intention is to apply the proposed scheme not to end equipment for improving performance of individual flows, but to network nodes via a network provider to improve overall network performance (and for increasing the number of flows accommodated). The above assumptions are applicable to such a scenario.

To evaluate the proposed scheme on real-time application flows in terms of increasing the number of packets that could be utilized by the receiver-side application, Effective Packet Loss Rate (EPLR) is introduced. EPLR is equal to one minus the ratio of the number of packets that successfully reach the receiver along a path in experiencing a total queuing delay shorter than the acceptable upper limit induced by the delay-sensitive application.

Since the flows competing for and sharing the network resource have individual different conditions, our goal is to achieve a good overall performance in the sense of max-min fairness on the application-induced delay performance among all flows. Namely, in the present work, we try to minimize the worst condition flows' EPLR on the network instead of the EPLR averaged over flows.

This paper is organized as follows. Related work is first discussed in Sect. 2. Section 3 introduces the MTQ and QTL mechanisms for the proposed early packet discarding scheme. An extensive network simulation and its results

Manuscript received October 3, 2006.
Manuscript revised January 24, 2007.

[†]The author is with the National Institute of Information and Communications Technology, Kitakyushu-shi, 802-0001 Japan.
[††]The authors are with Kyushu Institute of Technology, Iizuka-shi, 820-8502 Japan.
a) E-mail: kuma@kyushu.jgn2.jp

DOI: 10.1093/ietcom/e90–b.9.2481

are described in Sects. 4 (overview of simulation models), 5 (homogeneous environment), and 6 (heterogeneous environment). Finally, some conclusions are presented in Sect. 7.

2. Related Work

From the viewpoint of deployment, it is easier to implement controls only between end-to-end equipment in a network. However, networks can obviously provide more finely grained services when intermediate nodes work in conjunction with endpoint equipment. Various schemes using intermediate nodes have been proposed to realize better network performance.

Early packet discarding, in which some packets might be dropped even though the queuing buffer is not full, is not an entirely new concept, and was introduced in the context of Active Queue Management (AQM) [3]. For example, in Random Early Detection (RED) [4] and its variants, in order to mitigate instability and unfairness in the throughput of TCP flows traversing it, an intermediate node probabilistically discards incoming packets based on its queue length by introducing random instead of burst packet losses. In contrast, our scheme aims to reduce the delay in real-time application flows. Although AQM has the advantage of possibly reducing queuing delays [3], to the best of our knowledge no schemes have focused on methods to reduce the number of worthless packets in real-time flows which needlessly consume network resources. In recent research, a packet discarding technique was introduced to achieve fairness between wired and wireless TCP sessions [5], and another deterministic AQM was proposed to improve TCP throughput over 3G wireless links [6].

Within the context of TCP over ATM (where one large packet in the upper-layer is split into many small cells in the lower-layer), if one cell is dropped then all remaining cells belonging to the same packet are useless and damaging to the upper-layer performance even though they have not yet been dropped. Early Packet Discard (EPD) [7] was proposed to improve the performance of TCP with respect to this fragmentation problem. Our scheme is similar to EPD in terms of the basic idea of discarding packets in advance if they are likely to be useless to the overall performance of the application.

To reduce delays in real-time flows, a variety of packet scheduling policies, such as Earliest Deadline First (EDF) [8], have been developed along with various resource reservation schemes to optimally reorder the sequence of packets belonging to various flows. Our scheme, on the other hand, does not concern such scheduling policies, but can be combined with them. Note that while complex packet scheduling schemes are not generally scalable, our scheme is so lightweight that it can even be applied to heavily loaded nodes, for example core routers. The expected queuing delay of a packet in an intermediate node using MTQ and QTL mechanisms can easily be obtained from the queue length when the packet arrives there, which can readily be managed at the node. The delay is then simply compared with an MTQ/QTL value in the packet header or subtracted from the QTL value there. In QTL, which is similar to the TTL mechanism, the cost of updating the QTL value by subtraction is negligible.

Another way to reduce delays in real-time flows is by overprovisioning, where a network has sufficient bandwidth to ensure that all flows are within their delay limits [9]. However, whether this could be an appropriate solution or not depends on time and circumstances, because traffic demands change faster than network provisioning in many situations. A scheme to fully and efficiently utilize the existing network resources is still required, and both the above approaches will mutually complement each other.

3. Adaptive Early Packet Discarding at Intermediate Nodes

The MTQ and QTL mechanisms were originally proposed in a lightweight practical framework for active networks [10], and their effectiveness in time synchronizing over a network was shown [11]. However in this paper, we apply these mechanisms to more general delay-sensitive applications.

Our target model, in which real-time application flows compete for resources in a single domain network, is outlined in Fig. 1. The MTQ and/or QTL parameters are set in the header of each packet entering the domain at the edge nodes, and those packets are forwarded to intermediate nodes. Every time a packet arrives at the intermediate nodes, it is queued or discarded according to the MTQ and/or QTL mechanisms described below and in Fig. 2.

Packets using the MTQ mechanism are managed based on the local queuing delay limit at each intermediate node. The processing procedure for each packet at edge and intermediate nodes is as follows:

1. A value for MTQ (i.e. the maximum queuing delay in one node) is set in the header of each packet at an edge node.
2. When a packet arrives at an intermediate node and if the queuing buffer bound for the next node to which the packet will be forwarded is not full, (1) the local queuing delay for the packet is calculated based on the queue length and output link bandwidth, and (2) if the value for MTQ in the packet header is larger than the

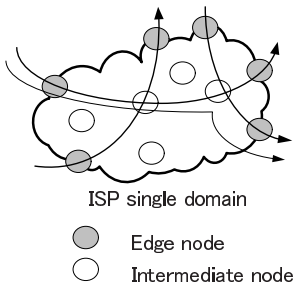


Fig. 1 Target network model.

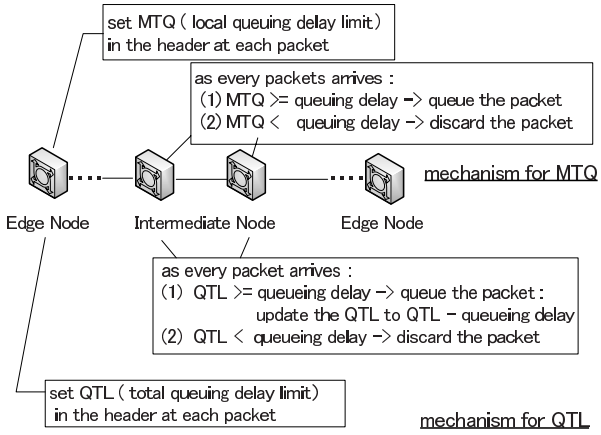


Fig. 2 MTQ and QTL mechanisms.

calculated queuing delay, the packet will be queued. On the other hand, (3) if the calculated queuing delay is larger than the MTQ value, the packet is discarded.

Because the MTQ mechanism limits the local queuing delay at each intermediate node, setting the MTQ parameter is equivalent to limiting the size of the queuing buffer at every node through which the packet passes.

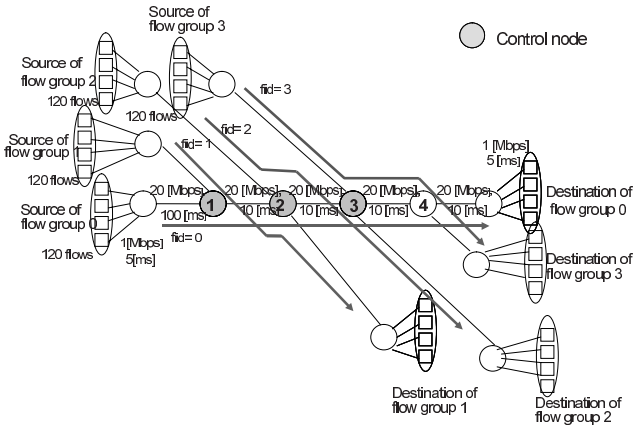
Packets using the QTL mechanism, on the other hand, are managed based on the global (total) queuing delay limit throughout the network (from an ingress edge node to an egress edge node):

1. A value for QTL (i.e. the maximum queuing delay in the network) is set in the header of each packet at an edge node.
2. When a packet arrives at an intermediate node and if the queuing buffer bound for the next node to which the packet will be forwarded is not full, (1) the local queuing delay is calculated based on the queue length and output link bandwidth. (2) If the value for QTL in the packet header is larger than the calculated queuing delay, the value for QTL in the packet header is updated to be "QTL minus the calculated queuing delay" and the packet will be queued. On the other hand, (3) if the calculated queuing delay is larger than the QTL value, the packet is discarded.

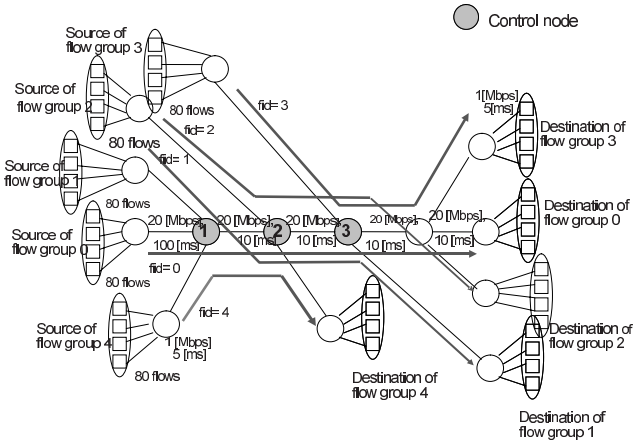
Because the packet is discarded if the updated QTL value is not positive, the initial value of the QTL parameter corresponds to the global queuing delay limit.

4. Simulation Models

We performed the simulation using ns release 2.27 [12], with MTQ and QTL mechanisms additionally implemented to manage the queues. The simulation models we used are outlined in Fig. 3. We focused on two network configurations, models 1 and 2. In both models, the length of each queuing buffer is equal to 200 packets for each output link with bandwidth of 20 [Mbps]. In both models, a number of real-time application flows on the network are grouped such



(a) model 1.



(b) model 2.

Fig. 3 Simulation model.

that flows in the same group have the same source and destination along the same path, and thus will experience an identical delay on average. These flows meet queuing delays only at nodes 1, 2, and 3 (the gray-colored nodes in the figure) due to flow competition.

Table 1 lists the number of flows for each group and the total transmission rate of the flows in the group. In this configuration, flow group 0 is most likely to compete with other flow groups (at three nodes) and to suffer the largest queuing delay. Only two flow groups (group 0 and one other) will compete with each other at a node in model 1, while three flow groups will do so in model 2. We targeted one traffic configuration in model 1, where the average utilization is 96%, while two traffic configurations are considered in model 2, where average utilization are 96% and 84%. Since the results we observed in the simulation showed similar trends for all configurations, the simulation results for model 2(average utilization is 96%) only will be shown, unless otherwise noted.

We applied an ON-OFF burst model to traffic of each real-time flow, in which an exponential distribution for its ON and OFF periods [13] was used. We examined several types of traffic, with average ON/OFF durations set to 250/650 [ms], 350/650 [ms], 450/650 [ms] and

Table 1 No. of flows in heavily congested scenario in model 1 (top) and in model 2 (middle) and in moderately congested scenario in model 2 (bottom).

(a) model 1, average link utilization: 96%

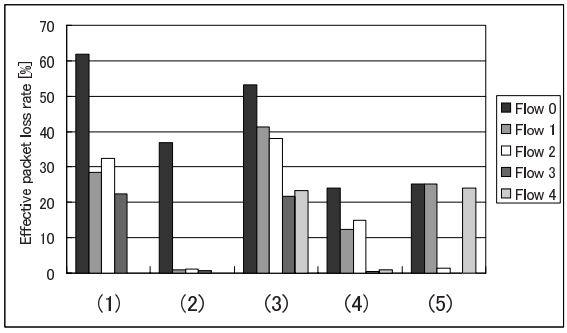
Congested link	Flow id(fid)				No. of flows	Total rate [Mbps]
	0	1	2	3		
$1 \rightarrow 2$	120	120			240	19.2
$2 \rightarrow 3$	120		120		240	19.2
$3 \rightarrow 4$	120			120	240	19.2

(b) model 2, average link utilization: 96%

Congested link	Flow id(fid)					No. of flows	Total rate [Mbps]
	0	1	2	3	4		
$1 \rightarrow 2$	80	80			80	240	19.2
$2 \rightarrow 3$	80	80	80			240	19.2
$3 \rightarrow 4$	80		80	80		240	19.2

(c) model 2, average link utilization: 84%

Congested link	Flow id(fid)					No. of flows	Total rate [Mbps]
	0	1	2	3	4		
$1 \rightarrow 2$	70	70			70	210	16.8
$2 \rightarrow 3$	70	70	70			210	16.8
$3 \rightarrow 4$	70		70	70		210	16.8



Parameter list of Fig. 4

Cases	Models	acceptable queuing delays [ms]
(1)	model 1	10
(2)	model 1	20
(3)	model 2	10
(4)	model 2	20
(5)	model 2 in Sect. 6.1	10

Fig. 4 *EPLR* (effective packet loss rate), using neither MTQ nor QTL.

3500/6500 [ms]. In each ON period, fixed sized packets flow at a constant rate. The average rate (over ON and OFF periods) of each flow was fixed at 80 [kbps], and the packet length was 200 [byte] (e.g. G.711 VoIP codec). Since we observed similar characteristics for the results for all traffic types, the results for only one type (that with a reasonable burstiness, i.e. average ON/OFF duration of 350/650 [ms]) will be shown due to space limitations.

In our simulation models, the maximum queuing delay at each node is 16 [ms], because at most 200 packets each of length 200 [byte] are waiting to be sent to a link with a bandwidth of 20 [Mbps]. A packet in flow group 0 competes with traffic in the other flow groups at three nodes along the path, and thus the total queuing delay can be as much as 48 [ms]. Hence, we examined three cases of acceptable total queuing delay: 10 (strict), 20 (moderate), and 30 (loose) [ms], which could correspond to three different application delay constraints or be derived from three different network environments that have different fixed delays. Since we observed similar characteristics for the results for the two cases with acceptable delay of 20 [ms] and 30 [ms], only the results for cases with 10 [ms] and 20 [ms] will be shown due to space limitations.

As mentioned in Sect. 1, we use Effective Packet Loss Rate (EPLR) as a metric for evaluating performance with respect to delay in real-time applications. This is the ratio of the number of packets discarded at nodes or invalidated by the receiver (due to a larger delay than the acceptable total queuing delay), to the total number of packets sent by the sender.

5. Simulation Results in Homogeneous Environments

We evaluated our mechanisms via network simulations in homogeneously congested environments in which all flows had an identical delay requirement (i.e. an acceptable total queuing delay).

Each flow traversed one or more nodes in which it competed with other flows, where each node had an output queuing buffer corresponding to a maximum local queuing delay of 16 [ms]. Therefore, if neither MTQ nor QTL were used, very high EPLRs could be seen even for a moderate acceptable total queuing delay of 20 [ms], as illustrated in Fig. 4. In particular, flows in flow group 0 experienced very high EPLRs because they competed with other flows either in the same flow group or in different ones at three nodes. In order to reduce the high EPLRs, in Sects. 5.1, 5.2, and 5.3, schemes with MTQ only, QTL only, and QTL plus MTQ are used.

Note that an MTQ value of more than 16 [ms] is meaningless, because that is the maximum queuing delay at each node. Meanwhile, a QTL value of more than 48 [ms] can also be considered meaningless, because this is higher than the total queuing delay in traversing at most three congested nodes along a path.

5.1 Effectiveness of MTQ Mechanism

Setting MTQ for a packet is equivalent to limiting the length of the queuing buffer at every node the packet traverses. In general, as the length of the queuing buffer at network nodes decreases, the queuing delays experienced by packets decrease. The number of packets invalidated due to the acceptable delay limit at the receiver also decreases, while the packet losses due to buffer overflow at network nodes increase. This gives rise to a trade-off for lowering EPLR. To clarify the relationship between EPLR and MTQ, we first show the packet losses occurring at intermediate nodes in the network, and then investigate EPLR.

Figure 5 represents the network packet loss rate of each flow group when applying the MTQ mechanism for a variety of different MTQ values in the widest possible range. In this case, for the original queuing buffer length of 16 [ms], the packet loss rate is about 2% at worst (i.e. in flow group

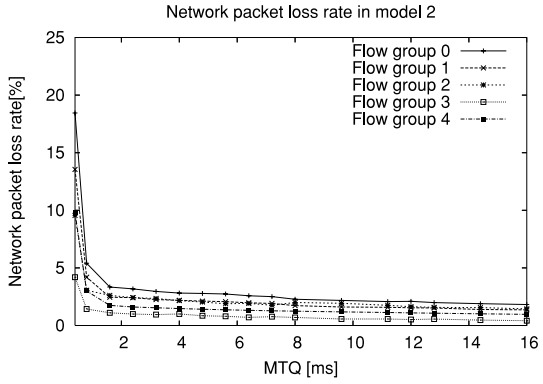
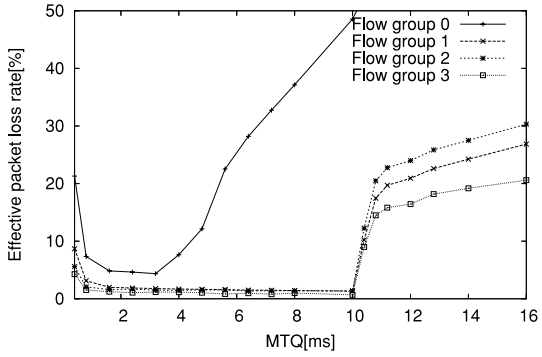


Fig. 5 Network packet loss rate, using MTQ alone.

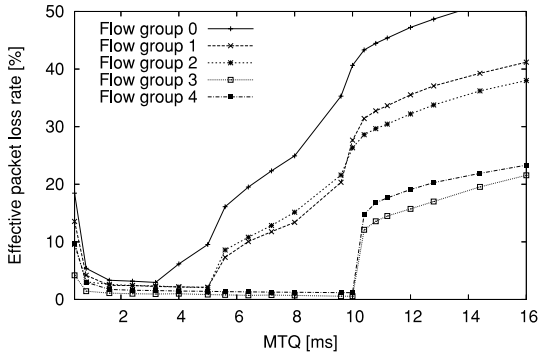
0). As the MTQ value decreases, the loss rate increases just slightly as long as the MTQ value is larger than a threshold of about 1.8 [ms]. Below this threshold however, the loss rate drastically increases due to the queuing buffer being so short that it cannot absorb a moderate burstiness in those flows.

Figures 6(a) and (b) show EPLR for a strict acceptable queuing delay of 10 [ms] with a variety of MTQ values in models 1 and 2, respectively. Figure 6(c) also shows EPLR against MTQ, for a moderate acceptable queuing delay of 20 [ms] in model 2. Note that each figure shows a very large EPLR corresponding to an MTQ of 16 [ms], and this is equivalent to the EPLR achieved without applying our scheme. This can also be seen in cases (1)–(4) of Fig. 4. This clearly indicates a general and significant reduction of EPLR by applying the MTQ mechanism.

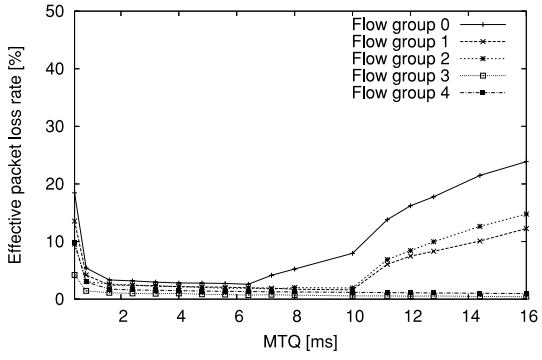
In Figs. 6(a) and (b) for an acceptable total queuing delay of 10 [ms], when the MTQ value exceeds about 3.4 [ms] the EPLR of flow group 0 suddenly increases. This indicates that a number of packets in that group suffer from large queuing delays over the 10 [ms] limit allowed by the application. That is, setting the MTQ value in a range from 1.8 to 3.3 [ms] significantly improves the EPLR of that flow group traversing three congested nodes along its path. Considering flow groups 1, 2 and 3 in the first figure, these groups which traverse one congested node show acceptably small EPLRs when MTQ is set in a wide range from 1.8 to 10 [ms], even though the reduction of MTQ within this range results in negligible deterioration (increase) of EPLR. Therefore, an MTQ value of 3.3 [ms] can be considered to be optimal for the overall performance of all flow groups for this setting. Similarly, in the second figure, while the optimal MTQ for flow groups 1 and 2 which traverse two congested nodes is about 5 [ms], the same MTQ value of 3.3 [ms] for flow group 0 still seems to be optimal. These results imply that, for a flow obeying an acceptable total queuing delay of D and traversing N equally congested links along a path, setting the MTQ to D/N (or slightly less) significantly reduces the queuing delay experienced by the flow. An MTQ value less than D/N ensures that the total queuing delay experienced by the packet traversing N nodes is less than D if the packet is not discarded at those nodes.



(a) Acceptable queuing delay: 10 [ms] in model 1.



(b) Acceptable queuing delay: 10 [ms] in model 2.



(c) Acceptable queuing delay: 20 [ms] in model 2.

Fig. 6 *EPLR* (effective packet loss rate), using MTQ alone.

Consequently, an appropriate MTQ value range could keep EPLR small for all flows. In the above scenario, Fig. 5 indicates that packet losses remain low if the MTQ is larger than a threshold (1.8 [ms]), which is smaller than D/N ($10/3 \approx 3.3$ [ms]), where N is the largest number of congested links through which the worst case flows traverse. Therefore, for every flow, setting MTQ to 3.3 [ms] means that no packets experience a total queuing delay larger than 10 [ms], and only a small number of packets are discarded at intermediate nodes. Figure 6(c) shows EPLR for an acceptable total queuing delay of 20 [ms], and also supports the above findings. Compared with the previous figures, the delay performance of flow groups 3 and 4 seems not to be improved by any MTQ value. The reason might be that since these flow groups traverse only one congested node with an acceptable queuing delay of 20 [ms], the original queuing

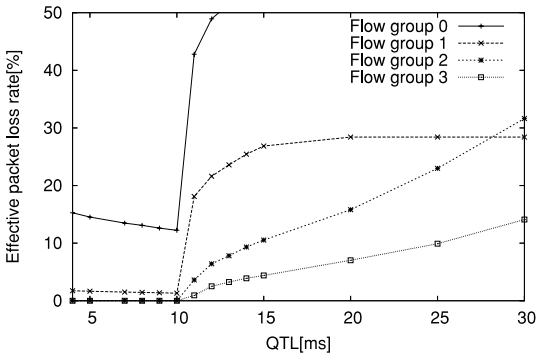
buffer length of 16 [ms] is short enough to avoid significant delays. Note that, obviously the above simple D/N-rule is not always suitable. For example, the number N of the congested nodes may not be known or N may be such large that D/N becomes shorter than a limit queuing buffer length necessary to avoid unacceptable packet losses.

In congested networks, a packet experiencing a long queuing delay at a node is likely to ultimately exceed the acceptable total queuing delay limit due to additional queuing delays at following nodes, even if the delay experienced at that node does not exceed its limit. Setting MTQ to an appropriate value might aggressively discard in advance those packets that will probably exceed the limit before reaching their destinations, and thus improve the overall delay performance. However, an appropriate value of MTQ, derived from a trade-off between network packet losses and queuing delay, is sometimes difficult to determine in reality. The lower boundary of the effective value range of MTQ depends directly on packet losses at a node, which are related to link utilization and incoming traffic characteristics (e.g. burstiness) at that node. The upper boundary is strongly dependant on how long a queuing delay will be experienced at each of the subsequent nodes, which is related to the flow competition at those nodes. Both are not always easy to estimate, and in some cases are difficult. Note that in this simple version of MTQ, the MTQ value in a packet traversing network nodes does not change node by node. This simple version might therefore be ineffective when, for example, a number of heterogeneously congested nodes are on a path. Thus, using only an MTQ mechanism in a network may not always be effective. We will examine the use of QTL in the next subsection.

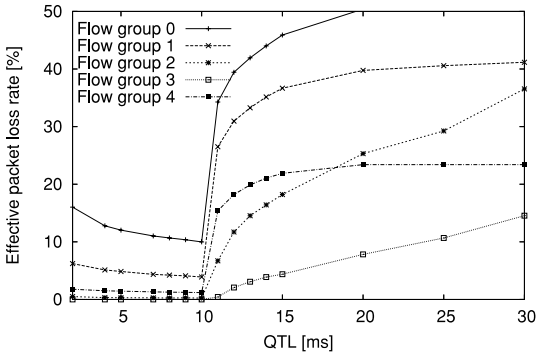
5.2 Effectiveness of QTL Mechanism

The EPLRs when only the QTL mechanism is used are shown in Figs. 7(a), (b) and (c) for a variety of QTL values.

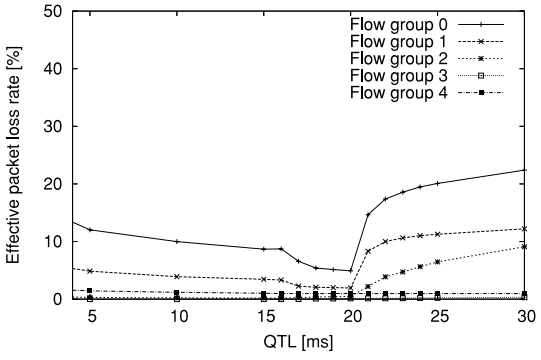
These figures clearly show that the EPLR is reduced most for every flow when QTL is set to exactly the acceptable total queuing delay limit; if QTL exceeds this optimal limit, the delay performance deteriorates greatly. QTL can be set to a smaller value in order to aggressively discard in advance those packets likely to exceed the limit before reaching their destinations. However, the figures show that setting QTL to a value smaller than the optimal limit is detrimental to some degree, unlike for MTQ. Note that Fig. 7(a) for model 1 shows an interesting phenomenon due to the nature of QTL. For flow groups 1, 2, and 3 traversing one congested node only, setting QTL is equivalent to setting MTQ, and thus a QTL value larger than the original queuing buffer length of 16 [ms] would be expected to have no effect. However, this is not the case for flow groups 2 and 3 because if the QTL value becomes larger, the larger volume of traffic in flow group 0 will survive when it competes with groups 2 and 3 at nodes 2 and 3 respectively. This implies that an appropriate QTL setting will become more important as the number of nodes along a path increases.



(a) Acceptable queuing delay: 10 [ms] in model 1.



(b) Acceptable queuing delay: 10 [ms] in model 2.



(c) Acceptable queuing delay: 20 [ms] in model 2.

Fig. 7 *EPLR* (effective packet loss rate), using QTL alone.

A QTL value exactly equal to the acceptable total queuing delay results in the conservative discarding of packets which have already exceeded the total delay limit. Other simulation results (obtained in our simulation but not shown in this paper) also support this observation that such a QTL setting is always optimal in terms of reducing EPLR regardless of the simulation model, optimal values, or degree of traffic burstiness.

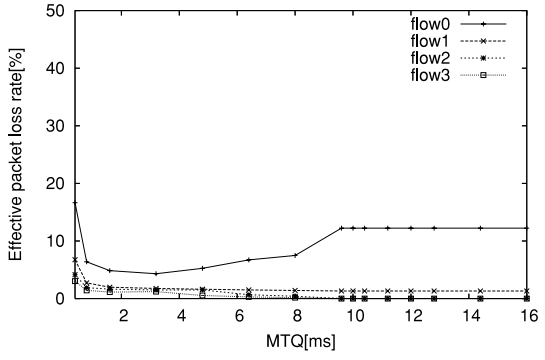
This simple rule for setting the QTL value is of practical importance from an operational standpoint. However, the EPLR reduction achieved by using an optimal QTL alone seems inferior compared to using optimal MTQ alone. Therefore, in the next subsection we will examine the usage of QTL and MTQ together, in order to exploit the combination of conservative discarding via QTL and aggressive discarding through MTQ.

5.3 Effectiveness of Setting MTQ and QTL Simultaneously

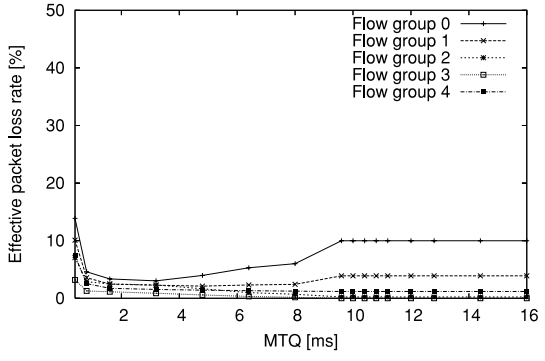
Figures 8(a) and (b) show EPLR for an acceptable total queuing delay of 10 [ms] when QTL and MTQ are used together. QTL is set to 10 [ms], equal to the queuing delay requirement. MTQ is set within a range from 0 to 16 [ms], although an MTQ value greater than the QTL value (10 [ms]) is meaningless in this combination (i.e. it is equivalent to setting QTL alone).

The benefits of using MTQ together with QTL (compared to QTL alone) are clearly shown by the fact that reducing MTQ from 10 [ms] causes a reduction in EPLR for flow group 0, which is the worst flow group in terms of EPLR.

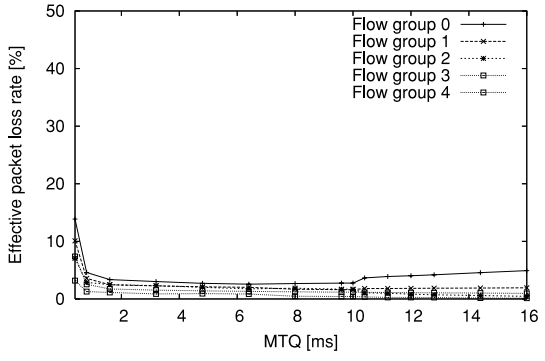
Compared with Figs. 6(a) and (b), the advantage of using QTL with MTQ (compared to MTQ alone) is that a wider range of MTQ values (at least those less than the QTL value) can result in a considerable reduction of EPLR. On the other hand, when the acceptable total queuing delay is 20 [ms] (Fig. 8(c)), the effect of setting MTQ is not clear because the queuing buffer (equivalent to 16 [ms]) is already shorter than the acceptable total queuing delay. Note that setting QTL to a value larger than $n \times MTQ$, where n is the number of intermediate nodes along the path, is meaningless when setting MTQ and QTL simultaneously. Because the maximum waiting time for each packet per node is MTQ, the total queuing delay can not be longer than $n \times MTQ$. Thus, if MTQ and QTL will be used simultaneously to exploit a combined effect, they should be set within the range: $MTQ < QTL < n \times MTQ$ where n is the number of intermediate nodes.



(a) Acceptable queuing delay: 10 [ms] in model 1, QTL=10 [ms].



(b) Acceptable queuing delay: 10 [ms] in model 2, QTL=10 [ms].



(c) Acceptable queuing delay: 20 [ms] in model 2, QTL=20 [ms].

Fig. 8 *EPLR* (effective packet loss rate), using QTL plus MTQ.

Before ending this subsection, we present in-depth analysis of the performance of the proposed scheme, in using four parameter configurations, (a) no parameters set, (b) MTQ = 5 [ms], (c) QTL = 10 [ms] and (d) MTQ = 5 [ms], QTL = 10 [ms] in model 2, where total acceptable queuing delay is 10 [ms].

Figure 9 presents the EPLR of each flow, showing its proportion of (1): the packets discarded at nodes due to buffer overflow, MTQ, or QTL, and (2): the packets discarded by the application. By adopting our scheme, the number of packets discarded by the application is drastically reduced, while the number of packet discarded at nodes is slightly increased.

Table 2 presents the number of packets discarded at each nodes due to the buffer overflow or MTQ(1*), due to the QTL(2*), the sum of packets discarded at all nodes(3*)

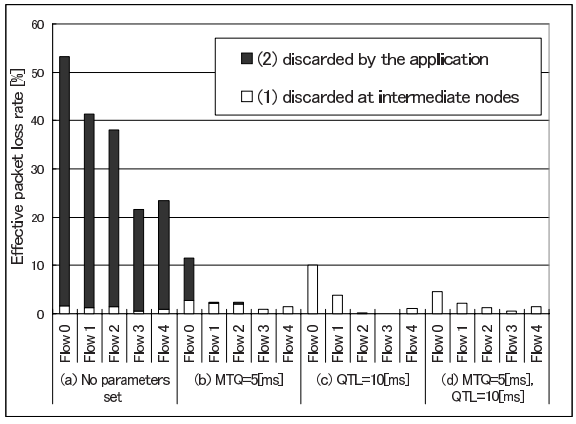


Fig. 9 Proportion of EPLR cased by application-side and network-side.

Table 2 No. of packet losses at each node.

	node1	node2	node3	3*	4*
	1*/2*	1*/2*	1*/2*		
(a)	10970/-	7527/-	5440/-	23937	687194
(b)	16011/-	11460/-	8731/-	36202	38105
(c)	-/13116	-/22520	-/24527	60163	0
(d)	16011/0	11460/0	4671/5645	37787	0

Table 3 Average queuing delay at each node.

	node 1 [ms]	node 2 [ms]	node 3 [ms]
(a)	4.24	3.92	3.76
(b)	1.52	1.28	1.20
(c)	2.64	1.36	0.88
(d)	1.52	1.28	0.96

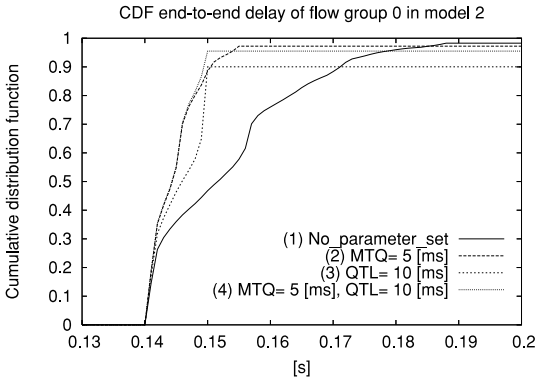


Fig. 10 Cumulative distribution function for end-to-end-delay in flow group 0.

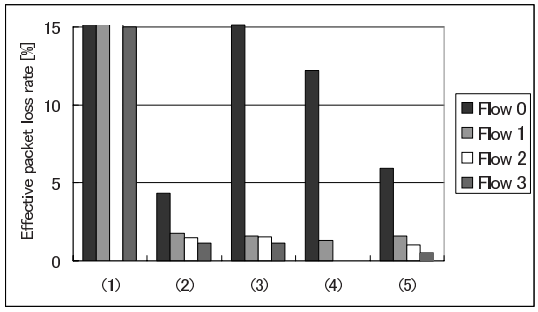
and the sum of packets discarded at the receiver-side application due to violating its delay limitation(4*). In case (a) where the no parameters set, the number of packets discarded at each node(3*) is small, while the number of packets discarded at the application(4*) becomes large. In case (b), the number of discarded packet decreases in descendant (downstream) nodes, while a considerable number of packets are discarded by the application (4*). On the other hand, in case (c), the number of discarded packets increases especially at descendant nodes, while no packets are discarded by the application. Finally, in case (d) where adopting MTQ and QTL simultaneously, the number of discarded packets at each node is relatively balanced and the total number of discarded packet is minimized.

In addition, average queuing delay at each node is presented in Table 3. We found that MTQ, in case (b), seems to evenly reduce the average queuing delay at all nodes, while QTL, in case (c), improves that at descendant nodes. Thus, the degree of improvement in the delay characteristics will be high by adopting both parameters simultaneously.

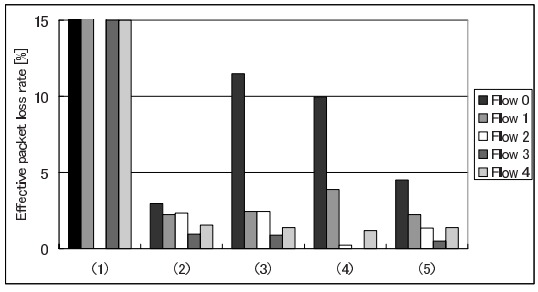
For each setting, the distribution of the end-to-end delay (i.e. the total delay along the path) for packets in flow group 0 is plotted in Fig. 10. By setting QTL to 10 [ms], the end-to-end delay would be limited to 150 [ms] (the fixed delay of 140 [ms] plus the maximum queuing delay of 10 [ms]). By applying our scheme, the proportion of packets experiencing shorter delays would increase.

5.4 Summary in Homogeneous Environments

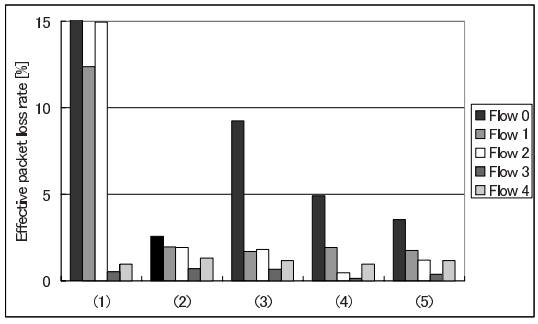
In Fig. 11, we will summarize the effectiveness of using MTQ alone, QTL alone, and a combination of both. Figure 11(a) shows the EPLR for each flow group in model 1 when the acceptable queuing delay is 10 [ms], while



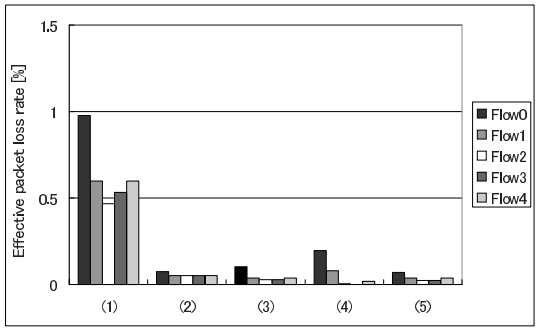
(a) Acceptable queuing delay: 10 [ms] in model 1, average link utilization: 96%.



(b) Acceptable queuing delay: 10 [ms] in model 2, average link utilization: 96%.



(c) Acceptable queuing delay: 20 [ms] in model 2, average link utilization: 96%.



(d) Acceptable queuing delay: 10 [ms] in model 2, average link utilization: 84%.

Parameter list of Fig. 11

	Parameters [ms]	
Case	(a), (b), (d)	(c)
(1)	No parameters set	
(2)	MTQ : 3.3	MTQ : 6.6
(3)	MTQ : 5	MTQ : 10
(4)	QTL : 10	QTL : 20
(5)	MTQ : 5, QTL : 10	MTQ : 10, QTL : 20

Fig. 11 Comparison of *EPLR* (effective packet loss rate) in using none, MTQ alone, QTL alone, and MTQ plus QTL.

Figs. 11(b) and (c) show cases where the acceptable total queuing delays are 10 and 20 [ms] in model 2 in congested networks. Obviously, in all of these scenarios, the EPLR is reduced drastically by setting MTQ and/or QTL (cases (2) through (5) in the figures) compared with case (1) where neither MTQ nor QTL is used.

Since the optimal MTQ is not always easily found, case (2) is not always realistic even though they exhibit the best performance. The case with QTL only (case (4)) can be improved by adding MTQ (case (5)), while the case with MTQ only (case (3)) can also be improved by adding QTL(case (5)), in terms of reducing the EPLR of the worst flow group 0 and balancing EPLRs of all flow groups. In other words, the worst flows can be improved at the expense of performance deterioration in the other good flows. It can be concluded that combining QTL and MTQ leads to a relatively good overall delay performance when QTL is set exactly to the acceptable total queuing delay limit and MTQ is set to some non-optimal value smaller than the limit, e.g. 50% of the limit.

Furthermore, we examine other scenarios in which the network is not so congested. We can find that proposed scheme is effective to improve the delay characteristics of real time flows also in those scenarios. Figure 11(d) shows the EPLR for each flow group when the traffic load is relatively light as shown in Table 1(c). We also examine less congested two scenarios where the number of flows in each flow group is 40 (average link utilization is 48%) or 60 (that is 72%). In both scenarios, EPLR becomes zero for each flow group regardless of MTQ and/or QTL, i.e., in all five cases. Thus, our proposed scheme is not harmful (even useless) in lightly congested networks.

6. Simulation Results in Heterogeneous Environments

We examine our mechanisms in three heterogeneous environments using model 2. Section 6.1 deals with heterogeneity in terms of traffic volume over different paths/links, where the number of flows on each different path is not the same. Sections 6.2 and 6.3 examine configurations with heterogeneity with respect to the acceptable queuing delay for different flows. In Sect. 6.2, flows on the same path have different acceptable queuing delay requirements, while in Sect. 6.3 flows on different paths have different acceptable queuing delays. To ensure the marginal quality of every flow, we try to find an appropriate setting of MTQ/QTL so as to improve worst flow EPLR even if there will be different types or conditions of flows.

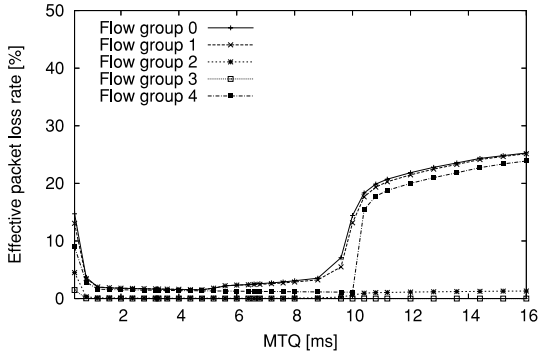
6.1 Different Number of Flows on Different Paths

We consider a scenario such that the number of flows for each flow group is different as listed in Table 4. The number of competing flows at each link (240, 220, and 200), and thus the utilization also, decreases link by link from the flow source to the sink. Case (5) in Fig. 4 shows the EPLR observed for each flow group for an acceptable queuing delay of 10 [ms] when neither MTQ nor QTL is used. This indicates that only node 1 (link 1 → 2) is so congested that only the flow groups which compete with one another at that node (i.e. groups 0, 1, and 4) suffer from an equally large EPLR.

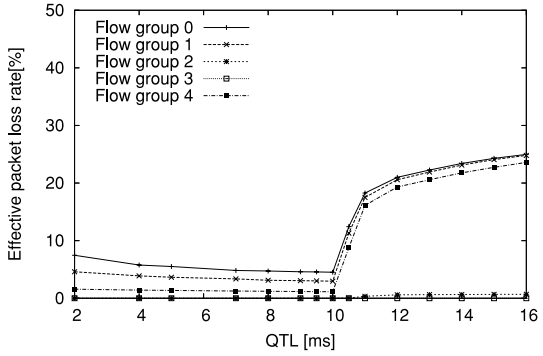
Figure 12 shows the relationship between the parameter settings of MTQ/QTL and the EPLR of each flow group for an acceptable queuing delay of 10 [ms]. Figures 12(a), (b), and (c) show results for scenarios using MTQ alone, QTL alone, and QTL (set to 10 [ms]) plus MTQ, respec-

Table 4 No. of flows in model 2.

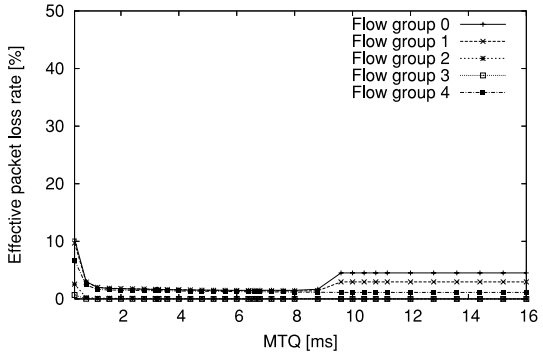
Congested link	Flow id(fid)					No. of flows	Total rate [Mbps]
	0	1	2	3	4		
1 → 2	60	80			100	240	19.2
2 → 3	60	80	80			220	17.6
3 → 4	60		80	60		200	16.0



(a) Adopting MTQ alone.



(b) Adopting QTL alone.



(c) Adopting QTL (=10 [ms]) plus MTQ.

Fig. 12 *EPLR* (effective packet loss rate) when the number of flows in each group is different, acceptable queuing delay: 10 [ms].

tively.

In Fig. 12(a), the range of suitable values for MTQ is different to that observed in Fig. 6(b) for the homogeneous case. This can be explained as follows. Consider a packet in flow group 0 at the first node, which experiences a queuing delay larger than the upper bound (3.3 [ms]) of the appropriate MTQ value range for the homogeneous case, but much smaller than the acceptable queuing delay limit (10 [ms]). Even with this delay, additional delays at the following nodes 2 and 3 are unlikely to make the packet ultimately exceed the delay limit, because these nodes are less congested than the first node 1. Similar arguments can be applied to the EPLRs for flow groups 1 and 2. This result implies that MTQ values should be carefully set depending on various network conditions.

The optimal value for QTL seen in Fig. 12(b), on the other hand, is 10 [ms], which is identical to the optimal value in Fig. 7(b). From Fig. 12(c) we can further conclude that using QTL together with MTQ can effectively reduce the EPLR for a wide range of MTQ values, even in this heterogeneous scenario.

6.2 Different Delay Requirements of Flows on the Same Path

Consider a scenario such that the flows in each group in Fig. 3(b) are divided into two subgroups, referred to as subgroups 1 and 2, each of which has 40 flows. As listed in Table 5, the flows in subgroup 1 have a stricter delay requirement than those in subgroup 2.

Table 5 Acceptable queuing delay limit for each subgroup flow.

Flow group	Subgroup	Indication	Acceptable queuing delay [ms]
0	1	f0-1	10
	2	f0-2	20
1	1	f1-1	10
	2	f1-2	20
2	1	f2-1	10
	2	f2-2	20
3	1	f3-1	10
	2	f3-2	20
4	1	f4-1	10
	2	f4-2	20

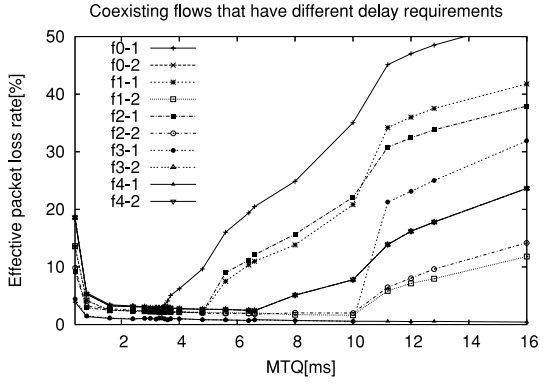
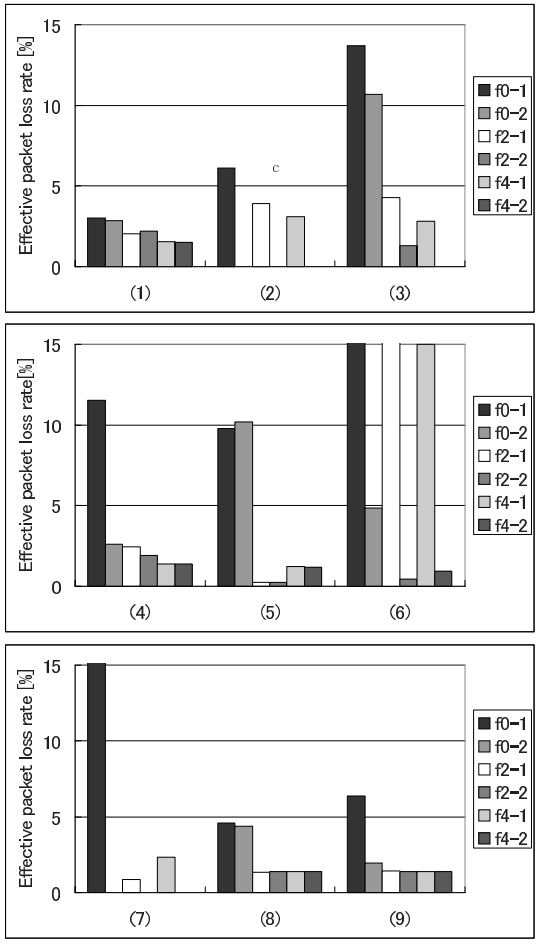


Fig. 13 *EPLR* (effective packet loss rate), using MTQ alone.

Figures 14 show the EPLR for each subgroup. Note that when our scheme was not used, very high EPLRs were observed in this congested network, e.g. over 50% for subgroup 1 in flow group 0, over 40% for subgroup 1 in flow group 1, which can be seen the EPLRs at MTQ value of 16 [ms] in Fig. 13

In Fig. 14, cases (1)–(4) present the EPLR results us-



Parameter list of Fig. 14

Cases	parameters [ms]
(1)	f0-1–f4-2 : MTQ=3.3
(2)	f0-1,f1-1,f2-1,f3-1,f4-1 : MTQ=3.3
	f0-2,f1-2,f2-2,f3-2,f4-2 : MTQ=6.6
(3)	f0-1,f1-1,f2-1,f3-1,f4-1 : MTQ=5
	f0-2,f1-2,f2-2,f3-2,f4-2 : MTQ=10
(4)	f0-1–f4-2 : MTQ=5.0
(5)	f0-1–f4-2 : QTL=10
(6)	f0-1–f4-2 : QTL=20
(7)	f0-1,f1-1,f2-1,f3-1,f4-1 : QTL=10
	f0-2,f1-2,f2-2,f3-2,f4-2 : QTL=20
(8)	f0-1–f4-2 : MTQ=5, QTL=10
(9)	f0-1,f1-1,f2-1,f3-1,f4-1 : MTQ=5, QTL=10
	f0-2,f1-2,f2-2,f3-2,f4-2 : MTQ=5, QTL=20

Fig. 14 *EPLR* (effective loss rate) when coexisting flows on the same path have different queuing delay requirements.

ing MTQ alone, where several tactics for setting MTQ are examined. First we investigated the case where all flows have an identical MTQ value, with results shown in Fig. 13. We found an optimal value for MTQ of 3.3 [ms], which is the same as in the homogeneous case illustrated in Fig. 6(b). The EPLR for each flow using this MTQ setting is shown in case (1) of Fig. 14, which performs the best of all.

In case (2), the MTQ is set differently in each subgroup such that an optimal value for each individual flow in the homogeneous case is applied. Thus, the MTQ setting is 3.3 [ms] for the flows having a strict delay limit (in subgroup 1), and 6.6 [ms] for the non-strict flows (in subgroup 2). It can be clearly seen that the strict flows suffer greatly in competing with the non-strict flows, because whenever the queue length increases to more than 3.3 [ms], all packets in the strict flows will be discarded while those in the non-strict flows will not. Compared with case (1), EPLR of subgroup 2 is improved, at the expense of degrading subgroup 1's EPLR.

In case (4), the MTQ is set to be the same in all flows as some non-optimal but moderate (i.e. conservative) value, say 50% of the acceptable queuing delay for the strict flows in subgroup 1. Similarly to case (2), case (3) set the MTQ differently to each subgroup where a MTQ value larger than the optimal one for each individual flow is applied. These results indicate that when MTQ alone is applied, to achieve an overall good EPLR all flows should have the same MTQ value set based on their strictest queuing delay requirement, that is the value optimal to the strictest flows in the homogeneous case.

Cases (5)–(7) of Fig. 14 display the results for when only QTL is applied. In case (5), the QTL is set to be the same in all flows as the optimal value for the strict flows, that is the acceptable queuing delay of 10 [ms] for flow subgroup 1. In case (6) however, the QTL is set for all flows to be the same as the optimal value for the non-strict flows (subgroup 2). In case (7), the QTL is set differently for each subgroup such that the subgroup's acceptable queuing delay is applied, that is 10 [ms] for the strict subgroup 1 flows, and 20 [ms] for the non-strict subgroup 2 flows. As in case (2) for MTQ, the strict flow performance is degraded through competition with the non-strict flows. These results show that when QTL only is used, all flows should be set to the same QTL value based on their strictest queuing delay requirement.

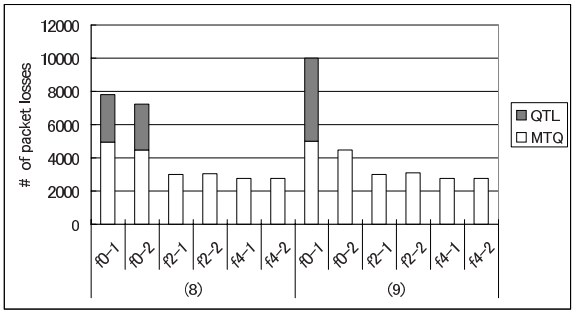


Fig. 15 Proportion of network-based packet losses by QTL and MTQ.

Cases (8)–(9) of Fig. 14 show the EPLR where both MTQ and QTL parameters are set. In these cases, we assumed that the optimal value for MTQ was unknown, so instead an MTQ value of 50% of the strictest acceptable queuing delay requirement was used, i.e. 5.0 [ms]. The QTL is set to be the same in all flows based on the delay requirement for subgroup 1 in case (8). In case (9) however, the QTL is set differently in each subgroup so that the subgroup's own delay requirement is applied.

We investigate the main cause of packet discarding on each flow group at intermediate nodes in cases (8) and (9). As illustrated in Fig. 15, the QTL-based packet loss (i.e. more than 10 [ms] cumulative queuing delay experienced by a packet) occurred only on flow group 0 while the MTQ-based packet loss (i.e. more than 5 [ms] local queuing delay experienced by a packet) appeared on every flow group. This is because only flow group 0 competes with other flow groups at more than two nodes along its path where the maximum queuing delay at each node could not exceed 5 [ms].

Comparing results in Fig. 14 seems to indicate that case (8) shows the best performance in terms of ensuring the marginal quality of every flow and that even when both QTL and MTQ are used, all flows should have the same QTL value set equal to the strictest acceptable queuing delay, i.e. 10 [ms] in this case.

6.3 Different Delay Requirements of Flows on Different Paths

For another type of heterogeneity, we consider a scenario in which each flow group traversing each different path has a different delay requirement, as listed in Table 6. Here, flow groups 0, 3, and 4 have a strict acceptable queuing delay limit, while flow groups 1 and 2 obey a non-strict delay limit. First we investigated the situation in which all flows have an identical MTQ value, as shown in Fig. 16. We found that in terms of averaged EPLR over all flows, the optimal value for MTQ is 3.3 [ms], which is again the same value as observed in Figs. 6 and 13.

Figure 17 shows the EPLR in each flow group for four cases: (1) using neither MTQ nor QTL, (2) MTQ only (optimal value), (3) QTL only (optimal value), and (4) MTQ (moderate value) plus QTL (optimal value). These results are consistent with those obtained in the previous scenarios, and it can be seen that (at least in heavily congested networks),

1. MTQ/QTL can significantly improve the EPLR of all flows;

Table 6 Acceptable queuing delay limit for all subgroup flows.

Indication	Flow group no.	Acceptable queuing delay [ms]
f0	0	10
f1	1	20
f2	2	20
f3	3	10
f4	4	10

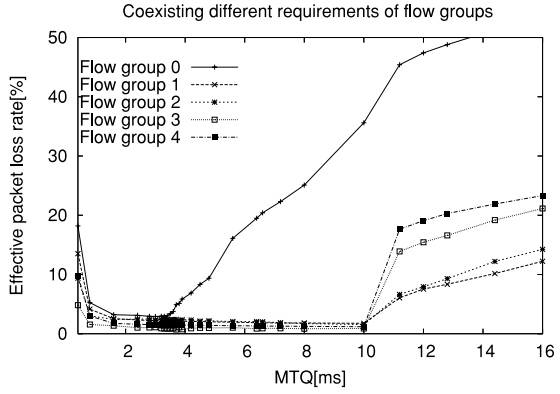
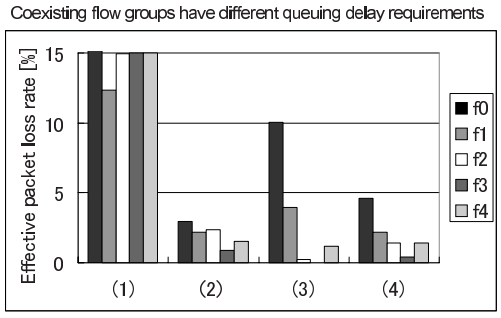


Fig. 16 *EPLR* (effective packet loss rate), using MTQ alone.



Parameter list of Fig. 17

No.	Parameters [ms]
(1)	Set no parameters
(2)	f0–f4 : MTQ=3.3
(3)	f0–f4 : QTL=10
(4)	f0–f4 : MTQ=5, QTL=10

Fig. 17 *EPLR* (effective packet loss rate) when coexisting flows on different paths have different queuing delay requirements.

2. setting MTQ to the optimal value achieves the best result in terms of the averaged EPLR over all flows, although the optimal value may not always be easy to determine;
3. applying QTL alone (even when set to the optimal value) is not always effective in improving the EPLR of the worst case flows (e.g. flow group 0);
4. using MTQ and QTL together can produce very good EPLR performance when the QTL value is optimal (that is, the strictest acceptable queuing delay), even if the MTQ value is non-optimal (say, 50% of the strictest acceptable queuing delay).

7. Concluding Remarks

In this paper, an adaptive scheme was proposed for earlier discarding of packets in real-time application flows by using two mechanisms (MTQ and QTL). In the scheme, a packet experiencing too great a queuing delay is discarded at intermediate nodes based on a limit for the total queuing delay the packet is experiencing along the path (QTL) and/or a limit for the local queuing delay the packet is experiencing at each node (MTQ).

The approach was evaluated through network simulations in both homogeneous and heterogeneous environments. In the homogeneous scenarios, the number of flows on each path was identical and all flows had an identical queuing delay requirement (i.e. limit). Meanwhile in the heterogeneous cases, the number of flows on each different path was not the same or the flows coexisting on the network had different delay requirements. Using an appropriate MTQ value effectively improved delay characteristics (i.e. reduced EPLR), but it was generally not easy to determine such a suitable value for MTQ. Delay characteristics were also improved by using QTL alone, with a value set to the acceptable queuing delay limit for flows with the strictest requirements; however there may still be room for improvement when the delay limit is relatively small compared with the length of the queuing buffer at intermediate nodes. By using QTL plus MTQ appropriately, i.e. setting QTL to the strictest acceptable queuing delay limit and MTQ to some value smaller than the delay limit (e.g. 50% of the delay limit), good overall delay performance was achieved.

Experimental results for the proposed scheme in basic scenarios have been shown, and these indicate that the scheme has great potential in improving the queuing delay performance of real-time flows in a congested network. However, further investigation and enhancement are necessary in order to develop practical systems based on the scheme. To clarify the proper scope and limitations of the scheme and to allow it to be used reliably, more theoretical analysis may be needed, as well as more extensive simulations or real world experiments. From such future investigations, a more qualitative relationship could be established between the delay requirement (the acceptable queuing delay), the achievable performance (the EPLR), and the network environment (e.g. flow topology, traffic burstiness, and link utilization).

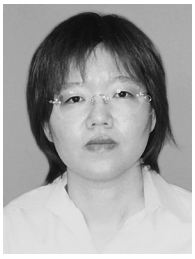
Real networks generally have a more complex meshed topology, involving a number of network nodes and with a large heterogeneity in both traffic characteristics and delay requirements. When applying the scheme to such networks, more sophisticated tuning of the MTQ parameter may be required in order to achieve a satisfactory balance in overall delay performance. To do this, the fairness of the queuing delay performance over all flows in the network should be addressed quantitatively. The MTQ mechanism presented in this paper is quite a simple version that just results in limiting the queuing buffer length equally at every node the packets traverse. Although this is shown to be effective in basic scenarios, a possible enhancement to fully exploit the potential of hop-by-hop processing for each packet is to make MTQ values more dynamic, whereby the value can be changed hop-by-hop like QTL. This could provide more flexibility to cope with greater traffic heterogeneity. This kind of enhancement, however, would also need to take cost and scalability into consideration.

This work was supported in part by the Japan Society for the Promotion of Science, Grant-in-Aid for Scientific

Research (S) (No. 18100001).

References

[1] A. Schmitter, A. Th. Schwarzbacher, and T.D. Smith, "Analysis of network conformity with voice over IP specifications," ISSC2003, pp.82–86, July 2003.
[2] K. Kumazoe, M. Tsuru, and Y. Oie, "Improving delay characteristics of real-time flows by adaptive early packet discarding," The International Conference on Information Networking, LNCS 3961, pp.463–472, Jan. 2006.
[3] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Parridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang, "Recommendation on queue management and congestion avoidance in the Internet," RFC2309, April 1998.
[4] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," IEEE/ACM Trans. Netw., vol.1, no.4, pp.397–413, Aug. 1993.
[5] Y. Matsushita, T. Matsuda, and M. Yamamoto, "A packet-discarding technique achieving fairness between wired and wireless TCP sessions," International Conference on Mobile Computing nd Ubiquitous Networking, pp.18–23, Jan. 2004.
[6] J.J. Alcaraz and F. Cerdan, "Slope based discard: A buffer management scheme for 3G links supporting TCP traffic," International Wireless Communications and Mobile Computing Conference, pp.1459–1464, July 2006.
[7] A. Romanow and S. Floyd, "Dynamics of TCP traffic over ATM networks," IEEE J. Sel. Areas Commun., vol.13, no.4, pp.633–641, May 1995.
[8] L. Georgladls, R. Guerin, and A. Parekh, "Optimal multiplexing on a single link: Delay and buffer requirements," IEEE Trans. Inf. Theory, vol.43, no.5, pp.1518–1535, Sept. 1997.
[9] C. Diot, C. Fraleigh, and F. Tobagi, "Provisioning IP backbone networks to support latency sensitive traffic," INFOCOM2003, pp.375–385, 2003.
[10] M. Tsuru, Y. Kitaguchi, H. Fukuoka, and Y. Oie, "On the practical active network with the minimal functionality," Second International Workshop on Active Network Technologies and Applications, pp.45–52, May 2003.
[11] Y. Kitaguchi, A. Machizawa, M. Tsuru, Y. Oie, and K. Hakozaki, "The advanded network time synchronous system by self-discarding packet technique," Information Processing Society of Japan, vol.46, no.4, pp.1017–1024, April 2005.
[12] The Network Simulator ns-2, http://www.isi.edu/nsnam/ns/
[13] P.M. Fiorini, "Voice over IP (voip) for enterprise networks: Performance implications and solutions," International CMG Conference, pp.545–556, 2000.



Kazumi Kumazoe received B.E., M.E. and D.E. degrees from Kyushu Institute of Technology, Iizuka, Japan in 1993, 1995 and 2007, respectively. From 1995 to 2000, she worked at NEC Corporation, Abiko. From 2000, she has been working at Japan Telecom Information Service Co., Ltd. And she has been working at Kyushu Research Center of National Institute of Information and Communications Technology from 2004. Her research interests include performance evaluation of computer communication networks and high speed networks.

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., Information Science Center, Nagasaki University, Japan Telecom Information Service Co., Ltd., and Telecommunications Advancement Organization of Japan. 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 IPSJ, JSSST and ACM.

Yuji Oie received B.E., M.E. and D.E. degrees from Kyoto University, Kyoto, Japan in 1978, 1980 and 1987, respectively. From 1980 to 1983, he worked at Nippon Denso Company Ltd., Kariya. From 1983 to 1990, he was with the Department of Electrical Engineering, Sasebo College of Technology, Sasebo. From 1990 to 1995, he was an Associate Professor in the Department of Computer Science and Electronics, Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology, Iizuka. From 1995 to 1997, he was a Professor in the Information Technology Center, Nara Institute of Science and Technology. Since April 1997, he has been a Professor in the Department of Computer Science and Electronics, Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology. His research interests include performance evaluation of computer communication networks, high speed networks, and queueing systems. He is a fellow of the IPSJ and a member of the IEEE.