Regular Paper

Fragmented Patching Technique: Super Optimal Asynchronous Multicasting for On-Demand Video Distribution

Katsuhiko Sato,[†,††] Michiaki Katsumoto,[†††] Minoru Terada[††] and Tetsuya Miki[††]

Two asynchronous multicasting techniques — patching and hierarchical merging — which optimize network bandwidth utilization are promising for on-demand video distribution. The hierarchical merging technique requires less network bandwidth than the patching technique, but requires more frequent grafting of the multicast tree. This paper proposes a more optimal form of asynchronous multicasting which we call fragmented patching, where even patch flows are sent in multicasting. This paper compares the required bandwidth and operational complexity on the network with the three techniques. It is the first to present mathematical models for the average usage of link bandwidth with the traffic intensity (Erlang) and the average frequency of tree grafting for multicasting. The mathematical models show that fragmented patching lessens the traffic intensity by 55% (trunk link) and by 75% and 76% (branch links which are branched to four and eight links, respectively), compared with that of hierarchical merging when the request rate is 100 and the content length is 2. However, the technique allows the rate of multicast tree grafting to rise. We allow broadcasting of the patch flows, thus, the rate of multicast tree grafting with fragmented patching remains the same as that with the patching technique. In this case, fragmented patching reduces traffic by 27% (trunk) but adds traffic by 21% and 74% (branches) when the request rate is 20. However, it reduces traffic by 55% (trunk link) and by 30% and 4% (branches) when the request rate is 100.

1. Introduction

There have been many reports on techniques to reduce network bandwidth usage for on-demand video distribution. Two recent asynchronous multicast techniques — patching [7)~9)] and hierarchical merging [11)~14)] — are particularly promising. The hierarchical merging technique requires less network bandwidth than the patching technique, but introduces more frequent grafting of the multicast tree. This paper proposes a more optimal form of asynchronous multicasting based on the patching technique; we call this fragmented patching. In this technique, the patch flows are broken up into segments and each of them is sent in multicasting to be shared with multiple clients, as opposed to the patching technique where the patch flows are sent simply through unicasting. As a result, fragmented patching reduces traffic much further than either patching or hierarchical merging. This paper compares the three techniques by using mathematical models, which is the first to present the average usage of link bandwidth with the traffic intensity (Erlang) and the average frequency of tree grafting for multicasting. Through numerical analysis, we show that fragmented patching considerably lessens the traffic at both trunk and branch links along the distribution tree, e.g., compared with use of the hierarchical merging technique, it reduces the traffic intensity by 55% on the trunk link and by 75% and 76% on the branch links (which are branched to four and eight links, respectively), when the request rate is 100 and the content length is 2. However, the fragmented patching allows the rate of multicast tree grafting to rise. To reduce network operational complexity, we allow broadcasting of the segments. In this case, although the frequency of multicast tree grafting with fragmented patching is the same as with the patching technique, it allows the effectiveness of retrenching traffic on the branch links to decline when the request rate is low. For example, compared with the hierarchical merging, fragmented patching reduces traffic by 27% on the trunk but adds traffic by 21% and 74% on the branches when the request rate is 20. However, the technique reduces traffic further than hierarchicla merging as the request rate becomes high. For example, it reduces traffic by 55% on the trunk and by 30% and 4% on

† Japan Radio Co., Ltd.
†† The University of Electro Communication
††† National Institute of Information and Communications Technology

the branches when the request rate is 100.

2. Background

A number of multicast techniques for on-demand video distribution have been studied. The batching technique [1] is an early example. This technique periodically distributes multicast flows, and a client then waits to receive content until a next multicast flow has started transmitting. Although the technique substantially reduces the necessary server bandwidth, clients must accept an undesirable latency before they receive content. In contrast, the piggybacking technique [2] provides immediate delivery service to each client. This technique merges clients by speeding up and slowing down the client's playback rate (typically within 5% of what the user is willing to tolerate). However, the piggybacking technique is premised on the requests of merged clients occurring at nearly the same time.

More advanced techniques have included the block-transfer technique of Woo and Kim [3] and Kalva and Fuhrt [4], where the video content is divided into small data units and each of units is multicast to clients that needs to receive. The data-transmission rate is three to four times as high as the rate at which the data are played back. Also, Uno and Tode [5),6] developed a burst-transfer technique where such small data units are distributed in a burst transmission manner. These two techniques succeed in completely immediate delivery service, but require wide bandwidth availability at client's network interface and a large number of multicast groups to be assigned to each data unit, which will introduce unacceptable complexity on network operation, that is, ceaseless construction of multicast trees.

Carter and Long [7], Hua and Cai [8),9], and Gao and Towsley [10] reported patching techniques based on streaming-transfer. The techniques perform both multicasting (for shared data) and unicasting (for patched data) where the data are transmitted at a rate equal to the data playback rate (Section 2.1 describes in detail). They require less bandwidth availability at client's network interface and fewer numbers of multicast groups, in addition to immediate delivery service. Therefore, patching technique can be considered as one of promising scheme in reality. Gao and Towsley showed the required server bandwidth and optimal generation rate of multicast flows with patching technique through a mathematical approach. Eager et al. proposed the hierarchical merging technique [11)~13], where two neighboring patched data flows are merged (Section 2.2 describes in detail). This technique outperforms the previously reported patching technique.

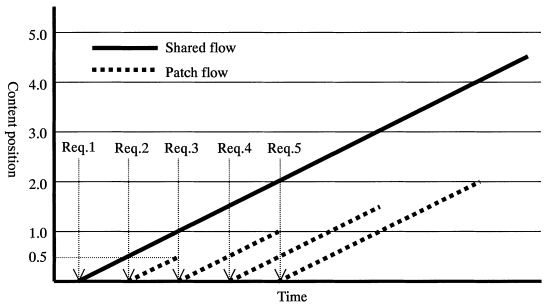

Fig. 1 Patching technique.

All of the above work gave consideration of way to reduce the required server bandwidth, not the network bandwidth. Focusing on the network, Zhao and Eager analyzed the required bandwidth on both trunk and branch links along the distribution tree for their hierarchical merging technique [14]. Also, Sato and Katsumoto, et al. [15)~18] proposed an optimal form of traffic control at the trunk and branch links to meet network bandwidth design requirements by using a patching technique.

2.1 Patching Technique

When a patching technique is used, media content sent by multicasting is called the shared flow as it is shared among clients whom make requests at about the same time. The initial content data, which clients making requests later cannot obtain, are individually delivered through unicasting — which is called the patch flow. One client receives a shared flow only, and the other subsequent clients receive both shared and patch flows. As shown in **Fig. 1**, the first client (Req. 1) receives a shared flow only, and later arriving clients (Req. 2, 3, 4, and 5) receive both shared and patch flows. The patch flow for the Req. 2 client, for example, provides the initial portion of data from the beginning of the content to the 0.5 content position which corresponds to the arrival time of Req. 2. The shared data are not immediately played back, but are buffered until the patch flow data have been completely played back. To minimize network traffic, the technique dynamically determines the generation rate of shared flows according to a current request rate.

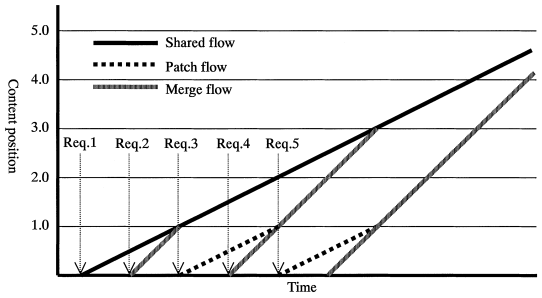

Fig. 2 Hierarchical merging technique.

2.2 Hierarchical Merging Technique

In hierarchical merging, media content is also multicast as a shared flow to clients whose requests are made at about the same time. The initial content data is delivered by both unicasting (the patch flow) and multicasting (the merge flow). The merge flow is shared by two neighboring later clients and transmitted at a rate higher than the data playback rate.

Figure 2 depicts the hierarchical merging technique. Clients of Req. 2 and 4 initially receive a merge flow and then start receiving a shared flow at content positions 1.0 and 3.0, respectively. Clients of Req. 3 and 5 initially receive a patch flow, then start receiving a merge flow at content position 1.0, and then start receiving a shared flow. This technique can also be used to control the generation rate of shared flows to minimize network traffic.

3. Fragmented Patching Technique

This paper proposes a new technique, fragmented patching, based on the patching technique. In fragmented patching, the content is divided into segments whose length is the reciprocal of the request arrival rate (i.e., the request interval). Each segment is sent in both shared and patch flows and is shared by multiple clients. The main features are as follows.

- Each segment in patch flows is multicast to multiple clients that need to receive; clients share even patch flows to reduce the patch flow traffic.
- Segments in a shared flow are distributed with one multicast group as with patching technique. This prevents expansion of the number of multicast groups to be managed.
- All segments are transmitted at a rate equal to the data playback rate as with patching technique, which does not require wide bandwidth availability at client's network interface.

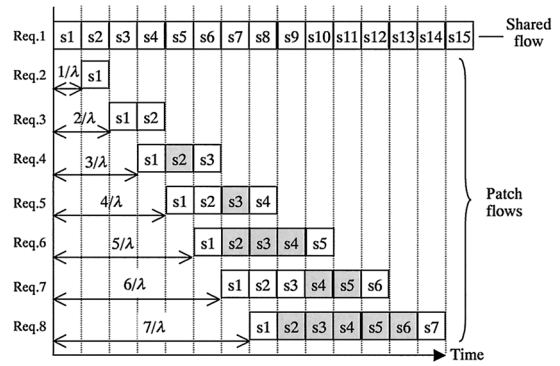

Fig. 3 Fragmented patching technique: segments shown in gray are shared with those needed for a previous request; i.e., they are not actually sent.

- The segment length is dynamically determined according to the current request rate to constantly keep the traffic and the number of multicast groups at minimum possible.

Figure 3 shows how to share the patch flows. Each segment is expressed as $s1$, $s2$, ... in order from the beginning of the content. Assuming that the request rate is now λ, every segment has the same length: $1/\lambda$. Consider a client corresponding to request 6, for example. This request arrives $5/\lambda$ after request 1, and the client then needs a patch flow that includes five of the segments: $s1$, $s2$, $s3$, $s4$, and $s5$. In this case, $s1$ and $s5$ must be newly sent, but $s2$, $s3$, and $s4$ can be shared with the previous clients corresponding to requests 5, 4, and 5, respectively. (In the figure, segments that can be shared with a previous client are shown in gray.) Figure 3 shows one important rule; $s1$ is newly sent for every request, $s2$ is newly sent for every second request and $s3$ is newly sent for every third request. Namely, sn is newly sent for every nth request.

To help clarify the basic concept of fragmented patching, the above explanation premises that every request interval is the same. In reality, though, intervals are always unsettled. **Figure 4** shows an example of random request arrivals. Requests 4 and 5 arrive, respectively, $0.5/\lambda$ and $1/\lambda$ earlier than the normal time. Requests 6 and 8 arrive, respectively, $1/\lambda$ and $0.5/\lambda$ later than the normal time. In this situation, patch flows segments are shared as follows.

The premature request 5 allows the client to receive $s2$, which is shared with the previous

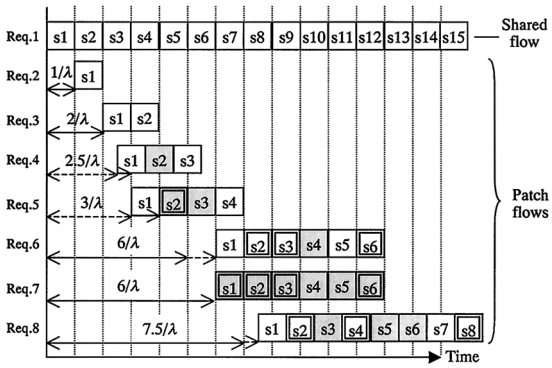

Fig. 4 Fragmented patching technique: requests arrive at random. The segments with a double box are those that differ from the segments in Fig. 3.

client of request 3. The delayed request 6 means that for the client to receive $s2$, $s3$, and $s6$, the server must newly send these segments. However, this allows the next client to receive $s1$, $s2$, $s3$, and $s6$ as a flow shared with the client of request 6. The delayed request 8 means that for the client to receive $s2$ and $s4$, the server must newly send these segments.

As can be seen from Fig. 3 and 4, the number of segments that the server actually sends to clients remains approximately the same whether the request arrival is random or uniform. That is, the required bandwidth should not be affected even if requests arrive at random.

To implement fragmented patching, the content server performs the following simple process.

Upon receiving a request, the server
- checks the lapsed time from the start time of the last shared flow
- determines which segments the new client needs to receive
- sees if any of those segments have already been scheduled to be sent
- sees if any of the scheduled segments can be received by the new client
- determines which segments must be newly sent
- records the segment numbers and time to be sent in the schedule table
- informs the client of the schedule
- sends the segments according to the schedule table

In addition, the server calculates the average request rate every time it receives request. The server changes the segment length if it detects a significant change in the rate. Meanwhile,

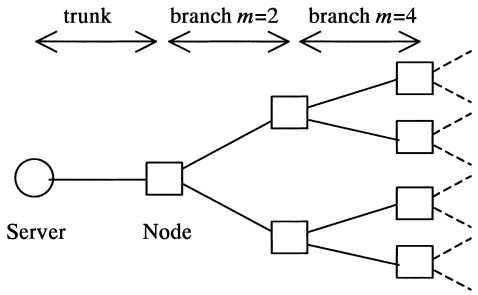

Fig. 5 Network model.

clients join multicast groups according to the received schedule table. There is some protocol overhead (delay) of constructing multicast tree to deliver the segment, the clients thus joins multicast group (grafts tree) in advance with anticipation of the delay.

4. Mathematical Models

This section considers the required network bandwidth for both trunk and branch links and the frequency of grafting multicast trees for each of the three techniques. The required network bandwidth can be expressed as the traffic intensity (Erlang), which is the product of the average request rate, the average flow length, and the average flow bandwidth. (Refer to Appendix A for details.)

This paper lets h, λ and τ denote, respectively, the content length, the request rate, and the generation rate of shared flows. We assume that the content is transmitted at a constant bit rate with bandwidth 1, except for the merge flow in hierarchical merging. **Figure 5** shows the network model and the link definitions. The network is a balanced tree topology. The trunk link is a single link that directly connects to the server. The branch links are those that are branched by the node being capable of multicasting. The number of branch links is expressed as m. The reason of using symmetric tree is that we can assume the request rate on m branched links as the request rate on the trunk link divided by m, assuming that the same number of clients connect to each of links and requests take place equally, i.e., there is no deviation in the request rate among links. (Note that a real network does not always form symmetric tree. In such case, the request rate basically depends on the number of downstream clients, so that the traffic intensity varies according to the request rate.) Each flow is actually delivered at varied intervals, although equal

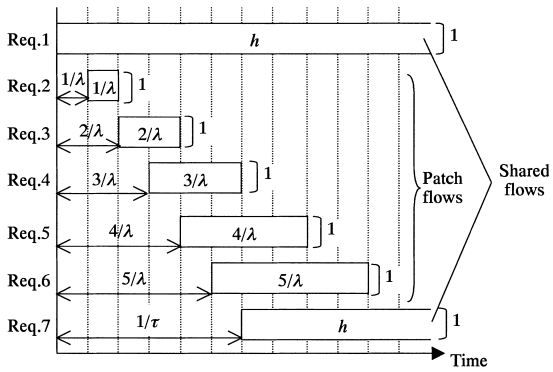

Fig. 6 Trunk link flows with the patching technique.

intervals are shown in figures below. We assume that requests arrive randomly in micro time span, i.e., each request occurs independently without any correlation between each of them.

4.1 Patching Technique

4.1.1 Traffic Intensity on the Trunk Link

Let's first consider the traffic intensity of shared flows. The bandwidth $b_{p\_shared\_t}$, the rate $r_{p\_shared\_t}$, and the length $l_{p\_shared\_t}$ of a shared flow are, respectively,

$$b_{p\_shared\_t} = 1 \quad (1)$$

$$r_{p\_shared\_t} = \tau \quad (2)$$

$$l_{p\_shared\_t} = h \quad (3)$$

The traffic intensity of shared flows $\rho_{p\_shared\_t}$ is then

$$\rho_{p\_shared\_t} = b_{p\_shared\_t} \times r_{p\_shared\_t} \times l_{p\_shared\_t} = \tau h \quad (4)$$

Next, we consider the traffic intensity of patch flows. The bandwidth $b_{p\_patch\_t}$ and the rate $r_{p\_patch\_t}$ of a patch flow are, respectively,

$$b_{p\_patch\_t} = 1, \quad (5)$$

$$r_{p\_patch\_t} = \lambda - \tau \quad (6)$$

The number of patch flows between two shared flows is $\lambda/\tau - 1$. As **Fig. 6** shows, the patch flow lengths are $1/\lambda, 2/\lambda, 3/\lambda, \ldots$, so the average length is

$$l_{p\_patch\_t} = \frac{1}{\lambda/\tau - 1} \sum_{k=1}^{\lambda/\tau - 1} \frac{k}{\lambda} = \frac{1}{2\tau} \quad (7)$$

The traffic intensity of patch flows $\rho_{p\_patch\_t}$ is then

$$\rho_{p\_patch\_t} = b_{p\_patch\_t} \times r_{p\_patch\_t} \times l_{p\_patch\_t} = 1 \times (\lambda - \tau) \times \frac{1}{2\tau} = \frac{\lambda - \tau}{2\tau} \quad (8)$$

Therefore, with the patching technique the total traffic intensity on the trunk link ρ_{p_t} is

$$\rho_{p\_t} = \rho_{p\_shared\_t} + \rho_{p\_patch\_t} = \tau h + \frac{\lambda - \tau}{2\tau} \quad (9)$$

4.1.2 Traffic Intensity on Branch Links

Starting with the traffic intensity of shared flows, the bandwidth $b_{p\_shared\_b}$ of a shared flow is

$$b_{p\_shared\_b} = 1 \quad (10)$$

The number of requests between two shared flows is expressed as $N (N = \lambda/\tau)$. The possibility that a shared flow will not occur on a branch link is $((m-1)/m)^N$. The expected rate of shared flow occurrence on the branch link, $r_{p\_shared\_b}$, is

$$r_{p\_shared\_b} = \left(\frac{m-1}{m}\right)^N \times 0 + \left(1 - \left(\frac{m-1}{m}\right)^N\right) \times \tau = \left(1 - \left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right) \times \tau \quad (11)$$

The shared flow on a branch link will have one of two lengths. When a request triggering the dispatch of a shared flow occurs on the branch link, the length of the shared flow is h (case 1 in **Fig. 7**). Otherwise, when a request that does not trigger the dispatch of a shared flow occurs on the branch link, the length of the shared flow is $h - 1/(\lambda/m)$ (case 2 in Fig. 7). Here, $1/(\lambda/m)$ is the time until the first patch flow occurs on the branch link. The possibility of case 1 is $1/N$ (i.e., as τ approaches λ — as the proportion of the shared flow becomes large — the frequency of case 1 increases).

Therefore, the expected value for the length of the shared flow, $l_{p\_shared\_b}$, is

$$l_{p\_shared\_b} = h \times \frac{1}{N} + \left(h - \frac{1}{\lambda/m}\right)\left(1 - \frac{1}{N}\right) = h + \frac{m\tau}{\lambda^2} - \frac{m}{\lambda} \quad (12)$$

The traffic intensity of shared flows $\rho_{p\_shared\_b}$ is then

$$\rho_{p\_shared\_b} = b_{p\_shared\_b} \times r_{p\_shared\_b} \times l_{p\_shared\_b} = 1 \times \left(1 - \left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right) \times \tau$$

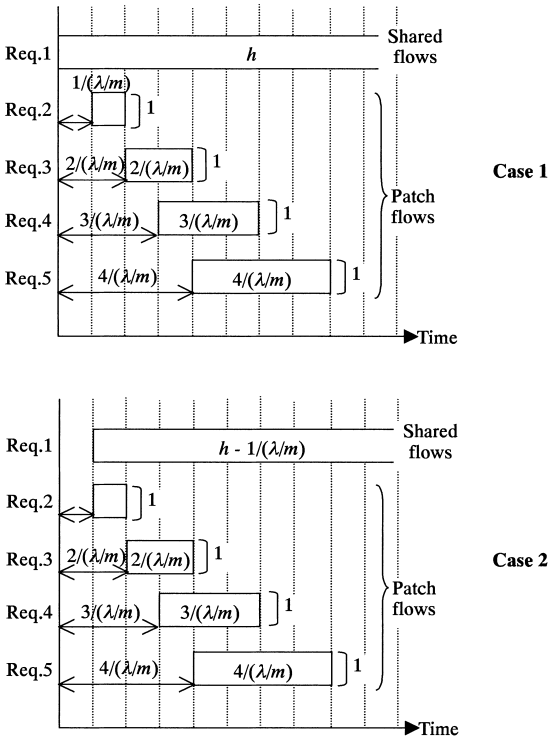

Fig. 7 Branch link flows with the patching technique.

$$\times \left(h + \frac{m\tau}{\lambda^2} - \frac{m}{\lambda}\right) \quad (13)$$

Regarding the traffic intensity for patch flows, the bandwidth $b_{p\_patch\_b}$ is

$$b_{p\_patch\_b} = 1 \quad (14)$$

When the number of branch links is m, the rate $r_{p\_patch\_b}$ is

$$r_{p\_patch\_b} = \frac{\lambda - \tau}{m} \quad (15)$$

The average length of patch flows on a branch link is the same as on a trunk link:

$$l_{p\_patch\_b} = \frac{1}{2\tau} \quad (16)$$

The traffic intensity of patch flows $\rho_{p\_patch\_b}$ is then

$$\begin{aligned}\rho_{p\_patch\_b} &= b_{p\_patch\_b} \times r_{p\_patch\_b} \times l_{p\_patch\_b} \\ &= 1 \times \frac{\lambda - \tau}{m} \times \frac{1}{2\tau} = \frac{\lambda - \tau}{2m\tau}\end{aligned} \quad (17)$$

Therefore, with the patching technique the total traffic intensity on the branch link, ρ_{p_b}, is

$$\begin{aligned}\rho_{p\_b} &= \rho_{p\_shared\_b} + \rho_{p\_patch\_b} \\ &= \left(1 - \left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right) \times \tau \\ &\times \left(h + \frac{m\tau}{\lambda^2} - \frac{m}{\lambda}\right) + \frac{\lambda - \tau}{2m\tau}\end{aligned} \quad (18)$$

4.1.3 Frequency of Tree Grafting

The number of multicast groups per unit time, g_p, is identical to the generation rate of shared flows;

$$g_p = \tau \quad (19)$$

The number of members that join the group for each shared flow, φ_p, is

$$\varphi_p = \frac{\lambda}{\tau} \quad (20)$$

With the patching technique, the frequency of grafting a multicast tree, o_p, is then

$$o_p = g_p \times \varphi_p = \tau \times \frac{\lambda}{\tau} = \lambda \quad (21)$$

4.2 Hierarchical Merging Technique

4.2.1 Traffic Intensity on the Trunk Link

The hierarchical merging technique has three kinds of flow (shared, patch, and merge flows), and we consider the traffic intensity of shared flows first. The bandwidth $b_{m\_shared\_t}$, the rate $r_{m\_shared\_t}$, and the length $l_{m\_shared\_t}$ of a shared flow are, respectively,

$$b_{m\_shared\_t} = 1 \quad (22)$$

$$r_{m\_shared\_t} = \tau \quad (23)$$

$$l_{m\_shared\_t} = h \quad (24)$$

The traffic intensity of shared flows $\rho_{m\_shared\_t}$ is then

$$\begin{aligned}\rho_{m\_shared\_t} &= b_{m\_shared\_t} \times r_{m\_shared\_t} \\ &\times l_{m\_shared\_t} = \tau h\end{aligned} \quad (25)$$

This is the same as with the patching technique.

Next, we consider the traffic intensity of patch flows. The bandwidth $b_{m\_patch\_t}$ of a patch flow is

$$b_{m\_patch\_t} = 1 \quad (26)$$

The patch flow and the merge flow are dispatched alternately. The patch flow rate $r_{m\_patch\_t}$ is then

$$r_{m\_patch\_t} = \frac{\lambda - \tau}{2} \quad (27)$$

The patch flow is sent until the subsequent merge flow has caught up with it. Every patch flow length is basically the same. As shown in **Fig. 8**, the patch flow length $l_{m\_patch\_t}$ is

$$l_{m\_patch\_t} = \frac{1}{\lambda} + \frac{1}{\lambda(a-1)} = \frac{a}{\lambda(a-1)} \quad (28)$$

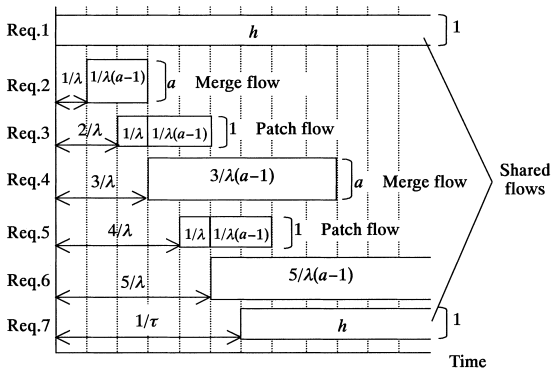

Fig. 8 Flows of the hierarchical merging technique on the trunk.

The patch flow traffic intensity $\rho_{m\_patch\_t}$ is then

$$\begin{aligned}&\rho_{m\_patch\_t}\\&=b_{m\_patch\_t}\times r_{m\_patch\_t}\times l_{m\_patch\_t}\\&=1\times\frac{\lambda-\tau}{2}\times\frac{a}{\lambda(a-1)}=\frac{a(\lambda-\tau)}{2\lambda(a-1)}\end{aligned}\quad(29)$$

Last, we consider the merge flow traffic intensity. The merge flows are sent at a higher bit rate than patch flows. The bandwidth $b_{m\_merge\_t}$ is

$$b_{m\_merge\_t}=a\quad(a>1).\qquad(30)$$

The rate $r_{m\_merge\_t}$ of merge flows is the same with that of patch flows, then,

$$r_{m\_merge\_t}=\frac{\lambda-\tau}{2}\qquad(31)$$

As Fig. 8 shows, each merge flow length is expressed as

$$\frac{1}{\lambda(a-1)},\frac{3}{\lambda(a-1)},\cdots,\frac{2k-1}{\lambda(a-1)}$$
$$k=1,2,\ldots,n/2,\quad n=\lambda/\tau-1$$

Here, n is the number of requests between two shared flows. The average length is

$$\begin{aligned}l_{m\_merge\_t}&=\frac{1}{n/2}\sum_{k=1}^{n/2}\frac{2k-1}{\lambda(a-1)}\\&=\frac{2}{n\lambda(a-1)}\sum_{k=1}^{n/2}(2k-1)\\&=\frac{n}{2\lambda(a-1)}=\frac{\lambda/\tau-1}{2\lambda(a-1)}\end{aligned}\quad(32)$$

The traffic intensity of merge flows is then

$$\begin{aligned}&\rho_{m\_merge\_t}\\&=b_{m\_merge\_t}\times r_{m\_merge\_t}\times l_{m\_merge\_t}\\&=a\times\frac{\lambda-\tau}{2}\times\frac{\lambda/\tau-1}{2\lambda(a-1)}\\&=\frac{a(\lambda/\tau-1)(\lambda-\tau)}{4\lambda(a-1)}\end{aligned}\qquad(33)$$

Therefore, with the hierarchical merging technique the total traffic intensity on the trunk link, ρ_{m_t}, is

$$\begin{aligned}\rho_{m\_t}&=\rho_{m\_shared\_t}+\rho_{m\_patch\_t}+\rho_{m\_merge\_t}\\&=\tau h+\frac{a(\lambda-\tau)}{2\lambda(a-1)}+\frac{a(\lambda/\tau-1)(\lambda-\tau)}{4\lambda(a-1)}\end{aligned}\qquad(34)$$

4.2.2 Traffic Intensity on Branch Links

First, we consider the traffic intensity of shared flows. The bandwidth $b_{m\_shared\_b}$ of a shared flow is

$$b_{m\_shared\_b}=1\qquad(35)$$

The number of requests between two shared flows is, as with the patching technique, expressed as $N(N=\lambda/\tau)$. The possibility that a shared flow will not occur on a branch link is $((m-1)/m)^N$. The expected rate of shared flow occurrence on the branch link, $r_{m\_shared\_b}$, is

$$\begin{aligned}r_{m\_shared\_b}&=\left(1-\left(\frac{m-1}{m}\right)^{N}\right)\times\tau\\&=\left(1-\left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right)\times\tau\end{aligned}\qquad(36)$$

The shared flow on a branch link will have one of two lengths. When a request triggering the dispatch of a shared flow occurs on the branch link, the length of the shared flow is h (case 1 in **Fig. 9**). Otherwise, when a request that does not trigger the dispatch of a shared flow occurs on the branch link, the length of the shared flow is $h-m/\{\lambda(a-1)\}$ (case 2 in Fig. 9). Here, $h-m/\{\lambda(a-1)\}$ is the time until the first merge flow on the branch link has caught up with the shared flow. As with the patching technique, the possibility of case 1 is $1/N$. Therefore, the expected shared flow length, $l_{m\_shared\_b}$, is

$$\begin{aligned}l_{m\_shared\_b}&=\frac{1}{N}\times h+\left(1-\frac{1}{N}\right)\\&\quad\times\left(h-\left(\frac{m}{\lambda}+\frac{m}{\lambda(a-1)}\right)\right)\\&=h+\frac{ma(1-N)}{N\lambda(a-1)}\\&=h+\frac{ma(1-\lambda/\tau)}{\lambda^2(a-1)/\tau}\end{aligned}\qquad(37)$$

The traffic intensity of shared flows $\rho_{m\_shared\_b}$ is then

$$\rho_{m\_shared\_b}=b_{m\_shared\_b}\times r_{m\_shared\_b}$$

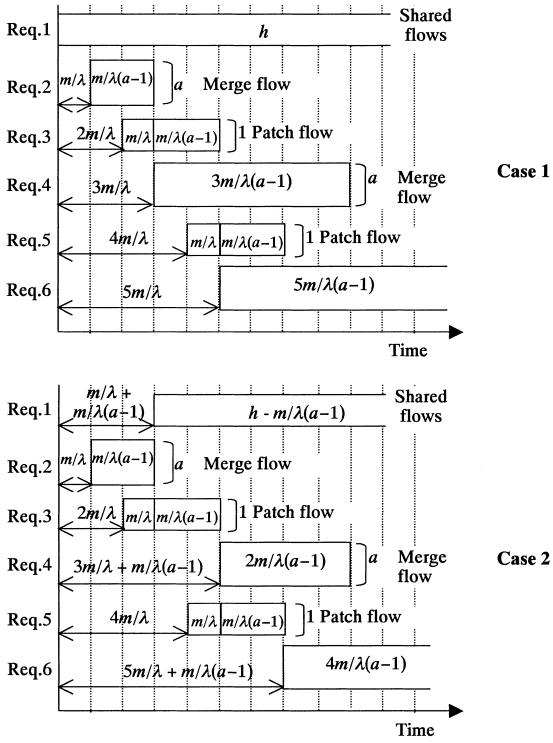

Fig. 9 Flows of the hierarchical merging technique on a branch.

$$\times l_{m\_shared\_b}$$
$$= 1 \times \left(1 - \left(\frac{m-1}{m}\right)^{\lambda/\tau}\right) \times \tau \times \left(h + \frac{ma(1-\lambda/\tau)}{\lambda^2(a-1)/\tau}\right) \quad (38)$$

Next, we consider the traffic intensity of patch flows. The bandwidth $b_{m\_patch\_b}$ of a patch flow is

$$b_{m\_patch\_b} = 1 \quad (39)$$

When the number of branch links is m, the rate $r_{m\_patch\_b}$ is

$$r_{m\_patch\_b} = \frac{\lambda - \tau}{2m} \quad (40)$$

As on the trunk link, the patch flow is sent until the subsequent merge flow has caught up with it, and every patch flow length is basically the same. As Fig. 9 shows, the patch flow length $l_{m\_patch\_b}$ is

$$l_{m\_patch\_b} = \frac{m}{\lambda} + \frac{m}{\lambda(a-1)} = \frac{am}{\lambda(a-1)} \quad (41)$$

The traffic intensity of patch flows $\rho_{m\_patch\_b}$ is then

$$\rho_{m\_patch\_b} = b_{m\_patch\_b} \times r_{m\_patch\_b} \times l_{m\_patch\_b} = 1 \times \frac{\lambda - \tau}{2m} \times \frac{am}{\lambda(a-1)} = \frac{a(\lambda-\tau)}{2\lambda(a-1)} \quad (42)$$

As a result, the traffic intensity is the same as on a trunk link.

Next, we consider the traffic intensity of merge flows. The bandwidth $b_{m\_merge\_b}$ of a merge flow is

$$b_{m\_merge\_b} = a \quad (a > 1). \quad (43)$$

The possibility of a merge flow occurring on a branch link is

$$1 - \left(\frac{m-1}{m}\right)^2$$

The rate $r_{m\_merge\_b}$ of merge flows is then

$$r_{m\_merge\_b} = \left(1 - \left(\frac{m-1}{m}\right)^2\right) \frac{\lambda - \tau}{2} \quad (44)$$

As Fig. 9 shows, each merge flow length is expressed as

$$\frac{m}{\lambda(a-1)}, \frac{(2+1/2)m}{\lambda(a-1)}, \frac{(4+1/2)m}{\lambda(a-1)}, \ldots = \begin{cases} \dfrac{m}{\lambda(a-1)} & (k=0) \\ \dfrac{m(2k+1/2)}{\lambda(a-1)} & (k=1,2,\ldots,n/2m-1) \end{cases}$$
$$n = \lambda/\tau - 1$$

where n is the number of requests between two shared flows as on the trunk link. Its average $l_{m\_merge\_b}$ is thus

$$l_{m\_merge\_b} = \frac{1}{n/2m}\left\{\frac{m}{\lambda(a-1)} + \sum_{k=1}^{n/2m-1} \frac{m(2k+1/2)}{\lambda(a-1)}\right\} = \frac{2m^2}{n\lambda(a-1)}\left\{1 + \sum_{k=1}^{n/2m-1}(2k+1/2)\right\} = \frac{n-m}{2\lambda(a-1)} = \frac{\lambda/\tau - m - 1}{2\lambda(a-1)} \quad (45)$$

The traffic intensity of merge flows $\rho_{m\_merge\_b}$ is then

$$\rho_{m\_merge\_b} = b_{m\_merge\_b} \times r_{m\_merge\_b} \times l_{m\_merge\_b} = a \times \left(1 - \left(\frac{m-1}{m}\right)^2\right) \frac{\lambda-\tau}{2} \times \frac{\lambda/\tau - m - 1}{2\lambda(a-1)} = \left(1 - \left(\frac{m-1}{m}\right)^2\right) \frac{a(\lambda-\tau)(\lambda/\tau - m - 1)}{4\lambda(a-1)} \quad (46)$$

Therefore, with the hierarchical merging technique the total traffic intensity on the branch link, ρ_{m_b}, is

$$\begin{aligned}\rho_{m\_b} &= \rho_{m\_shared\_b} + \rho_{m\_patch\_b} + \rho_{m\_merge\_b} \\ &= \left(1 - \left(\frac{m-1}{m}\right)^{\lambda/\tau}\right) \times \tau \\ &\quad \times \left(h + \frac{ma(1-\lambda/\tau)}{\lambda^2(a-1)/\tau}\right) + \frac{a(\lambda-\tau)}{2\lambda(a-1)} \\ &\quad + \left(1 - \left(\frac{m-1}{m}\right)^2\right) \frac{a(\lambda-\tau)(\lambda/\tau - m - 1)}{4\lambda(a-1)}\end{aligned} \tag{47}$$

4.2.3 Frequency of Tree Grafting

The number of multicast groups per unit time, corresponding to the generation rate of shared flows,g_{m_shared} is

$$g_{m\_shared} = \tau \tag{48}$$

The number of multicast groups corresponding to the generation rate of merge flows, g_{m_merge} is

$$g_{m\_merge} = \frac{\lambda - \tau}{2} \tag{49}$$

The number of members that join the group for each shared flow, φ_{m_shared}, is

$$\varphi_{m\_shared} = \frac{\lambda}{\tau} \tag{50}$$

The number of members that join the group for each merge flow, φ_{m_merge}, is

$$\varphi_{m\_merge} = 2 \tag{51}$$

The frequency of grafting a multicast tree with the hierarchical merging technique o_m is then

$$\begin{aligned}o_m &= g_{m\_shared} \times \varphi_{m\_shared} + g_{m\_merge} \\ &\quad \times \varphi_{m\_merge} = \tau \times \frac{\lambda}{\tau} + \frac{\lambda - \tau}{2} \times 2 \\ &= 2\lambda - \tau\end{aligned} \tag{52}$$

The traffic intensity, $\rho = f(\tau)$ in Eq. 47 takes its minimum value when τ is approximately

$$\tau \approx \sqrt{\frac{\lambda}{4mh}} + \frac{4}{mh} + 1$$

Therefore, Eq. 52 becomes

$$\begin{aligned}o_m &= 2\lambda - \sqrt{\frac{\lambda}{4mh}} - \frac{4}{mh} - 1 \\ &\leq \lim_{m\to\infty}\left(2\lambda - \sqrt{\frac{\lambda}{4mh}} - \frac{4}{mh} - 1\right) \\ &= 2\lambda - 1\end{aligned} \tag{53}$$

4.3 Fragmented Patching Technique

4.3.1 Traffic Intensity on the Trunk Link

The fragmented patching technique divides the shared flow and patch flow into fixed-length segments. We consider the traffic intensity of shared flows first. The bandwidth $b_{f\_shared\_t}$, the rate $r_{f\_shared\_t}$, and the length $l_{f\_shared\_t}$ of a shared flow are, respectively,

$$b_{f\_shared\_t} = 1 \tag{54}$$

$$r_{f\_shared\_t} = \tau \tag{55}$$

$$l_{f\_shared\_t} = h \tag{56}$$

The traffic intensity of shared flows $\rho_{f\_shared\_t}$ is then

$$\begin{aligned}\rho_{f\_shared\_t} &= b_{f\_shared\_t} \times r_{f\_shared\_t} \\ &\quad \times l_{f\_shared\_t} = \tau h\end{aligned} \tag{57}$$

This is the same as with the two above techniques.

Now we consider the traffic intensity of the fragmented patch flow. The bandwidth $b_{f\_fragment\_t}$ and the rate $r_{f\_fragment\_t}$ are

$$b_{f\_fragment\_t} = 1 \tag{58}$$

$$r_{f\_fragment\_t} = \lambda - \tau \tag{59}$$

As shown in Fig. 3, the total length of fragmented patch flows can be expressed as

$$\begin{aligned}&n \times \frac{1}{\lambda} + \left\lceil \frac{n}{2} \right\rceil \times \frac{1}{\lambda} + , \ldots , + \left\lceil \frac{n}{n-1} \right\rceil \times \frac{1}{\lambda} \\ &= \frac{1}{\lambda} \sum_{k=1}^{n-1} \left\lceil \frac{n}{k} \right\rceil \quad n = \lambda/\tau - 1\end{aligned}$$

where n is the number of requests between two shared flows. The average $l_{f\_fragment\_t}$ is

$$\begin{aligned}l_{f\_fragment\_t} &= \frac{1}{n} \times \frac{1}{\lambda} \sum_{k=1}^{n-1} \left\lceil \frac{n}{k} \right\rceil \\ &= \frac{1}{\lambda(\lambda/\tau - 1)} \sum_{k=1}^{\lambda/\tau - 2} \left\lceil \frac{\lambda/\tau - 1}{k} \right\rceil\end{aligned} \tag{60}$$

The traffic intensity of a fragmented patch flow $\rho_{f\_fragment\_t}$ is then

$$\begin{aligned}&\rho_{f\_fragment\_t} \\ &= b_{f\_fragment\_t} \times r_{f\_fragment\_t} \\ &\quad \times l_{f\_fragment\_t} \\ &= \frac{\lambda - \tau}{\lambda(\lambda/\tau - 1)} \sum_{k=1}^{\lambda/\tau - 2} \left\lceil \frac{\lambda/\tau - 1}{k} \right\rceil\end{aligned} \tag{61}$$

Therefore, the total traffic intensity of the

fragmented patching technique on the trunk link, ρ_{f_t}, is

$$\rho_{f\_t} = \rho_{f\_shared\_t} + \rho_{f\_fragment\_t} = \tau h + \frac{\lambda-\tau}{\lambda(\lambda/\tau-1)} \sum_{k=1}^{\lambda/\tau-2} \left\lceil \frac{\lambda/\tau-1}{k} \right\rceil \quad (62)$$

4.3.2 Traffic Intensity on Branch Links

We consider the traffic intensity of shared flows first. The bandwidth $b_{f\_shared\_b}$ of a shared flow is

$$b_{f\_shared\_b} = 1 \quad (63)$$

As with the patching technique, the possibility that a shared flow will not occur on a branch link is $((m-1)/m)^N (N = \lambda/\tau)$. The expected rate of shared flow occurrence on the branch link, $r_{f\_shared\_b}$, is

$$r_{f\_shared\_b} = \left(1 - \left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right) \times \tau \quad (64)$$

Also, as in the patching technique, the shared flow has one of two lengths: h or $h - 1/(\lambda/m)$. The expected value for the length of the shared flow, $l_{f\_shared\_b}$, is

$$l_{f\_shared\_b} = h + \frac{m\tau}{\lambda^2} - \frac{m}{\lambda} \quad (65)$$

The traffic intensity of shared flows $\rho_{f\_shared\_b}$ is then

$$\rho_{f\_shared\_b} = b_{f\_shared\_b} \times r_{f\_shared\_b} \times l_{f\_shared\_b} = 1 \times \left(1 - \left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right) \times \tau \times \left(h + \frac{m\tau}{\lambda^2} - \frac{m}{\lambda}\right) \quad (66)$$

Next, we consider the traffic intensity of fragmented patch flows. The bandwidth $b_{f\_fragment\_b}$ is

$$b_{f\_fragment\_b} = 1 \quad (67)$$

When the number of branch links is m, the rate $r_{f\_fragment\_b}$ is

$$r_{f\_fragment\_b} = \frac{\lambda-\tau}{m} \quad (68)$$

The length $l_{f\_fragment\_b}$ is the same as on the trunk link, so

$$l_{f\_fragment\_b} = \frac{1}{\lambda(\lambda/\tau-1)} \sum_{k=1}^{\lambda/\tau-2} \left\lceil \frac{\lambda/\tau-1}{k} \right\rceil \quad (69)$$

The traffic intensity of a fragmented patch flow $\rho_{f\_fragment\_b}$ is then

$$\rho_{f\_fragment\_b} = b_{f\_fragment\_b} \times r_{f\_fragment\_b} \times l_{f\_fragment\_b} = \frac{\lambda-\tau}{m\lambda(\lambda/\tau-1)} \sum_{k=1}^{\lambda/\tau-2} \left\lceil \frac{\lambda/\tau-1}{k} \right\rceil \quad (70)$$

Therefore, with the fragmented patching technique the total traffic intensity on the branch link, ρ_{f_b}, is

$$\rho_{f\_b} = \rho_{f\_shared\_b} + \rho_{f\_fragment\_b} = \left(1 - \left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right) \times \tau \times \left(h + \frac{m\tau}{\lambda^2} - \frac{m}{\lambda}\right) + \frac{\lambda-\tau}{m\lambda(\lambda/\tau-1)} \sum_{k=1}^{\lambda/\tau-2} \left\lceil \frac{\lambda/\tau-1}{k} \right\rceil \quad (71)$$

4.3.3 Frequency of Tree Grafting

The number of multicast groups per unit time, corresponding to the generation rate of shared flows, g_{f_shared} is

$$g_{f\_shared} = \tau \quad (72)$$

The number of multicast groups corresponding to the generation rate of each segment, $g_{f\_fragment}$, is

$$g_{f\_fragment} = \left\lceil \frac{n+1}{2} \right\rceil, \left\lceil \frac{n+1}{3} \right\rceil, \ldots, \left\lceil \frac{n+1}{n} \right\rceil \quad (73)$$

The number of members that join the group for each shared flow, φ_{f_shared}, is

$$\varphi_{f\_shared} = \frac{\lambda}{\tau} \quad (74)$$

The number of members that join the group for each segment, $\varphi_{f\_fragment}$, is

$$\varphi_{f\_fragment} = 2, 3, \ldots, n \quad (75)$$

The frequency of grafting a multicast tree with the fragmented patching merging technique, o_f, is

$$o_f = g_{f\_shared} \times \varphi_{f\_shared} + \sum_{k=2}^{n} g_{f\_fragment}^{k} \times \varphi_{f\_fragment}^{k} = \tau \times \frac{\lambda}{\tau} + \sum_{k=2}^{n} \left\lceil \frac{n+1}{k} \right\rceil \times k$$

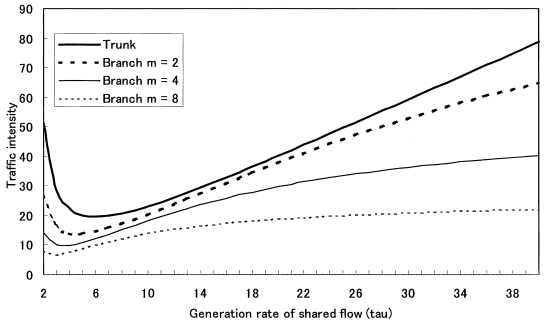

Fig. 10 Traffic intensity on each link with the patching technique.

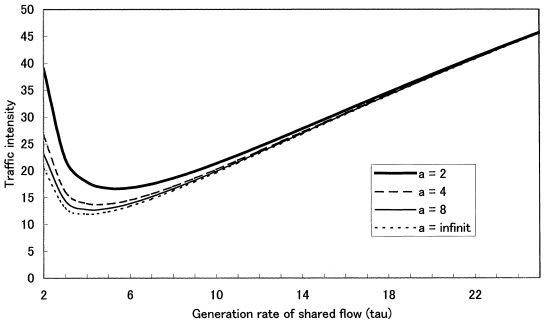

Fig. 11 Traffic intensity on a branch link, $m = 2$, with the hierarchical merging technique.

$$= \lambda + \sum_{k=2}^{n} \left\lceil \frac{\lambda/\tau}{k} \right\rceil \times k \tag{76}$$

5. Analysis and Considerations

5.1 Analysis on Mathematical Models

This section compares the three techniques using the mathematical models.

Figure 10 depicts the traffic intensity on both the trunk and branch links with the patching technique. The curves are functions of $\tau, \rho_{p\_t} = f(\tau)$, and $\rho_{p\_b} = f(\tau)$ in Eqs. 9 and 18, where the request rate λ and the content length h are set to100 and 2, respectively. Figure 10 shows that the functions are downward convex curves, which consist of liner function for shared flow's traffic and fractional function for patch flow's traffic, and each ρ takes its minimum value when τ is a particular value, $\tau_{\min}$. The traffic intensity at any τ decreases as the number of branches m increases. This is because the request rate on the link simply falls as m rises. As τ approaches to λ (every flow becomes to be transmitted as shared flow), ρ comes to the same traffic intensity as with simple unicast distribution, i.e., λh on the trunk and $\lambda h/m$ on the branches, respectively. The traffic intensity can be constantly maintained at a minimum by dynamically updating the generation rate of shared flows, $\tau_{\min}$, from the observed request rate. Using this feature, in our previous work [18] we proposed a dynamic-bandwidth allocation and traffic-control technique based on a patching technique to enable efficient use of network bandwidth resources, where the network traffic was adjusted to match the bandwidth dynamically assigned according to request rates for each content delivery.

With the hierarchical merging technique, the total traffic intensity decreases as the bandwidth of the merge flow, a, rises in Eqs. 34 and 47. **Figure 11** shows a function of τ, $\rho_{m\_b} = f(\tau)$ in Eq. 47, for example, and compares curves with different value of a. The figure suggests we can minimize the traffic intensity at any τ by transmitting the merge flow ideally with infinity of bandwidth. When the value of a approaches infinity, Eqs. 34 and 47 are respectively expressed as

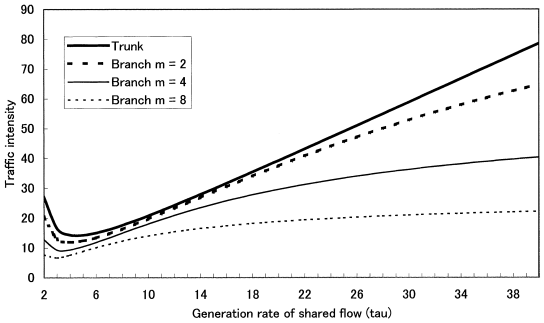

Fig. 12 Traffic intensity on each link with the hierarchical merging technique.

$$\lim_{a\to\infty} \rho_{m\_t} = \tau h + \frac{\lambda-\tau}{2\lambda} + \frac{(\lambda/\tau-1)(\lambda-\tau)}{4\lambda} \tag{77}$$

$$\begin{aligned}\lim_{a\to\infty} \rho_{m\_b} &= \left(1 - \left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right) \times \tau \\ &\times \left(h + \frac{m(1-\lambda/\tau)}{\lambda^2/\tau}\right) + \frac{\lambda-\tau}{2\lambda} \\ &+ \left(1 - \left(\frac{m-1}{m}\right)^2\right) \frac{(\lambda/\tau-m-1)(\lambda-\tau)}{4\lambda}\end{aligned} \tag{78}$$

Figure 12 depicts the traffic intensity on both trunk and branch links with the hierarchical merging technique. The curves are the func-

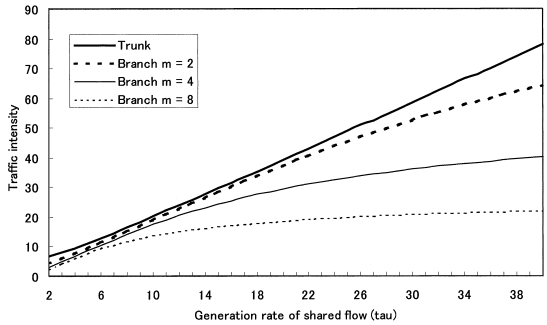

Fig. 13 Traffic intensity on each link with the fragmented patching technique.

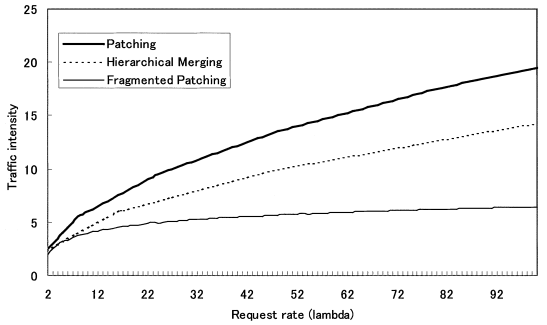

Fig. 14 Comparison of minimized traffic intensity on the trunk link for each technique.

tions $\rho_{m\_t} = f(\tau)$ and $\rho_{m\_b} = f(\tau)$ in Eqs. 77 and 78, where the request rate λ and the content length h are again set to100 and 2, respectively. As with the patching technique, the functions are downward convex curves and each ρ takes its minimum value when τ is a particular value, $\tau_{\min}$. As with patching, the traffic intensity at any τ decreases as the number of branches m increases, and as τ approaches to λ, ρ becomes the same traffic intensity as with simple unicast distribution.

Figure 13 depicts the traffic intensity with the fragmented patching technique. The curves are the functions $\rho_{f\_t} = f(\tau)$ and $\rho_{f\_b} = f(\tau)$ in Eqs. 62 and 71, where λ and h are set to 100 and 2, respectively. In fragmented patching, the traffic for patch flows is much further reduced, so that the traffic for shared flows is conspicuous. Therefore the curves become monotone increasing. We can thus assume that each ρ takes its minimum value at the smallest τ, namely, $\tau_{\min} = 1/h$. Substituting $\tau_{\min}$ for the Eqs. 62 and 71,

$$\rho_{f\_t} = 1 + \frac{\lambda - 1/h}{\lambda(\lambda h - 1)} \sum_{k=1}^{\lambda h - 2} \left\lceil \frac{\lambda h - 1}{k} \right\rceil \quad (79)$$

$$\rho_{f\_b} = \left(1 - \left(\frac{m-1}{m}\right)^{\lambda h}\right) \times \left(1 + \frac{m}{\lambda^2 h^2} - \frac{m}{\lambda h}\right) + \frac{\lambda - 1/h}{m\lambda(\lambda h - 1)} \sum_{k=1}^{\lambda h - 2} \left\lceil \frac{\lambda h - 1}{k} \right\rceil \quad (80)$$

As with the patching and the hierarchical merging, the traffic intensity at any τ decreases as the number of branches m increases, and as τ approaches to λ, ρ becomes the same traffic intensity as with simple unicast distribution.

Figure 14 compares the minimized traffic on the trunk link with the three techniques. The curves are the functions $\rho_{p\_t} = f(\lambda)$, $\rho_{m\_t} = f(\lambda)$ and $\rho_{f\_t} = f(\lambda)$ in Eqs. 9, 77 and 79, respectively, where h is set to 2 and $\tau_{\min}$ is computed for each λ. Meanwhile, **Fig. 15** compares the minimized traffic on branch links ($m = 4$ and 8) with the three techniques. The curves are the functions $\rho_{p\_b} = f(\lambda)$, $\rho_{m\_b} = f(\lambda)$ and $\rho_{f\_b} = f(\lambda)$ in Eqs. 18, 78 and 80, respectively, where h is set to 2 and $\tau_{\min}$ is again computed for each λ.

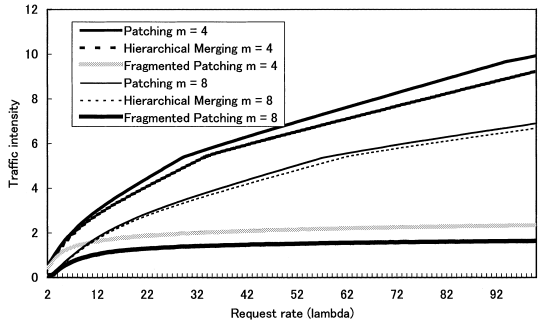

Fig. 15 Comparison of minimized traffic intensity on the branch links ($m = 4$ and 8) for each technique.

Fragmented patching clearly reduces the traffic on both trunk and branch links much better than the other techniques. The increase rate of the traffic intensity (the gradients of curves) in the fragmented patching is much smaller than in patching and hierarchical merging. The difference of traffic between fragmented patching and the others thus expands as request rate increases. The main reason is that the traffic of shared flows with the fragment patching is constant at any request rate (as shown in Eqs. 79 and 80, the rate of shared flows is $1/h$ at any time and its traffic intensity is then constantly 1). Namely, the increase of the traffic with increasing request rate is attributed to only the rise of patch flow's traffic with fragmented

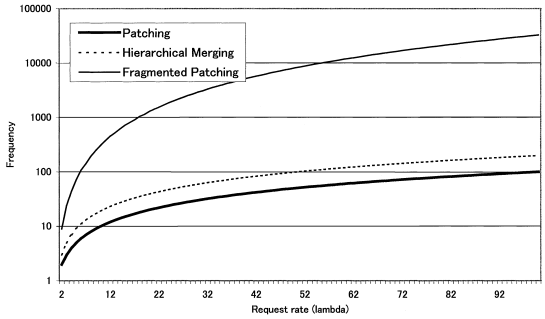

Fig. 16 Comparison of the frequency of multicast tree grafting for each technique.

patching, as opposed to the rise of both shared and patch flow's traffic with the other two techniques. The difference of traffic between patching and hierarchical merging is attributed to aggregation of neighboring two patch flows in hierarchical merging, which diminishes as the number of branches, m, increases (since the rate of patch flows on the branch link falls as m rises).

Compared with hierarchical merging, fragmented patching reduces traffic by 27% on the trunk and by 52% and 51% on the branches ($m = 4$ and 8 respectively) when the request rate $\lambda = 20$, and it reduces traffic by 43% on the trunk and by 67% and 68% on the same branches when $\lambda = 50$. Furthermore, it reduce traffic by 55% on the trunk and by 75% and 76% on the same branches when $\lambda = 100$.

5.2 Broadcasting of Patch Flows in Fragmented Patching

However, fragmented patching adds to network operational complexity. The number of multicast groups assigned to each segment increases as the request rate rises (since segment size becomes small). Eq. 76 indicates that the frequency of grafting multicast tree increases as τ becomes small to minimize the traffic intensity. When τ is $1/h$, Eq. 76 becomes

$$o_f = \lambda + \sum_{k=2}^{n} \left\lceil \frac{\lambda h}{k} \right\rceil \times k \tag{81}$$

Figure 16 compares the frequency of multicast trees being grafted when the three techniques are used (Eqs. 21, 53, and 81). As shown, the frequency with fragmented patching becomes extremely high as the request rate increases.

In fragmented patching we therefore propose broadcasting the segments on patch flows when the request rate is high. The frequency of multicast tree grafting can thus be reduced; i.e., the network only has to manage multicasting for the shared flow. In this case, the number of multicast groups per unit time g_{fb_shared} is

$$g_{fb\_shared} = \tau = \frac{1}{h} \tag{82}$$

The number of members that join the group φ_{fb_shared}, is

$$\varphi_{fb\_shared} = \frac{\lambda}{\tau} = \lambda h \tag{83}$$

The frequency of grafting a multicast tree, o_{fb}, is then

$$o_{fb} = g_{fb\_shared} \times \varphi_{fb\_shared} = \lambda \tag{84}$$

The frequency results in the same as with the patching technique.

We again consider the traffic intensity of fragmented patching with this method. It essentially maintains the same traffic intensity on the trunk link (the number of flows is equivalent on a single link whether segments are multicast or broadcast.) However, the method raises the traffic intensity on the branch links since segments are broadcast to even the links with no client that needs to receive them. Consider the traffic intensity of patch flows on the branch link. The bandwidth $b_{fb\_fragment\_b}$ and length $l_{fb\_fragment\_b}$ are the same as with the original fragmented patching (Eqs. 67 and 69), so

$$b_{fb\_fragment\_b} = 1 \tag{85}$$

$$\begin{aligned} l_{fb\_fragment\_b} &= \frac{1}{\lambda(\lambda/\tau - 1)} \sum_{k=1}^{\lambda/\tau - 2} \left\lceil \frac{\lambda/\tau - 1}{k} \right\rceil \\ &= \frac{1}{\lambda(\lambda h - 1)} \sum_{k=1}^{\lambda h - 2} \left\lceil \frac{\lambda h - 1}{k} \right\rceil \end{aligned} \tag{86}$$

The rate $r_{fb\_fragment\_b}$ is changed when segments are broadcast. Eq. 68 becomes

$$r_{fb\_fragment\_b} = \lambda - \tau = \lambda - \frac{1}{h} \tag{87}$$

The traffic intensity of fragmented patch flows $\rho_{fb\_fragment\_b}$ is then

$$\begin{aligned} \rho_{fb\_fragment\_b} &= b_{fb\_fragment\_b} \times r_{fb\_fragment\_b} \\ &\quad \times l_{fb\_fragment\_b} \\ &= \frac{\lambda - 1/h}{\lambda(\lambda h - 1)} \sum_{k=1}^{\lambda h - 2} \left\lceil \frac{\lambda h - 1}{k} \right\rceil \end{aligned} \tag{88}$$

The traffic intensity of shared flows on the branch link is the same as with the original frag-

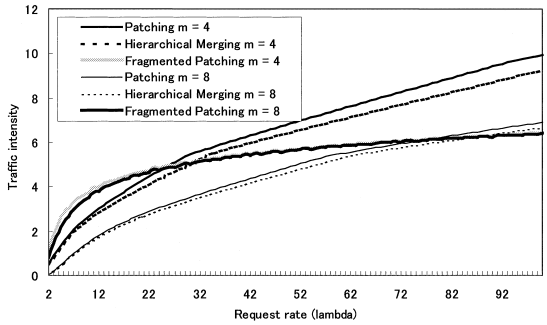

Fig. 17 Comparison of minimized traffic intensity for each technique on the branch links (m= 4 and 8), the segments on the patch flow are broadcasted in the fragment patching technique.

mented patching (Eq. 66). Therefore, the total traffic intensity on the branch link with the segment broadcasting in the fragmented patching technique, ρ_{fb_b}, is

$$\rho_{f\_b} = \left(1 - \left(\frac{m-1}{m}\right)^{\lambda h}\right) \times \left(1 + \frac{m}{\lambda^2 h^2} - \frac{m}{\lambda h}\right) + \frac{\lambda - 1/h}{\lambda(\lambda h - 1)} \sum_{k=1}^{\lambda h - 2} \left\lceil \frac{\lambda h - 1}{k} \right\rceil \quad (89)$$

Figure 17 again compares the minimized traffic on branch links ($m = 4$ and 8) with the three techniques. The curves are the functions $\rho_{p\_b} = f(\lambda)$, $\rho_{m\_b} = f(\lambda)$ and $\rho_{fb\_b} = f(\lambda)$ in Eqs. 18, 78 and 89, respectively.

In fragment patching using the segment broadcast, unnecessary segments take place on the branch link, so that the traffic intensity on branch link thus increases. The difference of the traffic between the segment broadcast and segment multicast is the increase of traffic for unnecessary segments, which is not attributed to the traffic for shared flows. Therefore, The increase rate of the traffic intensity (the gradients of curves) with the segment broadcast is similar to that with segment multicast, i.e., it is much smaller than that with patching and hierarchical merging, which includes the increase of shared flow traffic. Hence, the traffic on the branch links is higher than that with patching and hierarchical merging when the request rate is low and the number of branches is large, however, the traffic get lower as the request rate becomes large.

Compared with the hierarchical merging, fragmented patching reduces traffic by 27% on the trunk but adds traffic by 21% and 74% on the branches ($m = 4$ and 8 respectively) when the request rate $\lambda = 20$, and it reduces traffic by 43% on the trunk and by 12% on the branch ($m = 4$) but adds traffic by 20% on the branch ($m = 8$) when $\lambda = 50$. Furthermore, it reduces traffic by 55% on the trunk and by 75% and 76% on the branches ($m = 4$ and 8 respectively) when $\lambda = 100$.

Although fragmented patching with segment broadcasting causes the traffic intensity on the branch links to become relatively high as the number of branch links rises, this is not an important drawback since the effectiveness of traffic reduction in multicasting essentially decreases on the branch links as their number becomes large.

6. Future Work

Fragmented patching reduces the required network bandwidth at maximum when the shared flow rate τ is set to at minimum, i.e., the reciprocal to contents length. In this case, a large size of buffer memory is required at client systems (the buffer memory is used to buffer the shared data until patched data have been played back and the required buffer size falls as τ increases since the maximum length of patch flow corresponds to $1/\tau$.) Hence, we plan to consider applying the dynamic bandwidth allocation and traffic adjusting scheme [17),18)] to fragmented patching to realize efficient use of network bandwidth resource as reducing the required memory resource at clients.

Fragmented patching requires some special protocols. We plan to develop two protocols. A session control protocol is needed as application layer function, which establishes a session between server and clients to initiate fragmented patching and determines which multicast groups are assigned to both shared and patch flows to be received by client. Also, a quick constructing of multicast tree is strongly needed as network layer function. The protocol should be enhanced to an explicitly tree grafting basis protocol such as PIM-SM [19)] or PIM-SSM [20)], where the tree is grafted by an explicit join message sent from clients. These protocols, though, have inevitable overhead in constructing multicast tree, which is attributed to the bootstrap mechanism (flooding the multicast group information) and the join forwarding method with dependence on unicast routing. However, the bootstrap mechanism can be dispensable when the multicast souce (content delivery server) point is predetermined, and the forwarding delay can be lessened when the net-

work is statically formed so that join forwarding can be independent of unicast routing.

7. Conclusion

We have proposed a super optimal asynchronous multicast technique for on-demand video distribution. The main contributions in this paper are as follows.

We have described a new methodology of asynchronous multicasting based on patching technique, which we call fragmented patching. In this technique, even patch flows are sent through multicasting to be shared by multiple clients. As a result, the bandwidth required for the patch flows can be reduced.

We have presented mathematical models of the required network bandwidth and the frequency of multicast tree grafting for the three techniques — patching, hierarchical merging and fragment patching. Our analysis indicated that fragmented patching considerably lessens the traffic intensity at both trunk and branch links along the distribution tree, for example, compared with use of the hierarchical merging technique, it reduces the traffic intensity by 55% on the trunk link and by 75% and 76% on the branch links (which are branched to four and eight links, respectively) when the request rate is 100 and the content length is 2. However, the technique allows the rate of multicast tree grafting to rise. If we allow broadcasting of the patch flows, though, the rate of multicast tree grafting with fragmented patching remains the same as that with the patching technique. Broadcasting of the patch flows allows the effectiveness of retrenching traffic on the branch links to decline while the request rate is low, but as a result fragmented patching reduces traffic as the request rate become high. For example, compared with the hierarchical merging, fragmented patching reduces traffic by 27% on the trunk but adds traffic by 21% and 74% on the branches when the request rate is 20. However, it reduces traffic by 55% on the trunk and by 30% and 4% on the branches when the request rate is 100.

Recent growth of broadband network and digitalized broadcast service inspire us to expect high-quality video on-demand service to spread wireless and mobile communication area even such as personal cellular. An effective scheme is strongly required to such a wireless environment, where bandwidth resources will be strictly limited, thus, fragmented patching will extremely contribute to it.

Appendix

This appendix proves that the average usage of link bandwidth can be expressed as the traffic intensity (Erlang). The traffic intensity ρ is assumed to be the product of the average flow rate λ, the average flow length μ and the bandwidth of flow a:

$$\rho = a \times \lambda \times \mu \tag{90}$$

The average flow length μ is expressed as,

$$\mu = (\mu_1 + \mu_2 + \ldots + \mu_\lambda)/\lambda \tag{91}$$

Therefore, Eq. 90 becomes

$$\begin{aligned}\rho &= a \times (\mu_1 + \mu_2 + \ldots + \mu_\lambda) \\ &= a \times \sum_{k=1}^{\lambda} \mu_k = a \times \Lambda\end{aligned} \tag{92}$$

In Eq. 92, Λ is the total length of flows per unit time.

Also, we define a certain time-window as T, and divide T into ν time segments Δt. Letting $s_1, s_2, \ldots, s_\nu$ denote the number of flows in a link for each Δt, the total length of flows in the link can be expressed as

$$\begin{aligned}&s_1 \cdot \Delta t + s_2 \cdot \Delta t + \cdots + s_\nu \cdot \Delta t \\ &\quad = (s_1 + s_2 + \cdots + s_\nu)\Delta t \\ &\quad = \sum_{k=1}^{\nu} s_k \cdot \Delta t\end{aligned} \tag{93}$$

Let T be a unit of time (i.e., $T = 1$); Eq. 93 becomes the total length of flows per unit time, Λ. Note that $\nu \Delta t = 1$,

$$\begin{aligned}&s_1 \cdot \Delta t + s_2 \cdot \Delta t + \cdots + s_\nu \cdot \Delta t \\ &\quad = (s_1 + s_2 + \cdots + s_\nu)\frac{1}{\nu} \cdot \nu \Delta t = \frac{1}{\nu} \cdot \sum_{k=1}^{\nu} s_k \\ &\quad = \Lambda\end{aligned} \tag{94}$$

Equation 94 expresses the average number of flows in a link. If each bandwidth needed for a flow is a, $a \times \Lambda$ is the average link bandwidth usage. Therefore, the traffic intensity ρ in Eq. 92 is the average link bandwidth usage.

References

1) Dan, A., Sitaram, D. and Shahabuddin, P.: Scheduling Policies for an On-Demand Video Server with Batching, *Proc. 2nd ACM Int'l. Multimedia Conference*, San Francisco, CA (Oct. 1994).
2) Golubchik, L., Lui, J. and Muntz, R.: Adaptive Piggy-back: A Novel Technique for Data Sharing in Video-On-Demand Storage Servers, *ACM Multimedia System Journal*, pp.140–155, Vol.4, No.3 (1996).

3) Woo, H. and Kim, C.K.: Multicast scheduling for VOD services, *Multimedia Tools and Applications*, Vol.2, No.2, pp.157–171 (Mar. 1996).
4) Kalva, H. and Fuhrt, B.: Techniques for improving the capacity of video-on-demand systems, *Proc. 29th Annual Hawaii Int'l. Conf. on System Sciences*, pp.308–315, Wailea, HI, USA, IEEE Computer Society Press (Jan. 1996).
5) Uno, S., Tode, H. and Murakami, K.: Simple and Efficient Video-on-Demand Scheme with Segment Transmission over High Speed Network, *IEICE Trans. Comm.*, pp.106–115, Vol.E84-B, No.1 (Jan. 2001).
6) Xie, Z., Uno, S., Tode, H. and Murakami, K.: The Scheduling of Contents Delivery with Multicast and Burst Transfer, *IEICE Technical Report*, NS2002-58 (Jun. 2002).
7) Carter, S.W. and Long, D.E.: Improving Video-on-demand Server Efficiency Through Streaming Tapping, *Proc. Int'l. Conf. on Computer Communication and Networks*, pp.200–207, Las Vegas (Sep. 1997).
8) Hua, K.A., Cai, Y. and Sheu, S.: Patching: A Multicast Technique for True Video-On-Demand Services, *Proc. ACM Multimedia '98*, Bristol, U.K. (Sept. 1998).
9) Cai, Y., Hua, K.A. and Vu, K.: Optimizing Patching Performance, *Proc. Multimedia Computing and Networking 1999*, San Jose, CA (Jan. 1999).
10) Gao, L. and Towsley, D.: Supplying Instantaneous Video-on-Demand Services Using Controlled Multicast, *Proc. IEEE Int'l. Conf. on Multimedia Computing and Systems '99*, Florence, Italy (Jun. 1999).
11) Eager, D.L., Vernon, M.K. and Zahorjan, J.: Minimizing Bandwidth Requirements for On-Demand Data Delivery, *Proc. 5th Int'l. Workshop on Multimedia Information System*, Indian Wells, CA (Jan. 1999).
12) Eager, D.L., Vernon, M.K. and Zahorjan, J.: Optimal and Efficient Merging Schedule for Video-on-Demand Servers, *Proc. 7th ACM Int'l. Multimedia Conference*, Orlando, FL (Nov. 1999).
13) Eager, D.L., Vernon, M.K. and Zahorjan, J.: Bandwidth Skimming: A Technique for Cost-Effective Video-on-Demand, *Proc. Multimedia Computing and Networking 2000*, San Jose, CA (Jan. 2000).
14) Zhao, Y., Eager, D.L. and Vernon, M.K.: Network Bandwidth Requirement for Scalable On-Demand Streaming, *Proc. 21st Annual Joint Conf. IEEE INFOCOM 2002*, New York, NY (Jun. 2002).
15) Sato, K. and Katsumoto, M.: A Proposal of Multicast for Personalized Media Stream Delivery, *Proc. 16th Int'l. Conf. on Information Networking*, Vol.2 4D-4 (Jan. 2002).
16) Sato, K. and Katsumoto, M.: Multicast Techniques for Personalized Media Stream Delivery, *IPSJ Journal*, Vol.44, No.2 (Feb. 2003).
17) Sato, K., Katsumoto, M. and Miki, T.: Asynchronous Media Casting Network: An Optimal Network Scheme for On-demand Video Distribution, *Proc. 17th Int'l. Conf. on Advanced Information Networking and Application* (Mar. 2003).
18) Sato, K., Katsumoto, M. and Miki, T.: An Optimal Network Scheme for On-demand Video Distribution with Asynchronous Multicasting, *IPSJ Journal*, Vol.44, No.12 (Dec. 2003).
19) Estrin, D., Farinacci, D., et al.: Protocol Independent Multicast-Sparse Mode, RFC2362 (1998).
20) Holbrook, H. and Cain, B.: Source-Specific Multicast, IETF Internet Draft (Nov. 2001).

(Received January 20, 2004)
(Accepted September 3, 2004)

Katsuhiko Sato received the B.E. degree from Aoyama Gakuin University in 1992, and the M.E. degree from The University of Electro Communication in 2002. He has been working for Japan Radio Co., Ltd., where he has developed Frame Relay Router, ATM Switch, 3G Wireless Base station and Multi-layer Switch, and he is currently developing Wireless Mesh Router. He was a visiting researcher at Communication Research Laboratory in 2002–2003. He is currently working toward the Ph.D. in The University of Electro Communication. His research interests include mobile and multicast routing and VOD system. He is a member of IPSJ, IEEE Communication Society.

Michiaki Katsumoto received the B.S., M.S. and Ph.D. degrees from Toyo University in 1991, 1993 and 1996 respectively. He is working now National Institute of Information and Communications Technology. His research interests include next generation Internet applications and high-quality multimedia contents. He is a member of IPSJ, IEICE, IEEE Computer Society and ACM.

Minoru Terada received the B.E. degree in mathematical engineering in 1981, and the M.E. and Ph.D. degrees in information engineering from The University of Tokyo, in 1983 and 1990 respectively. From 1983 to 1991, he was an assistant at The University of Tokyo and the University of Electro-communications. From 1991 to 2002, he was an associate professor at The University of Tokyo. In 2002, he joined the University of Electro-communications, where he is an associate professor currently. His research interests include programming languages, programming environments and interactive systems. He is a member of JSSST and ACM.

Tetsuya Miki received a B.E. degree from the University of Electro-Communications in 1965, and M.E. and Ph.D. degrees from Tohoku University in 1967 and 1970, respectively. He is currently a Professor at the Department of Information and Communication Engineering, the University of Electro-Communications. His research involves optical and wireless access systems, photonic networks, multimedia information communications, and network architectures. He joined the Electrical Communication Laboratories of NTT in 1970, where he was engaged in the research and development of high-speed digital transmission systems using coaxial cable, optical transmission systems, optical access networks, ATM and multimedia transport networks, network operation systems and network architecture. He was the Executive Manager of the NTT Optical Network Systems Laboratories in 1992–1995. He received the IEICE Achievement Award in 1978 for his contribution to the early development of optical transmission systems. He is a Fellow of IEEE, and was Vice-President of IEEE Communications Society during 1998 and 1999. He is also a Fellow of IEICE, and is currently Vice-President of IEICE.