Vol. 44 No. 12 IPSJ Journal Dec. 2003

Recommended Paper

An Optimal Network Scheme for On-demand Video Distribution with Asynchronous Multicasting

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

This paper describes a new network scheme that minimizes the consumption of network resources for on-demand streaming-video distribution through multicasting. In general, on-demand video distribution has been realized through unicasting. There have been a number of studies that attempt to reduce the load on the VOD (video on-demand) server by using multicasting, which is called Asynchronous multicast technique, but they have given little consideration of ways to reduce the whole network load. Focusing on this point, we have developed an effective network distribution technique using asynchronous multicasting. In this paper, we present a statistical traffic-control theory based on asynchronous multicasting that can be used to adjust traffic on both trunk and branch links along the distribution tree to meet the network bandwidth design requirements to ensure QOS. In addition, we propose a dynamic bandwidth-allocation and traffic-adaptation algorithm that allows finite resources to be shared among different video deliveries by assigning the available bandwidth for each according to request rates. To show the implementability of these ideas, we consider their operation with protocols that autonomously construct a multicast tree and guarantee the QOS. Simulation results supported the validity of our new traffic control theory and indicated that the dynamic bandwidth-allocation and traffic adaptation algorithm could reduce the service-blocking rate of video delivery. The algorithm, for example, lessened 17% bandwidth resource in the four kind of video contents distribution (each video length is 2 hours, transmit rate is 1 [Mbps] and average requests rate is 100 [request/hour]).

1. Introduction

The broadband infrastructure for information distribution based on Internet technologies continues to grow and a number of Internet broadcast services, called Web casting [1], have been developed. These services provide, for example, live broadcasting and on-demand delivery via real-time streaming transfer. Since these services are typically provided at a charge, the network is highly required to be able to provide a guaranteed quality of distribution. Today, such a requirement is often met by forming a locally closed network for video distribution —— a Content Distribution Network (CDN) —— and the network operators strive for network resource optimization by using decentralized cache techniques.

We have developed a new network scheme, which we call 'Asynchronous Media Casting Network', to minimize the consumption of network resources for on-demand streaming-video distribution through multicasting, as opposed to unicasting that has generally been used to realize on-demand video distribution. Although a number of asynchronous multicast techniques for reducing the load on the VOD server have been reported, little attention has been directed towards reducing the whole network load. We have studied effective distribution techniques by using asynchronous multicasting focusing on the network resource optimization. In this paper, we first present a statistical traffic-control theory using asynchronous multicasting that allows adjustment of the traffic on both trunk and branch links along the distribution tree to meet network bandwidth design requirements and thus provide a guaranteed QOS. Second, we introduce a dynamic-bandwidth allocation and traffic-adaptation algorithm that allows the sharing of finite resources among many different video deliveries by assigning the available bandwidth for each according to request rates. Third, to determine the implementability of these ideas, we examine their operation with protocols that autonomously construct a multicast tree and guarantee the QOS.

† Japan Radio Co., Ltd
†† Communication Research Laboratory
††† The University of Electro Communication

The initial version of this paper was presented at the 10th DPS workshop held on Oct. 2002, which was sponsored by SIGDPS. This paper was recommended to be submitted to the Journal of IPSJ by the chairperson of SIGDPS.

Fourth, we examine the effects of using this scheme through numerical analysis and simulation regarding real traffic traces and the service-blocking rate of video delivery. Simulation results supported the validity of our new traffic control theory and indicated that the dynamic bandwidth-allocation and traffic adaptation algorithm could reduce the service-blocking rate.

2. General Problems with On-demand Video Distribution

In general, on-demand video distribution is realized through unicasting in which video data is carried to each user in separate flows. This approach uses up a lot of network resources.

In a CDN, a decentralized cache server is used to reduce the consumption of network resources. When multiple requests for delivery of the same video content occur at about the same time, the video for the first request is delivered from a server and for the subsequent requests the video is copied and delivered from the cache servers deployed between the server and receiver systems. This lowers the traffic between the server and cache servers, but does not reduce the traffic between the cache servers and receiver systems.

Multicasting is an alternative. In multicasting, a server sends video in a single flow that is shared with receivers that join the same multicast group. A network node copies the flow depending on the location of the receiving member and a distribution tree, called a multicast tree, is formed. A link on the tree that is near the server more effectively reduces traffic because it aggregates more flows to the receivers.

However, multicasting is basically unsuitable for on-demand streaming-video distribution. In general, multicasting for streaming-video targets live video distribution on a principle, where the video data are delivered to all multicast group members at the same time. In other word, the distribution must be 'synchronized' among all members.

3. Related Works: Multicast Technique for On-demand Video Distribution

A number of multicast techniques for on-demand video distribution have been studied. As an early study, the batching technique 3) was proposed. In this technique multicast flows are periodically distributed, a receiver system then waits until the next multicast transmission. This technique substantially reduces server bandwidth at the cost of undesirable latency to respond to a receiver system. The piggybacking technique 4) was next proposed, which provides immediate delivery service to each receiver. This technique merges receivers by speeding up and slowing down the receiver playback rate (typically within 5% that can be tolerated by a user). However, merging receivers cannot be achieved unless merged requests nearly occur.

Later, various asynchronous multicasting techniques have been developed (these are also called the patching techniques). The principle that underlies asynchronous multicasting is that video content are sent by multicasting, which is called shared flow, to users whose requests are made at about the same time, and the initial data of video content that subsequent receivers cannot obtain (i.e., the portion between the time when the shared flow starts and the time when a request occurs) is individually delivered by unicasting, which is called patch flow. One receiver receives a shared flow only, and the other subsequent receivers receive both shared and patch flows. In the latter case, the shared data are not immediately played back, but are buffered until the patch flow data have been completely played back. Woo and Kim 5) and Kalva and Fuhrt 6) showed a block-transfer based technique, where the video content is divided into small data units and the data-transmission rate is three to four times as fast as the rate at which the data are played back. Premising use of wide bandwidth, Uno and Tode 7),8) showed burst-transfer based technique, where the small data units are transmitted in a burst manner. Carter and Long 9), Hua and Cai 10),11) and Gao and Towsley 12) showed a stream-transfer based technique, where the video data is transmitted at a rate equal to the data playback rate. Gao and Towsley also showed required server bandwidth and optimal generation rate of multicast flows by a mathematical approach. Eager and et al. have been proposing the hierarchical merging technique 13)~15), which aggregates even patch flows that are transmitted at faster than playback rate. This technique is more efficient than the above simple patching techniques, requiring server bandwidth that scales as the logarithm of the request rate versus the square root of the request rate as simple patching.

All these works, however, have considered only the server bandwidth. The required network bandwidth has never considered until Zhao and Eager are recently aware of it. Their latest report [16] showed the required bandwidth on the branch link of distribution tree on the basis of their hierarchical merging technique. Meanwhile, we have been studying statistical traffic control technique [17)~19)] on the basis of stream based simple patching technique that Carter, Hua and Gao et al. had proposed. We have shown a mathematical model from a view of traffic intensity and traffic-adjustment technique to meet network bandwidth design requirement while striving for overall reduction of the use of receiver's buffer memory. Although the hierarchical merging technique that Eager et al. have proposed can reduce more bandwidth, it is required to construct a number of multicast trees, which introduce considerable protocol complexity and overhead. Therefore, we take up the simple patching technique, which is even enough to reduce the required bandwidth comparing it with a conventional unicast distribution method. We should rather consider reducing use of buffer memory at receivers by positively using the given bandwidth than consider only minimizing the required bandwidth. Zhao and Eager's report did not consider this point.

In our previous publications [19)], we demonstrated our statistical traffic-adjustment technique by simulation. Refs. 18) and 19) depicted network architectures and system behaviors to implement this technique on the actual Internet. This network has a hierarchical structure where a core network is laid as upstream of the distribution tree in which a multicast tree is statically constructed, and access networks are laid as downstream of the distribution tree in which a multicast tree is dynamically constructed. To further reduce traffic, patch flows were distributed from intermediate servers that are deployed at the boarder of core and access network.

However, in our designed network architecture the effectiveness of traffic retrenching fell rapidly as the number of branch links in the core network increased. This is because multicasting on the static tree was almost the same as broadcasting. In addition, our previous studies have not considered the traffic control at the branch links in the access network, so the effectiveness of traffic retrenching there should be verified more correctly. Furthermore, the centralized-control-based network model that we designed would have raised the network implementation cost and lowered the flexibility of network extensions, although it may have limited the protocol complexity.

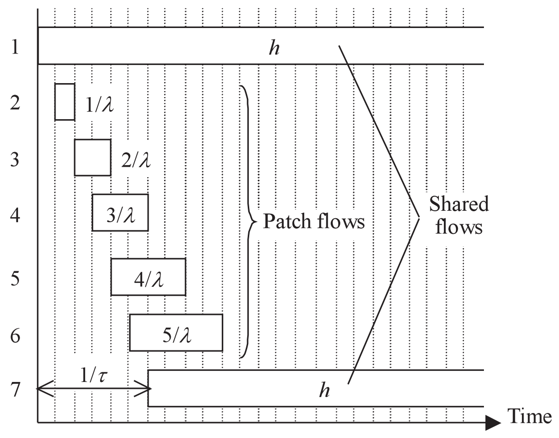

Fig. 1 Asynchronous multicasting.

4. New Traffic Control Technique with Asynchronous Multicasting

4.1 Basic Theory of Traffic Control

Here we explain basic theory of statistical traffic control described in our previous work [19)]. Consider the traffic intensity in stream-based asynchronous multicasting. We assume that the provided video is transmitted at a constant bit rate, that its length is h, and that delivery requests occur randomly (each request happens independently, i.e., there is no correlation between each of them). The average arrival rate of requests is λ, and the average arrival interval is $1/\lambda$. **Figure 1** depicts that shared flow is triggered by 1st request, which is shared with users requesting 1st to 6th by multicasting, and patch flows provide initial portion of data for users requesting 2nd to 6th by unicasting. Note that a flow is defined as a set of continuous packets and that each flow is actually delivered at varied intervals although it is depicted as at equal intervals in Fig. 1.

Here, the generation rate of shared flows is expressed as τ (τ is smaller than or equal to λ). The number of patch flows between two shared flows is $\lambda/\tau - 1$, the length of each patch flow is $1/\lambda, 2/\lambda, 3/\lambda, \ldots$, and the average length is thus,

$$\frac{1}{\lambda/\tau - 1} \sum_{k=1}^{\lambda/\tau-1} \frac{k}{\lambda} = \frac{1}{2\tau} \tag{1}$$

Traffic intensity (Erlang) is the product of the average rate and the average length. The traffic intensity of shared and patch flows are respectively τh and $(\lambda-\tau)/2\tau$. Total traffic intensity, ρ, is

$$\begin{aligned}\rho &= \tau h + (\lambda - \tau)\frac{1}{2\tau} \\ &= \tau h + \frac{\lambda}{2\tau} - \frac{1}{2} \quad \text{[erl]}\end{aligned} \tag{2}$$

In this equation, $\rho = f(\tau)$ is a downward convex curve and ρ takes its minimum value when $\tau = \sqrt{\lambda/2h}$. Hence, the traffic intensity can be constantly maintained at a minimum by updating τ to $\sqrt{\lambda/2h}$ from the observed λ and h.

The average use of the buffer memory at the receiver (this memory is used to buffer the shared-flow data and its size corresponds to the length of the patch flow) falls as τ increases. Therefore, in determining τ, there is a tradeoff between minimizing the traffic intensity and reducing the use of buffer memory.

Here, an upper bound of the traffic intensity, i.e., an available bandwidth for this on-demand media delivery service, is given as A: for example, we assume that a network service provider allocates the network bandwidth resources for each content delivery and other communication service to guarantee the quality of each service. The τ is determined as follows.

When the observed λh is below A, the traffic intensity does not exceed the upper bound even if all data is delivered by unicasting. Therefore, τ can be set to λ (i.e., all requested video content is delivered in a shared flow), so that use of the buffer memory at the receiver is zero.

When the observed λh is greater than A and the minimum value of ρ (i.e., $f(\sqrt{\lambda/2h}) = \sqrt{2\lambda h} - 1/2$) is smaller than A, τ can be set to the largest value that satisfies $f(\tau) \le A$. Here, the equation is

$$A = \tau h + \frac{\lambda}{2\tau} - \frac{1}{2} \quad \text{[erl]} \tag{3}$$

and the solutions for τ are

$$\tau = \frac{1/2 + A \pm \sqrt{(1/2+A)^2 - 2\lambda h}}{2h} \tag{4}$$

The larger solution is selected as τ.

When A is smaller than the minimum value of ρ (i.e., $f(\sqrt{\lambda/2h}) = \sqrt{2\lambda h} - 1/2$), τ can be set to $\sqrt{\lambda/2h}$ to minimize the traffic intensity.

Thus, Ref. 19) showed that we could flexibly control the traffic intensity with reducing the use of buffer memory at receivers in asynchronous multicasting by dynamically updating τ from the observed λ and h and determining A.

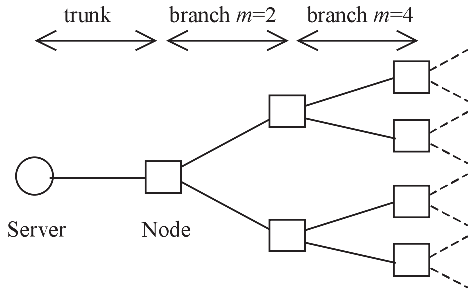

Fig. 2 Trunk and branch link.

4.2 Traffic Control Theory for Branch Links

This paper presents additional traffic control theory considering the traffic intensity for not only a trunk link but also branch links. As Section 4.1 concerns only the traffic intensity for a trunk link of a distribution tree, we should also look at the traffic intensity on the branch links. **Figure 2** shows the definitions of the links. The trunk link is a single link that directly connects to a delivery server. The branch links are ones branched by the node that performs multicasting not broadcasting. The number of branch links is expressed as m.

First, consider the shared-flow traffic on a branch link. 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, τ_b, is

$$\begin{aligned}\tau_b &= \left(\frac{m-1}{m}\right)^n \times 0 + \left(1 - \left(\frac{m-1}{m}\right)^n\right) \times \tau \\ &= \left(1 - \left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right) \times \tau\end{aligned} \tag{5}$$

The shared flow on a branch link will have one of two lengths. When a request triggering the generation of a shared flow occurs on the branch link, the length of the shared flow is h (case 1 in **Fig. 3**). Otherwise, when a request that dose not trigger the generation of a shared flow occurs on the branch link, the length of the shared flow is $h - 1/(\lambda/m)$ (case 2 in Fig. 3). $1/(\lambda/m)$

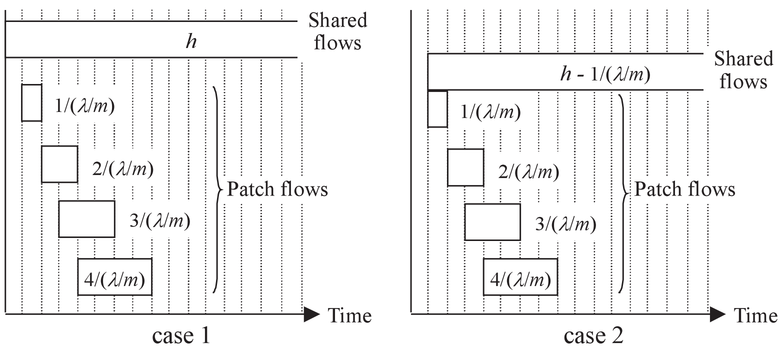

Fig. 3 Asynchronous multicasting for a branch link.

is the length of patch flow that occurs first 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, h_{b}, is

$$\begin{aligned} h_{\mathrm{b}} &= h \times \frac{1}{n} + \left(h - \frac{1}{\lambda/m}\right)\left(1 - \frac{1}{n}\right) \\ &= h + \frac{m\tau}{\lambda^2} - \frac{m}{\lambda} \end{aligned} \qquad (6)$$

The traffic intensity for the shared flow on a branch link is then

$$\begin{aligned} \tau_{\mathrm{b}} \times h_{\mathrm{b}} = &\left(1 - \left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right) \times \tau \\ &\times \left(h + \frac{m\tau}{\lambda^2} - \frac{m}{\lambda}\right) \ \text{[erl]} \end{aligned} \qquad (7)$$

Regarding the traffic intensity for patch flows, the rate of occurrence of patch flows on a branch link is $(\lambda - \tau)/m$ when the number of branch links is m, and the average length of patch flows on a branch link is $1/(2\tau)$, which is the same as in a trunk link. The traffic intensity is then

$$\frac{\lambda - \tau}{m} \times \frac{1}{2\tau} = \frac{\lambda}{2m\tau} - \frac{1}{2m} \ \text{[erl]} \qquad (8)$$

Therefore, the traffic intensity on a branch link is

$$\begin{aligned} \rho = &\left(1 - \left(\frac{m-1}{m}\right)^{\frac{\lambda}{\tau}}\right) \times \tau \\ &\times \left(h + \frac{m\tau}{\lambda^2} - \frac{m}{\lambda}\right) + \frac{\lambda}{2m\tau} - \frac{1}{2m} \ \text{[erl]} \end{aligned} \qquad (9)$$

Hence, we can express the traffic on a branch link as a mathematical model. This signifies that we can control the traffic on all links along the distribution tree.

4.3 Dynamic Bandwidth Allocation and Traffic-Adaptation Algorithm

In this paper, we propose dynamic bandwidth-allocation and traffic-adaptation algorithm that allows sharing of finite network resources among the various video deliveries by dynamically assigning each available bandwidth according to each request rate. A video delivery that is requested at a relatively low rate gives part of its available bandwidth to another video delivery that is requested at a higher rate. And the traffic for each video delivery is dynamically adjusted to the changed available bandwidth (we showed that a flexible traffic adjustment can be achieved throughout a network in the previous section). This allows τ to take a larger value, allowing us to reduce the use of receiver's buffer memory and lessen the service-blocking that arises from traffic exceeding the available bandwidth.

Figure 4 shows the algorithm. First, a request for video content C_i arrives at a server. In Proc. 1 the server records the number of arrivals within a predetermined period and updates request rate λ_i. In Proc. 2 the server determines τ_i based on λ_i as follows. τ_i is incremented from $\tau_{\min}$ ($\tau_{\min}$ must be predetermined considering the maximum use of buffer memory at the receivers is $1/\tau$) while ensuring that τ_i allows traffic to remain within the available bandwidth $A_i = \{a_i | a_{i1}, \ldots, a_{im}\}$ on all trunk and branch links (a_{i1} denotes the trunk link and a_{im} denotes for branch link m). Here, Eqs. (2) and (9) are used for the trunk and branch links, respectively, and τ_i is set to the largest value at which traffic does not exceed the available bandwidth on any link. If τ_i cannot be determined (i.e., there is no τ_i at which the traffic is within the available bandwidth on all links), in Proc. 3 the server finds another video content

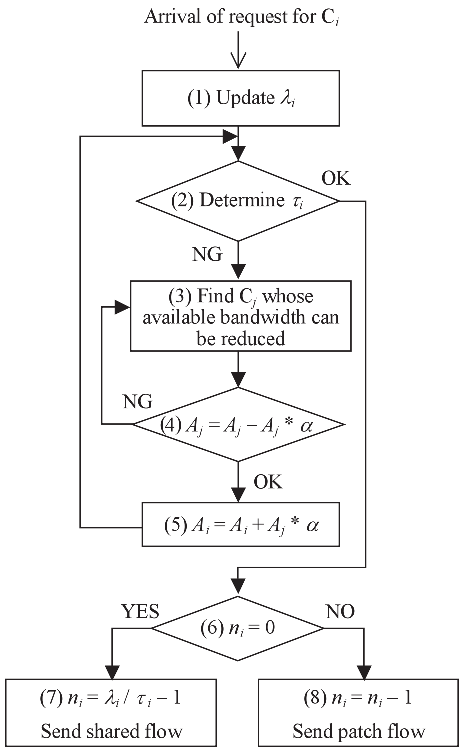
Arrival of request for C_i
(1) Update λ_i
(2) Determine τ_i
OK
NG
(3) Find C_j whose available bandwidth can be reduced
NG
(4) $A_j = A_j - A_j * \alpha$
OK
(5) $A_i = A_i + A_j * \alpha$
YES
(6) $n_i = 0$
NO
(7) $n_i = \lambda_i / \tau_i - 1$ Send shared flow
(8) $n_i = n_i - 1$ Send patch flow

Fig. 4 Algorithm.

C_j whose available bandwidth can be reduced. The server chooses C_j for which the largest τ is used of all C. In Proc. 4, the server removes $A_j \times \alpha$ $(0 < \alpha < 0.5)$ from the available bandwidth for the selected C_j and checks whether τ_j is still determinable in the same way as in Proc. 2 after the bandwidth reduction. If the available bandwidth for this C_j cannot be reduced, the server repeats Proc. 3 to try to find another video content that is a candidate to give up bandwidth. On the other hand, if the available bandwidth can be reduced, in Proc. 5 the removed bandwidth $A_j \times \alpha$ is added to the available bandwidth A_i for C_i. The server then retries to determine τ_i in Proc. 2. If τ_i can be determined, the server proceeds to Proc. 6. If the number of previously delivered patch flows is equal to zero $(n_i = 0)$, n_i is set to $\lambda_i/\tau_i - 1$ and the server sends a shared flow in Proc. 7. If n_i is not zero, n_i is decremented and the server sends the patch flow in Proc. 8.

5. Consideration of Implementation on an IP Network

5.1 Dynamic Construction of a Multicast Tree

In this paper, we consider a protocol that enables a network node system to dynamically form, in an autonomous manner, a multicast tree to distribute the shared flows. Many IP multicasting routing protocols have been developed in IETF. PIM-SM (Protocol-Independent Multicast Sparse Mode)[21], CBT (Core-Based Tree)[22] and SSM (Source-Specific Multicast)[23] are considered "receiver-driven" protocols where a multicast tree is formed in response to an explicit message sent by receivers to a rendezvous or source point indicating a wish to join a multicast group. With these protocols, traffic occurs only on links along the paths to receivers now requiring delivery, so that network resources are used efficiently.

Figure 5 shows the session initiation procedures between end systems on the basis of RTSP (Real-Time Streaming Protocol)[24], and multicast tree construction procedures among network node systems on the basis of PIM-SM and IGMP (Internet Group Management Protocol)[20]. An RTSP setup message establishes the transport-layer entities and provides receivers with a multicast address for a shared flow. An RTSP play message triggers the video transmission and lets receivers know the length of the patch flow. There is no change to RTSP specifications. The original information related to our technique is added to the messages as option parameter. The IGMP report and the PIM-SM register and join message are used to dynamically form multicast trees for the shared flow (any changes to IGMP and PIM-SM specifications are not required).

Here, we must consider the overhead time for the report and join procedures. Sequence-1 depicts a situation where a request from receiver-A triggers the delivery of a shared flow. The report and join procedures must finish grafting the tree before the first packet of the shared flow reaches the rendezvous point. Therefore, receiver-A issues a report message as soon as it receives a setup response message. Sequence-2 depicts a situation where receiver-B requests subsequently and receives both shared and patch flows. Using the advanced report and join procedures of sequence-1 could cause a problem in this case because unusable data could be sent to the receiver through a shared flow that precedes the patch flow as a result of advanced grafting. Therefore, receiver-B issues a report message after it receives a play response message and the processing time of the report and join procedures is added to the

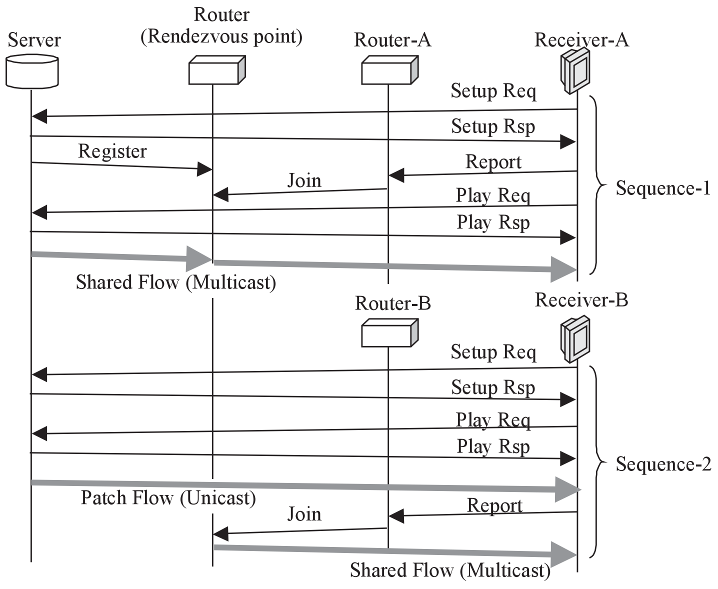

Fig. 5 Sequence between systems.

length of the patch flow.

These considerations do not require any modification to the above protocol's specifications. As an implementation matter they require some improvements at the server and receiver systems. Adding the above slight consideration, our technique can be easily realized without major impact to existing network system.

5.2 QOS Guarantee

To guarantee the quality of real-time streaming transfer, we use a provisioning-based QOS guarantee strategy, Diffserv[25], which guarantees QOS by estimating the statistical traffic for bunch of flows and overinvesting network resources. In Diffserv, EF-PHB (Expedited Forwarding-PHB)[26] is specified as a class to provide the highest QOS (low-delay, low-jitter, low-loss). Network nodes in a Diffserv (DS) domain must guarantee the minimum outgoing rate for such a traffic class, and ingress nodes of a DS domain must constrain incoming traffic not to exceed the minimum outgoing rate. The traffic-control technique that we have described is applied as a means of constraining incoming traffic. The server can control traffic by considering the minimum outgoing rate of EF-PHB as the available bandwidth, ΣA_i, for an on-demand media distribution service on each link along a distribution tree. Thus, our statistical traffic control is very compatible with Diffserv's QOS guarantee strategy (besides, it does not require any changes to Diffserv specifications).

When a fixed available bandwidth is allocated for the service, by using EF-PHB, it is not disturbed by fluctuation of other application's traffic sharing the same network. If we dynamically change the available bandwidth for the service according to fluctuation of other application's traffic, we can realize more optimal network resource utilization. In this case, it requires functions retrieving traffic information and setting EF-PHB configurations at whole network links in real-time. Detail design for these functions including retrieving network topology information is our future study.

6. Numerical Analysis of the Effect of Retrenching Traffic

Based on our traffic-control theory, we can calculate the minimum traffic intensity on trunk and branch links. Here, we consider the maximum effectiveness of traffic retrenching by comparing the traffic intensity with that in a unicast distribution scheme.

At the trunk link, the traffic intensity, $\rho = f(\tau)$ in Eq. (2), takes its minimum value when $\tau = \sqrt{\lambda/2h}$. We get the minimum traffic intensity as,

$$\rho = f\left(\sqrt{\frac{\lambda}{2h}}\right) = \sqrt{2\lambda h} - \frac{1}{2} \quad \text{[erl]} \qquad (10)$$

In unicasting, the traffic intensity on the trunk link is λh. Therefore, the ratio of trunk traffic intensity in our scheme to that in unicasting,

R_{T}, is

$$R_{\mathrm{T}} = \frac{\sqrt{2\lambda h} - 1/2}{\lambda h} \qquad (11)$$

At the branch links, the traffic intensity, $\rho = f(\tau)$ in Eq. (9), takes its minimum value when $\tau \approx \sqrt{\lambda/2mh}$. we get the minimum traffic intensity as,

$$\begin{aligned}\rho &= f\left(\sqrt{\frac{\lambda}{2mh}}\right)\\ &= \left(1 - \left(\frac{m-1}{m}\right)^{\sqrt{2m\lambda h}}\right)\\ &\quad \times \left(\sqrt{\frac{\lambda h}{2m} + \frac{1}{2\lambda h}} - \sqrt{\frac{1}{2m\lambda h}}\right)\\ &\quad + \sqrt{\frac{\lambda h}{2m}} - \frac{1}{2m} \quad [\mathrm{erl}] \qquad (12)\end{aligned}$$

In unicasting, the traffic intensity on the branch links is $\lambda h/m$. Therefore, the ratio of branch traffic intensity in our scheme to that in unicasting, R_{B}, is

$$\begin{aligned}R_{\mathrm{B}} &= f\left(\sqrt{\frac{\lambda}{2mh}}\right) \times \frac{m}{\lambda h}\\ &= \left(1 - \left(\frac{m-1}{m}\right)^{\sqrt{2m\lambda h}}\right)\\ &\quad \times \left(\sqrt{\frac{m}{2\lambda h} + \frac{m}{2\lambda^2 h^2}} - \sqrt{\frac{m}{2\lambda^3 h^3}}\right)\\ &\quad + \sqrt{\frac{m}{2\lambda h}} - \frac{1}{2\lambda h} \qquad (13)\end{aligned}$$

Figure 6 shows the ratio of traffic intensity in our scheme to that in unicasting at the trunk link and branch links ($m = 2, 4, 8$), where the video-content length h was 2, and the request rate λ was 2 to 60. The effect of traffic retrenching increased as the request rate rose. When λ was 60, the ratio was only 12% for the trunk link and 18, 25 and 36% at the branch link (for $m = 2$, 4 and 8 respectively). The effect of traffic retrenching decreased, though, as m is rose. Figure 6 shows the ratio was over 100% at the branch link ($m = 8$) when the request rate was very low. Clearly, it is important in network topology planning to determine the number of branches based on the estimated amount of requests.

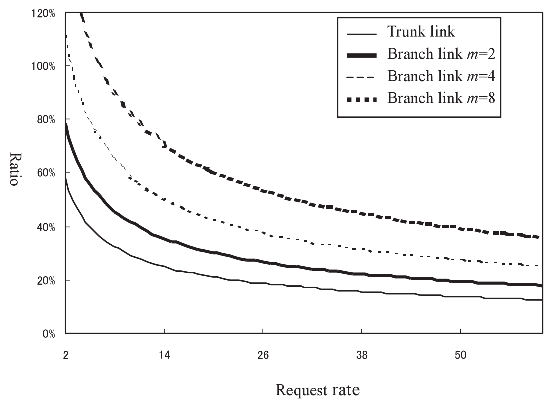

Fig. 6 Ratio of traffic in our scheme to that in unicasting.

7. Simulation

7.1 Simulation Model

We evaluated our traffic-control theory and algorithm through a simulation using a model where a network —— consisting of a video streaming server (or a cache server), routers, and receivers —— forms a balanced tree topology (**Fig. 7**). The reason we use the balanced tree is that our mathematical model premises on equable request rate on each branch link in every particular m. Applying our technique to an asymmetric trees will be our future study. Links between the server and the routers and those between the routers were point-point connections, and links between an edge router and receivers were point-multipoint connections (using a shared media network like Ethernet or Wireless LAN). The same numbers of receivers were connected to each edge router and there was no deviation in the request rate among the edge routers. The server had four video content items (C1–C4) whose length was 2 hours and that were delivered via real-time streaming transfer with a 1-Mbps bandwidth. We assumed that each delivery request for a video content arrive independently (there is no correlation between each request, i.e., requests arrive at random). Therefore, we simply generated requests followed by a Poisson distribution with average value λ. We expressed a trunk link between the server and router as T, branch links that make $m = 2$ as B2, branch links that make $m = 4$ as B4, and branch links that make $m = 8$ and those between the edge routers and receivers as B8.

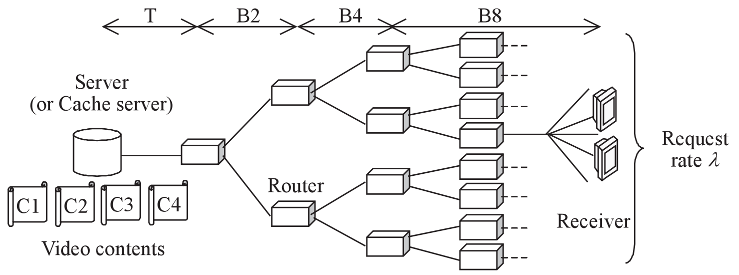

Fig. 7 Simulation model.

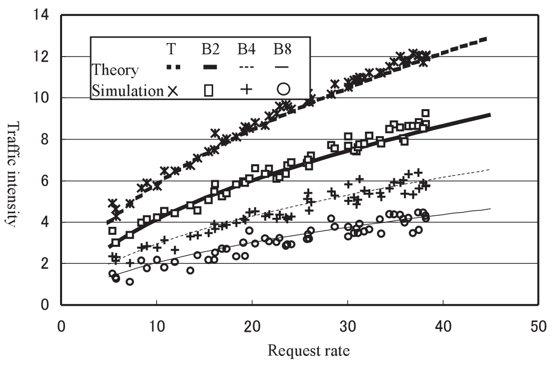

Fig. 8 Measured traffic and theoretical traffic.

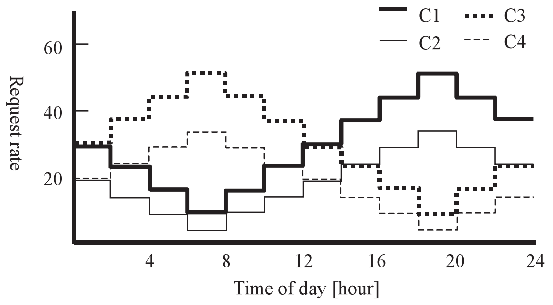

Fig. 9 Transition of average request rate.

7.2 Verifications for a Mathematical Model

First, we verified mathematical models derived in Sections 4.2 and 6. We generated five hundred requests at the same average rate and measured the traffic on each link. We experimented with a fifty average rate of request. In this simulation, the generating rate of the shared flow τ was always set to the value that minimized the traffic on each link. **Figure 8** compares the simulated traffic and the theoretical traffic obtained from Eqs. (10) and (12). The results indicate that both are approximately the same. Our mathematical models are considered as valid.

7.3 Verifications for Traffic Adjustment

Next, we verified that the traffic was adjusted to required bandwidth design. In this simulation, the average rate of requests for C1–C4 were allowed to fluctuate individually according to the time of day (**Fig. 9**). The request rates for C1 and C2 were lowest (10 and 5 requests/hour, respectively) between the 6th and 8th hour and highest (50 and 35 requests/hour, respectively) between the 18th and 20th hour. The request rates for C3 and C4 were highest (50 and 35 requests/hour, respectively) between the 6th and 8th hour and lowest (10 and 5 requests/hour, respectively) between the 18th and 20th hour.

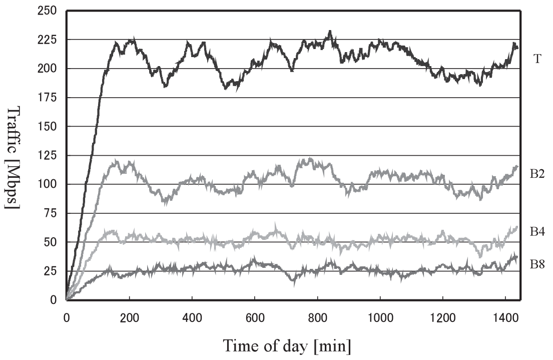

Fig. 10 Traced traffic 1.

Figures 10, **11**, and **12** show the traced traffic on each link type (T, B2, B4, and B8) when C1–C4 were delivered using our traffic-control at the above request rates.

In Fig. 10, unlimited bandwidth was available on all links (T, B2, B4, and B8) for all C1–C4 deliveries. In this case, the generation rate of the shared flow τ for each video content was set to the same value as each request rate λ. As a result, all content was delivered by shared flow; that is, the traced traffic was identical to the traffic by unicasting. In Fig. 9, the total request rate for C1–C4 was always 110 requests/hour and the content length was 2 hours, so the traffic on the T, B2, B4, and B8 links was theoret-

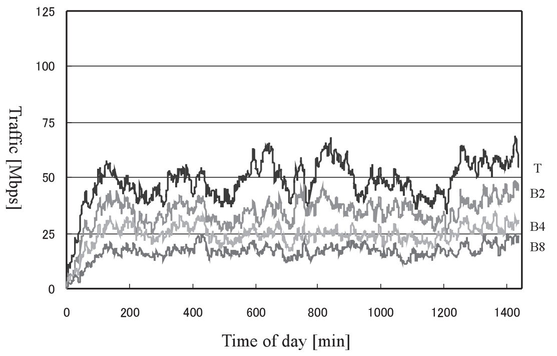

Fig. 11 Traced traffic 2.

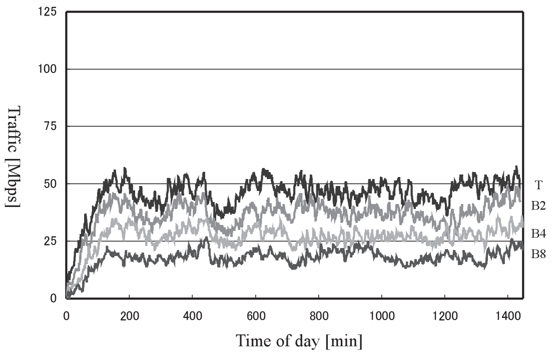

Fig. 12 Traced traffic 3.

ically 220, 110, 55, and 28 Mbps, respectively. Our simulation results were close to these values.

In Fig. 11, the available bandwidth on B4 and B8 links for all C1–C4 deliveries was limited to 30 and 20 Mbps, respectively. In this case, the B4 and B8 links became bottleneck links and the server adjusted the traffic to avoid exceeding the available bandwidth by controlling each τ. Initially, the same available bandwidth was assigned to C1–C4, and soon it was reassigned according to each request rate. In practice, the traced traffic on the B4 and B8 links was about 30 and 20 Mbps, respectively. In Fig. 12, the available bandwidth on T and B2 for all C1–C4 deliveries was limited to 50 Mbps. Here, T became the bottleneck link and the server adjusted traffic to avoid exceeding 50 Mbps on T. In practice, the traced traffic on T was about 50 Mbps.

7.4 Effectiveness of Dynamic Bandwidth-allocation and Traffic-adaptation

Furthermore, we confirmed an effectiveness of the dynamic bandwidth-allocation and traffic-adaptation algorithm from view of the blocking rate of delivery. In this simulation, the request rates were again generated according to Fig. 9. The available bandwidth on the T and B2 links was decreased from 100 to 50 Mbps in 1-Mbps steps while that on the B4 and B8 links was decreased from 50 to 25 Mbps in 0.5-Mbps steps. When the server received a request, it checked whether the resulting traffic would exceed the available bandwidth. If the server found that the traffic would exceed the available bandwidth on any link, it rejected the request (i.e., the delivery was blocked). The simulation was done with and without dynamic bandwidth assignment (**Fig. 13**). The dynamic bandwidth assignment clearly reduced the blocking rate. For a blocking rate of less than 0.1%, without the dynamic bandwidth assignment we would need at least 92 Mbps of available bandwidth on the T and B2 links and 46 Mbps on the B4 and B8 links; with the dynamic bandwidth assignment, the necessary bandwidth would be 77 Mbps on the T and B2 links and 38 Mbps on the B4 and B8 links.

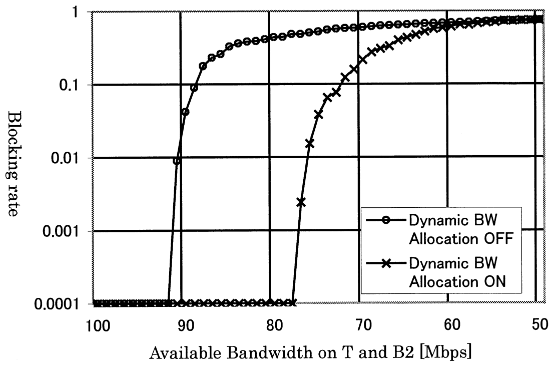

Fig. 13 Blocking rate of delivery.

7.5 Remarks

Our simulation results indicated the following.

First the simulation results confirmed the correctness of mathematical models derived in Sections 4.2 and 6. This suggests our theory can reduce traffic as shown in Fig. 6. In the second simulation, our technique with the theory demonstrated reduced substantial traffic on each link compared to unicasting and correctly adjusted the traffic to the designed bandwidth. We can see that the traced traffic was adaptively controlled to the bandwidth of bottleneck links. Third the simulation result proved the effectiveness of dynamic bandwidth-allocation. It enables network design with fewer bandwidth resources needed to satisfy a particular blocking rate requirement.

8. Conclusion

We presented an optimal scheme for on-demand video distribution through asynchronous multicasting. We call this 'Asynchronous Media Casting Network.' Main contributions in this paper are the following:

First, a new traffic-control theory was presented, which can make traffic adapt bandwidth design requirement throughout a network while striving to reduce the use of a receiver's buffer memory. This theory is on the basis of a simple patching technique. Second, dynamic bandwidth-allocation and traffic-adaptation algorithm was presented, which can achieve more effective distribution among the various video deliveries by applying the above theory. Third, we could show that these techniques can easily be implemented as an autonomous network by using the latest IP protocol that has been put into practical use. Simulation results supported the validity of our mathematical model and indicated that the dynamic bandwidth-allocation and traffic adaptation algorithm could reduce the service-blocking rate of video delivery. For example, the algorithm lessened 17% bandwidth resource for a blocking rate of less than 10^{-4} in the four kind of video contents distribution where each video length is 2 hours, transmit rate is 1 [Mbps] and total request rate is constantly about 100 [request/hour] (but each request rate independently fluctuates).

As broadband networks continue to grow, high-quality video delivery services are highly expected and further CDN are developed. In particular, we are interested in wireless-based CDNs. Our techniques contribute highly to such a wireless environment, where bandwidth resources will be strictly limited and often fluctuated due to environmental conditions.

As future work, we are planning to enhance our asynchronous media casting to make it more suitable for wireless and mobile networks. We must further study the following:

We need to develop our theory into various types of topologies (no balanced tree topology). It also needs consideration of ways to construct a logical tree in an autonomous manner. In addition, we must describe a more detailed protocol and make a prototype for a practical use.

References

1) Miles, P.: Internet World Guide to Webcasting, John Wiley & Son (1998).
2) Tashiro, S. and Nishikado, N.: State of the Art Content Distribution Technology, *IPSJ Magazine*, Vol.42, No.11 (2001).
3) 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).
4) Golubchik, L., Lui, J. and Muntz, R.: Adaptive Piggy-back: A Novel Techniques for Data Sharing in Video-On-Demand Storage Servers, *ACM Multimedia System Journal*, Vol.4, No.3, pp.140–155 (1996).
5) Woo, H. and Kim, C.K.: Multicast scheduling for VOD services, *Multimedia Tools and Applications*, Vol.2, No.2, pp.157–171 (Mar. 1996).
6) 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).
7) Uno, S., Tode, H. and Murakami, K.: Simple and Efficient Video-on-Demand Scheme with Segment Transmission over High Speed Network, *IEICE Trans. Comm.*, Vol.E84-B, No.1, pp.106–115 (2001).
8) 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).
9) Cater, 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).
10) 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. (Sep. 1998).
11) Cai, Y., Hua, K.A. and Vu, K.: Optimizing Patching Performance, *Proc. Multimedia Computing and Networking 1999*, San Jose, CA (Jan. 1999).
12) 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).
13) Eager, D.L., Vermon, 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).
14) Eager, D.L., Vermon, 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).
15) Eager, D.L., Vermon, 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).
16) Zhao, Y., Eager, D.L. and Vermon, M.K.: Network Bandwidth Requirement for Scalable On-Demand Streaming, *Proc. 21th Annual Joint Conf. IEEE INFOCOM 2002*, New York, NY (Jun. 2002).
17) Sato, K. and Katsumoto, M.: A proposal for personalized media stream delivery, ISPJ Report, DPS103-14 (Jun. 2001).
18) 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).
19) Sato, K. and Katsumoto, M.: A proposal and an evaluation of the bandwidth control method for on-demand media delivery using the asynchronous multicast, ISPJ Report, DPS107-26 (Jun. 2001).
20) Fenner, W.: Internet Group Management Protocol, version 2, RFC2236 (1997).
21) Estrin, D., Farinacci, D., et al.: Protocol Independent Multicast-Sparse Mode, RFC2362 (1998).
22) Ballardie, A.: Core Based Trees (CBT version 2) Multicast Routing, RFC2189 (1997).
23) Holbrook, H. and Cain, B.: Source-Specific Multicast, IETF Internet Draft (Nov. 2001).
24) Schulzrinne, H., Rao, A., et al.: Real Time Streaming Protocol, RFC2326 (1998).
25) Blake, S., Black, D., et al.: An Architecture for Differentiated Services, RFC 2475 (1998).
26) Jacobson, V., Nichols, K., et al.: An Expedited Forwarding PHB, RFC2598 (1999).

(Received February 13, 2003)
(Accepted September 5, 2003)

Editor's Recommendation

This paper proposes a new control scheme for minimizing the consumption of network resources of on-demand streaming-video distribution through multicasting. The paper was given the best paper award at the 10th DPS workshop. After careful discussion, the program committee of the workshop has decided to recommend this paper to the Journal of IPSJ.

(Chairman of SIGDPS Teruo Higashino)

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 have been working for Japan Radio Co., Ltd and developed Frame Relay router, ATM switch, 3G wireless base station and Multi-layer switch etc. Since 2002 he has been a visiting researcher at Communication Research Laboratory and working toward the Ph.D. in The University of Electro Communication. He is a member of IPSJ, IEEE Computer 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 Communications Research Laboratory. 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.

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.