QoS Control by Traffic Engineering in Content Delivery Networks

● Hitoshi Yamada ● Akiko Okamura ● Koji Nakamichi ● Kiyoshi Sakai
● Akira Chugo

(Manuscript received November 4, 2003)

Traffic engineering (TE) is a method of managing and controlling IP networks efficiently. This paper proposes a dynamic TE architecture that provides a QoS (Quality of Service) guaranteed service in addition to an existing best-effort service in a content delivery network (CDN). This architecture selects the optimum route and server for a user request based on the load of the network and content server. It aims to guarantee the QoS and effectively use network resources by dynamic load balancing. Also in this paper, we present the results of a simulation of the effects of our proposed method.

1. Introduction

With the increasing importance of the Internet as a commercial infrastructure and the increasing need for more functionality, the operation of the IP network is becoming more important and complex than ever. This situation makes operators aware of the importance of traffic engineering (TE).[1] We propose a dynamic TE concept that aims at automatic network optimization.[2]

This paper considers the use of the dynamic TE architecture in an intelligent content delivery network (CDN) system. Generally, CDNs contain a number of content servers and route user requests to the nearest or lowest-load server to quicken the response. Currently, conventional CDNs provide only best-effort services; however, in the future, Internet service providers (ISPs) will need to provide not only best-effort services but also QoS (Quality of Service) services to attract more users. Furthermore, it will be necessary to use network and server resources as efficiently as possible. To meet these future requirements, after receiving a request from a user, our dynamic TE architecture selects an optimum route and server based on the network congestion status and load of the content servers. Also, it provides guaranteed services using a QoS path-selection mechanism and utilizes network resources based on dynamic load balancing.

This paper introduces some state-of-the-art traffic control methods and then describes the architecture and control mechanisms we have developed. Next, it describes an end-to-end error reduction/recovery method for continuous media in best-effort environments. Then, we compare the simulated effects of our method with two conventional route and server selection methods. Lastly, we present some evaluation results of dynamic load balancing and our error recovery method.

2. Related work

Generally, CDNs have a server selection mechanism that selects a server in the network to satisfy user requests. Such services are sometimes called "anycast" services. A well-known method used in anycast services is to route a user request to the nearest server using a domain name service (DNS) mechanism. Another method is to measure the servers' loads and select the lowest-load server. In other cases, the network estimates the round trip time (RTT) or available bandwidth along the route to the server and then selects the server that provides the shortest transfer delay based on the size of the content.[3)]

These server selection methods are not always effective because, although some of them indirectly consider the network load, none of them directly select a route to the server. Even if the network load is considered by measuring the RTT, flexible control is not possible because the route itself is not selected. Furthermore, there is no QoS guarantee mechanism in most existing CDNs. Reference 4) proposes a CDN architecture that considers QoS. However, since this approach does not consider route control, it cannot resolve the problem of low network resource usage.

Consequently, we consider that a network must have a route control mechanism for QoS control and ensure efficient use of network resources. TE can provide such a route control mechanism.

General TE methods search for a route having the requested bandwidth based on bandwidth information such as the available bandwidth or reserved bandwidth (QoS routing). A typical QoS routing algorithm selects the route with the minimum number of hops between the source and destination node from among the links having the requested bandwidth (shortest path). Another algorithm selects the route that has the maximum available bandwidth; if there are multiple routes, it selects the minimum-hop route (a.k.a. the shortest-widest path). Unlike server selection mechanisms, these methods take QoS path selection into consideration and do not consider best-effort traffic.

As mentioned above, there are server selection mechanisms and route selection mechanisms. Their targets are, respectively, best-effort traffic and QoS traffic, but there is no method that treats both these targets together.

The evaluation results presented in this paper show that the simple combination of an existing server selection mechanism and route selection mechanism cannot achieve good performance. Therefore, we propose a dynamic TE architecture that not only enables selection of the optimum server and route by considering QoS, but also makes it possible to prevent a degradation of best-effort traffic caused by QoS traffic in a CDN.

3. Architecture

3.1 Overview

Figure 1 shows the network architecture of the CDN based on the MPLS (Multiprotocol Label Switching) platform. Multiple content servers, including cache servers, are connected to edge routers, and contents are distributed among these servers. End users request contents in a QoS-guaranteed service or best-effort service; in the case of the QoS-guaranteed service, users request a service with the required quality, for example, a certain bandwidth. One example of a QoS-guaranteed service is a streaming service in which there is a basic charge on the contents and users pay an extra fee for QoS guarantee.

The network is controlled and managed by a control server called the TE server. The TE server accepts requests for contents from end users and selects the appropriate content servers and routes. It aims to optimize a congested network by 1) selecting the optimum server and route using proactive load balancing and 2) performing dynamic rerouting using reactive load balancing.

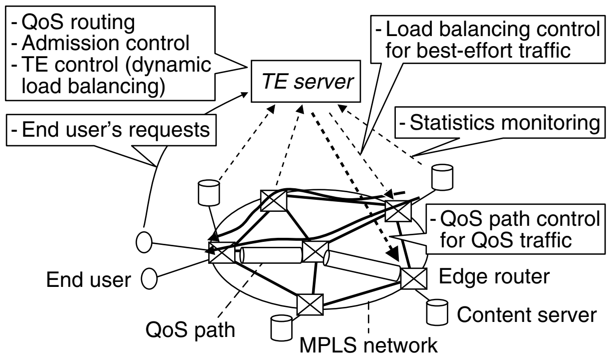

Figure 1
Network architecture.

3.2 Mechanism

We will now describe the control mechanism for QoS-guaranteed service traffic and best-effort service traffic.

3.2.1 QoS-guaranteed service

Here we consider the bandwidth as the QoS metric. The network provides a QoS-guaranteed service by using bandwidth-guaranteed paths (QoS paths) on MPLS.

The TE server periodically collects network statistics, including the load of each content server, by using a protocol such as SNMP (Simple Network Management Protocol). It also manages the available-bandwidth information of each link in the network.

When the TE server receives a request from an end user with QoS parameters such as the bandwidth, it searches for the optimum server and route to satisfy the user's request based on the network and server load balancing. The selection algorithm is described in the next section. If there is no server or route that satisfies the requested quality, the TE server notifies the user that the request has been rejected. When a server and a route from the sever-side edge router to the user-side edge router are selected, the TE server issues a request to set a bandwidth-guaranteed path along the selected route to the sever-side edge router. Then, the router sets up the path using the signaling protocol. After setting up the QoS path, the TE server redirects the user-side request to the selected content server and the user receives the content at the requested quality from the content server.

The above sequence describes per-flow based behavior; that is, a QoS path is set for each user request. However, flows should be aggregated when we consider the scalability issue. In the case of flow aggregation, a server and route are selected to accommodate multiple user flows, and requests for the same content from the same user-side edge router are aggregated into the same path. If we increase the aggregation granularity, we can improve scalability by reducing the number of paths in the network while reducing the optimization level.

3.2.2 Best-effort service

1) Load balancing

Existing server selection methods such as nearest server selection and lowest-load server selection can be used to meet content requests in a best-effort service. Best-effort traffic is forwarded along the shortest path without bandwidth guarantee. Best-effort traffic is given a lower priority than QoS traffic, so best-effort traffic can be discarded in locations where there is a large amount of QoS traffic or there is congestion due to a concentration of best-effort traffic. To improve the performance of best-effort traffic, congestion avoidance control is required. We have developed a dynamic load balancing method to meet this requirement.

Between each edge router, paths of the shortest route (we call them default paths) are set for best-effort traffic. The TE server periodically collects link statistics such as the link utilization and packet loss by using SNMP and observes the network's load status. When the TE server detects or predicts congestion in a network, it searches for the route that has the highest available bandwidth between the edges that use the path through the congested (or predicted to become congested) links and sets up a detour path along the selected route. Then, a portion of the traffic is dynamically rerouted from the default path to the detour paths to eliminate the congestion. The algorithm of our load balancing method is described briefly in the next section.

2) End-point QoS control

Even though some congestion avoidance control is performed at the network management level, there is still a possibility of packet loss in the best-effort network. Therefore, end-point QoS control, including reduction or recovery from packet losses, is a key issue in continuous media services such as audiovisual streaming. In these services, UDP (User Datagram Protocol) based transport protocol in combination with RTP/RTCP (Realtime Transport Protocol/RealTime Control Protocol)[5] is generally used. To realize packet loss reduction or recovery in the above protocol stack, the following options have been widely studied and implemented as the end-point QoS control.[6),7)]

- Forward error correction (FEC)

FEC methods send redundant data as parities to correct lost or corrupted packets at the receiver side (**Figure 2**). Generally, increasing the amount of parities improves the performance of recovery; however, at the same time it increases the required bandwidth between end points. FEC methods do not require return paths and are attractive for delay-sensitive applications such as interactive audio, remote live camera control, and VOD (Video On Demand) services with trick-play functionality. To maximize the effect of the FEC method, the transmission group (TG: a set of packets and associated parity packet) should be carefully designed, taking into account the size of the application data unit (ADU).

- Automatic repeat request (ARQ)

ARQ is a kind of retransmission protocol that typically uses a probing protocol such as RTCP (Real-Time Control Protocol) to recover lost or corrupted packets (**Figure 3**). In principle, this scheme introduces additional end-to-end delay and should be applied with care, especially in delay-sensitive applications. In addition, too much retransmission of lost packets may increase the end-to-end bandwidth, which may accelerate congestion.

3.3 Proposed architecture

We will now describe our algorithm for optimum server and route searching for QoS-guaranteed services and our algorithms for dynamic load balancing and end-point QoS control for best-effort services.

3.3.1 *Optimum server and route search algorithm*

The basic idea is to allocate each link and server a cost value that is based on its load (a high cost value indicates a high relative load). Then, we find the route with the lowest-cost intermediary links and server; that is the route having the maximum available bandwidth. In this way we achieve load balancing, because high-cost links and servers are unlikely to be selected.

We define the cost of a link between router *i* and an adjacent router *j* as follows:

$$link\_cost_{qos}(i, j) = 1/(R\_MAX_{ij} - R_{ij}), \qquad (1)$$

where *R_MAX* denotes the maximum reservable bandwidth and *R* denotes the reserved bandwidth of the link.

Next, we calculate the cost of using a particular server in terms of the server's current and maximum loads. To do this, we can use one or more load parameters, for example, the CPU load,

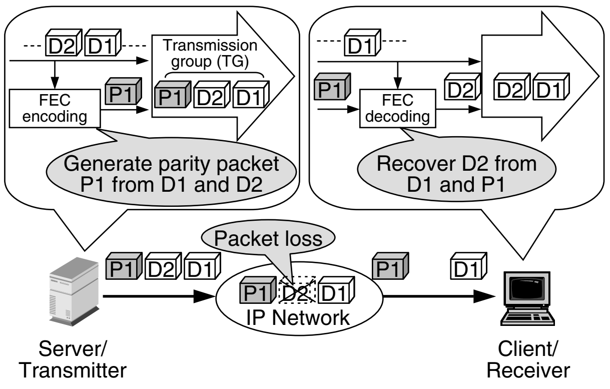

Figure 2
Operation of FEC.

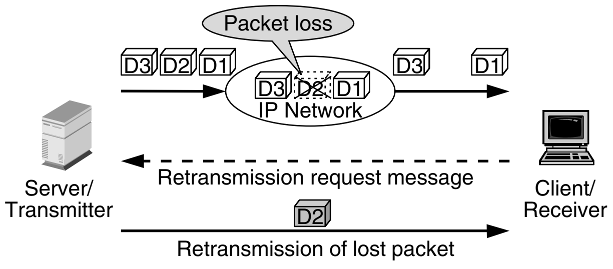

Figure 3
Operation of ARQ.

memory consumption, disk access load, number of user connections, and traffic sending rate. In the case of a streaming server, for example, its bottleneck is generally the IO or NIC, rather than the CPU load; therefore, we can define the cost of a server s as follows:

$$server\_cost_{qos}(s) = 1/(S\_MAX_s - S_s), \qquad (2)$$

where S_MAX denotes the maximum sending rate (i.e., the NIC rate) and S denotes the current sending rate.

After determining the cost of each server and link, we find the route with the lowest-cost intermediary links and server. Let t denote the route from the sever-side edge router to the user-side edge router. Then:

$$total\_cost_{qos}(s,t) = server\_cost_{qos}(s) + \sum_{(i,j)\in t} link\_cost_{qos}(i,j). \qquad (3)$$

Next, we select the server s and route t whose *totol_cost(s,t)* value is minimum by using Dijkstra's algorithm. This process achieves server load balancing and network load balancing at the same time.

3.3.2 Dynamic load balancing algorithm

Dynamic load balancing sets up multiple paths between edge routers when there is congestion and avoids congestion by splitting the IP flows into multiple paths. The following control is performed on each edge router pair.

1) Congestion detection

The TE server detects congestion on the path set up between edge router pairs. Specifically, it decides that the link is congested if the utilization of the link exceeds a threshold value for a specific amount of time. If the TE server can distinguish the best-effort traffic rate from the QoS traffic rate statistics information (e.g., when the router provides the MIB [Management Information Base] containing the per-LSP [Label Switched Path]), we can distinguish traffic in the best-effort path from traffic in the QoS path and predict the traffic rate to some extent. That is, we can predict when bandwidth that is reserved for QoS traffic but not currently in use is likely to be used in the near future. Therefore, we can predict whether congestion caused by an increase in QoS traffic will be sufficient to justify discarding of best-effort traffic and thereby avoid unnecessary discarding.

We denote the best-effort traffic rate by B, the QoS traffic rate by G, and the physical link bandwidth (capacity) by C. If we can distinguish B from G, we can define the predicted link utilization as follows:

$$link\_util_{be}(i,j) = (B_{ij} + R_{ij}) / C_{ij}, \qquad (4)$$

where R denotes the reserved bandwidth at the link. Otherwise, we define:

$$link\_util_{be}(i,j) = F_{ij} / C_{ij} \qquad (5)$$

by observing that $F = B + G$.

2) Detour route search

If congestion is detected on a path, the TE server searches for an alternative route to avoid congestion. The cost of each link is defined as follows:

$$link\_cost_{be}(i,j) = 1/(1 - link\_util_{be}(i,j)), \qquad (6)$$

and a search is made to find the minimum-cost route.

3) Load balancing calculation

Traffic is distributed per flow between multiple paths based on the load of each path. Each incoming packet to the ingress router is dispatched to a path based on the hash value of the IP header information. Hash values are divided into groups by hash boundaries, with each group corresponding to a path. The TE server controls the traffic rate of each path to average the link utilization in the network. It does this by indicating the ingress routers to adjust the hash boundaries. A more detailed algorithm can be seen in Reference 8).

3.3.3 End-point QoS control

The basic idea of end-point QoS control is to

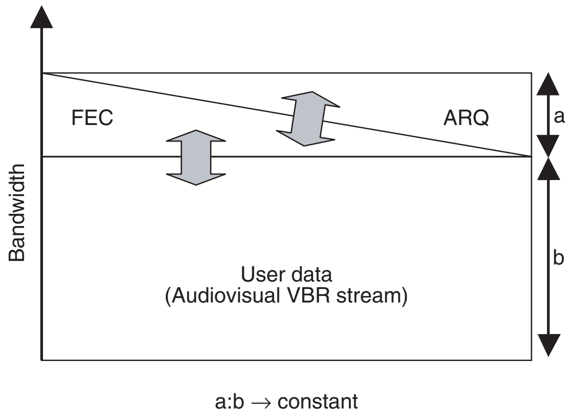

Figure 4
Bandwidth allocation control in proposed method.

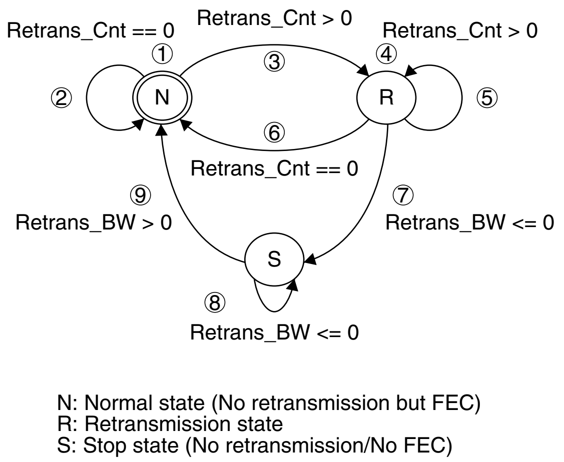

Figure 5
State transition in proposed method (Transmitter).

share the bandwidth for FEC and ARQ and adaptively switch between these two methods according to the end-to-end network condition probed by RTCP (**Figure 4**). For detailed QoS control, we developed a new algorithm based on the IETF (Internet Engineering Task Force) RFCs (Request for Comments) and Internet Drafts.$^{9)-11)}$ We will now describe the state control of the algorithm (**Figure 5**):

① The maximum available bandwidth for retransmission (Retrans_BW) is allocated to FEC. Retrans_BW is reset to an upper bound value at a certain timing.

② FEC packets are sent at a predefined frequency if the retransmission process is not activated.

③ If the retransmission process is activated, the retransmission counter value (Retrans_Cnt) is incremented by 1 and Retrans_BW is decremented by the used bandwidth.

④ If Retrans_Cnt is more than zero, the insertion frequency of FEC packets is reduced.

⑤ Operation ④ is continued until Retrans_Cnt reaches zero.

⑥ If Retrans_Cnt reaches zero, the insertion frequency of FEC packets is reset to the original value.

⑦ If Retrans_BW goes below zero, FEC and the retransmission operation are stopped and the value of Retrans_BW is reset.

⑧ FEC and the retransmission process remain stopped until Retrans_BW is reset.

⑨ When Retrans_BW is reset to an upper bound value, FEC packets are sent at a predefined frequency. Retrans_BW is reset at a predetermined cycle to maintain the ratio of user data to error protection data (FEC and ARQ packets). This ratio is shown as "a:b" in Figure 4.

4. Evaluations

4.1 Effects of route search algorithms

We evaluated our server and route selection algorithm for QoS-guaranteed services using simulations. Then, we compared our method with the following two methods that can be implemented using existing technology:

1) LSL (Lowest Server Load): This method selects the server with the lowest load then selects the minimum-cost route to the server.
2) DNS: This method selects the nearest server then selects the minimum-cost route to the server. The nearest server is selected according to the location of the user-side edge, which is shown by the dashed line in **Figure 6**.

4.1.1 Model

We used the 19-router ISP network model$^{12)}$ shown in Figure 6. The network consists of 155 Mb/s and 45 Mb/s links. There are four servers, and the edge routers connected to them are shown as black circles in Figure 6. End users connect with the other edge routers and request QoS paths with a bandwidth requirement. At each request, the user-side edge is selected at random and a requested bandwidth from 1 to 10 Mb/s is also selected at random. We assume the server capacity (i.e., the maximum sending rate) is 500 Mb/s. For simplicity, we assume that QoS paths are not released from bandwidth reservation.

4.1.2 Results

Figure 7 shows the number of accepted requests versus the number of requests. As the figure shows, our proposed method can accept a large number of requests.

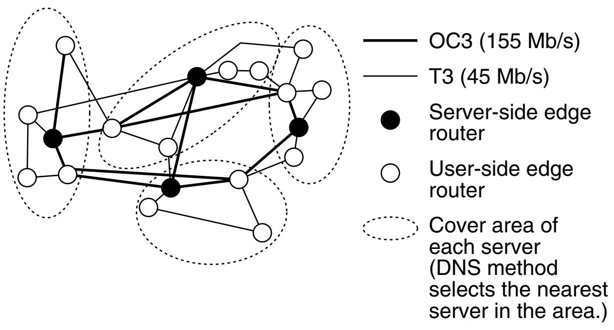
OC3 (155 Mb/s)
T3 (45 Mb/s)
Server-side edge router
User-side edge router
Cover area of each server (DNS method selects the nearest server in the area.)

Figure 6
ISP network model.

In the DNS method, user requests are rejected when there is no available route to the nearest server, even if routes to another server are available. In the LSL method, the server with the lowest load is selected first, even if the route to the server becomes long. Therefore, LSL tends to consume a lot of network resources for link bandwidth unnecessarily, which limits the number of accepted requests.

On the other hand, our proposed method will reject a low-load server with a long route in favor of a medium-load server with a short route. Moreover, in our method, when there is a short route to a server and the server's load is high, that server and route are unlikely to be selected. In this way, our method flexibly selects the appropriate server and route according to the server and network conditions, which reduces the probability that a request will be blocked.

Figure 8 shows the average number of hops versus the number of accepted requests. Our method keeps the number of hops small and uses network resources efficiently. The LSL method tends to select long routes, as we described earlier. It is interesting that even the DNS method, which selects the nearest server, suffers an increase in the number of hops when the number of

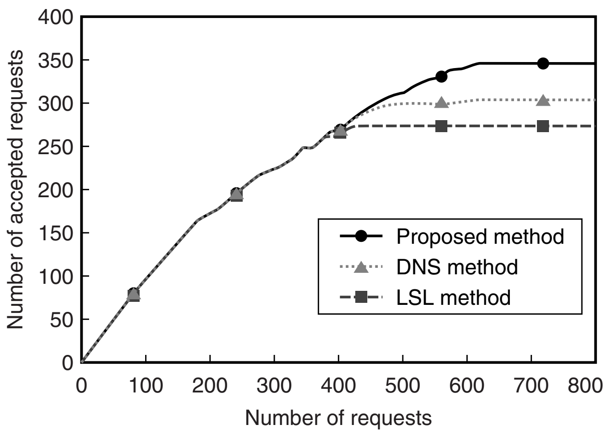

Figure 7
Comparison of number of accepted requests.

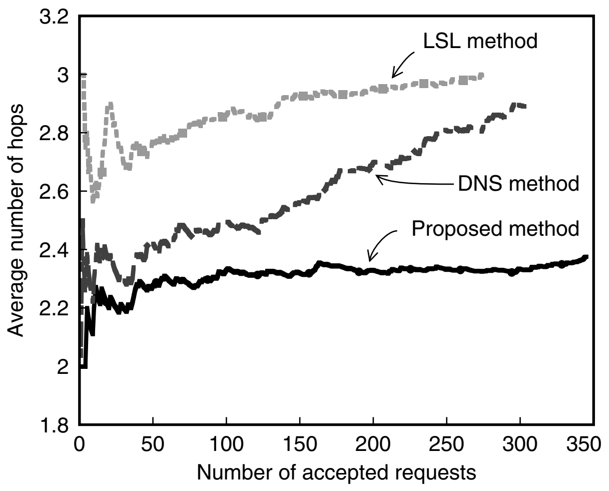

Figure 8
Comparison of average number of hops.

requests becomes large. This can be explained as follows: if the link utilization of the shortest route to the nearest server becomes high, a longer detour route is likely to be selected. DNS selects a longer route even if there is a shorter route to another server; therefore, the number of hops in DNS is larger than in our method.

The average link utilization in the network after receiving 800 requests was 43.6% with our method, 50.3% with LSL, and 49.2% with DNS. Our method can achieve a lower link-load than the other methods even if it accepts more requests. Therefore, our method optimizes the network to achieve efficient utilization of resources.

4.2 Effect of dynamic load balancing

Next, we show that dynamic load balancing (D-LB) can prevent a reduction of the best-effort traffic rate under a time-varying QoS traffic rate. For simplicity, we consider the flow of traffic between two points with two connecting paths and assume a link capacity of 100 Mb/s, a QoS maximum reservable bandwidth of 80 Mb/s, and an input best-effort traffic rate of 50 Mb/s.

QoS flows are added every 5 seconds with a reserved bandwidth of 6 Mb/s. We assume that the traffic rate of each QoS flow varies sinusoidally. We consider the definitions of link utilization given in Equations (4) and (5) and assume that dynamic load balancing begins to work after link utilization exceeds 80% for 30 seconds.

Figure 9 shows a simulation of the throughput versus time. Without D-LB, the best-effort (BE) rate is degraded when the QoS rate exceeds 50 Mb/s. However, with D-LB, the best-effort rate remains almost constant at 50 Mb/s. In the case of D-LB1, in which link utilization is based on the sum of the best-effort rate and QoS rate as in Equation (5), the throughput decreases for a while because there is a time lag between the start of congestion and traffic rerouting. On the other hand, in the case of D-LB2, in which congestion is predicted based on the reserved bandwidth using Equation (4), congestion is avoided in advance so there is no degradation of throughput.

4.3 Effect of our end-point QoS control scheme

To evaluate the effect of the proposed error recovery method, we built a testbed consisting of a realtime MPEG-2 (Moving Picture Experts Group phase2) encoder/decoder connected to the B-flets access network (**Figure 10**). We used this testbed to observe the packet loss characteristics under the following conditions:

1) Audiovisual coding: MPEG-2 VBR (average 3 Mb/s)
2) ADU size: 1400 bytes
3) Interval of FEC packets: 4
4) Maximum retransmissions per packet: 3
5) Overall end-to-end delay (maximum buffering at the decoder): 300 ms
6) End-to-end hop count: 14
7) Bandwidth allocation (Value of a:b in Figure 4): 1:4

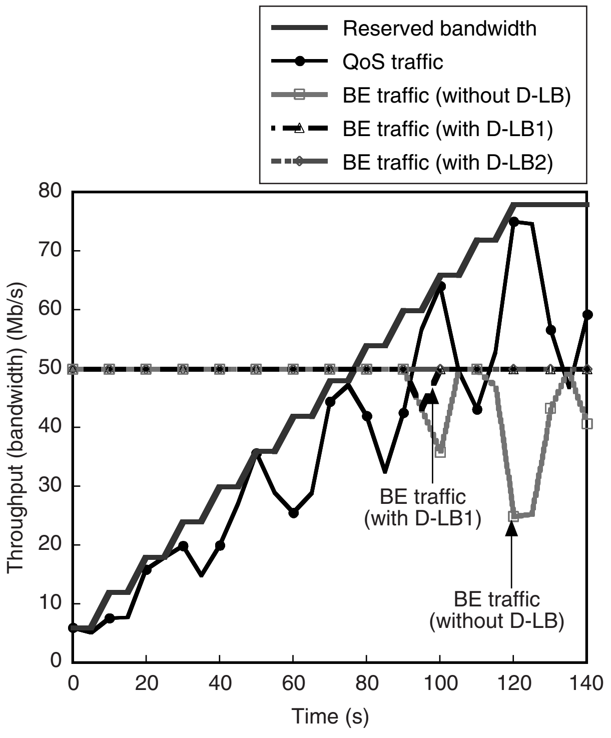

Figure 9
Effect of dynamic load balancing.

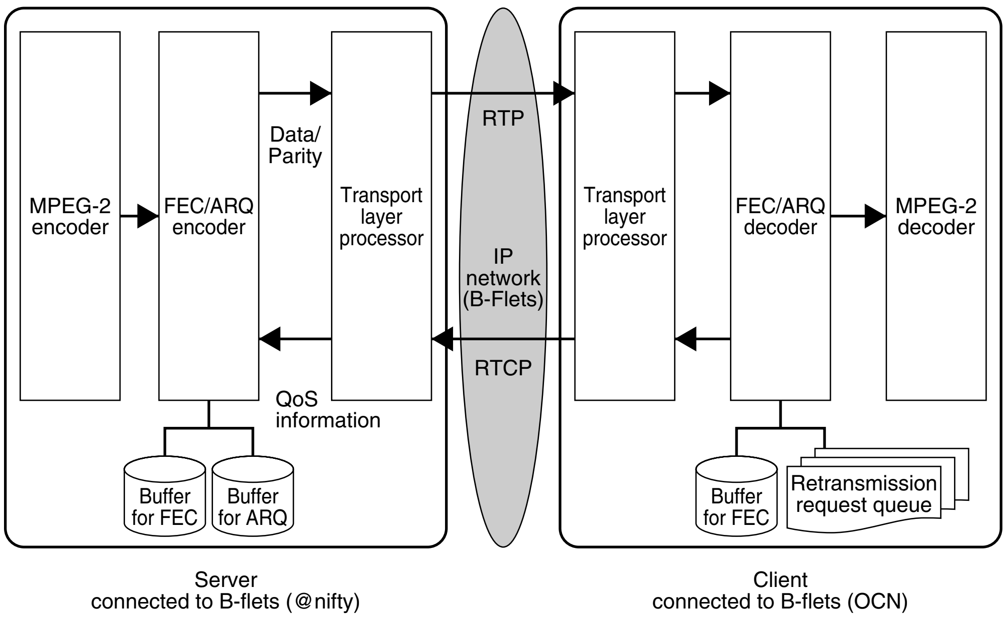

Figure 10
Block diagram of testbed for error recovery.

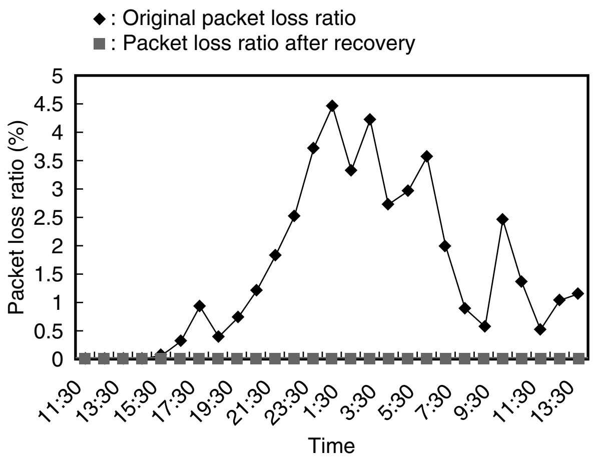

Figure 11
Effect of proposed error recovery method.

Figure 11 shows the packet loss ratio with and without our proposed method for a period of 27 hours. The original packet loss ratio reached 4.5%, but is dramatically reduced to 0.023% with our method.

5. Conclusion

In this paper, we proposed an architecture and method for dynamic TE. Dynamic TE aims at effective coexistence of QoS-guaranteed traffic and best-effort traffic by using an optimum server and route selection algorithm based on server and network loads and dynamic load balancing. Our optimum server and route selection algorithm achieved a low blocking probability and highly efficient resource utilization. Then, we showed that dynamic load balancing prevents a reduction of the best-effort traffic rate under a time-varying QoS traffic rate. We also proposed an end-to-end error reduction/recovery method for continuous media in best-effort environments. When the number of routers, servers, and requests become big in a large-scale network, we have to consider the calculation and control times and the number of paths in the network. We will study and evaluate these scalability issues in detail to realize our dynamic TE architecture.

References

1) D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao: Overview and principles of internet traffic engineering. IETF, RFC3272, 2002.
2) K. Takashima, H. Yamada, K. Nakamichi,

and A. Chugo: Dynamic traffic engineering: creating new services. 4th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT2001), 2001, p.238-241.

3) R. L. Carter and M. E. Crovella: Server selection using dynamic path characterization in wide-area networks. Proceedings of IEEE INFOCOM'99, 1997, p.1014-1021.

4) F. Hao, E. W. Zegura, and M. H. Ammar: QoS routing for anycast communications: Motivation and an architecture for diffserv networks. *IEEE Communications Magazine*, **40**, 6, p.48-56 (2002).

5) IETF RFC1889, RTP: A Transport Protocol for Real-Time Applications.

6) G. Carle and E. W. Biersack: Survey of Error Recovery Techniques for IP-based Audio-Visual Multicast Applications. *IEEE Network Magazine*, **11**, 6, p.24-36 (1997).

7) C. Perkins, O. Hodson, and V. Hardman: A Survey of Packet Loss Recovery Techniques for Streaming Audio. *IEEE Network Magazine*, **12**, 5, p.40-48 (1998).

8) K. Nakamichi, H. Yamada, and A. Chugo: Dynamic traffic engineering under different traffic patterns. 16th International Workshop on Communications Quality & Reliability (CQR2002), 2002, p.99-103.

9) IETF RTF Draft: Extended RTP Profile for RTCP-based Feedback.

10) IETF RFC2733: An RTP Payload Format for Generic Forward Error Correction.

11) IETF RFC Draft: RTP Payload Formats to Enable Multiple Selective Retransmissions.

12) Q. Ma and P. Steenkiste: On path selection for traffic with bandwidth guarantees. Proceedings of International Conference on Network Protocols (IEEE ICNP97), 1997, p.191-202.

Hitoshi Yamada received the B.E. and M.E. degrees in Electrical Engineering from the University of Tokyo, Tokyo, Japan in 1997 and 1999, respectively. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1999, where he has been engaged in research and development of network architecture and traffic control of IP networks. He is a member of the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan.

Kiyoshi Sakai received the B.E. degree in Electrical Engineering from Waseda University, Tokyo, Japan in 1985. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1985, where he has been engaged in research and development of signal processing for audio-visual systems, mainly video compression algorithms and their applications.

Akiko Okamura received the B.S. and M.S. degrees in Physics from Ochanomizu University, Tokyo, Japan in 1993 and 1995, respectively. She joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 2001, where she has been engaged in research and development of service control and traffic control of IP networks. She is a member of the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan.

Akira Chugo received the B.E. and M.E. degrees in Electrical Engineering from Waseda University, Tokyo, Japan in 1981 and 1983, respectively. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1986, where he has been engaged in research and development of high-speed data network systems, mainly in the areas of hardware-oriented routers and traffic control.

Koji Nakamichi received the B.E. and M.E. degrees in Material Engineering from Yokohama National University, Yokohama, Japan in 1992 and 1994, respectively. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1994, where he has been engaged in research and development of traffic control and resource management of ATM and IP networks. He is a member of the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan.