List of Publications

Journal Papers

1. Yoshitaka Nakamura, Hirozumi Yamaguchi, Akihito Hiromori, Keiichi Yasumoto, Teruo Higashino and Kenichi Taniguchi : "An Application Layer Multicast Protocol for Provisioning Quality of Service by Using End System's Video Filtering", *IPSJ Journal*, Vol. 45, No. 2 (February 2005), pp. 438–448 (in Japanese).

2. Hirozumi Yamaguchi, Yoshitaka Nakamura, Akihito Hiromori, Keiichi Yasumoto, Teruo Higashino and Kenichi Taniguchi : "An Application Level Multicast Protocol for Multi-party Visual Communication Systems", *JSSST Journal "Computer Software"*, Vol. 21, No. 2, pp. 1–11 (March 2004) (in Japanese).

3. Yoshitaka Nakamura, Tomoya Kitani, Akira Kimura, Hirozumi Yamaguchi, Akio Nakata and Teruo Higashino : "Secure Multiple Association Control Protocol for VPNs", *IPSJ Journal*, Vol. 48, No. 2 (February 2007) (in Japanese) (to appear).

Conferences Papers

1. Yoshitaka Nakamura, Hirozumi Yamaguchi, Akihito Hiromori, Keiichi Yasumoto, Teruo Higashino and Kenichi Taniguchi : "On Designing End-user Multicast for Multiple Video Sources", *Proceedings of the 2003 IEEE International Conference on Multimedia and Expo (ICME2003)*, Vol. III, pp. 497–500, Baltimore, Maryland, USA (July 2003).

2. Yoshitaka Nakamura, Guiquan Ren, Masatoshi Nakamura, Takaaki Umedu,

and Teruo Higashino : "Personally Customizable Group Navigation System using Cellular Phones and Wireless Ad-Hoc Communication", *Proceedings of the 2005 IEEE International Conference on Multimedia and Expo (ICME2005)*, CD-ROM, Amsterdam, Netherlands (July 2005).

3. Yoshitaka Nakamura, Hirozumi Yamaguchi and Teruo Higashino : "Maximizing User Gain in Multi-flow Multicast Streaming on Overlay Network", *Proceedings of the 2006 International Workshop on Future Mobile and Ubiquitous Information Technologies (FMUIT'06)*, pp. 158–162, Nara, Japan (May 2006).

Abstract

In recent years, broadband networks have come into wide use in many households as the progress of information network technology and the demand for network systems that communicate each other in a large-scale group has increased.

To achieve such a type of network systems, we need to solve how to keep secure communication among group members. In the future, the demand for such a communication will be increased in the project in which more companies participate. Moreover, there is a possibility that the communicated data becomes large data such as video data.

Up to now, the security of such communication has been achieved by constructing VPNs (Virtual Private Network). A VPN is a private communication network often used within a company, or by several companies or organizations, to communicate confidentially over a publicly accessible network. However, in a large-scale group, each member may want to associate multiple VPNs simultaneously.

In these group communications, it may be also useful to use simultaneous transmission to two or more members of a group. Recently, the delivery of rich contents such as movie data has come to be increased. When delivering such data, there is a problem of lacking scalability of group communication when the unicast channel is constructed between all two users.

In this thesis, the following two research topics are studied as solutions for the above purposes; (1)a method to achieve secure multiple association that considers indirect connection when multiple VPNs are constructed on the same VPN architecture, and (2)a method to achieve application layer multicast delivered maximizing user's priority of communication quality.

First we propose a VPN association control protocol that dynamically con-

trols a multiple association by efficiently collecting the policies regulated by each VPN and the information about indirect connection of multiple association, and judging the acceptance of association according to that information on existing VPN architecture. In the proposed method, the architecture of VPN must be composed of the provider edge router connected with the provider network and the customer edge router prepared on the site side on the network that the service provider provided. And the communication of VPN can use an existing protocol. Our proposed method can control a scalable association of VPNs by decentralized processing of each PE, and can resolve the conflict of association requests to keep the consistency of the information.

Second, we propose a protocol that constructs delivery trees on overlay networks. It dynamically controls which video streams should be delivered and how much bandwidth they should be used under the given condition about the stream forwarding ability of each host, the overlay link capacity, and the priority request (preference) that each host has specified for target video streams on the network. For instance, a participant of a video-conferencing system may specify higher preference for videos such as speakers, and specify comparatively low preference for videos such as other audiences. When two or more video streams compete for bandwidth of the target overlay network, the proposed protocol guarantees delivery quality of the video stream with higher preference by decreasing the quality of the video stream with lower preference. Moreover we assume that each end host has a function to adjust the transfer rate when the video is forwarded to other end hosts by multicast. By using the function, the proposed protocol implements the admission control to accept the receiving request of a new video stream by adjusting the transfer rate of existing video streams cooperatively among the end hosts. The simulation experiment as performance evaluation shows that the proposed technique has achieved more high-quality data delivery than existing methods.

Contents

List of Figures

Chapter 1

Introduction

In recent years, broadband networks have come into wide use in many households as the progress of information network technology and the demand for network systems that communicate each other in a large-scale group have increased.

First of all, to achieve such a type of network systems, we need to solve how to keep secure communication among group members. For instance, there is the demand that exchanges data like the trade secret between the offices or the companies that exists in remote place. In the future, the demand for such a communication will be increased in the project in which many companies participate. Moreover, there is a possibility that the communicated data become large such as video stream.

Up to now, the security of such communication has been achieved by constructing *Virtual Private Network* (*VPN*). A VPN is a private communications network often used within a company, or by several companies or organizations, to communicate confidentially over a publicly accessible network. However, in a large-scale group, each member may want to associate multiple VPNs simultaneously. For example, the project to extend over two or more companies is thought. It is desirable that the participant can easily exchange data mutually when the participant wants to share information on the project mutually. But it is also necessary to prevent information from leaking outside the project at the same time.

In the existing VPN architectures, the multiple association control decides acceptance of requests based on only the conditions for organizing a VPN that

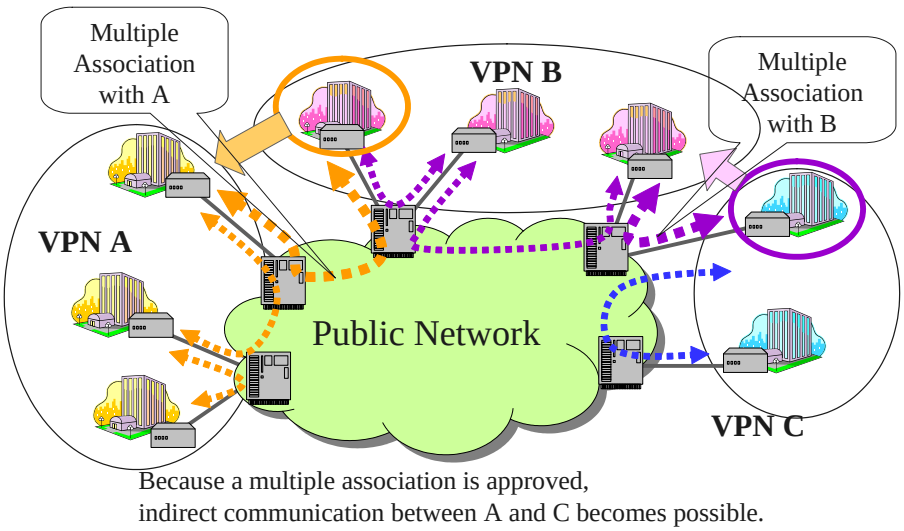
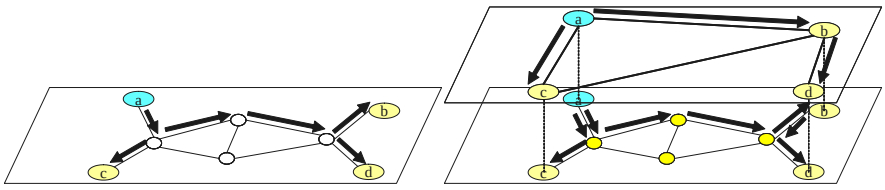


Figure 1.1: Multiple association and indirect communication

is specified for each VPN (called policies hereafter) where the requesting site associates and the VPN that receives the association request such as communication protocol and access control[1]. However, there is a possibility that the information leakage occurs between different VPNs. Because if the site is infected with malware (malicious software) etc. that copied the data that existed in a certain VPN and put it away an accessible area from another VPN, the site turns into the gateway, and indirect communication between these VPNs becomes possible (Shown in Fig. 1.1.) When the request is accepted, it is necessary to consider such an indirect connection because it is thought that it does not want to connect indirectly via a VPN to the company that exists mutually in a rival relation on the profit-pursuing side. Therefore, the association control has to take into consideration not only the policy of a site and a VPN of the source and destination of association request, but also the information on other VPNs that can indirectly communicate each VPN via another VPN. We need a method that the association control of multiple VPNs can be done securely.

In these group communications, it may be also useful to use simultaneous transmission to two or more members of a group. Recently, the delivery of rich contents such as movie data has come to be increased. When delivering such



(a)IP Multicast (b)ALM

Figure 1.2: Application layer multicast and IP multicast

data, there is a problem of lacking scalability of group communication when the unicast channel is constructed between any two users. IP multicast is considered as an attractive solution for such communication. However, it is necessary to prepare special hardware such as IP multicast enabled routers to achieve the IP multicast. Then, the method to deliver by using *application layer multicast* (ALM) is thought to be useful. ALM is a new kind of communication paradigm which realizes multicast communication in the application layer. On ALM, each end user constructs a network constructed over the application layer (called *overlay network* hereafter) and acts as multicast routers. As for ALM, there are some advantages: (A) it is possible not to depend on a specific infrastructure to achieve multicast on the Internet, and (B) the utilization of the resource becomes higher than that of the unicast. However, it is necessary to design the ALM protocol in consideration of the bandwidth restriction near the end host so that ALM uses the link between end hosts for the data delivery route. Therefore, when multiple videos are delivered by ALM, the bandwidth resource conflict is occurred because each video may consume a constant amount of the bandwidth on the overlay network. Users may have priority requirements to each video streams. For example, in the video conferencing, users may prefer the speaker's video than other audience's video.

Then, on this ALM the resource management method is needed to avoid the bandwidth competition taking user's priority into consideration and improve the delivery quality.

As described above, in this thesis, the following two research topics are studied as solutions for the above purpose, (1) a method to achieve secure multiple association that considers indirect connection when multiple VPNs are constructed on the same VPN architecture, (2) a method to achieve application layer multicast delivered maximizing user's priority of communication quality.

Chapter 3 presents a method to achieve secure multiple association that considers indirect connection when multiple VPNs are constructed on the same VPN architecture.

Since existing approaches of the VPN association are achieved by using a static setting by manual operation, it is necessary to set all the possibilities beforehand when dynamic association requests are assumed. Therefore, the association control becomes inefficient. On the other hand, in the MAVPN architecture [2], multiple association of VPNs becomes possible by grouping VLANs in the site, and judges the association policies by each VLAN. In this method, more efficient multiple association control can be achieved compared with the previously existing methods by controlling access limitation in each group. Moreover, in the VPN architectures([3, 4, 5]) that can dynamically control the association of VPNs, because the policy such as composing the routing table according to the situation of the site can be set, a dynamic control of multiple association is possible depending on the setting of the policy.

We propose a VPN association control protocol that dynamically controls a multiple association by efficiently collecting the policies regulated by each VPN and the information about indirect connection of multiple association, and judging the acceptance of association according to that information on existing VPN architecture. In the proposed method, the architecture of VPN must be composed of the provider edge router (called PE hereafter) connected with the provider network and the customer edge router (called CE hereafter) prepared on the site side on the network that the service provider provided. And the communication of VPN can use an existing protocol.

The policy about VPN that exists in the state that can be indirectly communicated is defined as a policy besides an existing VPN association policy. If the policy of all VPNs that has the connection relationship is not efficiently judged when the scale of VPN grows, the memory area which manages the collection time, the bandwidth, and control information needs very large area, and lacks the scalability. Then, in this method, policies collected from all VPNs are limited only about the connection relationship among VPNs, this information of connection of each VPN is maintained in PE. And the overlay network (called PE-graph hereafter) for collecting the information that consists only of PE with information on corresponding PE of each sets of VPN indirectly connected is

constructed.

Our proposed method can control a scalable association of VPNs by collecting this information by decentralized processing of each PE on this graph, and can resolve the conflict of association requests to keep the consistency of the information.

Chapter 4 presents a method to achieve Application Layer Multicast (ALM) delivered like maximizing user's priority of communication quality.

In the existing research of ALM, some protocols are proposed based on various design goals such as overhead of ALM, the cancellation of instability, and the pursuit of the scalability.

In the application that the streaming is delivered to multiple videos simultaneously and concurrently, the application may compete for the limited resource such as bandwidth on overlay link and host's forwarding ability on the overlay network. But an efficient control method of resource in that case is not considered.

According to the improvement of recent computer ability, the end hosts can deliver the video by lowering its bit rate in real time by video filtering process instead of stopping it when the bandwidth resource is not sufficiently available[6]. Therefore, when the each end host is assumed to have such a processing ability, we can manage the resources more flexibly to which a lot of videos can be delivered in the high quality as much as possible and there is a possibility that the high quality service can be provided that improves user's satisfaction rating.

We proposed a protocol that that constructs the delivery tree on overlay network and controls dynamically and in a decentralized way which video stream to deliver under the stream forwarding ability of each host and the limited overlay link capacity based on the priority request (*preference*) that each host specified for each video stream on the network. For instance, the participant of the conference specifies higher preference for videos such as speakers and specifies comparatively low preference for videos such as other audiences and the entire halls in the video-conferencing system. When two or more video streams compete for the bandwidth of overlay network, proposed protocol guarantee the delivery quality of the video stream with higher preference by decreasing the quality of the video stream with lower preference. Moreover we assumed that

each end host has the function to adjust the transfer rate when the video is forwarded to other end hosts by multicast. By using the function, the proposed protocol implements the admission control to accept the receiving request of a new video stream by adjusting the transfer rate of an existing video stream cooperatively among the end hosts. The simulation experiment as a performance evaluation shows that the proposed technique had achieved higher users' satisfactory data delivery than existing methods.

Chapter 2

Related work

2.1 Multiple Association of VPNs

In the existing method, multiple association of VPNs achieves by adjusting communication protocol, access control etc., and by changing the setting of the routers among managers of VPNs[1]. Since these approaches are achieved by using a static setting by the manual operation, it is necessary to set all the possibilities beforehand when dynamic association requests are assumed. Therefore, the association control becomes inefficiency.

On the other hand, in the MAVPN architecture [2], multiple association of VPNs becomes possible by grouping by VLAN in the site, and begin authenticated by each VPN that each host connects. In this method, more efficient multiple association control can be achieved compared with past method by controlling access limitation in each group.

Moreover, in the VPN architectures([3, 4, 5]) that can dynamically control the association of VPN, because the policy such as composing the routing table according to the situation of the site can be set, a dynamic control of multiple association is possible depending on the setting of the policy. In Ref. [3], a new policy-based VPN management architecture is proposed. This method deals with the problem to keep consistency of globally related policies that may change dynamically. The proposed VPN management architecture in Ref. [3] consists of local policy servers and a centralized global management server. The local policy servers manage local policies of each site, whereas the global management server controls the consistency of global policies. However, it cannot properly deal with

the global policy when it is a globally *distributed* policy, that is, a policy that each site can independently update its policy locally and the change may affect the violation of some global policy. The multiple association policy dealt with in this paper is one of such policies. Thus, Ref. [3] cannot solve our problem. Moreover, this approach is essentially a centralized/distributed hybrid approach and still does not scale when the number of sites becomes large. In Ref. [4], the dynamically changing globally distributed policy issue in VPN management is solved. In this method, the notion of *administrative domain* is considered. An administrative domain is a set of sites that a single administrator is able to manage. In the proposed VPN management architecture, there is a single domain management server for each administrative domain. To keep a global policy among different administrative domain (an inter-domain policy), domain management servers communicate each other and check if the policy is violated or not. In this method, global policies that are directly related to each domain of sites can be managed. However, it is not capable with managing indirectly related global policies such as the multiple association policy, which this paper deals with. In Ref. [5], a method to transform a high-level simplified policy into a concrete domain-specific policy with complex configuration parameters is proposed. This method can analyze indirect dependence of multiple policies and transform into the corresponding local policies for each site dynamically. However, the issue to keep the consistency of the global policy is not dealt with.

In our proposed method, it is necessary to collect the connection information of the entire network. BGP[7] is known as a technique for automatically collecting routing information on the network. The information about routes of the entire network is collected by exchanging routing information while avoiding the route loop between each two routers. Moreover, in BGP, the information is collected while checking the attribute of the route to select the best route. For the problem that we consider, the cost of the judgment process of the best route can be omitted because it only has to be able to know only the connection relationship in our method, and such information can be collected efficiently. Moreover, the technique for guaranteeing no contradiction of the association process is also needed in our method.

As for distributed consistency resolution, there are some related classical researches in the area of distributed systems[8, 9]. Ref. [10] has proposed the

concept of a logical clock (called Lamport's timestamp) to identify the total order of events in a distributed system. Ref. [11] applied Lamport's timestamp to the conflict resolution of distributed transactions. The idea is that when two transactions are in a conflict relation, the later transaction (in Lamport's timestamp) is aborted. Our conflict resolution algorithm for VPN multiple association management is essentially based on such a distributed transaction control method based on Lamport's timestamp.

As for distributed information collection algorithm, some basic distributed algorithm called wave algorithm[9] is proposed. In our proposed method, we use a variant of the wave algorithm called PIF (Propagation of Information with Feedback) algorithm[9] to collect multiple association information over distributed nodes efficiently.

2.2 Application Layer Multicast

For our research goals, IP multicast[12] is considered as an attractive solution for such a group communication with respect to efficient utilization of network resources. However, there is a problem that it is costly for us since the prior configuration as the address administration, the access control, and the group management for the application are needed. Moreover, in the IP multicast, there are the problems of lacking reliability compared to the unicast communication between end hosts and the problem that there is a design restriction in the degree of freedom of the application by the complexity of the protocol in the network layer.

It is thought that these costs can be reduced by using the multicast in the application layer, and high degree of freedom can be given to the application design. There are a lot of studies have been dedicated to designing Application Layer Multicast of different design goals. These designs can be roughly classified into infrastructure-based and host-based solutions. Infrastructure-based multicast implements multicast functionality in network nodes that are responsible for both constructing the multicast tree and replicating multicast packets at the branch points in that tree. Overcast[13] proposes a method to construct a bandwidth-aware shared tree over wide-area networks, for distribution of large files like VoD systems. Yoid[14] uses both a shared tree and a mesh-like network for the robustness of overlay networks. ALMI[15] tries to minimize

the total delay of a shared tree. RMX (Scattercast)[16] aims at providing a communication strategy for heterogeneous users. CAN[17] aims at simplifying multicast tree construction by mapping hosts on a virtual address space (overlay networks). These solutions have a low degree of flexibility, because the multicast tree construction algorithm is typically implemented at the nodes. Host-based multicast does not need support from any network nodes. Instead, the multicast functionality is implemented entirely by collection of end hosts participating in the multicast groups. HBM[18] mainly focuses on the construction of backup links for stable tree management. NICE[19] reduces state by using a hierarchical structure in constructing multicast tree. Each end-host is arranged in hierarchy of layers and clusters. Narada[20, 21], in particular in [21], considers tele-conferencing as a target application and constructs bandwidth and delay conscious multicast routing trees from multiple senders to avoid congestion of video streams on overlay networks. While these solutions are highly flexible since they do not have deployed infrastructure, they often suffer from scalability and robustness concerns.

As for structure of overlay trees, there are three types of designs. In the mesh-first approach, group members first distributedly organize themselves into the overlay mesh topology. Each member participates in a routing protocol on this control topology to distributedly compute unique overlay paths to every other member. This type of designs are RMX[16] and Narada[20, 21]. In the tree-first approach, protocols distributedly construct a shared data delivery tree first directly. Subsequently, each member discovers a few other members of the multicast group that are not its neighbors on the overlay tree and establishes and maintains additional control links to these members. Yoid[14],ALMI[15] uses this type of approach. In the implicit approach, protocols create a control topology with some specific properties. The data delivery path is implicitly defined on this control topology by some packet forwarding rule which leverages the specific properties of the control topology to create loop-free multicast paths. NICE[19] uses this approach.

Although there is much kind of researches with various targets of ALM, no method is designed to solve the resource conflict between multicast groups when the multiple video is delivered. These literatures give elegant design to achieve their own design goals.

Chapter 3

Distributed Control for Multiple Association of VPNs

3.1 Introduction

VPN (Virtual Private Network) over a public network is widespread as the means to communicate safely between offices geographically away, and to provide some information safely to specific business partner instead of using a physical private line. Compared with the private line, VPN can reduce the cost since VPN can work with cheaper public networks like the IP network., and the connection such as multicast can be easily achieved since a virtual link of VPN is constructed on public network. In addition to such an advantage, from the diversification of present network service the demand to intercommunicate between sites (the minimum unit that composes VPN) of different VPNs by constructing a virtual link called "bridge" is occurred. At this time, it is called that these sites multiply associate with both VPNs.

Usually, the condition to accept each association request (called *policy*), such as, which site is admitted to associate, and which encryption protocol is acceptable, is set to each VPN respectively, and VPN is constructed among the sites whose policy is fulfilled. If a certain site wants to multiply associate with VPNs, it is necessary to satisfy all the policies of those VPNs. Even in the existing VPN control protocol, the control of multiple association can be achieved by

the statically specified settings beforehand about the acceptance of the association request, and such a setting has to be manually changed according to the association request. However, in the former method the administrator should write the setting considering all the possibilities beforehand, and it is difficult to correspond to the dynamic association request. The latter one is unlikely since the response speed and the scale is too high to operate manually.

In the multiple association control of the existing model, only the policies of the requester (i.e. the policy of the VPN that the site that generated the association request belongs to) and the requestee (i.e. the policy of the VPN that received the request) is judged. However, there is a danger that the information leakage happens between some different VPNs not directly associated since indirect communication between them might occur via some other directly associated VPNs at the same time. When the association request is accepted, it is necessary to consider the connection relationship between such indirectly connected VPNs because it is thought that companies in rival relationship do not want to connect VPNs indirectly. Therefore, the acceptance of association should be decided by considering not only the policies of two VPNs which is the transmission former and destination of the association control but also the policies of each VPNs which can be indirectly communicated via another VPN.

In this Chapter, we propose a new VPN association control method that dynamically controls a multiple association by collecting information about connection of each VPN and policies regulated by each VPN on existing VPN architecture while considering efficiency, and judging the acceptance of association according to those information In this method, the architecture of VPN must be composed of the provider edge router (called PE) connected with the provider network and the customer edge router (called CE) prepared on the site on the network that the service provider provides, and the communication of each VPN use an existing VPN protocol. The information about VPNs that can be indirectly communicated is defined as a policy besides the existing VPN association policy. When the scale of VPN grows, if the policy of all VPNs that exists in the connection relationship is not efficiently judged, the storage area to maintain the collection time, the bandwidth, and control information grows very large, and the scalability cannot be achieved. Thus, we limit the information collected from multiple VPNs to only the connection relationship of them.

Moreover, we assume that the state of the connection of VPNs is maintained in each PE, and that an overlay network (called PE-graph hereafter) is maintained for each group of PEs that control VPNs that are indirectly connected each other. The policies are collected from only the related PEs via the overlay network by using a scalable distributed algorithm. Moreover, to keep consistency of information on each PE when multiple conflicting association requests occur, we also propose a distributed algorithm to resolve such conflicts using Lamport's[10] timestamp.

3.2 Multiple Association

In this section, we call that a site S *associates* with VPN V when the site S becomes a member of the VPN V and is enabled to communicate with other members of V.

In order to communicate between sites associated with different VPNs, a site needs to associate with two or more VPNs. Then, when a site associate with a VPN and another VPN simultaneously, that is, the site associates with two or more VPN, the site is defined to *multiply associate* with VPNs.

Fig.3.1 shows the example of multiple association. The site S_1 multiply associate with VPN A and VPN B, S_2 with B and C and S_3 with A, B and C.

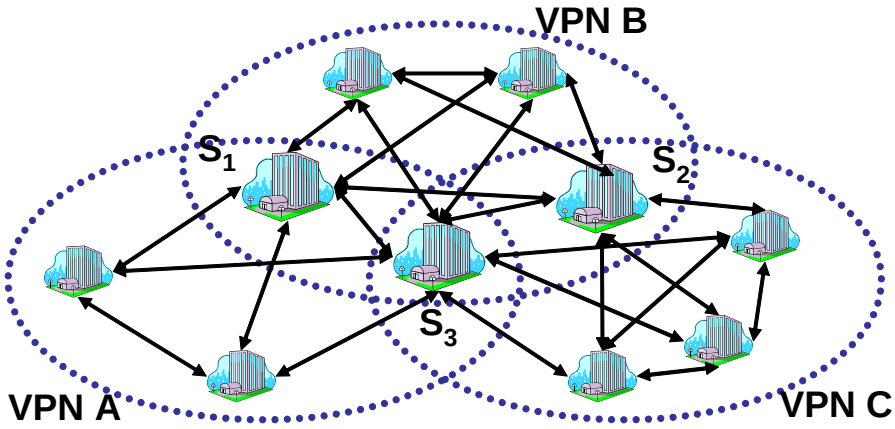


Figure 3.1: Multiple association

In the existing VPN architecture, though the following control of association is not assumed.

For example, it is assumed that there are three organizations of A, B and C as shown in Fig.3.2, and of each composes its original VPN. A and B, B and C are assumed to be respectively in a friendship relation, but A and C in a rival relation. It is also assumed that there is the site that multiply associated with both VPN of A and B (VPN of A and VPN of B can communicated each other). When the site in C generates the association request for B, it is possible to take the standpoint that the site in A doesn't want to accept association of B that exists in friendly relations, because A has the possibility of information leakage to C thorough B. In a word, the policy of A can be set that A cannot accept association of the site in C with B when B can be communicated with A itself, but in the other situation A can accept the associstion.

Moreover, such information leakage becomes an enough threat, because confidential information can be relayed not only by some malicious host in the intermediate VPNs but also by a spyware installed to some host unconsciously. The most frequently possible case is that some host is simply misconfigured by human error so as to relay such confidential information.

In order to achieve such an association control in existing VPN architecture, the VPN manager always observes the network situation, and when a vulnerable situation is occurred, it is necessary to change the policy. In the existing method[3, 4], the association control is possible according to its network situation depending on the setting of the policy, but it is not assumed that the policy takes the situation of other sites into consideration. Therefore, the association control like the one described above is difficult.

3.3 Secure Multiple Association

Even if the security of the VPN communication itself is improved, the information leakage caused by a multiple association cannot be prevented. In order to prevent such an information leakage, it is sufficient to trace for the entire network to check whether there is no indirectly connected site at the same time that we do not want to leak information to. However, it is undesirable to trace all the sites to perform such checking since it lacks scalability. Thus, we propose a method that can efficiently collect information of the connection relationship

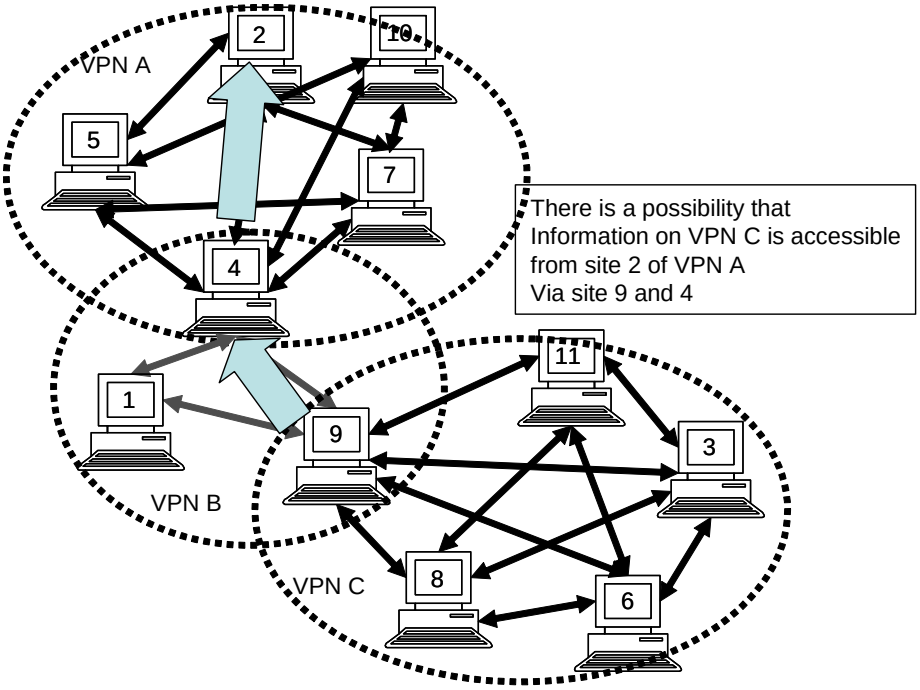


Figure 3.2: Information leakage

of VPNs by reducing the set of sites to be traced.

3.3.1 Architecture

In this subsection, we describe the target underlying VPN architecture of our method. Our method assumes *PPVPN* (*Provider Provided VPN*) architecture composed of *CE*(*Customer Edge router*) on each site and *PE* (*Provider Edge router*) on the provider, as shown in Fig. 3.3. CE has the function to transmit the association request of the site for VPN to PE, and to manage the policy for which the site provides. PE has the function to change the communication setting based on the request of the site, to manage the policy for which each VPN provides, and to exchange information about the policy etc. between other PEs. Moreover, each PE has some working memory to maintain the routing table with other PEs and the VPN information of CEs that the PE manages, and it is assumed that the function is enhanced to update these information by message exchanging.

3.3.2 Policy

We have only to check whether there is danger of indirect information leakage by an indirect connection aside from the security of the VPN communication itself. Thus, in our method, we consider that an association policy for a site is a set of other sites that do not want to be connected indirectly. We call such a set of sites as *prohibited sites*. We assume that such a policy is specified for each site. If such policies are assumed, there is a possibility of causing the violation of the policy by some VPNs that has associated now when the policy is updated. In that case, it is assumed that the user which updated the policy have to leave.

Moreover, it is not considered that a policy itself is misspecified at some site by human error causing unwanted information leakage. To cope with such a human error, the system might have to make a confirmation at every time a set of reachable site will be changed, which makes the system's performance considerably worse. Therefore, here we do not assume such a policy misspecification.

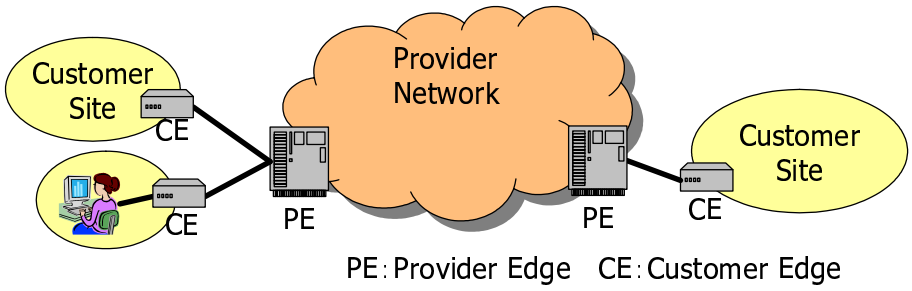


Figure 3.3: Overview of architecture

3.3.3 Preliminary Definitions

We define the topology control problem (Hereafter, we call "secure multiple association control problem") to prevent information leakage through multiple association of VPN and some concepts for the problem as follows.

Definition of Logical Network Topology of Sites

First of all, the sets of all sites s are defined as S and *VPN* v is defined as an arbitrary subset of sites, that is $v \in 2^S$. The set of all VPNs is denoted by V ($V = 2^S$). For arbitrary two VPNs v_1 and v_2, we say that they *multiply associate* if some site $s' \in S$ exists such that $s' \in v_1 \cap v_2$, denoted by *m-assoc*(v_1, v_2). A *multiple association graph* is a graph where each vertex is a VPN and an edge exists between two vertices if and only if the corresponding two VPNs multiply associate. Formally, a multiple association graph is a graph $G_{ma}(V, E_{ma})$ where $E_{ma} \stackrel{\text{def}}{=} \{(v_1, v_2) | v_1, v_2 \in V, \textit{m-assoc}(v_1, v_2)\}$. VPNs $v_1, v_2 \in V$ are said to be *reachable* if and only if there are some path between v_1 and v_2 in the multiple association graph G_{ma}, denoted by *reachable*(v_1, v_2). We call the sets of vertices $V_1, \ldots, V_k$ of the connected components of G_{ma} as *reachable VPN groups*. From the definition of the connected component of the undirected graph, for all $i, j \in \{1, \ldots, k\}$, $i \neq j$ implies $V_i \cap V_j = \emptyset$. Moreover, as it is easily proved, for any site $s \in S$, there is exactly one reachable VPN group V_i such that $s \in v$ and $v \in V_i$ for some $v \in V$. We say that two sites $s_1, s_2 \in S$ are *reachable* if there exist $v_1, v_2 \in V$ such that $s_1 \in v_1$, $s_2 \in v_2$, and *reachable*(v_1, v_2), denoted by *reachable*(s_1, s_2). The relations of these notations are illustrated in Fig. 3.4.

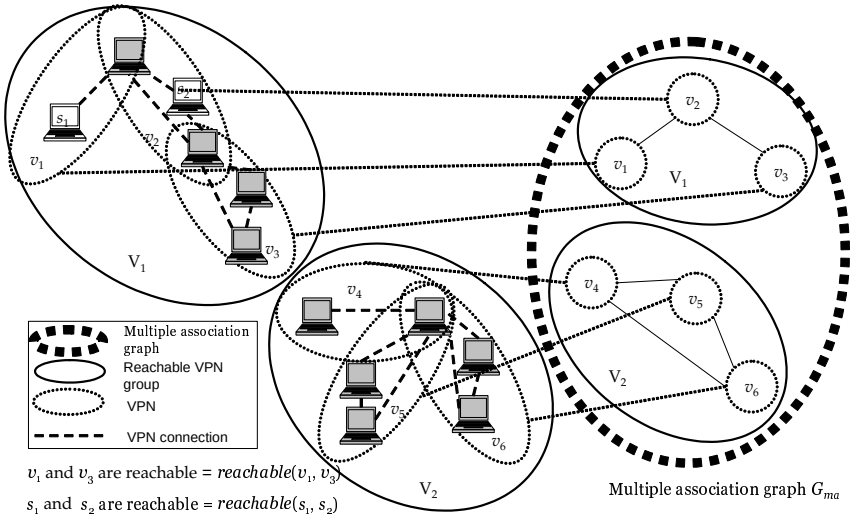


Figure 3.4: Multiple association graph and its relation to reachability

Definition of the Multiple Association Policy of a Site

A *multiple association policy* $Policy(s)$ of a site s is a subset of sites that s never wants to be reachable. Although it is possible that we consider other kind of policies, for simplicity, we focus on only such a kind of policies described above. A multiple association policy $Policy(s)$ of a site s is *satisfied* if and only if there are no site $s' \in Policy(s)$ such that $reachable(s, s')$. We say that the policy of a VPN v is *satisfied* if and only if for every site $s \in v$, the policy of s is satisfied. We say that the policy of a VPN group V_i is *satisfied* if and only if for every VPN $v \in V_i$, the policy of v is satisfied.

Definition of the Physical Network Topology of a Site

We assume that for each site $s \in S$ there is a PE that is responsible for the VPN association management of s, denoted by $PE(s)$. The set of all PEs is denoted by $PEs \stackrel{\text{def}}{=} \{PE(s) | s \in S\}$. We also assume that s and $PE(s)$ are directly connected by direct physical communication link through a CE, and each PE is connected with some other PEs by direct overlay communication link. A set of PEs which is directly connected to $PE(s)$ is denoted by $Neighbor(PE(s))$. *PE-graph* is an undirected graph $G_{PE} = (V_{PE}, E_{PE})$ where $V_{PE} \subseteq PEs$ and

$E_{PE} \stackrel{\text{def}}{=} \{(p,p')|p \in V_{PE} \wedge p' \in \mathit{Neighbor}(p)\}$. For each PE $p \in PEs$, $\mathit{Site}(p)$ denotes the set of all sites $s \in S$ such that p is responsible for the VPN association management of s, and $VPN(p)$ denotes the set of all VPNs $v \in V$ such that $s \in \mathit{Site}(p)$ is associated to v. Formally, $\mathit{Sites}(p) \stackrel{\text{def}}{=} \{s|s \in S \wedge p = PE(s)\}$ and $VPN(p) \stackrel{\text{def}}{=} \{v|v \in V \wedge s \in \mathit{Sites}(p) \wedge s \in v\}$.

We say that a PE-graph $G_{PE} = (V_{PE}, E_{PE})$ *covers* a VPN group $V_i \subseteq V$ if and only if for any two VPNs $v_1, v_2 \in V_i$ and for any two sites $s_1 \in v_1$ and $s_2 \in v_2$, there is a path in G_{PE} from $PE(s_1)$ to $PE(s_2)$. If a PE-graph G_{PE} covers any VPN groups V_i in a multiple association graph G_{ma}, then we say that G_{ma} *can be managed by* G_{PE}. For any VPN $v \in V$, we define a subgraph $G_{PE}(v) = (V_{PE}(v), E_{PE}(v))$ that covers any VPN group V_i such that $v \in V_i$, that is, $V_{PE}(v) \stackrel{\text{def}}{=} \{p|\exists v' \text{s.t.} \mathit{reachable}(v,v') \wedge v' \in VPN(p)\}$ and $E_{PE}(v) \stackrel{\text{def}}{=} \{(p,p')|p,p' \in V_{PE}(v) \wedge p \in \mathit{Neighbor}(p')\}$. We also denote the neighborhood set of each p in $G_{PE}(v)$ as $\mathit{Neighbor}(p,v) \stackrel{\text{def}}{=} \{p'|(p,p') \in E_{PE}(v)\}$.

The association request of s for arbitrary VPN is sent to $PE(s)$ through a CE. After $PE(s)$ communicates with other PE according to the communication topology by a PE-graph, the association control process of the request is started.

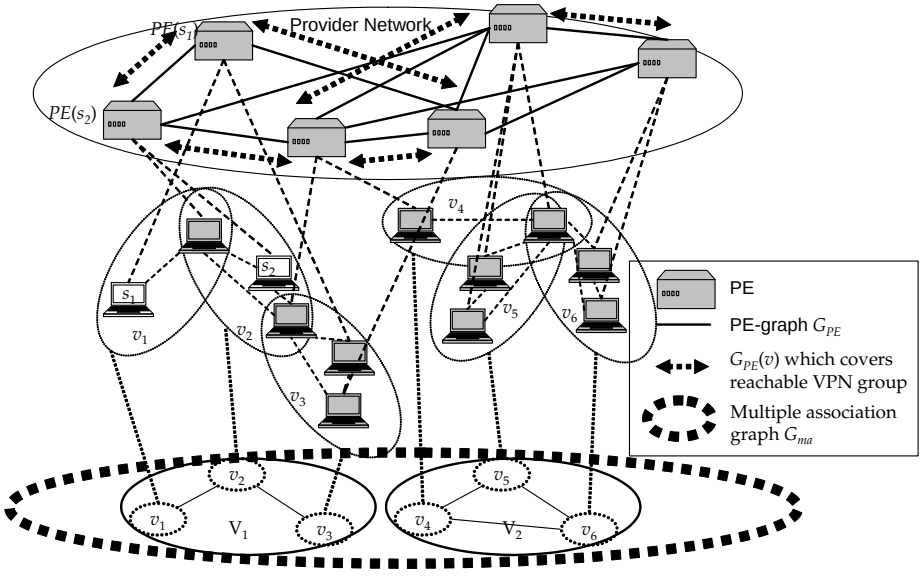


Figure 3.5: PE-Graph

Problem Definition

Now we define our problem precisely. *Secure multiple association control problem* is a problem to decide whether to grant each association request of a site s to a VPN v when the request is sent, while keeping the policies of all the sites satisfied, even if two or more association requests that are generated simultaneously on different network locations and in a conflict relation, that is, not all of the simultaneous requests can be granted if some site's policy is violated.

3.4 Multiple Association Protocol

3.4.1 Basic Idea

There are some possible approaches to control the association of VPNs while keeping global multiple association policy. One approach is that a server manages the association policies and controls association by these polices in order to control secure multiple association. But when policies or the information of VPN construction are often changed, such a centralized approach will suffer from scalability problem. For instance, if each site is a small unit like an individual person, and it frequently changes its policy, such a situation occurs. On the other hand, if policies over distributed PEs are collected and cached at each PE, efficient distributed policy checking may be possible. However, when distributed caches are used, it becomes difficult to keep their consistency over all PEs. Inconsistency of policy information may sometimes cause unintended policy violation, which is unacceptable from a security point of view. Therefore, in our proposed protocol, each PE collects the related policies from other PEs and checks them on the fly when needed, while keeping the consistency their policies strictly.

Specifically, the policies concerned with multiple association are based on the information about which site is reachable inside of a VPN group. Thus, when each PE will check whether a site's policy is satisfied, it is sufficient for the PE to collect such information of only from the inside of the VPN group that is reachable from the site's associated VPN (hereafter, we call such a VPN group as the *reachable VPN group*). Therefore, we restrict the domain of collecting the policy information into only the PEs those are responsible for the sites inside the reachable VPN group. To this end, we use a PE-graph covering

each reachable VPN group as the overlay network topology of PEs to collect such policy information. By this approach, we can check multiple association requests occurred at different reachable VPNs in parallel and distributedly. Only the critical case is that multiple association requests occurred at the different sites of the same reachable VPNs, and their requests are conflicting, that is, their policies are mutually exclusive. In this case, it is required to resolve such a conflict among requests so that the consistency of the policies can be maintained.

3.4.2 Overview of the Association Control Method

We will explain the overview of the association control method using Fig. 3.6. When a site s send an association request of VPN v to $PE(s)$(Process(1)), first, $PE(s)$ locally checks whether there exists a site s' that will be reachable when the request is accepted and thus violates the policy of the site s. Conversely, it also checks whether the association of s satisfies the policy of s'(Process(2)). If it does, $PE(s)$ sends a "no" message to s. Otherwise, $PE(s)$ forwards the request to all the PEs that manage any site associated with some VPN v' in the reachable VPN group of v(Process(3)) and collects the answers to the request(Process(4)) . Each PE uses the overlay network topology $G_{PE}(v)$ to forward the request. That is, each PE p forwards the request only to the neighboring PEs $\mathit{Neighbor}(p, v)$ in $G_{PE}(v)$. If all the forwarded messages are answered with "yes" messages, then each PE p answers "yes" to the PE from which the request is forwarded. Otherwise, it answers "no". If $PE(s)$ is answered with "yes" message, then the request from the site s is granted(Process(5)).

When the association request to v of s from other PE p' which is $p' \in \mathit{Neighbor}(p, v)$ has been forwarded (Process(a)), PE p checks locally whether he request satisfies the policies of s and all sites $s' \in \mathit{Sites}(p) \cap V_i$ where V_i is a reachable VPN group such that $v \in V_i$(Process(b)). If not satisfied, a "no" message is answered to p'. Otherwise, the PE p forwards the requests to the PEs in $\mathit{Neighbor}(p, v) \setminus \{p'\}$(Process(d)). In this case, only when "yes" messages are received from all of the PEs in $\mathit{Neighbor}(p, v) \setminus \{p'\}$, p answers "yes" to p'.

Thereby, the association request to v of s is forwarded to all the sites in the reachable VPN group V_i such that $v \in V_i$ along with the overlay network topology $G_{PE}(v)$, and only when all the answers to the request are "yes", it is guaranteed that $PE(s)$ answers "yes" to the original association request sent

by the site s.

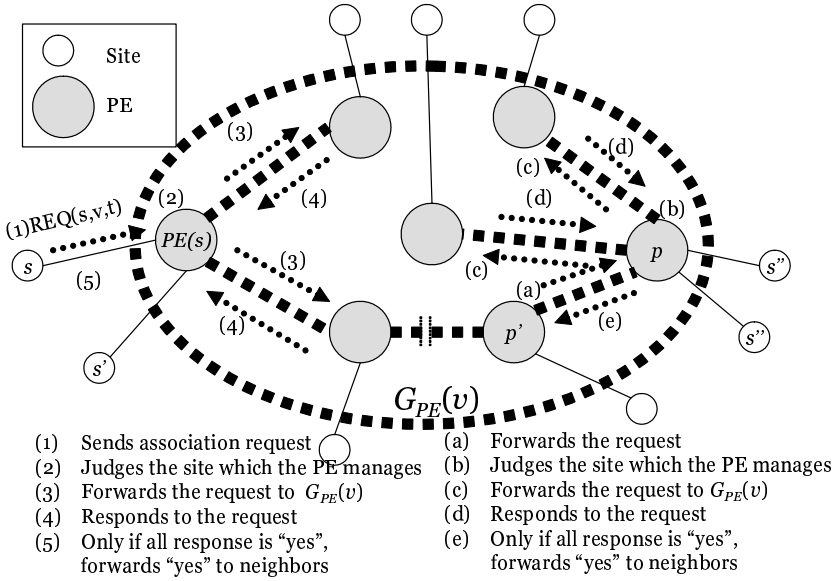


Figure 3.6: Overview of proposed protocol

3.4.3 Management of Correspondence of the Associated VPN and the Representing PE

A PE which received an association request from its managing site may not have already joined with the overlay network of the VPN with which the site is requesting to associate. In this case, the PE must join the overlay network via some already joined PE. To this end, it is necessary for the PE to know some selected PE from the overlay network to join with. If there are other PEs which manage the association site to v, it is necessary to choose the suitable *representing PE* from them, to forward the association request of v to p', and to connect with p' in PE-graph. It is necessary to enable the judgment of the existence of the PE which manages the associated site to v, and the search of representing PE for this process. Then, we assume the existence of a search server that answers the status of any VPN v and its representing PE. Each PE that firstly associates with some VPN becomes a representing PE of the VPN, and registers itself to the search server. Thereby, the pair of the address of PE

and the ID of the VPN is registered to the search server, and this information is referred to by other PEs. When some PE requests association with some VPN, first it searches the address of the representing PE of the VPN using the search server, and if found, the request is forwarded to the representing PE. Usually, the representing PE is set to the PE that associates with the VPN first, but when the representing PE leaves from the VPN by some reasons, it hands over its role as the representing PE to some neighboring PE, and it notifies to the search server.

3.4.4 Resolving Conflicts of the Requests

There is a possibility of receiving the same association requests for some VPN v' by some site s' from the different PEs within the same VPN group. If the same request is forwarded, the request can be safely disregarded since it is simply duplicated by some closed cycle of the PE-graph. However, when multiple different requests are received and forwarded, there is a risk of granting two conflicting association requests at the same time unless they are appropriately controlled. The technique for avoiding this problem is described as follows.

Basic Idea

To resolve the conflict of multiple association requests, we first record the *logical time* (also known as *Lamport's timestamp*[10]) at the time the request is firstly received by a representing PE. The logical time is a total ordering of distributed events that is consistent with the causal ordering of them, which is maintained by assigning a unique strictly increasing number to each event in the entire distributed system. If a PE p_1 firstly received an association request $REQ(s_1, v_1, t_1)$ of a site s_1 for a VPN v_1 with a logical time t_1 from some neighboring PE $p' \in \text{Neighbor}(p_1, v_1)$, and then receives another association request $REQ(s_2, v_2, t_2)$ of a site s_2 for a VPN v_2 with a logical time t_2 from some neighboring PE $p'' \in \text{Neighbor}(p_1, v_2)$, the VPNs v_1 and v_2 might be in the same VPN group. In this case, there is a possibility of request conflict according to the policies of the relating sites associated with any VPN in the VPN group.

To cope with this problem, such a PE p_1 firstly checks whether $REQ(s_2, v_2, t_2)$ satisfies the policy of the sites in $Sites(p_1)$ under the assumption that the previously arrived request $REQ(s_1, v_1, t_1)$ had been granted, that is, the VPN v_1

had been updated to $v_1 \cup \{s_1\}$. If the result of the check is true, $REQ(s_2, v_2, t_2)$ will not be in conflict with $REQ(s_1, v_1, t_1)$ as long as the PE p_1 knows, and thus, p_1 processes both of the two requests simultaneously. Otherwise, p_1 concludes that the two requests are in conflict. In this case, the logical times of the two requests are compared and only the earlier one is processed. For example, if $t_1 < t_2$, then p_1 decides to process $REQ(s_1, v_1, t_1)$ and immediately answers "no" to p'' from which $REQ(s_2, v_2, t_2)$ is received. That is, if $t_1 < t_2$, "no" is immediately answered to p'' that receives $REQ(s_2, v_2, t_2)$. Otherwise, $t_1 > t_2$ must hold (since the logical time for each event is unique, no two events have equal logical times). In this case, the PE p_1 locally checks whether $REQ(s_2, v_2, t_2)$ satisfies the policies in $Sites(p_1)$ under the assumption that $REQ(s_1, v_1, t_1)$ is not granted. If the result of the check is true, $REQ(s_2, v_2, t_2)$ is processed instead of $REQ(s_1, v_1, t_1)$. In this case, if the PE p_1 has not answered "no" to this request, it answers "no". Otherwise, it has already answered "yes" to the previously received request. In this case, the PE p_1 does nothing to the previously answered request and simply ignores it. If the request has already been answered, the PE does not process anything. Meanwhile, if another association request is received, logical times are compared and similar processing is done. On this policy, only the earliest association request in the logical time can receive the answer of "yes" from PE that manages all reachable sites from the site to the conflicting association request caused from all the PEs that are managing all the reachable sites. Any other conflicting association requests having later logical timestamps are ensured to receive "no" from at least one PE and thus they are ensured to be rejected.

For instance, if two association requests $REQ(s_1, v_1, t_1)$ and $REQ(s_2, v_2, t_2)$ are firstly received at different PEs p_1 and p_2 ($p_1 \neq p_2$), $REQ(s_1, v_1, t_1)$ must have been firstly received at p_1, and $REQ(s_2, v_2, t_2)$ must have been firstly received at p_2. It is guaranteed that p_1 answers "no" to $REQ(s_1, v_1, t_1)$ if $t_1 > t_2$, or p_2 answers "no" to $REQ(s_1, v_1, t_1)$ if $t_1 < t_2$ without making any inconsistency. Moreover, if two requests are received at the same site $p_1 = p_2$, then obviously it answers "yes" to the firstly received requests and "no" to the later one. Therefore, conflicting requests can be resolved in this distributed algorithm consistently.

Extending to the Case with Two or More Pending Requests

If there are two or more pending association requests at some PE, there is a possibility that all of such processing requests together are conflicting with a currently receiving request even if each of them is not individually conflicting with the current one. For instance, assume that the VPN association status is $v_1 = \{s_1\}$, $v_2 = \{s_2\}$, $v_3 = \{s_3\}$. Moreover, consider that some PE p has already received the requests $REQ(s_1, v_2, t_1)$ and $REQ(s_2, v_3, t_2)$, forwarded them to the other PEs, and is currently waiting for the answer of them. The two requests are currently pending states. Then, suppose that p has received $REQ(s_3, v_1, t_3)$ just now. Moreover, consider that the site s_3 has a policy that s_1 must not reachable. This association request does not conflict with any one of the two previously received requests. However, it does conflict with all of the two requests if they are assumed to be granted, since v_1 and v_3 become reachable. In this case, if the logical time of the currently received request is earlier than any of the pending requests, then the PE answers "no" to all the pending requests and it begins to process the currently received one. If there are some pending request such that its logical time is earlier than the currently received request, then the PE answers "no" to the currently received one and it continues to process the pending ones.

3.4.5 Maintaining Information of the Multiple Association Relationships

Since multiple association is generated in each site, there may exist two or more sites that multiply associate with the same pair of VPNs and managed by different (and possibly distant) PEs. For instance, two VPNs $v_1 = \{s_1\}$ and $v_2 = \{s_2\}$ are managed by the PE p_1, and suppose that p_1 thinks v_1 and v_2 are mutually unreachable, as shown in Fig. 3.7. Moreover, consider that some site s_3 begin to multiply associate with v_1 and v_2 at some PE p_2 that is placed far away from p_1. v_1 and v_2 become newly reachable. Now consider the case that another site s_4 sends a request to the PE p_1 that it wants to associate with v_1 and it has a policy that it does not want to be reachable with s_3(Process (4)). Then, the PE p_1 must reject this request since now v_1 and v_2 are reachable via the PE p_2. However, p_1 may fail to reject the request if the information that v_1 and v_2 become reachable is not propagated from p_2 to p_1. As illustrated

in this example, the existence of such a distributed multiple association makes a problem when the policies are checked locally at each PE and when VPN groups are merged at another PE. Then, each PE notifies to all the other PEs in the VPN group the information that two VPNs become reachable when some multiple association is generated, and that two VPNs become unreachable when some multiple association is destroyed.

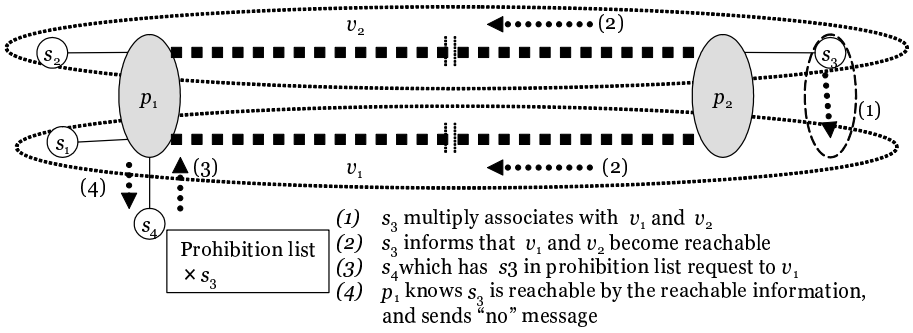


Figure 3.7: Notification of reachability

However, still there exists a problematic case that some PE decided that some two VPNs become unreachable and then sends the notification, while they are still reachable via some site at another PE far from the PE. For instance, suppose that s_1 leave from v_1 in the situation such as $v_1 = \{s_1\}$and $v_2 = \{s_1\}$ on PE p_1 as shown in Fig. 3.8. Then, p_1 locally decides that v_1 and v_2 become reachable and notifies that to other PEs. However, PE p_2 located far away from p_1 knows v_1 and v_2 are still reachable in the situation such as $v_1 = \{s_3\}$ and $v_2 = \{s_3\}$. In this case, when the unreachable notification from p_1 is received via its neighboring PE, p_2 sends back the reachable notification of v_1 and v_2 to cancel the unreachable notification when the notification from p_1 toward the direction in which the unreachable notification has been sent. To maintain consistency, any PE that receives these notifications rejects all the pending association requests by answering "no".

3.4.6 Updating the State of Each PE

The state variables that each PE $p \in PEs$ should maintain are an *unprocessed association request list*, a set of *neighboring PE sets for each VPN* v denoted by

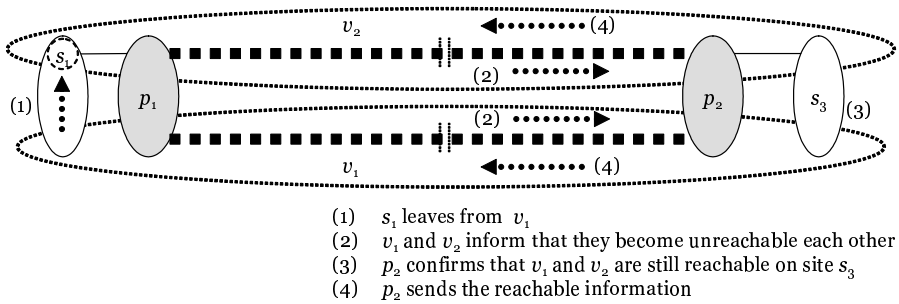


Figure 3.8: Correction of unreachability notification

Neighbor(p, v), and a *VPN association status* of the sites under the control. The element of each unprocessed association request list consists of an association request $REQ(s, v, t)$ that is received and has not answered yet, a source PE p' of the request ($p' = p$ if p directly received the request from the site that p manages), and its processing status which consists of the information that which PE it has forwarded the request and which answer it has received from the PE (either unreceived, "yes" has been received, or "no" has been received). Each VPN association status consists of (site ID, list of associated VPNs, associated VPN group ID) for each site s.

The state of each PE is updated (1)when the PE receives the association request, (2)when the PE receives the answer to the association request, (3)when the association request is finally granted and the site associates with the VPN, and (4)when the site leaves from a VPN. At this time, each state variable is updated respectively as follows.

Case(1), (2) When $REQ(s, v, t)$is received, and "no" is not answered immediately by a local policy checking, the entry of this association request is added to the pending association request list. At this time, the processing status of this request in the list is initialized to "unreceived" from any neighboring PE. The corresponding answer is updated every time the answer is received from a neighboring PE after the request is forwarded. If the PE received "no" from some neighboring PE, or "yes" from all the neighboring PE except the PE from which the request is forwarded, the PE deletes the entry of the corresponding association request after sending the answer of this request to the PE from which the request is forwarded.

Case(3),(4) When $REQ(s, v, t)$is finally granted, $PE(s)$ that manages s adds s to v, chooses some reachable site s' from s, and adds $PE(s)$ to the set Neighbor$(PE(s'), v)$. On the other hand, when $PE(s)$ receives the leaving request for v of s, $PE(s)$ deletes s from v. If all the sites managed by $PE(s)$ and reachable from v have become unreachable, $PE(s)$ leaves from the PE-graph $G_{PE}(v)$. In this case, $PE(s)$ selects suitable p' from $Neighbor(PE(s), v)$, $PE(s)$ is deleted from $Neighbor(pp, v)$ for all the neighboring PE $pp \in Neighbor(PE(s), v) \setminus \{p'\}$ and p' is added in place of $PE(s)$, and $PE(s)$ is deleted from $Neighbor(p', v)$ and all the elements of $Neighbor(PE(s), v) \setminus \{p'\}$ is added in place of $PE(s)$. Moreover, when p receives the notification that v_1 and v_2 become reachable, both values of $Neighbor(p, v_1)$ and $Neighbor(p, v_2)$ are updated into $Neighbor(p, v_1) \cup Neighbor(p, v_2)$.

3.4.7 Reconstruction of PE-graph

It is not necessary to reconstruct PE-graph only from a viewpoint of collection of exact information. The frequent reconstruction of PE-graph may cause inconsistency of association information. If the PE-graph is not reconstructed, the algorithm may unnecessarily forwards each association request to more PEs than included in each VPN group, but it is harmless since an unrelated PE simply answers "yes" to such a request. However, since increase of the size of PE-graph may cause performance overhead, it is desirable to reconstruct PE-graph to be as optimal as possible. Then, the proposed method collects the states of PE-graph periodically by using *distributed snapshot*[22], and reconstructs by locking association process of all PEs based on the suitable interval determined from the total number of PE.

3.5 Performance Evaluation

We have implemented the simulator and evaluated the performance of our proposed method by a simulation experiment.] In this section, we will describe the detail and the results of the experiment.

3.5.1 Environment of Experiment

In this experiment, we assume that the environment that frequently updates the policy and changes and adds association of VPN every tens of minutes by several thousand sites under the hundreds of PEs that the provider prepares.

In the simulation, 10-100 PE is prepared, and about 100-1000 sites as a whole generate the association request at random. The delay between PEs is given by the normal distribution that becomes 0.05 seconds on the average, and the judgment time in PE is assumed to be the number of management sites $\times$ each policy size (size of the prohibition list) $\times$ 0.001 second. Moreover, the generation interval of the association request and policy change are shortened, the situation in which the conflict of the request easily occurs, and whether this method operates correctly is confirmed under these situation.

3.5.2 Evaluation Items

The item evaluated with this thesis is as follows.

1. Response time of association request
2. Necessary amount of memory area for PE to add

To judge the association request, the time to trace PEs and to collect policies and the time judged on each PE are needed. And there is a request to wait until the judgment of one request ends when the conflict association request is generated at the same. Therefore, the response time including all the times together seems a good measure of the performance. Thus, we evaluate the response time to the request.

To achieve this method, the extra data described in Sect. 3.4.6 must be stored in addition to existing architecture. Therefore, it is necessary to add the storage area to store these information in each PE. It is necessary to confirm that this amount of the additional storage area is within a feasible value for an increase in the entire number of sites (The amount of the area does not increase explosively).

3.5.3 Response Time of the Request

When the number of total CEs is 100-1000, we evaluate the response time of the request under the situation in which the ratio of the number of PE and the

number of CE is adjusted to 1:5, 1:10, and 1:20 respectively. Fig. 3.9 shows the result. The number of total CE (= the number of total sites) is taken in x-axis, and time from the request generation to the answer reception is taken in y-axis. The increase in the average response time to the growth of total CE in numbers does not depend on the ratio of the number of PE, and be suppressed to low. Because unnecessary PE is not traced by using PE-graph when tracing PEs for the judgment of the policy, such a result is obtained. Moreover, the response time has increased as the number of sites increases. But by the centralized control, a worse result is expected since such a centralize server must process each request sequentially in order to resolve the policy conflict properly.

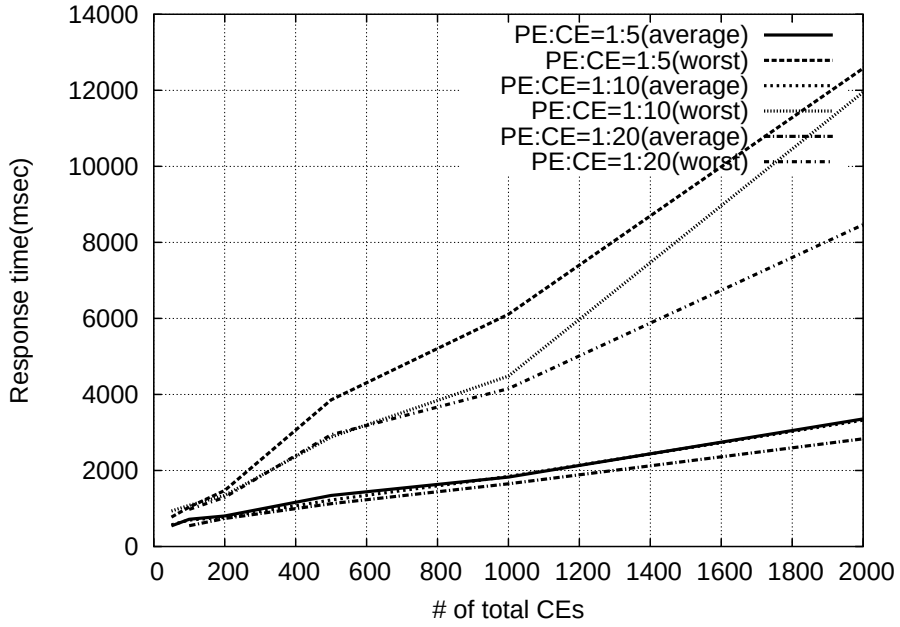


Figure 3.9: Response time for requests

3.5.4 Memory Area of PE

We evaluate the necessary amount of the memory area for PE to achieve this method. Here, the amount of the memory area that this method used in each PE is examined in following situations, (1)when the number of PE is fixed to 10, and the number of CE that each PE manages is made to change from 10

to 100 and (2) when the number of CE that each PE manages is fixed to 10, and the number of PE is made to change from 10 to 100. The 10% of CEs are selected from all CEs at random as prohibited sites, and it describes it in the policy (prohibition list). Moreover, to prepare for the simultaneous arrival of the request, the request queue of the length of 100 requests with the address in 16 bytes and the message data of 10 kbytes are able to be saved is prepared in each PE. Fig. 3.10 shows this result. The entire number of CE is taken in x axis and necessary amount of the memory area for y axis is taken. When 1000 of CEs are managed by 10 of PEs, that is, when a PE manages 100 of CEs , it is settled with memory area extension of about 1.2 Mbyte. Since most of the enhanced memory area is used to store the policy, and this policy is separately managed in each CE, it has a large influence on the amount of the memory area that the growth in numbers of CE each PE manages than the growth of all numbers of CE. In this experiment, 10% of CEs are chosen from all CEs, and the ID of the CEs is described in the prohibition list as the association policy. If the number of CEs managed on each one PE is kept small by increasing PEs as CE increases, the amount of the memory area can be also kept small.

Therefore, it is feasible to enhance existing architecture and to achieve the proposed method.

3.6 Conclusion

In this chapter, we propose a distributed algorithm to achieve dynamic multiple association control of VPN according to the situation of the association of VPN about other sites while considering efficiency. A dynamic, multiple association is achieved by the distributed control based on present security policy for which each site provides and VPN composition information. At this time, current VPN association information is managed by PE prepared by the provider, and by exchanging these information only between necessary PEs that relate to multiple association, the information for the judgment can be collected accurately and efficiently. It was confirmed to operate at realistic processing time when this method was mounted as a result of the simulation experiment.

The future work includes developing the dynamic reconstruction method (especially splitting) of PE-graphs, and optimization of the topology of the PE-graphs to minimize the response time.

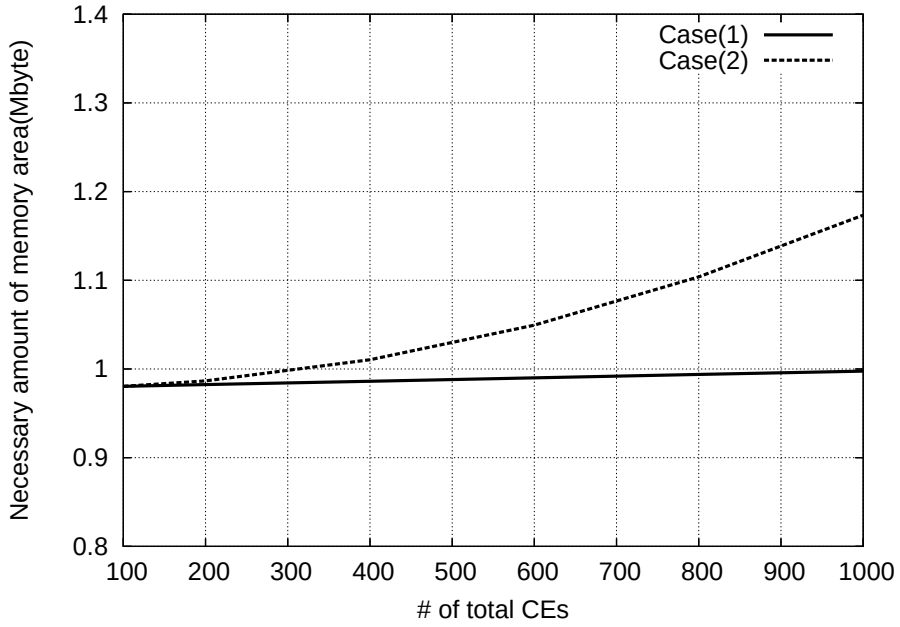


Figure 3.10: Necessary amount of memory area

Chapter 4

Maximizing User Gain in Multi-flow Multicast Streaming on Overlay Networks

4.1 Introduction

Group communication over the Internet is one of key features for the deployment of network applications. In such group communication, the delivery of the same data to all/some members within a group is required. IP multicast is considered as an attractive solution for such group communication. However, in the current Internet infrastructure, there is a deployment issue over wide-area networks. On the other hand, using multiple unicast connections lacks scalability as the group size grows.

As a realistic solution for the above problem, a new kind of communication paradigm which realizes multicast communication in the application layer (called ALM: application layer multicast) has had much attention. In ALM, end-hosts manage multicast distribution trees and forward packets on an overlay network which consists of unicast tunnels (or some methods such as Ref.[14] allow IP multicast or hybrid of unicast tunnels and IP multicast) between those end-hosts. For example, in Fig. 4.1(a), host B receives a packet from A, and forwards it to C and D by duplicating it (Fig.4.1(b) depicts the multicast distribution tree). In ALM, several disadvantages are inevitable, for example, redun-

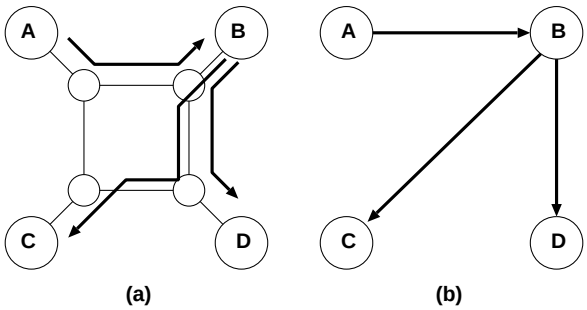


Figure 4.1: Application layer multicast

dant packet duplication on a single physical link (*link stress*), cumulative delay of multiple unicast connections (*link stretch*), and instability of overlay networks due to joining/leaving of end-hosts. On the other hand, ALM does not require any more than the current Internet infrastructure, and the functionalities of unicast transport protocols such as retransmission control and flow/rate control are available. Moreover, application developers may design their original routing policy based on their applications.

We have some research goals to realize ALM protocol. First, we would like to provide flexible control for an application such as teleconferencing, *i.e.*, we would like to utilize limited bandwidth on overlay network. Many literatures have shown the efficiency of overlay multicast especially in practical aspects for such a situation in group communication where the size of a group is not so large but there exist a large number of groups. Both server-based and unicast-based solutions may not work well in this situation. However, bandwidth on overlay networks is limited by not only end-to-end bandwidth but also the bandwidth of local area network interfaces (*e.g.* ADSL subscribers may have to be aware of uplink bandwidth). On the other hand, in spite of recent progress of end hosts' computing power, computing capabilities are not fully utilized in most network applications. Therefore, under the limitation of overlay link capacities, we may consider to use end hosts' computing power to filter the quality of streams to achieve higher utilization of overlay link resources. This assumption is also useful for heterogeneous environments where users communicate using various terminals such as PCs, PDAs and so on. Under this assumption, our aim is to maximize the gain of all the users, according to given user gain functions for

available bandwidth.

Secondly, we would like to give theoretical analysis for the construction problem of multiple multicast trees on a given overlay network, maximizing the gain of all users. We call this problem *Maximum Gain Multi-flow Streaming (MGMS)* problem and prove that this problem is NP-hard. Considering this complexity and the fact that MGMS problem does not consider join/leave operations of user nodes, we re-formulate this MGMS problem into more realistic one called *dynamic MGMS* problem that maximizes the gain of all users when a given request to obtain bandwidth on a path on the overlay network should be accepted. In dynamic MGMS problem, it is allowed to take bandwidth from the existing streams on the path to accept the request which may enlarge the gain of all the users.

In this chapter, we propose a new application layer multicast protocol called Emma/QoS. In Emma/QoS, based on a shortest path tree rooted at each host built on an overlay network and priority values given by the host to the others hosts' video flows, each host determines which flows to be relayed to which neighboring nodes to maximize the total sum of satisfied priority values under capacity limitation of each overlay link. Emma/QoS also implements an optimal and polynomial time algorithm of the dynamic MGMS problem in a decentralized manner to enlarge the gain of overall users. To our best knowledge, none of the existing methods, including our previous approach Emma, has not treated MGMS problem and dynamic MGMS problem formally. Our experimental results have shown that using only a small amount of network traffic to realize the distributed version of dynamic MGMS problem, Emma/QoS could archive higher gain of users compared with some existing methods. This shows the efficiency of dynamic MGMS problem treated in this chapter. We have also shown that the other facilities such as overlay network construction could work well.

Note that MGMS problem may look similar with transcoding techniques on P2P networks, and several researches have been proposed [23]. They assume that several proxy nodes for media services such as transcoding services are located in overlay networks, and have proposed algorithms to find an optimal path where some proxy nodes perform a series of transcoding to satisfy bandwidth constraints, transcoding costs, quality requirements and so on. However, those existing algorithms can be regarded as QoS routing algorithms (*i.e.* they try to

find paths to satisfy requests) which are quite different from our situation.

This chapter is organized as follows. Section 4.2 gives the formulation of the MGMS and dynamic MGMS problems treated in this chapter. Then in Section 4.3, we present the basic design of Emma/QoS. In particular, the distributed optimal algorithm of dynamic MGMS problem is described in Section 4.4. Section 4.6 gives performance evaluation of Emma/QoS and Section 4.7 concludes the chapter.

4.2 Problem Formulation

We assume that each user node (simply called *node* hereafter) can degrade some quality levels' packets of a multicast streaming flow (called *stream* hereafter)in order to fit the bandwidth of the stream to a certain level when relaying it to the other nodes. Fig. 4.2(a) shows a simple example where node b degrades the bandwidth of stream a from two to one when forwarding it to node c (we call the stream sent from source node m simply "stream m"). If some nodes do not have such functionality, we can assume layered coding at stream sources in order to enable relaying nodes drop some packets without knowing the contents of streams. Several filtering techniques have been investigated so far and Ref. [6] gives a good summary.

4.2.1 User Gain Function

We assume that every user defines a user gain function for every stream. It represents the user's additional gain when a unit of bandwidth is added to the current receiving stream. The user gain functions may be application specific, however, specifying such a function for every stream is not an easy task. Therefore, we show a reference specification defined from a *utility function* (bandwidth-quality function) in Ref. [24].

In Ref. [24], a utility function of rate-adaptive applications is defined as shown in Fig. 4.3(a). Here, we let h denote a certain amount of bandwidth on the overlay links and we regard it as a unit of bandwidth. We also let $Bmax_m$ denote the maximum bandwidth that stream m uses. Thus $\lceil \frac{Bmax_m}{h} \rceil$ units of bandwidth can be offered to stream m as the maximum (this value is denoted by K_m). Based on the above, we can use an approximate function of the given utility function as shown in Fig. 4.3(b). Then we can obtain the following

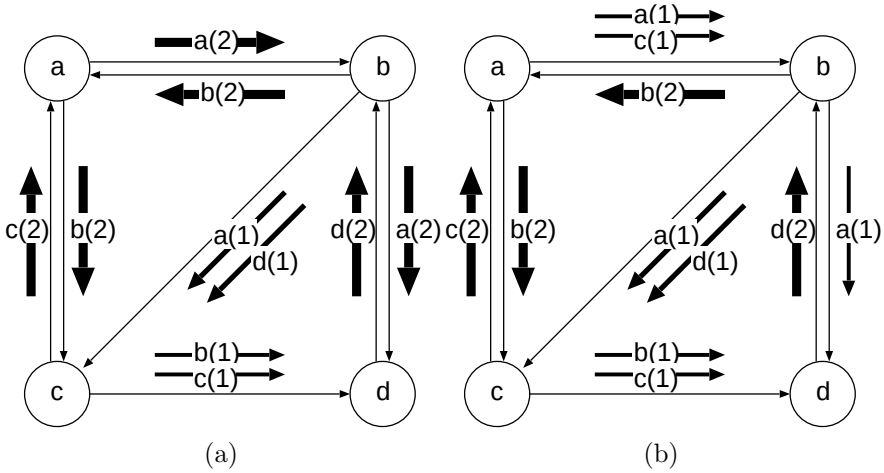


Figure 4.2: Streaming on an overlay network. Each overlay link (thin directed edge) has two units of bandwidth as its capacity and $x(k)$ on thick arrow indicates that stream x uses k units of bandwidth.

discrete function $gain_m^k(v)$. This returns the additional gain of node v which is currently using $k-1$ units of bandwidth for stream m when the node v additionally obtains the k-th unit of bandwidth for stream m ($k = 1..K_m$)

$$gain_m^k(v) = U_m(hk) - U_m(h(k-1))$$

where U_m is a utility function of stream m. Of course, this is a sample specification of user gain functions. Since our approach does not depend on specific functions, application managers may design their favorite ones. For example, if some users need to be prioritized due to application semantics, we may be able to set higher return values of the user gain functions of such prioritized users.

4.2.2 Maximum Gain Multi-flow Streaming Problem

Given an overlay network and user gain functions, we formulate the problem called *Maximum Gain Multi-flow Streaming (MGMS) problem* which determines how the streams are transmitted on a given overlay network, maximizing the user gain of all the users. This formulation gives the basis for our problem formulated in the next section.

We assume that the followings are given.

- A user gain function $gain_m^k(v)$ for every pair of node v and stream m

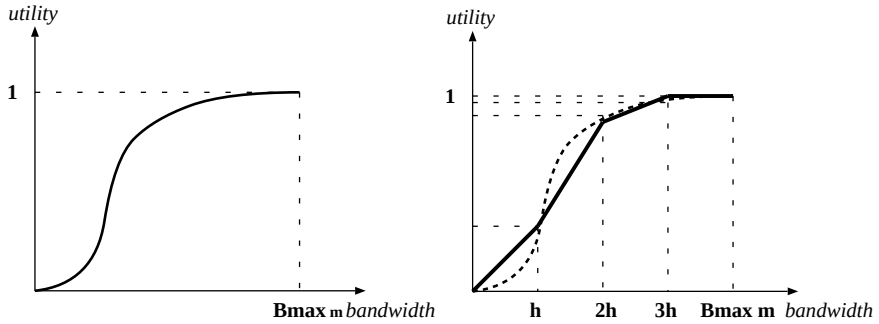


Figure 4.3: (a) A utility function and (b) its linear approximation for rate-adaptive applications

($k = 1..K_m$). It denotes the additional gain when node v obtains the additional k-th unit of bandwidth for stream m.

- A directed network graph $G = (V, E)$ which represents an overlay network where $E \subseteq V \times V$ and three assignments C, D and C_{node}. $C : E \rightarrow N^+$ and $D : E \rightarrow N^+$ assign link capacity and delay to each overlay link respectively, and $C_{node} : V \rightarrow N^+$ assigns a node capacity to each node (the maximum number of bandwidth units which the node can simultaneously use to send streams).

- The upper bound D_{bound} of overlay path delay from every source to its receivers when the receivers receive the stream from the source.

Then the MGMS problem is formulated as a problem to determine how many units of bandwidth can be used for each stream on each link, maximizing the total gain of all the nodes, under the constraints of overlay link capacity, node capacity and overlay path delay bound[1]. Here, we formulate the problem as an integer linear programming problem in Fig. 4.4 where we introduce the following variables:

- $\pi_m^k(i, j)$: one *iff* stream m can use k units of bandwidth on link (i, j) (from node i to node j) otherwise zero, and

[1]We assume that users who may not want to receive a stream specifies zero values as their user gain functions for the stream. Then we can formulate the MGMS problem assuming that every user wants every stream.

[Objective Function]

$$\max \sum_{j \in V} \sum_{m \in V, m \neq j} \sum_{i \in V, i \neq j} \sum_{k \in 1..K_m} gain_m^k(j) * \pi_m^k(i,j) \tag{4.1}$$

[Constraints]

$$\forall m, \forall (i,j) \in E, \forall k \in 1..K_m \;\; [j \neq m]; \sum_{l:(l,i) \in E, i \neq m, l \neq j} \pi_m^k(l,i) \geq \pi_m^k(i,j) \tag{4.2}$$

$$\forall m, \forall k \in 1..K_m \;\; [j \neq m]; \quad 1 \geq \sum_{(i,j) \in E} \pi_m^k(i,j) \tag{4.3}$$

$$\forall m, \forall (i,j) \in E, \forall k \in 1..(K_m - 1); \;\; \pi_m^k(i,j) \geq \pi_m^{k+1}(i,j) \tag{4.4}$$

$$\forall (i,j) \in E; \sum_{m \neq j} \sum_{k \in 1..K_m} \pi_m^k(i,j) \leq C(i,j) \tag{4.5}$$

$$\forall i \in V; \sum_{j \in V, j \neq i} \sum_{k \in 1..K_m} \pi_m^k(i,j) \leq C_{node}(i) \tag{4.6}$$

$$\forall m; \quad d_m(m) = 0 \tag{4.7}$$

$$\forall m, \forall i; \quad D_{bound} \geq d_m(i) \tag{4.8}$$

$$\forall m, \forall (i,j) \in E \; [m \neq j]; \quad d_m(j) \geq \pi_m^1(i,j) \tag{4.9}$$

$$\forall m, \forall (i,j) \in E \; [m \neq j]; d_m(j) \geq d_m(i) + D(i,j) * \pi_m^1(i,j) - D_{bound} * (1 - \pi_m^1(i,j)) \tag{4.10}$$

Figure 4.4: ILP formulation of Maximum Gain Multi-flow Streaming problem

- $d_m(j)$: the overlay path delay from the source node m to node j when node j receives stream m.

Note that $\pi_m^k(i,j) = 1$ also indicates that node j receives stream m from node i using at least k units of bandwidth.

In Fig. 4.4, the constraints (4.2) and (4.3) are required for tree-formed streaming. They represent that if a node i sends stream m to node j using k units of bandwidth, at least one neighboring (up-link) node l of node i must send the stream m to node i using k units of bandwidth. Also, the number of such up-link nodes must be at most one. The constraint (4.4) represents the consistent relationship between $(k+1)$-th unit and k-th unit of bandwidth of a single stream. If $(k+1)$-th unit of bandwidth is added in order to transmit a stream m on link (i,j), k-th unit of bandwidth must have been already allocated to the stream m on the same link. The constraints (4.5) and (4.6) represent the capacity constraints for each link and node, respectively.

The constraints (4.7), (4.8), (4.9) and (4.10) are related with the overlay path delay bound. Every node which receives a stream m using at least one unit of bandwidth must satisfy the delay bound. We note that in the constraint (4.10), the last term of the right-hand side makes this constraint effective only if $\pi_m^1(i,j) = 1$. Otherwise, the value of the right-hand side becomes always negative, which makes this constraint always true for any values of $d_m(j)$ and $d_m(i)$ that satisfy the constraint (4.8).

In the following, we prove that this problem is NP-hard.

Theorem 1 *The MGMS problem is NP-hard.*

Proof: We will transform a 3-SAT problem to the MGMS problem as follows. Here, we give an outline of the proof.

Assume that a 3-SAT problem g such as an example Boolean expression (a product of sums) in Fig. 4.5 is given. Then, we introduce nodes x_i (i=1, ..., 4) where the set $\{x_1, ..., x_4\}$ of nodes correspond to the variables used in g. For each x_i, we also introduce nodes x_{i_t} and x_{i_f} which represent true and false assignment to x_i, respectively. We assume that $C_{node}(x_i)$ is one for every x_i and that $C(x_i, x_{it}) = C(x_i, x_{if}) = 1$. That is, only one stream with one unit of bandwidth is permitted from node x_i. Therefore, either x_{i_t} or x_{i_f} can receive a stream with one unit of bandwidth.

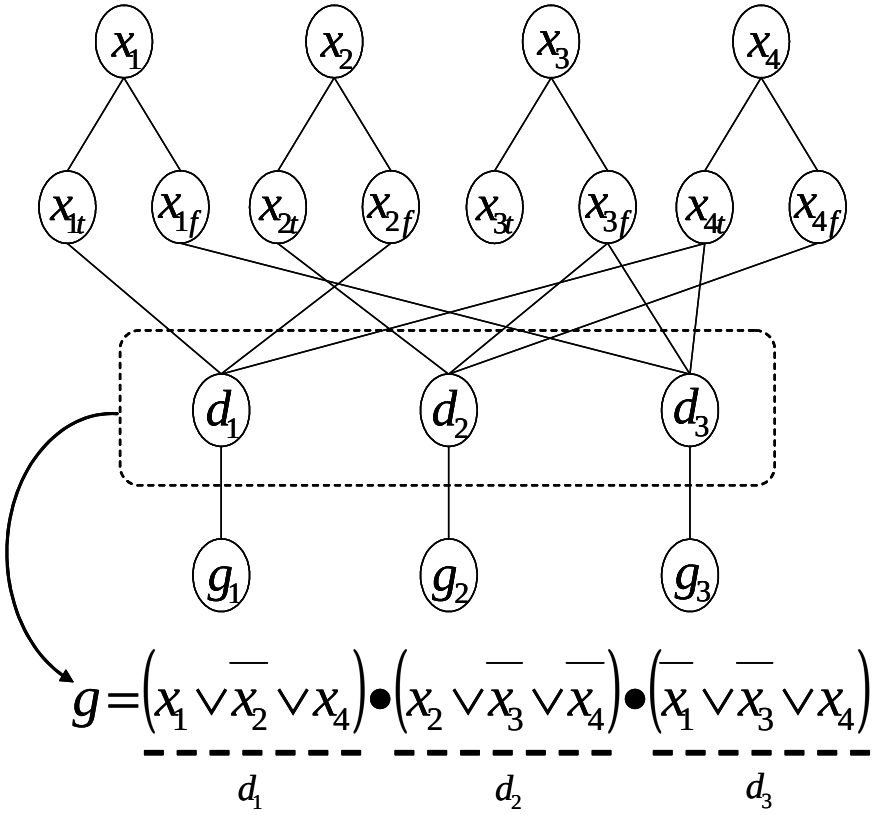


Figure 4.5: NP-hardness of MGMS problem: proof strategy

For each clause d_j in g (*e.g.*, d_1, d_2 and d_3 in Fig. 4.5, we introduce the corresponding node d_j. Since we consider 3-SAT problems, each clause has three literals such as $(x_1 \vee \overline{x_2} \vee x_4)$. If it contains a positive literal x_k, then we add an edge (x_{k_t}, d_j). If it contains a negative literal $\overline{x_k}$, then we add an edge (x_{k_f}, d_j). If this clause is satisfiable, then the node d_j can receive at least one stream from three connected nodes where each stream has one unit of bandwidth.

Finally, we add node g_j for each node d_j where we assume that $C(d_j, g_j)$=1. Then, each node g_j can receive a stream with one unit of bandwidth *iff* the corresponding clause d_j in g is satisfiable. Note that even if node d_j receives two or more streams, node g_j can only receive one stream because of its link capacity constraint $C(d_j, g_j)$=1.

We have assumed only one unit of bandwidth, that is, we have assumed

$K_m = 1$. If we define that the gain function $gain^1_{x_i}(g_j)$=1 for each pair of nodes x_i and g_j, and that $gain^1_x(y)$=0 for other pairs of nodes x and y, then we can conclude that the given Boolean expression g is satisfiable *iff* the total sum S of the gain functions is the number of clauses in g (for the example in Fig. 4.5, S=3). We can check whether the total sum S is equal to the number of clauses in g by calculating the value of the object function for the ILP problem in Fig. 4.4. This shows that the MGMS problem is NP-hard.

4.2.3 Dynamic MGMS Problem

Considering the NP-hardness of the MGMS problem which does not consider join/leave operations of user nodes, we re-formulate this MGMS problem into more realistic one called *dynamic MGMS* problem.

The dynamic MGMS problem is defined as follows. We say that a 3-dimensional matrix A that assigns 0 or 1 to each tuple of $V \times K \times E$ ($A : V \times K_{max} \times E \rightarrow 0/1$ where $K_{max} = \max_{m \in V} K_m$) is *consistent iff* the value assignment to $\pi^k_m(i,j)$ by A satisfies all the constraints of the MGMS problem. The inputs of dynamic MGMS problem are the same as the MGMS problem plus the followings;

- a node $v \in V$,
- a path $P(m', v) = \{(m', a), (a, b), ..., (z, v)\}$ on G and
- a consistent matrix A which assigns 0/1 values to $\pi^k_m(i,j)$, provided that A satisfies the followings: (i) node m' receives stream m using at least k units of bandwidth, (ii) node v receives stream m using $k-1$ units of bandwidth and (iii) stream m uses $k-1$ units of bandwidth on the path $P(m', v)$.

The output is a consistent assignment A' provided that (i) node v receives stream m using k units of bandwidth and (ii) for each tuple of $i \in V$, $k_i \leq K_i$ and $e \in E$ where e does not include any link on $P(m'v)$ as its upstream link, $A'(i, k_i, e) = A(i, k_i, e)$. Also A' must be optimal in the sense that it maximizes the objective function (4.1), and the value of the objective function for A' must be larger than that for A.

Intuitively, the dynamic MGMS problem is to find the optimal way to keep the k-th unit of bandwidth for node v to receive stream m from m', which has

already received stream m at least k units of bandwidth. We allow to take a unit of bandwidth from streams on the path $P(m', v)$ for the purpose. To find an optimal A', we have to minimize the loss of gain (negative gain) by taking the bandwidth from the existing streams.

Fig. 4.2(b) shows a very simple example after applying the dynamic MGMS problem to Fig. 4.2(a). For simplicity, we omit concrete values of gains, but the situation is as follows. If node b is the node which requests the first unit of bandwidth for stream c (with higher user gain, for example) on the path $P(a, b) = \{(a, b)\}$, node a degrades the stream a from two to one in forwarding it to nodes b. As a result, node b has a unit of bandwidth on the path $P(a, b)$, and can use this for stream c (this is the positive gain of the dynamic MGMS problem). Node a degrades the bandwidth of stream c from two to one to send it to node b (this is the negative gain). We need to minimize the negative gain to obtain an optimal solution of dynamic MGMS problem.

We implement a polynomial time algorithm to solve the dynamic MGMS problem in Emma/QoS protocol in a fully-decentralized manner. This is presented in Section 4.4.

4.3 Basic Operations of Emma/QoS

4.3.1 Overlay Network Construction

The overlay network topology affects the overall performance of the protocol running on it. Our protocol constructs mesh-like topology as an overlay network G incrementally as nodes join, like many other overlay multicast protocols. Over the overlay network G, we present in the next section how the dynamic MGMS problem can be solved in a distributed way. In this section we present how G is constructed.

Due to the today's high-speed wired/wireless LANs technology such as IEEE 802.3z (Gigabit Ethernet) and IEEE802.11a, users may not care for bandwidth overhead between end hosts in the same LAN (or closer LANs) but may care for relaying delay at end hosts. On the other hand, bandwidth overhead is still applicable between end hosts in LANs which are far from each other on the Internet. Considering this fact, we let the protocol to construct two-tier overlay networks. In the network, nodes in the same or closer LANs can connect to

each other with higher probability to prevent streams from going through many end hosts. This dense node group is called *domain*. On the other hand, nodes in different domains connect to each other with lower probability. The other nodes which do not have direct connections to the other domains sends/receives streams through the nodes which have direct connections to the other domains. To avoid many hops inside the domain, we let a node in a domain connect to some nodes in other domains if it knows that it cannot go outside the domain via a certain number of the other nodes in the same domain.

For this purpose, we let some nodes be a master controller and a domain controller in each domain. The master controller knows the IP addresses of domain controllers and each domain controller knows all the IP addresses of nodes in the domain. Both master and domain controllers can be end hosts. If a node wants to join a group, it first sends an inquiry to the master controller and gets the list of current domain controllers. Then the node measures round-trip delays to the domain controllers and determines whether it should belong to one of the domains or should become a new domain controller. Once it determines to belong to a domain, it contact with the domain controller to obtain the list of current nodes in the domain. Also, if the node needs to have connections to the other domains, it asks the domain controllers of the domains to obtain some candidate nodes in the domains. If the node needs to be a domain controller, it registers itself to the master controller.

We note that if a domain controller node leaves the overlay network, its neighboring nodes broadcast a delegation message within the group so that some other node can become a domain controller.

4.3.2 Route Query and Bandwidth Request

For a node v which wants the first unit of bandwidth for a stream m, Emma/QoS first searches the potential routes to nodes on demand which are currently receiving the stream m. This is done by flooding *route query* messages.

Each node which receives a route query message relays it to the rest of the neighbors, calculating the dynamic MGMS problem on each message path. This algorithm will be presented in Section 4.4.

When a message finds a node m' which is currently receiving the stream m, node m' knows the solution of dynamic MGMS problem to keep a unit of

bandwidth on the message path from m' to v, say $P(m', v)$. Then m' returns a *route reply* message to v along the reverse path. Node v may receive several route reply messages each of which includes a potential route and the corresponding dynamic MGMS solution to obtain a unit of bandwidth on the path. Then node v selects the one (i) whose path delay from m is less than D_{bound}, (ii) the value of loss of gain (called *negative gain*) is the minimum and (iii) the negative gain is less than node v's gain to have the requested bandwidth. Node v sends a *bandwidth request* message along the selected path. The intermediate nodes forwards the request message, and node m' sends back a *bandwidth reply* message that lets the intermediate nodes on the path follow the decision by the calculation results of the corresponding dynamic MGMS problem. Finally, the unit of bandwidth is given on the path $P(m', v)$ for stream m.

We note that for two-tier networks, we can take the following improvement which prevents route query messages from being flooded all over the overlay network. At the first step, the route query messages are exchanged only inside the domain. If a node currently receiving the requested stream is found inside the domain, it is the same case as we presented above. If not, nodes which have links to the other domains send reply to the sender of the route query messages which tells that it can send the query message to the outside of the domain. The sender node then selects the best one with respect to delay to the node, and asks him to find a route to receive the stream outside the domain. This two-step query is expected to reduce redundant route query messages.

If node v needs an additional unit of bandwidth for the currently receiving stream m, node v directly sends a bandwidth request message to obtain the unit of bandwidth on the existing route. In this case, the calculation of dynamic MGMS problem is done along with this bandwidth request message.

4.3.3 Leaving and Failure Management

If a node leaves an overlay network, the overlay network can be repaired automatically so that the separated streams can be connected again. We take a simple approach for such a situation.

Let us assume that a node u leaves an overlay network. Let v denote each neighboring node which has detected node u's leaving. For each stream m delivered via node u, node v decides an alternative node which can forward

stream m and then connects itself to the node. Accordingly, the descendant nodes of v can continue to receive the stream m.

4.4 Decentralized Algorithm for Dynamic MGMS Problem

For a node v which requests an additional k-th unit of bandwidth for a stream m, Emma/QoS tries to find a unit of bandwidth on the path with minimum negative gain. In this section, we present how this is realized on a given overlay network G in a decentralized way.

Hereafter, a node v is said to be a *descendant of a node u on stream m iff* node v receives stream m through node u. In opposite, node u is said to be an ancestor of node v on stream m.

4.4.1 Periodical Collection of User Gain

Each node v periodically sends a status report called a *MEDIA/Keep* message for each stream m which node v receives. A MEDIA/Keep message includes an *aggregated negative gain*. An aggregated negative gain, say $gain_m^k(v)^-$, is the sum of negative gain of node v and its descendant nodes on stream m when a unit of bandwidth is taken from stream m at node u, where node u is currently using k units of bandwidth and the parent node of node v. This value is used for calculating dynamic MGMS presented later. Obviously, if MEDIA/Keep reports are sent from the leaf nodes toward the root node m, aggregated negative gain can be calculated in a recursive way at each node as follows.

Assume that v receives a MEDIA/Keep report from each child node (say w). If the MEDIA/Keep report from each w includes the aggregated negative gain $gain_m^{h'}(w)^-$ for each $h' \geq h$ where h denotes the units of bandwidth that stream m currently uses on link (v, w), v can calculate its aggregated negative gain $gain_m^{k'}(v)^-$ for each $k' \geq k$ where k denotes the units of bandwidth that stream m currently uses on link (u, v);

$$gain_m^k(v)^- = -gain_m^k(v) * \pi_m^k(u, v) + \sum_w gain_m^k(w)^- \tag{4.11}$$

Then v sends a MEDIA/Keep report with $gain_m^{k'}(v)^-$ for each $k' \geq k$ to its parent node u. Note that MEDIA/Keep reports have another role to tell "keeping

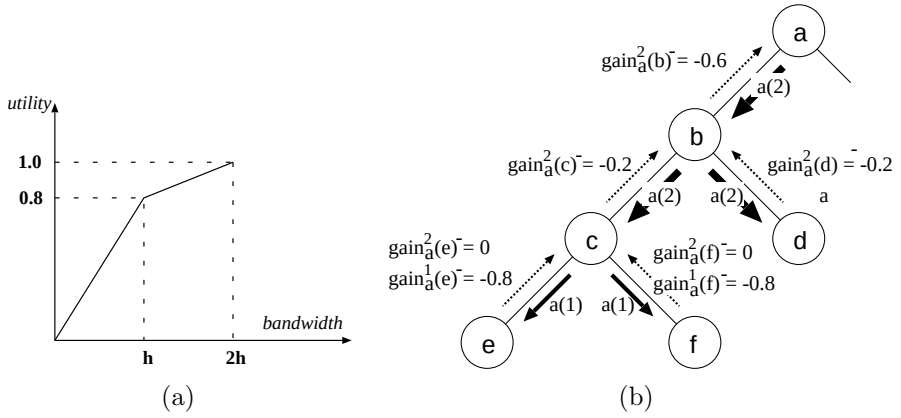


Figure 4.6: Forwarding MEDIA/Keep messages

alive" to upper nodes, as well as collecting negative gain.

As an example, we assume that a utility function in Fig. 4.6(a) is used for stream a by all the nodes. In Fig. 4.6(b), stream a is sent to nodes $b..f$ where node c degrades the bandwidth from two to one. Node c receives from nodes e two negative gains $gain^2_a(e)^- = 0$ and $gain^1_a(e)^- = -gain^1_a(e) = -0.8$, and also from nodes f two negative gains $gain^2_a(f)^- = 0$ and $gain^1_a(f)^- = -gain^1_a(f) = -0.8$. Then node c calculates the aggregated negative gain $gain^2_a(c)^- = -0.2 + 0 + 0 = -0.2$ and includes it to the MEDIA/Keep report sent to the upper node (the node b). Similarly, node b calculates $gain^2_a(b)^- = -0.2 + (-0.2) + (-0.2) = -0.6$ and sends it to node a.

4.4.2 Calculation of Dynamic MGMS Problem

As stated in Section 4.3.2, dynamic MGMS problem is calculated from a node (say r) which requests bandwidth for stream m on $P(m', r)$ in forwarding a route query message or a bandwidth request message from r to m'. In this section, how this calculation is done.

Let us assume that a node v on $P(m', r)$ receives a route request or a bandwidth request message from its child node w (also on $P(m', r)$). Here, let $optgain_i(v)^-$ denote the minimum negative gain to keep one unit of bandwidth on $P(v, r)$, in case that node v's parent node u on $P(m', r)$ removes one unit of bandwidth from stream m. $optgain_i(v)^-$ is defined as follows.

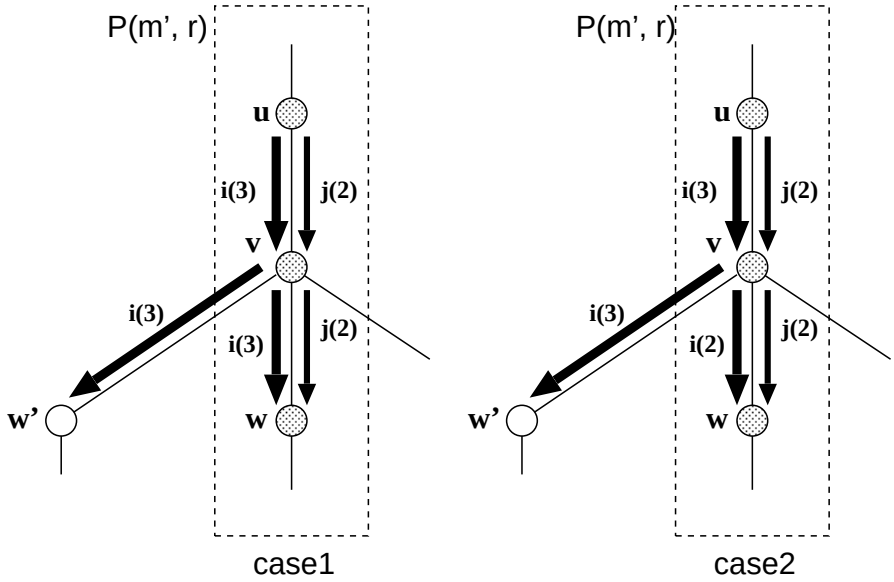


Figure 4.7: Calculation of $optgain_i(v)^-$

$optgain_i(v)^-$

$$= \begin{cases} -gain_i^k(v) + \sum_{w'} gain_i^k(w')^- \\ +optgain_i(w)^- \\ \quad (case1 : \text{if } \pi_i^k(v,w) = 1) \\ \\ -gain_i^k(v) + \sum_{w'} gain_i^k(w')^- \\ +min_j \{optgain_j(w)^-\} \\ \quad (case2 : \text{if } \pi_i^k(v,w) = 0) \end{cases}$$

Fig. 4.7 shows the concept of $optgain_i(v)^-$. Each node w' and its descendant nodes are not concerned with the request for stream m. However, they are receiving stream i and thus suffer negative gain $gain_i^k(v)^-$ if node u takes the k-th unit of bandwidth from stream i. In case 1, the k-th unit of bandwidth is also allocated to stream i on link (v, w). In this case, node w and its descendant nodes also suffer negative gain $optgain_i(w)^-$ since taking the k-th unit of bandwidth from stream i at node u consequently takes the k-th unit from stream i on link (v, w) as well. On the other hand, in case 2, there are some possibilities from which we should take a unit of bandwidth on link (v, w). To make the negative gain minimum, node v determines to take a unit of bandwidth from the stream j that minimizes the negative gain $optgain_j(w)^-$. Consequently, in forwarding the message to m', this calculation is carried out using the negative

gain collected by the MEDIA/Keep reports and for each $optgain_i(w)^-$ included in the route query or bandwidth request message from w to v. Finally node m' knows the solution of dynamic MGMS problem.

4.4.3 Complexity

The optimality of the algorithm is clear if we see the recursive definition of $optgain_i(v)^-$. Here, we show the complexity of the algorithm above.

We denote the number of links on the path $P(m', r)$ by M. For every link on $P(m', r)$, we check for every stream i $optgain_i(w)^-$ to find minimum negative gain in case 2. Then the complexity of the algorithm is $O(MN)$, which is very simple. We will show in the next section the iterative application of dynamic MGMS problem is very effective to improve the user gain even though it is the very simple re-formulation of MGMS problem.

4.5 Some Design Issues

4.5.1 Overlay Link Capacity

It is not easy to determine the overlay link capacity, because of the difficulty to estimate end-to-end bandwidth. Therefore, each node first determines the link capacity based on its computing capability, access bandwidth through its network interface and so on. After streaming starts, nodes can measure packet loss rates of each stream and can apply a rate adaptation method using transport layer protocols such as RTP/RTCP.

4.5.2 Policy Management

Greedy users may submit invalid gain values in requesting bandwidth or reporting status. In order to cope with the problem, a possible solution is that neighboring nodes monitor the submitted values, and check the validity of the submitted values. We now try to incorporate this mechanism into MEDIA/Keep reports.

4.5.3 Computing/Communication Complexity

In order to show the efficiency of Emma/QoS, analyzing complexity of the protocol is important. Here, we look at the time and space complexity at a node. In

forwarding a MEDIA/Join request, each node requires at most $O(k)$ time where k is the maximum overlay link capacity since it requires k stream comparison in computing minimum loss. Also each node continuously keeps loss values sent from its child nodes by MEDIA/Keep reports. These values require only $O(km)$ space where m is the maximum degree of overlay links. So the complexity is reasonable enough.

4.6 Performance Evaluation

We have implemented a simulator of Emma/QoS using the object oriented scripting language *Ruby*, and evaluated its performance by examining some characteristics such as link/tree stresses, user gain and path stretch. We have compared our protocol with Narada[21] which is one of the most popular ALMs.

4.6.1 Measurement Items

In this section, we have examined the following items.

1. link stress: the number of copies of a single packet delivered on a physical link
2. path stretch: the ratio of the sum of unicast hops of the overlay links between two nodes to that on the shortest path on the underlying physical network.
3. link utilization: the ratio of the current link utilization for each overlay link capacity.
4. tree stress: the number of trees on an overlay link.
5. user satisfaction ratio: the ratio of the sum of user gain obtained by each node to that of user gains requested by the node.
6. variation of user gain for a given application scenario.
7. the sum of all the users' gain which represents their total satisfaction.
8. network utilization ratio for adaptation.

In general, in order to evaluate the usefulness of a given ALM, the quality of the obtained delivery trees is important. The link stress and path stretch are considered as the important evaluation items in many ALMs, and it is known that they have some correlations between them. If the link stress becomes high, the path stretch becomes short in most cases, and it comes close to unicast. In order for a given ALM to be useful, we need to achieve well-balanced link stress and path stretch. Here, we have compared our protocol with unicast and Narada [21].

In Emma/QoS, a delivery tree is constructed for each source, and multiple delivery trees share the overlay links. On the other hand, most ALMs use a common delivery tree and multiple sources share it. Therefore, the tree stress is the original measure of Emma/QoS. Emma/QoS aims at avoiding the overlap of multiple delivery trees by allocating different routes to those trees and/or adjusting transmission rates of streams. If suitable load balance is achieved by Emma/QoS, we can obtain a reasonable tree stress for each link. We have evaluated the variation of this value when multiple leave/join operations are repeated in order to show that the tree stress is not so high even when many leave operations are carried out.

The user satisfaction ratio and the variation of user gain are also the original evaluation items of Emma/QoS, which uses the notion of the user satisfaction. In Emma/QoS, even if a user requires an additional stream's reception in the middle of a session, it provides a flexible admission control mechanism. It is the most characteristic feature of Emma/QoS. We have measured those values for checking how this control mechanism can work well. We have also compared our protocol with Narada. Narada does not assume the indication of multiple streams simultaneously, however, we can assume that it can drop packets at relay hosts and degrade bandwidth of the streams when enough bandwidth cannot be obtained. In our measurement, we assume that each relay node uses this mechanism, and we have implemented Narada's mechanism as follows; it degrades bandwidth of all the streams at the relay node and accepts a request for a unit of bandwidth when there is not a unit of bandwidth. We have also evaluated the user satisfaction by using the utility function explained in Section 4.2.1.

In order to evaluate the scalability of our protocol, we have measured the variation of the user gain changing the number of user nodes.

4.6.2 Simulation Scenario

In this section, we have used the following scenario for the evaluation. We have constructed a network based on a hierarchical topology model called tiers model [25] consisting of LAN, MAN and WAN where there are about 400 to 2,000 routers on the network. We assume that bandwidths of LAN, MAN and WAN are 6Mbps, 50Mbps and 100Mbps, respectively. This setting means that bandwidth of the links nearby the user nodes may become the bottleneck. In this setting, we use the notion of [26] and make the packet loss happen for each end-to-end link. Moreover, we have set the maximum bandwidth of each stream as 1Mbps, and have assumed that each node can filter a stream by every 0.2 Mbps units. That is, one unit of bandwidth is 0.2 Mbps. We have constructed an overlay network by about 60 to 1,000 overlay nodes where each node has three overlay links when it joins the session. If the three overlay links use the maximum bandwidth, it corresponds to use a half of LAN's link capacity. The user gain function for each stream is specified based on the function defined in Section 4.2.1. We have restricted the maximum number of hops to four (maximum delay bound). We have prioritized each request of bandwidth based on QoS ticket model in [27] where the allotted points are divided into the requests depending on the users' priority request to streams. In our experiment, each user has 70 points at the beginning, and divides it to the request of streams based on Zipf's law[28]. This was intended to reflect users' preference to different streams as stated in Section 4.2.1.

We have assumed a typical video conferencing scenario which consists of four phases: join (time 0-10), request (time 10-15), change of priority (time 15-20) and leave (time 20-30). In the join phase, about 50% of all the nodes joined the session. In the request phase, 50% of the joined nodes sent request in a certain probability. In the following change of request phase, each node changes the priority to streams, and finally in the leave phase, almost all the current node left the session. In Fig. 4.9, the variation of the number of users is shown as a bar chart. For other figures, we also show it as a bar chart.

4.6.3 Implementation of Narada

In order to compare our protocol with Narada, we have implemented Narada's mechanism as follows.

Each node chooses k nodes randomly from its node list, and sends connection requests. If the request for a node is accepted, it constructs an overlay link to the node. This overlay network is refreshed periodically. At first, each node periodically calculates the hop counts on its physical network for all the other members. Each member receives a copy of the routing table from its each neighboring node. Using this table, it can calculate the hop counts for the other nodes. Two nodes may have an overlay link if the hop count between them is greater than a certain threshold. Also, if the number of routes using this link is less than a threshold, the link is deleted.

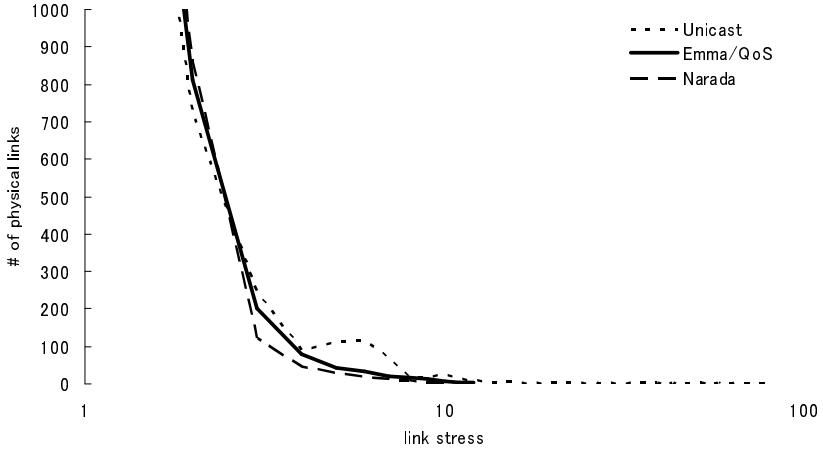
Narada constructs trees on this overlay network. Narada selects the shortest path from the paths where residual bandwidth exceeds a threshold, and connects it to the existing tree. Each stream is delivered on this tree. If it is difficult to relay the stream with the same quality as the received one due to the capacity constraints, then we let Narada takes one unit from the received stream and then delivers it.

If a node wants to leave a session, then the node informs it to its neighbor nodes. Each node receiving this message informs it to the other nodes along with the overlay network, and re-calculates new routes which do not pass the leaving nodes.

4.6.4 Evaluation of Link Stress and Path Stretch

We have measured link stresses and path stretches at time 15, just before the change of priority phase. Around this time, the number of nodes on the overlay network is relatively stable.

We show the distribution of link stresses in Fig. 4.8. Here, x-axis represents the values of link stresses in the logarithmic scale, and the y-axis represents the number of physical links which have the corresponding link stress on x-axis. We have shown the values of unicast and Narada in the figure to compare the performance. We see that Emma/QoS has better values than the unicast. The maximum link stress of Emma/QoS is about a tenth of the unicast. In Narada, the overlay network is constructed by choosing nodes randomly, and the shortest



598 users (1898 node networks)

Figure 4.8: Distribution of link stresses

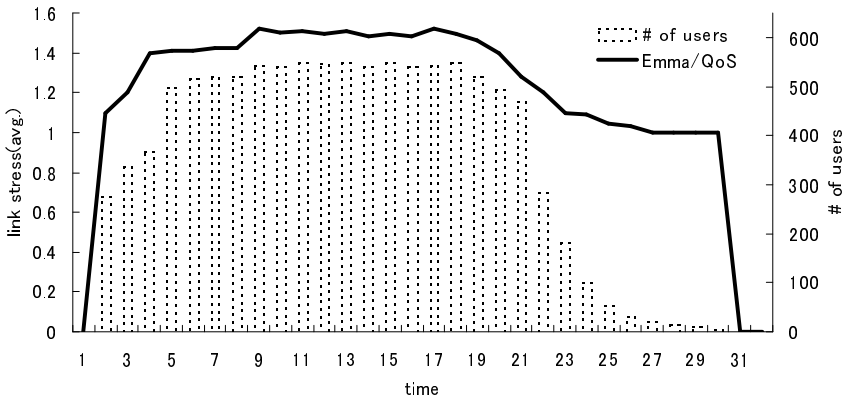
widest path tree is constructed on the network. Therefore the performance of Narada is not so quite different from that of Emma/QoS. Then we can see in Fig. 4.9 the distribution of link stresses. x-axis represents the value of the link stress and y-axis represents the number of links which have the corresponding link stress. We can see that the link stress follows the number of nodes and does not show extreme values.

Next, we show in Fig. 4.9, the variation of average link stress when applying the scenario. Here, x- and y-axes mean the simulation time and the average link stresses, respectively. From the figure, we see that link stresses do not become extremely large even when the number of nodes largely changes.

Fig. 4.10 shows the distribution of path stretches where x- and y-axes represent the value of the path stretches and the number of paths which have the corresponding path stretches. Fig. 4.11 shows the variation of path stretches when applying the scenario.

4.6.5 Evaluation of Link Utilization

In Emma/QoS, routes are determined on demand. Therefore, we can know the efficiency of the trees by investigating whether each overlay link is included in



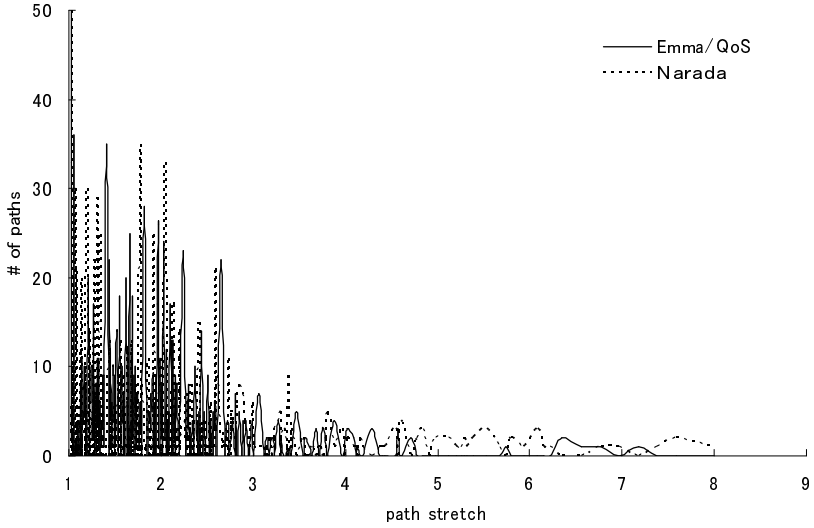
598 users (1898 node networks)

Figure 4.9: Variation of link stresses

the tree without bias. We call this usage of each overlay link as *link utilization*. At time 15 before the request phase, we have measured the link utilization of each overlay network. The result is shown in Fig. 4.12. We can see that the number of links with full link utilization is least within other plotted utilization ratios. This means that we do not have to execute so many dynamic MGMS problems which results in lower overhead.

4.6.6 Evaluation of User's Satisfaction

We define the *users' satisfaction* as the ratio of the total gain of a user to that of all requests. Fig. 4.13 depicts the distribution of the values. Here, x- and y-axes represent the users' satisfaction and the number of users which have the satisfaction. We have compared the result with that of Narada. As another index, we also used FCFS (First Come First Serve) which accepts the requests in order of their arrival as long as available capacity on overlay links can accommodate the requests. In Emma/QoS, the number of users whose requests are not accepted at all is much smaller than Narada and FCFS, since Emma/QoS can reduce the transmission rates of the streams with less priority. In video-conference systems, unlike simple video streaming systems where a single high quality video is requested, users are likely to request multiple videos even if those videos' quality is reduced. In this respect, the admission control



598 users (1898 node networks)

Figure 4.10: Distribution of path stretches

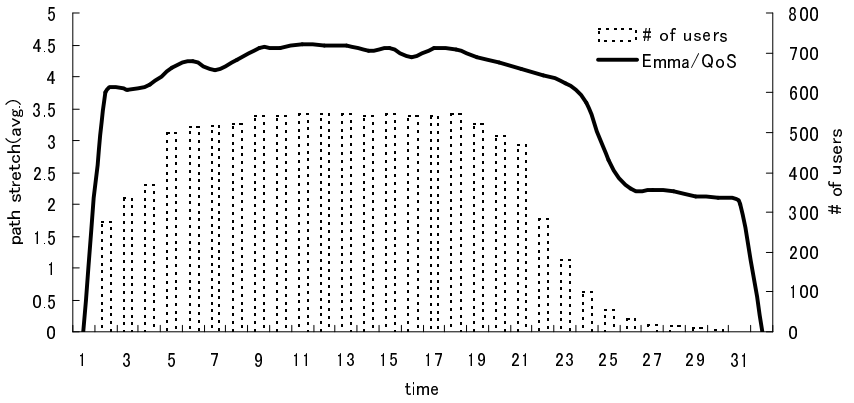
mechanism of Emma/QoS is useful to such an application system.

4.6.7 Measuring Distribution of User Gain

We have measured the variation of the average user gain in the simulation time. The result is shown in Fig. 4.14. x-axis represents the simulation time, and y-axis does both the average user gain and the number of users at the time. We used Narada for comparison. From this result, we see that Emma/QoS achieves much higher user gain than Narada for both cases: (i) when users join/leave during the session, and (ii) when user preferences to videos are changed during the session. This was achieved by the stream control mechanism of Emma/QoS which is based on priorities. Consequently, we see that this mechanism gives higher satisfaction degrees than existing methods.

4.6.8 Overhead Caused by Many Nodes

In order to evaluate the scalability of Emma/QoS, we have measured the total user gain when the number of nodes increases. We compared the result with Narada. Here, we conducted simulations by changing the number of nodes 33,



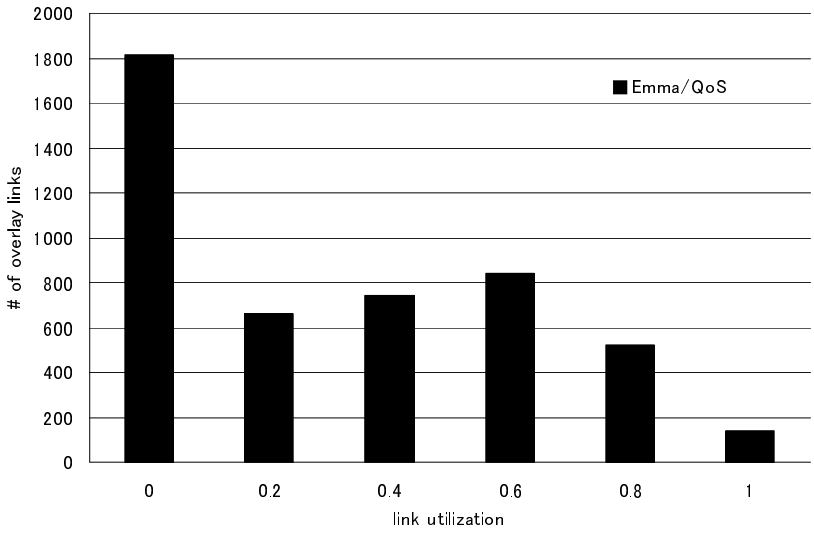
598 users (1898 node networks)

Figure 4.11: Variation of path stretches

149, 323, 598 and 987. We show the total sum of user gains in Fig. 4.15. We compared the result with Narada and FCFS. From this result, Emma/QoS still dominates other methods w.r.t. the total sum of user gain as the number of nodes increases. We see that the stream control mechanism of Emma/QoS is useful.

4.6.9 Link Adaptation

Finally, in order to evaluate adaptability of Emma/QoS against link bandwidth variation, we have measured the link usage and the variation of the total user gain when the available bandwidth on overlay links changes. Here, we have chosen one overlay link and have generated congestion on the link. We measured the total user gain applying the above scenario. The result is f in Fig. 4.16 and compared the result with Narada. Here, x-, $y-1$- and $y-2$-axes represent the simulation time, the total user gain after adaptation was done, and the network usage, respectively. Narada responds to the link variation by degrading the quality of streams, while Emma/QoS degrades the quality of streams gradually when the available bandwidth is not enough for the streams, and improves the network usage by adding bandwidth unit when the available bandwidth increases. From this result, Narada takes some time to restore the quality of streams after the available bandwidth has been restored. In Emma/QoS, the



598 users (1893 node networks)

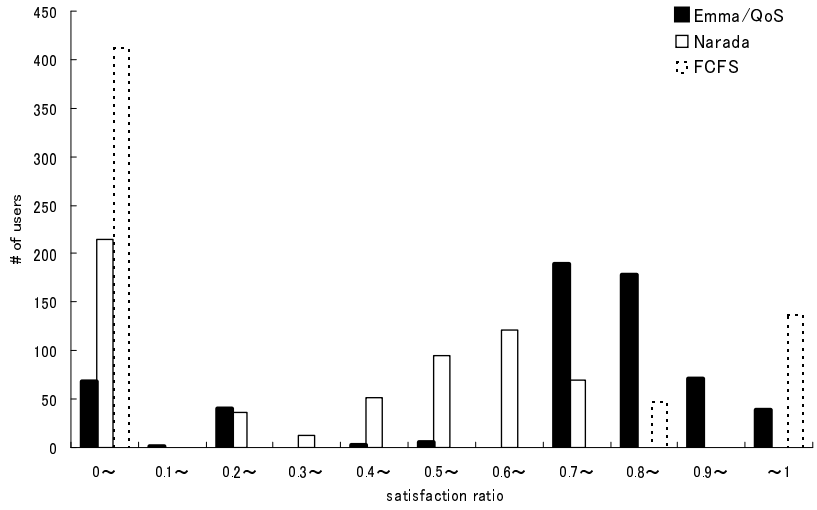
Figure 4.12: Distribution of link utilization

quality of streams is restored much more quickly than Narada.

4.7 Conclusion

In this chapter, we have proposed a QoS-aware ALM protocol called Emma/QoS for providing QoS in simultaneous transmission of multiple video sources on overlay networks in a fully distributed manner.

Overlay multicast is considered efficient and many literatures have shown the efficiency of overlay multicast especially in practical aspects for such a situation in group communication that the size of a group is not so large but there exist a large number of groups. Traditional server-based and unicast-based solutions may not work well in this situation. However, bandwidth on overlay networks is limited by not only end-to-end bandwidth but also the bandwidth of local area network, while recent progress of end hosts' computing performance, computing capabilities have been improved but are not fully utilized in most network applications. The idea in this chapter is that using such computing power of end hosts, we provide QoS on overlay network. For this purpose, Emma/QoS has been designed to control multiple video stream using application specific

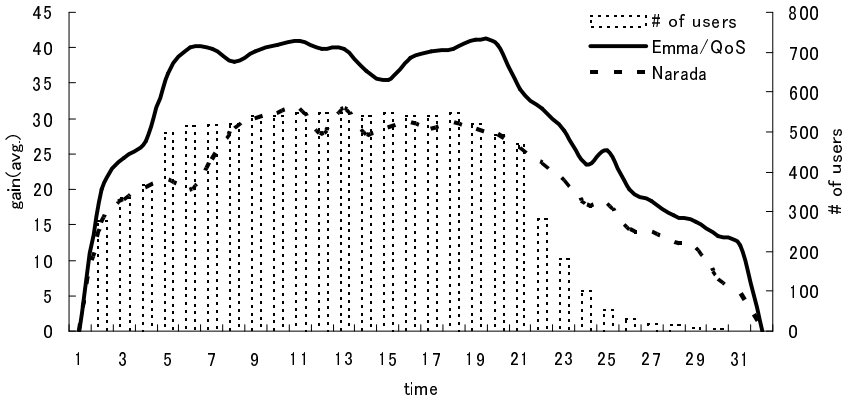


598 users (1898 node networks)

Figure 4.13: Distribution of users' satisfaction

parameters.

We are now designing and implementing Java middleware based on Emma/QoS presented in this chapter.



598 users (1898 node networks)

Figure 4.14: Variation of user gain

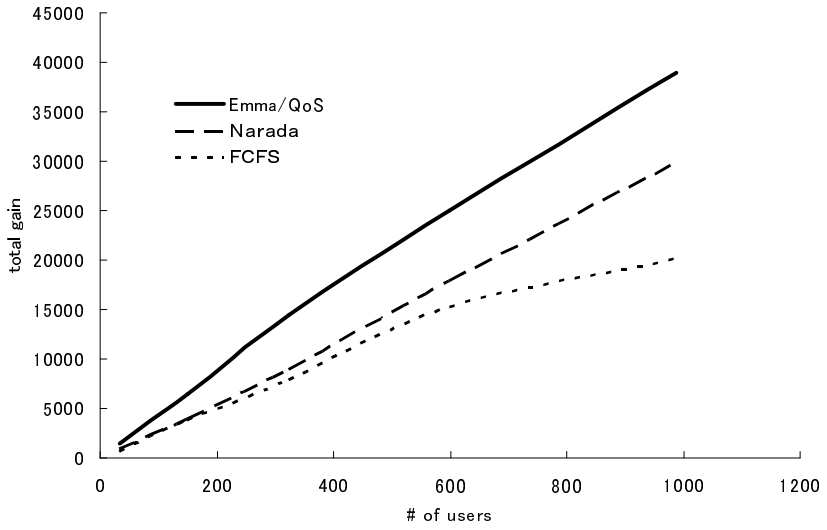


Figure 4.15: Total sum of user gain

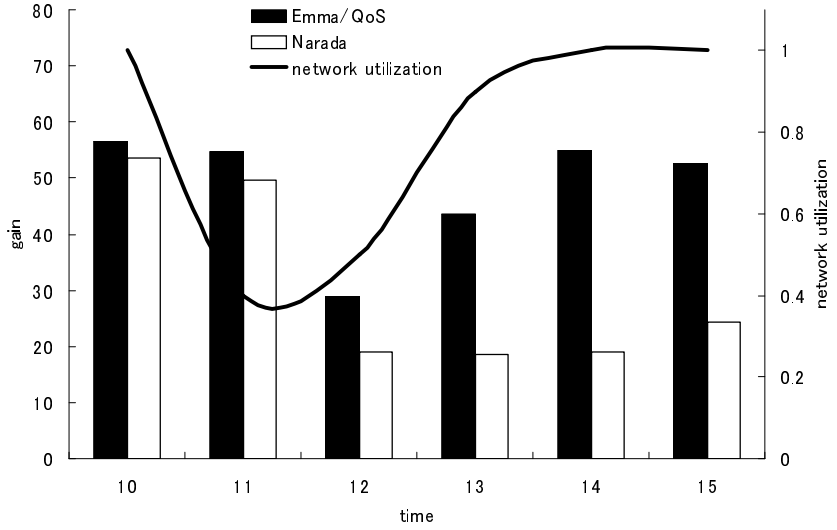


Figure 4.16: Link adaptation

Chapter 5

Conclusion

In this thesis, we have studied the following two research topics.

1. A method to achieve secure multiple association that considers indirect connection when multiple VPNs are constructed on the same VPN architecture

2. A method to achieve application layer multicast delivered maximizing user's priority of communication quality

In Chapter 2, we present related work of this thesis. First, some related work about VPN association control is mentioned. Next, we describe some ALM protocols, and are pointing out their problem.

In Chapter 3, we have proposed a VPN association control protocol that dynamically controls a multiple association by efficiently collecting the policies regulated by each VPN and the information about indirect connection of multiple association, and judging the acceptance of association according to that information on existing VPN architecture. In the proposed method, the architecture of VPN must be composed of the provider edge router (called PE) connected with the provider network and the customer edge router (called CE) prepared on the site side on the network that the service provider provided. And the communication of VPN can use any existing protocol.

We define that an association policy for a site is a set of other sites that do not want to be connected indirectly in addition to the existing VPN association policy. If the policy of all VPNs that has the connection relationship is not efficiently judged when the scale of VPN grows, the memory area which manages

the collection time, the bandwidth, and control information needs very large area, and lacks the scalability. Then, in this method, policies collected from all VPNs are limited only about the connection relationship among VPNs. This information of connection of each VPN is maintained by each PE. And the overlay network (called PE-graph) for collecting the information that consists only of PE with information on corresponding PE of each sets of VPN indirectly connected is constructed. Our proposed method can control a scalable association of VPNs in a scalable way by collecting this information by decentralized processing of each PE on this graph, and can resolve the conflicts of association requests to keep the consistency of the information.

In Chapter 4, we have proposed a protocol that that constructs the ALM delivery tree on overlay network and controls dynamically and in a decentralized way which video streams should be delivered and how much bandwidth they should be used under the given condition about the stream forwarding ability of each host, the overlay link capacity, and the priority request (preference) that each host has specified for target video streams on the network. For instance, the participant of the conference specifies higher preference for videos such as speakers and specifies comparatively low preference for videos such as other audiences in the video-conferencing system. When two or more video streams compete for the bandwidth of overlay network, proposed protocol guarantee the delivery quality of the video stream with higher preference by decreasing the quality of the video stream with lower preference. Moreover we assumed that each end host has the function to adjust the transfer rate when the video is forwarded to other end hosts by multicast. By using the function, the proposed protocol implements the admission control to accept the receiving request of a new video stream by adjusting the transfer rate of an existing video stream cooperatively among the end hosts. The simulation experiment as a performance evaluation shows that the proposed technique had achieved higher users' satisfactory data delivery than existing methods.

As a future work about VPN multiple association control, we are planning to examine the method of dividing the PE-graph when the reachable VPN group is divided by sites' leaving and the method of optimizing the PE-graph.

For application layer multicast, we are planning to design and implement Java middleware based on Emma/QoS.

Acknowledgement

This work could be achieved owing to a great deal of helps of many individuals.

First, I would like to thank my supervisor Professor Teruo Higashino of Osaka University, for his continuous support, encouragement and guidance of the work.

I am very grateful to Professor Makoto Imase, Professor Koso Murakami, Professor Masayuki Murata and Professor Hirotaka Nakano of Osaka University for their invaluable comments and helpful suggestions concerning this thesis.

Many of the courses in Osaka University that I have taken during my graduate career have been helpful to write this thesis. I would like to acknowledge the guidance of Associate Professors Go Hasegawa, Hiroyuki Ohsaki, Hideki Tode and Naoki Wakamiya.

I would like to express my thanks to Associate Professor Akio Nakata of Osaka University and Associate Professor Keiichi Yasumoto of Nara Institute of Science and Technology who have provided many valuable comments.

I would like to thank Assistant Professor Hirozumi Yamaguchi of Osaka University for his valuable comments and discussions.

I am very grateful to Associate Professor Junji Kitamichi of Aizu University and Assistant Professor Takaaki Umedu of Osaka University for their insightful and constructive comments.

At last, I would like to thank all the members of Higashino Laboratory of Osaka University for their helpful advice.

Bibliography

[1] H. Hamed, E. Al-Shaer, and W. Marrero. Modeling and verification of IPSec and VPN security policies. In *Proceedings of the 13th IEEE International Conference on Network Protocols (ICNP 2005)*, pages 259–278, November 2005.

[2] O. Honda, H. Ohsaki, M. Imase, J. Murayama, and K. Matsuda. A prototype implementation of vpn enabling user-based multiple association. In *Proceedings of the Ninth IASTED International Conference on Internet & Multimedia Systems & Applications (IMSA 2005)*, pages 59–64, August 2005.

[3] S. J. Beak, M. S. Jeong, and J. T. Park. Policy-based hybrid management architecture for IP-based VPN. In *Proceedings of the 2000 IEEE/IFIP Network Operations and Management Symposium (NOMS 2000)*, pages 987–988, April 2000.

[4] F. Barrere, A. Benzekri, F. Grasset, and R. Laborde. A multi-domain security policy distribution architecture for dynamic IP based VPN management. In *Proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks (POLICY 2002)*, pages 5–7, June 2002.

[5] M. Beigi, S. Calo, and D. Verma. Policy transformation techniques in policy-based systems management. In *Proceedings of the 5th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2004)*, pages 13–22, June 2004.

[6] N.J. Yeadon, F. García, D. Hutchison, and D. Shepherd. Filters: Qos support mechanisms for multipeer communications. *IEEE Journal on Selected Areas in Communications*, 14(7):1245–1262, September 1996.

[7] K. Lougheed and Y. Rekhter. A border gateway protocol 'BGP'. *Network Working Group Request for Comments (RFC) 1105*, June 1989.

[8] A. Tanenbaum and M. Steen. *DISTRIBUTED SYSTEMS: Principles and Paradigms*. Prentice Hall, first edition, 2002.

[9] G. Tel. *Introduction to Distributed Algorithms*. Cambridge University Press, second edition, 2000.

[10] L. Lamport. Time, clocks, and the ordering of events in distributed systems. *Communications of the ACM*, 21(7):558–565, July 1978.

[11] J. Gray and A. Reuter. *Transaction Processing : Concepts and Techniques*. Morgan Kaufmann, 1992.

[12] C. Diot, W. Dabbous, and J. Crowcroft. Multipoint communication: A survey of protocols, functions, and mechanisms. *IEEE Journal on Selected Areas in Communications*, 15(3):277–290, April 1997.

[13] J. Jannotti, D.K. Gifford, K.L. Johnson, M.F. Kaashoek, and J.W. O'Toole, Jr. Overcast: Reliable multicasting with an overlay network. In *Proceedings of the 4th Usenix Symposium on Operating Systems Design and Implementation (OSDI 2000)*, pages 197–212, October 2000.

[14] P. Francis. Yoid: Extending the internet multicast architecture. Unrefereed report, April 2000.

[15] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel. ALMI: An application level multicast infrastructure. In *Proceedings of the 3rd Usenix Symposium on Internet Technologies and Systems (USITS '01)*, pages 49–60, March 2001.

[16] Y. Chawathe, S. McCanne, and E.A. Brewer. RMX: Reliable multicast for heterogeneous networks. In *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 2000)*, pages 795–804, March 2000.

[17] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Application-level multicast using content-addressable networks. In *Proceedings of the 3rd International Workshop on Networked Group Communication (NGC 2001)*,

number 2233 in Lecture Notes in Computer Science, pages 14–29. Springer-Verlag, November 2001.

[18] V. Roca and A. El-Sayed. A host-based multicast (HBM) solution for group communications. In *Proceedings of the IEEE International Conference on Networking (ICN '01)*, number 2093 in Lecture Notes in Computer Science, pages 610–619. Springer-Verlag, July 2001.

[19] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *Proceedings of ACM SIGCOMM 2002*, pages 205–217, August 2002.

[20] Y.-H. Chu, Sanjay G. Rao, and H. Zhang. A case for end system multicast. In *Proceedings of the ACM SIGMETRICS 2000*, pages 1–12, June 2000.

[21] Y.-H. Chu, Sanjay G. Rao, S. Seshan, and H. Zhang. Enabling conferencing applications on the internet using an overlay multicast architecture. In *Proceedings of the ACM SIGCOMM 2001*, pages 55–67, August 2001.

[22] K. Chandy and L. Lamport. Distributed snapshots : Determining global states of distributed systems. *ACM Transactions on Computer Systems*, 3(1):63–75, February 1985.

[23] D. Xu and K. Nahrstedt. Finding service paths in a media service proxy network. In *Proceedings of the ACM/SPIE Conference on Multimedia Computing and Networking 2002 (MMCN 2002)*, pages 171–185, January 2002.

[24] S. Shenker. Fundamental design issues for the future internet. *IEEE Journal on Selected Areas in Communications*, 13(7):1176–1188, September 1995.

[25] K.L. Calvert, M.B. Doar, and E.W. Zegura. Modeling internet topology. *IEEE Communications Magazine*, 35(6):160–163, June 1997.

[26] S. Yeung and J. Lehoczkey. End-to-end delay analysis for real-time networks. In *Proceedings of the 22nd IEEE Real-Time Systems Symposium (RTSS 2001)*, pages 299–309, December 2001.

[27] K. Kawachiya and H. Tokuda. Dynamic qos control based on the qos-ticket model. In *Proceedings of the 1996 IEEE International Conference*

on Multimedia Computing and Systems (ICMCS 1996), pages 78–85, June 1996.

[28] G.K. Zipf. Relative frequency as a determinant of phonetic change. Harvard studies in classical philiology, 1929.

[29] W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, and B. Palter. Layer two tunneling protocol 'L2TP'. *Network Working Group Request for Comments (RFC) 2661*, August 1999.

[30] L. Mamakos, K. Lidl, J. Evarts, D. Carrel, D. Simone, and R. Wheeler. A method for transmitting ppp over ethernet (PPPoE). *Network Working Group Request for Comments (RFC) 2516*, February 1999.

[31] S. Kent and R. Atkinson. Security architecture for the internet protocol. *Network Working Group Request for Comments (RFC) 2401*, November 1998.

[32] SUB-IP Area IETF, Concluded Working Groups. Provider provisioned virtual private networks (ppvpn). `http://www.ietf.org/html.charters/OLD/ppvpn-charter.html`.

[33] B. Gleeson, A. Lin, J. Heinanen, G. Armitage, and A. Malis. A framework for IP based virtual private networks. *Network Working Group Request for Comments (RFC) 2764*, February 2000.

[34] D. Kindred and D. Sterne. Dynamic VPN communities : Implementation and experience. In *Proceedings of the 2nd DARPA Information Survivability Conference and Exposition (DISCEX II)*, pages 254–263, June 2001.

[35] G. Perez and A. Skarmeta. Policy-based dynamic provision of IP services in a secure VPN coalition scenario. *IEEE Communication Magazine*, 42(1):14–16, November 2004.

[36] F. Clemente, G. Millan, J. Re, G. Perez, and A. Skarmeta. Deployment of a policy-based management system for the dynamic provision of IPsec-based VPNs in IPv6 networks. In *Proceedings of the 2005 Symposium on Application and the Internet Workshops (SAINT-W 2005)*, pages 10–13, January 2005.

[37] H. Wang, S. Jha, M. Livny, and P. D. McDanie. Security policy reconciliation in distributed computing environments. In *Proceedings of the 5th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2004)*, pages 137–146, June 2004.

[38] B. Moore and E. Ellesson. Policy core information model. *Network Working Group Request for Comments (RFC) 3060*, February 2001.

[39] B. Moore. Policy core information model (PCIM) extensions. *Network Working Group Request for Comments (RFC) 3460*, January 2003.

[40] D. Durham, J. Boyle, R. Cohen, S. Herzog, R. Rajan, and A. Sastry. The COPS (common open policy service) protocol. *Network Working Group Request for Comments (RFC) 2748*, January 2000.

[41] K. Chan, J. Seligson, D. Durham, S. Gai, K. McCloghrie, S. Herzog, F. Reichmeyer, R. Yavatkar, and A. Smith. The COPS (common open policy service) protocol. *Network Working Group Request for Comments (RFC) 3084*, March 2001.

[42] K. Feeney, K. Quinn, D. Lewis, D. O'Sullivan, and V. Wade. Relationship-driven policy engineering for autonomic organisations. In *Proceedings of the 6th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2005)*, pages 89–98, June 2005.

[43] C. Metz. The latest in virtual private network : Part i. *IEEE Internet Computing*, 7(1):87–91, January/February 2003.

[44] R. Yuan and W. T. Strayer. *Virtual Private Networks.* Addison Wesley, April 2001.

[45] C. Scott, P. Wolfe, and M. Erwin. *Virtual Private Networks, 2nd Edition.* Oreilly & Associates Inc., December 1998.

[46] B. Davie and Y. Rekhter. *MPLS: Technology and Applications.* Morgan Kaufmann Pub., May 2000.

[47] I. Pepelnjak, J. Guichard, and J. Apcar. *MPLS & VPN Architectures.* Cisco Systems, October 2000.

[48] E. Rosen and Y. Rekhter. BGP/MPLS VPNs. *Network Working Group Request for Comments (RFC) 2547*, March 1999.

[49] C. Semeria. RFC 2547bis: BGP/MPLS VPN fundamentals. *Juniper Netowrks, Inc. White Paper*, December 2000. `http://www.juniper.net/solutions/literature/white\_papers/200012.pdf`.

[50] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Pub., April 1997.

[51] Y.-H. Chu, Sanjay G. Rao, S. Seshan, and H. Zhang. A case for end system multicast. *IEEE Journal on Selected Areas in Communication*, 20(8):1456–1471, October 2002.

[52] V. Roca, L. Costa, R. Vida, A. Dracinschi, and S. Fdida. A survey of multicast technologies. Technical report, September 2000.

[53] R. Cohen and G. Kaempfer. A unicast-based approach for streaming multicast. In *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 2001)*, pages 440–448, April 2001.

[54] S.Q. Zhuang, B.Y. Zhao, A.D. Joseph, R.H. Katz, and J.D. Kubiatowicz. Bayeux: An architecture for scalable and fault-tolerant wide-area data dissemination. In *Proceedings of the 11th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2001)*, pages 11–20, June 2001.

[55] F. Baccelli, D. Kofman, and J.L. Rougier. Self organizing hierarchical multicast trees and their optimization. In *Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 1999)*, pages 1081–1099, March 1999.

[56] C.E. Luna, L.P. Kondi, and A.K. Katsaggelos. Maximizing user utility in video streaming applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(2):141–148, February 2003.

[57] K.C. Almeroth. A long-term analysis of growth and usage patterns in the multicast backbone (MBone). In *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 2000)*, pages 824–833, March 2000.

[58] P. Parnes, K. Synnes, and D. Schefstrom. mStar: Enabling collaborative applications on the internet. *IEEE Internet Computing*, 4(5):32–39, September/October 2000.

[59] S. McCanne and V. Jacobson. Vic: A flexible framework for packet video. In *Proceedings of the ACM Multimedia 1995*, pages 511–522, November 1995. `http://ee.lbl.gov/vic/`.

[60] V. Jacobson and S. McCanne. vat - lbnl audio conferencing tool, 1993. `http://ee.lbl.gov/vat/`.

[61] V. Jacobson. pathchar, 1997. `ftp://ftp.ee.lbl.gov/pathchar/`.

[62] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A transport protocol for real-time applications. RFC 1889, January 1996.

[63] D.G. Andersen, H. Balakrishnan, M.F. Kaashoek, and R. Morris. The case for resilient overlay networks. In *Proceedings of the 8th Annual Workshop on Hot Topics in Operating Systems (HotOS-VIII)*, pages 152–157, May 2001.

[64] D.G. Andersen, H. Balakrishnan, M.F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP 2001)*, pages 131–145, October 2001.

[65] L. Mingyan, R.R. Talpade, A. Mcauley, and E. Bommaiah. AMRoute: Adhoc multicast routing protocol. Technical Report 99-1, August 1999.

[66] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proceedings of the ACM SIGCOMM 2001*, pages 27–31, August 2001.

[67] J. Liebeherr and M. Nahas. Application-layer multicast with delaunay triangulations. In *Proceedings of the IEEE Global Telecommunications Conference (Globecom 2001)*, volume 3, pages 1651–1655, November 2001.

[68] J. Crowcroft and K. Carlberg. MAPEX: Application level multicast architectural requirements for apex. Internet draft: draft-crowcroft-apex-multicast-01, October 2001.

[69] J. Touch. Dynamic internet overlay deployment and management using the x-bone. *Computer Networks*, 36(2/3):117–135, July 2001.

[70] Gnutella.com. `http://www.gnutella.com/`.

[71] I. Clarke, O. Sandberg, B. Wiley, and T.W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Proceedings of the International Workshop on Design Issues in Anonymity and Unobservability 2000*, number 2009 in Lecture Notes in Computer Science, pages 46–66. Springer-Verlag, July 2001.

[72] jxta.org. `http://www.jxta.org/`.

[73] I. Stoica, R. Morris, D.R. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A peer-to-peer lookup service for internet applications. In *Proceedings of the 9th ACM International Multimedia Conference (ACM Multimedia 2001)*, pages 149–160, August 2001.

[74] I. Stoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions on Networking*, 11(1):17–32, February 2003.

[75] Akamai: The business internet. `http://www.akamai.com/`.

[76] B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th annual international conference on Mobile computingand networking (MobiCom 2000)*, pages 149–160, August 2000.

[77] Y. Chawathe, A. Schuett, E. Amir, S. Fink, T. Hodes, A. Huang, S. Raman, C. Romer, A. Swan, T. Wong, K. Wright, S. McCanne, and R. Katz. An active service infrastructure for real-time conferencing. In *Proceedings of the 6th ACM International Multimedia Conference (ACM Multimedia '98)*, September 1998.

[78] L. Wei and D. Estrin. The trade-offs of multicast trees and algorithms. In *Proceedings of the 1994 International Conference on Computer Communications and Networks (ICCCN '94)*, pages 17–24, September 1994.

[79] B.M. Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, February 1988.

[80] B.M. Waxman. Performance evaluation of multipoint routing algorithms. In *Proceedings of the 12th Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 1993)*, pages 980–986, March/April 1993.

[81] M. Parsa and J.J. Garcia-Luna-Aceves. A protocol for scalable loop-free multicast routings. *IEEE Journal on Selected Areas in Communications*, 15(3):316–331, April 1997.

[82] University of California Berkeley MASH Research Group. The network simulator ns-2, 2000. `http://www-mash.ce.berkeley.edu/ns/`.