Integrated Resource Allocation Scheme for Real-Time Video Multicast

Taketo YAMASHITA Naoki WAKAMIYA
Masayuki MURATA Hideo MIYAHARA

Department of Informatics and Mathematical Science,
Graduate School of Engineering Science, Osaka University
1–3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan

Tel: +81–6–6850–6588 Fax: +81–6–6850–6589
E-mail: tkt-ymst@ics.es.osaka-u.ac.jp

Abstract

To provide distributed multimedia applications with end-to-end QoS (Quality of Service) guarantees, resource reservation-based control mechanisms should be employed in both networks and end systems. In this paper, we propose a resource allocation scheme for real-time video multicasting. The resource allocation is described as a utility maximization problem, in which the utility is represented as a relationship between the benefit obtained through the allocated resources and the cost paid for them. In this scheme, clients are first devided into multicast groups by means of a clustering technique. Then system resources are allocated to each group so that the total utility is maximized. We have confirmed that our proposed scheme can achieve effective use of resources while providing high-quality video to users.

1 Introduction

Due to dramatic improvements in computing power, network bandwidth, and video data compression techniques, distributed and real-time multimedia systems are now widely used. In these systems, a server captures video data, then encodes and sends it to clients via networks. The clients receive the coded data, decode it, and display it to users. To provide the users with a high-quality multimedia presentation, QoS should be guaranteed in terms of the data transfer delay, and the regularity of the video encoding and decoding [1]. Resource reservation-based systems are effective in guaranteeing these types of QoS for video applications. Network level

QoS, such as packet loss ratio and transfer delay, can be statically guaranteed in bandwidth reservation-based networks [2]. A real-time OS that reserves and schedules CPU resources can provide high-speed, high-quality video coding and decoding on end systems [3, 4].

However, even if we can successfully build a distributed multimedia system by combining these platforms, high-quality, real-time video transfer cannot be achieved efficiently without appropriate prediction and reservation mechanisms for both the network and end systems resources. We have formulated the effect of the MPEG-2 coding parameters on the required resources and the video quality, and we found that there is a strong relationship between them [5]. For example, by employing a larger quantizer scale factor, the required amounts of both bandwidth and client CPU resources decrease while the required amount of server CPU resources does not change much. Based on such relationships, the resource allocation scheme proposed in [6] enables high-quality video transfer within limited resources by maximizing a user's "utility", which is represented as a relationship between the "benefit" obtained through allocated resources and the "cost" paid for them.

In this paper, based on our previous work, we propose a new resource allocation scheme for video multicast systems with heterogeneous clients. We take into account a network topology, including the locations of the server and clients and the link bandwidth. Our scheme first divides the clients into multicast groups based on the available amounts of CPU resources and the access link bandwidth. Next, the shared resources—i.e., the server CPU and the network bandwidth—are equally allocated to each multicast group. Within each multicast group, the resource allocation among system entities is performed by considering the relationships between the server CPU, the network bandwidth, and the client CPUs. The remaining resources are additionally allocated to the multicast group with the highest potential to increase the total utility. The multicast group then performs local resource allocation to maximize its utility. By iteratively repeating this two-level resource allocation, the total utility is maximized. Appropriate multicast groups are thus established, so that users are provided with a video stream of the highest possible quality.

This paper is organized as follows. In Section 2, we explain the QoS guarantee mechanisms

in resource reservation-based systems and briefly introduce the relationships between video quality and the required resources. In Section 3, we describe the multicast system we consider. We first describe our clustering and multicast tree construction methods in Section 4, then outline our resource allocation scheme in Section 5. In Section 6, we evaluate the effectiveness of the proposed scheme. We finally conclude our paper in Section 7.

2 Resource Reservation-based System

By combining a bandwidth reservation network and a real-time OS, a required QoS can be guaranteed for video applications. However, to provide high-quality, real-time video presentation to users, it is necessary to allocate resources efficiently for each entity in the whole system. If there are sufficient resources, every user can enjoy high-quality video presentation. In an actual situation, however, users compete for the limited bandwidth of the network resources. Server CPU resources are also competed for by the clients, and the client CPUs are shared by several tasks. In addition to the availability of the various resources, we should also consider the interdependence of these resources. By allocating much bandwidth, for example, a client can receive video data with a low compression ratio, which does not require complex, heavy decoding. In this section, we briefly introduce the relationships between video quality and the required network/end systems resources [5].

2.1 QoS guarantee on networks

Bandwidth reservation networks such as ATM, RSVP, or TTCP/ITM [4] can be used to achieve real-time video transfer with low delay jitter and no data loss.

On resource reservation networks, such network-level QoS can be guaranteed as long as the traffic rate never exceeds the reserved bandwidth [7]. If the traffic characteristics are known a priori and sufficient bandwidth is available, it is sufficient to reserve the bandwidth at the peak traffic rate. A precise estimator for the required bandwidth is thus necessary in this scenario. When the estimated bandwidth is less than the actual peak rate, the perceived video quality deteriorates due to packet losses. On the other hand, when the bandwidth reservation exceeding

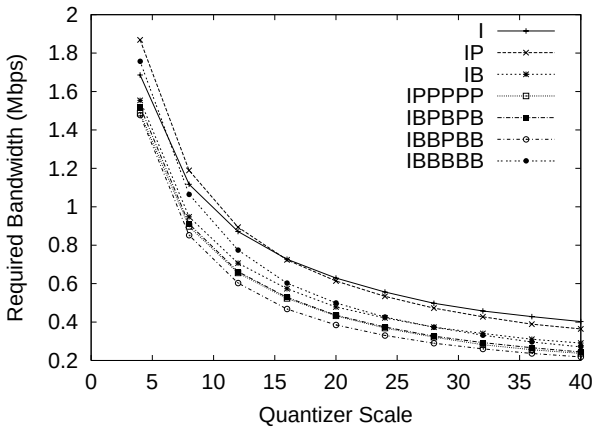

Figure 1: Relationships between video quality and required bandwidth

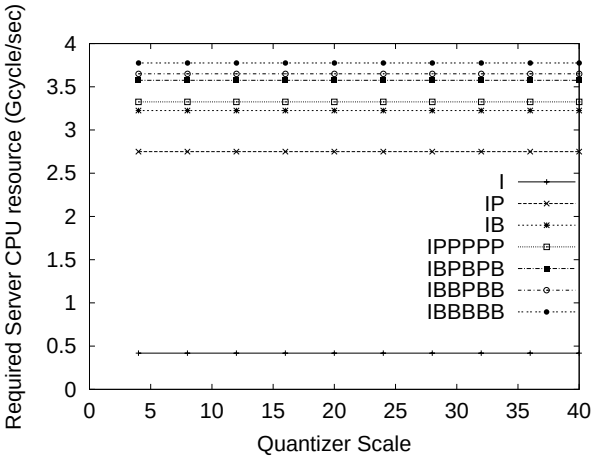

Figure 2: Relationships between video quality and required server CPU resource

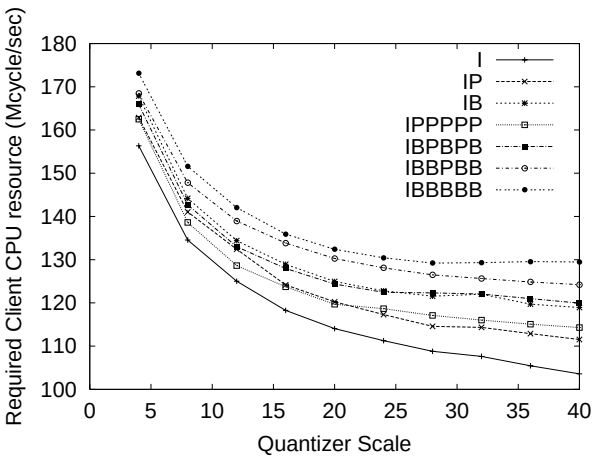

Figure 3: Relationships between video quality and required client CPU resource

the peak rate, this leads to a waste of network resources.

In a previous work [5], we derived relationships between the video quality and the required network and end systems resources for MPEG-2 video data. Examples of these relationships are shown in Fig. 1 for a video sequence called "Animation". The required bandwidth W Mbps for video transfer can be estimated from the MPEG-2 coding parameters, which directly affect the video quality. These parameters are the spatial resolution R pixels, SNR (Signal-to-Noise Ratio) resolution Q (i.e., the quantizer scale), temporal resolution F fps, and GoP structure G. The

estimator can be represented as:

$$W(R,Q,F,G) \cong 3.1^{\log_4 \frac{R}{640\times480}}(\alpha + \frac{\beta}{Q} - \frac{\gamma}{Q^2})\frac{F}{30}W_{base}\,, \tag{1}$$

where W_{base} is a constant value corresponding to the required bandwidth for the reference sequence with parameter set $(R,Q,F,G) = (640\times480, 10, 30, G)$. The constants α, β and γ can be determined for a given GoP structure.

By inversely applying the estimator, we can derive an appropriate set of coding parameters to make a given amount of video traffic fit the available bandwidth.

2.2 QoS guarantee in end systems

In end systems, the regularity of the video encoding/decoding process must be ensured by guaranteeing QoS for the maximum processing delay or the probability of deadline miss. To provide these types of QoS, sufficient CPU resources should be allocated to each task in the systems by means of a real-time OS, such as Real-Time Mach [8, 9] or HiTactix [4].

We have explained the relationship between video quality and required end system resources for MPEG-2 video data and proposed a good estimator in our previous work [5]. Examples of the relationships between video quality and the required server and client CPU resources for the video sequence "Animation" are shown in Figs. 2 and 3, respectively. First, the required amount of CPU resource to code video data at the server, S Mcycle/sec, can be estimated by using constant S_G, which depends on the GoP structure G as follows:

$$S \cong S_G \frac{R}{640\times480}\frac{F}{30}\,. \tag{2}$$

Next, the required amount of CPU resources to decode the video data at the client C Mcycle/sec is proportional to the required bandwidth W as follows:

$$C \cong W\times 40 + (\lambda + \frac{N_p}{N}\delta + \frac{N_b}{N}\varepsilon)\times\frac{R}{640\times480}\frac{F}{30}\,, \tag{3}$$

where N is the number of frames, and N_p and N_b are the numbers of P and B pictures in a GoP. λ, δ, and ε are the rates of increase in the amounts of CPU resources required to decode I, P, and

B pictures, respectively. For example, they were obtained as $\lambda = 870$, $\delta = 280$, and $\varepsilon = 420$ for the MPEG-2 video sequence "Scenery" in our previous research [5].

By using equations (1) through (3), we can estimate the required amounts of resources in both the network and end systems for real-time video transfer. Thus, given the amount of available resources, we can find an appropriate set of coding parameters to achieve a high-quality video stream with those resources.

3 Scenario of Video Multicast System

In this section, we briefly summarize how a video multicast is accomplished with our resource allocation scheme. The network has a general topology and is capable of multicasting. The amounts of available bandwidth are diverse and differ among links. We assume that the network offers bandwidth reservation mechanisms such as ATM, DiffServ, IntServ or TTCP/ITM [4]. The CPU resources can be controlled and reserved with a real-time OS such as Real-Time Mach [8] or HiTactix [4]. We assume that heterogeneous end systems are involved in the video multicast, and that the available amounts of CPU resources also vary.

In our system, the server first notifies users about the video session, including the starting time, the subscription due time, and the contents of the service, either through broadcasting or by using a dedicated multicast address as in SDP [10]. Then the client for a user intending to join reserves the available CPU resources and informs the server of the amount successfully reserved. At the same time, the server examines its own resource availability and the network conditions (i.e., the available bandwidth of the links), through a bandwidth reservation protocol. Based on this information, the server then composes multicast groups by dividing the clients into clusters based of their available access link bandwidth and CPU resources. The server then constructs multicast trees. We explain the algorithms in detail in Section 4.

Next, the shared resources—i.e., the server CPU and the link bandwidth—are allocated to each session. Initially, identical amounts of shared resources are allocated to each multicast group sharing the same bottleneck link. In each group, resource allocation is performed so as to maximize the quality of the video transfer, given the relationship between resources (local

resource allocation). Then, the remaining resources are re-allocated to the cluster, which is expected to contribute to increasing the total utility (global resource allocation). This two-level resource allocation is formulated as a utility maximization problem in which the utility is represented as a relation between the benefit and the cost, i.e., the obtained video quality and the required amount of resources. We describe this procedure in Section 5.

By iteratively repeating these global and local utility maximizations, the server determines the resource allocation. Then, it reserves its own CPU resources, reserves bandwidth for each session by a method such as RSVP, and notifies the clients of the required amounts of CPU resources. Each client confirms it has reserved the required CPU resources, and real-time video multicasting begins.

4 Multicast Session Construction by Clustering

The heterogeneity in the available client CPU and network resources must be taken into account to achieve high-quality video multicasting efficiently. There are several approaches to tackling this heterogeneity, such as a simulcast in which the video server multicasts multiple video streams of the same content but with varying quality, a layered multicast [11], and active networks [12]. Our approach is similar to a simulcast in that the server generates independent video streams of varying quality. However, the number of multicast groups is reduced through a clustering technique, a video stream of appropriate quality is chosen for each cluster, and resources are adequately allocated among clusters and system entities. Our scheme is also applicable to a layered video stream through straightforward extension by regarding the selection of coding parameters as the selection of layers.

In our scheme, the clients are first grouped into clusters based on the available access link bandwidth and CPU resources. Each cluster corresponds to a multicast group carrying a video stream of a quality appropriate for the group's members. Although heterogeneity exists in the available bandwidth in the network for video transfer, in the clustering phase we only take into account the CPU resources and access link bandwidth. This is because the available bandwidth for the multicast session cannot be determined until the multicast trees are constructed. In the

following subsections, we consider issues related to the clustering.

4.1 Clustering of clients

There are several clustering algorithms for grouping samples by their similarities [13]. The k-mean algorithm is one widely known clustering algorithms. It generates k clusters based on Euclidean distance and is favored for its simplicity. The algorithm starts out by selecting the initial k seeds. The data are associated with the closest seed in terms of Euclidean distance and constitute k sets of data, i.e., clusters. Then, the k center points of clusters are calculated. Data association is performed again, and these two steps are repeated until no change occurs. Since the initial k points are chosen at random in the k-mean clustering algorithm, the speed of convergence and the feasibility of the obtained clusters vary from trial to trial [14]. Therefore, we have to repeatedly try clustering to obtain better results. The KA algorithm was thus proposed to avoid the instability of the k-mean clustering algorithm, and it have been verified that the KA algorithm obtains a unique initial state that leads to more centralized clusters [14, 15]. Kaufman and Rousseeuw's work [15] provides a detailed description of the KA algorithm.

To apply the KA and k-mean algorithms, we must first determine the number of clusters, k. We repeatedly try clustering over a range of values from $k = 1$—that is, all clients are accommodated in a single multicast group and provided with a single video stream—to a specified maximum number of clusters, say, K. In an extreme case, K is identical to the number of clients and causes a scalability problem. In an actual situation, however, K is limited to a realistic number since the system resources cannot accommodate too many simultaneous multicast sessions. We investigate the effect of K and the initial state setting in Section 6.

4.2 Mapping client resources

To apply a clustering algorithm to grouping heterogeneous clients on the basis of their resource availabilities, the amounts of available resources must be normalized to a range from 0 to 1. We take the access link bandwidth and client CPU resources into account in clustering the clients. Since the possible combinations of these two parameters, which are determined from a set of coding parameters such as the quantizer scale and the GoP structure, does not form a linear

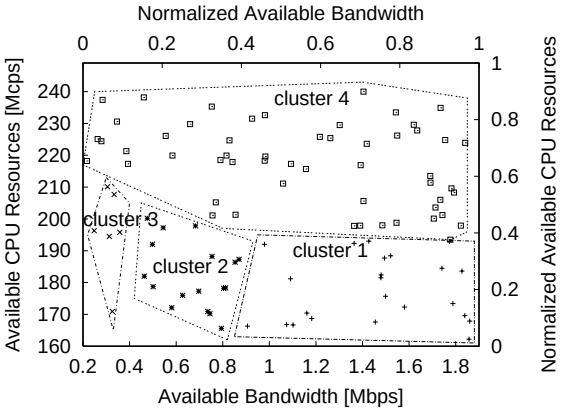

Figure 4: Linear normalization

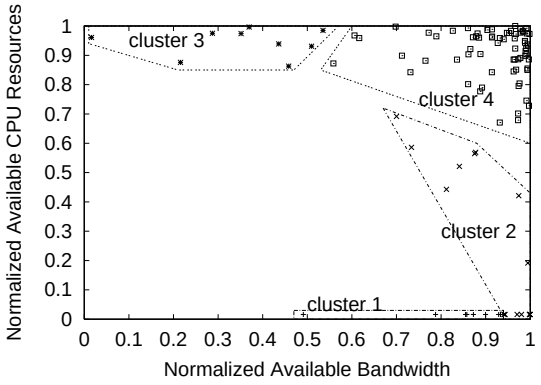

Figure 5: Non-linear normalization

function, we apply a non-linear normalization derived from the data in Figs. 1 to 3. An example of a comparison between the linear and non-linear normalization methods is shown in Figs. 4 and 5. Figure 4 depicts the distribution of the clients in accordance with their available resources and the resultant clusters with a linear normalization. With non-linear normalization, the resulting clusters change as shown in Fig. 5.

4.3 Multicast tree construction

Each cluster corresponds to a multicast session and receives a video stream sent through the multicast tree. For a high-quality, efficient video multicast, the multicast tree is constructed to contain less number of links that have larger available bandwidth. The Steiner Tree problem consists of searching the k-MST (Minimum Spanning Tree) spanning to specific nodes k, and this problem is NP-complete [16]. We define the cost of a link as the reciprocal of the available bandwidth and employ an LCM algorithm [17] for tree construction, giving an approximate solution to the Steiner Tree problem.

5 Integrated Resource Allocation Scheme to Maximize Users' Utility

To provide high-quality, real-time video presentation to users in resource reservation-based systems, it is necessary to efficiently allocate resources to the application for each entity in the whole system. We formulate the resource allocation as a maximization problem of "utility",

which is defined as the ratio of “benefit” to “cost”. The benefit forms a monotonically increasing function in the case of video quality. The cost reflects the load on the system and monotonically increases with respect to the amount of allocated resources. With these functions, we can expect effective video multicasting when high-quality video streams are provided without increasing the resource usage.

Utility maximization is performed with respect to the whole system and within each cluster. The maximization of the total utility considers the resource allocation among clusters. The idea behind this is illustrated in Figure 6. This figure shows a typical example of the relationship between benefit and cost taken from our preceding research work on MPEG-2 video streams [5]. Under such conditions, the total utility can be increased by taking only a portion of the shared resources allocated to rich cluster A, which is receiving a high benefit, and giving it to poor cluster B, which is receiving a low benefit because its allocated resources are insufficient. It is expected that this operation will hardly degrade the benefit of cluster A but greatly increase that of cluster B. As a result of this resource re-allocation, the total utility is expected to increase.

We also consider the interdependence among resources to maximize each cluster’s utility. For example, if plenty of bandwidth is allocated to one cluster, its members can receive video data coded at a low compression ratio and avoid complex, heavy decoding. However, the required bandwidth can be decreased if the end systems devote a large amount of CPU resources into coding and decoding tasks and accommodate a highly compressed video stream. A cluster’s utility can be maximized by combining Eqs. (1) through (3) and carefully choosing the

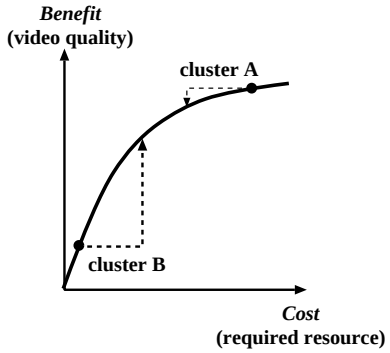

Figure 6: Relationship between benefit and cost

coding parameters. By considering those relationships, the appropriate resource utilization and a high-quality video stream can be achieved within the limitations of the available resources.

5.1 Definition of utility

The total utility U is given as the sum of the clusters' utilities U_i, i.e.,

$$U = \sum_{i}^{k} U_i \ . \tag{4}$$

The utility of cluster i is defined as a function of the benefit B_i with the allocated resources and the cost P_i paid for them:

$$U_i = B_i / P_i \ . \tag{5}$$

In this manner, a higher utility can be obtained by providing a higher quality of video to users while keeping the resource utilization lower.

The benefit of cluster i, B_i, is represented as a product of the video quality q_i, which we define as a reciprocal of the quantizer scale Q_i, and the number of clients m_i. Any other definition of q_i fits our scheme as long as it forms a monotonically increasing function with reapect to resources:

$$B_i = q_i \times m_i \ . \tag{6}$$

A cluster's cost consists of three different costs, based on the server CPU, the network bandwidth, and the client CPU resources. We define the cost function as follows:

$$P_i = \zeta\{P_i^W\}^2 + \eta\{P_i^S\}^2 + \theta\{P_i^C\}^2 \, , \tag{7}$$

where ζ, η and θ are positive constants that define the importance of each resource. Appropriate determination of these constants is beyond the scope of this paper and remains as a topic for future research work. In our experiments, these constants are all set to one. For example, however, we could make ζ two times larger than the others if bandwidth were the most expensive resource among them.

The bandwidth cost P_i^W is related to the bandwidth usage of the multicast tree i. Since the bandwidth cost increases as the number of links n_i in the multicast tree increases, the cost is

defined as follows:

$$P_i^W = n_i \times \frac{W_i}{W_i^{free}}, \tag{8}$$

where W_i is the bandwidth required for the video stream assigned to cluster i, and W_i^{free} indicates the available bandwidth of multicast tree i —that is, the available bandwidth of the tree's bottleneck link.

The server CPU cost P_i^S is defined as the ratio of the amount of resources required for encoding video data to the available amount of server CPU resources S_i^{free} allocated to the cluster i:

$$P_i^S = \frac{S_i}{S_i^{free}}. \tag{9}$$

The cluster CPU cost P_i^C is given as the average utilization of the client CPU resource:

$$P_i^C = \frac{1}{m_i} \sum_j \frac{C_i}{C_{ij}^{free}}, \tag{10}$$

where C_i and C_{ij}^{free} stand for the amount of CPU resources required for decoding the video data and the available amount of CPU resources for client j of cluster i, respectively.

5.2 Utility maximization

In this section, we formulate resource allocation as a maximization problem of the total utility as defined in Section 5.1. By solving this optimization problem, we can determine an efficient resource allocation for the whole system.

$$maximize\ U \tag{11}$$

$$Under\ the\ constraints$$

$$\forall l \quad \sum_i^k W_i \cdot Z(i,l) \ \leq \ L_l^{free} \tag{12}$$

$$\forall i \quad W_i \ \leq \ W_i^{free} \tag{13}$$

$$\forall i \quad S_i \ \leq \ S_i^{free} \tag{14}$$

$$\sum_i^k S_i^{free} \ \leq \ S^{free} \tag{15}$$

$$\forall i,j \quad C_i \ \leq \ C_{ij}^{free}, \tag{16}$$

Table 1: Candidate GoP structures

G_0	I
G_1	IP
G_2	IB
G_3	IPPPPP
G_4	IBPBPB
G_5	IBBPBB
G_6	IBBBBB

where k and L_l^{free} stand for the number of clusters and the available bandwidth of link l, respectively; And $Z(i,l)$ is one if a multicast tree contains link l, and zero otherwise.

We solve the problem by the following heuristic algorithm:

1. Allocate the server CPU resources equally to each cluster. Bandwidth has already been allocated to the clusters during the tree construction phase.
2. Determine the resource allocation that maximizes the utility of each cluster within the available resources by choosing appropriate set of coding parameters with Eqs. (1), (2) and (3).
3. Subtract the allocated resources from the system resources.
4. Re-allocate the remaining resources on the server CPU and the links to the cluster whose utility increases the most with the newly allocated resources.
5. Repeat steps 2 through 4 until no cluster can increase its utility or no resources remain.

By using this scheme, the resource allocation can be decided for each cluster and the quality of the video can also be determined from the relationship described in Section 2.

6 Evaluation

In this section, we evaluate the effectiveness and appropriateness of our proposed scheme by applying it to a multicast network with heterogenous clients.

6.1 Simulation model

In our simulation experiments, we use the general-topology network model shown in Fig. 7, which is taken from an MCI network and consists of 19 nodes and 32 internal links. A server

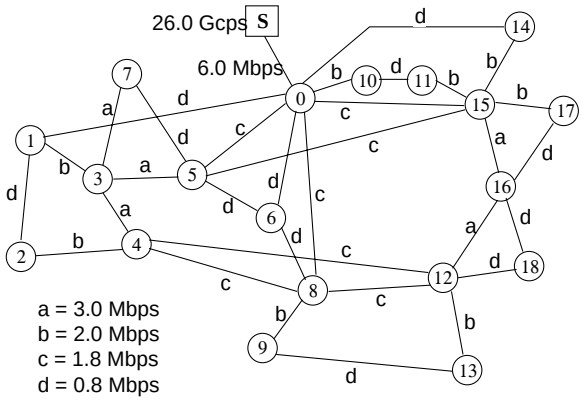

Figure 7: General-topology network model

is connected to node "0" via a 6.5-Mbps link. Clients are connected to randomly chosen nodes. The bandwidth available for a video session on each access link and the CPU resources available for decoding tasks are chosen at random as long as they can enable a video stream of the minimum quality. In our experiments, all clients are assumed to join the same video session for the sequence "Animation", whose spatial resolution is 160×120 pixels and temporal resolution is 30 fps. This means that only the SNR resolution Q and the GoP structure contribute to resource allocation. In the resource allocation phase, the server determines an appropriate set of coding parameters to maximize cluster utility. The SNR resolution (i.e., the quantizer scale Q_i) ranges from 4 (33.25 dB) to 40 (18.93 dB) at intervals of four. In this paper, based on our previous research work, only the quantizer scale determines the perceived video quality [2]. The GoP structure G_i is chosen from the list shown in Table 1.

6.2 General-topology network model case

The results of experiments for ten clients in the general network model are summarized in Table 2 and the transition of the total utility is shown in Fig. 8. The original k-mean clustering algorithm was used, and we show the results for all k from one to ten. Each member of a tuple U_i / m_i and (G_i, Q_i) corresponds to a cluster's utility, the number of clients accommodated in the cluster, and the set of coding parameters employed in the video data provided to the cluster. In the bottom row, the total utility U and the utilization of resources u^W, u^S, and u^C are shown. A total

Table 2: Results of integrated resource allocation

Cluster No.	Number of clusters									
	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$	$k=9$	$k=10$
1			0.06 / 1 (G_5,40)	1.02 / 2 (G_3,32)	0.49 / 1 (G_0,40)	0.70 / 2 (G_3,32)	0.99 / 2 (G_3,32)	0.99 / 2 (G_3,32)	0.42 / 1 (G_3,32)	
2			1.00 / 2 (G_0,40)	0.06 / 1 (G_5,40)	0.74 / 2 (G_3,32)	0.30 / 1 (G_0,40)	0.72 / 1 (G_5,32)	0.16 / 1 (G_5,40)	0.72 / 1 (G_5,32)	
3			0.20 / 7 (G_3,40)	0.99 / 2 (G_0,40)	0.06 / 1 (G_5,40)	0.20 / 2 (G_5,40)	0.30 / 1 (G_0,40)	0.88 / 1 (G_5,20)	0.36 / 1 (G_5,40)	
4				0.17 / 5 (G_5,40)	1.40 / 1 (G_0,24)	0.06 / 1 (G_5,40)	0.23 / 1 (G_5,40)	0.45 / 1 (G_0,40)	0.30 / 1 (G_0,40)	
5					0.17 / 5 (G_5,40)	1.30 / 1 (G_0,24)	0.05 / 2 (G_5,40)	0.54 / 2 (G_5,40)	0.38 / 1 (G_5,40)	
6						0.32 / 3 (G_5,40)	1.19 / 1 (G_0,24)	0.06 / 1 (G_5,40)	0.56 / 1 (G_3,28)	
7							0.76 / 2 (G_5,28)	1.09 / 1 (G_0,24)	0.05 / 2 (G_5,40)	
8								0.68 / 1 (G_5,40)	0.99 / 1 (G_0,24)	
9									0.40 / 1 (G_5,40)	
Total utility	0.00	0.00	1.26	2.25	2.86	2.87	4.24	4.86	4.19	0.00
u^W			98.6	98.6	98.6	98.6	98.6	98.6	98.6	
u^S			28.4	42.5	44.1	58.1	72.2	86.2	99.0	
u^C			99.5	99.5	99.5	99.5	99.5	99.5	99.5	

utility of zero implies that the resource allocation failed. The reason is that there is a trade-off between the network bandwidth and the end system resources. When a client with insufficient CPU resources and another client with a narrow access link are clustered into the same multicast group, no set of parameters can simultaneously satisfy both clients and the resource allocation fails.

The utilization of the network bandwidth is given as

$$u^W = \max_l \left(\frac{\sum_i^k W_i \cdot Z(i,l)}{L_l^{free}} \right), \tag{17}$$

where k and L_l^{free} stand for the number of clusters and the available bandwidth of link l, respectively; $Z(i,l)$ is one if a multicast tree contains link l, and zero otherwise; and u^W denotes the

utilization of the most congested link. For the server CPU resources, the utilization u^S is derived from the following equation:

$$u^S = \frac{\sum_i^k S_i}{S^{free}}. \tag{18}$$

Finally, the utilization of the client CPU resource is given as,

$$u^C = \max_{i,j} \left(\frac{C_i}{C_{ij}^{free}} \right). \tag{19}$$

Table 2 clearly shows the trade-off between the number of clusters and the total utility. In these experiments, clusters of one and two failed since some clients had insufficient resources and no set of coding parameters could satisfy all the clients in a cluster. As the number of clusters k increased, the clustering algorithm could group clients into multicast sessions according to their resource availability in more effective and appropriate ways. Then, the resource allocation algorithm successfully determined sets of coding parameters and amounts of resources to allocate to each multicast session. However, beyond $k = 8$ the total utility began to decrease. This is because a larger k leads to division of the shared resources into small pieces, and thus the quality of the video streams provided to the clusters is degraded. In addition, the resource utilization increases, and therefore the cost increases. As mentioned before, the quantizer scale alone determines the perceived video quality in our experiments, so we can conclude that constructing eight multicast groups is the best way to efficiently provide clients with video streams of the highest possible quality.

For comparison purposes, we also conducted resource allocation with only the first two steps of the heuristic algorithm That is, the server CPU resources were divided equally, the bandwidth was allocated in a max-min fair manner, and utility maximization was carried out within each cluster. Thus, global utility maximization by means of resource re-allocation was not performed. This strategy, called “equal resource allocation”, can be regarded as a rather conventional simulcast, except that the video quality is determined so as to maximize the cluster’s utility within the allocated resources.

In the case of equal resource allocation, whose results are shown in Fig. 8, all trials failed

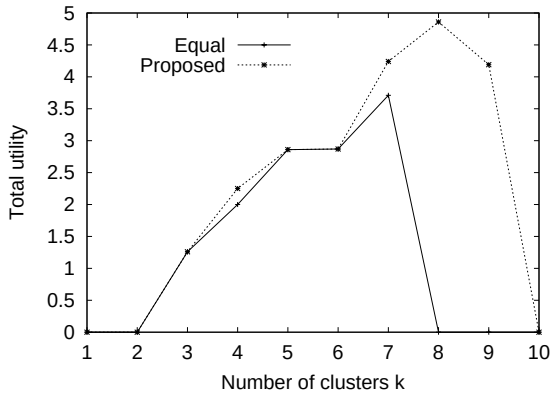

Figure 8: Transition of utility (10 clients)

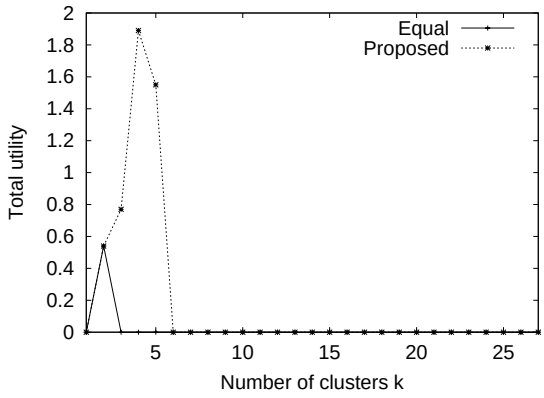

Figure 9: Transition of utility (100 clients)

except from $k = 3$ to 7. The maximum utility was obtained in the case of $k = 7$, but it was 3.71, far below that obtained with our scheme, and, of course, the quality of the video stream provided to the clients was low.

Figure 9 shows that our scheme can obtain a feasible resource allocation in a general network with 100 clients. Only four multicast groups were sufficient for 100 heterogeneous clients. In this case, we do not need to try utility maximization beyond $k = 27$ clusters even for more than 100 clients. This number is derived by dividing the server access link bandwidth by the bandwidth required to transfer a video stream of the minimum quality. In practice, for a 100-client session, trying utility maximization more than dozen of clusters does not lead to a better allocation.

6.3 Initial state setting of clustering

As described in Section 4.1, the k-mean algorithm has difficulty in determining the initial state. In this section, we present some results from our experiments on the KA algorithm in the integrated resource allocation scheme. We compared the KA algorithm to the original k-mean algorithm, referred to as RA, where the initial state is given at random. To find a better allocation or the best allocation we ran the RA algorithm 100 times per value of k.

Figures 10 and 11 show the transitions of the maximum utilities obtained by the KA and RA algorithms. From these series of results, we found that the RA better utilities than the KA algorithm in most cases. This is because the centralized clusters obtained by the KA algorithm

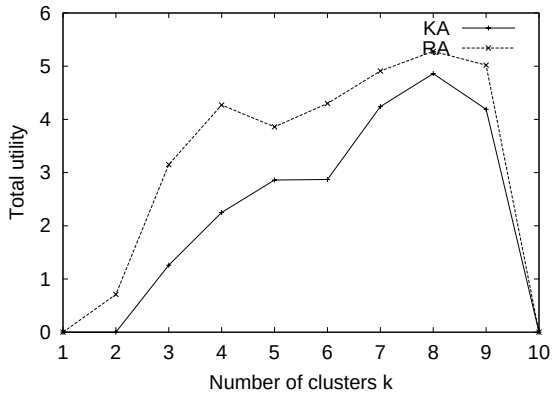

Figure 10: Transition of total utility by KA and RA (10 clients)

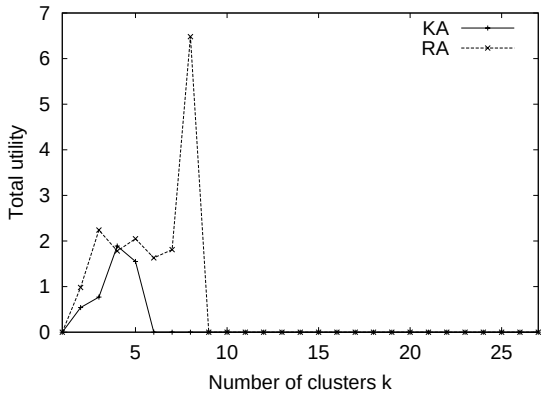

Figure 11: Transition of total utility by KA and RA (100 clients)

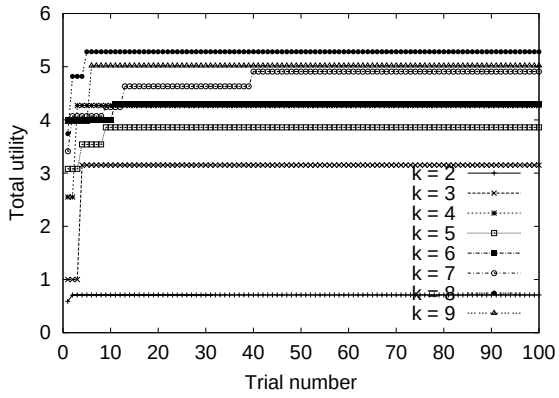

Figure 12: Transition of maximum uility by RA (10 clients)

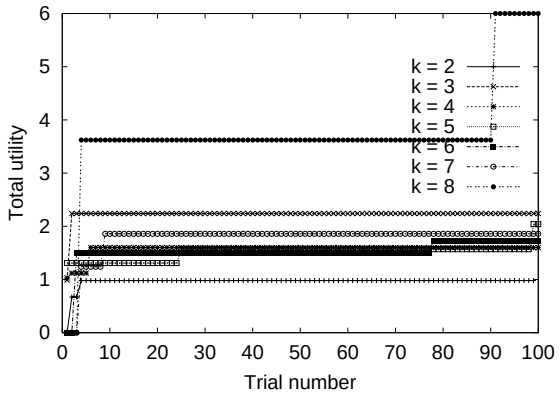

Figure 13: Transition of maximum utility by RA (100 clients)

not necessarily lead to or guarantee construction of multicast groups that are preferable for utility maximization.

We next show how fast the RA algorithm obtained the highest utility through 100 trials in Figs. 12 and 13. The x-axes corresponds to the trial number i and the y-axes show the highest utility obtained up to i trials. In most cases the RA algorithm found the best allocation within a few dozen trials. However, in the case of eight clusters in a general-topology network with 100 clients (Fig. 13), the RA algorithm obtained a feasible resource allocation at the 91st trial, which was one of eight successful allocations among 100 trials. Since the resultant clusters depend on the randomly chosen initial state, the RA algorithm can not guarantee an optimal allocation within a limited number of trials.

7 Conclusion

In this paper, we have described a resource allocation scheme to provide efficient, high-quality video multicast services for heterogeneous clients. The resource allocation is formulated as a utility maximization problem that takes into account the relationships among resources. Several issues still remain. One is that the clustering phase does not necessarily lead to a feasible result. We should also consider a new clustering algorithm tied up with our resource allocation scheme. Another problem is related to the practicality of our scheme: it may lack scalability due to the fact that the algorithms are designed for centralized control. We now plan to build a resource reservation-based video multicast system to tackle these problems.

Acknowledgements

This work was partly supported by Special Coordination Funds for promoting Science and Technology, and a Grant-in-Aid for Encouragement of Young Scientists (12750322) from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

[1] K. Lakshman and R. Yavatkar, "Integrated CPU and network-I/O QoS management in an endsystem," in *Proceedings of IFIP IWQoS '97*, pp. 167–178, May 1997.

[2] K. Fukuda, N. Wakamiya, M. Murata, and H. Miyahara, "QoS mapping between user's preference and bandwidth control for video transport," in *Proceedings of IFIP IWQoS '97*, pp. 291–302, May 1997.

[3] C. Lee, R. Rajkumar, and C. Mercer, "Experiences with processor reservation and dynamic QOS in Real-Time Mach," in *Proceedings of Multimedia Japan*, March 1996.

[4] M. Iwasaki, T. Takeuchi, M. Nakahara, and T. Nakano, "Isochronous scheduling and its application to traffic control," in *Proceedings of 19th IEEE Real-Time Systems Symposium '98*, pp. 14–25, December 1998.

[5] K. Fukuda, N. Wakamiya, M. Murata, and H. Miyahara, "QoS guarantees based on end-to-end resource reservation for real-time video communications," in *Proceedings of 16th*

International Teletraffic Congress, pp. 857–866, June 1999.

[6] K. Fukuda, *Integrated QoS Control Mechanisms for Real-Time Multimedia Systems in Reservation-Based Networks*. PhD thesis, Osaka University, January 2000.

[7] P. Newman, "Traffic management for ATM local area networks," *IEEE Communication Magazine*, vol. 32, pp. 44–51, 1994.

[8] H. Tokuda, T. Nakajima, and P. Rao, "Real-Time Mach: Towards a predictable Real-Time system," in *Proceedings of USENIX Mach Workshop*, pp. 73–82, October 1990.

[9] C. W. Mercer, S. Savage, and H. Tokuda, "Processor capacity reserves: Operating system support for multimedia applications," in *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, pp. 90–99, May 1994.

[10] M. Handley and V. Jacobson, "SDP: Session description protocol," *RFC2327*, April 1998.

[11] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," in *Proceedings of ACM SIGCOMM '96*, pp. 117–130, September 1996.

[12] J. Smith, K. Calvert, S. Murphy, H. Orman, and L. Peterson, "Activating networks: A progress report," *IEEE Computer*, vol. 32, pp. 32–41, April 1999.

[13] J. A. Hartigan, *Clustering algorithms*. John Wiley & Sons, 1975.

[14] J. M. Peña, J. A. Lozano, and P. Larrañaga, "An empirical comparison of four initialization methods for the k–means algorithm," *Pattern Recognition Letters 20*, pp. 1027–1040, July 1999.

[15] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data. An Introduction to Cluster Analysis*. John Wiley & Sons, Inc., 1990.

[16] B. Wang and J. C. Hou, "Multicast routing and its QoS extension: Problems, algorithms, and protocols," *IEEE Network*, January 2000.

[17] A. Shaikh, S. Lu, and K. Shin, "Localized multicast routing," in *Proceedings of IEEE GLOBECOM '95, (Singapore)*, pp. 1352–1356, November 1995.