Implementation and Evaluation of Integrated Resource Allocation Scheme for Real-Time Video Transfer to Maximize Users' Utility

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 – For providing distributed multimedia applications with end-to-end QoS (Quality of Service) guarantees, resource reservation based control mechanisms should be employed in both of networks and end systems. In this paper, we implement and evaluate a resource allocation scheme to allocate both of network and server CPU resources to clients in an integrated manner. For this purpose, we use HiTactix for real-time OS and TTCP/ITM for the resource reservation network. Through experimentations using our implemented video transfer system, we confirm that our proposed scheme can achieve an effective use of resources while providing high quality video transfer.

1 Introduction

With dramatic improvements in computing power, network bandwidth and video data compression techniques, distributed multimedia systems become widely used. To provide good presentation of multimedia on those systems, various QoS guarantees are required, such as the data transferring delay, the regularity of video encoding and decoding [1].

The resource reservation based systems are preferable to guarantee those QoS for the video applications. The network level QoS such as loss ratio and transfer delay can be guaranteed by the bandwidth reservation based networks [2]. The real-time OS which can reserve and schedule CPU resources provide high speed and high quality video coding on end systems [3, 4].

However, even if we can successfully construct a distributed multimedia system by combining them, the high quality and real-time video transfer cannot be achieved without appropriate prediction/reservation mechanisms for both network and end systems resources. On MPEG-2 video transfer, there exists a strong relationship between required network/end systems resources and video quality obtained through them [5]. Based on those relationships, a resource allocation scheme proposed in [6] achieves the high quality video transfer with limited resource while maximizing user's utility, which is represented as relation between benefit obtained through allocated resources and cost paid for them. The proposed scheme also considers how shared resources, such as the network bandwidth and the server CPU resource, should be allocated among clients with different processing capabilities.

In this paper, we implement and evaluate the proposed scheme in the actual resource-reservation based system. The system is constructed with TTCP/ITM [4] providing the bandwidth reservation mechanism on Ethernet and HiTactix [4] allocating the CPU resource to tasks in the order of milliseconds. Of course, there are some mismatches among the proposed scheme and the system environment and some features are not clearly defined in [6]. Therefore, we first verify that the relationships in [5] also hold in our system. Then, we propose the method on how end systems communicate with each other to obtain information necessary for the resource allocation. By using the implemented video system, we confirm that the proposed scheme can achieve an effective use of resources while providing high quality video distribution.

This paper is organized as follows. In Section 2, we briefly introduce relationships among the video quality and the required resources, then outline the resource allocation scheme we proposed in the previous work. In Section 3, we discuss implementation-related issues on our system. In Section 4, we show some experimental results and evaluate the proposed scheme. Finally, we summarize our paper in Section 5.

2 Integrated Resource Allocation Scheme for Maximizing Users' Utility

By combining a bandwidth reservation network and a real-time OS, required QoS can be guaranteed for video applications. However, to provide high quality and real-time video presentation to users, it is necessary to allocate resources efficiently on each entity in the whole system. If there are sufficient resources, every user can enjoy high quality video presentation. However, in an actual situation, users compete for the limited bandwidth of the network resources. The server CPU resource is also competed by the clients. At the client, CPU is shared by several tasks. In addition to the availability on the various resources, we should consider the interdependence of resources. By

allocating much bandwidth, for example, the client can receive the video data of low-compression ratio and become free from the complex and heavy decoding task. In this section, we briefly introduce the relationships among the video quality and required network/end systems resources [5], then outline the resource allocation scheme we have proposed [6].

2.1 Relationship among Video Quality and Required Resources

In the previous work [5], we derive the relationships between the video quality and required network/end systems resources for MPEG-2 video data.

First, required bandwidth BW [Mbps] for video transfer can be estimated by the coding parameters of MPEG-2, which directly affect the video quality. Those are, spatial resolution R [pixels], SNR (Signal to Noise Ratio) resolution Q, temporal resolution F [fps], and GoP structure G. It is represented as;

$$\begin{aligned} &BW(R,Q,F,G) \\ &\quad \cong (3.1)^{\log_4 \frac{R}{640\times480}}(\alpha+\frac{\beta}{Q}-\frac{\gamma}{Q^2})\frac{F}{30}BW_{base} \end{aligned} \tag{1}$$

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

Next, the required CPU resource to code video data at the server S [Mcycle/sec], can be estimated by constant value S_G dependent on the GoP structure G as:

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

Finally, the required CPU resource to decode the video data at the client C [Mcycle/sec] is proportional to the required bandwidth BW as follows:

$$\begin{aligned} C \cong BW\times40+(870+\frac{N_p}{N}\delta+\frac{N_b}{N}\varepsilon) \\ \times\frac{R}{640\times480}\times\frac{F}{30} \end{aligned} \tag{3}$$

where N is the number of frames, and N_p and N_b are numbers of P and B pictures in a GoP. δ and ε are increasing rates against the amount of the CPU resource required to decode P and B pictures.

Using Eqs. (1) through (3), we can estimate a required amount of resources at network/end system for real-time video transfer.

2.2 Maximizing Users' Utility

To provide high quality and real-time video transfer to clients, a resource allocation scheme is proposed in [6]. The scheme takes into account both network and end systems resources, achieves efficient resource allocation with limited resources and maximizes user's utility (the words "user" and "client" are used interchangeably in this paper).

The utility of user i is defined as a function of the user's *Benefit* obtained through allocated resources and the *Cost* paid for them as follows;

$$U_i = Benefit_i(BW_i,S_i,C_i) - Cost_i(BW_i,S_i,C_i) \tag{4}$$

where BW_i means the bandwidth allocated to user i. S_i and C_i denote the amount of allocated CPU resources at the server and the client, respectively.

When the spatial resolution R and the temporal resolution F are specificaly determined by the application, the required amount of server CPU resource can be determined by an only GoP structure (see Eq.(2)). On the other hand, that of clients depends on GoP structure and bandwidth (see Eq.(3)). Therefore, user's benefit is determined by the GoP structure G_i and the allocated bandwidth BW_i as $Benefit_{G_i}(BW_i)$.

The cost function introduced to prevent users from monopolizing resources is defined as follows;

$$\begin{aligned} &Cost_{G_i}(BW_i) \\ &\quad = \left\{\alpha\frac{BW_i}{BW^{free}}+\beta\frac{S_{G_i}}{S^{free}}+\gamma\frac{C_{G_i}(BW_i)}{C_i^{free}}\right\} \end{aligned} \tag{5}$$

where BW^{free}, S^{free} and C_i^{free} denote the available bandwidth and CPU resources at the server and the client i, respectively. α, β and γ are positive constants to represent an importance of the each resource.

From these equations and the relationships between video quality and resources described in Section 2.1, each user's utility $U_{G_i}=Benefit_{G_i}(BW_i)-Cost_{G_i}(BW_i)$ becomes a convex function of the bandwidth. Then, we can formulate the resource allocation to maximize total utility. By solving this optimization problem, we can determine efficient resource allocation for the whole system.

$$\begin{aligned} \text{maximize}\ &\textstyle\sum_i U_{G_i} = \sum_i\{Benefit_{G_i}(BW_i) \\ &\qquad -Cost_{G_i}(BW_i)\} \\ \text{subject to}\ &\textstyle\sum_i BW_i \le BW^{free}, \\ &\textstyle\sum_i S_{G_i} \le S^{free}, \\ &C_{G_i}(BW_i) \le C_i^{free}\ \forall i \end{aligned} \tag{6}$$

3 Implementation of Resource Allocation Scheme

By applying the resource allocation scheme in Section 2.2 to an actual video transfer system, we can provide the efficient and integrated resource allocation and high quality video data to clients. However, there are some mismatches among the proposed scheme and the system environment. In this section, we clarify those problems and propose solutions to them.

3.1 System Description

In Fig. 1, we illustrate our video transfer system. The server captures the original video with a camera. Captured pictures are coded by an MPEG-2 algorithm and

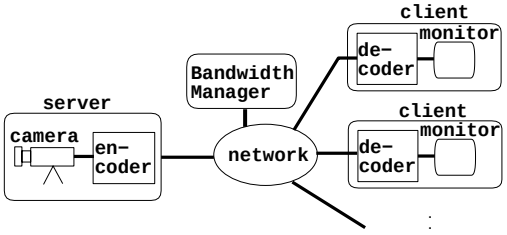

Figure 1: System Model

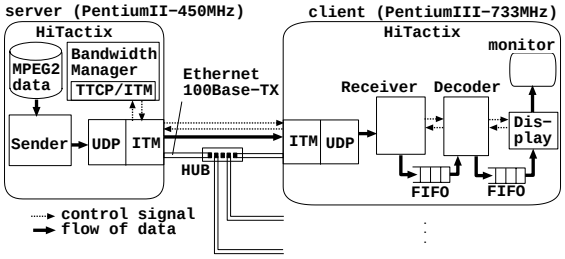

Figure 2: Implemented System

sent to clients via the bandwidth reservation based network. The server CPU resource is allocated to each of capturing, coding and sending tasks for each client according to our resource allocation scheme. The network bandwidth i.e, capacity of the server's output link is also shared among clients. Each client receives the video, decodes it, and displays it on the monitor. Each of receiving, decoding and displaying tasks is allocated a portion of the client CPU resource.

Our implemented system is depicted in Fig. 2. In this paper, we construct a real-time video transfer system with TTCP/ITM [4] providing the bandwidth reservation mechanism on Ethernet, and real-time OS HiTactix [4] allocating CPU resource to tasks in the order of milliseconds. Unfortunately, however, the insufficient capability of our server's CPU power inhibited real-time encoding. We therefore used the video data that had already been coded and stored in the disk storage. Those are read out by the sending module "Sender" and sent to clients using UDP/ITM protocol stack. The bandwidth manager exists in the sever and manages the 100Base-TX network. Each client has the receiving, decoding and displaying modules, which are "Receiver", "Decoder" and "Display", respectively. The data are passed among modules via FIFO queues. In the following sections, we give some detailed discussions on how QoS is guaranteed in our implemented system.

3.2 QoS Guarantee on Networks

To achieve real-time video data transfer, we guarantee the network level QoS by using the resource reservation network protocol, TTCP/ITM. The TTCP/ITM can provide real-time data transfer by determining amount of sending data in certain interval. The TTCP (Total Traffic Control Protocol) is a signaling protocol which manages the bandwidth allocation to suppress the excess increases in total traffic flows for both real-time and non-real-time data transmissions in Ethernet. The ITM (Isochronous Transmit Mode) module is a traffic shaper which communicates with the bandwidth manager and regulates the amount of outgoing traffic.

3.3 Enabling Real-Time Encoding

To provide high quality video presentation to users, it is necessary to reserve the CPU resource for coding tasks at end systems. The HiTactix can allocate the CPU resource to each thread at the order of millisecond. In the video transfer system of Fig.1, the server captures the original video data, encodes it in a real-time fashion and sends them to the clients. To assign those tasks into a single thread leads to the waste of the allocated CPU resource because the processing delay introduced to each task by, for example, I/O interruption stops the entire thread operation. Therefore, we should prepare three independent threads, each of which is responsible for capturing, coding and sending task, respectively. However, in this paper, the server only reads out the MPEG-2 data at the regular interval of 1/F seconds from a disk as shown in Fig.2, and there is only one thread for reading and sending tasks.

3.4 Enabling Real-Time Decoding

Each client receives the video from the server, decodes it in a real-time fashion and displays it on the monitor (Fig. 1). We implemented three threads, each of which is responsible for receiving, decoding and displaying task (see Fig.2). Further, the video data are passed among threads via FIFO queues of 30 Mbytes capacity. The video can be played smoothly by parallel execution of synchronized threads. There are some mismatches and unmentioned factors about the client in [5]. Eq.(3), which is used to estimate the required amount of client CPU resource, only considers the decoding task. Thus, we should first check whether Eq.(3) is applicable to our system. Then, we investigate how much CPU resources are required for the "Receiver" and "Display" through experiments.

The required amount of CPU resource with HiTactix in decoding "Animation" video sequences of 160×120 pixels and 30 fps are summarized in Fig. 3. Each point on lines corresponds to the SNR resolution, that is, the quantizer scale (4,8,12,16,20,24,28,32,36,40). Figure 3 shows that the amount of CPU resource needed for the Decoder thread is in proportion to the bandwidth for given GoP structure as shown in Eq.(3). Therefore, we can build the following relation from Eq.(3).

$$C_{decode} \cong BW \times 40.71 + (89.57 + \frac{N_p}{N}18.13 + \frac{N_b}{N}37.18) \times \frac{R}{160\times120} \times \frac{F}{30} \quad (7)$$

Although not shown in figures, the CPU resource $C_{display}$ required for the Display thread is nearly con-

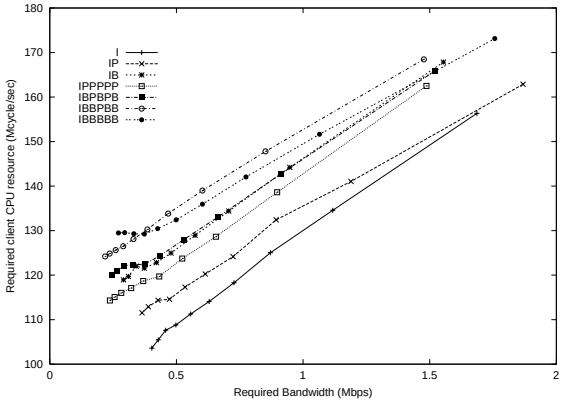

Figure 3: Required amount of CPU resources and bandwidth

stant and is predictable with the following equation.

$$C_{display} \cong 55.90 \times \frac{R}{160 \times 120} \times \frac{F}{30} \qquad (8)$$

The required amount for the receiving task was considerably small in our system. Further, HiTactix allocates the sufficient CPU resource to socket operations. We therefore did not need explicitly specify and allocate the CPU resource to the Receiver thread.

The required CPU resource at the receiver to provide the QoS guarantee for real-time video playout is then derived as;

$$C_{total} \cong BW \times 40.71 + (145.47 + \frac{N_p}{N} 18.13 + \frac{N_b}{N} 37.18) \times \frac{R}{160 \times 120} \times \frac{F}{30} \qquad (9)$$

3.5 Information Exchanges for Resource Allocation

To perform the resource allocation, the available amount of system resources should be known a priori. To apply the scheme to an actual system, there must be some appropriate signalling protocols for the server to obtain those information. In the application in which the start time is known in advance such as VoD (Video on Demand) or a live telecasting, the possible procedure is as follows;

1. The server broadcasts informations on the video distribution, including the start time, the subscription due time and the service contents.
2. The client intending to receive the service reserves the available CPU resource and informs the server of its amount.
3. The server investigates its own resource availability and inquires about the available bandwidth to the network bandwidth manager.
4. The server determines the resource allocation considering the available resources by using the resource allocation algorithm.
5. The server reserves the bandwidth and notifies the clients with the amount of client CPU resource.
6. Each client confirms the reserved CPU resource for the video application.
7. The server sends the video data to the clients.

4 Experimentation and Evaluation

In this section, we evaluate the effectiveness and appropriateness of our implemented system by observing the resource utilization and the video quality. For comparison purpose, we also implemented the "fair resource allocation" scheme where all clients are allocated the same share of the server CPU and the network bandwidth independent on the client CPU resource availability.

In experiments, we assume that the amount of server CPU is 15.0 Gcycle/sec and 4 clients are connected to the server. The available network bandwidth is assumed to be 4.0 Mbps and the amount of available client CPU resource is randomly chosen from 160 to 300 Mcycle/sec. We use the "Animation" video sequence of 1500 pictures with 160×120 pixels large. Namely, the entire video can be played in 50 seconds by 30 fps playout. Although GoP structure and all three resolutions, i.e. spatial, temporal and SNR, can be appropriately determined in our scheme (see Section 2.2), only the GoP structure and the SNR scalability are considered in the experiments. The weights α, β and γ in Eq.(5) are set to be 0.5, 0.1 and 0.1 respectively.

Results are summarized in Tables 1 and 2. In Table 1, we show the available CPU resource of each client C_i^{free}, and the selected GoP structure and quantizer scale for the client by our scheme. Table 2 summarizes the amount of resources allocated to the video application on server, client and network and their utilization ratio given as the ratio of the allocated amount to the available. The number of the deadline misses is also shown. In our experiments, deadline misses only occur at the Display thread.

In the case of the fair resource allocation where all clients are provided with the identical video data, the clients' CPU are not effectively utilized at client 2 and 3 whose available resources are sufficient while the client 1 and 4 with insufficient CPU is overloaded and deadline misses occur. On the contrary, with our scheme, client 1 and 4 are assigned the video sequence of lower compression ratio than the others by considering their insufficient CPU resources as shown in Table 1. As a result, no deadline miss occurs and the video is played out at 30 fps on client 1 and 4.

From results above, we conclude that we can achieve the video transfer which accomplishes efficient use of resources, less deadline miss and high quality video distribution with the proposed scheme.

5 Conclusion

In this paper, we implemented the resource allocation scheme which maximizes users' utility consid-

Table 1: Available amount of client CPU resource and selected video data type

	client No.	C_i^{free} [Mcycle/sec]	selected GoP	Q
proposed scheme	1	183	I	12
	2	265	IBBPBB	8
	3	215	IBBPBB	8
	4	197	I	8
fair resource allocation	1	183	IB	8
	2	265	IB	8
	3	215	IB	8
	4	197	IB	8

Table 2: Allocated server, client and network resources and number of deadline misses

	client No.	allocated server CPU [Gcycle/sec] (utilization ratio [%])	allocated client CPU [Mcycle/sec] (utilization ratio [%])	allocated BW [Mbps] (utilization ratio [%])	deadline miss
proposed scheme	1	0.42 (2.8)	182 (99.4)	0.865 (21.6)	0
	2	3.65 (24.3)	209 (79.0)	0.851 (21.3)	0
	3	3.65 (24.8)	209 (97.3)	0.851 (21.3)	0
	4	0.42 (2.8)	193 (98.0)	1.136 (28.4)	0
fair resource allocation	1	3.23 (21.5)	207 (113.1)	0.958 (24.0)	139
	2	3.23 (21.5)	207 (78.1)	0.958 (24.0)	0
	3	3.23 (21.5)	207 (96.3)	0.958 (24.0)	0
	4	3.23 (21.5)	207 (105.1)	0.958 (24.0)	41

ering available resources. We clarified some implementation issues and proposed the solutions for them. Through experimentations on video transfer, we have shown that we could achieve efficient resource allocation in compliance with the system environment and provide high quality video to users.

However, there still remain several research works. For example, we do not consider the dynamic resource allocation against client's join and/or leave behaviors. When we allocate the network bandwidth and the server CPU resource to a newly joining client, we should take into account the degree of quality degradation of the video data provided to existing clients.

Acknowledgement

This work was partly supported by Special Coordination Funds for promoting Science and Technology of the Science and Technology Agency of the Japanese Government, Research for the Future Program of Japan Society for the Promotion of Science under the Project "Integrated Network Architecture for Advanced Multimedia Application Systems," Telecommunication Advancement Organization of Japan under the Project "Global Experimental Networks for Information Society Project," a Grant-in-Aid for Scientific Research (A) 11305030 and a Grant-in-Aid for Encouragement of Young Scientists 12750322 from The Ministry of Education, Science, Sports and Culture of Japan.

References

[1] K. Lakshman and R. Yavatkar, "Integrated CPU and network-I/O QoS management in an endsystem," *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," *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," *Proceedings of Multimedia Japan*, March 1996.

[4] M. Iwasaki, T. Takeuchi, M. Nakahara, and T. Nakano, "Isochronous scheduling and its application to traffic control," *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," *Proceedings of 16th International Teletraffic Congress*, pp. 857–866, June 1999.

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