Application–Level QoS of Web Access and Streaming Services with AF Service on DiffServ

Jun Takeo and Shuji Tasaka
Department of Electrical and Computer Engineering,
Nagoya Institute of Technology Nagoya 466–8555, Japan
Email: {takeo, tasaka}@inl.elcom.nitech.ac.jp

Abstract— **This paper assesses application–level QoS of both web access and voice–video streaming services on the Internet. We investigate how AFCP values in the two services affects their application–level QoS for various network scales, which are represented by the end–to–end delay between the web server and the client. As a result of the experiment, we see that we can keep higher application–level QoS of the two services if the web access service is offered as a different class from that of the streaming service or in the same class as that of the video flows with the video–flows' higher drop precedence. Furthermore, as the end–to–end delay increases, the effect of the choice of the AFCP values decreases. This implies that in such a network environment we do not need to give much consideration to AFCP marking.**

I. Introduction

Recently, a great number of services are provided on the Internet to satisfy the users' need. Among them, web access and voice–video streaming services are very popular with the Internet users.

The current Internet offers the best–effort service; the QoS (Quality of Service) deteriorates when the network is congested. Thus, some control methods are needed to improve the QoS. We can specify QoS control methods at each level of the protocol stack, e.g., at the application layer and at the network one [1]. For each application, we must define appropriate QoS parameters at each level and measure them to assess its QoS.

The users' perceptual quality can be expressed by MOS (Mean Opinion Score) for instance; this is one of the user–level QoS parameters. However, it is greatly labor–intensive to measure the user–level QoS parameters accurately. Thus, in this paper, we resort to the assessment of the application–level QoS, which closely correlates with the perceptual quality, instead of the user–level QoS.

We need to consider the application–level QoS taking account of the characteristics of target applications, since desirable QoS control for each application depends on its characteristics. Regarding the web access service, network congestion makes the time for retrieving files long; hence the application–level QoS becomes degraded, and congestion control is required in this application. In streaming services (e.g., video conference), on the other hand, packet loss and delay jitter disturb the temporal structure of voice and video streams; this means degradation of the application–level QoS. In this case, we apply *media synchronization control* [2] as application–level QoS control.

Network–level QoS control is one of the most challenging research subjects because of the recent progress of network equipment. DiffServ (Differentiated Services) [3] is this type of control, which IETF (Internet Engineering Task Force) has proposed as well as IntServ (Integrated Services) [4]. Since DiffServ has an advantage of the scalability over IntServ, it is considered a next generation architecture of the Internet. Therefore, this paper focuses on the assessment of the application–level QoS when DiffServ is applied.

DiffServ provides relative difference in transfer quality between aggregated flows, each of which is referred to as BA (Behavior Aggregate) [3]. Each BA obeys the corresponding rules described in PHB (Per–Hop Behavior) [3]. The set of rules is identified by the DSCP (DiffServ CodePoint [5]) value in the IP packet header. Typical PHBs in DiffServ include *EF (Expedited Forwarding)* [6], *AF (Assured Forwarding)* [7] and *Default* [5] ones. Each PHB provides its own QoS which is different from the others.

In this paper, we focus on the service provided by the AF PHB, which is referred to as the AF service, and the transfer behavior in the AF PHB is identified by the AFCP (AF CodePoint) value. The DSCP for AF PHB is translated into the AFCP. This service can provide better transfer quality than the best–effort service. We may consider that the EF service is suitable for transmission of voice and video, since it can provide low latency and low jitter. However, the EF service transfers flows with specified bit rates strictly. If the bit rates are high, the transmission of the other services may be impeded. Compared with the EF service, the AF service ensures only the minimum amount of resources (e.g., bandwidth or buffer) to each class and shares the remains on the basis of the weight of the priority in general. This implies that the AF service may behave itself better than the EF service. Thus, we employ the AF service in this study.

Many of previous studies on DiffServ do not assess the application–level QoS but the network–level QoS, whose parameters include the packet loss rate and the throughput. For example, references [8]–[10] discuss the difference in QoS between transport layer protocols in the AF service. In [8], Seddigh *et al.* investigate a fairness issue in different drop precedence assignments when UDP and TCP traffic share the same AF PHB class; they conclude that the fairness between TCP and UDP in an under–provisioned network cannot be completely achieved by using separate drop precedence with the recommended four Class and three Drop Precedence. In [10], Karam and Tobagi examine how to allocate audio, video and computer data flows to the classes; they notice that the flows are transferred efficiently when each application has their own class even on a low speed line such as T1.

We can find many studies on the web access service [11]–[14]. In [11], for example, Bhatti *et al.* investigate the relationship between response time and the number of pages users read in the e–commerce. In [12], Chandra *et al.* clarify how transcoding of JPEG pictures as web contents affects the response time and the consumed bandwidth.

It should be noted that most of the previous studies suppose only one service. Even if they target a couple of services, they assess the services separately. In the actual network, however, a number of flows generated by several types of applications are transferred on the same link. In this case, we should assess their QoS simultaneously by defining their QoS parameters and measuring them.

In this paper, we focus on the application–level QoS of both web access and voice–video streaming services, and we measure their QoS parameters by experiment. The two services are typical in the current Internet. The former usually deals with discrete media, which has no temporal structure among their packets. The latter deals with continuous media, which has the temporal structure. In our experiment, we transmit HTML flow as the web access service, two pairs of voice–video as the streaming service and a UDP data flow as the network load; then, we examine the relationship between the application–level QoS and their AFCPs for various delays between the web server and the client to simulate the network scale.

The rest of this paper is organized as follows. Section II describes the functions of the AF service. Section III shows the experimental system configurations and the method of our experiment. In Section IV, we present measurement results and discuss them. Finally, conclusions of this paper are given in Section V.

II. Functions of AF Service

The AF service provides better transfer quality than the best–effort service. It is realized by offering relative differences of the quality among BAs assigned to AF PHB. RFC 2597 [7] recommends implementing four classes with three levels of drop precedence. An IP packet that belongs to AF class i $(1 \leq i \leq 4)$ and has drop precedence j $(1 \leq j \leq 3)$ is marked with the AFCP (AF CodePoint) as AFij.

RFC 2597 also says that we can choose any transfer function to provide the service, e.g., a dropper and a scheduler. In this paper, we employ the packet dropper and the packet scheduler. These are implemented on Cisco's routers which we use in our experiment. The former on the Cisco's routers is WRED (Weighted Random Early Detection) [15]; the latter on them is CBWFQ (Class–Based Weighted Fair Queueing) [15].

In this environment, packets assigned to AF PHB may be dropped by WRED. Next, the packets are scheduled on the basis of the weight of each class by CBWFQ. Then, they are transferred to the next node.

A. WRED

WRED is a kind of RED [16]. WRED introduces grades of service among BAs based on packet drop probability and has selective RED parameters based on AFCP. The parameters are a pair of thresholds $(min_{\mathrm{th}}, max_{\mathrm{th}})$ and two constants (P_{d}, α).

When a packet arrives, the average queue length avr is renewed with the current queue length q and the weight factor α as follows:

$$avr \leftarrow (1-\alpha)avr + \alpha q.$$

Then, the following equation determines the packet dropping probability $P(avr)$ by comparing the calculated avr with the thresholds:

$$P(avr) = \begin{cases} 0, & \text{if} \quad avr < min_{\mathrm{th}} \\ 1, & \text{if} \quad max_{\mathrm{th}} \leq avr \\ \dfrac{avr - min_{\mathrm{th}}}{P_{\mathrm{d}}(max_{\mathrm{th}} - min_{\mathrm{th}})}, & \text{otherwise.} \end{cases}$$

B. CBWFQ

CBWFQ is a type of WFQ [17], [18]. CBWFQ forms an individual queue for each class, while WFQ forms an individual queue for each flow[1].

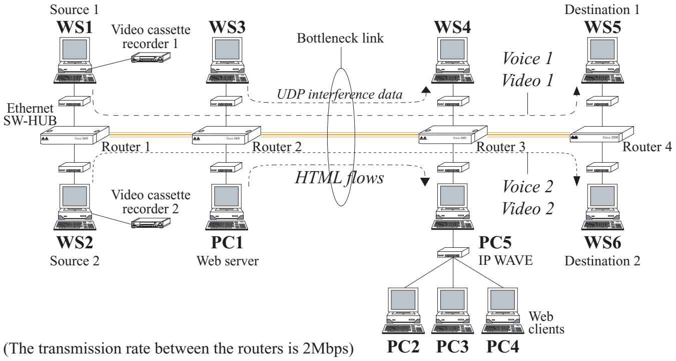

Fig. 1. Configuration of experimental system.

CBWFQ schedules packets on the basis of the required minimum bandwidth per each class. If the sum of all the request rates is smaller than the output link capacity, each request rate is assured; otherwise, the allocated rate becomes smaller than the request one. In this case, the fraction of the output link capacity available to a class is the ratio of the required minimum bandwidth of the class to the sum of the required minimum bandwidth of the backlogged classes at the instant.

III. Experimental System

In this paper, we perform an experiment to assess application–level QoS of both web access and streaming services. We transfer HTML flow, two pairs of voice–video streams and an interference data flow in the experiment. We employ DiffServ AF service for network–level QoS control and the enhanced *VTR (Virtual–Time Rendering) algorithm* [19], which is a media synchronization algorithm, for application–level QoS control.

A. Configuration of the Experimental System

Figure 1 illustrates the configuration of the experimental system; six workstations (WS1 through WS6) and five personal computers (PC1 through PC5) are connected to the network which comprises four routers (Routers 1 through 4).

Routers 1 through 3 are Cisco System's 2611, and Router 4 is 2514. Routers 1, 3 and 4 each has a memory of 2 Mbytes, and Router 2 has that of 32 Mbytes, respectively. Each of Routers 1 through 3 runs Cisco IOS (Internetworking Operating System) 12.0(7)T, and Router 4 works with Cisco IOS 11.2. The link connecting the adjacent routers is a V. 35 serial line, whose transmission rate is set to 2 Mbps. The link between the router and the computer is an Ethernet (10BASE–T, half–duplex).

In this experiment, the link between Router 2 and Router 3 becomes a bottleneck; we attach the DiffServ function to Router 2, while the other routers use only FIFO. It decodes the DSCP filed of each incoming packet; then the packet is aggregated into the corresponding BA on the basis of the DSCP value. The packet is treated with the dropper and the scheduler; then, it is transferred to Router 3. The AF PHB on Router 2 is shown in Table I. We set P_{d} and α to 3 and 2^{-9}, respectively. The value of P_{d} is determined on the basis of a preliminary experiment, while we use the default value of the routers for α.

Both of WS1 and WS2 are Sun Ultra 2; WS3 through WS6 are Sun Ultra 1. WS1, WS2, WS5 and WS6 each has Parallax Graphics's Power Video, which is an add–on board to encode/decode JPEG video by hardware in real time.

[1] *class* means a queue identified by i of AFij, while *flow* implies a stream of packets (e.g., an audio stream) in this paper.

TABLE I

AF PHB ON ROUTER 2.

			Class 1	Class 2	Class 3	Class 4
bandwidth [kbps]			200	400	600	800
Drop 1	$min_{\rm th}$	24	AF11	AF21	AF31	AF41
	$max_{\rm th}$	48	`001010`	`010010`	`011010`	`100010`
Drop 2	$min_{\rm th}$	16	AF12	AF22	AF32	AF42
	$max_{\rm th}$	40	`001100`	`010100`	`011100`	`100100`
Drop 3	$min_{\rm th}$	8	AF13	AF23	AF33	AF43
	$max_{\rm th}$	32	`001110`	`010110`	`011110`	`100110`

Note : The upper row in each entry indicates AFCP,
and the lower one shows DSCP.

TABLE II

SPECIFICATION OF COMPUTERS.

name	CPU	clock [MHz]	memory [Mbytes]
WS1 and WS2	Ultra SPARC	200	128
WS3 – WS6	Ultra SPARC	143	64
PC1	Celeron	433	128
PC2	Pentium II	400	384
PC3 and PC4	Pentium II	300	128
PC5	Pentium III	1200	256

PC1 through PC5 are PC/AT compatible computers. PC1 is a web server (Apache 1.3.19), and PC2 through PC4 are web clients (WebStone 2.5 [20]). The TCP window size of each PC is 16 kbytes. PC5 is equipped with a network impairment emulator (IP WAVE), which is used for providing propagation delay to packets that flow between each pair of web server–client. The details of WSs and PCs are shown in Table II.

WS3 transmits a UDP flow toward WS4. It becomes an interference flow for both services. The flow consists of 1472 byte–UDP datagrams sent at exponentially distributed intervals. The DSCP of each packet is marked in WS3 when the packet is transmitted.

B. Traffic of the Web Access Service

In our experiment, HTML flow is transferred from PC1 to PC2, PC3 and PC4 according to the configuration of WebStone.

WebStone is a web server evaluation tool which retrieves files from target web servers continuously. At first, WebStone generates web client processes on specified hosts. Next, those client processes retrieve specified files from target web servers independently. They use HTTP 1.0 GET request [21], so that the TCP connection is released for each transmission.

Table III shows the set of files to be retrieved in our experiment. This set contains the same items in `filelist.standard`, which is distributed with WebStone. In this table, `file5k.html`, for example, means a file of 5 kbytes and is retrieved with probability 0.5. In this experiment, WebStone generates four web clients: one client on each of PC2 and PC4, and two clients on PC3. Namely, the maximum number of TCP connections to be established at a time is four. The DSCP fields of packets generated by PC1 are modified by Router 2.

C. Traffic of the Streaming Service

As the streaming service, we transfer two pairs of voice–video streams. Table IV shows the specifications of those streams. WS1 and WS2 input voice and video from each video cassette recorder to encode them in real time. The voice and video captured by WS1 are referred to as voice 1 and video 1, respectively; voice 2 and video 2 are captured by WS2. A voice MU (Media Unit: the unit of media synchronization control) is constructed with 400 samples of encoded voice, and a video one corresponds to a video frame. WS1 and WS2 mark DSCP in the packets of those MUs and transfer the packets to WS5 and WS6, respectively. The voice and video are transferred as separate streams by RTP [22]/UDP.

TABLE III

THE SET OF FILES TO BE RETRIEVED.

file name	size [bytes]	probability
`file500.html`	500	0.35
`file5k.html`	5,125	0.5
`file50k.html`	51,250	0.14
`file500k.html`	512,500	0.009
`file5m.html`	5,248,000	0.001

TABLE IV

SPECIFICATIONS OF THE VOICE AND VIDEO STREAMS.

	Voice 1 Voice 2	Video 1	Video 2
coding scheme	ITU–T G.711 μ–law	JPEG	
image size [pixels]	——	320 × 240	
average MU size [bytes]	400 (constant)	3,253 (average)	3,247 (average)
original MU rate [MU/s]	20.0		
original inter–MU time [ms]	50.0		
original bit rate [kbps]	64.0 (constant)	520.7 (average)	519.6 (average)
measurement time [s]	120		

WS5 and WS6 receive the MUs and output them according to the enhanced VTR algorithm [19]. Regarding the parameters in the algorithm, we set the initial buffering time $J_{\max}$ [19] to 100 ms and the maximum allowable delay $\Delta_{\rm al}$ [19] to 400 ms. The choice of 400 ms for $\Delta_{\rm al}$ is made on the basis of ITU–T Recommendation G. 114 [23], which regards delays of 150 to 400 ms as acceptable provided that Administrations are aware of the transmission quality of user applications. The values of the other parameters are the same as those in [24].

D. Method of the Measurement

In our experiment, we assess the application–level QoS of the two types of services, which deal with discrete and continuous media. We examine them with six combinations of AFCP for various values of the end–to–end delay between the web server and the client, which reflects the network scale.

IP WAVE on PC5 produces an additional delay between the web server and a web client. Let d_i $(i = 2, 3, 4)$ denote the amount of the delay for PCi, which is given by the following equation:

$$
\begin{aligned}
&\left(\begin{array}{ccc} d_2 & d_3 & d_4 \end{array}\right) \\
&= \begin{cases} \left(\begin{array}{ccc} 0 & 0 & 0 \end{array}\right), & m = 0 \\ \left(\begin{array}{ccc} 50(m-1) & 50m & 50(m+1) \end{array}\right), & \\ & 1 \le m \le 4. \end{cases}
\end{aligned}
$$

We refer to the average of the delay as the *average additional delay*. Our experiment was conducted for various values of the average additional delay by setting m to 0, 1, 2, 3 or 4.

In addition, the average bit rate of the UDP flow is set to 0.6 Mbps.

TABLE V

A COMPARATIVE TABLE OF BAs.

case 1

	Class 1		Class 2		Class 3		Class 4	
Drop 1	HTML	200	Voice 1 [64]	400	Video 1 [520]	600	Load	800
Drop 2			Voice 2 [64]		Video 2 [519]			
Drop 3								

case 2

	Class 1		Class 2		Class 3		Class 4	
Drop 1		0	Voice 1 [64]	444	Video 1 [520]	666	Load	889
Drop 2			Voice 2 [64]		Video 2 [519]			
Drop 3			HTML					

case 3

	Class 1		Class 2		Class 3		Class 4	
Drop 1		0	Voice 1 [64]	444	Video 1 [520]	666	Load	889
Drop 2			Voice 2 [64]		Video 2 [519]			
Drop 3					HTML			

case 4

	Class 1		Class 2		Class 3		Class 4	
Drop 1		0	HTML	444		666	Load	889
Drop 2			Voice 1 [64]		Video 1 [520]			
Drop 3			Voice 2 [64]		Video 2 [519]			

case 5

	Class 1		Class 2		Class 3		Class 4	
Drop 1		0		444	HTML	666	Load	889
Drop 2			Voice 1 [64]		Video 1 [520]			
Drop 3			Voice 2 [64]		Video 2 [519]			

case 6

	Class 1		Class 2		Class 3		Class 4	
Drop 1		0	Voice 1 [64]	444	Video 1 [520]	666	Load, HTML	889
Drop 2			Voice 2 [64]		Video 2 [519]			
Drop 3								

Note : The value in each bracket means the bit rate of each stream in kbps, while an approximate allocated bit rate is shown in the right column of each Class. The shaded cells indicate the web access flows.

The combinations of AFCP values we have adopted in this paper are shown in Table V. This configuration is based on the result of our previous experiment in [25], which aims to assess application–level QoS of a voice–video streaming application in a middle–scale network. In the study, we conclude that higher application–level QoS is provided if voice and video are transferred as two separate classes. According to the result, we have allocated voice and video streams to Class 2 and Class 3, respectively. Then, we have assigned various AFCP values to the web access flow in order to observe the effect of their AFCP values. The value for the UDP load is assigned as a separate class from the voice–video streams.

As shown in Tables I and V, the approximate bit rate allocated to Class 2 is much higher than the sum of the bit rates of the two voice streams, and Class 3 approximately has a bit rate of almost 60 % of the total bit rate of the video streams. It is often the case that the bandwidth of video is hardly guaranteed compared with that of voice. As just described, partial guarantee of the required resources is a typical characteristic of the AF service.

In this paper, we aim to assess the QoS in a middle–scale network. This is because only the scheduler (e.g., WFQ) might be enough to control the QoS for small scale networks, while in the large scale networks we must reconsider the configuration of PHB to accommodate many applications' flows to 12 BAs, which are recommended by RFC 2597.

E. Application–Level QoS Parameters

It is necessary to define QoS parameters for each application to assess the application–level QoS quantitatively. This paper deals with the two applications: the web access and the streaming services. We define the QoS parameters for them below.

Web access service

- *Average retrieval time* : This is the average time from the moment each web client sends a request for establishment of a TCP connection to transfer an HTML file until the instant that the whole of the HTML file is transferred.
- *Average transfer time* : This value indicates the average time for just transferring an HTML file, namely, the average time from the moment a TCP connection has been established to transfer an HTML file until the instant the transmission of the HTML file is finished. Regarding this measure, the 95 % confidence interval will be shown for each measurement result.
- *HTML throughput* : This shows the throughput of the flow of the HTML files. This is calculated by dividing the total number of bits in the HTML files whose transmissions have been completed by the measurement time.
- *Completely transferred file number* : This is the number of HTML files transferred completely during an experiment.

Streaming service

- *Coefficient of variation of output interval* : This is a key measure for quantitative assessment of intra–stream synchronization quality. A smaller value means that the output of the stream is smoother. This value is the ratio of the standard deviation of MU output interval to the average output interval.
- *Average MU rate* : This value also expresses the smoothness of output of a stream. This is calculated by dividing the number of output MUs during the experiment by the experiment time.
- *Mean square error of inter–stream synchronization* : This represents the inter–stream synchronization quality, which is defined as the average square of the difference between the output time of each video MU and its derived output time. The derived output time of each video MU is defined as the output time of the corresponding voice MU plus the difference between the timestamps of the two MUs [2]. Referring to [26], we can consider this value less than 6,400 (=80^2) ms^2 to be high inter–stream synchronization quality, while more than 25,600 (=160^2) ms^2 to be out of synchronization.

IV. EXPERIMENTAL RESULTS

This section presents the experimental results and discusses how the combination of AFCP on each flow affects the application–level QoS of the two services for various additional delays. We show the values of the QoS parameters of the web access service measured on PC4 and those of the streaming service measured on WS5.

Figures 2 and 3 depict the average retrieval time and the average transfer time of 5125–byte HTML files, respectively, as a function of the average additional delay. The HTML throughput of all the HTML files is illustrated in Fig. 4. Regarding the streaming service, Figs. 5 and 6 show the coefficient of variation of output interval for voice 1 and that for video 1, respectively. Figure 7 displays the average MU rate for video 1. Figure 8 plots the mean square error of inter–stream synchronization between voice 1 and video 1.

First, we discuss the application–level QoS of the web access service. For all the additional delays, Case 3 takes larger values in Figs. 2 and 3, and it has larger increasing rates than the other cases in Fig. 2. Especially, from the 95 % confidence intervals in Fig. 3, we see that the variance of the values in Case 3 is the largest among all the cases. Therefore, we consider that the QoS of the web access service in Case 3 is degraded. This is because many packets of the HTML flow are dropped and are retried to be sent. In Case 3, Class 3

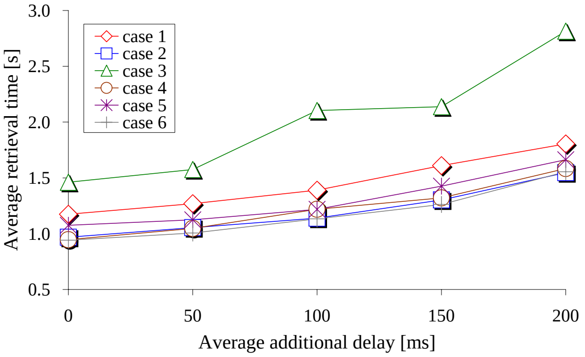

Fig. 2. Average retrieval time.

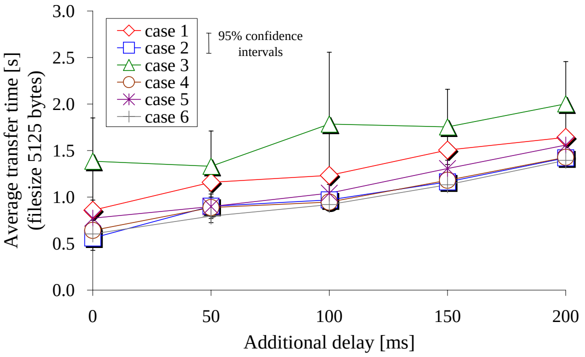

Fig. 3. Average transfer time for 5125–byte HTML files.

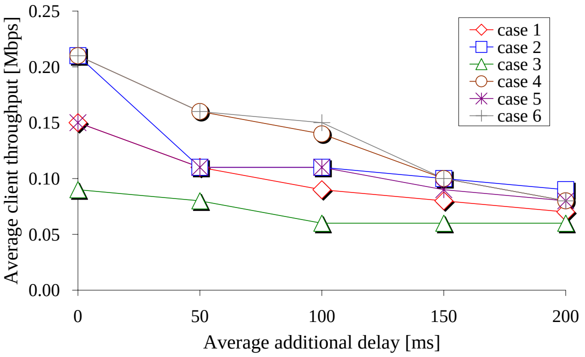

Fig. 4. HTML throughput

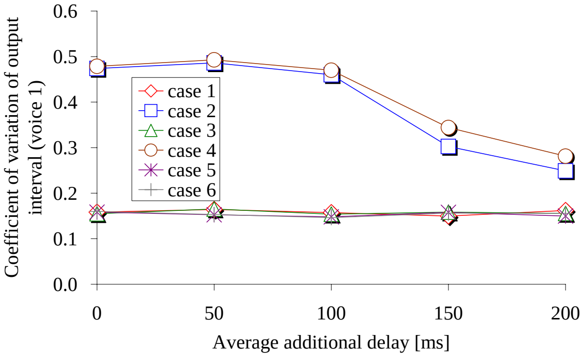

Fig. 5. Coefficient of variation of output interval for voice 1.

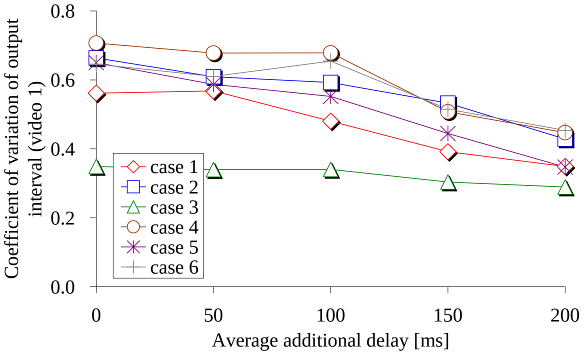

Fig. 6. Coefficient of variation of output interval for video 1.

accommodates both HTML flow and two video streams. The sum of those bit rates exceeds the guaranteed ones; in addition, the packets of the HTML files are marked with the highest drop precedence. Thus, the packets of the HTML flow drop frequently.

The HTML throughput in Case 3 is the smallest for all the additional delays in Fig. 4. However, we observe that the difference between the cases decreases as the additional delay increases. This is because, for the large additional delays, it takes longer time to expand the TCP window size. Then, the throughput cannot increase so rapidly in all the cases, even if the packets scarcely drop.

Next, we focus on the streaming service. In Fig. 5, we find that the coefficients of variation of output interval for voice 1 in Cases 2 and 4 are larger than those of the others. In the two cases, two voice streams and the HTML flow share the guaranteed bit rates of Class 2; the throughput of the HTML flow increases and then decreases alternately according to the TCP's congestion avoidance algorithm. Therefore, the temporal structure of the voice flows is disturbed. However, the difference in the coefficient value between the group of Cases 2 and 4 and the other group decreases as the additional delay increases. The reason is that the bandwidth which the HTML flow can occupy becomes narrower because of the slower expansion of the TCP sliding window; this allows the voice streams to use larger bandwidth.

In Fig. 6, we can see that the coefficient of variation of output interval for video 1 in Case 3 keeps small for all the additional delays in this figure. That is, the video quality in Case 3 keeps high and is hardly disturbed by the web access service, while the QoS of the web access service in this case is much degraded as we have already noticed. Also, Case 1 is the second best. Thus, we recommend Case 1 as the next choice to Case 3 from the intra–stream synchronization quality point of view.

In Fig. 7, we find that the average MU rate of video 1 has the same tendency of quality as that of the coefficient of variation of output interval of video 1 in Fig. 6. That is, the MU rate of Case 3 is the largest for all the additional delays in this figure. However, as the additional delay increases, the MU rates of the other cases get close to that of Case 3.

In Fig. 8, we see that the mean square error of inter–stream synchronization in each case gradually decreases as the additional delay increases, especially when the delay is more than 100 ms. Furthermore, the mean square errors for all the cases except Case 2 are less than 6400 ms^2 when the additional delay is 200 ms. This means that the inter–stream synchronization quality of those cases is fine [26]. Moreover, the mean square error in Case 3 is also the smallest in this figure when the additional delay is 50 ms through 150 ms.

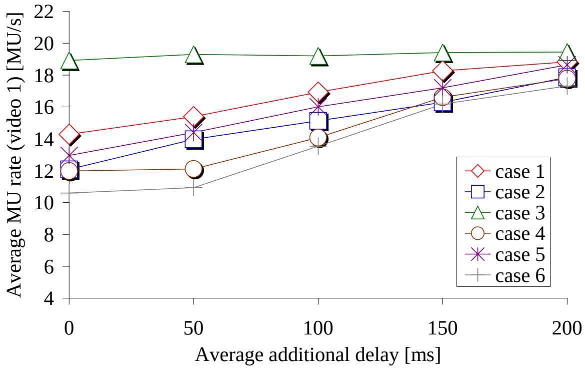

Fig. 7. Average MU rate for video 1.

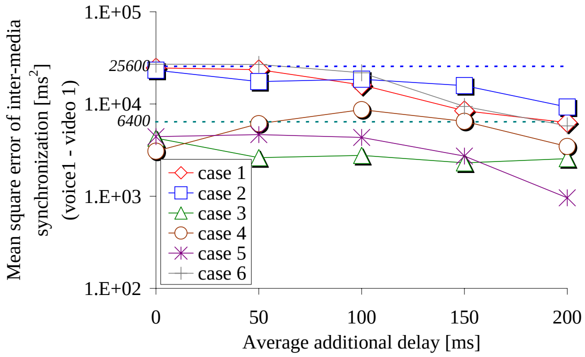

Fig. 8. Mean square error of inter–stream synchronization for voice 1–video 1.

V. Conclusions

This paper investigated how AFCP affects the application–level QoS in the AF service on DiffServ. We defined the application–level QoS parameters for the web access and streaming services; we then carried out an experiment to assess them for six cases (Case 1 through Case 6) of AFCP combinations for various additional delays between the web server and the web client.

The experiment showed the following: (i) Recommendable combinations of AFCP are Cases 1, 5 and 6 from the application–level QoS point of view. In Cases 1 and 6, the web access and streaming services are transferred as separate classes. Case 5 signifies that the web access flow and video ones are transferred as the same class, where the packets of the web access service are marked with the lower drop precedence than that of the video flows. (ii) The effect of the AFCP value on each flow becomes smaller for both services as the additional delay increases (e.g., the web clients are located far away from the target web servers). Therefore, we do not need to give much consideration to AFCP marking in such environments.

As the next step of our research, we need to investigate scalability problems, e.g., when a large number of flows are transferred or when flows are forwarded through two or more DiffServ domains.

Acknowledgment

This work was supported by the Grant–In–Aid for Scientific Research of Japan Society for the Promotion of Science under Grant 14350200.

References

[1] S. Tasaka and Y. Ishibashi, “Mutually compensatory property of multimedia QoS,” in *Conf. Rec. IEEE ICC2002*, pp. 1105–1111, Apr./May 2002.

[2] Y. Ishibashi and S. Tasaka, “A synchronization mechanism for continuous media in multimedia communications,” in *Proc. IEEE INFOCOM’95*, pp. 1010–1019, Apr. 1995.

[3] S. Blake, D. Blake, M. Carlson, E. Davies, Z. Wang and W. Weiss, “An architecture for differentiated services,” RFC 2475, Dec. 1998.

[4] R. Braden, D. Clark and S. Shenker, “Integrated services in the Internet architecture: An overview,” RFC 1633, June 1994.

[5] K. Nichols, S. Blake, F. Baker and D. Black, “Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers,” RFC 2474, Dec. 1998.

[6] B. Davie, A. Charny, J.C.R. Bennet, K. Benson, J.Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu and D. Stiliadis, “An expedited forwarding PHB (Per–Hop Behavior),” RFC 3246, Mar. 2002.

[7] J. Heinanen, F. Baker, W. Weiss and J. Wroclawski, “Assured forwarding PHB group,” RFC 2597, June 1999.

[8] N. Seddigh, B. Nandy and P. Pieda, “Study of TCP and UDP interaction for the AF PHB,” `draft-nsbnpp-diffserv-tcpudpaf-00`, June 1999.

[9] M. Goyal, A. Durresi, R. Jain and C. Liu, “Performance analysis of assured forwarding,” `draft-goyal-diffserv-afstdy-00`, Feb. 2000.

[10] M. J. Karam and F. A. Tobagi, “On traffic types and service classes in the Internet,” in *Conf. Rec. IEEE GLOBECOM2000*, pp. 548–554, Nov./Dec. 2000.

[11] N. Bhatti, A. Bouch and A. Kuchinsky, “Integrating user–perceived quality into web server design,” *Computer Networks*, vol. 33, no. 1–6, pp. 1–16, 2000.

[12] S. Chandra, C. S. Ellis and A. Vahdat, “Applicatoin–level differentiated multimedia web services using quality aware transcoding,” *IEEE J. Sel. Areas in Commun.*, Dec. 2000.

[13] M. Christiansen, K. Jeffay, D. Ott and F. D. Smith, “Tuning RED for web traffic,” in *Proc. ACM SIGCOMM2000*, pp. 139–150, Aug./Sep. 2000.

[14] H. Chen and P. Mohapatra, “Session–based overload control in QoS–aware web servers,” in *Proc. IEEE INFOCOM2002*, June 2002.

[15] S. Vegesna. *IP Quality of Service*. Cisco Press, 2001.

[16] S. Floyd and V. Jacobson, “Random early detection gateways for congestion avoidance,” *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[17] A. Demers and S. Shenker, “Analysis and simulation of a fair queueing algorithm,” in *Proc. ACM SIGCOMM’89*, vol. 19, no. 4, pp. 1–12, Sep. 1989.

[18] A. K. Parekh and R. G. Gallager, “A generalized processor sharing approach to flow control in integrated services networks: The single–node case,” *IEEE/ACM Trans. Networking*, vol. 1, no. 3, pp. 344–357, June 1993.

[19] S. Tasaka, T. Nunome and Y. Ishibashi, “Live media synchronization quality of a retransmission–based error recovery scheme,” in *Conf. Rec. IEEE ICC2000*, pp. 1535–1541, June 2000.

[20] Mindcraft Inc., “Mindcraft — webstone benchmark information,” *http://www.mindcraft.com/webstone/*.

[21] T. Berners-Lee, R. Fielding and H. Frystyk, “Hypertext transfer protocol — HTTP/1.0,” RFC 1945, May 1996.

[22] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, “RTP: A transport protocol for real–time applications,” RFC 1889, Jan. 1996.

[23] ITU–T Recommendation G. 114, “Transmission systems and media, general characteristics of international telephone connections and international telephone circuits: One–way transmission time,” Feb. 1996.

[24] J. Takeo, S. Tasaka and Y. Ishibashi, “Application level QoS assessment of continuous media transmission using the AF service in DiffServ,” (in Japanese), *Trans. of IEICE B*, vol. J85–B, no. 12, pp. 2331–2341, Dec. 2002.

[25] K. Ito, J. Takeo, S. Tasaka and Y. Ishibashi, “Media synchronization quality of AF service in DiffServ,” (in Japanese), *Technical Report of IEICE, CQ2001–38*, July 2001.

[26] R. Steinmetz, “Human perception of jitter and media synchronization,” *IEEE J. Sel. Areas in Commun.*, vol. 14, no. 1, pp. 61–72, Jan. 1996.