Streaming MPEG-4 Audio-Visual Objects with Quality Adaptation

Toufik Ahmed[1,2], Youssef Iraqi[1], Raouf Boutaba[1] and Ahmed Mehaoua[2]

[1] University of Waterloo, Dept. of Computer Science
200 University Avenue West, Waterloo,
Ont. N2L 3G1, Canada
[2] University of Versailles, CNRS-PR*i*SM Lab.
45 av. des Etats-Unis, 78000, Versailles, France

Abstract: **This paper presents an Object-based Quality Adaptation Mechanism (OQAM) for streaming unicast MPEG-4 Audio-Visual content over the Internet. This mechanism dynamically adds and drops MPEG-4 Audio-Visual Objects (AVOs) by using a TCP-Friendly Rate Control (TFRC) mechanism. TFRC adjusts the number of AVOs streamed to meet the need for rapid change in transmission rate caused by network congestion and the need for stable perceptual audio-visual quality. This end-to-end quality adaptation is combined with a Diffserv marking scheme to guarantee AVOs prioritization within the network. Performance evaluation shows that the quality of the received video adapts gracefully to network state and to heterogeneous clients capabilities.**

Keywords: Streaming video, Adaptive video streaming, MPEG-4, TCP-Friendly, Next Generation Internet, QoS.

1 INTRODUCTION

Streaming audio and video on the Internet is becoming more popular. This rapid expansion underlies a new challenge for efficient handling of Internet traffic. The majority of multimedia applications perform over an RTP stack that is implemented on top of UDP/IP. However, UDP offers no congestion control mechanism and therefore is unaware of network condition and unfair towards other competing traffic. Today's Internet traffic is dominated by TCP. TCP uses several mechanisms to handle network congestion such as: AIMD (Additive Increase and Multiplicative Decrease), slow start, congestion avoidance, fast retransmit and fast recovery. Thus, it is crucial that UDP traffic performs also TCP-friendly congestion control [1].

In this article, we consider the application scenario of streaming object oriented video over IP with congestion control mechanism. Figure 1 shows our target environment which is composed of a media server, typically MPEG-4 video server that plays video on demand for many heterogeneous clients. The client can be any terminal (PC or mobile phone) capable of rendering MPEG-4 video sequences. We consider in our topology Diffserv-enabled routers to handle stream prioritization.

We investigate Quality of Service (QoS) interaction provisioning between an MPEG-4 video application and the IP Diffserv network. To achieve the best possible QoS, all the components involved in the transmission process must collaborate together. In this regards, we propose two mechanisms. The first one is the rate adaptation mechanism. The server performs rate adaptation through the adjustment of the number of streamed object based on network state. We use a TCP-friendly to adapt the server rate to network condition. The server tries to deliver the maximum number of objects that can fit in the current available bandwidth. The second mechanism is a DiffServ marking scheme. The server must be aware of each audio-visual object (AVO) in the scene and must be able to classify these objects in a hierarchical manner, from less important to more important object. This mechanism allows the server to: (1) deal with network congestion by stopping streaming less important object when congestion is detected and (2) prioritize the transport of important object by an intelligent marking for the Diffserv network. When network congestion occurs less important AVOs will be dropped automatically by the network. Lost packets make the server reduce its transmission rate.

The idea of TCP-friendly helps to prevent the application entering congestion collapse in which the network link is heavily used and little useful work is being done. So to prevent such situation, all applications must perform TCP-like congestion control mechanisms. Traffic that does not perform in TCP-friendly manner is dropped by the router [2].

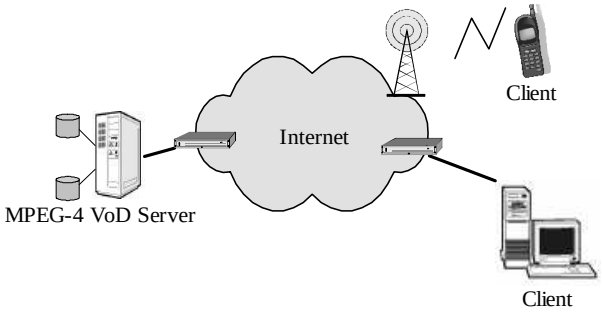


Figure 1: MPEG-4 Video on Demand service

The paper is organized as follows: Section 2 reviews related work. The MPEG-4 framework is presented in Section 3. OQAM is presented in Section 4. Simulation model and performance analysis are presented in Section 5 and 6 respectively. Finally, Section 7 concludes the paper.

2 BACKGROUND AND RELATED WORK

Adaptive video streaming is not a new topic. Several researches have been conducted in this area and various approaches have been proposed. While all the work done until now concerns essentially layered video streaming, our work is different from others by using the new concept of object scalability introduced in the MPEG-4 standard. This section presents the different mechanisms of adaptive video streaming in general, and then focuses on quality adaptation mechanisms used in the context of scalable video streaming.

2.1 Adaptive Video Streaming

Adaptive video streaming uses a feedback mechanism which may be received from the network or from the end system. Video can be adapted according to changing conditions in the network (congestion level, bandwidth usage, etc.). Figure 2 shows two possible feedback mechanisms.

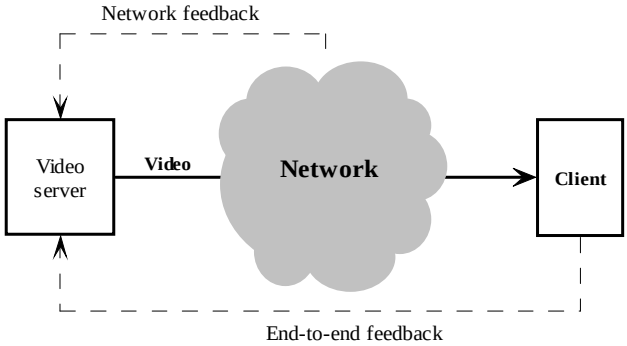


Figure 2: A Video adaptation with a feedback mechanism.

Video adaptation can be achieved using different approaches which include:

1. Single stream adaptation approach:

   In this approach a single encoded video stream is transmitted from the video encoder to the receiver. Feedback information is used by the source to adapt its rate by tuning the codec quantization parameters. This approach cannot cope with receiver heterogeneity.

2. Replicated stream adaptation approach:

   In this approach the source sends multiple streams carrying the same video with different qualities and bit rates to different receivers. The feedback information is used to adjust the rate of the streams. While this approach addresses the problem of heterogeneity, it requires the network to carry redundant information.

3. Layered video stream adaptation approach:

   In this approach the source divides the video stream into layers. A base layer and one or more enhancement layers. The enhancements layers improve the video quality but cannot be decoded without the base layer. The feedback information is used to adapt the rate of each layer and the number of layers being generated. The drawback of this approach is that a particular layer cannot be decoded if one lower layer is missing.

2.2 Related Work

In the context of multicast video delivery, the authors in [3] attempt to improve the received video quality within each layer by retransmitting lost packets that have an acceptable recovery time and by applying an adaptive playback scheme. Hierarchical rate control is used to adapt to network congestion state and to receiver heterogeneity.

In [4], McCanne proposes a method that enables different receivers to adapt to bandwidth fluctuations by adjusting the number of layers to which they subscribe. The video stream is divided into a number of multi-resolution layers. Each layer is transmitted to a separate multicast group.

In the context of unicast video delivery, the authors in [5] assume that video layers are being coded at the same rate. They propose an algorithm to adapt the video quality to network state by adding and dropping layers to efficiently use

the available bandwidth. Their algorithm takes into consideration the status of the receiver buffer, making sure that base layer packets are always available for playback.

Another type of server rate adaptation is to adjust the codec quantization parameters according to the received feedback. In [6] and [7] the video server continually negotiates the available bandwidth and modifies the codec quantization values accordingly. Quality degradation and buffer management are not considered in these works. Moreover, adapting the codec quantization values is a CPU-intensive task which affects the performance of the video server. The idea of quantizer scale was also used in the context of MPEG-4 in the work presented in [8]. The later employs Fine Granular Scalability which uses a layered coding algorithm. A TCP-friendly rate control algorithm adjusts the rate of each video layer by regulating the level of quantization.

CPU-based adaptation is investigated in [9] in the context of local area networks, where each client receives a particular rate depending on its CPU power.

In contrast to previous adaptive video streaming mechanisms, the proposed approach in this paper, uses the concept of object scalability introduced in MPEG-4. It adapts video quality to network state by adding or dropping objects and their associated layers according to the network state. This solves the problems of heterogeneity of receivers and redundancy of data. Objects are encoded separately which does not prevent one object from being decoded if another one is not received.

3 MPEG-4 AUDIO VISUAL OBJECTS

The MPEG-4 standard [10][11][12] introduces a new technique of coding multimedia scenes called “object-based compression”. This technique allows the encoding of different audio-visual objects in the scene independently.

This new technique provides a support of new ways of communication, access, and interaction with digital audio-visual data. It also offers solutions to some common technical problems in various telecommunications, broadcast, and interactive services. MPEG-4 is designed to be used in a broad range of existing and emerging multimedia applications such as video on the Internet, multimedia broadcasting, content-based audio-visual database access, games, audio-visual home editing, advanced audio-visual communications, and video over mobile networks.

An MPEG-4 scene consists of one or more AVOs, each of them is characterized by temporal and spatial information. The hierarchical composition of an MPEG-4 scene is depicted in Figure 3. Each Video Object (VO) may be encoded in a scalable (multi-layer) or non scalable (single layer) form. A layer is composed of a sequence of a Group of Video-Object-Plane (GOV). A Video Object Plane (VOP) is similar to the MPEG-2 frame. VOP supports intra coded (I-VOP) temporally predicted (P-VOP) and bi directionally predicted (B-VOP)

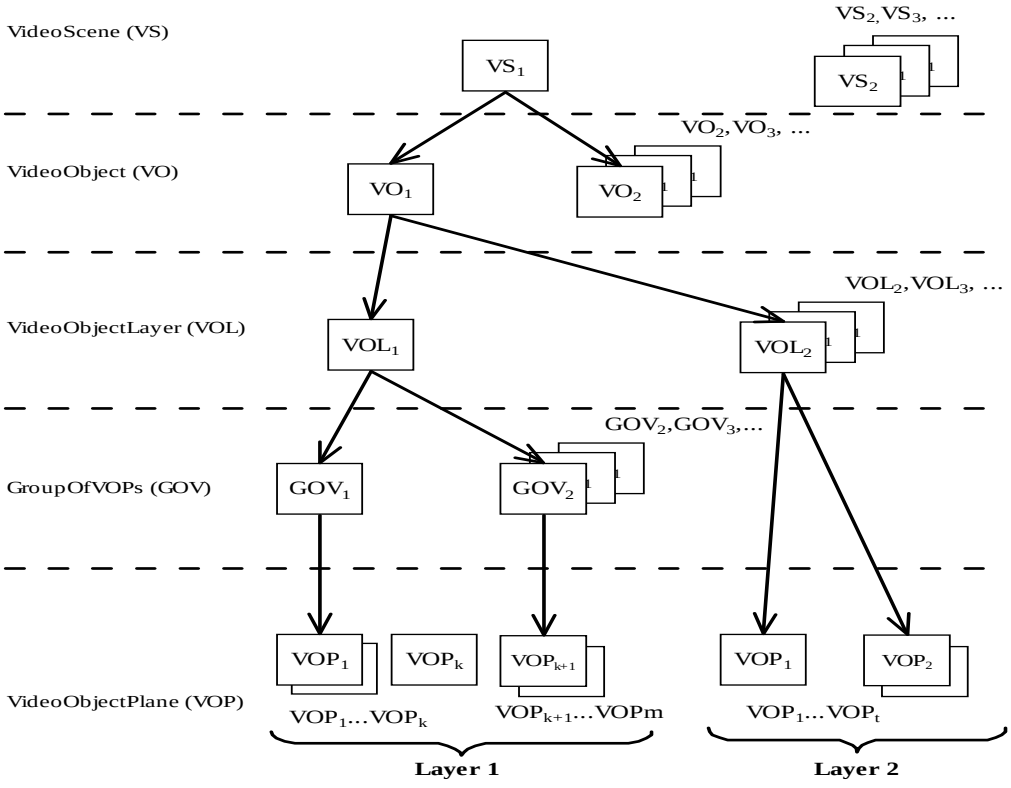


Figure 3: Hierarchical composition of an MPEG-4 scene

To take benefits from the object-based compression, we propose to use an intelligent adaptation to cope with network congestion and client terminal heterogeneity. We propose to sort MPEG-4 AVOs at the video server from most important AVO to least important AVO. Several methods can be used for objects classification. During scene creation, one can affect the adequate priorities to each object in the scene. For scenes with no assigned object priorities, MPEG-4 object descriptors or MPEG-7 [13] can provide the relevant information needed to handle object priority. Object priority is out of the scope of this paper and will be considered in future work.

Figure 4 depicts an MPEG-4 scene, along with the associated object tree structure. The scene shows a weather presentation that contains four objects: logo, speaker, background and speech. These objects can be sorted from most important to least important. In this example, it appears that the logo object has less importance for the end users. Hence, we can classify these objects according to their degree of importance in the scene. For example: speech, background, speaker and logo.

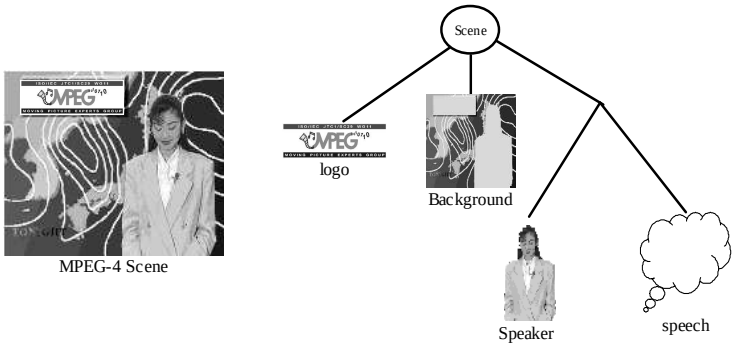


Figure 4: Simple MPEG-4 scene with four objects

By sorting these objects, we provide a first level of scalability called object scalability. It gives the server the ability to add and drop video objects dynamically and deal with network congestion intelligently.

In this paper we propose a framework for video transmission using the following two mechanisms:

- A mechanism for adding and dropping AVO according to network condition. It is performed by the server to maintain an acceptable transmission rate while being fair to other network traffic.
- A mechanism for AVO DiffServ marking to deal with congestion in the network.

These mechanisms are orthogonal to each other. The transport mechanism uses a TCP-friendly scheme to adapt its rate to the congestion level, while the network handles stream prioritization based on the marking in the IP packet header decided by the server. Combining these two mechanisms provides the best level of QoS of the received MPEG-4 scene.

4 TCP-FRIENDLY OBJECT-BASED QUALITY ADAPTATION MECHANISM (OQAM)

The idea of TCP-friendly transport protocol is to emulate TCP behavior without replicating TCP mechanism. By definition, a flow is said to be TCP-friendly if its arrival rate does not exceed the arrival rate of a conformant TCP implementation in the same circumstances [14]. Many TCP-friendly congestion control mechanisms were developed recently, among which: Rate Adaption Protocol (RAP) [15], Loss-Delay Based Adaptation Algorithm (LDP) [16] and TCP-friendly Rate Control Protocol (TFRC) [17]. While these protocols and others are comparable in their features in simulating TCP behavior, TFRC seems to be more robust and is expected to become a standard. It provides sufficient responsiveness by taking into consideration all the parameters that affect the TCP rate such as loss, Round-Trip Time (RTT) and retransmission timeout value. The key advantage of TFRC is that it has a more stable rate during the session lifetime.

Our video quality adaptation mechanism is based on TFRC. It operates as follows:

- The receiver measures the loss rate and feeds this information back to the sender. This is achieved by a modified version of RTP and RTCP protocols [18]. Each RTP packet has a timestamp and a sequence number that allow the receiver to compute the packet loss and the sender to compute the RTT.
- The loss rate and the RTT are then fed into the TFRC module to get the appropriate transmission rate (cf. Eq. 1 below).
- The sender then adds or drops audio-visual objects and the associated layers if any, to adjust its transmission rate to match the target rate (i.e. allowed rate).

The calculated rate is obtained by using the TFRC equation [17]:

$$R_{TCP} \cong \frac{s}{RTT\sqrt{\frac{2bp}{3}} + t_{RTO}(3\sqrt{\frac{2bp}{8}})\, p(1+32p^2)} \quad (1)$$

Where R_{TCP} is the target transmission rate or the allowed transmission rate, s is the packet size, RTT is the round trip time, p is the loss rate, t_{RTO} is the TCP retransmission timeout value and b is the number of packets acknowledged by a single TCP acknowledgement.

4.1 Notations and Parameters

Let S be a set of MPEG-4 AVOs containing n AVOs O_j, with j∈{1, 2…n}. Without loss of generality, we assume that these objects are sorted in a decreasing order of priority. Each object O_j may consist of m_j layers ($m_j \geq 1$). Note that lower layers within an object have higher priorities than higher layers.

Let **P** be the function that returns the priority of a particular object or layer. Without loss of generality, we assume that:

$$\forall j, 1 \leq j < n : P(O_{j+1}) \leq P(O_j)$$
$$\forall j, 1 \leq j < n, \forall l, 1 \leq l < m_j : P(L_{j,l+1}) < P(L_{j,l}) \quad (2)$$

$L_{j,l}$ is the Layer number l of the Object O_j

By using Eq. 2 we can construct an Audio-Visual Entity set called E composed of all object layers ordered by their priorities.
$E = \{L_{1,1}, L_{1,2}…L_{1,m1}, L_{2,1}, L_{2,2}…L_{2,m2}, …, L_{n,1}, L_{n,2}…L_{n,mn}\}$. We will note E as follows:

$E = \{e_1, e_2, ..., e_W\}$ with $w = |E| = \sum_{j=1}^{n} m_j$

Note that if two objects have the same priority, then the associated layers of an object have the same priority as the object (in relation to other objects) with the lower layers having higher priorities than higher layers.

At time t_i, the function R_i gives the instantaneous transmission rate of an audio-visual entity. For example, the audio-visual entity $\mathbf{e_p}$ has an instantaneous transmission rate equal to $R_i(e_p)$, and the object O_j has the instantaneous transmission rate equal to $R_i(O_j)$.

OQAM (Object-based Quality Adaptation Mechanism) operates as follows: The server evaluates the network state from the information gathered (i.e. RTT and loss rate) at time t_i, then computes the allowed sending rate R_{TCP} using Eq. 1. The server tries to send as much as possible of the audio-visual entities without exceeding R_{TCP} taking into consideration entities priorities. Details of the adding and the dropping process will be presented in section 4.3 and 4.4 respectively.

4.2 Example

Assume that we have an MPEG-4 scene composed of four audio-visual objects: O_1, O_2, O_3 and O_4. Assume that O_1 is composed of a single layer, and that each of O_2, O_3 and O_4 is composed of three layers (one base layer and two enhancement layers). Also assume that the associated priorities are as follows (see Figure 5):

- O_1 is the most important
- O_2 and O_3 have the same priority
- O_4 is the less important

Then, E= { $L_{1,1}$, $L_{2,1}$, $L_{3,1}$, $L_{2,2}$, $L_{3,2}$, $L_{2,3}$, $L_{3,3}$, $L_{4,1}$, $L_{4,2}$, $L_{4,3}$} = {e_1,e_2, …, e_{10}}. Here w=10.

The video server adds audio-visual entities in the order of their importance (i.e. form left to right in the set E). Entities are dropped in reverse order (i.e. form right to left) until matching the target sending rate.

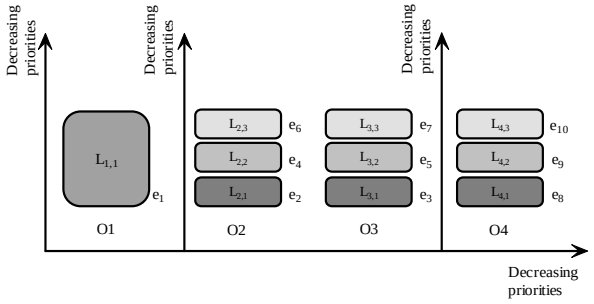


Figure 5: Handling priorities between layers and objects

4.3 Adding an Audio-Visual Object

The server adds a new audio-visual entity as soon as the target rate exceeds the current sending rate of current entities plus the new entity. Assume that the server is streaming ***k*** entities at time t_i. We assume also that the client has sufficient resources to play all the entities being sent by the server. Therefore, at time t_{i+1} the server can add a new entity while the following condition is satisfied:

$$\sum_{j=1}^{k+1} R_{i+1}(e_j) \leq R_{TCP}$$

At the client side, the new audio-visual entity must be buffered and synchronized to the current playback time.

4.4 Dropping an Audio-Visual Object

When the estimated throughput of the TCP session indicates that the video server is transmitting more data than it should, then the video server must reduce its sending rate by dropping one or more audio-visual entities. Therefore, the server drops entities while the following condition is satisfied:

$$\sum_{j=1}^{k} R_{i+1}(e_j) > R_{TCP}$$

4.5 Handling stability

Since the TFRC compute the new target rate each RTT, adding and dropping audio-visual entities can lead to undesired oscillation and poor video quality at the receiver. To prevent from such behavior, several measures are taken into consideration.

First, the TFRC module copes with oscillation behavior by using EWMA (Exponentially Weighted Moving Average) to detect out-of-control situations quickly. EWMA statistics are used to attempt to respond dynamically to the changing value in the measured RTT and loss and attempt to regulate this value to reflect as much as possible the reality. In TFRC, the loss rate is measured in terms of loss interval which represents the number between two consecutive loss events [17]. The mechanism reacts too strongly to single loss events and ensures that allowed sending rate do not change aggressively.

Second, we propose to adjust the server transmission rate at the beginning of each GOV (Group of video object plane). Thus, the new transmission rate obtained from TFRC module is used to adapt video sending rate. Figure 6 shows four GOVs (each group has twelve VOPs). The average line in the Figure shows the server transmitting rate at the beginning of each GOV of a current Video Object (VO). If this value does not fit in the current available bandwidth then the server does not stream the object.

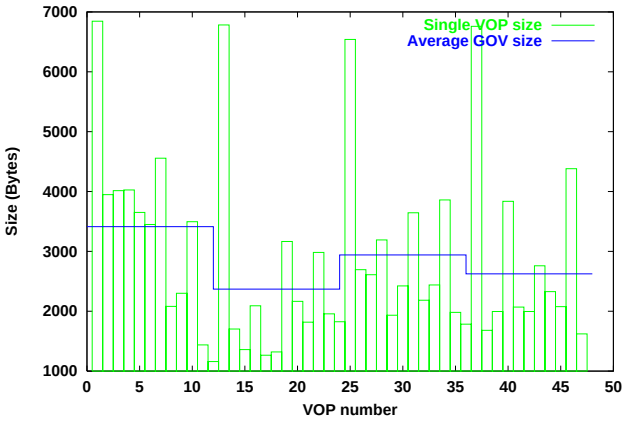


Figure 6: Video Object Plan sizes

4.6 System architecture

Figure 7 depicts the general block diagram of our MPEG-4 Video on Demand system. It is composed of a video server and a video client. The server streams the audio-visual object to the client via an IP network using the RTP protocol. The client decodes and composes the original MPEG-4 scene. As shown in Figure 4, each AVO is coded separately so the decoding process decodes also each AVO separately and then the composition module composes the original scene. The target transmission rate of the video server is calculated by the TFRC module. This information is sent to the

"add/drop module" which adapts the video transmission rate using add/drop algorithms. IP Diffserv Marker module handles the marking of the different RTP packet with Diffserv Code Point before entering the Diffserv network.

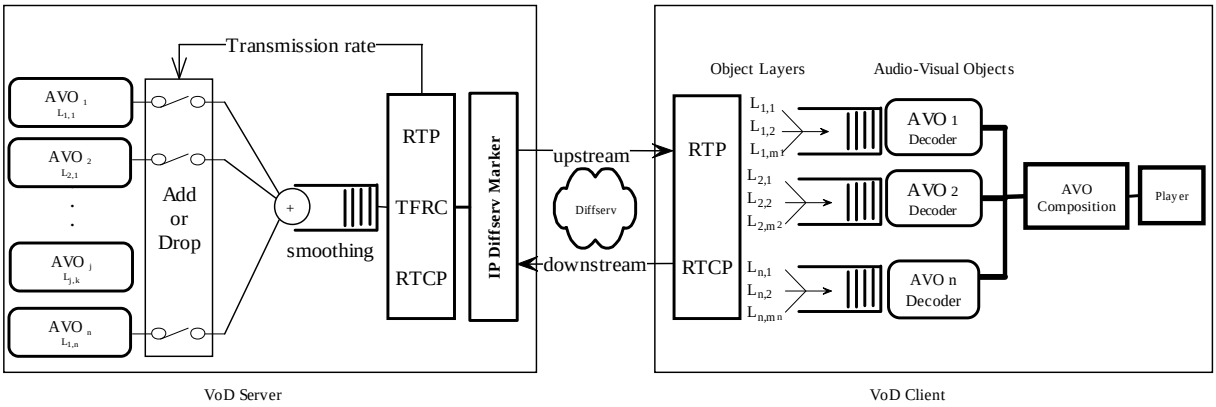


Figure 7: General Block Diagram of the MPEG-4 VoD System

Diffserv object prioritization aims to privilege the transport of some AVOs compared to others. When network congestion occurs, less important AVOs streams are dropped automatically by the active queue implemented in the Diffserv router. The work detailed in [19] presents a method to handle a layered MPEG-4 stream over IP Diffserv. We extend this approach by handling MPEG-4 AVOs streams prioritization over IP Diffserv network. Recall that the MPEG-4 scene contains many MPEG-4 AVOs sorted according to their importance in the presentation. Therefore, the IP Diffserv Marker tags each video data packet belonging to one AVO with one of the supported Diffserv class of service to reflect object priority. Hence, important objects will be marked with a low drop precedence to guarantee a minimum loss.

It is worth noting that the choice of TCP or TFRC-based congestion control mechanisms is completely orthogonal to whether the traffic is best-effort or not. If the transport protocol is using conformant end-to-end congestion control, then the transport protocol does not have to know whether the traffic is being treated as best-effort or as part of a Diffserv class.

5 SIMULATION MODEL

5.1 Network architecture

Simulations are conduced using the network simulator *ns2*. We used the network architecture shown in Figure 8 to simulate a unicast service provided by the MPEG-4 server attached to the node "S". The server sends data to the client attached to the node "C". Our server is an ns2 agent that uses TFRC module to adapt the number of transmitted AVO. The client is also an ns2 agent which extends the capabilities of the RTP sink by reporting statistic information to the server. The network is loaded by *n* FTP streams carried over TCP (*n* ranges from 0 to 8). This allows the link between the routers "R1" and "R2" to be congestioned differently. FTP sources always have a packet to send and always send a maximal-sized (1000-bytes) packet as soon as the congestion control window allows them to do so. FTP sink immediately sends an ACK packet when it receives a data packet. The queue in the routers has a size of 50 packets. The core IP Diffserv router examines incoming packets and reacts according to the marking, whereas "R1" is an edge router that implements Marking/Classification policy on incoming packets. R1 uses A Two Rate Three Color Marker (**TR3CM**) [20] to mark the background. Therefore, background traffic is evenly distributed among the different Diffserv classes. We recall that the video traffic is marked at the MPEG-4 server according to AVOs priorities. The bottleneck link between the core router and R2 has a 5 Mbit/s of bandwidth.

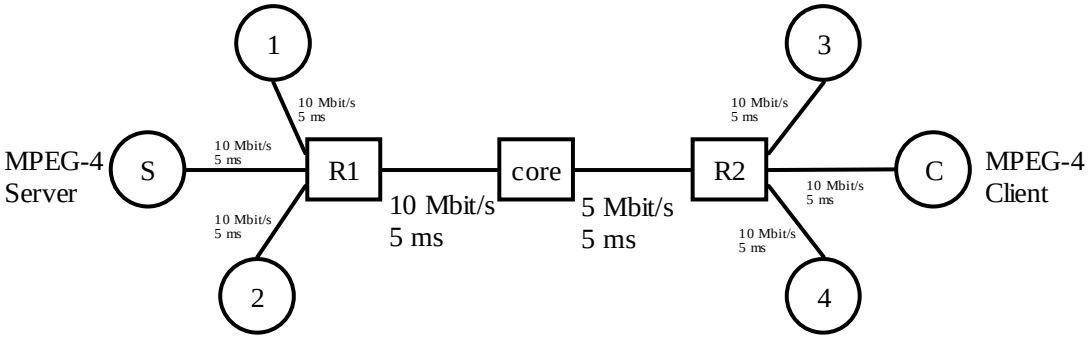


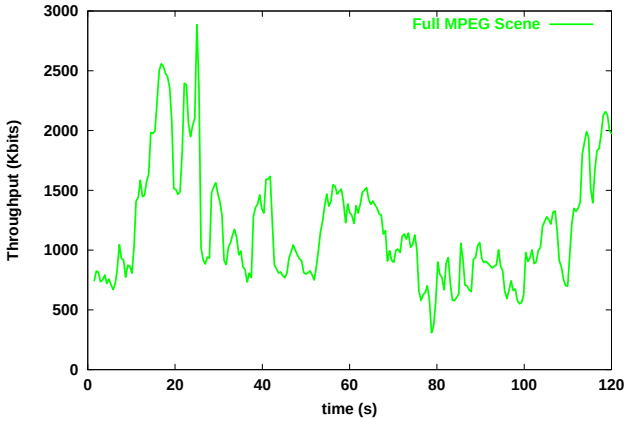
Figure 8: Network topology

5.2 MPEG-4 Traffic model

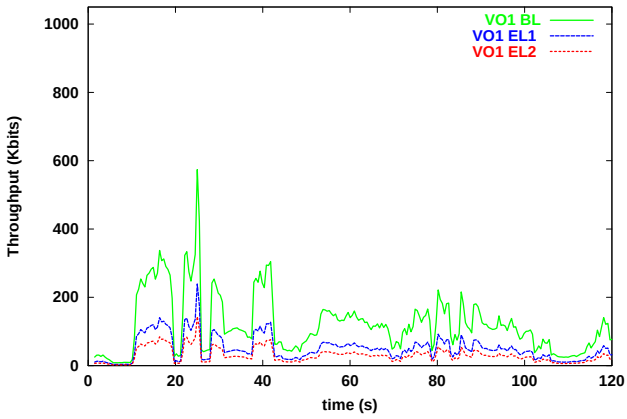
The MPEG-4 traffic is obtained from the MPEG-4 trace file presented in [21]. In our simulation, the MPEG-4 presentation was obtained by using a set of AVOs components. We simulate the weather presentation shown in Figure 4 by using four multimedia objects: AO (audio speech), VO1 (background), VO1 (speaker) and VO3 (logo). These objects are sorted as follows:

- AO has the priority 1, it is the most important object in this scene. It is marked with Diffserv PHB AF11 (low drop precedence).
- VO1 and VO2 have the priority 2. They are marked with Diffserv PHB AF12 (medium drop precedence). Each Object is composed of 3 layers (one base layer and 2 enhancement layers)
- VO3 has the priority 3, it is the least important object in this scene. It is marked with Diffserv PHB AF13 (high drop precedence).

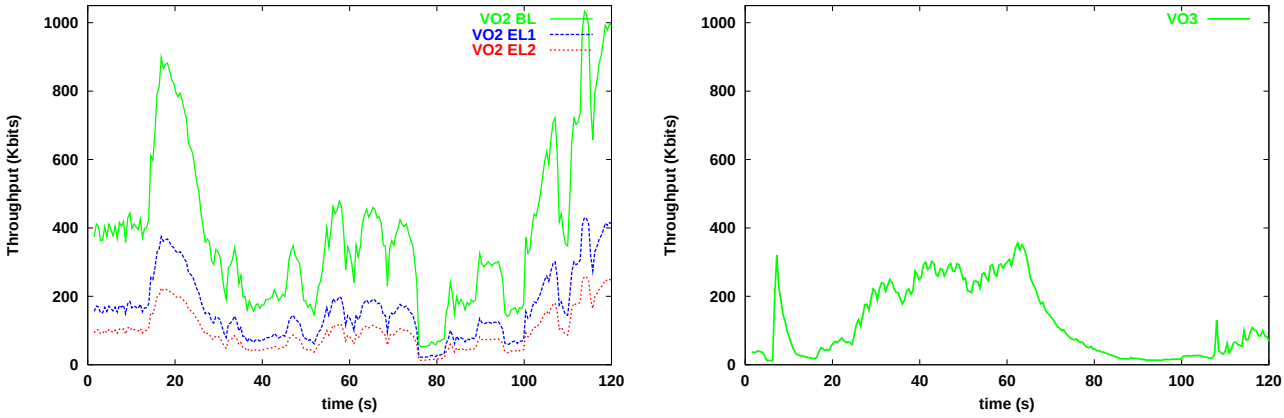
Figure 9 shows the bit-rate of the MPEG-4 video objects that can be sent from the MPEG-4 server to the client during a period of 120 seconds. The complete scene is shown in Figure 9 (a). The Audio Object is a constant bit rate at 64Kbits/s. An Audio packet is sent each 125ms. Video object 1 has an average throughput of 200 Kbit/s and a peak rate of 956 Kbit/s. This object is composed of three Layers: BL (Base Layer), EL1 (Enhancement Layer 1) and EL2 (Enhancement Layer 2). The throughputs of the different layers are shown in Figure 9 (b). Video object 2 has an average throughput of 650 Kbit/s and a peak rate of 1722 Kbit/s. This object is composed of three Layers: BL, EL1 and EL2. The throughputs of the different layers are shown in Figure 9 (c). Video object 3 has an average throughput of 124 Kbit/s and a peak rate of 356 Kbit/s. It is composed of one single layer (see Figure 9 (c)).



(a)



(b)



(c) (d)

Figure 9: Instantaneous throughput of the different MPEG-4 Video Object

6 SIMULATION ANALYSIS

We perform an intensive simulation, each time with different parameters to see the behavior of our video on demand system. We vary the number *n* of FTP source according to following scenarios: (1) **Scenario A**: one FTP source; (2) **Scenario B**: two FTP sources; (3) **Scenario C**: four FTP sources; (4) **Scenario D**: eight FTP sources. FTP sources send data from time t=30s until time= 90s.

This section presents some QoS measurement such as, the video server throughput as a function of network state, packet loss and end-to-end packet transmission delay.

6.1 Video Server Throughput

The video server regulates its transmission rate to reflect the allowed rate by adding or dropping audio-visual entities. Results obtained of the different scenarios are shown in Figures below. Also, to simplify the interpretation of the results, Table 1 summarizes the transmission ratio per AVO stream observed during the period of the simulations (120s). Note that the FTP sources begin data transmission at time t=30s, and stop at time t=90s. VO3 has the low ratio since it has the lowest priority is the scene. VO1 and VO2 have the same priority, so the corresponding layers have more or less the same transmission ratio.

From the result in scenario A (Figure 10), we can see that the MPEG-4 video is transmitted entirely. The FTP source adapts to the change caused by the video throughput and tries to consume all the available bandwidth. The bottleneck link is 100% used when the FTP source starts the transmission. In this scenario, there is no loss because we have two streams that fairly share network resources. This gives a portion of 2.5 Mbit/s per stream. Since our MPEG-4 scene consumes less than 2.5 Mbit/s, the rest of the bandwidth is used by the FTP source.

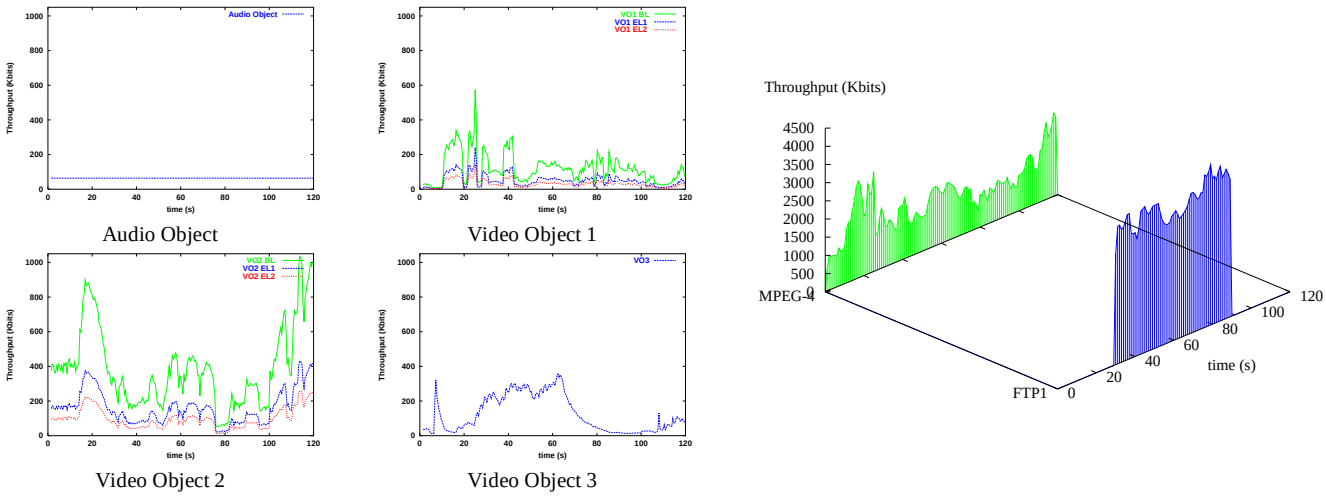


Figure 10: Scenario A

In scenario B (Figure 11), we have two FTP sources. The results show that the video stream is transmitted entirely and that the two FTP sources fairly share the remaining bandwidth. This gives a portion of 1.66 Mbit/s per flow since we have three streams. In this period, when the two FTP streams are active (30s-90s), the video stream consumes less than 1.66 Mbit/s, the remaining bandwidth is fairly shared with the two FTP sources.

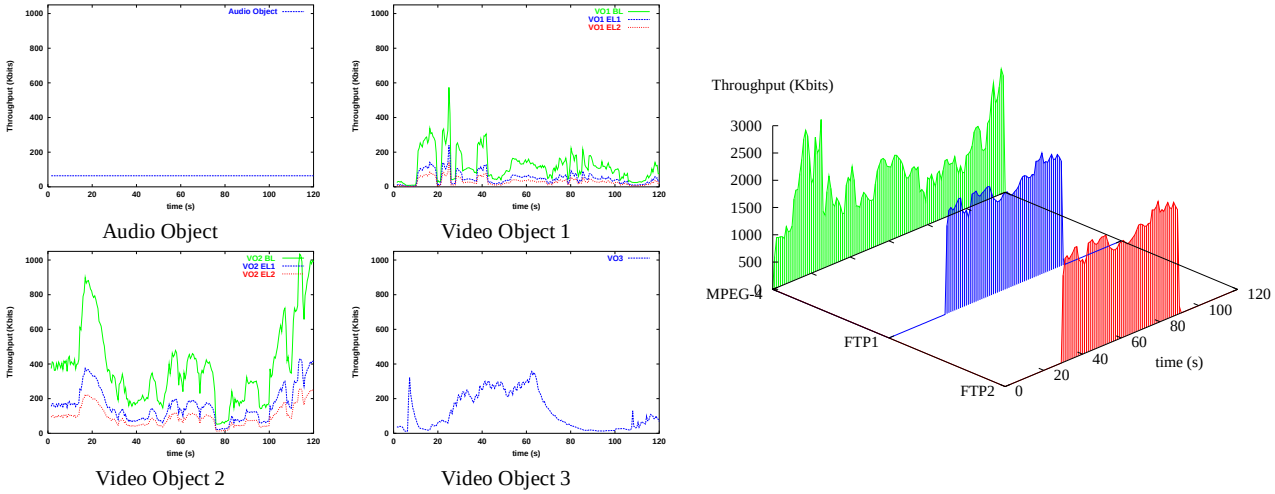


Figure 11: Scenario B

Scenario D is interesting since we see the effect of our adaptation mechanism. We can see that the audio object is always present and that less important objects (respectively object layers) are not transmitted when the shared bandwidth is not sufficient. Our adaptation mechanism begins transmitting data from important audio-visual entity to less important. We can see that all the streams (FTP and video) fairly share the bandwidth.

Scenario C confirms the previous result and shows the effect of our adaptation mechanism. A minimum of QoS is guaranteed by our adaptation mechanism. The network does not enter in congestion collapse since the video stream is aware of network condition.

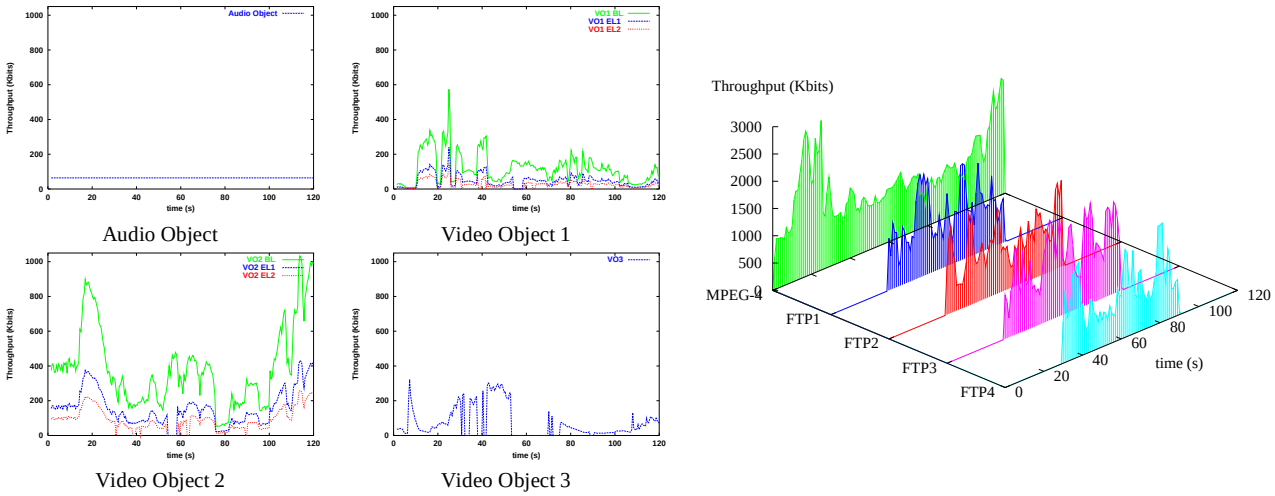


Figure 12: Scenario C

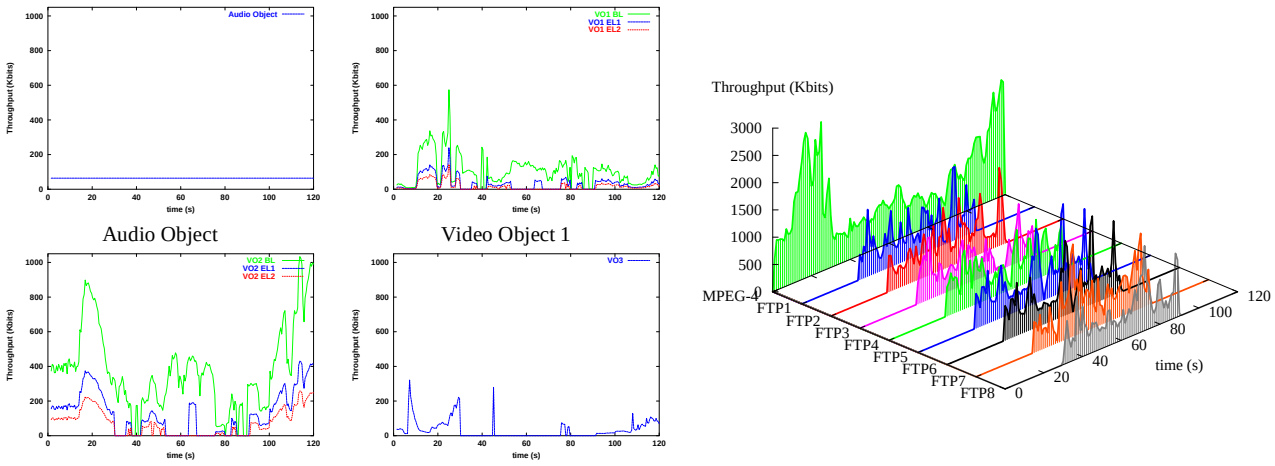


Figure 13: Scenario D

The Table below shows the server transmission ratio per audio-visual entity observed during the period of the simulations (120s). Note that VO3 has the low ratio since it has the lowest priority is the scene. VO1 and VO2 have the same priority, so the corresponding layers have more or less the same transmission ratio.

Table 1: Transmission ratio per MPEG-4 objects

AVO / Scenario	Audio	VO1			VO2			VO3
		BL	EL1	EL2	BL	EL1	EL2	
A	100%	100%	100%	100%	100%	100%	100%	100%
B	100%	100%	100%	100%	100%	100%	100%	100%
C	100%	100%	94%	87%	100%	96%	92%	55%
D	100%	89%	60%	53%	97%	77%	71%	26%

Without using OQAM, the server transmits the audio-visual entities without any regulation as shown in Figure 9. The loss may increase and the network may enter in congestion collapse.

6.2 Packet Loss

Figure 14 shows lost packets for scenarios B, C and D using OQAM. Scenario A does not experience any loss. In scenario B, some lost packets are observed on VO3. This is due to the active queue of the DiffServ router which drops lower priority packets when a predefined threshold is reached to prevent congestion. In scenario C, we observe also some loss on lower priority packets but in scenario D high priority packet are also dropped. This is due to: (1) lower priority packets are not transmitted because OQAM regulates the server transmission rate by stopping streaming lower priority packets and (2) AVO1 and AVO2 request more bandwidth in our scene and cause some congestion.

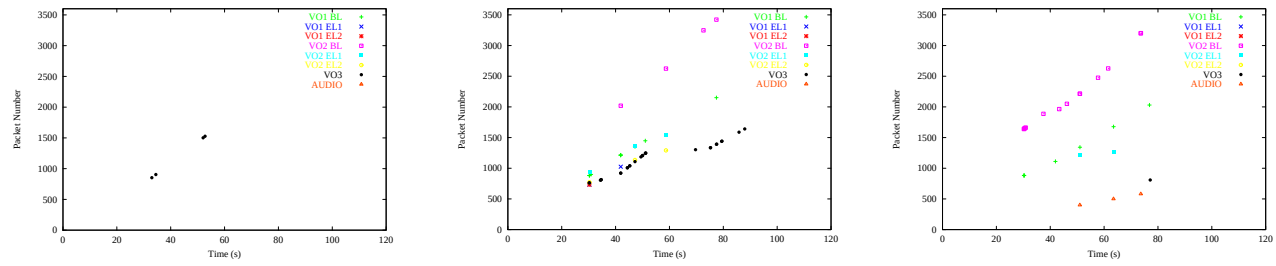
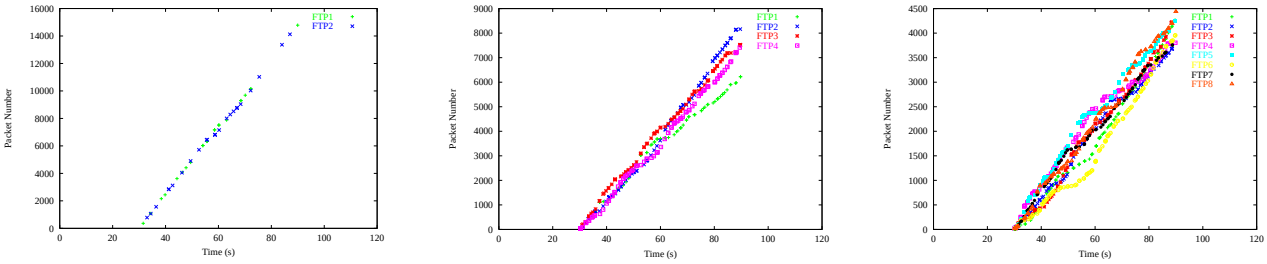


Figure 14: MPEG-4 Packet Loss

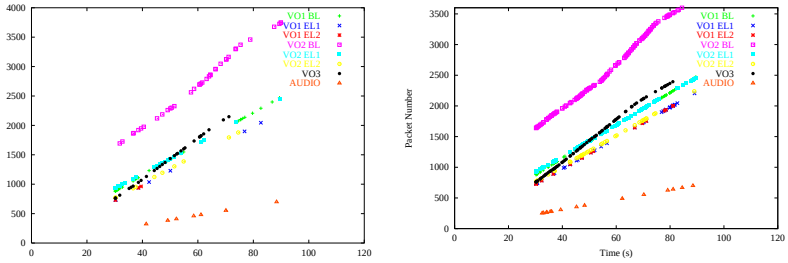
Figure 15 shows FTP packet loss observed in the same scenarios. FTP packets encounter more loss than the video packets due to two factors. First factor is that FTP traffic is marked using TR3CM marker which distributed the marked traffic among the different classes of the DiffServ network. We remark that the majority of dropped packets are those marked with high drop precedence. Second factor is that FTP source does not regulate the traffic by computing the allowed transmission rate rather it uses window-based congestion control mechanism.



Scenario B Scenario C Scenario D

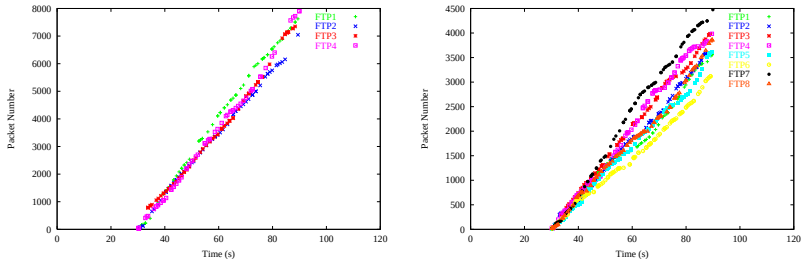
Figure 15: FTP Packet Loss

We have redone the same simulation without using OQAM. The results are shown in Figure 16 and Figure 17. Remark that in scenario B no loss is observed in both of MPEG-4 and FTP streams. In scenario with OQAM loss is due to the active queue on the core router which prevents the congestion by an early packet discard mechanism. Without OQAM, more packets are dropped because the server does not regulate its transmission rate at the source. Important video packets are also dropped with the same probability due to Drop Tail queue management used by the router. The damage caused by some data loss in some reference picture such as I-VOP or P-VOP will affect subsequent picture(s) due to inter-frame predictions. For example when the I-VOP is lost, the whole dependant P-VOP and B-VOP cannot be decoded. The same conclusion is valid for hierarchical streams. Hence, Enhancement Layer 1 cannot be decoded without the reception of Base Layer, and so on. When using OQAM, low important audio-visual entities (those marked with high drop precedence) are not transmitted by the server when the allowed rate decreases. This helps to prevent a future drop by the router. So the regulation is done at the server and demonstrates clearly the advantage of our mechanism.



Scenario C Scenario D

Figure 16: MPEG-4 Packet Loss without OQAM

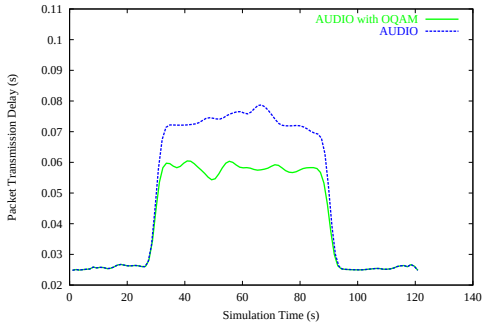


Scenario C Scenario D

Figure 17: FTP Packet Loss without OQAM

6.3 End-to-end Transmission Delay

Figure 18 shows the end-to-end packet transmission delay (PTD) for the audio object in case when using OQAM and without OQAM. PTD variations are correlated with the router queue size and the packets lost. The more the queue is on congestion, the more the delay for a packet to reach the destination is increased. Since the queue in the DiffServ is based on active queue management that maintains the queue size as small as possible, then PTD is small in case of OQAM. Same results are obtained with the others streams.



Scenario D

Figure 18: End-to-end Packet Transmission Delay

7 CONCLUSION

We have proposed an adaptation mechanism for MPEG-4 video streams that uses a TCP-Friendly Rate Control. Our mechanism adds and drops MPEG-4 Audio-Visual Objects to perform rate adaptation and congestion control. We have evaluated the proposed mechanism through simulations using *ns2*. The MPEG-4 server implemented in ns2 uses the TFRC module as an equation-based congestion control mechanism. We coupled end-to-end congestion control with a Diffserv network that guarantees objects prioritization within the network. The simulation results show that important multimedia entities are maintained by the router in case of network congestion. Combining these mechanisms in one architecture demonstrates clearly the gains obtained.

REFERENCE

[1] Mahdavi, and S.Floyd "TCP-Friendly Unicast Rate-Based Flow Control", Technical note sent to the end2end-interest mailing list, January 8, 1997.
[2] S.Floyd, M. Handley, J. Padhye, and J. Widmer "Equation-based congestion control for unicast applications" Proc. of ACM SIGCOMM, pages 43–56, 2000.
[3] X. Li, S.Paul, and M. Ammar "Layered Video Multicast with Retransmission (LVMR): Hierarchical Rate Control Schemes" Proc. of Infocom 98, San Francisco, CA, March 1998.
[4] S. McCanne" Scalable compression and transmission over Internet multicast video" Ph.D thesis University of California, Berkeley. December 1996.
[5] R.Rejaie, M.Handley, and D. Estrin "Layered quality adaptation for Internet video streaming" IEEE Journal of selected areas in communications, vol. 18, No. 12, December 2000.
[6] T.V.Lakshman, P.P.Mishra, and K.K.Ramakrishnan "Transporting compressed video over ATM networks with explicit rate feedback control," in *Proc. IEEE* Infocom 97, pp. 38–47, April 1997.
[7] N.G.Duffield, K.K.Ramakrishnan, and A.R. Reibman "SAVE: An Algorithm for Smoothed Adaptive Video Over Explicit Rate Networks" IEEE/ACM transactions on networking, vol. 6, no. 6, December 1998.
[8] N.Wakamiya, M.Miyabayashi, M.Murata, and H.Miyahara "MPEG-4 Video Transfer with TCP-friendly Rate Control" in IFIP/IEEE MMNS 2001, pp 29-42 October 2001.
[9] S.D. Servetto, R.Puri, J.P. Wagner, P.Scholtes, and M.Vetterli "Video Multicast in (Large) Local Area Networks" in the Proceedings of IEEE Infocom 02, New York, NY, June 2002.
[10] ISO/IEC 14496-1 "Coding of audio-visual objects, Part 1: Systems", final committee draft, May 1998.
[11] ISO/IEC 14496-2 "Coding of audio-visual objects, Part 2: Visual", final committee draft, May 1998.
[12] ISO/IEC 14496-3 "Coding of audio-visual objects, Part 3: Audio", final committee draft, May 1998.
[13] ISO/IEC JTC1/SC29/WG11, "MPEG-7: Requirements document ver. 11.0," N2723, March 1999
[14] S.Floyd, and K. Fall "Promoting the Use of End-to-End Congestion Control in the Internet" Proc. of IEEE/ACM Transaction on Networking, 7(4), pp.458-472, August 1999.
[15] R.Rejaie, M.Handley, and D.Estrin "RAP: An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet" Proc. IEEE Infocom'99, pp. 1337-1345, March 1999.
[16] D.Sisalem, and H.Schulzrinne "The Loss-Delay Adjustment Algorithm: A TCP-friendly Adaptation Scheme, Network and Operating System Support for Digital Audio and Video" (NOSSDAV), Cambridge, UK, 8-10, July 1998.
[17] M.Handley, J.Padhye, S.Floyd, and J.Widmer "TCP Friendly Rate Control (TFRC): Protocol Specification" Internet draft, (expires October 2002), work in progress. 2002.
[18] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson "RFC1889 RTP: A Transport Protocol for Real-Time Applications", IETF, January 1996.
[19] T. Ahmed, A. Mehaoua, and G. Buridant "Implementing MPEG-4 video on demand over IP differentiated services" Proc. IEEE Globecom'01, pp. 2489 - 2493, November 2001.
[20] J. Heinanen and, R. Guerin "RFC2698: A Two Rate Three Color Marker TRTCM" IETF (September 1999)
[21] Frank H.P. Fitzek, and Martin Reisslein "MPEG-4 and H.263 Video Traces for Network Performance Evaluation" IEEE Network, vol 5, no 6, pp 40-54, November 2001.