University of Cyprus
Department of Computer Science
Networks Research Laboratory

Adaptive Methods for the Transmission of Video Streams in Wireless Networks

Deliverable 3.2

Performance evaluation results of the proposed algorithms

Project Title	:	Adaptive Methods for the Transmission of Video Streams in Wireless Networks
Deliverable Type	:	Final WP Report (M24)

Deliverable Number	:	D3.2
Title of Deliverable	:	Performance evaluation results of the proposed algorithms.
Work Package Title	:	Simulation of Adaptation Techniques
Work Package Number	:	WP3
Internal Document Number	:	
Contractual Delivery Date	:	
Actual Delivery Date	:	
Author(s)	:	Pavlos Antoniou Marios Lestas Andreas Pitsillides Vasos Vassiliou Ioakim Sykopetritis
Email(s)	:	paul.antoniou@cs.ucy.ac.cy lestas@cs.ucy.ac.cy Andreas.Pitsillides@ucy.ac.cy vasosv@cs.ucy.ac.cy asykopetritis@gmail.com

Abstract

In the previous deliverables we analyzed and presented the design of two adaptive algorithms for the increase of the objective as well as the subjective (perceptual) quality during video streaming and we provided the corresponding simulation models for the evaluation of their performance. In addition, we proposed two novel congestion control algorithms for high speed networks. In this last deliverable of the project we will conduct several scenarios based on the models we implemented previously in order to test and evaluate their performance under various realistic or extreme network conditions.

Keywords: ADIVIS algorithm, RAF algorithm, ACP algorithm, Queue Length Based Internet Congestion Control protocol.

Table of Contents

List of Figures

List of Tables

1. Introduction

In this last deliverable of the ADAVIDEO project we will try to present and analyze the results obtained by the simulations conducted using the models we described in the previous deliverable. Our aim is to test the performance of all the proposed algorithms under various realistic or extreme network conditions.

The rest of this deliverable is organized as follows. Section 2 describes the evaluation setup and scenarios including the variable test parameters and the test sequences. Moreover we present the evaluation results focusing on the effectiveness and performance of each algorithm.

2. Simulations and Results

In the previous deliverable we analyzed and presented the requirements as well as the large variety of tools needed for the evaluation of the performance of the two proposed algorithms. In this section we will present the different evaluation topologies and scenarios as well as the results carried out after simulations. The rest of this section is organised as follows: Section 2.1 deals with topologies, scenarios and results carried out using ADIVIS algorithm. Section 2.2 presents the same issues for RAF algorithm. Section 2.3 focuses on the performance evaluation of the ACP protocol and Section 2.4 presents the results of the Queue Length Based Internet Congestion Control protocol.

2.1 ADIVIS Performance Evaluations

ADIVIS algorithm was tested in NS2 simulation environment using real video traces as input to the video streaming server. As mentioned before, this scheme deals with video streaming over the wireless Internet using a fuzzy controlled decision algorithm at video streaming server side and a feedback algorithm as well which takes into consideration both receiver's critical information and network-oriented measurements in order to evaluate the available bandwidth of the network path.

This scheme requires that the video streams are encoded in a layered manner using a scalable encoder. Layered information needs to be adapted for a number of transmission rates in order to have smooth and optimal adaptation to the available bandwidth of the network path. The techniques for reducing the transmitted information are primarily based on dropping or adding layers.

We will compare adding/dropping layers and switching among different versions of the video and we will investigate how the layered information needs to be adapted for a number of transmission rates.

2.1.1 Fuzzy Rate Controller Evaluation

We investigate the ability of the fuzzy rate controller to sense the available bandwidth of a bottleneck link in the presence of CBR and FTP background cross traffic and adapt the transmission rate of a 1Mbps CBR video stream. A bottleneck link along the end to end path was represented by a dumbbell topology in the Network Simulator (ns2) [1]. The propagation delay across the link was set to 10ms. We considered RED-enabled routers having buffer capacity of 50 packets. Also, the min_{th} and max_{th} of each queue are set to 10 and 30 packets respectively and the p_{max} to 0.1.

Moreover, the interval T between transmissions of RR packets was set to 0.3 seconds. The selection of 0.3 seconds is dictated by the desire to maintain responsiveness to changes in the network state. Further analysis of T, including sensitivity, responsiveness and signalling load, is planned for future work.

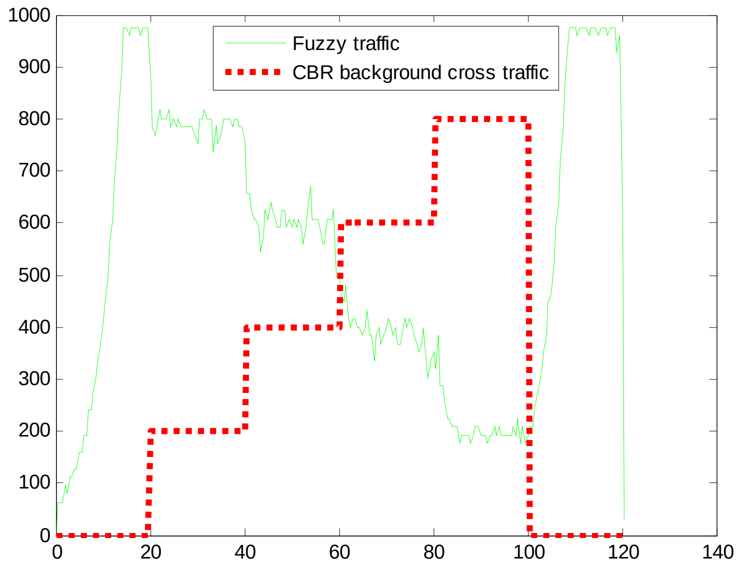


Fig. 1. Instantaneous fuzzy rate for 1Mbps bottleneck link with CBR cross traffic.

Fig. 1 depicts the instantaneous transmission rate of the CBR video stream as the CBR cross traffic rate changes over the time. We can see that the bottleneck link bandwidth is 1Mbps and the CBR cross traffic rate increases from 200Kbps to 800Kbps. As can be seen, the video transmission rate driven by the fuzzy rate controller, evolves at a slow and smooth pace in order to prevent fluctuations.

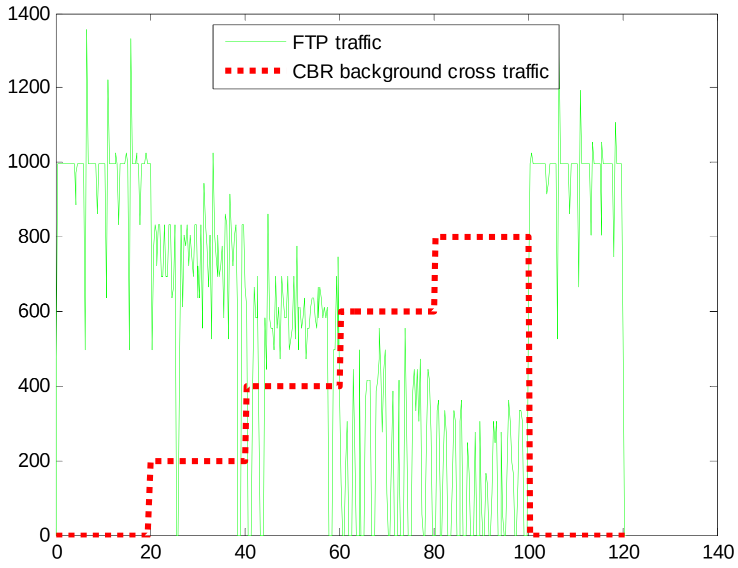


Fig. 2. Instantaneous FTP rate for 1Mbps bottleneck link with CBR cross traffic.

Fig. 2 illustrates the FTP sending rate evolution under the same circumstances of CBR cross traffic rate. It is obvious that the FTP sending rate reveals the classic saw tooth pattern of TCP behavior and its inappropriateness to support real-time video streaming. On the other hand, our fuzzy rate controller estimates accurately the available bandwidth and then matches the transmitted video bit rate to it.

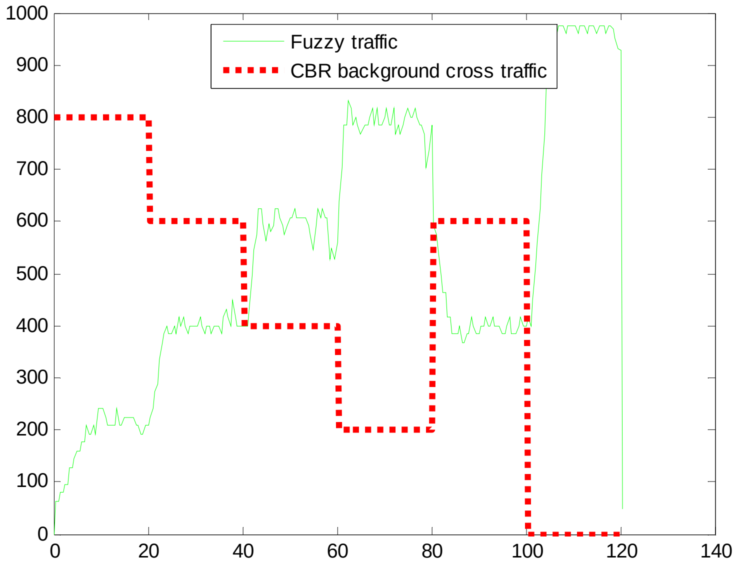


Fig. 3. Instantaneous fuzzy rate for 1Mbps bottleneck link with CBR cross traffic (different scenario).

Similarly, Fig. 3 illustrates the transmission rate of the CBR video stream as the CBR cross traffic rate decreases from 800Kbps to 200Kbps with a slight increase to 400Kbps. The results demonstrate that the fuzzy rate controller is able to detect the available bandwidth of the bottleneck link and adapt to it. Furthermore, the results reveal that, as the available bandwidth increases, the fuzzy-controlled stream acquires more of that bandwidth in a non-aggressive way, while it adapts fast to bandwidth reduction. The same results can be seen in Fig. 4 where the bottleneck link bandwidth is 500Kbps.

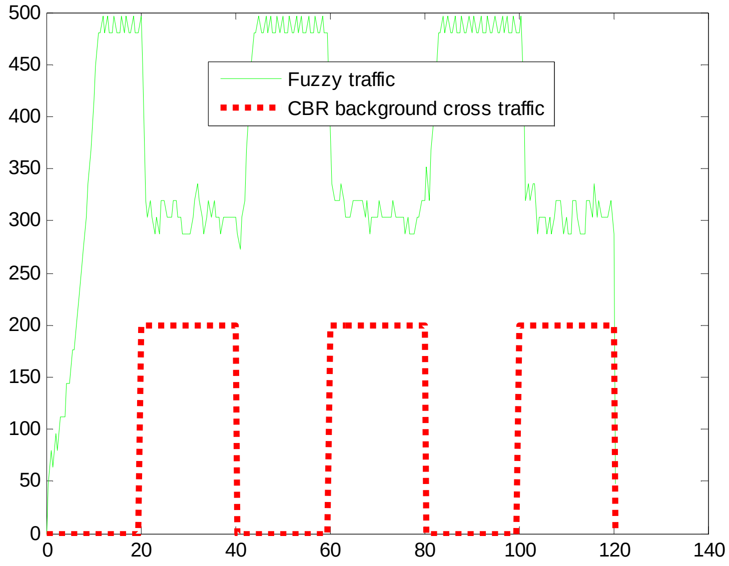


Fig. 4. Instantaneous fuzzy rate for 500Kbps bottleneck link with CBR cross traffic.

Fig. 5 illustrates the transmission rate of the CBR video stream in the presence of FTP background cross traffic. Although the FTP cross traffic is more bursty than CBR cross traffic shown in Fig. 1 and Fig. 3, the fuzzy controller senses the available capacity of the bottleneck link and finely adapts the video rate to it. The fuzzy-controlled flow appears to be TCP-friendly against an FTP flow, as it does not aggressively consume the available bandwidth but further analysis is planned for future work.

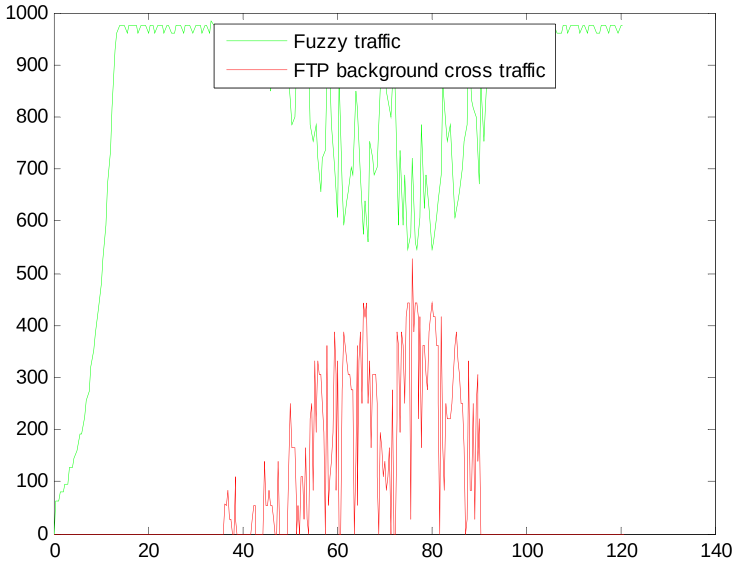


Fig. 5. Instantaneous fuzzy rate for 1Mbps bottleneck link with FTP cross traffic.

2.1.2 Topology

As we can see Fig. 6 illustrates the topology we used in the performance evaluation. The topology consists of two routers directly connected with a link having variable characteristics. A video streaming server is attached to the first router. Mobile wireless clients are connected to the second router over wireless links. In order to make our scenarios more realistic we added background traffic initiated by the FTP server.

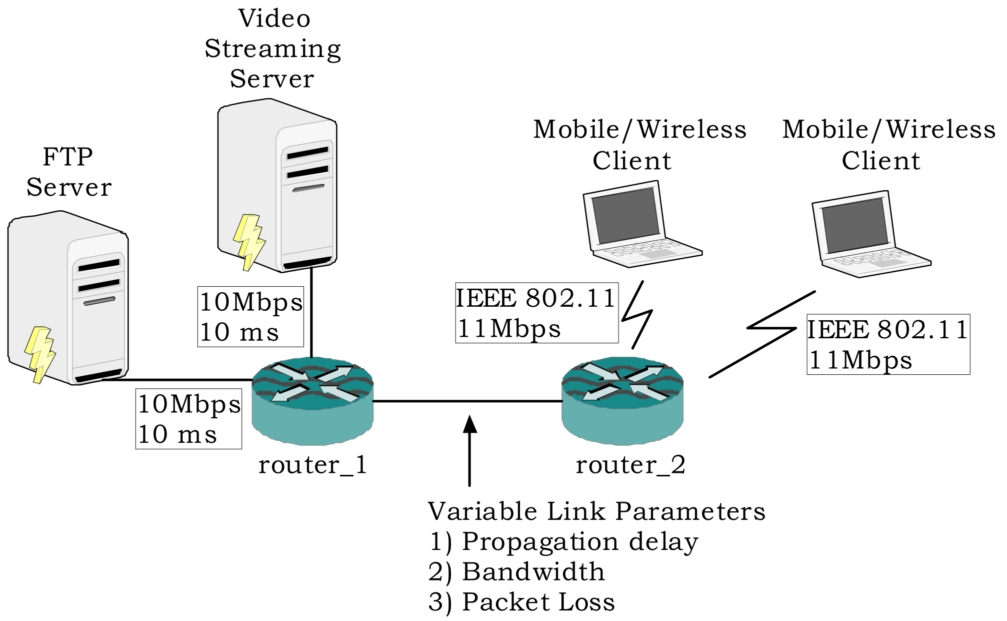


Fig. 6. Evaluation Topology for ADIVIS.

2.1.3 Evaluation parameters

In order to simulate the video traffic patterns we used a well known real video sequence named Foreman. Foreman test sequence was chosen because of its characteristics. It is a stream with a fair amount of movement and change of background. The characteristics of this sequence are shown in Table 1. The sample sequence was encoded in MPEG4 format with a free software tool called FFMPEG encoder [6]. The sequence has temporal resolution 30 frames per second, and GoP (Group of Pictures) pattern IBBPBBPBBPBB.

Trace	Resolution	Total Frames	# I frames	# P frames	# B frames
Foreman.yuv	176x144	400	34	100	266

Table 1. Video Characteristics.

We encoded this sequence in 8 different bit rates listed in the next table. Each encoded video stream has different bit rate and corresponds to a separate layer. The video stream bit rate1 varies from 64Kbps to 768Kbps. All these encoded sequences (layers) were attached on the video streaming server.

Layer	Bitrate (Kbps)
0	64
1	96
2	128
3	256
4	192
5	384
6	512
7	768

Table 2. Layers and bit rates used in simulations of ADIVIS.

The different parameter values used to characterize the variable link between the two routers are presented in Table 3.

Link Bandwidth (Kbps)	Propagation Delay (ms)	Packet Loss
64	10	0%
128	100	5%
256	200	
384	400	
512	800	
768		
1000		

Table 3. Variable Link Parameters.

The choice of the parameters used in the video quality evaluations was based on the typical characteristics of mobile and wireless networks.

We set the maximum capacity of each buffer to 50 packets, the min_{th} and max_{th} of the queue as 10 and 30 packets respectively and the p_{max} to 0.1. The interval T between transmissions of RR packets was set to 0.3 seconds

2.1.4 Simulations

In this section we analyze the results obtained from the above scenario evaluations. We present scenarios involving one and two wireless users. All other parameters are variable as shown in Table 3. Video quality is measured by taking the average of the Peak Signal-to-Noise Ratio (PSNR) over all the decoded frames.

2.1.4.1 Scenarios involving one mobile/wireless client

The effect of propagation delay and link bandwidth on the PSNR in the absence of background traffic is presented in Fig. 7 and Fig. 8.

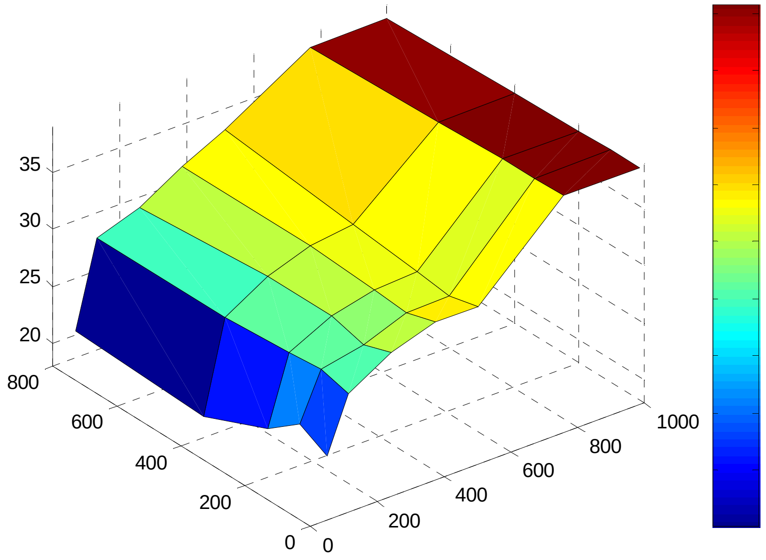


Fig. 7. Mean PSNR vs. Link BW and Prop. Delay, No FTP, Packet Loss = 0%.

.

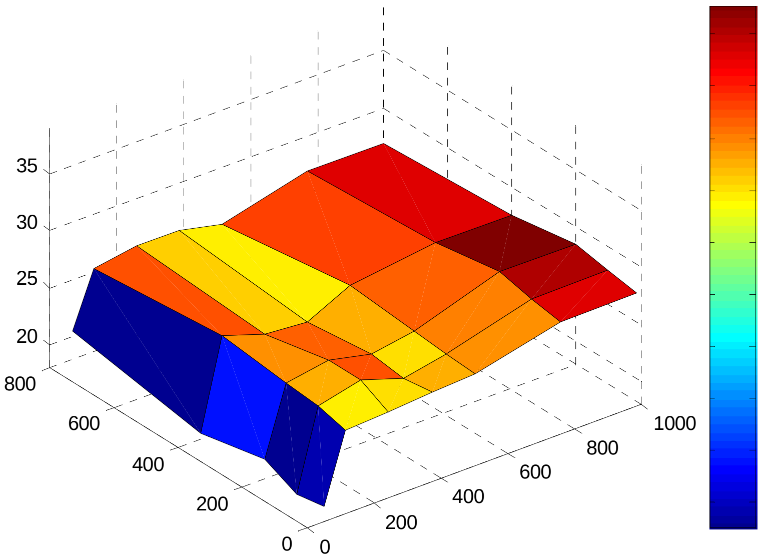


Fig. 8. Mean PSNR vs. Link BW and Prop. Delay, No FTP, Packet Loss = 5%.

The results obtained by scenarios where the packet loss is 0% (Fig. 7) reveal that the PSNR values are increasing at a steady pace (up to 38dB) as the link bandwidth increases.

PSNR values are decreased (less than 25dB) in scenarios where the link bandwidth is equal to the bit rate of the lowest layer (64Kbps), since there is a strong possibility of packet loss. Fig. 8 presents the objective quality evaluations obtained by scenarios involving packet loss of 5%. Obviously the values of PSNR have been significantly decreased compared to those of Fig. 7. This is because the decision algorithm recognises the high packet drop rates and strives to maintain an acceptable level of video quality, whilst satisfying the worsening network state, by sending fewer layers, resulting in lower PSNR values. The PSNR metric partially ignores the effect of the propagation delay, but as it can be seen, the delay can indirectly influence the objective quality of a video stream. Actually, the larger the delay the larger the interval between reception of two successive RR packets. Under these circumstances, the system will experience delayed decision-making that will influence the quality of the video stream. As shown in both figures, PSNR values are slightly increased for low delay values especially regarding scenarios involving high bandwidth links. This is because the content adaptation to network parameters evolves at a faster pace. All in all, delay does not influence the values of PSNR the way the link bandwidth does.

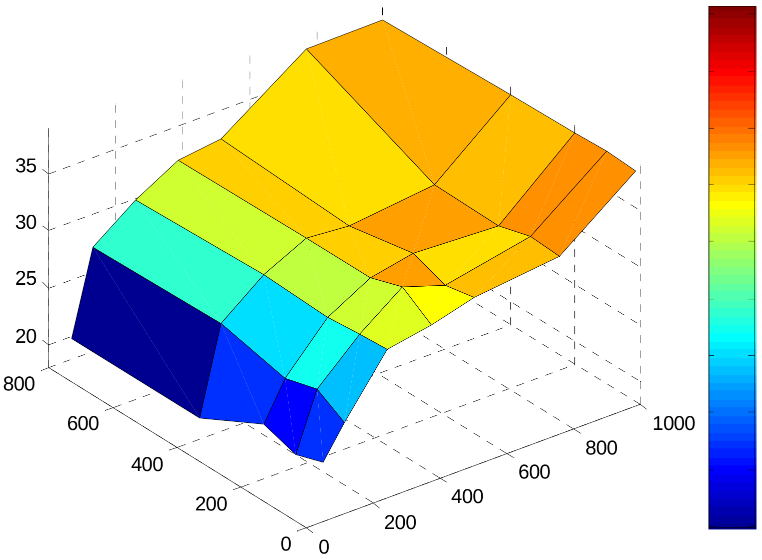


Fig. 9. Mean PSNR vs. Link BW and Prop. Delay, FTP, Packet Loss = 0\%.

Fig. 9 shows PSNR values for scenarios involving background FTP traffic while the packet loss is 0%. We observe a slight decrease in PSNR values for scenarios having link bandwidth less or equal to 256Kbps due to the excessive FTP traffic load. As the link bandwidth increases (more than 256Kbps), the quality of a video stream is not severely affected by the FTP traffic since the decision algorithm adjusts the number of layers sent according to the network conditions. The PSNR values are slightly fluctuating due to the saw-tooth behaviour of the FTP sending rate evolution, which forces the video streaming server to add/drop layers accordingly. We perceive a lower objective quality for low delay values, because the FTP sending rate evolves at a faster and aggressive pace compared with scenarios with larger delay resulting in higher packet drop rates.

2.1.4.2 Scenarios involving two mobile/wireless client

In this section we present a comparison between the results obtained previously and those concerning scenarios involving two mobile users. Fig. 10 and Fig. 9Fig. 11 show the PSNR values for different values of propagation delay.

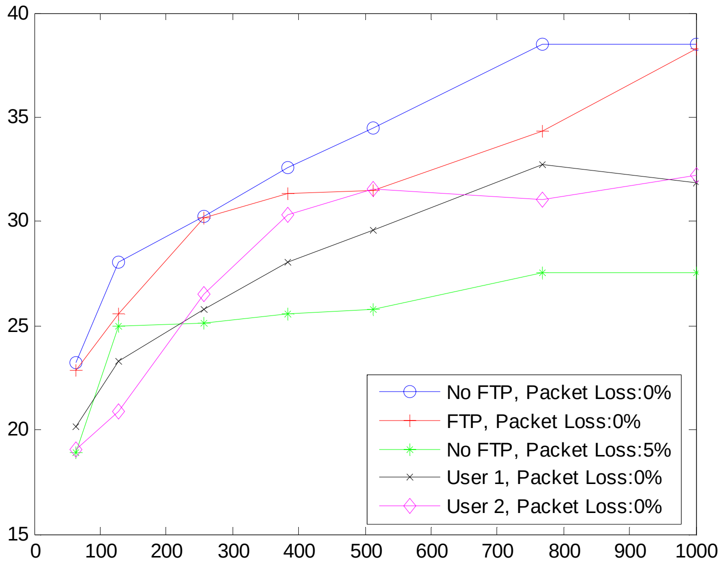


Fig. 10. Mean PSNR vs. Link BW, Prop. Delay = 10ms.

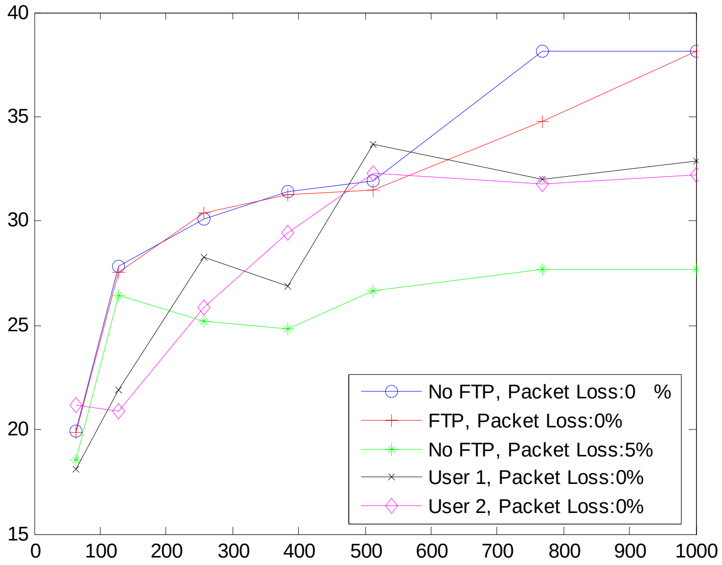


Fig. 11. Mean PSNR vs. Link BW, Prop. Delay = 400ms.

Both figures reveal that in case we have two users (moving randomly), our algorithm provides fairness, because no one of the two users takes advantage over the other, as both users perceive almost the same quality in terms of PSNR. Fig. 10 shows that the PSNR values in case of one mobile user and loss of 0% outperform all the others. Similarly, a mobile user in the presence of FTP traffic exhibits higher PSNR values compared to the scenarios involving 2, because FTP sending rate is actually lower than the cumulative video sending rate. On the other hand, in Fig. 11, scenarios involving one user and loss of 0%, exhibit lower PSNR values than before, because the adaptation evolves at a slower pace. The objective quality under FTP traffic ranges at slightly higher levels than before. This is because FTP sending rate evolves at a slower pace, which means that the influence on the adaptive flow is lessened.

2.2 RAF Performance Evaluations

In order to test our second algorithm using a layered transmission scheme we conducted some simulations in NS2 environment.

2.2.1 Topology

The next diagram illustrates the topology we used in our simulations. In order to make our scenarios more realistic we add some kind of background traffic (modelled as CBR traffic patterns). In this way we will try to make a bottleneck link (between the two routers) and examine the behaviour of our algorithm under extreme conditions of high volume traffic. The video streaming server is attached on the first router using a link of 10Mbps and 5ms propagation delay. The same applies for the CBR server. The link between the two routers has a variable bandwidth ranging from 56Kbps up to 1.6Mbps.

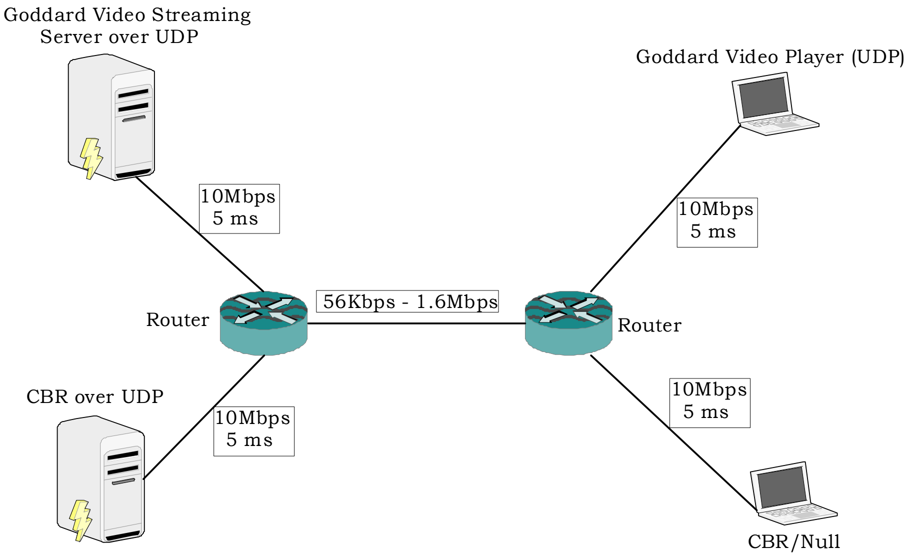


Fig. 12. Evaluation topology for RAF.

2.2.2 Evaluation parameters

In this section we will analyze and present our results regarding RAF algorithm. First of all we have to mention that we used a five layer scheme as shown in the next table. Goddard server will be able to support these five layers with the corresponding bit rates.

Parameter	Layer	Bitrate (bps)
bitrate_0_	0	56000
bitrate_1_	1	128000
bitrate_2_	2	256000
bitrate_3_	3	512000
bitrate_4_	4	768000

Table 4. Layers and bit rates used in simulations of RAF.

We will consider five different scenarios and their results will be treated separately. Initial settings are based on user requirements such as requested video quality, frame width and frames per second. The initial values used in our simulations are shown in Table 5.

Parameter	Description
fps_ = 15	*User requests for video stream of 15 fps.*
aspect_ratio_ = 1.33333333	*Requested aspect ratio is 4:3.*
frm_width_ = 480	*The maximum frame width should not exceed 480 pixels. Based on frame width and aspect ratio the algorithm will calculate the frame height.*
bpp_quality_ = 0.2	*Good frame quality.*

Table 5. Initial settings used in simulations of RAF.

The parameters involved in our scenarios can take the values shown in the following table.

Parameter	Possible Values
fps	10 15 20 25 30
aspect ratio	1.33 1.77
frame width	Unconstraint
frame height	
bpp	0.100 – 0.250
link bitrate	56 kbps 128 kbps 256 kbps 512 kbps 768 kbps
available bandwidth	56 kbps – 1.6 Mbps

Table 6. Values of parameters in RAF scenarios.

2.2.3 Simulations

The next table presents the elements and the description of each element shown in Table 8, Table 9, Table 10, Table 11, Table 12.

Element	Description
Time	*Time of the adaptation.*
Calculated bitrate	*Calculated video bit rate based on network constraints.*
Max available bandwidth	*Maximum possible transient bandwidth. It can be evaluated using feedback information.*
Max layer bitrate	*Bit rate of the transmitted layer.*
Frame width	*Frame width after the adaptation.*
Frame height	*Frame height after the adaptation.*
Bpp	*Quality of video frames after the adaptation.*
Fps	*Frame rate.*
Loss Rate	*Calculated percentage of lost packets.*

Table 7. Elements and description.

In this point we would like to keep in mind that in case a user requires specific frame dimensions then our algorithm will strive to satisfy user needs by decreasing the quality and the frame rate of the video stream (bpp and fps). But the quality will be decreased down to a lower bound given by Table 5 in D2.2. Further decrease of the quality leads to inadequate user perceived quality thus the algorithm will reduce frame dimensions.

Simulation A:
Using only the first layer: 56kbps

Time	Calculated bitrate	Max available bandwidth	Max Layer bitrate supported	Frame width	Frame height	Bpp	Fps	Loss rate	MOS (χωρίς το Loss)	MOS (με το Loss)
0.113912	54756	2048000	55920	156	117	0.100	30	0.000000	42.00	42.00
0.123912	54756	2048000	55920	156	117	0.100	30	0.000000	42.00	42.00
1.062818	54756	483365	55920	156	117	0.100	30	0.000000	42.00	42.00
2.062818	54756	687524	55920	156	117	0.100	30	0.000000	42.00	42.00
3.062818	55296	427638	55920	128	96	0.225	20	0.180328	63.00	43.75
4.062818	55296	103115	55920	128	96	0.225	20	0.214286	63.00	43.75
5.062818	55296	157852	55920	128	96	0.225	20	0.181818	63.00	43.75
6.062818	55296	466414	55920	128	96	0.225	20	0.230769	63.00	43.75
7.062818	54756	583089	55920	104	78	0.225	30	0.444444	94.50	47.25
8.062818	54756	75898	55920	104	78	0.225	30	0.404762	94.50	47.25
9.062818	52920	105247	55920	112	84	0.225	25	0.350877	78.75	43.75
12.062818	55296	305500	55920	128	96	0.225	20	0.200000	63.00	43.75
13.062818	55296	446461	55920	128	96	0.225	20	0.200000	63.00	43.75
18.062818	55296	430702	55920	128	96	0.225	20	0.200000	63.00	43.75
22.062818	54756	553428	55920	156	117	0.100	30	0.000000	42.00	42.00
25.062818	54756	610739	55920	156	117	0.100	30	0.000000	42.00	42.00

Table 8. Results of simulations conducted in the presence of only one layer of 56Kbps.

Simulation B:
Using 2 layers: 56kbps and 128kbps

Time	Calculated bitrate	Max available bandwidth	Max Layer bitrate supported	Frame width	Frame height	Bpp	Fps	Loss rate	MOS (χωρίς το Loss)	MOS (με το Loss)
0.114167	125316	2048000	127920	236	177	0.100	30	0.000000	42.00	42.00
0.124167	125316	2048000	127920	236	177	0.100	30	0.000000	42.00	42.00
1.063293	125316	483365	127920	236	177	0.100	30	0.000000	42.00	42.00
2.063293	125316	687524	127920	236	177	0.100	30	0.000000	42.00	42.00
3.063293	124416	427638	127920	192	144	0.225	20	0.180328	63.00	43.75
4.063293	55296	103115	55920	128	96	0.225	20	0.214286	63.00	43.75
5.063293	124416	157852	127920	192	144	0.225	20	0.193878	63.00	43.75
6.063293	125316	466414	127920	236	177	0.100	30	0.076923	42.00	42.00
7.063293	124416	583089	127920	192	144	0.225	20	0.142857	63.00	43.75
8.063293	55296	75898	55920	128	96	0.225	20	0.159091	63.00	43.75
9.063293	55296	105247	55920	128	96	0.225	20	0.118644	63.00	43.75
12.063293	123201	305500	127920	156	117	0.225	30	0.600000	94.50	47.25
13.063293	123201	446461	127920	156	117	0.225	30	0.600000	94.50	47.25
18.063293	123201	430702	127920	156	117	0.225	30	0.600000	94.50	47.25
22.063293	123201	553428	127920	156	117	0.225	30	0.600000	94.50	47.25
25.063293	123201	610739	127920	156	117	0.225	30	0.600000	94.50	47.25

Table 9. Results of simulations conducted in the presence of layers 56Kbps and 128Kbps.

Simulation C:

Using 3 layers: 56 kbps, 128 kbps and 256 kbps.

Time	Calculated bitrate	Max available bandwidth	Max Layer bitrate supported	Frame width	Frame height	Bpp	Fps	Loss rate	MOS (χωρίς το Loss)	MOS (με το Loss)
0.114622	254016	2048000	255960	336	252	0.100	30	0.000000	42.00	42.00
0.124622	254016	2048000	255960	336	252	0.100	30	0.000000	42.00	42.00
1.063768	254016	483365	255960	336	252	0.100	30	0.000000	42.00	42.00
2.063768	254016	687524	255960	336	252	0.100	30	0.000000	42.00	42.00
3.063768	249696	427638	255960	272	204	0.225	20	0.180328	63.00	43.75
4.063768	55296	103115	55920	128	96	0.225	20	0.228916	63.00	43.75
5.063768	124416	157852	127920	192	144	0.225	20	0.232323	63.00	43.75
6.063768	254016	466414	255960	336	252	0.100	30	0.000000	42.00	42.00
7.063768	254016	583089	255960	336	252	0.100	30	0.071429	42.00	42.00
8.063768	55296	75898	55920	128	96	0.225	20	0.142857	63.00	43.75
9.063768	55296	105247	55920	128	96	0.225	20	0.105263	63.00	43.75
12.063768	254016	305500	255960	336	252	0.100	30	0.000000	42.00	42.00
13.063768	254016	446461	255960	336	252	0.100	30	0.000000	42.00	42.00
18.063768	254016	430702	255960	336	252	0.100	30	0.000000	42.00	42.00
22.063768	254016	553428	255960	336	252	0.100	30	0.000000	42.00	42.00
25.063768	254016	610739	255960	336	252	0.100	30	0.000000	42.00	42.00

Table 10. Results of simulations conducted in the presence of layers 56Kbps, 128Kbps and 256Kbps.

Simulation D:
Using 4 layers: 56 kbps, 128 kbps, 256 kbps and 512 kbps

Time	Calculated bitrate	Max available bandwidth	Max Layer bitrate supported	Frame width	Frame height	Bpp	Fps	Loss rate	MOS (χωρίς το Loss)	MOS (με το Loss)
0.114878	509796	2048000	511920	476	357	0.100	30	0.000000	42.00	42.00
0.124878	509796	2048000	511920	476	357	0.100	30	0.000000	42.00	42.00
1.064243	254016	483365	255960	336	252	0.100	30	0.000000	42.00	42.00
2.064243	509796	687524	511920	476	357	0.100	30	0.000000	42.00	42.00
3.064243	249696	427638	255960	272	204	0.225	20	0.172414	63.00	43.75
4.064243	55296	103115	55920	128	96	0.225	20	0.250000	63.00	43.75
5.064243	124416	157852	127920	192	144	0.225	20	0.224490	63.00	43.75
6.064243	254016	466414	255960	336	252	0.100	30	0.071429	42.00	42.00
7.064243	509796	583089	511920	476	357	0.100	30	0.068966	42.00	42.00
8.064243	54756	75898	55920	156	117	0.100	30	0.069767	42.00	42.00
9.064243	54756	105247	55920	156	117	0.100	30	0.084746	42.00	42.00
12.064243	254016	305500	255960	224	168	0.225	30	0.600000	94.50	47.25
13.064243	254016	446461	255960	224	168	0.225	30	0.600000	94.50	47.25
18.064243	254016	430702	255960	224	168	0.225	30	0.600000	94.50	47.25
22.064243	505521	553428	511920	316	237	0.225	30	0.600000	94.50	47.25
25.064243	505521	610739	511920	316	237	0.225	30	0.600000	94.50	47.25

Table 11. Results of simulations conducted in the presence of layers 56Kbps, 128Kbps, 256Kbps and 512Kbps.

Simulation E:

Using 5 layers: 56 kbps, 128 kbps, 256 kbps, 512 kbps and 768 kbps

Time	Calculated bitrate	Max available bandwidth	Max Layer bitrate supported	Fram e width	Frame height	Bpp	Fps	Loss rate	MOS (χωρίς το Loss)	MOS (με το Loss)
0.115133	518400	2048000	768000	480	360	0.200	15	0.000000	42.00	42.00
0.125133	518400	2048000	768000	480	360	0.200	15	0.000000	42.00	42.00
1.064718	254016	483365	255960	336	252	0.100	30	0.000000	42.00	42.00
2.064718	509796	687524	511920	476	357	0.100	30	0.000000	42.00	42.00
3.064718	249696	427638	255960	272	204	0.225	20	0.196721	63.00	43.75
4.064718	55296	103115	55920	128	96	0.225	20	0.238095	63.00	43.75
5.064718	124416	157852	127920	192	144	0.225	20	0.224490	63.00	43.75
6.064718	254016	466414	255960	336	252	0.100	30	0.071429	42.00	42.00
7.064718	509796	583089	511920	476	357	0.100	30	0.068966	42.00	42.00
8.064718	55296	75898	55920	128	96	0.225	20	0.113636	63.00	43.75
9.064718	55296	105247	55920	128	96	0.225	20	0.133333	63.00	43.75
12.064718	254016	305500	255960	336	252	0.100	30	0.000000	42.00	42.00
13.064718	254016	446461	255960	336	252	0.100	30	0.000000	42.00	42.00
18.064718	254016	430702	255960	336	252	0.100	30	0.000000	42.00	42.00
22.064718	509796	553428	511920	476	357	0.100	30	0.000000	42.00	42.00
25.064718	509796	610739	511920	476	357	0.100	30	0.000000	42.00	42.00

Table 12. Results of simulations conducted in the presence of layers 56Kbps, 128Kbps, 256Kbps, 512Kbps and 768Kbps.

According to the results shown in the previous tables we can see that the algorithm is able to adapt to the new conditions of the core networks. In any case the maximum bit rate of the highest layer poses a constraint to the system.

A remarkable observation can be made in the cases we have a relatively large number of packets are lost. Results show that the algorithm strives to preserve a satisfactory quality of service by increasing the frame rate while sacrificing the dimensions of the frame the same time. In this way the user may not realise any possible loss of packets during the transmission.

The last two columns of the tables reveal that the adaptation of the algorithm provides satisfactory user perceived quality which is independent of exogenous fluctuating factors. The last but one column presents the values of the mean opinion score (MOS) as evaluated by the algorithm using Table 5 in D2.2 without taking into consideration the number of packets lost in the core network. The last column reveals the real values of MOS taking packets lost into account.

Generally, the quality of the transmitted video streams remains above the predefined lower limit. Of course the frame dimensions are fluctuating and may have smaller values than those requested but the most important thing is to keep quality in satisfactory levels. The proposed algorithm is able to ensure satisfactory user perceived quality using the larger possible frame dimensions based on the available bandwidth and the maximum bit rate of the highest supported layer.

2.2.3.1 Bit rate fluctuations

The next diagram depicts bit rate fluctuations when more layers are used for video transmission. If we see this diagram and the results obtained from Simulation E we can mention that the video stream bit rate reaches up to 518.4Kbps while the maximum theoretical bit rate – according to the highest layer – could be 768Kbps. This is not algorithm's fault but the video stream bit rate reaches this value upon satisfaction of the user's requirements.

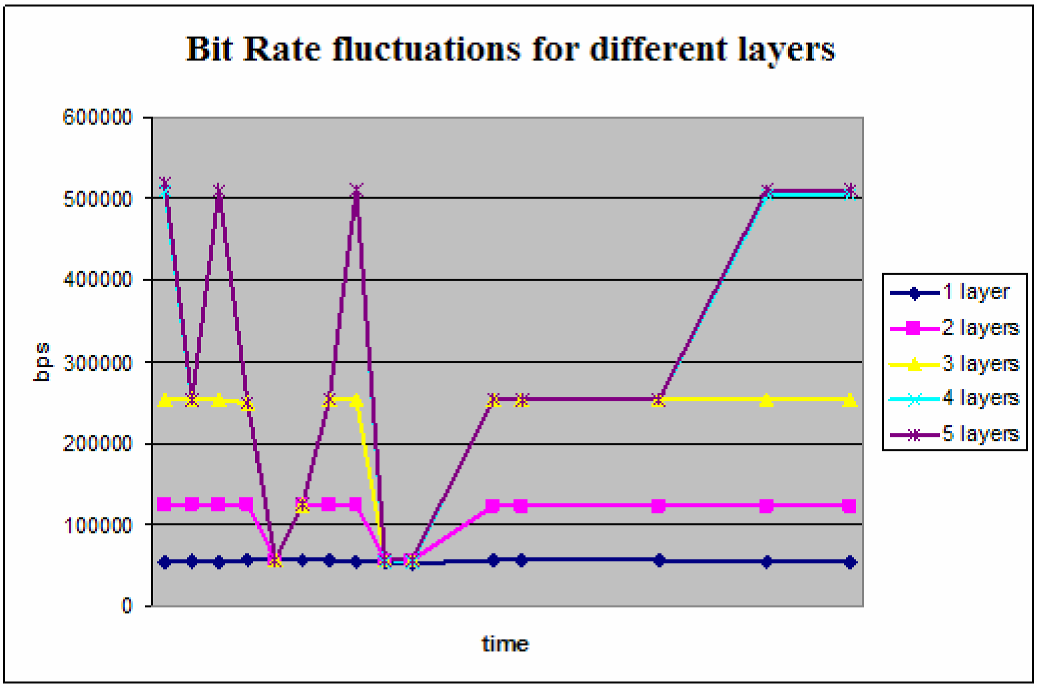


Fig. 13. Bit rate fluctuations for different layers.

2.2.3.2 Frame quality evaluations

In this section we present some frames taken from video streams that were transmitted by the enhanced Goddard Streaming Media System that incorporates our algorithm.

In the following figure we compare three discrete cases using different number of layers. In the first scenario we have only one layer of 56Kbps, the second scenario deals with two layers namely 56Kbps and 128Kbps whereas the third one deals with three layers 56Kbps, 128Kbps and 256Kbps.

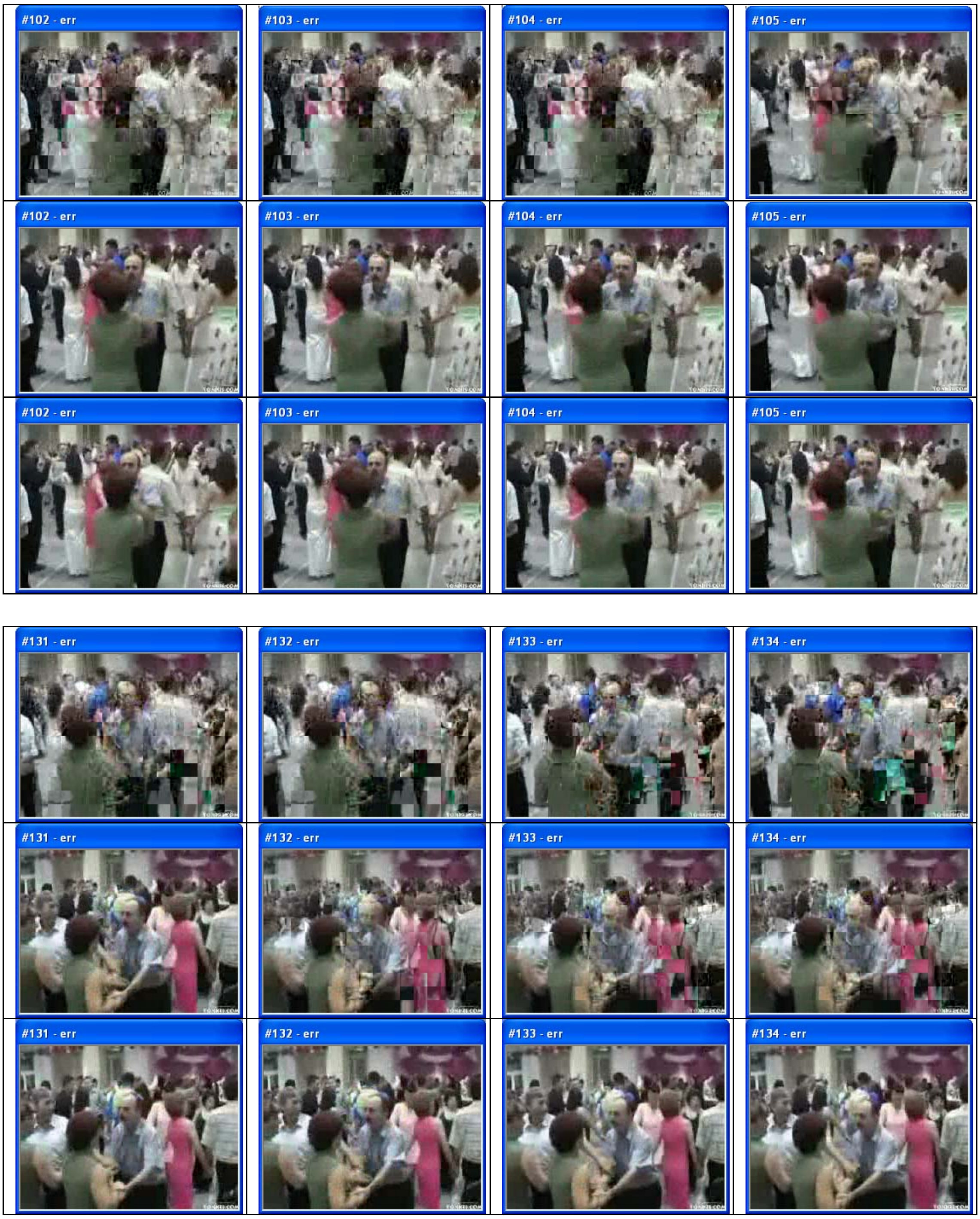


Fig.14. Frames taken by scenarios involving different number of layers.

2.2.3.3 Frame size evaluations

The next diagram shows the difference in frame sizes between scenarios involving encoding rates of 512Kbps and 768Kbps.

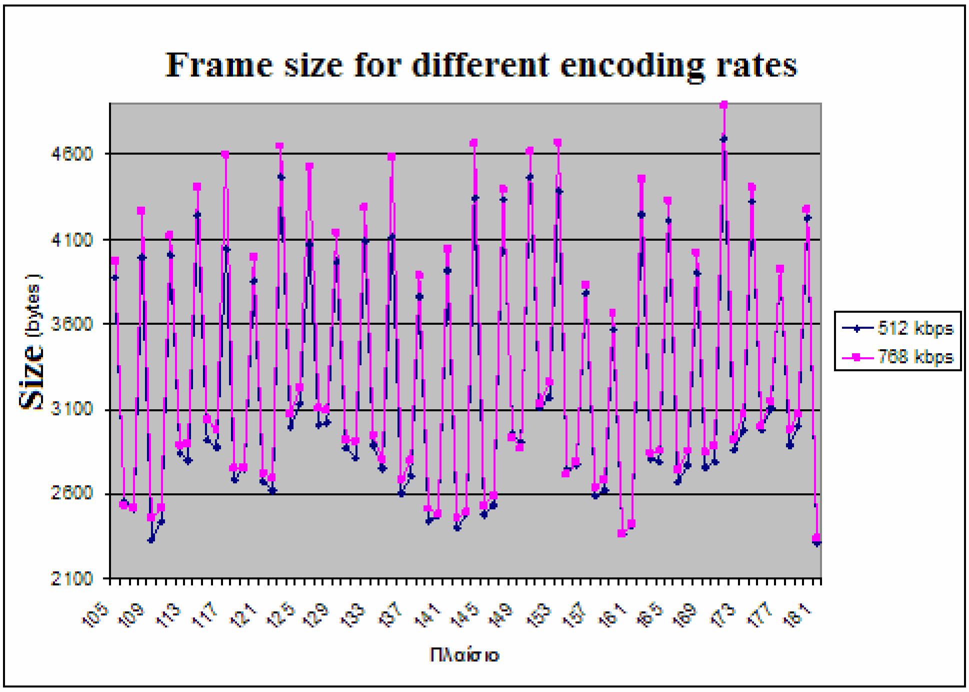


Fig. 15. Frame size for different encoding rates.

As can be seen frame sizes are almost the same regardless the different encoding rate.

2.2.3.4 MOS evaluations

In order to evaluate the user perceived quality of the received video stream under high volume of background traffic in the core network we conducted some scenarios shown below. In every single scenario we altered the bit rate regarding the background CBR traffic and we monitored its influence to the quality of the video stream. As expected as the background traffic increases the available bandwidth decreases and our algorithm forces to adapt the video stream bit rate accordingly. The following diagram depicts the results obtained from these scenarios.

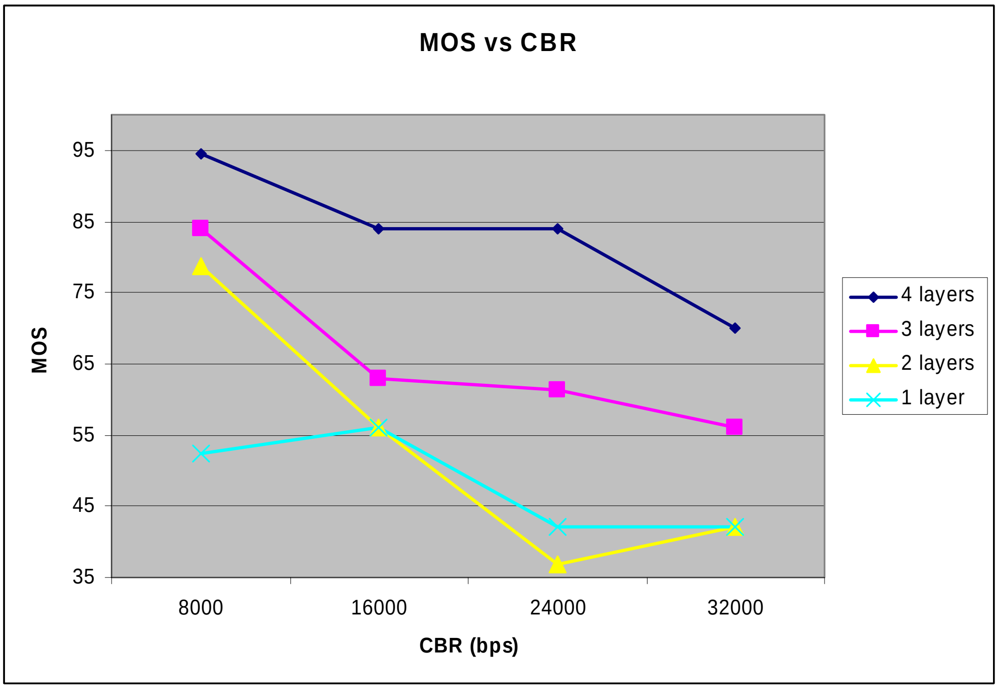


Fig. 16. MOS variance in the presence of background traffic.

As can be seen regardless the number of layers used in simulations, MOS is getting smaller as the background traffic increases. In particular when 4 layers are used, MOS gets higher values than when smaller number of layers is used something that is absolutely expected.

We can also remark that the algorithm is adaptive in terms of video stream bit rate as we can see that it strives to stabilize the values of MOS in the presence of higher background traffic even though the subjective quality (in terms of MOS) tends to be reduced gradually.

2.2.3.5 Calculated bit rate vs. layer bit rate

In this section we present some results that show the relationship between the maximum bit rate of the supported layers and the calculated bit rate. We also consider the maximum available bandwidth in order to understand the behaviour of the curves. These results reveal the adaptive ability of our algorithm based on the supported layers and the maximum available bandwidth as well as the variable conditions within the core network.

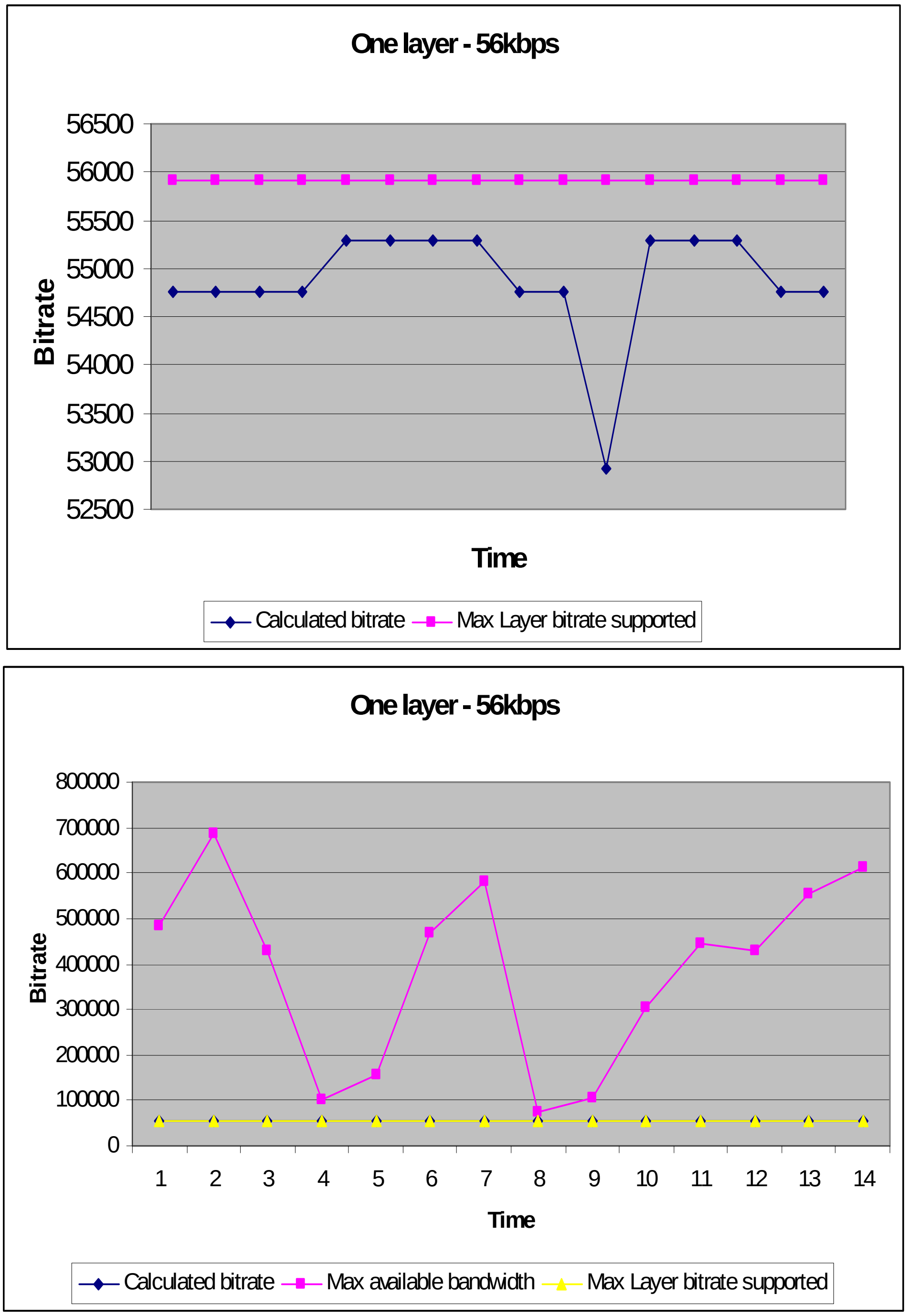


Fig. 17. Calculated bit rate vs. max layer bit rate vs. max available bandwidth (1 layer: 56Kbps).

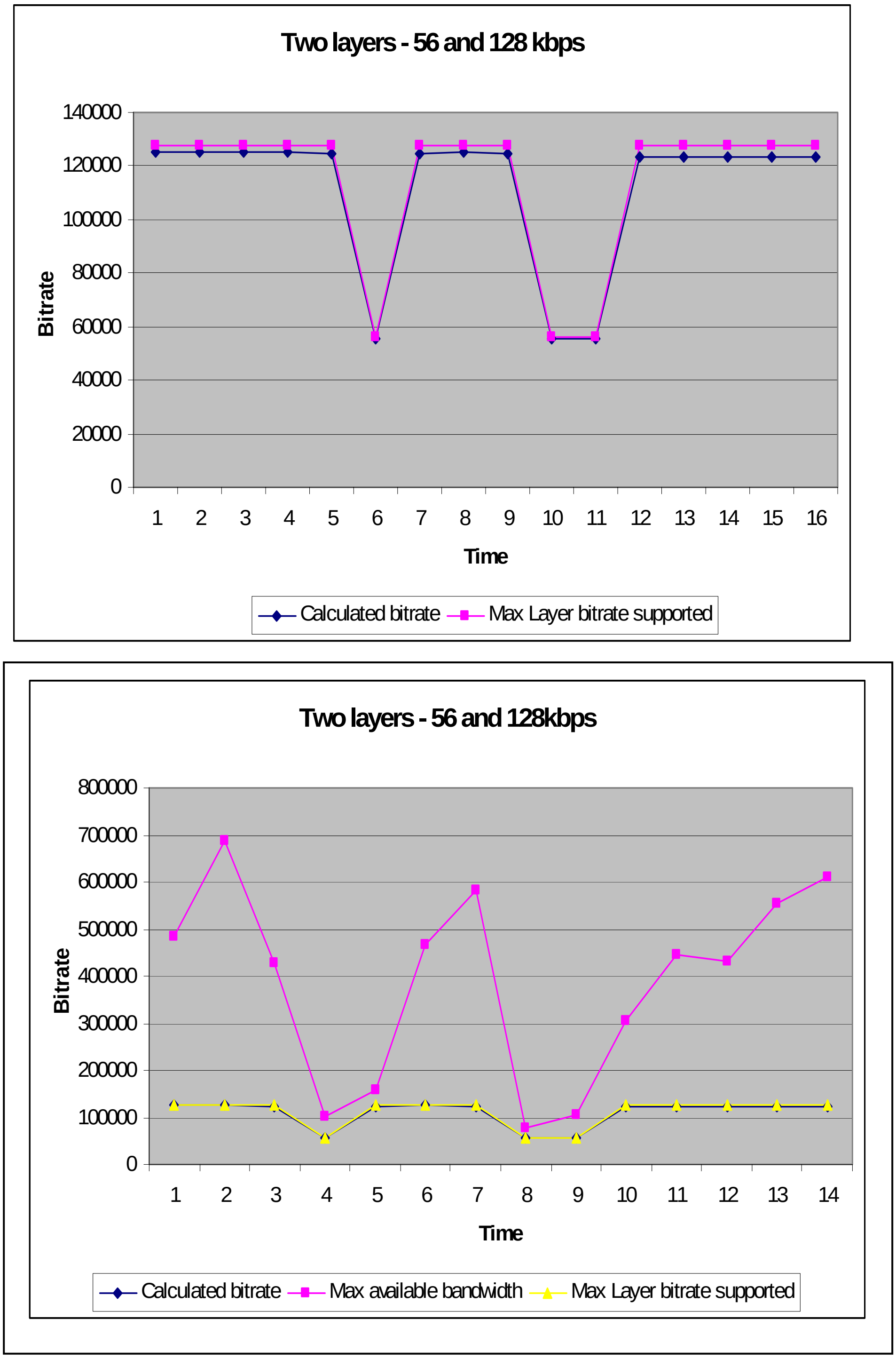


Fig. 18. Calculated bit rate vs. max layer bit rate vs. max available bandwidth (2 layers: 56Kbps and 128Kbps).

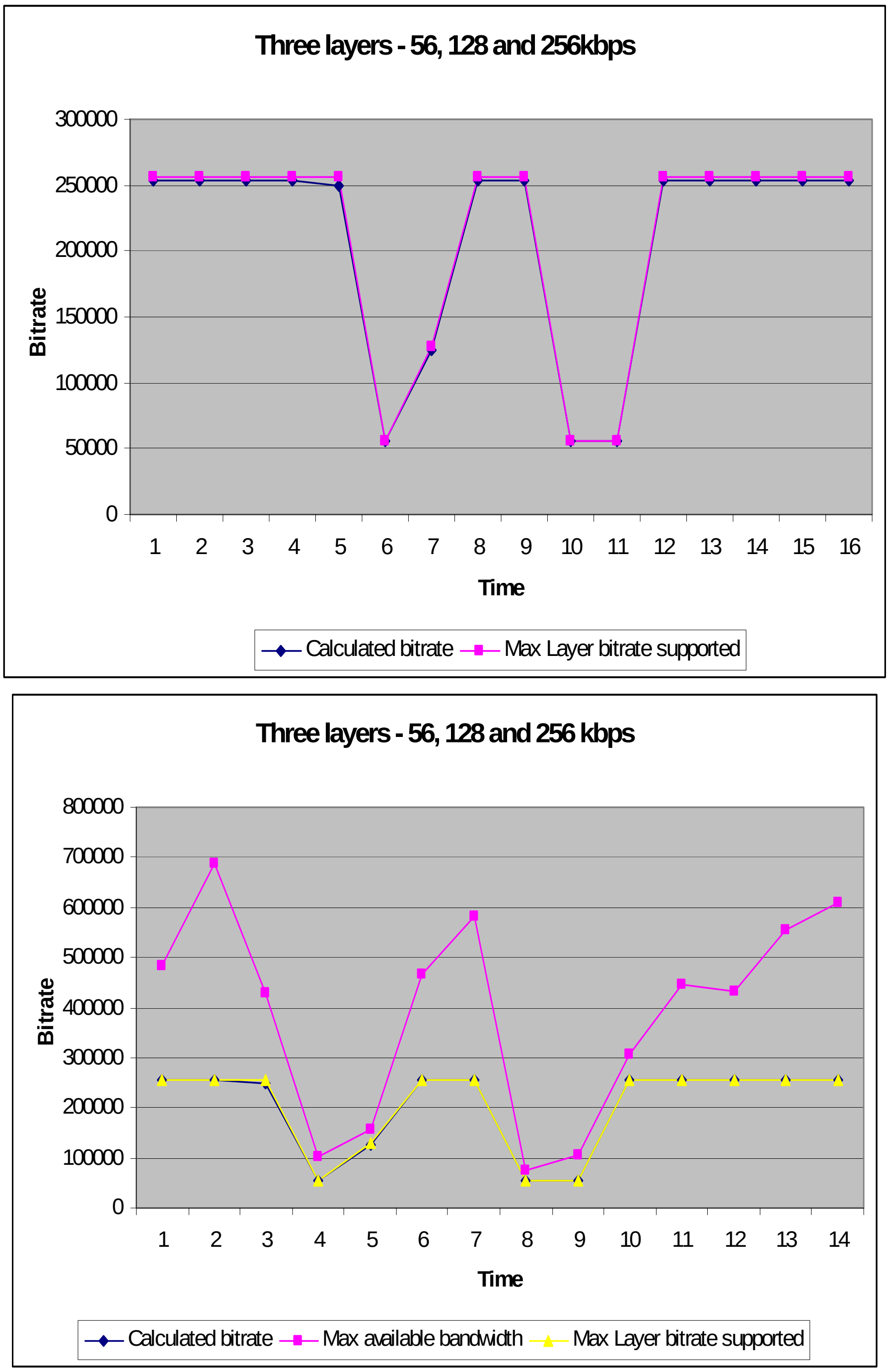
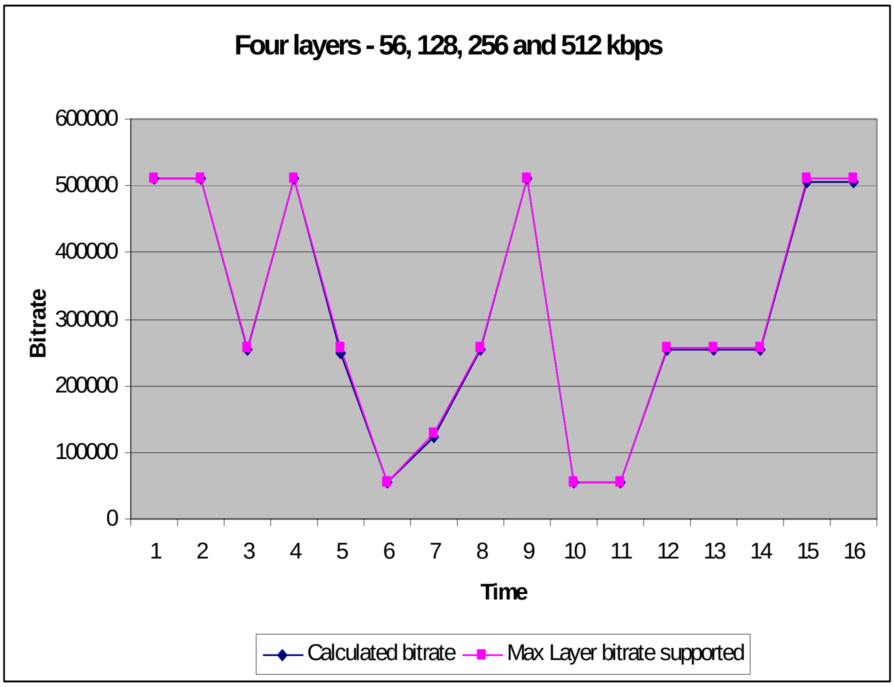


Fig. 19. Calculated bit rate vs. max layer bit rate vs. max available bandwidth (3 layers: 56Kbps, 128Kbps and 256Kbps).



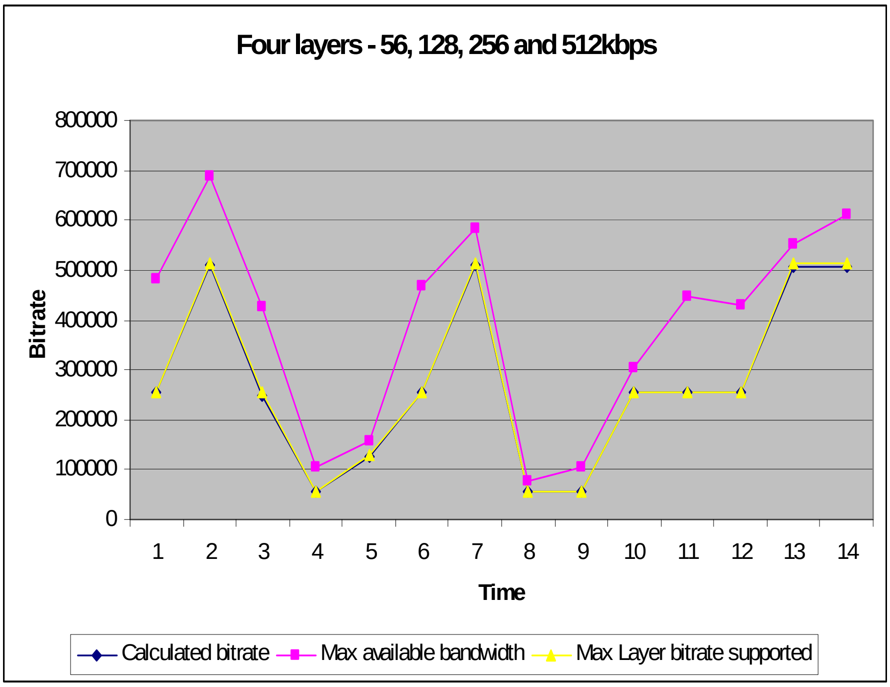


Fig. 20. Calculated bit rate vs. max layer bit rate vs. max available bandwidth (4 layers: 56Kbps, 128Kbps, 256Kbps and 512Kbps).

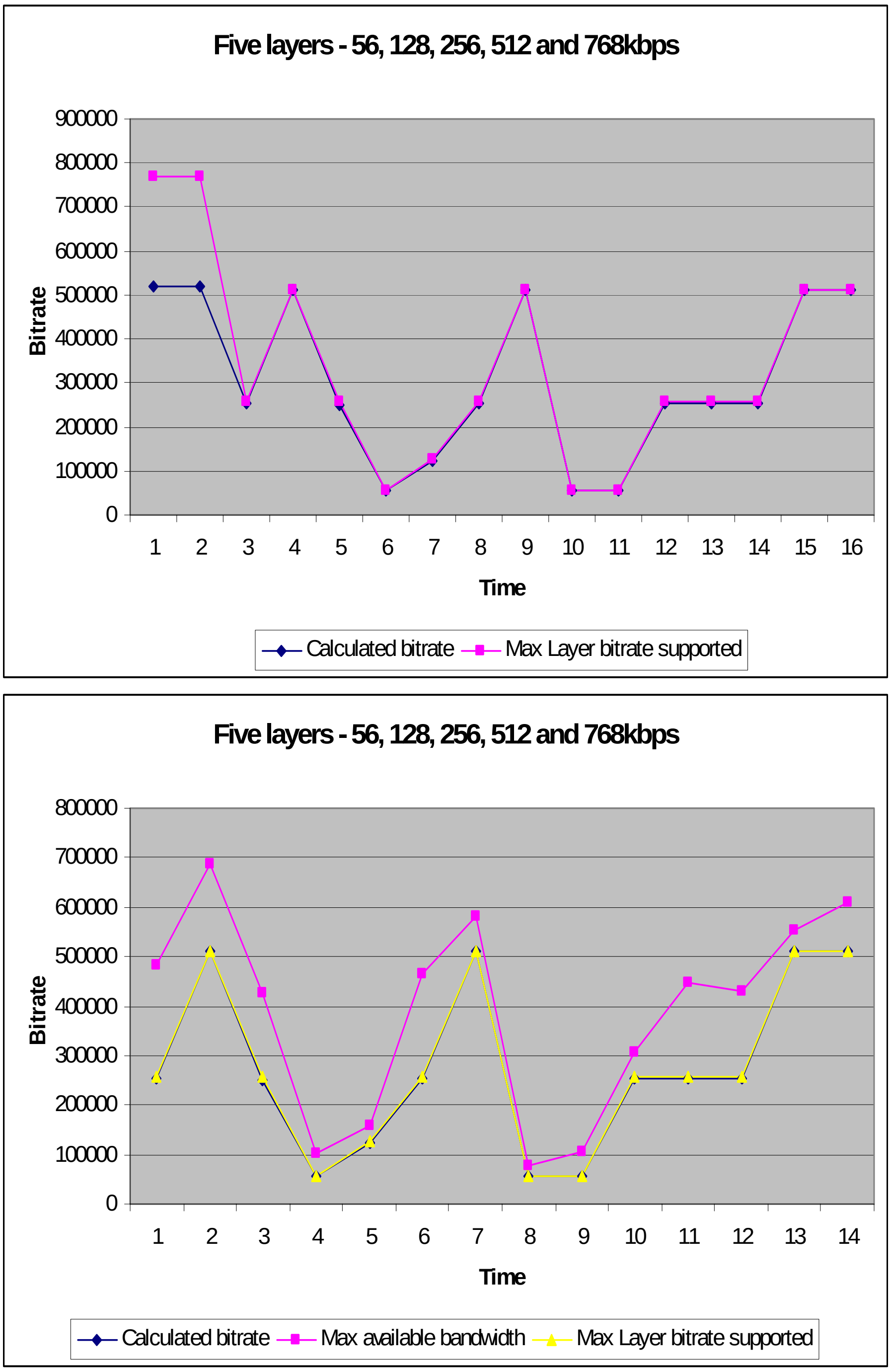


Fig. 21. Calculated bit rate vs. max layer bit rate vs. max available bandwidth (5 layers: 56Kbps, 128Kbps, 256Kbps, 512Kbps and 768Kbps).

2.2.3.6 Percentage of packet loss vs frame dimensions

Here we can see some figures that reveal the relationship between the frame loss rate and the frame dimensions. This relationship is presented separately for every layer so we can see clearly the increase of the frame dimensions as more layers are added.

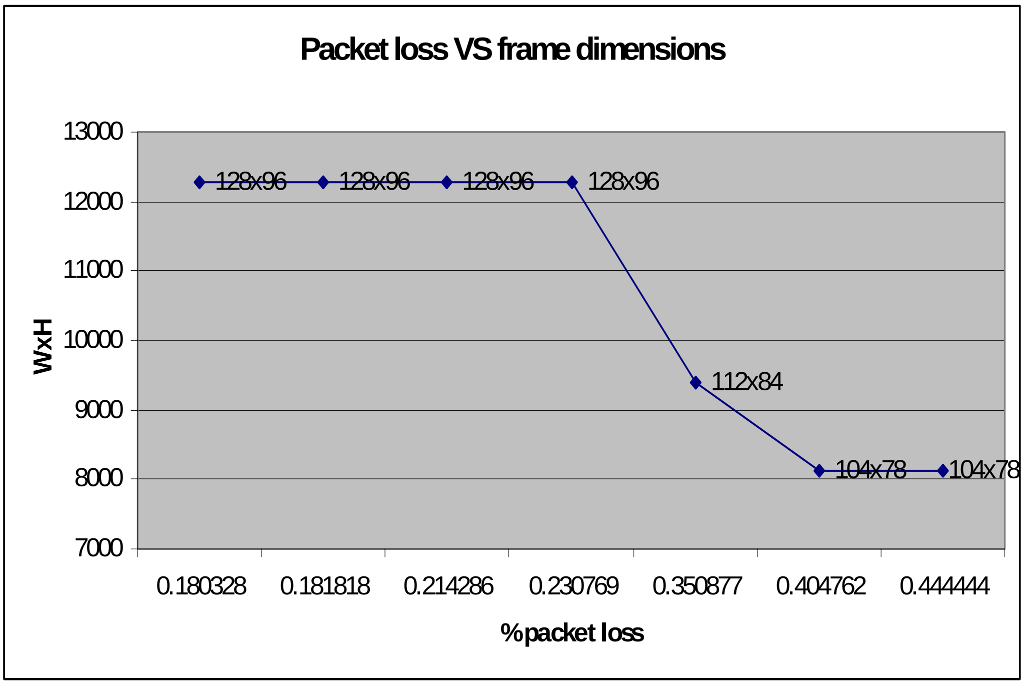


Fig. 22. Packet loss vs frame dimensions (1 layer: 56Kbps).

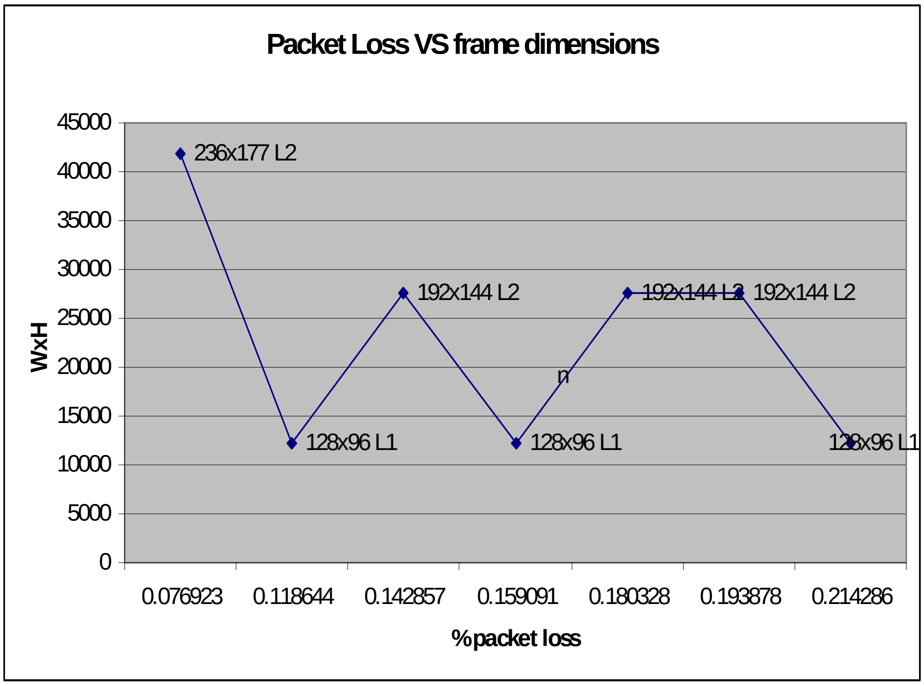


Fig. 23. Packet loss vs. frame dimensions (2 layers: 56Kbps and 128Kbps).

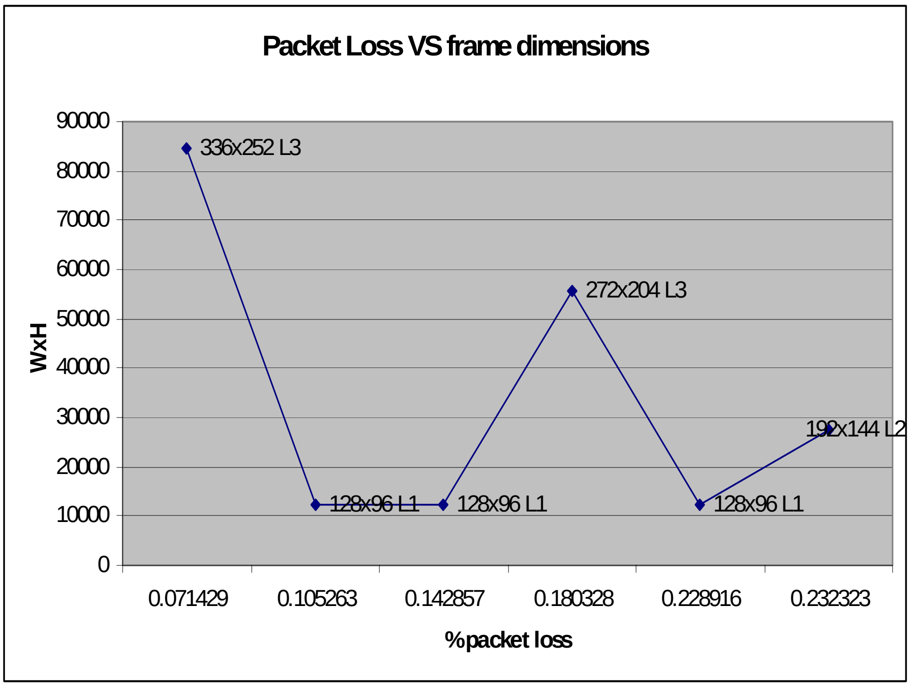


Fig. 24. Packet loss vs frame dimensions (3 layers: 56Kbps, 128Kbps and 256Kbps).

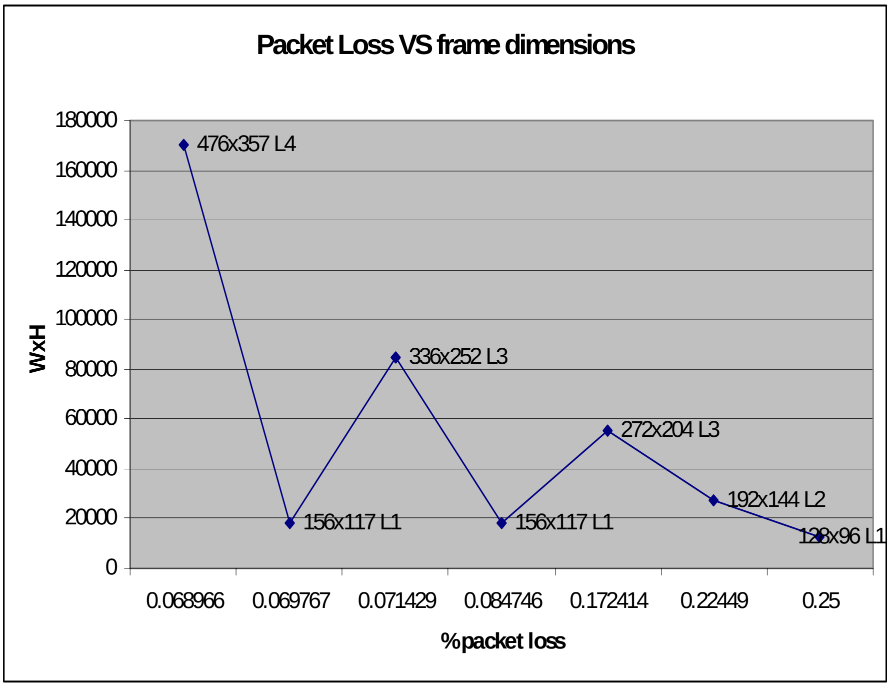


Fig. 25. Packet loss vs. frame dimensions (4 layers: 56Kbps, 128Kbps, 256Kbps and 512Kbps).

2.2.4 Examples and Screenshots

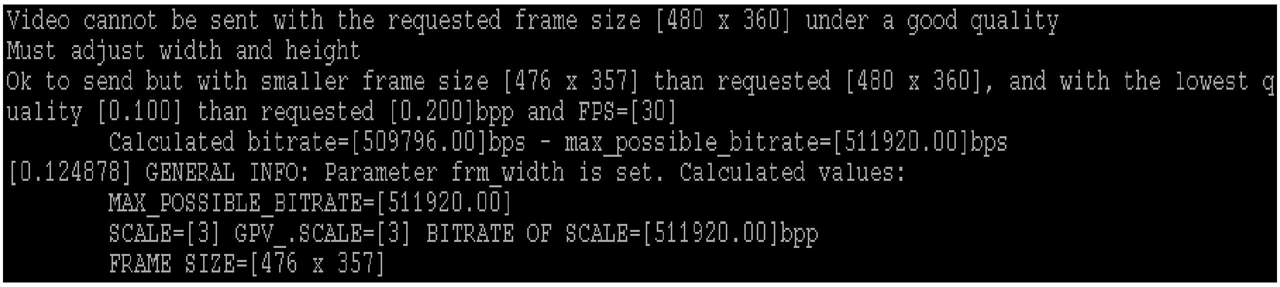
In this section we provide some screen shots where we can see how our algorithm operates regarding the decisions taken during simulation time. The following table includes the settings assigned by a user.

Parameter	Value
No. of layers	5
Bit rate of layers	56, 128, 256, 512 and 768Kbps

frame width	480
frame height (calculated)	360
bpp	0.2
fps	15
aspect ratio	1.33 (i.e. 4:3)

Table 13. User defined settings.

Screen shot 1



```
Video cannot be sent with the requested frame size [480 x 360] under a good quality
Must adjust width and height
Ok to send but with smaller frame size [476 x 357] than requested [480 x 360], and with the lowest q
uality [0.100] than requested [0.200]bpp and FPS=[30]
        Calculated bitrate=[509796.00]bps - max_possible_bitrate=[511920.00]bps
[0.124878] GENERAL INFO: Parameter frm_width is set. Calculated values:
        MAX_POSSIBLE_BITRATE=[511920.00]
        SCALE=[3] GPV_.SCALE=[3] BITRATE OF SCALE=[511920.00]bpp
        FRAME SIZE=[476 x 357]
```

Fig. 26. Screenshot 1.

As shown in Fig. 26 the maximum possible bit rate in that time was 511.92Kbps, so the video stream bit rate cannot overcome this constraint. In other words the higher layer that can be transmitted is the 4th layer. Thus our algorithm has to calculate the higher possible video stream bit rate in order to satisfy this constraint as well as user needs and requirements shown in Table 13.

Initially the algorithm calculates video stream bit rate according to user settings using the following equation:

$$\text{Bitrate} = 0.2 \times 15 \times 480 \times 360 = 518400 \text{ bps}$$

But as we stated above the video stream cannot be sent having frame dimensions of 480x360, bpp parameter equal to 0.2 and frame rate equal to 15fps as user requested because the calculated bit rate exceeds 511.92Kbps. Further calculations must be done while keeping the frame dimensions constant. According to Table 5 of D2.2 the new values of bpp and fps are 0.1 and 30fps respectively (keeping the frame dimensions constant, the lower bound for medium quality is 0.1). The new calculated bit rate is:

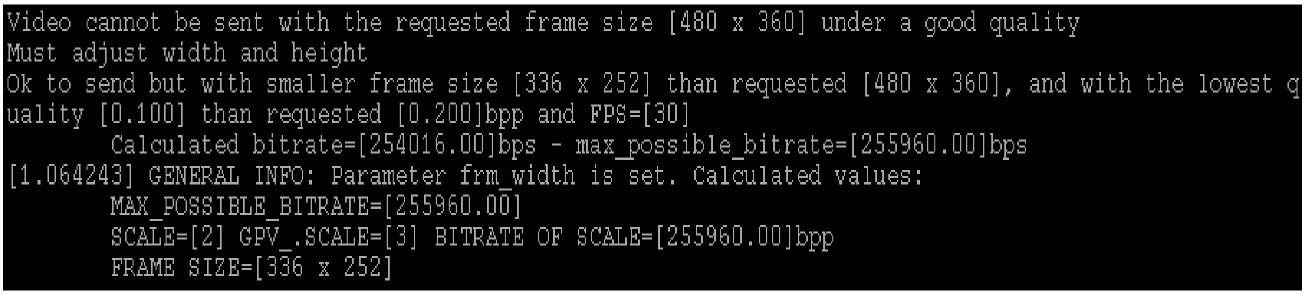
$$\text{Bitrate} = 0.1 \times 30 \times 480 \times 360 = 518400$$

As we see this new bit rate value remains above the limit of 511.92Kbps. In this case the algorithm is forced to reduce the frame dimensions (it has already reduced the quality down to the lower value). Due to the fact that the aspect ratio is 4:3 we divide the frame width by 4 and frame height by 3 and the new frame size is equal to 476x357 pixels. The new calculated bit rate is now below the limit as shown:

$$\text{Bitrate} = 0.1 \times 30 \times 476 \times 357 = 509796$$

Now the video streaming server may send the video stream using the parameters evaluated in the final step.

Screen shot 2



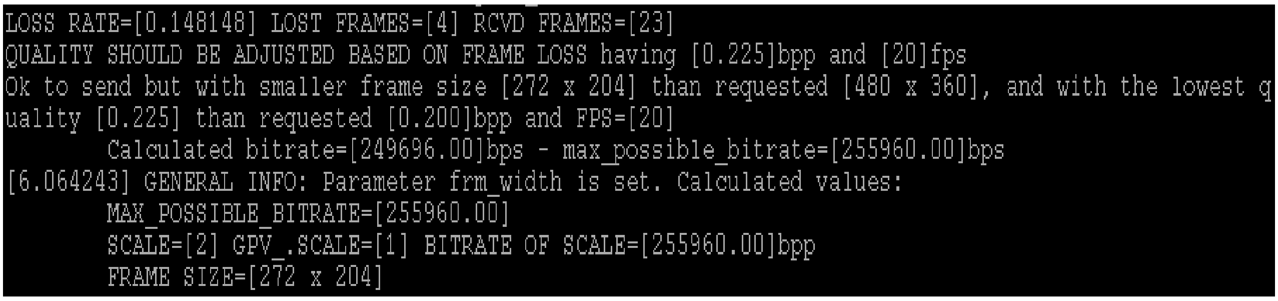
```
Video cannot be sent with the requested frame size [480 x 360] under a good quality
Must adjust width and height
Ok to send but with smaller frame size [336 x 252] than requested [480 x 360], and with the lowest q
uality [0.100] than requested [0.200]bpp and FPS=[30]
        Calculated bitrate=[254016.00]bps - max_possible_bitrate=[255960.00]bps
[1.064243] GENERAL INFO: Parameter frm_width is set. Calculated values:
        MAX_POSSIBLE_BITRATE=[255960.00]
        SCALE=[2] GPV_.SCALE=[3] BITRATE OF SCALE=[255960.00]bpp
        FRAME SIZE=[336 x 252]
```

Fig. 27. Screenshot 2.

In this case the maximum possible bit rate is 255.96Kbps therefore the streaming server can send up to the third layer. The initial calculated bit rate (according to user needs) is above this value so the algorithm decreases the quality down to a lower bound but (as before) the value obtained is again above the limit. In this point the algorithm reduces the frame dimensions up to the point that the calculated bit rate is smaller than the maximum possible bit rate. The smaller acceptable value for frame size is 336x252 pixels where bit rate is 254.016 kbps. The video stream can be sent using these parameters.

Bitrate = 0.2 x 15 x 480 x 360 = 518400
Bitrate = 0.1 x 30 x 480 x 360 = 518400
Bitrate = 0.1 x 30 x 476 x 357 = 509796
…
Bitrate = 0.1 x 30 x 336 x 252 = 254016

Screen shot 3



```
LOSS RATE=[0.148148] LOST FRAMES=[4] RCVD FRAMES=[23]
QUALITY SHOULD BE ADJUSTED BASED ON FRAME LOSS having [0.225]bpp and [20]fps
Ok to send but with smaller frame size [272 x 204] than requested [480 x 360], and with the lowest q
uality [0.225] than requested [0.200]bpp and FPS=[20]
        Calculated bitrate=[249696.00]bps - max_possible_bitrate=[255960.00]bps
[6.064243] GENERAL INFO: Parameter frm_width is set. Calculated values:
        MAX_POSSIBLE_BITRATE=[255960.00]
        SCALE=[2] GPV_.SCALE=[1] BITRATE OF SCALE=[255960.00]bpp
        FRAME SIZE=[272 x 204]
```

Fig. 28. Screenshot 3.

The scenario depicted above involves packet loss due to congestion within the core network. In this scenario the packet loss percentage reaches 14.8% therefore the adaptive algorithm has to calculate the number of excessive frames that should be sent so that the user will not perceive that many packets were lost. According to section 6.2.2 of D2.2 (equation 12) the number of frames should be sent is:

(100 x 15) / (100 – 14.8) = 17.6 frames

which can be rounded up to 20. By looking Table 5 of D2.2 we can see that the combination that provides the highest quality given the fps equal to 20 corresponds to bpp 0.225. Using these parameters our algorithm will calculate the video stream bit rate while keeping the frame size constant. If the transmission of the stream is not feasible then the algorithm will reduce the dimensions of the frame as shown below:

Bitrate = 0.225 x 20 x 480 x 360 = 777600
Bitrate = 0.225 x 20 x 476 x 357 = 764694
…
Bitrate = 0.225 x 20 x 272 x 204 = 249696

Finally the video will be transmitted having frame size of 272 x 204 pixels, quality parameter bpp equal to 0.225 whereas the frame rate will be equal to 20 fps.

2.3 ACP Performance Evaluations

Our objective has been to develop a window based protocol which does not require maintenance of per flow states within the network and satisfies all the design objectives of congestion control protocols. These objectives have been outlined in section II. In this section, we demonstrate through simulations that ACP satisfies these objectives to a very good extent. We also conduct a comparative study and demonstrate how ACP fixes the performance problems encountered by XCP [18]. We conduct our simulations on the ns-2 simulator. In our simulations we mainly consider bulk data transfers but we also evaluate the performance of the protocol in the presence of short web like flows.

2.3.1 Scalability

It is important for congestion control protocols to be able to maintain their properties as network characteristics change. We thus investigate the scalability of ACP with respect to changing link bandwidths, propagation delays and number of users utilizing the network.

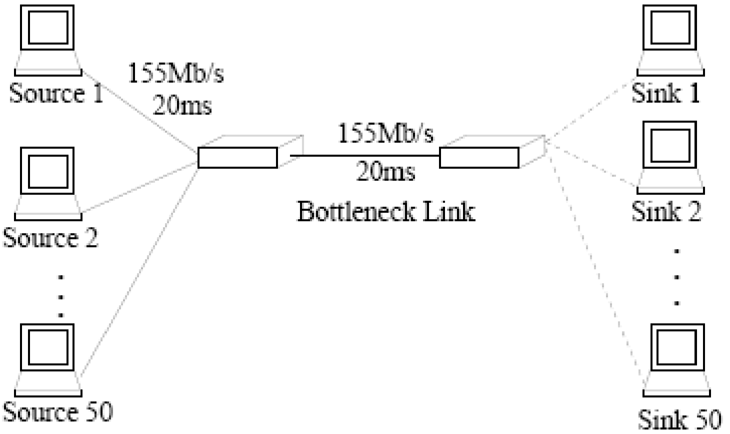
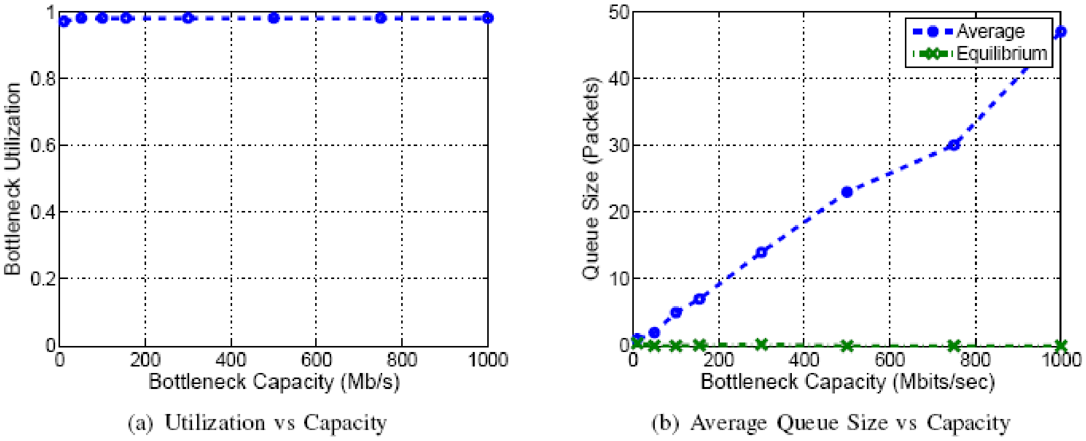


Fig. 29. Single bottleneck link topology used to investigate the scalability of ACP with respect to changing link capacities, delays, and number of users.

We conduct our study by considering the single bottleneck link network shown in Fig. 29. In the basic setup, 50 users share the bottleneck link through access links. The bandwidth of all links in the network is set equal to 155Mb/sec and their propagation delay is set equal to 20msec. As mentioned above, the purpose of this study is to investigate the scalability of ACP with respect to changing bandwidths, delays and number of users utilizing the network. When investigating the scalability of the protocol with respect to a particular parameter, we fix the other parameters to the values of the basic setup and we evaluate the performance of the protocol as we change the parameter under investigation. We consider bandwidths in the range 10Mbits/s-1Gbit/sec, delays in the range 10msec-1sec and number of users in the range 1-1000. The performance metrics that we use in this study is the average utilization of the bottleneck link and the queue size of the buffer at the bottleneck link. We consider two measures for the queue size: the average queue size and the equilibrium queue size. The average queue size is calculated over the entire duration of the simulation and thus contains information about the transient behaviour of the system. The equilibrium queue size is calculated by averaging the queue length values recorded after the system has converged to its equilibrium state. We do not report packet drops, as in all simulations we do not observe any. In addition, we do not show fairness plots, as in all simulations the network users are assigned the same sending rate at equilibrium, which implies that max-min fairness is achieved in all cases. The dynamics of the protocol and its ability to perform well in more complex network topologies are investigated in separate studies in later sections.

In our simulations, we consider persistent FTP sources. The packet size is equal to 1000 bytes and the buffer size of all links is set equal to the bandwidth delay product. The simulation time is not constant. It varies depending on the round trip propagation delay. We simulate for a sufficiently long time to ensure that the system has reached an equilibrium state. It is highly unlikely that in an actual network the network users will enter the network simultaneously. So, in all scenarios, the users enter the network with an average rate of one user per round trip time.

Effect of Capacity: We first evaluate the performance of the ACP protocol as we change the link bandwidths. We fix the number of users to 50, we fix the propagation delays to 20msec and we consider link bandwidths in the range 10Mbits/s-1Gbit/s. Plots of the bottleneck utilization and the average queue size versus the link capacity are shown in Fig. 30. We observe that ACP scales well with increasing bandwidths. The protocol achieves high network utilization (≈ 98%) at all bandwidths. Moreover, the queue size always converges to an equilibrium value which is close to zero. The average queue size remains very small but we do observe an increasing pattern. The reason for this becomes apparent when we investigate the transient properties of the protocol.



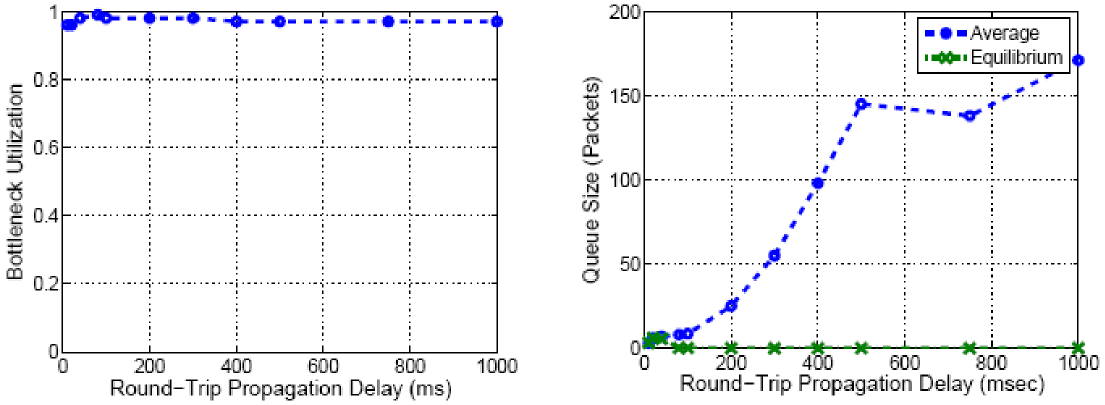
(a) Utilization vs Capacity

(b) Average Queue Size vs Capacity

Fig. 30. ACP achieves high network utilization and experiences no drops as the capacity increases. The average queue size increases with increasing capacity due to larger instantaneous queue sizes in the transient period. However, at all capacities, the queue size at equilibrium is close to zero.

In the transient period, during which the users gradually enter the network, the queue size at the bottleneck link experiences an instantaneous overshoot, before settling down to a value which is close to zero. As the bandwidth increases the maximum value of this overshoot increases, thus causing the average queue size to increase as well. However, in all cases the queue size at equilibrium is small as required.

Effect of Delays: We then investigate the performance of ACP as we change the propagation delay of the links. Any change in the link propagation delay causes a corresponding change in the round trip propagation delay of all source destination paths. We fix the link bandwidths to 155Mbits/s, we fix the number of users to 50 and we consider round-trip propagation delays in the range 10ms-1sec. Plots of the bottleneck utilization and the average queue size versus the round trip propagation delays are shown in Fig. 31.



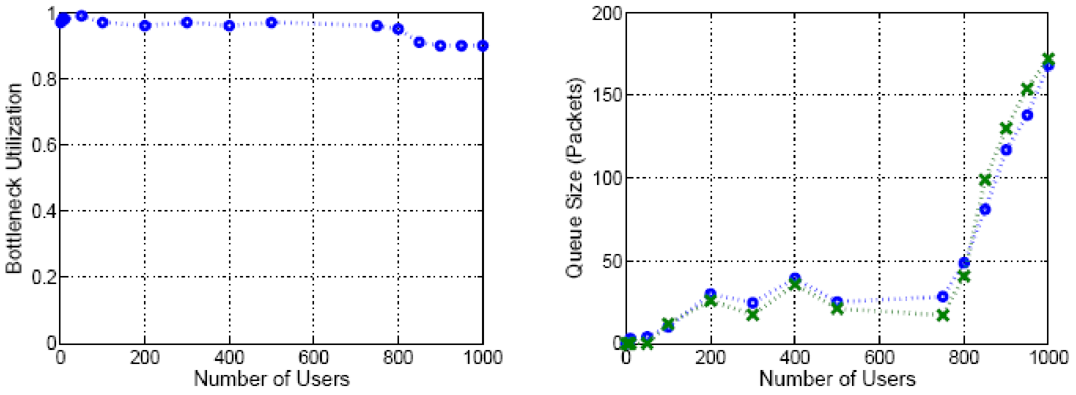
(a) Utilization vs Delay

(b) Average Queue Size vs Delay

Fig. 31. ACP achieves high network utilization and experiences no drops as the round trip propagation delay increases. The average queue size increases with increasing propagation delay due to larger instantaneous queue sizes in the transient period. However, at all delays, the queue size at equilibrium is close to zero.

The results are similar to the results obtained when investigating the effect of changing capacities. Fig. 31 (a) demonstrates that the protocol achieves high network utilization at all delays. The equilibrium queue size remains very small; however, the average queue size increases. This trend, as in the case of capacities, is due to the increasing instantaneous queue size in the transient period. As the propagation delays increase, the maximum of the overshoot observed in the transient period increases, thus causing an increase in the average queue size. Although, the average queue size increases the queue size at equilibrium is close to zero as required.

Effect of the Number of Users: We finally investigate the performance of ACP as we increase the number of users utilizing the single bottleneck link network in Fig. 29. We consider different number of users in the range 1-1000. Plots of the bottleneck utilization and the average queue size versus the number of users are shown in Fig. 32.



(a) Utilization vs Number of Users (b) Average Queue Size vs Number of Users

Fig. 32. ACP achieves high network utilization and experiences no drops as the number of users increases. At high number pf users, the utilization drops slightly and the average queue size increases. The reason is that the fair congestion window is small (close to 1). Since the congestion window can only take integer values both the utilization and queue size oscillate thus causing a slight degradation in performance.

We observe that up to approximately 800 users the protocol satisfies the control objectives as it achieves high network utilization and small queue sizes. However, unlike the previous two cases, the equilibrium queue size is not close to zero. It exhibits similar behavior to the behavior of the average queue size.

The reason for this is that as the number of users increases the queue size experiences oscillations. These oscillations dominate the overshoots observed during the transient period and so the equilibrium queue size calculated is very close to the average queue size. The oscillatory behaviour at equilibrium is caused by the fact that the congestion window can only take integer values. When the fair congestion window is not an integer (which is the common case), the desired sending at the link is forced to oscillate about the equilibrium value, thus causing oscillations of the input date rate and the queue size. As the number of users increases, these oscillations grow in amplitude and at some point they cause a significant degradation in performance. We observe in Fig. 32 that when the network is utilized by more than 800 users, the utilization drops to about 90% and the average queue size increases. The reason is that at such a high number of users, the fair congestion window is close to 1. Since the congestion window can only take integer values, it oscillates between 1 and 2. These oscillations of the congestion window cause both the utilization and the queue size to oscillate. This behaviour causes a decrease in the observed average utilization and an increase in the observed average and equilibrium queue size.

2.3.2. Performance in the presence of short flows

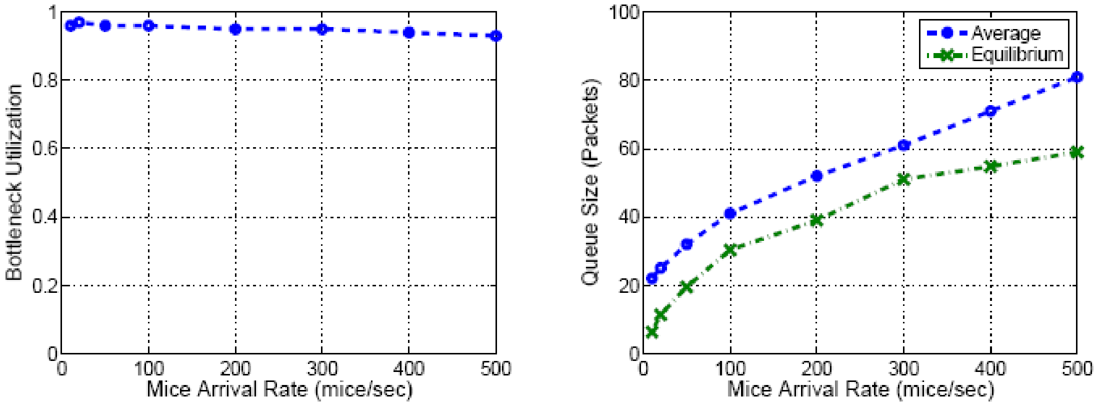
In our performance analysis so far we have only considered persistent FTP flows which generate bulk data transfers. Internet traffic, however, consists of both short and long flows. The set of flows is dominated by a relatively few elephants (long flows) and a very large number of mice (short flows).

Elephants, although smaller in number, account for the biggest percentage of the network traffic. Short flows account for a smaller percentage which however, cannot be ignored. In this section, we evaluate the performance of ACP in the presence of short web like flows.

We consider the single bottleneck link network shown in Fig. 29. The bandwidth of each link is set equal to 155Mbits/sec and the round trip propagation delay is equal to

80msec. 50 persistent FTP flows share the single bottleneck link with short web like flows. Short flows arrive according to a Poisson process.

We conduct a number of tests where we change to mean of this arrival process to emulate different traffic loads. The transfer size is derived from a Pareto distribution with an average of 30 packets. The shape of this distribution is set to 1.35.



(a) Utilization vs mice arrival rate

(b) Average queue size vs mice arrival rate

Fig. 33. ACP achieves high network utilization and maintains small queue sizes as the arrival rate of short web-like flows increases. Note that 500 users per second corresponds to a link load of 75%. In simulation, the transfer rate of short flows is derived from Pareto distribution with an average of 30 packets and a shape factor equal to 1.35.

In Fig. 33 we show plots of the utilization and the average queue size at the bottleneck link versus the mean arrival rate of the short flows. We observe that as we increase the arrival rate, the utilization drops slightly whereas both the average queue size and the equilibrium queue size increase. The important thing is that the queue size remains small and no packet drops are observed. It must be noted that 500 users per second corresponds to a link load of 75%. Experiments have shown that short flows account for about 20% of the traffic. In this regime, the utilization recorded at the bottleneck link is 96% which is satisfactory.

2.3.3 Fairness

Our objective in this work has been to develop a congestion control protocol which at equilibrium achieves max-min fairness. In this section we investigate the effectiveness of ACP to achieve max-min fairness in a scenario where the max-min fair sending rates change dynamically due to changes in the network load.

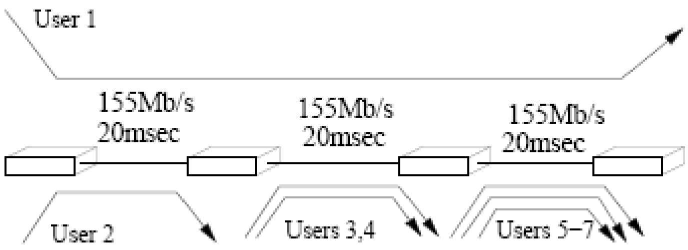


Fig. 34. A three link network used to investigate the ability of ACP to achieve max-min fairness. The first two users utilize the network throughout the simulation, users 3 and 4 start sending data at 20 seconds and users 5-7 start sending data at 40 seconds.

We consider the three link network shown in Fig. 34. The bandwidth of each link is set equal to 155Mbits/s and the propagation delay of each link is set equal to 20msec. 7 users utilize the network at different time intervals. At the beginning only users 1 and 2 utilize the network. The path of the first user traverses all three links while the path of the second user traverses the first link only. During the time that only these two users are active, the first link is the bottleneck link of the network and the fair sending rate for the two links is 77.5Mbits/s. At 20 seconds users 3 and 4 enter the network. Both users traverse the second link which becomes the bottleneck link for users 1, 3 and 4. User 2 is still bottlenecked at the first link since this is the only link that it utilizes. Note that at 20 seconds, user 2 increases its window to take up the slack created by user 1 sharing the bandwidth of link 2 with the other 2 users. At 40 seconds users 5-7 start sending data through the third link which now becomes the bottleneck link for users 1, 5, 6 and 7. User 2 is bottlenecked at the first link whereas users 3 and 4 are still bottlenecked at the second link.

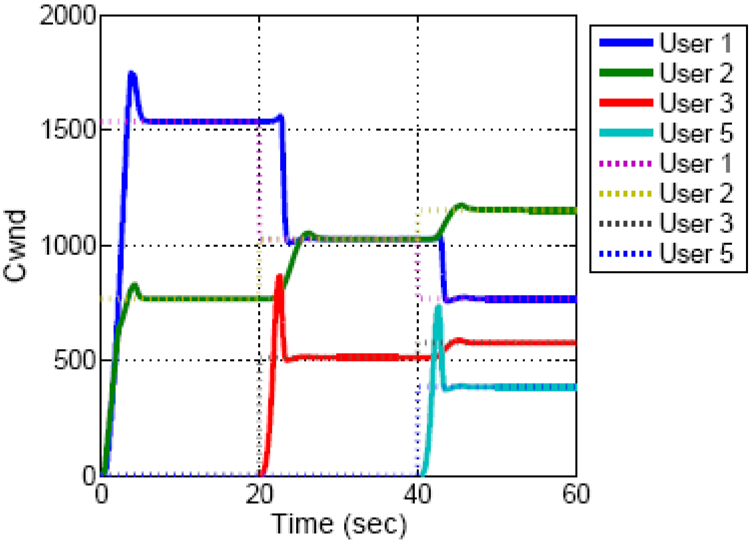


Fig. 35. Time response of the congestion window of a representative number of users compared with the theoretical max-min values. The theoretical values are denoted by dotted lines.

In Fig. 35 we show the time responses of the congestion window of a representative number of users. These responses are compared with the theoretical max-min allocation values at each time. The actual responses are denoted by solid lines whereas the theoretical values are denoted by dotted lines. We observe that at equilibrium, the actual values match exactly the theoretical values which implies that max-min fairness is achieved at all times. One thing to notice is that during the first 20 seconds, the congestion windows of users 1 and 2 are different, despite the fact that their theoretical max-min sending rates in this period are the same. There is no inconsistency between the two observations. The two users experience different round trip propagation delays as they travel different number of hops. Although their sending rates are identical, the different round trip times generate different congestion windows. This demonstrates the ability of ACP to achieve fairness in the presence of flows with different round trip times and number of hops. Also note that the response of user 4 equals the response of user 3 and the response of users 6 and 7 are equal to the response of user 5 and are thus not shown.

Another interesting observation is the overshoot in the response of user 3. This is a result of the second link becoming a bottleneck link only when users 3 and 4 enter the network. During the time that only users 1 and 2 utilize the network, the two users are bottlenecked at the first link, and so the input data rate in the second link is consistently less than the capacity. This causes the algorithm which updates the desired sending rate at the link, to consistently increase the desired sending rate. Basically, the link asks for more data, the users do not comply because they are bottlenecked elsewhere and the link reacts by asking for even more data. The desired sending rate, however, does not increase indefinitely. A projection operator in the link algorithm causes the desired sending rate at the second link to converge to the link capacity.

When users 3 and 4 enter the network the second link becomes their bottleneck link. Their sending rate thus becomes equal to the desired sending rate computed at the link. Since the desired sending rate is originally equal to the link capacity, the congestion windows of the two users experience an overshoot before settling down to their equilibrium value. This can be observed in Fig. 35. Despite this overshoot the system does not experience any packet drops. The above setting can be used to emulate the case where network users cannot comply with the network's request because they do not have enough data to send. The above shows the ability of ACP to also cope with this case.

2.3.4 Dynamics of ACP

To fully characterize the performance of the proposed protocol, apart from the properties of the system at equilibrium, we need to investigate its transient properties. The protocol must generate smooth responses which are well damped and converge fast to the desired equilibrium state. To evaluate the transient behaviour of ACP, we consider the single bottleneck link network shown in Fig. 29 and we generate a dynamic environment where users enter and leave the network at different times. In such an environment, we investigate the dynamics of the user sending rates, we examine the queuing dynamics at the bottleneck link and we also evaluate the performance of the estimator which is used to track the number of users utilizing the network.

To conduct our study we consider the following scenario. 30 users originally utilize the single bottleneck link network shown in Fig. 29. At 30 seconds 20 of theses users stop sending data simultaneously. So the number of users utilizing the network is reduced to 10. At 45 seconds, however, 40 additional users enter the network thus causing the number of users to increase to 50.

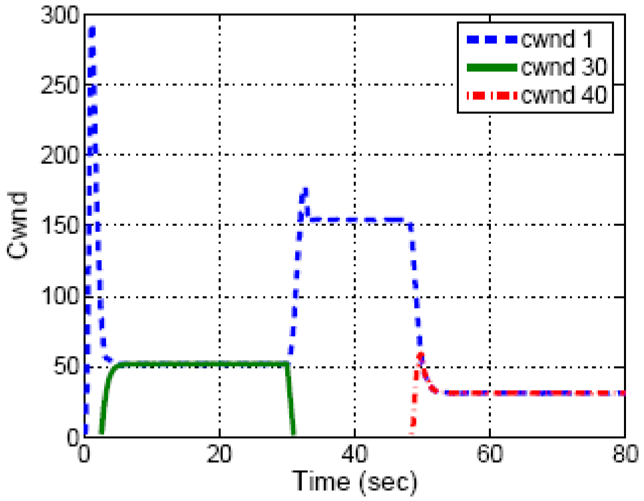
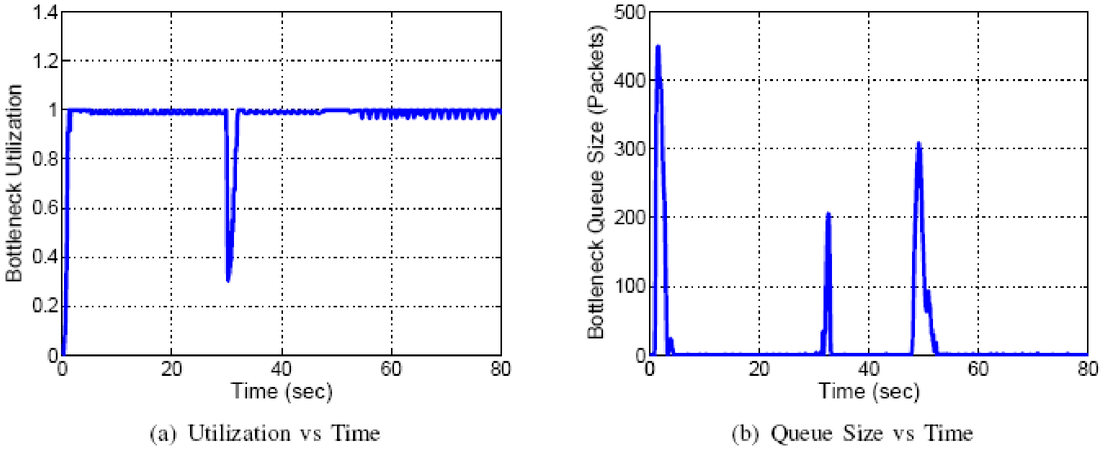


Fig. 36. Time response of the congestion window of three users. User 1 utilizes the network throughout the simulation, user 30 stops sending data at 30 seconds and user 40 enters the network at 45 seconds. We observe fast and smooth responses with no oscillations.

In Fig. 36 we present the time responses of the congestion window of a representative number of users. User 1 utilizes the network throughout the simulation, user 30 stops sending data at 30 seconds and user 40 enters the network at 45 seconds. The transient behaviour of the other users is very similar to the ones shown in Fig. 36. We observe that the protocol achieves smooth responses which converge fast to the desired equilibrium with no oscillations. However, in some cases, they experience overshoots. When user 1 starts sending data it converges fast to its max-min fair allocation. Since the users gradually enter the network, the max-min allocation gradually decreases. This is why the congestion window of user 1 experiences a large overshoot before settling down to its equilibrium value. Note, however, that once the desired sending rate calculated at the bottleneck link has settled down to an equilibrium value, a new user, such as user 30, converges fast to the max-min allocation value with no overshoots. When the 20 users suddenly stop sending data at 30 seconds the flow of data through the bottleneck link drops thus causing an instantaneous underutilization of the link. The link identifies this drop in the input data rate and reacts by increasing its desired sending rate. This causes user 1 to increase its congestion window. The time response in Fig. 36 indicates fast convergence to the new equilibrium value with no oscillations. However, the response does experience a small overshoot before settling down to its equilibrium value. This slight overshoot is caused by the feedback delays and the pure integral action of the congestion controller. It can be avoided by introducing proportional action. However, such a modification would increase the complexity of the algorithm without significantly improving the performance and is thus avoided. When 40 new users enter the network at 45 seconds, the max-min fair sending rate decreases. The controller at the bottleneck link iteratively calculates this rate and communicates this information to the end users.

This causes user 1 to decrease its congestion window and user 40 which has just entered the network to gradually increase its congestion window to the equilibrium value. We observe from Fig. 36 that user 1 converges fast to the new equilibrium value with no undershoots or oscillations. We also observe that the time response of the congestion window of user 40 experiences a small overshoot before settling down to its equilibrium value. This is due to the fact that the user sets its sending rate equal to the desired sending rate calculated at the bottleneck link while the latter is still decreasing.

The next thing we investigate is the transient behaviour of the utilization and the queue size at the bottleneck link. In Fig. 37 we show the time responses of the utilization and the queue size at the bottleneck link.



(a) Utilization vs Time

(b) Queue Size vs Time

Fig. 37. Time response of the instantaneous utilization and the queue size at the bottleneck link. Utilization convergences fast to a value which is close to 1. There is an instantaneous drop when the 20 users leave the network but the protocol manages to recover quickly. The queue size experiences instantaneous increases when new users enter the network but at equilibrium at the queue size is almost zero.

We observe that the link utilization converges fast to a value which is close to 1. When the 20 users leave the network, the flow of data suddenly decreases thus causing an instantaneous decrease in the utilization. However, the system reacts quickly by increasing the sending rate of the remaining users, thus achieving almost full utilization in a very short period of time.

The time response of the queue size indicates that the latter converges to a value which is close to 0. This is what is required by the congestion control protocol in order to avoid excessive queueing delays on the long run. However, in the transient periods during which new users enter or leave the network, the queue size experiences an instantaneous increase. It might seem strange that we observe increasing queue sizes when users leave the network. This is caused by the fact that the remaining users, while they increase their sending rate to take up the slack created, they experience overshoots. It must be noted that the maximum queue size recorded in the transient period, increases as the bandwidth delay product increases. This is why in our study of the scalability properties of ACP, the average queue size increases as we increase the bandwidths and the delays. However, careful choice of the control parameters at the links and the delayed increase policy that we apply at the sources ensure that these overshoots do not exceed the buffer size and thus do not lead to packet drops.

A distinct feature of the proposed congestion control strategy is the implementation at each link of an estimation algorithm which estimates the number of flows utilizing the link. These estimates are required to maintain stability in the presence of delays. Here, we evaluate the performance of the proposed estimation algorithm. In the scenario that we have described in the previous subsection, the number of users utilizing the single bottleneck link network changes from 30 to 10 at 30 seconds and

it becomes 50 at 45 seconds. So, we evaluate the performance of the proposed estimation algorithm by investigating how well the estimator tracks these changes.
In Fig. 38 we show the time response of the output of the estimator. We observe that the estimator generates smooth responses with no overshoots or oscillations. In addition, the estimator tracks the changes in the number of users and produces correct estimates at equilibrium.

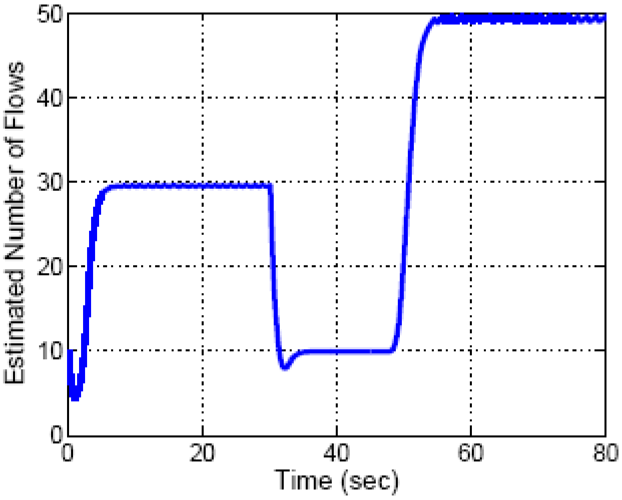


Fig. 38. Time response of the estimated number of users utilizing the bottleneck link. We observe almost perfect tracking at equilibrium an fast responses with no overshoots.

2.3.5 A multi-link example

Until now we have evaluated the performance of ACP in simple network topologies which include 1, 2 or 3 links. Our objective in this section is to investigate how ACP performs in a more complex network topology. We consider the parking lot topology shown in Fig. 39.

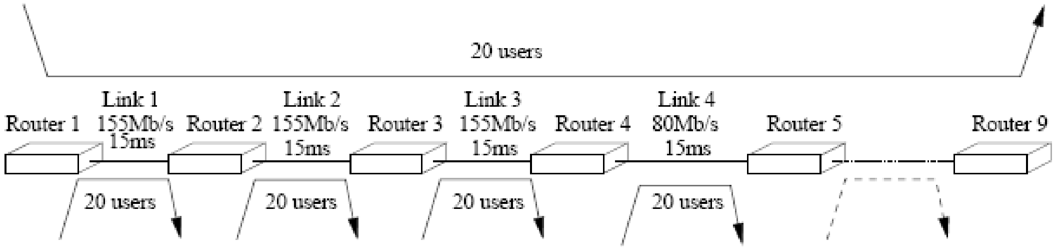
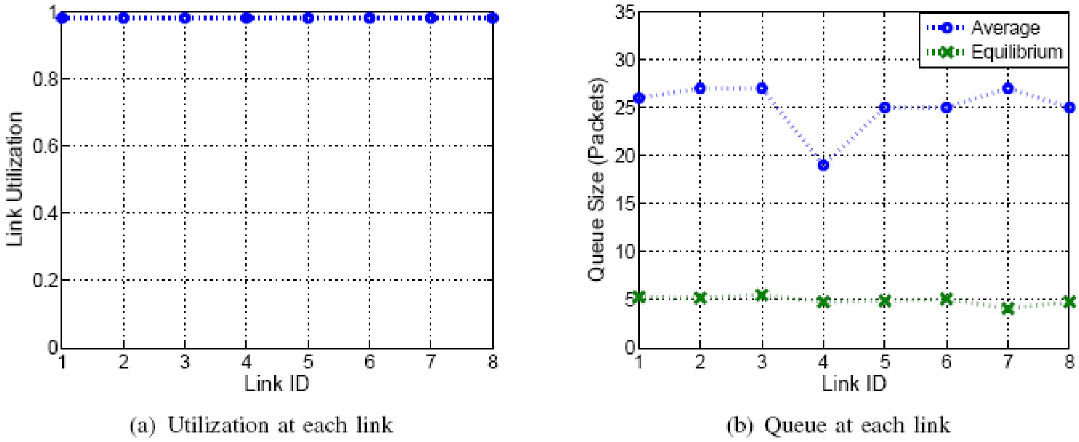


Fig. 39. A parking lot network topology.

The network consists of 8 links which are connected in series. All links have a bandwidth of 155Mbits/sec except link 4 which has a bandwidth of 80Mbits/sec. The propagation delay of all links is set equal to 15msec. 20 users utilize the network by traversing all 8 links. Moreover, each link in the network is utilized by an additional 20 users which have single hop paths as shown in Fig. 39. In this way, all links in the network are bottleneck links and link 4 is the single bottleneck link for the 20 users which traverse the whole network. We evaluate the performance of ACP by examining the utilization and the average queue size observed at each link. We do not report packets drops, as we do not observe any. In Fig. 40, we show on separate graphs the utilization achieved at each link and the average and equilibrium queue size recorded at the link.

Since all links in the network are bottleneck links for some flows, we do expect them to be fully utilized. Indeed, we observe that ACP achieves almost full utilization at all links. In addition, both the equilibrium queue size and the average queue size remain small. At link 4 we observe smaller average queue size. This is due to its smaller bandwidth delay product. This is consistent with our observations in previous sections.



(a) Utilization at each link

(b) Queue at each link

Fig. 40. ACP achieves high utilization at all links and experiences no packet drops. In addition it manages to maintain small queue sizes.

2.3.6 Comparison with XCP

Our objective in this work has been to develop a congestion control protocol which does not require maintenance of per flow states within the network and satisfies all the design objectives. An explicit congestion control protocol (XCP) which has been recently developed in [18], satisfies most of the design objectives but fails to achieve max-min fairness in the case of multiple congested links. It has been shown through analysis and simulations that when the majority of flows at a particular link are bottlenecked elsewhere, the remaining flows do not make efficient use of the residual bandwidth ([19]). In this section, we consider a topology where the above problem is evident and we demonstrate that ACP fixes this problem and achieves max-min fairness.

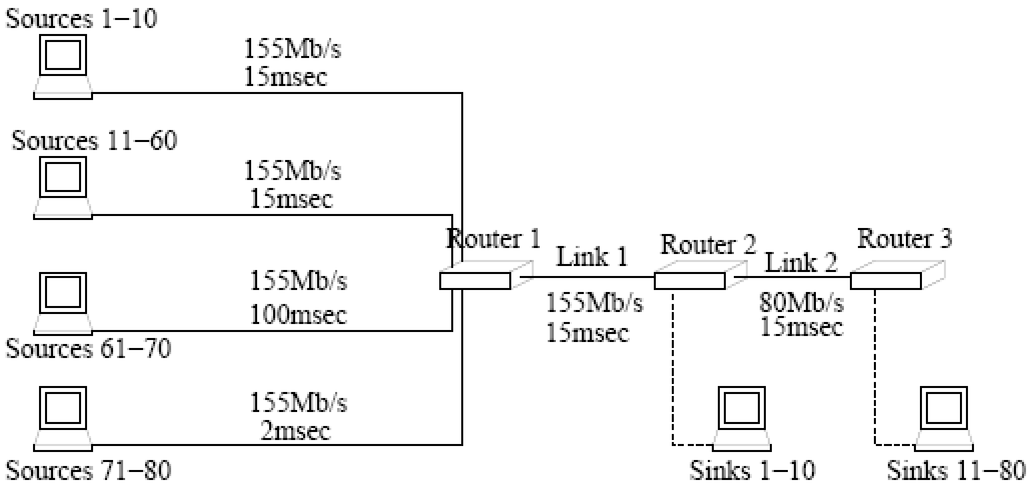


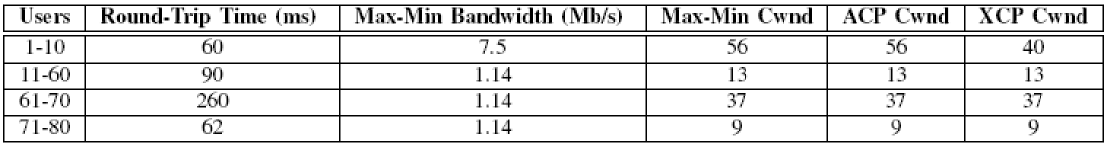
Fig. 41. A two-link network, used to investigate the ability of ACP to achieve max-min fairness at equilibrium. We consider a simulation scenario which involves users with heterogeneous round-trip times.

We consider the two link network shown in Fig. 41. Link 1 has a bandwidth of 155Mbits/sec whereas link 2 has a bandwidth of 80Mbits/sec. 80 users access the network though 155Mbits/sec access links.

The access links of the first 60 users have a propagation delay of 15msec, the access links of the next 10 users have a propagation delay of 100msec and the propagation delay of the last 10 users are set to 2msec.

We have chosen a variety of propagation delays to investigate the ability of ACP to achieve fairness in the presence of flows with multiple round trip times. The first 10 users of the network have connection sinks at the first router and the rest of the users have connection sinks at the second router. This has been done to ensure that both links are bottleneck links for some flows. The first 10 users are bottlenecked at link 1 whereas the remaining users are bottlenecked at link 2.

We simulate the above scenario using both XCP and ACP users. In Table 14 we compare the theoretical max-min congestion window values with the equilibrium values achieved by ACP and XCP. We observe that ACP matches exactly the theoretical values, whereas XCP does not. XCP fails to assign max-min sending rates to the first 10 users which utilize link 1 only. This is consistent with the findings in [19]. The other users traversing link 1 are bottlenecked at link 2 and so the 10 users which are bottlenecked at link 1 do not make efficient use of the available bandwidth. This inefficiency causes underutilization of link 1.



Users	Round-Trip Time (ms)	Max-Min Bandwidth (Mb/s)	Max-Min Cwnd	ACP Cwnd	XCP Cwnd
1-10	60	7.5	56	56	40
11-60	90	1.14	13	13	13
61-70	260	1.14	37	37	37
71-80	62	1.14	9	9	9

Table 14. Theoretical max-min fair values, compared with equilibrium values achieved by ACP and XCP.

This is demonstrated in Fig. 42 where we plot the time response of the utilization achieved at link 1 by the ACP and the XCP users. Obviously XCP causes underutilization of the link, whereas ACP achieves almost full utilization of the link at equilibrium. This example demonstrates that ACP outperforms XCP in both utilization and fairness. Another thing to note in Table 14 is the ability of ACP to achieve max-min fairness despite the presence of flows with a variety of round trip times.

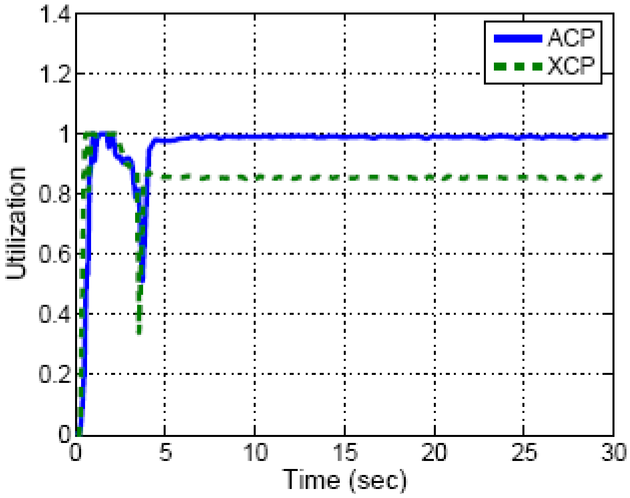


Fig. 42. Time response of the utilization of at the first link achieved by ACP and XCP. We observer that ACP achieves higher utilization.

2.4 Queue Length Based Internet Congestion Control protocol

Our objective has been to develop a window based protocol which does not require maintenance of per flow states within the network and satisfies all the design objectives of congestion control protocols. In this section, we demonstrate through simulations that the proposed protocol satisfies its design objectives to a very good extent.

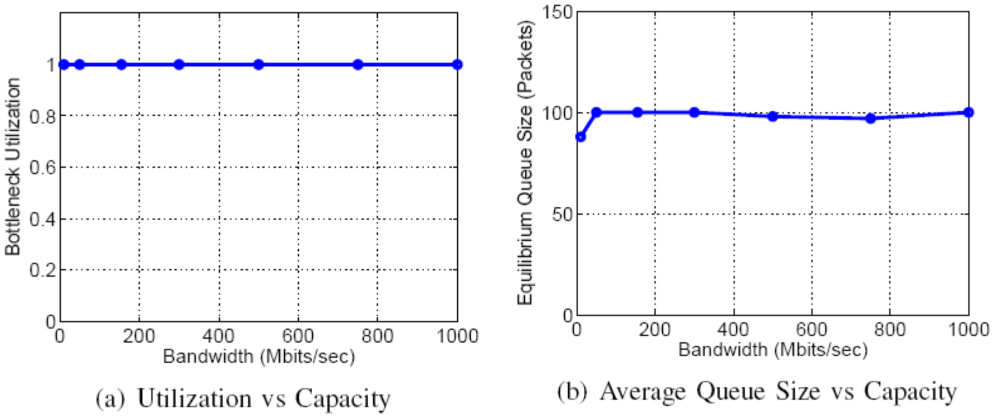
2.4.1 Scalability

It is important for congestion control protocols to be able to maintain their properties as network characteristics change. We thus investigate the scalability of the proposed protocol with respect to changing link bandwidths, propagation delays and number of users utilizing the network.

We conduct our study by considering the single bottleneck link network shown in Fig. 29. In the basic setup, 50 users share the bottleneck link through access links. The bandwidth of all links in the network is set equal to 155Mb/sec and the propagation delay is set equal to 20msec. The access links have different propagation delays. The propagation delay of the access link of the first user is set equal to the same value as that of the bottleneck link and the propagation delays of the access links of the rest of the users differ by increments of 0.5msec. In this way we create an asynchronous network. As mentioned above, the purpose of this study is to investigate the scalability of the protocol with respect to changing bandwidths, delays and number of users utilizing the network. We consider bandwidths in the range 10Mbits/s-1Gbit/sec, delays in the range 10msec-1sec and number of users in the range 1-1000. The performance metrics that we use in this study are the equilibrium utilization and the equilibrium queue size at the bottleneck link. The equilibrium values are calculated by averaging the values recorded after the system has converged to its equilibrium state. We do not report packet drops, as in all simulations we do not observe any. In addition, we do not show fairness plots, as in all simulations the network users are assigned the same sending rate at equilibrium, which implies that max-min fairness is achieved in all cases. The dynamics of the protocol and its ability to perform well in more complex network topologies are investigated in separate studies later in this section.

In our simulations, we consider persistent FTP sources. The packet size is equal to 1000 bytes and the buffer size of all links is set equal to the bandwidth delay product. The reference queue size q_{ref} is chosen to be equal to 100 packets. The simulation time is not constant. It varies depending on the round trip propagation delay. We simulate for a sufficiently long time to ensure that the system has reached an equilibrium state. It is highly unlikely that in an actual network the network users will enter the network simultaneously. So, in all scenarios, the users enter the network with an average rate of one user per round trip time.

Effect of Capacity: We first evaluate the performance of the proposed protocol as we change the link bandwidths. We fix the number of users to 50, we fix the propagation delays to 20msec and we consider link bandwidths in the range 10Mbits/s-1Gbit/s. Plots of the bottleneck utilization and the average queue size versus the link capacity are shown in Fig. 3.



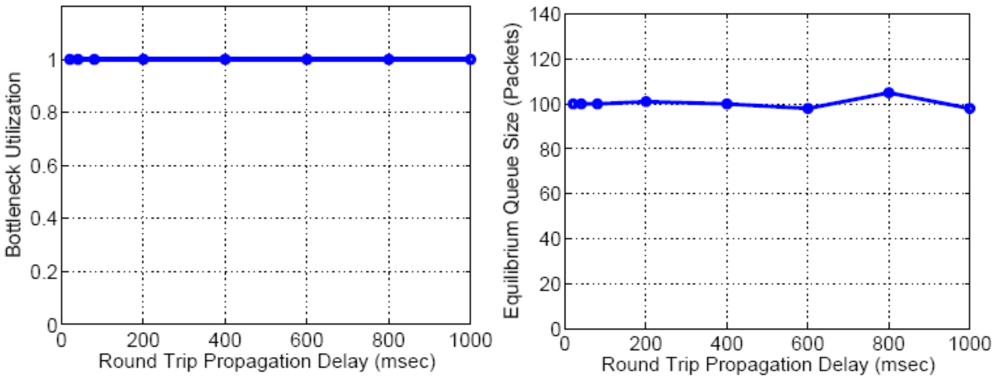
(a) Utilization vs Capacity

(b) Average Queue Size vs Capacity

Fig. 43. The protocol achieves full network utilization and experiences no drops as the capacity increases. The equilibrium queue size is always close to 100 which is the reference value.

We observe that the proposed scales well with increasing bandwidths. The protocol achieves full network utilization (100%) at all bandwidths. Moreover, the queue size always converges to an equilibrium value which is close to 100 as required.

Effect of Delays: We then investigate the performance of the protocol as we change the propagation delay of the links. Any change in the link propagation delay causes a corresponding change in the round trip propagation delay of all source destination paths. We fix the link bandwidths to 155Mbits/s, we fix the number of users to 50 and we consider round-trip propagation delays in the range 10ms-1sec. It must be noted that each user of the network has different round trip propagation delay, since the propagation delay of each access link is different. So, when we refer to the round trip propagation delay of a particular simulation we refer to the minimum round trip propagation delay among the network users. Plots of the bottleneck utilization and the average queue size versus the round trip propagation delays are shown in Fig. 44.

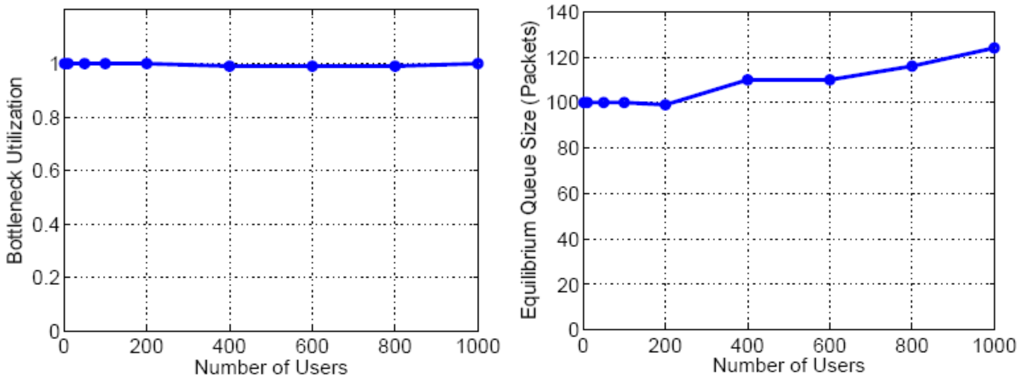


(a) Utilization vs Delay (b) Average Queue Size vs Delay

Fig. 44. The protocol achieves full network utilization and experiences no drops as the round trip propagation delay increases. The equilibrium queue size is close to 100 at all delays as required.

The results are similar to the results obtained when investigating the effect of changing capacities. Fig. 44 (a) demonstrates that the protocol achieves full network utilization at all delays and that the queue size at equilibrium is close to 100 as required.

Effect of the Number of Users We finally investigate the performance of the proposed protocol as we increase the number of users utilizing the single bottleneck link network in Fig. 29. We consider different number of users in the range 1-1000. Plots of the bottleneck utilization and the average queue size versus the number of users are shown in Fig. 45. We observe that the protocol achieves full network utilization in all cases. However, we also observe that the equilibrium queue size starts deviating from the reference queue size as the number of users increases. The reason for this is that as the number of users increases, the queue size experiences oscillations of increasing magnitude with a corresponding shifting of the mean value towards a higher value. However, this increase in the value of the queue size at equilibrium is relatively small and no packet losses are observed.



(a) Utilization vs Number of Users (b) Average Queue Size vs Number of Users

Fig. 45. The protocol achieves full network utilization and experiences no drops in all cases. However, the equilibrium queue size increases with increasing number of users.

2.4.2 The Dynamics of the protocol

To fully characterize the performance of the proposed protocol, apart from the properties of the system at equilibrium, we need to investigate its transient properties. The protocol must generate smooth responses which are well damped and converge fast to the desired equilibrium state. To conduct our study we consider the following dynamic scenario. 30 users originally utilize the single bottleneck link network shown in Fig. 29. At 30 seconds 20 of theses users stop sending data simultaneously. So the number of users utilizing the network is reduced to 10. At 45 seconds, however, 40 additional users enter the network thus causing the number of users to increase to 50.

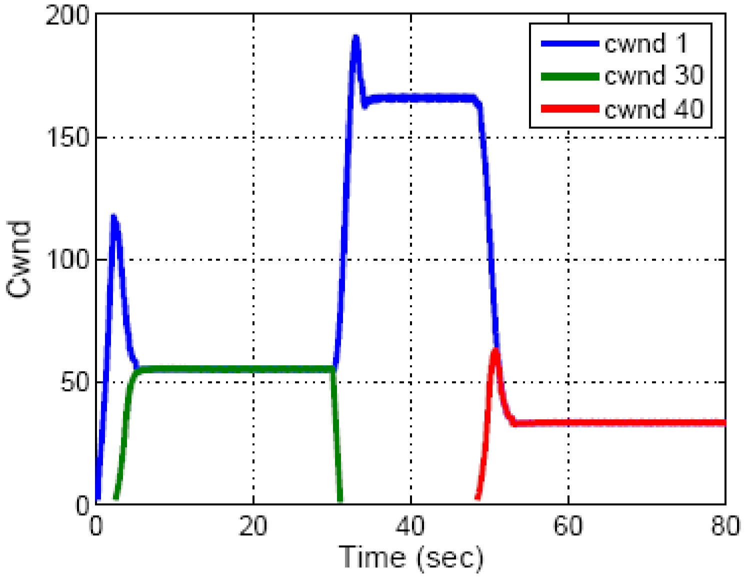
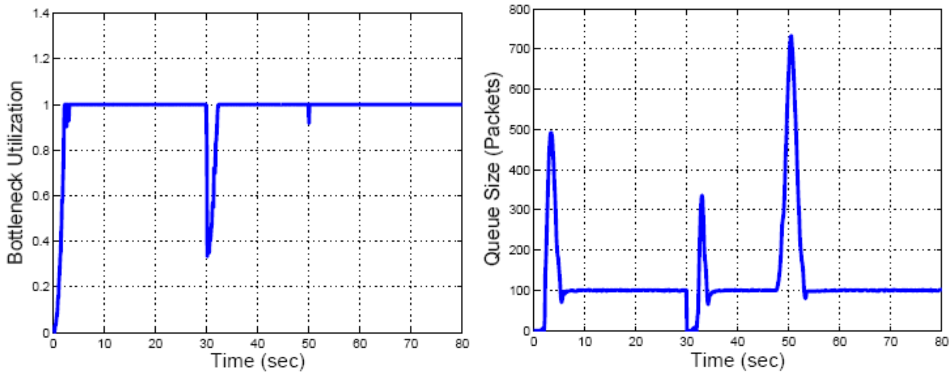


Fig. 46. Time response of the congestion window of three users. User 1 utilizes the network throughout the simulation, user 30 stops sending data at 30 seconds and user 40 enters the network at 45 seconds. We observe smooth and fast responses with no oscillations.

In Fig. 46 we present the time responses of the congestion window of a representative number of users. User 1 utilizes the network throughout the simulation, user 30 stops sending data at 30 seconds and user 40 enters the network at 45 seconds. The transient behavior of the other users is very similar to the ones shown in Fig. 46. We observe that the protocol achieves smooth responses which converge fast to the desired equilibrium with no oscillations. However, in some cases, they experience overshoots. When user 1 starts sending data it converges fast to its max-min fair allocation. Since the users gradually enter the network, the max-min allocation gradually decreases. This is why the congestion window of user 1 experiences a large overshoot before settling down to its equilibrium value. Note, however, that once the desired sending rate calculated at the bottleneck link has settled down to an equilibrium value, a new user, such as user 30, converges fast to the max-min allocation value with no overshoots. When the 20 users suddenly stop sending data at 30 seconds the flow of data through the bottleneck link drops thus causing an instantaneous underutilization of the link. The link identifies this drop in the input data rate and reacts by increasing its desired sending rate. This causes user 1 to increase its congestion window. The time response in Fig. 46 indicates fast convergence to the new equilibrium value with no oscillations. However, the response does experience a small overshoot before settling down to its equilibrium value. When 40 new users enter the network at 45 seconds, the max-min fair sending rate decreases. The controller at the bottleneck link iteratively calculates this rate and communicates this information to the end users. This causes user 1 to decrease its congestion window and user 40 which has just entered the network to gradually increase its congestion window to the equilibrium value. We observe from Fig. 46 that user 1 converges fast to the new equilibrium value with no undershoots or oscillations. We also observe that the time response of the congestion window of user

40 experiences a small overshoot before settling down to its equilibrium value. This is due to the fact that the user sets its sending rate equal to the desired sending rate calculated at the bottleneck link while the latter is still decreasing. The next thing we investigate is the transient behaviour of the utilization and the queue size at the bottleneck link. In Fig. 47 we show the time responses of the utilization and the queue size at the bottleneck link.



(a) Utilization vs Time

(b) Queue Size vs Time

Fig. 47. Time response of the instantaneous utilization and the queue size at the bottleneck link. Utilization convergences fast to a value close to 1. The queue size experiences instantaneous increases when new users enter the network but at equilibrium the queue size is equal to the reference value.

We observe that the link utilization converges fast to a value which is close to 1. When the 20 users leave the network, the flow of data suddenly decreases thus causing an instantaneous decrease in the utilization. However, the system reacts quickly by increasing the sending rate of the remaining users, thus achieving almost full utilization in a very short period of time. The time response of the queue size indicates that the latter converges to a value which is close to 100. This is the main objective of the congestion control protocol. However, in the transient periods during which new users enter or leave the network, the queue size experiences an instantaneous increase.

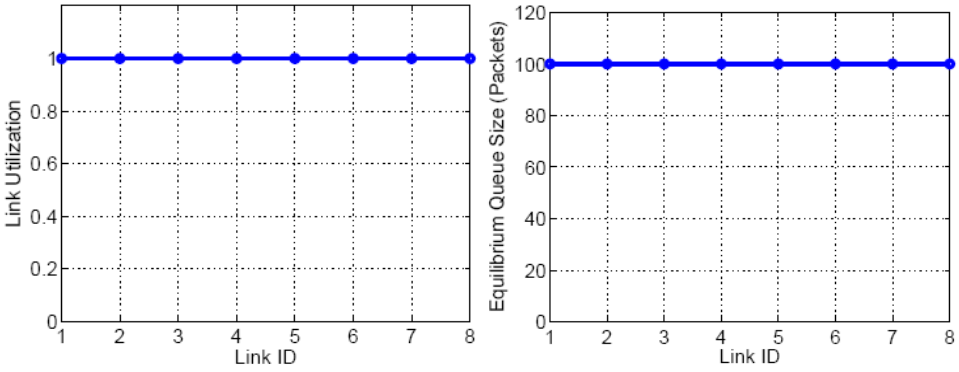
It might seem strange that we observe increasing queue sizes when users leave the network. This is caused by the fact that the remaining users, while they increase their sending rate to take up the slack created, they experience overshoots. However, careful choice of the control parameters at the links and the delayed increase policy that we apply at the sources ensure that these overshoots do not exceed the buffer size and thus do not lead to packet drops.

2.4.3 A multi-link example

Until now we have evaluated the performance of the proposed protocol in a single bottleneck link network topology. Our objective in this section is to investigate how the protocol performs in a more complex network topology. We consider the parking lot topology shown in Fig. 39.

The network consists of 8 links which are connected in series. All links have a bandwidth of 155Mbits/sec except link 4 which has a bandwidth of 80Mbits/sec. The propagation delay of all links is set equal to 15msec. 20 users utilize the network by traversing all 8 links. Moreover, each link in the network is utilized by an additional 20 users which have single hop paths as shown in Fig. 39. In this way, all links in the network are bottleneck links and link 4 is the single bottleneck link for the 20 users which traverse the whole network. In Fig. 48, we show on separate graphs the equilibrium utilization and the equilibrium queue size recorded at each link.

Since all links in the network are bottleneck links for some flows, we do expect them to be fully utilized. Indeed, we observe that the proposed protocol achieves full utilization at all links. In addition, at all links the equilibrium queue size is equal to the reference queue size as required.



(a) Utilization at each link (b) Queue at each link

Fig. 48. the protocol achieves full utilization at all links and experiences no packet drops. In addition, the equilibrium queue size is equal to 100 as required.

3. Conclusions

In this deliverable we analyzed and presented the two proposed algorithms for the adaptive video transmission over the Internet and two congestion control algorithms for high speed networks. The algorithms can operate in both wired and wireless error prone environments.

ADIVIS algorithm based on a decision module and a feedback module implemented on server side (sender-driven architecture) whereas RAF algorithm implements a receiver-driven architecture where the receiver takes the decision about the number of layers sent.

In this deliverable we evaluated ADIVIS which is specifically designed for video streaming over the Internet. Our main objective is to provide a framework that incorporates both Content Adaptation and Network Adaptation Techniques. Towards this direction, we introduce two new components; a feedback mechanism and a decision algorithm, that deal with layered video streams.

We evaluated our fuzzy rate control system under conditions of high congestion across a bottleneck link. Simulations showed that the fuzzy controller can finely adapt the video transmission rate to the available bandwidth of the link, based on loss rate per second and percentage of marked packets over a decision period T. Basically, fuzzy controller clearly detect the available bandwidth in the presence of CBR or FTP background cross traffic, and finely adapts the video transmission rate to it. Moreover, our fuzzy system achieves smooth rate change over the time something that makes it appropriate for video streaming over the Internet.

We evaluated our decision algorithm under error-free and error-prone environments and our preliminary results indicate that the algorithm can finely adapt the video stream bit rate to the available bandwidth, while providing high and stable objective quality of service at the same time. Moreover, simulations showed that the system performs best in the absence of background traffic like FTP but the objective quality remains acceptable in the presence of background FTP as well. Additionally, it seems that our algorithm provides fairness; however, this is an issue which will be further investigated in the presence of multiple concurrent users.

For future work we want to determine the sensitivity of our algorithm to various parameters (i.e. time hysteresis, decision period T). To continue with further evaluation of our adaptation approach, we need to look at the interaction between our adaptive flow and other network flows sharing the same routers. In addition, the effect of delay variation (jitter) will be taken into consideration when designing the fuzzy inference engine. Moreover, subjective tests should be considered given the fact that PSNR is inappropriate for the evaluation of the actual user perceived quality of service because it is poorly correlated to human vision.

RAF algorithm is an adaptive algorithm whose operation is based on user requirements as well as on the dynamically changing conditions of the network path. Adaptation of the content is based on the values of the parameters. The adjustment of these parameters is done prior the establishment of the connection. Some of the parameters are taken from the choices made by a user. The algorithm finds the better combination among the different parameters which maximizes the user perceived quality according to the transient conditions of the network.

One of the most basic operations of the algorithm is that it manages to correlate the objective quality of service with the user perceived quality (MOS). This functionality is embedded on a table that combines the MOS with the different combinations of the parameters. Thus the algorithm can deduce how the user perceives the quality of the received video stream according to the values of the parameters shown in this table.

Moreover RAF algorithm can maintain high user perceived quality under extreme network conditions with high packet loss percentage by calibrating the value of frames per second.

Needless to say that both algorithms can be extended in order to take some more parameters into consideration like the computational power of the video streaming server and the video client as well, video client’s memory size or even its graphic card.

Another contribution in this project is the development of an Adaptive Congestion control Protocol (ACP) which is shown through simulations and analysis to satisfy all the design requirements as outlined in this deliverable and thus outperforms previous TCP proposals. ACP is a window based protocol which does not require maintenance pf per flow states within the network. It utilizes an explicit multi-bit feedback signalling scheme to convey congestion information from the network to the end users and vice versa. A distinct feature of the protocol is the implementation at each link of an estimation algorithm which is derived using on line parameter identification techniques. The algorithm generates estimates of the number of users utilizing the link which are used to tune the control parameters in order to maintain stability. This feature enables the protocol to adapt to dynamically changing network conditions. Extensive simulations indicate that the protocol is able to guide the network to a stable equilibrium which is characterized by max-min fairness, high utilization, small queue sizes and no observable packet drops. In addition, it is found to be scalable with respect to changing bandwidths, delays and number of users utilizing the network. The protocol also exhibits nice transient properties such as smooth responses with no oscillations and fast convergence. Apart from its practical significance, this work also demonstrates the effectiveness of formal control techniques in general and adaptive control techniques in particular in delivering efficient solutions in a highly complex networked system such as the Internet. Our next objective is to verify the properties of ACP analytically in networks of arbitrary topology.

In this project we also present a new Internet congestion control protocol whose objective is to regulate the queue size at each link so that it tracks a reference queue size chosen by the designer. Extensive simulations indicate that the protocol is able to guide the network to a stable equilibrium which is characterized by max-min fairness, high utilization, queue sizes close to the chosen reference value and no observable packet drops. In addition, it is found to be scalable with respect to changing bandwidths, delays and number of users utilizing the network. The protocol also exhibits nice transient properties such as smooth responses with no oscillations and fast convergence.

We demonstrate through simulations that the protocol meets its design objectives to a very good extent. Our next objective is to further evaluate its performance in more complex topologies and in the presence of realistic web-like traffic. We also aim at establishing its properties analytically in networks of arbitrary topology.

References

[1] Network Simulator – ns-2 site. http://www.isi.edu/nsnam/ns/.

[2] Fedora Project site. http://fedora.redhat.com/.

[3] Evalvid in NS2. http://140.116.72.80/~smallko/ns2/Evalvid_in_NS2.rar.

[4] C.-H. Ke, C.-H. Lin, C.-K. Shieh, W.-S. Hwang, "A Novel Realistic Simulation Tool for Video Transmission over Wireless Network," The IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC2006), June 5-7, 2006, Taichung, Taiwan.

[5] J. Klaue, B. Rathke and A. Wolish, "Evalvid – A Framework for Video Transmission and Quality Evaluation," http://www.tkn.tu-berlin.de/publications/papers/evalvid.pdf.

[6] FFmpeg Multimedia System site. http://ffmpeg.mplayerhq.hu/.

[7] J. Klaue. Evalvid – http://www.tkn.tu-berlin.de/research/evalvid/fw.html.

[8] Worcester Polytechnic Institute. "Research related to performance of networks, specifically congestion control and multimedia systems," http://perform.wpi.edu

[9] MPEG4encoder site. http://www.megaera.ee.nctu.edu.tw/mpeg/ Department of Electronics Engineering National Chiao-Tung University, Taiwan.

[10] V. Vassiliou, P. Antoniou, I. Giannakou and A. Pitsillides, "Requirements for the Transmission of Streaming Video in Mobile Wireless Networks," In Proceedings of the International Conference on Artificial Neural Networks (ICANN 06), Athens, Greece, September 10-14 2006.

[11] M. Neophytou, K. Stavrou, V. Vassiliou and A. Pitsillides "The Importance of Adaptive Applications in Mobile Wireless Networks, In Proceedings of SoftCom 2006.

[12] V. Vassiliou and A. Pitsillides, "QoS Adaptation Control in Virtual Circuit Switched Mobile Networks," In Proceedings of the International Conference on Intelligent Systems And Computing: Theory And Applications (ISYC) 2006, Ayia Napa, Cyprus, July 6-7, 2006.

[13] P. Antoniou, A. Pitsillides, and V. Vassiliou, "ADIVIS: An Adaptive Feedback Algorithm for Internet Video Streaming based on Fuzzy Control," Submitted to the IFIP Networking 2007, Atlanta, GA, USA, May 14-18, 2007.

[14] P. Antoniou, A. Pitsillides, and V. Vassiliou, "Adaptive Feedback Algorithm for Internet Video Streaming based on Fuzzy Rate Control," Submitted to the IEEE

Symposium on Computers and Communications (ISCC'07), Aveiro, Portugal, July 1-4, 2007.

[15] M. Lestas, A. Pitsillides, P. Ioannou and G. Hadjipollas, “Adaptive Congestion Protocol: A Congestion Control Protocol with Learning Capability,” Accepted for publication in Computer Networks Journal (subject to revision). An earlier version appears in UCY Report TR-06-03 CS UCY, 7 February 2006.

[16] M. Lestas, A. Pitsillides, P. Ioannou and G. Hadjipollas, “Adaptive Congestion Protocol: A Congestion Control Protocol with Learning Capability,” ISYC 2006, International Conference on Intelligent Systems and Computing: Theory and Applications, Ayia Napa, Cyprus, July 6-7, 2006, pp. 57-70.

[17] M. Lestas, A. Pitsillides, P. Ioannou and G. Hadjipollas, “Queue Length Based Internet Congestion Control,” IEEE International Conference On Networking, Sensing and Control (IEEE ICNSC07), London, United Kingdom, April, 2007 (to appear).

[18] D. Katabi, M. Handley, and C. Rohrs, “Internet Congestion Control for high-bandwidth-delay products,” In Proceedings of ACM SIGCOM, August 2002.

[19] S. H. Low, L. L. H. Andrew, and B. P. Wydrowski, “Understanding XCP: Equilibrium and Fairness,” In Proceedings of IEEE INFOCOM, Vol. 2, March 2005, pp. 1025-1036.