

TCP based-user control for adaptive video streaming

Yassine Douga¹  · Malika Bourenane¹ ·
Abdelhamid Mellouk² · Yassine Hadjadj-Aoul³

Received: 15 December 2014 / Revised: 2 June 2015 / Accepted: 3 August 2015 /

Published online: 15 August 2015

© Springer Science+Business Media New York 2015

Abstract Nowadays, Media streaming services over TCP have become very popular because of the TCP's reliability, which provides remarkable stability to the Internet. However, in order to offer a high media quality and a good user satisfaction, the media streaming service requires that transport protocols can be adapted continuously with the network parameters. However, the diversity, of terminals (i.e., tablet, smart phones, laptop ... etc.) and their corresponding capabilities, means that users' agnostic solutions are inefficient to cope with such diverse contexts. Indeed, the intrinsic characteristics and parameters of the terminal users (i.e., devices) need to be taken into account on the video streaming adaptation process. The classic adaptive video streaming services do not consider the user parameters on the adaptation process. In this paper, we propose an adaptive video streaming solution to improve the user satisfaction factor by adapting the TCP parameters according to the user's parameters on mobile networks. The user satisfaction factor is calculated according to some metrics driven from the user's quality of experience (QoE). The work is validated through our proposal based on a new mobile agent (which does all the work) developed on a Linux script platform and tested on different kinds of devices with different scenarios.

Keywords QoE · QoS · TCP tuning · Video streaming · User parameters · Terminal device · Multimedia

✉ Yassine Douga
maximussse@hotmail.com

Malika Bourenane
mb_regina@yahoo.fr

Abdelhamid Mellouk
mellouk@u-pec.fr

Yassine Hadjadj-Aoul
yhadjadj@irisa.fr

¹ LRIIR laboratory, Computer Science Department, University of Oran 1 Ahmed Ben Bella, Es-Sania Oran, Algeria

² LISSI Laboratory & Department of Networks and Telecoms, IUT CV, University of Paris-Est (UPEC), Créteil, France

³ IRISA Laboratory, INRIA Dionysos team-project, University of Rennes, Rennes, France

1 Introduction

Over the last few years, the widespread adoption of broadband connectivity, for both wired and wireless (mobile) networks, allows the effective functioning of intensive applications, such as multimedia applications; like video streaming, IPTV, and videoconference. These applications may use a very high resolution, which can reach 4096×2160 (4 K) with 60fps of frame rate. However, such rich media contents require adequate processing resources for decoding a compressed bit stream in the order of 10 Mbps [10].

On the first video streaming services, the video was delivered as any data using greedy TCP connections. The video stream was buffered at the receiver side for a while before the playing is started so that short-term mismatches between the video bitrate and the available network bandwidth can be absorbed, and video interruptions could be mitigated. Nevertheless, if the mismatch persists the buffer could eventually get empty and playback interruptions could occur affecting the user experience. To solve this problem, a new kind, of video streaming services, appears: it's the adaptive video streaming service in which the video source is available in different qualities, and the user can select dynamically the best quality matching its available bandwidth.

Adaptive Video Streaming services are becoming a commodity in mobile devices. With the recent popularity of smartphones, tablet, netbooks and laptops, the mobile use of streaming services is dramatically expanding. In fact, according to the study in [1], between 2009 and 2014, the mobile data traffic has grown by a factor of 40 and therefore expected to be more than double every year [9]. It can be clearly seen in Fig. 1 that the video traffic represents 66 % of the total amount of the mobile data [5]. To satisfy the users' needs, video streaming services have to improve their Quality of Experience (QoE) by enhancing the video quality, reducing the start-up delay, reacting rapidly to users' demands ... etc [11]. The idea of QoE has gained greater attention in network service provider [14]. Its main objective is not only to consider and evaluate the network Quality of Service (QoS), but also to keep it nearest to end users for better estimation of perceived quality of services [12].

In this paper, we focus on a particular adaptive streaming approach based on user-satisfaction and using TCP [15]. The proposed approach is based on the Dynamic and Adaptive Streaming over HTTP (DASH) standard for which we consider low-level feedbacks (i.e., metrics) for the switching from the current quality to the optimal one. The variation of the values of these feedbacks is directly related to the variation of TCP factors. For example the measured available bandwidth has a direct relation with the TCP sliding window. In addition to

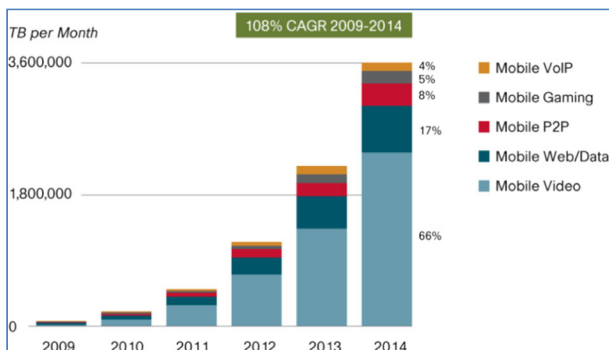


Fig. 1 Video Account for 66 % of Global Mobile Data Traffic by 2014 [1]

adapting the transmitted video bitrate with the network feedbacks, such as the measured available bandwidth, we adapt the TCP factors to the user parameters such as the terminal resolution, the CPU, and the power consumption. The main idea behind our contribution is to satisfy not only the network needs but also the users' satisfaction.

The Quality of Experience (QoE) is defined here as the measure of the overall acceptability of an application or a service, which is perceived subjectively by the end user. The users' profiles are very important metrics that influence the QoE [17].

The paper here proposes an extension of a previous contribution [4]. It proposes a reinforcement learning solution including a reward function in opposition to [16]. Moreover, in order to have realistic results we considered an emulation environment in which we measure the QoE using an important panel of users (around 100), in opposition to the previous work, which evaluated the QoE using a simulation environment and a less effective tool (i.e., PSNR).

The rest of the paper is organized as follows: Section 2 provides a brief review of the related works on adaptive streaming algorithms. In section 3, we formulate the studied problem. Section 4 describes the proposed idea and its advantages. The emulation scenarios and environment are described in the section 4, the emulation results are discussed in the section 5. Finally, Section 6 concludes the paper.

2 Related works

2.1 Adaptive streaming techniques

In the last decade, a vast literature on video streaming has been produced. These topics address generally one of the following issues: 1) the design of transport protocols specifically tailored for video streaming, 2) adaptation techniques, 3) scalable codecs.

Concerning the first topic, several transport protocols designed for video streaming have been proposed, such as the TCP Friendly Rate Control (TFRC), Real Time Streaming Protocol (RTSP), Microsoft Media Services (MMS) and Real Time Messaging Protocol (RTMP). Some of the mentioned protocols have been employed in commercial products such as Real Networks, Windows Media Player and Flash Player. Even though TCP has been regarded in the past as inappropriate for the transport of video streaming protocols, it is getting recently a wider acceptance and it is being used with HTTP. This interest is motivated by the following factors : i) Applications are beginning to be increasingly used across web browsers; ii) HTTP-based streaming services are cheaper to deploy since standard HTTP servers; iii) TCP has built-in NAT traversal functionalities; and iv) TCP is easy to deploy within Content Delivery Networks (CDN)). Thus, TCP, which already delivers most part of the Internet traffic, is able to guarantee the stability of the network while reaching all the connected devices.

In [3], the authors developed analytical performance models to assess the performance of TCP when used to transport a live video streaming source without the use of quality adaptation. The obtained results, which consider a constant bit rate (CBR) video source, suggested that in order to achieve good performance in terms of start-up delay and percentage of late packet arrivals, TCP requires a network bandwidth which is roughly two times the video bit rate [3]. It is important to stress out that such bandwidth over-provisioning would systematically induce a waste of half of the bandwidth.

Regarding the adaptation techniques, different approaches have been proposed in the literature, so far. The issue here is how to automatically adjust the video quality to match

the available resources (i.e., network bandwidth, CPU ...) so that the user receives the video at the best possible quality. The proposed techniques to adapt the video source bitrate to the variable bandwidth can be classified into three main groups: 1) a group transcoding on the fly the video content, 2) a group exploiting scalable video coding, and 3) a group, in which videos are pre-encoded into multiple bit rates (MBR), allowing a stream-switching.

The transcoding-based schemes involve the dynamic adaptation of the video content to match a specific bitrate by transcoding raw video data on the fly. These approaches can achieve customized bit rates/quality by selecting optimally the frame rate, the compression, and the video resolution. However, this comes at the cost of increased processing burden and poor scalability, when it needs to be done for each client. Besides, these techniques present high costs and deployment difficulties in CDNs environments.

Another important class of adaptation algorithms employs scalable codecs such as H.264/SVC. The main advantage of such approaches is to have the possibility to modify spatial, temporal and quality scalabilities without requiring to re-encode the raw video content. Thus, in opposition to transcoding-based approaches, scalable codecs reduce the computation complexity. However, SVC-based coding requires specialised servers implementing the adaptation logic. This presents some difficulties for their support into CDNs and to cache this type of content into proxies [3].

Stream-switching algorithms requires the segmentation of a media file into chunks of same duration. Chunks are proposed in several formats, corresponding e.g., to different coding rates, and are encapsulated in HTTP. The different clients request successive chunks from the server after estimating the bottleneck size between in the communication path and thus adapts the requests to the available bandwidth, by requesting the appropriate coding for each chunk. This enables seamless switching from one coding rate to another when network conditions change. Another important advantage of such algorithms is that they do not rely on particular functionalities and thus can be easily cached by proxies and supported by CDNs. Compared to SVC-based approaches, these approaches require an increased storage capacity [8]. Moreover, the granularity directly depends on the chunks' duration, which is generally around 10s (Figs. 2 and 3).

2.2 TCP tuning

TCP tuning, mainly, consists in regulating the network congestion avoidance parameters of TCP connections over high-bandwidth and high-latency networks. An optimal tuning of these parameters allows improving significantly the performance. On the other hand, some

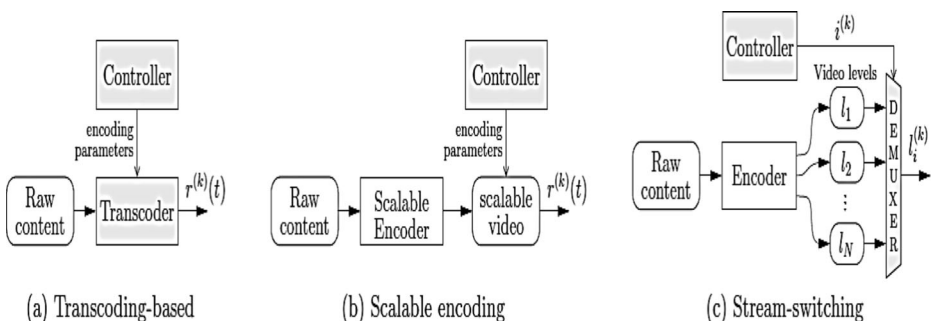


Fig. 2 Adapting streaming techniques

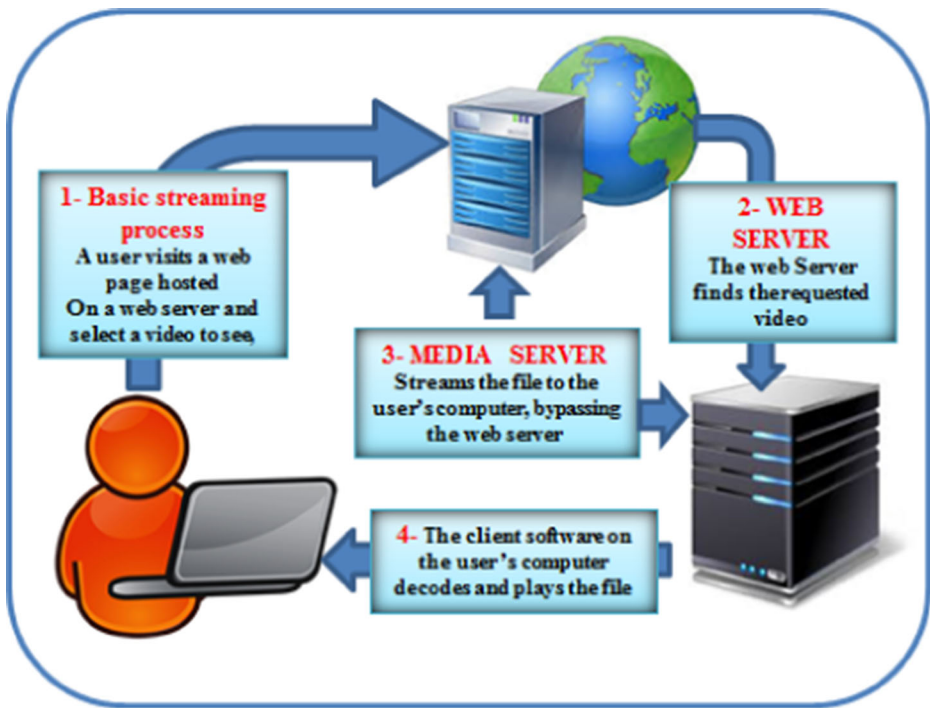


Fig. 3 Client–server video data transfer

configurations may degrade the TCP performance. In this way, there's a need for a smart mechanism for an efficient parameters' adjustment.

Note that the TCP tuning is highly related to the type of considered networks. Indeed, wireless networks, for example, experience high loss ratio while high-speed networks are characterised by no congestion and a difficulty to measure the available bandwidth.

TCP uses the congestion window (CWND), to decide how many packets can be sent without interruptions. The larger is the congestion window size, the higher is the throughput. The congestion window is determined using the algorithms the TCP “slow start” and the TCP “congestion avoidance”. The maximum congestion window size is, however, related to the amount of buffer space allocated for each socket by the kernel. A default buffer size, which can be modified using a system library, is initialized when initializing a socket. The optimal buffer size value is generally considered to be two times the bandwidth delay product. It is, thus, dependent of the delay between the source and the destination [1]. Indeed, too small buffers induce slow communications while too large buffers may involve the overwhelming of the receiver by the sender.

The most used parameters for TCP tuning are:

- TCP segment size: the maximum size of a single TCP segment.
- TCP Window Size: The data amounts the device can handle before it pass to the application.
- Number of sessions: the maximum number of available sessions.

- Number of TCP connections: the maximum number of available TCP connections.
The table below summarizes which parameters that may need to be changed on the Linux OS (Table 1):

3 Problem formulation

The adaptive video streaming services stream the video from a server to the user by taking into consideration only the state of the connection and essentially the available bandwidth. This may give some satisfaction to the user because it delivers the best available video according to user connection parameters, but this leads to a big consumption of the available bandwidth. As shown on the introduction, one of the advantages of the video streaming is that it can be used on several types of terminals with different characteristics, anywhere, any time and with any broadband Internet technology. These facts request to take in consideration some questions: Did the users really need the best video resolution in order to be satisfied and give a good QoE results? Did they really receive the best available resolution? Is it always necessary to give the best video resolution to different users with different kind of terminals just according to their network parameters?

The answers of all these questions are correlated by the fact that users do not always need the best available resolution to be satisfied because it may not match with the type of user terminal or it is just too high to be noticed between other lower video resolutions on some kind of terminal. So even if the users want the best available video resolution according to their network parameters, they do not sometimes receive it because their network parameters are not dynamically well adjusted.

The purpose of the desired solution is to solve the mentioned problem of received video resolution to answer the question about the best video resolution that will give the best QOE

Table 1 Linux commands for tuning the TCP parameters

Command	Description
echo "1" > /proc/sys/net/ipv4/tcp_window_scaling	Activate window scaling according to RFC 1323
echo "1" > /proc/sys/net/ipv4/tcp_timestamps	Activate timestamps according to RFC 1323
echo [wmax] > /proc/sys/net/core/rmem_max	Set maximum size of TCP receive window.
echo [wmax] > /proc/sys/net/core/wmem_max	Set maximum size of TCP transmit window.
echo [wmax] > /proc/sys/net/core/rmem_default	Set default size of TCP receive window.
echo [wmax] > /proc/sys/net/core/wmem_default	Set default size of TCP transmit window.
echo "[wmin] [wstd] [wmax]" > /proc/sys/net/ipv4/tcp_rmem	Set min, default, max receive window. Used by the autotuning function.
echo "[wmin] [wstd] [wmax]" > /proc/sys/net/ipv4/tcp_wmem	Set min, default, max transmit window. Used by the autotuning function.
echo "bmin bdef bmax" > /proc/sys/net/ipv4/tcp_mem	Set maximum total TCP buffer-space allocatable. Used by the autotuning function.
ifconfig eth? txqueuelen 1000	Define length of transmit queue. Replace "?" with actual interface number.

results and optimize the available bandwidth consumption of the users according to the their (terminal) parameters and more important make them satisfied with the service.

4 Proposal: a user-based adaptive streaming (UAS) mechanism

One of the greatest weaknesses of the adaptive streaming techniques is that they do not give enough importance to the user satisfaction (QoE) at least comparing to the network satisfaction (the quality of service (QoS)). In the proposed solution, we provide a new adaptive video streaming method. In this new proposal, in addition to what the techniques of the adaptive streaming services do in term of improving the QoS we add a new mechanism that will indirectly improve the user QoE by adapting the network parameters according to the user factors before executing the QoS techniques that are available on the adaptive video streaming services. Thus, the selection of broadcasting video rates will depend on the network parameters adapted to those of the user. To tune the network parameters, we have chosen to adjust some factors of TCP. Such adjustment has a direct impact on the network parameters. Thus, we have chosen the TCP sliding window to tune the available bandwidth.

The user parameters that we have considered are:

- The terminal resolution and screen size→ it is the first parameter that users want to switch and improve [13].
- The available battery→ one of the most important resources of mobile devices [7].
- The available bandwidth→ the most important parameter of the network.

First, we need to calculate the value of user factor (UF) which will characterize each user; this value is calculated in function of the terminal screen size, the terminal resolution and the terminal available battery. To do so, we gave to each parameter a weighting according to its quality and its importance. The Tables 2, 3 and 4 shows the used weighting for each parameters. These weighting values have been calculated empirically according to a statistic study that we have done on a hundred of students about the importance of the three listed parameters.

The global user factor (UF) is equal to the sum of the three user factors (FR, FB, FS), this value is a mark out of 20 where:

FR is the resolution factor.

FB is the battery factor.

Table 2 Value of FR (the resolution factor)

Video resolution (TR) in Pixel	FR=
$2160 \leq TR$	7
$1440 \leq TR < 2160$	6
$1080 \leq TR < 1440$	5
$720 \leq TR < 1080$	4
$480 \leq TR < 720$	3
$240 \leq TR < 480$	2

Table 3 Value of FB (the battery factor)

Battery life (BC) in %	FB=
40≤BC	6
20≤BC<40	4
0≤BC<20	2

FS is the screen size factor.

UF is the global user factor.

$$UF = FR + FB + FS. \tag{1}$$

To adapt the bandwidth to user parameters (screen resolution, screen size, available battery), we used two Tables 5 and 6: the first Table 5 gives to each global terminal factor (UF) a value of required video quality and the second one [VI] accord to each video quality a value of minimum required bandwidth.

The adaptation process is executed each 5 min due the fact that the user parameters can evolve continuously on time. In fact, the battery consumption is very variable on time and the user can change the display screen or its resolution.

In order to give more users satisfaction and better results, we equipped our solution with a reinforcement learning mechanism [2], this mechanism will updates the values of the Table 5 according to user’s feedbacks which is no more than a mark out of 5 which represent the user’s mean opinion score.

– Formalism of the Proposed Model:

We formulate the optimal adaptation problem as a finite Markov Decision Process (MDP), which can deal with the dynamic network conditions.

The proposed approach operates as follows. The video server initially sends out a description of video version. A user requests a segment based on local throughput measurements. Such request only serves as a preliminary suggestion and therefore no corresponding local optimization in the client is performed. We use a virtual interface (VI) and a mobile operator that acts in it for the network-assisted adaptation. The mobile agent re-formulates the request as a finite MDP to take into account the adaptation problem and forwards it to the video server.

We consider that for each of the N users there is a set of global factors indexed by $l=1, 2, \dots, L$. Hence, we called this set by $U = \{UF^l / i=1, 2, \dots, N \ \& \ l=1, 2, \dots, L\}$. There are also \forall a set of user feedbacks indexed by $j=1, \dots, M$ where $V = \{V^j / i=1, 2, \dots, N \ \& \ l=1, 2, \dots, M\}$.

Table 4 Value of FS (the screen size factor)

Screen size (SS) in inch	FS=
55≤SS	7
42≤SS<55	6
32≤SS<42	5
15≤SS<32	4
15≤SS<7	3
SS≤7	2

Table 5 Different video qualities assigned for each UF (User factor)

User factor (UF)	Video quality (VQ)
8	144p
9	240p
10	240p
11	360p
12	360p
13	360p
14	420p
15	720p
16	720p
17	720p
18	1080p
19	1080p
20	1440p

We propose a Markov Decision Process (MDP) based framework to carry out the adaptation decision, where we jointly consider the optimal bandwidth required by the user according of its global factors. We consider that each user has K states of available bandwidth and each state has a global user factor that goes with, namely $B = \{bw_k | k=1, 2, \dots, K\}$ where bw_k is the required bandwidth according of a user factor at step k and for everyone, the measured bandwidth (that may not be exactly the value of bw_k) is reported periodically to the Table 6, to serve as an input to the adaptation framework.

- Markov Decision Process Formulation

An MDP is a 4-tuple $(S; A; P; r)$ reinforcement learning task where S is the states set of the considered system, A is the actions set, $P(s)$ is the transition probability from $s_t \in S$ at time t to $s_{t+1} \in S$ at time $t+1$, when the action $a_t \in A$ is applied. Finally $r(s)$ is a reward function indicating the immediate reward obtained when applying a_t in state s_t . At each state in the environment, the agent perceives a numerical reward, providing feedback to the agent's actions. The agent's goal is to learn which action to take in a given state of the environment in order to maximize the cumulative numerical reward.

Table 6 The minimum required bandwidth for each video quality

Video quality (VQ)	Required bandwidth (RB) (Kbps)
144p	32
240p	57
360p	128
420p	245
720p	430
1080p	720
1440p	1057

The four components of the MDP are:

System States: The system state at time period t can be defined as $s_t = (VQ_t, UF_t^1, BW_t)$ where the vector represents the rate VQ (quality version) and the bandwidth limit BW_t that goes with the global user factor UF_t^1 at the instant t . Besides, the system state set $S = \{s_1, s_2, \dots, s_T\}$ essentially keeps the path of the evolution of the entire streaming system, where T is session duration. Since the system state depends only on its most recent (previous) state, the Markov property holds. We define the state space S as: $S = U^N \times B^N$.

System Actions: We define the sequential actions as $\{a_t\}$ for $(t=0, 1, \dots)$ where action a_t is the decision made at step t . The action set for a given state is $A(s) = \{A_{sbw}, A_{ubw}\}$ where A_{sbw} selects the bandwidth for a user according to his global factor UF_t^1 and closely as possible to VQ_{t+1} . Action A_{ubw} means to upgrade the last bandwidth to be considered on the upcoming adaptation period according to the user feedbacks.

This is done using the Tables 5, 6 and 7.

State Transition: The state transition from s_t to s_{t+1} is determined by user factor UF_t^1 at the instant t and the available bandwidth at time period t . The state transition probability can be obtained by the following formula:

$$\begin{aligned}
 P_{at}(S_t, S_{t+1}) &= Pr\{S_{t+1} \mid S_t, a_t\} \\
 &= Pr\{(UF_{t+1}^1, BW_{t+1}) \mid (R_t, BW_t), a_t\} \\
 &= Pr\{UF_{t+1}^1 \mid UF_t^1, UF_{t+1}^1 = a_t\} Pr\{BW_{t+1} \mid BW_t\}
 \end{aligned}
 \tag{2}$$

Where the term $Pr\{BW_{t+1} \mid BW_t\}$ can be obtained as follow:

Given the available bandwidth BW_t for downloading the current segment, we can estimate the probability distribution of the bandwidth BW_{t+1} for the next segment using the transition matrix of the Markov model. $Pr\{R_{t+1} \mid R_t, R_{t+1} = a_t\}$ is decided by the action policy.

Reward Function: Since the reward function is the fundamental guide for the RL agent to learn the desired policy, we want the reward function to be a measure for the QoE. The reward in MDP is the payoff obtained when a particular action is taken at a state. In our case, we only focus on the reward function which captures user satisfaction for accessible bandwidth values. Then,

$$r_{t+1} = R_{Q_t}(s_t = s) \tag{3}$$

where R_{a_t} maps the state to a reward when action a_t is taken, such that it assigns the value of the attribute ‘‘user satisfaction’’ according to adaptation parameters. Table 7 lists the rewards defined for different states.

Table 7 The values of rewards for each obtained value of MOS

MOS	Reward
1	20 %
2	10 %
3	0 %
4	-10 %
5	-20 %

The maximum reward is obtained when the rate VQ_{t+1} may satisfy the adapted bandwidth BW_{t+1} according to the user factors. Formellement this results in :

$VQ_{t+1} \leq BW_{t+1}$, where VQ_{t+1} is the quality video at step $t+1$ and $BW_{t+1} \in B$ is the estimated bandwidth of link. Then, the reward value is highest and in this case, the playback should be normal.

By giving minimum values to the reward when the adaptation is not well done, we can minimize playback interruption and authorize the playback to be smooth.

Note that the action A_{ubw} taken in state S will not change the state, i.e., the terminal state will not affect the decision process since the reward is 0.

Finally, we can formulate the TCP parameters adaptation problem as an optimization problem and we solve it by a dynamic programming based algorithm. The objective is to find an optimal policy $\pi(s)$ for the action taken at a state s_t such as the reward received in the long run can be maximized.

Provided that all components of the MDP are clearly defined, the optimum policy may be obtained on-line by RL. RL aims at estimating a good policy without requiring an accurate knowledge of the $P(s_{t+1} | s_t, a_t)$. There are several classes of on-line RL algorithms. A commonly used RL algorithm is Q-Learning [16] in which knowledge regarding both reward prospects and environmental state transitions are obtained through interaction with the environment. In Q-Learning, a Q-function is used to measure the quality of a state-action combination, based on the perceived rewards.

Let $Q^\pi(s_t, a_t)$ the state-action quality-function from s_t to the end state s_T :

$$Q^\pi(s_t, a_t) = Q(s_t, a_t) + \alpha \left[r + \gamma \max_{\alpha'} Q(s_{t+1}, \alpha') - Q(s_t, a_t) \right] \tag{4}$$

Where $\alpha \in [0; 1]$ and $\gamma \in [0; 1]$ are the learning rate and the discount factor respectively. The learning rate α determines to what extent the newly acquired information over rides the old information, while the discount factor is a measure of the importance of future rewards. We are dealing with an episodic streaming task, and then we can set γ to 1.

We compute the expected rewards for all transitions obtained from state S to s' while executing action a . We derive then, the optimal rewards by selecting the action achieving the highest expected rewards. The selected action is captured by the optimal policy π^* that should maximize the state-value function in the long run. Hence:

$$\text{Where} \quad \pi^*(s) = \arg \max_a Q^*(s_t, a_t) \tag{5}$$

$$Q^*(s_t, a_t) = \max_{\pi} Q^\pi(s_t, a_t) \tag{6}$$

Algorithm: Rate Adaptation Algorithm

Initialization

While $t \leq T$ **do**

For all possible $s_t \in S$ **do**

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right] \quad Q(s_{t+1}, a_{t+1}) \leftarrow Q(s_t, a_t)$$

$t \leftarrow t + 1$

Derive the optimal policy π^*

Algorithm 2: Bandwidth adaptation on the user side.

```

1- Begin
2- While [playing.video=1] do // user is watching a video.
3- {
4- Wait (300000); // execute operation each 5 min.
5- Get (TR,BC,SS) //get the terminal parameters.
6- //calculating the user facotor
7- UF=user_factor(TR,BC,SS,Tab_I[],Tab_II[],Tab_III[]);
8- // chose the video quality that goes with the user factor.
9- VQ=chose_quality(UF,Tab_V[]);
10- // chose the bandwidth that goes with the video quality.
11- RB=chose_bdw(VQ, Tab_VI[]);
12- // Set RB as maximum bandwidth of the output
13- // wireless interface.
14- Set (RB,Wlan0);
15-}
16- // after the end of the video the user gives his feedback.
17- Get(user_feedback);
18- // update the tables 'V, VII' according to user
19- // feedback using reinforcement learning functions.
20- Update (Tab_V[],user_feedback);
21- Update (Tab_VI[], Tab_VII[], user_feedback);
22- end.

```

5 Emulation

5.1 Emulation environment

To validate our work, we have chosen to develop Linux scripts to be executed on different kinds of terminals. Three types of terminals are considered in our experiments: a smart TV, a laptop and a smartphone. Each device's characteristics are described on the Table 8 (Table 9).

Note that the proposed emulation environment can be considered with any type of video streaming server. However, we have chosen to use the YOUTUBE server. Indeed, YouTube is one of the fastest-growing websites, and has become one of the most accessed sites in the Internet. It has a significant impact on the Internet traffic distribution, but itself is suffering from severe scalability constraints and quality of service. Understanding the features of YouTube is, thus, crucial to network traffic engineering and sustainable development of this new generation of services [6].

5.2 Emulation scenario

To receive the required video quality, which is adapted to the user's parameters and his feedbacks, the script works according to these steps:

- 1- Create a virtual interface where all the video streaming flows will be routed through this interface to the principal interface.

Table 8 Parameters of the devices used in the emulation

Device	Screen size	MAX screen resolution	Battery life (watching a video)	CPU
Smart TV	107 cm	1080p	–	–
Laptop	39,6 cm	1080p	4 h30	2.9 Ghz
Smart phone	12.19 cm	720p	7 h45	–

- 2- The script computes the global user factor and the bandwidth that goes with this factor.
- 3- Set the computed bandwidth that goes with the user parameters, like maximum supported bandwidth of the virtual interface.
- 4- Update the computing tables according to the user feedback after the end of the watched video.

By doing these steps, the script limits the bandwidth of the output video streaming flows according to the user parameters and his feedback, this will optimizes the consumption of the bandwidth and of the battery while keeping a good user satisfaction (Figs. 4 and 5).

To test and compare our solution we considered the four, following, scenarios:

- 1- First scenario:

We consider here the analysis of the available bandwidth when the user is watching a video on YOUTUBE. The three different types of devices were considered. To measure the available bandwidth we used the “*Speedtest-cli*” tool, which is available on Linux to calculate this metric.

- 2- Second scenario:

We focused here on the battery consumption while watching a video on the Laptop.

- 3- Third scenario:

The CPU activity evolution is measured while running the studied solutions separately on a laptop with the same video and network conditions.

- 4- Fourth scenario:

We have done here a set of statistics at the laboratory level by considering a panel of 100 students. The same YOUTUBE video was requested by the students, which use the three kinds of devices introduced in Table 8. The users evaluated the video quality using a Mean Opinion Score (MOS) scale; from 1 (very bad) to 5 (excellent). Finally, we have averaged the different marks in a per-device basis.

Table 9 Emulations description

Emulation	User bandwidth	User screen resolution	Available battery	CPU
Figure 6	2 Mbps	variable	100 %	Not constrained
Figure 7	2 Mbps	1080p	Variable	Not constrained
Figure 8	2 Mbps	1080p	100 %	Not constrained
Figure 10	2 Mbps	variable	Variable	Not constrained

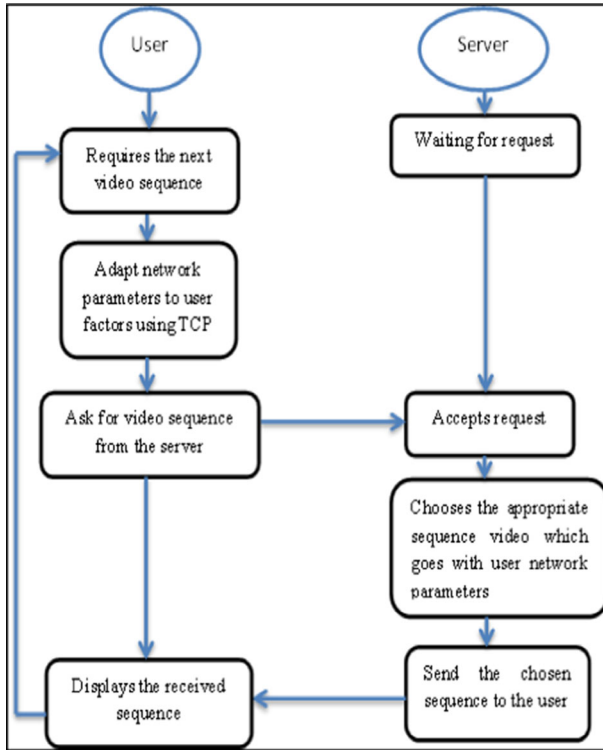


Fig. 4 Datagram of the proposed algorithm

To assess the efficiency of the proposal, we performed a comparative test using all the scenarios, introduced below, with a classical approach, where the proposed add-ons are disabled.

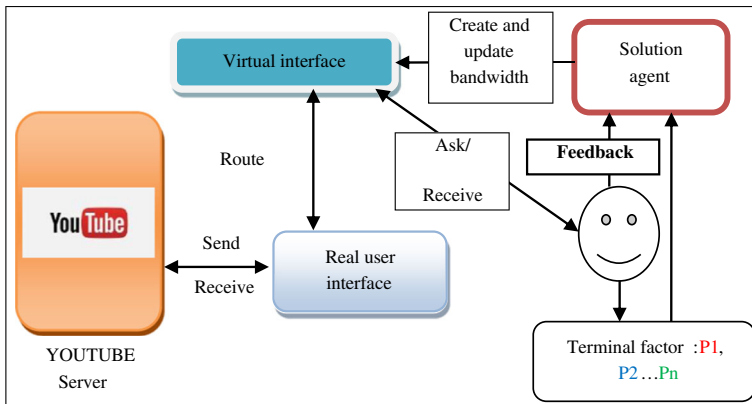


Fig. 5 Emulation scenario

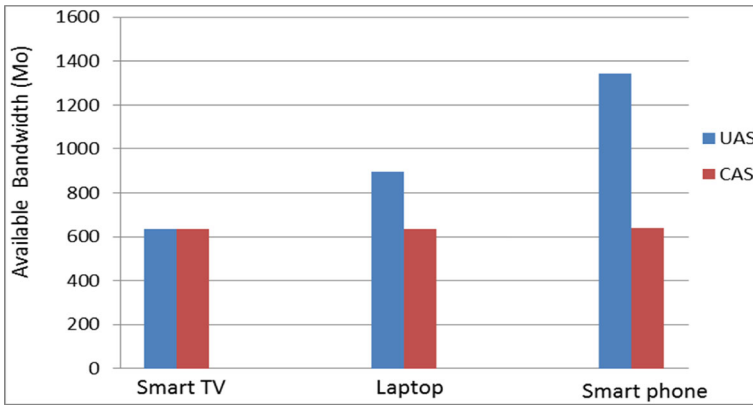


Fig. 6 Average available bandwidth

6 Results

The Table 4 describes all the initial conditions of the emulation:

In the following, CAS refers to the classical approach of streaming, while the UAS represent the proposed streaming solution.

7 Emulation discussion

We note, from Fig. 6, that varying the end terminals when considering the classical approach (i.e., UAS) when we watch the same video from YOUTUBE with 2 Mbps of bandwidth in different kinds of terminals having the same conditions, the bandwidth consumption and the available bandwidth are the same on all the devices. Indeed, all the devices received the same video quality. In our proposed solution

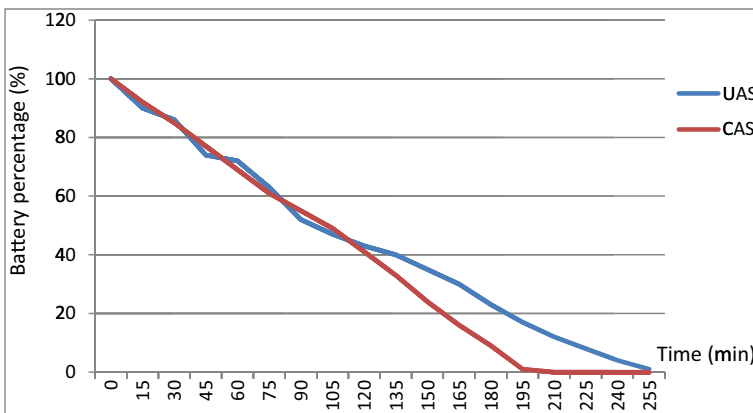


Fig. 7 Battery consumption

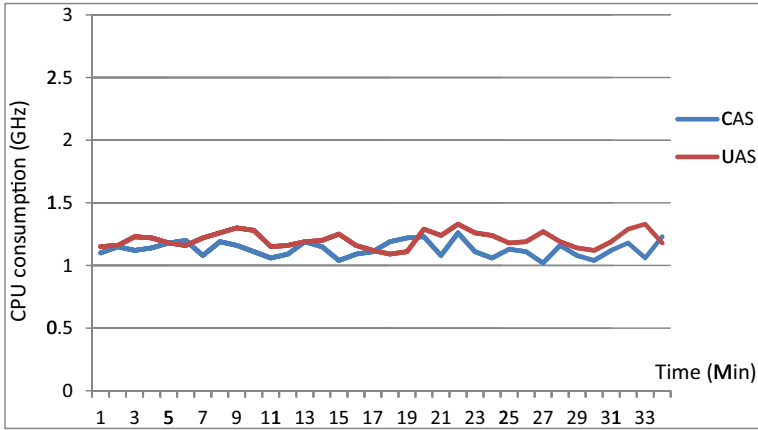


Fig. 8 CPU consumption

we note that the bandwidth consumption is not the same on all the devices. The more the device is small and less powerful, the more it receives lowest video quality. This has a direct consequence on the bandwidth consumption, which increases the available bandwidth.

Figure 7 compares the evolution of the battery consumption on the same device when considering the two approaches. It shows that the battery consumption of both studied methods is very close when the percentage of the consumed battery is higher than 40 %. However, from that threshold the battery consumption of the proposed solution will decrease in comparison to the other method. This is due to the fact that the proposed solution will decrease the received video quality when the battery percentage is lower than 40 %, which will decrease the battery consumption. In fact, the more the video quality is low, the more the battery consumption is decreased.

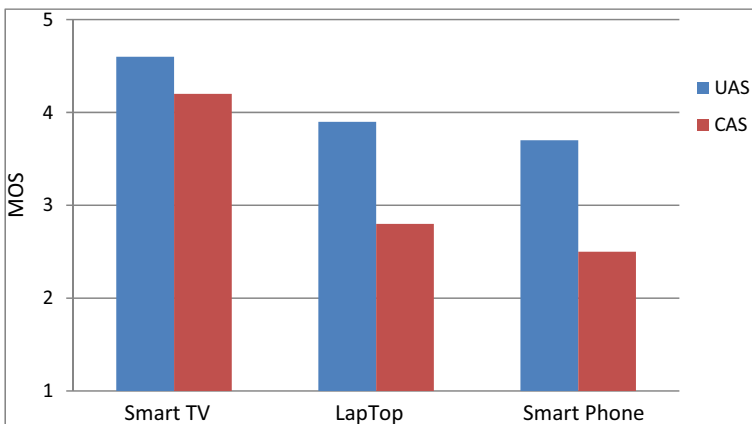


Fig. 9 Mean opinion scores

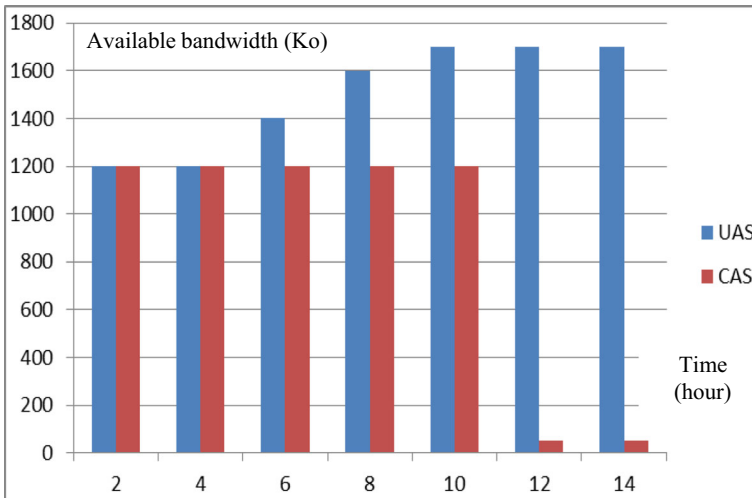


Fig. 10 Evolution of the Available bandwidth in the time

The evolution of the CPU activity on Fig. 8 shows that the proposed solution consumes a little bit more CPU than the other method. This consumption is mainly due to the calculation and the update of the user factor, which is not needed in the classical approach. This is, also, due to the routing of all the video streaming flows through the virtual interface.

The average results of the MOS that are shown on Fig. 9 show how far our solution improves the user satisfaction in term of available bandwidth, battery consumption and received video quality on different kind of devices. This improved QoE is a direct consequence of the proposed adaptation approach, which adapts to the user’s communication parameters and the received video quality (Figs. 10 and 11).

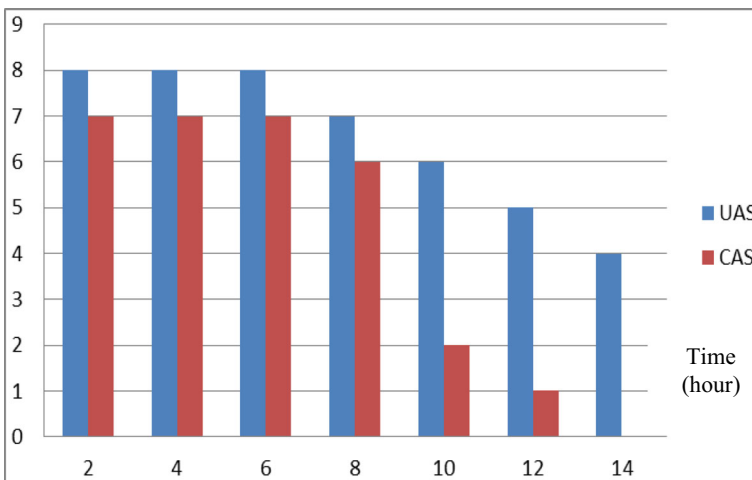


Fig. 11 QoE results according to the time

8 Conclusion

In this paper, we proposed a new approach for video streaming adaptation. Our main idea is to adapt the video according to the user's parameters. The emulation results have shown a good QoE/QoS performance and an improved user satisfaction. To implement our idea, we considered the transport protocol layer TCP as a bridge between the user parameters and the network parameters. As a perspective, we plan to integrate additional parameters (from the user and from the transport layer) into the adaptation process to improve the QoE metrics.

References

1. Cisco White Paper: Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2009–2014, <http://bit.ly/bwGY7L>
2. Claey's M, Latré S, Famaey J, Wu T, Leekwijck W, Turck F (2013) Design Of A Q-learning-based client quality selection algorithm for Http adaptive video streaming. In Proc Of Adaptive And Learning Agents Workshop, USA
3. Deep Singh K, Hadjadj-Aoul Y, Rubino G (2012) Quality of experience estimation for adaptive Http/Tcp video streaming using H.264/Avc, Cenc"- Ieee Consumer Communications & Networking Conference Hal-00713723, Version 1
4. Douga Y, Bourenane M, Mellouk A (2014) Adaptive video streaming using tcp factors control with user parameters, proceeding of elsevier fnc'14, the 9th international conference on Future Networks and Communication, volume 34, 2014, pages 526–531, Niagara Falls, Canada
5. Fiadino P et al (2014) On the detection of network traffic anomalies in content delivery network services. In ITC 26
6. Haddad M, Altman E, El-Azouzi R, Jiménez T, Elayoubi S, Jamaa S, Legout A, Rao A (2011) A survey on youtube streaming service. In Proc Of The 5th International Iest Conference On Performance Evaluation Methodologies And Tools, Belgium
7. Hossfeld T, Seufert M, Hirth M, Zinner T, Tran-Gia P, Schatz R (2011) Quantification of YouTube QoE via crowdsourcing. In IEEE International Symposium on Multimedia (ISM), pages 494–499
8. IstvánKetykó, De Moor K, De Pessemier T, Verdejo AJ, Vanhecke K, Joseph W, Martens L, De Marez L (2010) QoE measurement of mobile youtube video streaming. MoViD '10 Proceedings of the 3rd workshop on Mobile video delivery, Pages 27–32, ISBN: 978-1-4503-0165-7, ACM New York, NY, USA
9. Kim HJ, Yun DG, Kim H-S, Cho KS, Choi SG (2012) Qoe assessment model for video streaming service using qos parameters in wired-wireless network. In Advanced Communication Technology (ICACT), 2012 14th International Conference on, pages 459–464
10. Linck S, Mory E, Bourgeois J, Dedu E, Spies F (2014) Adaptive multimedia streaming using a simulation test bed. J Comput Sci. doi:10.1016/j.jocs.2014.02.004
11. Mellouk A, Cuadra Sanchez A (2014) Quality of experience engineering for customer added value services: from evaluation to monitoring, Wiley Ed./ISTE Publishing Knowledge, 12 Chapters (272 pages), ISBN: 9781848216723
12. Mellouk A, Hoceini S, Tran HA (2013) Quality of experience for multimedia, Wiley Ed./ISTE Publishing Knowledge, ISBN: 9781848215634
13. Metri G, Shi W, Brockmeyer M, Agrawal A (2014) Battery extender: an adaptive user-guided tool for power management of mobile devices, Ubicomp'14, Spetember 13–17, Seattle, WA, USA
14. Mushtaq S, Augustin B, Mellouk A (2014) QoE: user profile analysis for multimedia services. In Proc. Of the IEEE International Conference on Communications (ICC), Sydney, Australia, pp. 1–6
15. Stewart L, Hayes DA, Armitage G, Welzl M, Petlund A (2011) Multimedia-unfriendly tcp congestion control and home gateway queue management. Mmsys '11 Proceedings Of The Second Annual Acm

Conference On Multimedia Systems, Pages 169–174, Isbn: 978-1-4503-0518-1, Acm New York, Ny, Usa

16. Sutton RS, Barto AG (1998) Reinforcement learning: an introduction (Adaptive Computation and Machine Learning). The MIT Press
17. Truong T-H, Nguyen T-H, Nguyen H-T (2012) On relationship between quality of experience and quality of service metrics for IMS-Based IPTV networks. In IEEE International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF), pages 1–6



Yassine Douga is Ph.D student in computer science University Es-Senia Oran Algeria. He is member and researcher in the LRIIR laboratory, his main research topics are: wireless networks, WSN, network protocols, video streaming, QoE, QoS. He is also a member of IEEE Algeria.



Malika Bourenane received her diploma (doctorate of state) in computer science from University Es-Senia Oran Algeria. Since 1984 she has been a permanent teacher at department of computer science. She is a researcher in the computer science laboratory and her main research interests include mobile ad-hoc networks, quality of service networks, queuing and scheduling theory, dynamic routing and Networks policy-based management and Real Time Application. She is member of international scientific committees of international conferences and journal.



Abdelhamid Mellouk is Full Professor at University of Paris-Est C-VdM (UPEC), Networks & Telecommunications (N&T) Department and LiSSi Laboratory, France. Head of several executive national and international positions, he was the founder of the Network Control Research activity at UPEC with extensive international academic and industrial collaborations. His general area of research is in adaptive real-time control for high-speed new generation dynamic wired/wireless networks in order to maintain acceptable Quality of Service/Experience for added-value services. He is an active member of the IEEE Communications Society and has held several offices including leadership positions in IEEE Communications Society Technical Committees.



Yassine Hadjadj-Aoul is currently working as an associate professor at the University of Rennes 1, France, where he is also a member of the IRISA Laboratory and the INRIA project-team Dionysos. He received a B.Sc. in computer engineering with high honours from Mohamed Boudiaf University, Oran, Algeria, in 1999. Dr. Hadjadj received his Master's and Ph.D. degrees in computer science from the University of Versailles, France, in 2002 and 2007, respectively. He was an assistant professor at the University of Versailles from 2005 to 2007, where he was involved in several national and European projects such as NMS, IST-ATHENA, and IST-IMOSAN. He was also a post-doctoral fellow at the University of Lille 1 and a research fellow, under the EUFP6 EIF Marie Curie Action, at the National University of Dublin (UCD), where he was involved in the DOM'COM and IST-CARMEN projects, which aim at developing mixed Wi-Fi/WiMAX wireless mesh networks to support carrier grade services. His main research interests concern the fields of wireless networking, multimedia streaming architectures and protocols, congestion control protocols and QoS provisioning, and satellite and space communications. Dr. Hadjadj has been on the technical program committee of different IEEE conferences, including Globecom, ICC, VTC, PIMRC and IWCMC. His work on multimedia and wireless communications has led to more than 40 technical papers in journals and international conference proceedings.