Enhancement of channel switching scenario and IGMPv3 Protocol Implementation in Multicast IPTV Networks

Benoit Hilt[1], Mounir Sarni[1, 2] and Pascal Lorenz[1]

[1] MIPS/GRTC
Université de Haute Alsace, France
[benoit.hilt, mounir.sarni, pascal.lorenz]@uha.fr

[2] Cable Departement
Vialis, France
m.sarni@calixo.net

***Abstract*—Nowadays, delivering television over IP technologies is increasingly used by Internet Service Providers. Having using for years the traditional TV broadcasting systems, viewers expect at least the same Quality of Experience from IPTV providers. In this paper we focus on the zapping time which is one of the most important elements for the quality experienced by the viewers. To reduce it, we propose to invert the leaving and joining operations which are traditionally used in this order. As a second step, we modify the source code of the IGMPv3 protocol implementation to leave and join IPTV channels by sending a unique message. To evaluate our solution in terms of bandwidth overhead and blackout time we conduct simulations using IGMPv2 and IGMPv3 signalling.**

We show that with our proposition, in each case, the overhead stays limited in the network core. Moreover, the overhead tends to decrease when the number of the active viewers increases. Additionally, the proposed approach reduces the blackout time during a zapping process.

Keywords-component; zapping time, IGMP protocol, IPTV, bandwidth demand, channel overlapping.

I. INTRODUCTION

As the cable operator we talked about in [1], many Internet Service Providers (ISP) propose video services over IP technologies which are commonly known as IPTV (Internet Protocol Television). Unlike Internet TV [2], IPTV channels are delivered by the ISP in their own networks. An IPTV network architecture is usually set up as presented in Fig. 1. The core of the network contains an IPTV Broadband and routers. This core is connected to the clients by some active equipments which depend on the last mile technologies deployed by the ISP, such as Fiber To The Home (FTTH) technologies or Digital Subscriber Line (DSL) technologies. Finally, a Set Top Box (STB) or a Computer Software Solution allows the viewer to select and watch the channels on the TV or computer screen.

Assuming that most viewers are watching the same channels (the most popular ones), multicast streams seem to be the best way to deliver IPTV services and effectively manage the bandwidth demand of viewers. Two protocols are used to deal with multicast IPTV streams. In the core network side, a multicast routing protocol like Protocol Independent Multicast–Sparse Mode (PIM-SM) [3] is used. It allows building distribution trees for several groups (or IPTV channels) from the source (the IPTV Broadband) to the receivers' through routers and switches. As for last mile, Internet Group Management Protocol (IGMP) [4, 5] is used by the IPTV devices to leave or join multicast IPTV streams.

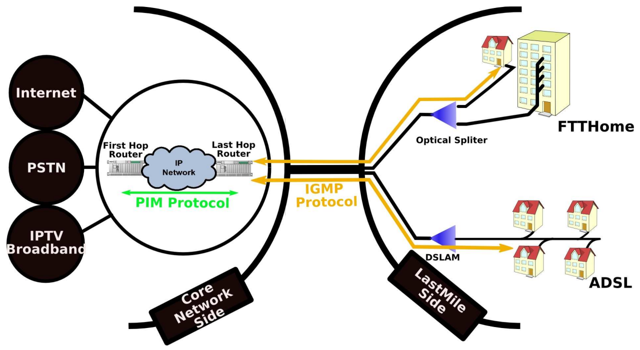

Figure 1: Typical ISP Network architecture

The use of these protocols to manage multicast streams may introduce a long network delay. This can impact the Quality of Experience (QoE) of how viewers, who have for years watched channels delivered from traditional broadcasting services (terrestrial, cable or satellite). As presented in (a) of Fig. 2, unlike the IPTV systems, in the traditional broadcasting systems, all channels are available at the user side regardless if they are requested or not. In a multicast IPTV network, only the requested channels are delivered to the viewers. So, as presented in (b) of Fig. 2 zapping delay is inevitable when viewers switch from one channel to another.

It becomes then clear that channel zapping time is one of the most important Quality of Experience parameters to be performed to deliver IPTV services in best conditions and satisfy costumers.

In the following of this section, extending the work presented in [1], we describe how we enhance channel switching scenario and IGMP protocol to efficiently switch between IPTV channels and reduce the network delay. Section II of this paper provides a background and describes some related works to reduce channel zapping time. Section

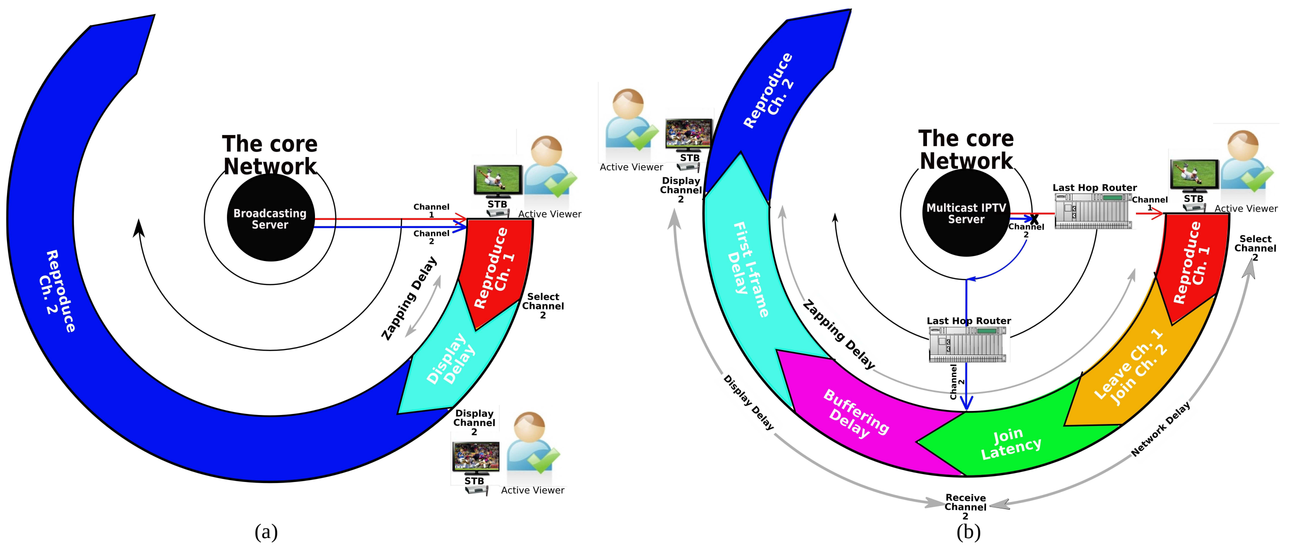

Figure 2: Broadcasting service system architecture: (a) the traditional TV broadcasting service and (b) the IPTV service

III presents our solution to improve the zapping time and section IV presents the developed multicast model with which we evaluate the bandwidth demand in the standard and the proposed approach. Section V is an overview of the IGMPv3 source code modifications made to merge the IGMP leave and IGMP join message to one and in Section VI, we show the result of our simulations. Finally, we conclude and present future work in section VII.

II. BACKROUND AND RELATED WORK

As represented in Fig. 3, when a viewer requests a new channel, its IPTV device sends out two IGMP messages to switch between channel regardless of the IGMP version used.

When a client wants to watch a new channel he or she pushes the remote control button. After a processing delay, the IPTV device sends an IGMP-Leave message for channel being left (channel#1). When the access router receives this message, it must check, by sending an IGMP Group-Specific-Query message, if there is an IPTV receiver remaining in the network which still wants to receive channel#1. After a Last-Membership-Query-Interval which is typically set to 1 second and if the Last-Member-Query-Count is equal to 1, the access router will stop forwarding data of channel#1. If the Member-Query-Count is equal to n $(n\geq 1)$, the access router will repeat the leaving process n times, as long as no response is received. The time interval between the sending of the IGMP-Leave message and the receiving of the last multicast packet of channel#1 is called the Leave Latency (*LL*).

After a delay we call the Channel Switch Delay (*CSD*), the viewer's IPTV device sends out an IGMP-Join message for the requested channel (channel#2). If this channel is not yet available on the access router, this IGMP join message is forwarded through a PIM-Join message toward the first router having the requested channel flow available. The time interval between the sending of the IGMP-Join message and the receiving of the first multicast packet of channel#2 is called the Join Latency (*JL*) of channel#2.

Depending on *LL*, *CSD*, and *JL* values (when $LL>CSD+JL$), channel#1 could overlap with channel#2 during the Channel Overlap Delay (*COD*).

We can define the Network Delay (*ND*) as the time interval between the sending of the IGMP-Leave message for channel#1 and the receiving of the first multicast packet of channel#2. Finally, after the buffering and the decoding delay [6], the viewer can watch the requested channel on his or her computer or television screen.

This zapping scenario can be played out with version 2 of IGMP protocol which is defined in RFC 2236 [4] or with version 3 which is defined in RFC 3376 [5]. This last version has additional capabilities, but during a zapping process, the main difference with IGMPv2 is that in IGMPv3 the leave and join messages are sent to the ALL-IGMPv3-Routers multicast address (224.0.0.22) and not to the requested channel group address as in IGMPv2. This allows the router to make explicit tracking to maintain an up to date receivers and groups' lists. Therefore in the case of an IGMPv3-Leave message, if no more groups' members are registered, the router can immediately stop sending the channel data flow. This can greatly reduce the probability of having a channel overlap. However, even with IGMPv3 protocol, for code implementation reasons, two messages are still needed to achieve a zapping process.

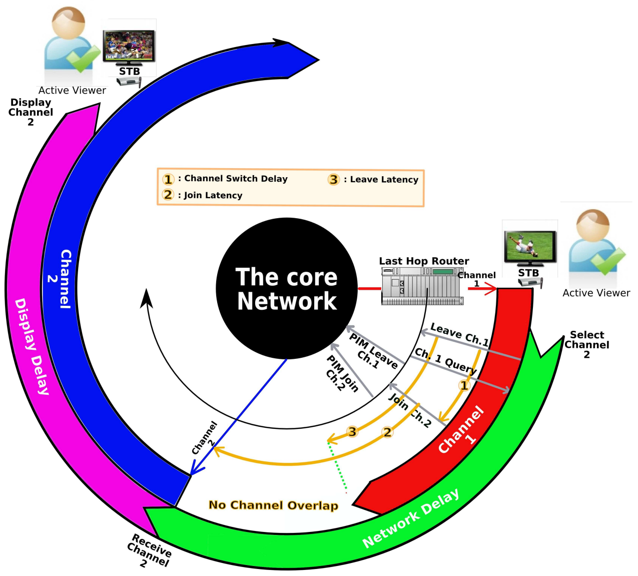

Figure 3: Standard channel switching scenario (IGMPv2, IGMPv3)

Based on the explained IPTV channel switching scenario, several ways were explored to improve channel zapping time. Some of them focused on the improvement of the Network Delay [7, 8], some others tried to reduce the zapping time by reducing the Display Delay [9, 10, 11, 12, 13, 14, 15, 16] and others tried to reduce the zapping time by predicting the behavior of the viewers during surfing [17, 18, 19, 20, 21].

A. Reduction of the Network Delay

In [7], the authors propose to reduce some IGMPv2 parameters like the General-Query-Interval (GQI) to reduce the Join Latency if the first report or join messages are lost. Over a Wavelength Division Multiplexing based Passive Optical Network (WDM-PON), they prove that when they reduce the GQI and set the Last-Membership-Query-Interval to 100 ms to quickly drop the left channel, the join delay can be reduced up to 100 ms while in the standard case it approaches 500 ms.

D. E. Smith proposes in [8] to send users unicast streams at higher than usual rates when surfing happens during commercial breaks. But even if the delay to build multicast distribution trees is avoided, the illustration of the developed model shows that this approach will highly increase the bandwidth demand by two the steady state.

B. Reduction of the Display Delay

To reduce the Display Delay by reducing the buffering delay, the authors propose in [9] to add a secondary multicast stream replicated from the main multicast stream with a constant delay. During a zapping process the viewer's IPTV device must join both streams until the initial buffer is filled up. Then, the secondary stream is left. With a developed packet ordering rule for the secondary stream and a multiplied transmission rate *r*, the authors confirm a maximum reduction of 1.1 second of the zapping delay during a commercial break and a maximum zapping delay equal to 2.1 seconds.

In [10] a Multicast Assisted Zap Acceleration is presented. The aim of this method is to reduce the First I-Frame Delay (*FID*) by adding time-shifted sub-channels of the multicast mean channel. When a zapping process happens, and depending on which time the viewer requests the new channel, one of those sub-channels is considered as the main stream. Additionally, Meta-Channel that specifies the sub-channel to be chosen by the STB when a channel is requested is constantly broadcasted by the zapping accelerator. A migration solution from the sub-channel to the main IPTV channel is proposed too. Regardless of the number of the viewers, the simulations results show that this solution reduces the waiting delay of the first I-frame without requiring additional resources.

U. Jennehag and T. Zhang propose in [11] the Synchronization Frames for Channels Switching (*SFCS*) method to decrease the decoding process delay by adding a secondary stream with which resynchronization frames are sent. Viewers who want to decode the IPTV channel must join both streams. This will avoid the *FID* but will significantly increase the bandwidth utilization.

To reduce the *FID*, authors propose in [12] to change the encoding structure by adding periodically to the normal video frames additional I-frames encoded at lower bit rate. Because of these additional I-frames, the decoding process will be faster but as in [11] the bandwidth utilization is increased.

Multicast Instant Channel Change (*Multicast ICC*) method is proposed in [13]. A low bit rate multicast stream carrying only I-frames is associated to each IPTV channel. In each zapping process, the viewers must join and display first of all this secondary stream and then, when the play out buffer is filled up with the mean stream, the full quality video stream is displayed. Unlike the Unicast Instant Channel Change (*Unicast ICC*) explained in [14], this method reduces the *FID* without increasing the resource needs.

In [15], a channel control algorithm is proposed to determine the number of extra I-frames to put in the mean stream. The aim of this algorithm is to pursue an effective trade-off between the decoding delay and the bandwidth utilization.

Scalable Video Coding (SVC) was proposed in [16] to reduce the decoding delay by embedding a secondary stream in the normal stream which will reduce the bandwidth overhead compared to the MPEG coding and transport system.

C. Viewers surfing behaviors

Methods based on pre-join mechanisms are studied in [17, 18, 19]. Based on the currently watched channel, and assuming that most users use the up/down button of their remote control to surf, adjacent channels can be joined in advance [17] or sent by the IPTV Head End in low resolution [18]. In [19], the authors propose to also pre-join the most popular channels, assuming that most viewers watch only the most popular channels, but all these schemes cannot fit with each user's preferences. In [20], a new method is proposed to reduce the channel zapping time by reflecting, for each viewer, the channel surfing behavior and the preferences, based on the pushed buttons of the remote control and the program preference of each viewers. To know the viewer's program preferences, a personalized recommendation system for Electronic Program Guide (*EPG*) installed in the STBs is proposed in [21].

This analysis shows that improving channel zapping time requires several actions. Besides the pre-join methods based on viewers' behaviors and preferences during channel surfing, solutions that reduce the Network Delay can be mixed with the propositions explained above to reduce the Buffering and the First I-frame Delay. However, all those solutions refer to the same channel switching scenario, which could be optimized.

In this paper, we propose for each version of the IGMP protocol, a novel channel switching scenario to reduce the Network Delay which stays compatible with the existing solutions presented above. With simulations, we evaluate the impact of our proposition on the network in terms of bandwidth consumption and user Quality of Experience (blackout time).

III. Proposed approach

A standard channel switching scenario consists in leaving first the currently watched channel and then joining the requested channel regardless of the IGMP protocol version used. The Network Delay is then expressed by:

$$ND_{Standard\ Approach} = CSD_{IPTV\ device} + JL_{Requested\ channel}$$

The Join Latency (*JL*) depends on several parameters like the processing capacity of the equipments used by the IPTV service provider , the number and the bit rate of the IPTV channels and the values of the IGMP protocol parameters (Last-Membership-Query-Interval, Last-Member-Query-Count and General-Query-Interval). In [7], over a WDM-PON network, the Join Latency approaches 100 ms in the best cases.

The Channel Switch Delay (*CSD*) depends only on the hardware/software implementations of the used IPTV device for closing the processing of the old channel, and opening a new socket to join and start the processing of the requested channel [16]. Table I summarizes the *CSD* measurements of some STB from different manufacturers and some IPTV providers software solutions [22, 23]. The measured values vary from 20 ms to 200 ms. We clearly conclude that the *CSD* may increase the channel zapping time, especially when viewers use computer software solutions.

TABLE I
MEASURED CSD VALUES

Name	Type	Values
NetGem	Set Top Box	20 ms
Aminet110	Set Top Box	40 ms
BeePlayer	Set Top Box	50 ms
VLC0.8.6h *(Linux Debian)*	computer software solution	100 ms
MyFreeTV *(Windonw XP)*	computer software solution	200 ms

To eliminate the Channel Switch Delay, we propose to modify the channel switching scenario for each version of the IGMP protocol.

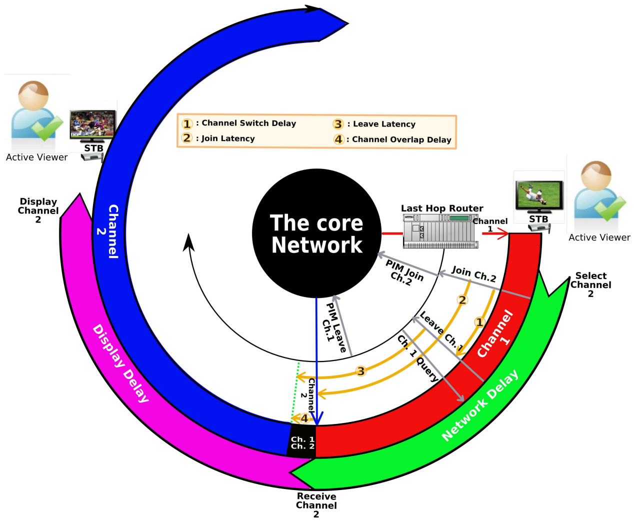

Figure 4: Proposed channel switching scenario with IGMPv2 protocol

With IGMPv2 protocol, as it's presented in Fig. 4, we suggest to join at first the requested channel and then to leave the currently watched channel. This proposal does not need any modification of the implemented protocol but requires only modifying the embedded software of the IPTV device. Note that in this case, unlike in the standard approach, Channel Switch Delay becomes the time interval between the sending of the IGMP-Join message and the IGMP-Leave message.

With IGMPv3 protocol, we propose to send a single message to switch between channels, as it's presented in Fig. 5. Despite the fact that IGMPv3 protocol can natively manage such a message, some code modifications are needed to make the IGMP Application Program Interface suitable to send a unique IGMP-Report message to switch between channels. These code modifications are presented in section V.

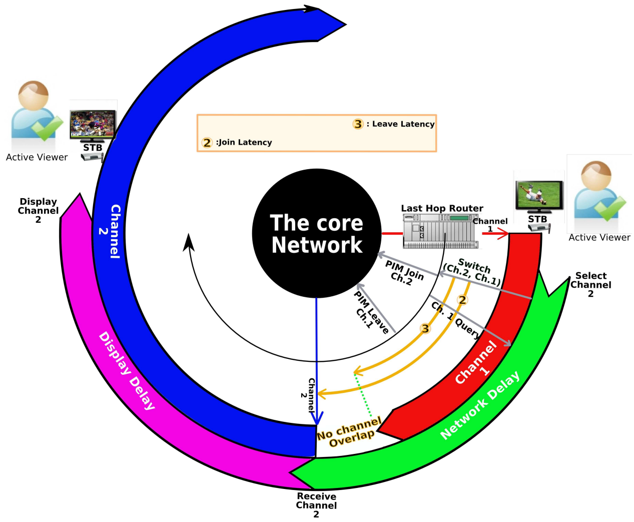

Figure 5: Proposed channel switching scenario with IGMPv3 protocol

So in both cases, regardless of the IGMP protocol version used, in light of our proposal, the Network Delay will be reduced to the Join Latency of the requested channel and therefore, can be expressed by:

$$ND_{Proposed\ Approach}=JL_{Requested\ channel}$$

If for example, as in [7], the Join Latency is equal to 100 ms, based on the values of Table I our proposal can reduce the Network Delay from 17% up to 67 %.

If we now focus on the channel overlap, as presented in Fig. 6, for the same *JL*, *CSD* and *LL* values, the Channel Overlap Delay may be greater in our proposed approach than in the standard approach, increasing the bandwidth consumption during surfing time.

Therefore, to evaluate the impact of our proposition in terms of bandwidth increasing during zapping times, we model the multicast IPTV service system to estimate this increasing in different points of the network and compare it to the standard approach through simulations.

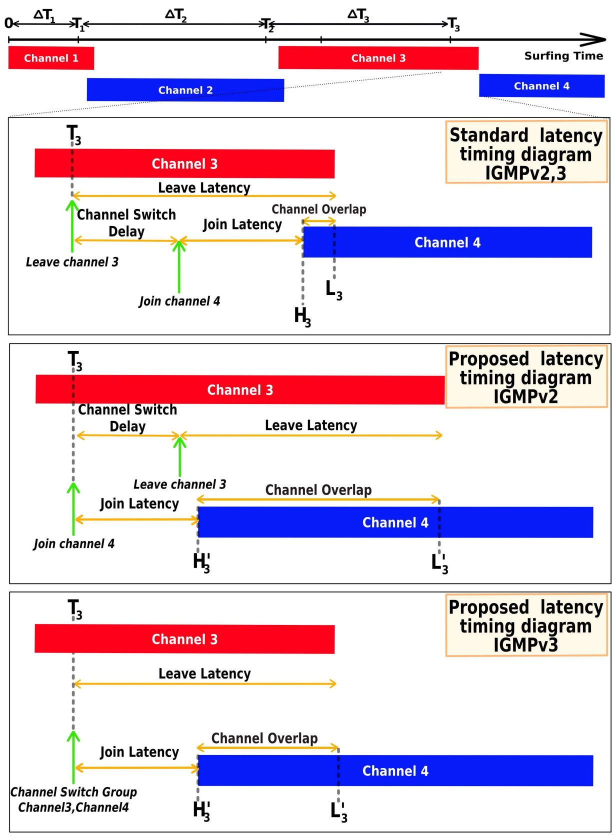

Figure 6: Channel surfing diagram

IV. MODELING IPTV SERVICE SYSTEM

Modeling of an IPTV service system takes into account several factors, including program popularity, channel definitions, viewer surfing behavior and various parameters related to the network itself. In each part of the modeling process, to finally estimate the bandwidth demand of multiple viewers, we have based our work on previous works [8, 20, 24, 25].

A. *Modeling IPTV Channel Popularity*

The effectiveness of the multicast model is based on how many viewers are watching the same channel at the same time. Many researches [20, 24, 25] has suggested that TV channel ranking follows a *Zipf* distribution. If we order the IPTV channels from the most popular to the least popular, the *Zipf* distribution implies that a few IPTV channels are highly ranked, whereas many are lower ranked. The *Zipf* distribution also known as *Power Law* distribution is defined by:

$$P_i = \frac{1/i^{\alpha}}{\sum_{c=1}^{N} 1/c^{\alpha}}$$

Where P_i is the probability that a viewer will choose the i^{th} raked channel. N is the number of the available channels in the network and α characterizes the form of the distribution.

To set the maximum peak usage of their networks, many broadband operators apply this distribution law with a given number of viewers and broadcast channels. This allows them to calculate the blocking probability which is the probability that the bandwidth of the requested channels exceed the available bandwidth.

Fig. 7 illustrates the *Zipf* distribution law when the number of IPTV channels is equal to 40, i is equal to 1 and α varies from 0.5 to 1.5 As we can see, the number of the most popular channel tends to increase when α is going down. A study presented in [25] shows that the typical values of α goes from 0.5 to 0.95 in normal event days and could be greater than 1.5 when special event happens.

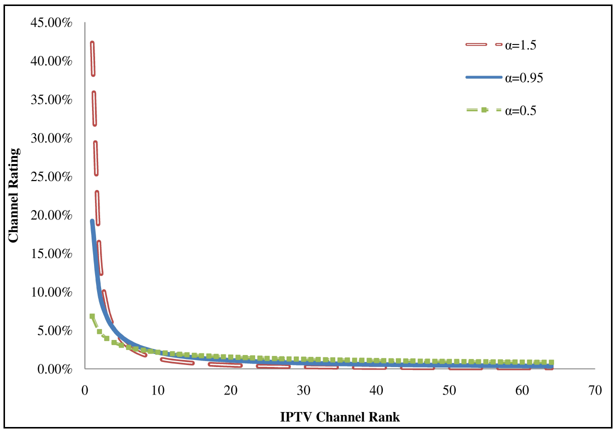

Figure 7: Channel popularity for deferent α values

B. Modeling Multicast Channel Surfing

The aim of modeling multicast channel surfing is to assess the maximum bandwidth usage of viewers watching multicast IPTV streams and then, to set the network load capability of the IPTV service provider to bring the blocking probability close to zero.

Basically, the total bandwidth usage at a given point of time is equal to the sum of the bit rates of the channels watched by the viewers. Therefore we will first model the bandwidth demand of a single viewer in the standard and the proposed approach and then we will infer the total bandwidth utilization of all viewers.

B.1. Modeling the bandwidth demand of one single viewer

According to previous works [8, 20], the surfing behavior of a single viewer is represented as a Poisson distribution with an average instant of channel changing equal to λ. To take into account the difference between the two IGMP protocol versions in our proposition when a zapping happens, we first model the bandwidth demand of a single viewer when IGMPv2 protocol is used. We will then easily extrapolate to the IGMPv3 case.

Denoted by ΔT_i $(i\geq 1)$ the time intervals between every channels changing. As we said earlier, ΔT_i follows a Poisson distribution. T_i $(i\geq 1)$, are the times when the channel change happens, it can be described by:

$$T_i = \sum_{k=1}^{i} \Delta T_k$$

To define the bandwidth demand of one viewer, we will focus on IGMP messages. So at each T_i $(i\geq 1)$ time, according to viewer surfing behavior, to start the channel switching process, the IPTV device will send an IGMP-Leave message in the standard case or an IGMP-Join message in our approach.

Suppose now that at time $T_0(=0)$, the viewer is watching channel # 1. As represented in Fig. 6, in the standard case, at each time T_i $(i\geq 1)$ until the end of the surfing time interval (which is in our simulation the mean duration of a commercial break), the IPTV device will send an IGMP-Leave message for channel $\#i$ and then (after the Channel Switch Delay) an IGMP-Join message for channel $\#i+1$. H_i described in (1) is the first instant, after T_i in which the bandwidth demand will change. Depending on the Leave Latency (*LL*), the Channel Switch Delay (*CSD*) and the Join Latency of channel $\#i+1$ (JL_{i+1}), the bandwidth demand will jump to a high state if a channel overlapping occurs or fall to zero if not. Equation (2) describes the condition for which the Channel Overlap Delay (*COD*) will not be equal to zero.

$$H_i = \min(LL, CSD + JL_{i+1}), i \geq 1 \qquad (1)$$

$$COD_i = \begin{cases} LL-(CSD+JL_{i+1}) & if \quad LL>CSD+JL_{i+1}, \quad i\geq 1 \\ 0 & if \quad LL\leq CSD+JL_{i+1}, \quad i\geq 1 \end{cases} \qquad (2)$$

Similarly, L_i described in (3) defines the instant, after T_i+H_i at which the bandwidth demand will change again to be equal to the bit rate of channel $\#i+1$.

$$L_i = \max(LL, CSD + JL_{i+1}), i \geq 1 \quad (3)$$

In a standard channel switching scenario, the bandwidth demand of a viewer v can therefore be expressed by:

$$B_v(t) = \begin{cases} CD_1 & \text{if} \quad t < T_1 + H_1 \\ 0 & \text{if} \quad T_i + H_i \leq t < T_i + L_i \text{ and } LL < CSD + JL_{i+1},\ i \geq 1 \\ CD_i + CD_{i+1} & \text{if} \quad T_i + H_i \leq t < T_i + L_i \text{ and } LL > CSD + JL_{i+1},\ i \geq 1 \\ CD_{i+1} & \text{if} \quad T_i + L_i \leq t < T_{i+1} + H_{i+1},\ i \geq 1 \end{cases} \quad (4)$$

Where CD_i is the bit rate of channel #i according to its definition (High or Simple Definition).

In the proposed channel switching scenario, when IGMPv2 protocol is used, at each T_i $(i \geq 1)$ time, the IPTV device sends at first an IGMP-Join message for channel #$i+1$ and then an IGMP-Leave message for channel #i. So, we can define:

$$H_i^{'} = \min(LL + CSD, JL_{i+1}), i \geq 1$$

$$COD_i^{'} = \begin{cases} JL_{i+1} - (CSD + LL) & \text{if} \quad JL_{i+1} < CSD + LL, \quad i \geq 1 \\ 0 & \text{if} \quad JL_{i+1} \geq CSD + LL, \quad i \geq 1 \end{cases} \quad (5)$$

and

$$L_i^{'} = \max(LL + CSD, JL_{i+1}), i \geq 1$$

Following the same reasoning as in the standard case, at any given point of time during the surfing, the bandwidth demand of a viewer v can be expressed by:

$$B_v^{'}(t) = \begin{cases} CD_1 & \text{if} \quad t < T_1 + H_1^{'} \\ 0 & \text{if} \quad T_i + H_i^{'} \leq t < T_i + L_i^{'} \text{ and } LL + CSD < JL_{i+1},\ i \geq 1 \\ CD_i + CD_{i+1} & \text{if} \quad T_i + H_i^{'} \leq t < T_i + L_i^{'} \text{ and } LL + CSD > JL_{i+1},\ i \geq 1 \\ CD_{i+1} & \text{if} \quad T_i + L_i^{'} \leq t < T_{i+1} + H_{i+1}^{'},\ i \geq 1 \end{cases} \quad (6)$$

In the case of using IGMPv3 protocol, the bandwidth demand has the same expression as in (6), except that when one message is sent to switch between channels instead of two, the Channel Switch Delay is equal to zero.

B.2. Modeling the bandwidth demand of m viewers.

The bandwidth demand depends on both viewers who are surfing and those who are not. To describe the bandwidth of m viewers during surfing, we must first calculate the bandwidth demand of each of them according to equation (4) or (6). Based on the bandwidth demand of each of the m viewers, the total bandwidth demand of multiple viewers can be described by:

$$B_{MV}(t) = \sum_{c=1}^{n} CD_c I(N_c(t) > 0)$$

Where n is the number of the channels available in the network, $N_c(t)$ is the number of the viewers who are watching channel #c and I is an indicator function.

Now that we have modeled the IPTV channel popularity and the bandwidth demand of multiple viewers, to run our simulations and estimate the bandwidth utilization based on equations (4) and (6), we need first to measure real parameters like Join Latency for a given Channel Switch Delay and Leave Latency value and that, in each approach and with each IGMP protocol version. To measure these parameters with IGMPv3 protocol when our proposition is applied, some kernel code modifications are needed to switch between channels by sending a unique message. These code modifications are presented in the following section.

V. IGMPv3 Code Modifications

The idea of switching from one IPTV channel to another with a unique message is attractive, but the current IGMPv3 Application Program Interface (API) is not suitable for this. In this section we will show how the API can be updated to make this available. When a viewer wants to join or leave a specific IPTV channel, his/her IPTV device will use the classical IP API ***setsockopt()*** function call with multicast options, to send a Join or a Leave IGMP message. These options are sorted in IPv4, IPv6 and IP version-independent options [26]. To illustrate our proposition we focus only in IPv4.

With the current available IGMP specific options of the ***setsockopt()*** function, the switching from a currently watched group (or IPTV channel) to a new one needs two successive calls. All tested software/hardware solutions send first an IGMP-Leave message and then an IGMP-Join message. These requests are expressed at the socket layer by successive ***setsockopt (socket, IPPROTO_IP, ACTION, imr, sizeof(imr))*** calls, where "**ACTION**" is first set to "IP_DROP_MEMBERSHIP" to drop the left multicast channel and then to "IP_ADD_MEMBERSHIP" to join the new one. ***imr*** is an ***ip_mreq*** structure containing the multicast address and the device on which the group will be joined or left.

Branches 1 and 2 of Fig. 8 represent the IGMP functions called to switch between channels in the standard case. When the viewer's IPTV device calls the ***setsockopt()*** function to join or leave an IPTV channel, the corresponding ***do_ip_setsockopt()*** (line 402:***ip_sockglue.c***) [27] function in the kernel space is called. Depending on the "**ACTION**" parameter the program goes to the join or to the leave part.

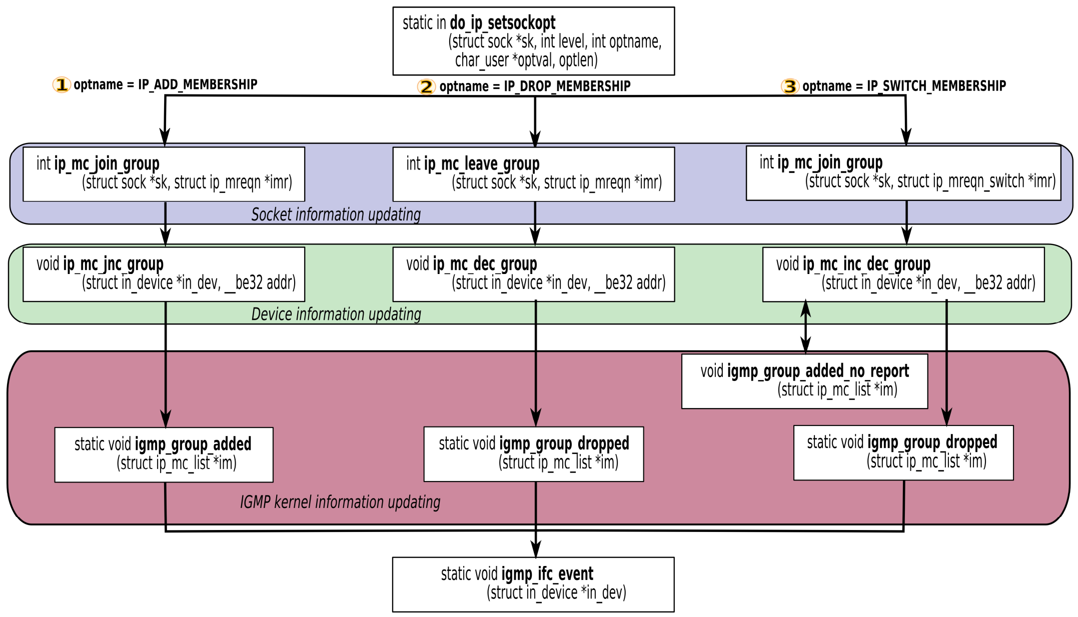

Figure 8: Flowchart of the called functions to switch between channels

The behavior is similar in both cases and follows these steps:

1. Firstly, in the *ip_mc_join_group()* (line 1720:***igmp.c***) or *ip_mc_leave_group()* (line 1792:***igmp.c***) function, the elements of the current socket are updated by adding the multicast group to be joined or deleting the multicast group to be left,

2. Secondly a call to the i*p_mc_inc_group()* (line 1198***:igmp.c****) or ip_mc_dec_group()* (line 1257***:igmp.c***) updates the elements of the physical device on which the socket is connected. This is done by adding the multicast group to join or by deleting the multicast group to leave. Additionally the elements like IGMPv3 timers and filter modes are set [5],

3. Thirdly, the *igmp_group_added() (*line 1159***:igmp.c)*** or ***igmp_group_dropped()*** *(*line 1122***:igmp.c)*** function is called to mainly deal with the active version of IGMP (version 2 or version 3).

4. Finally, the *igmp_ifc_event()* (line 716:***igmp.c****)* function is called to trigger the IGMP-Report message to join or to leave the IPV channel.

   As indicated earlier, depending on the value of the IGMP Robustness Variable, these steps could be repeated.

According to our proposition made in section III, we define a new "**ACTION**" option called "IP_SWITCH_MEMBERSHIP", which can be used in the ***setpsockopt()*** function . Its goal is to produce a single message carrying both *Join and Leave* information**.**

A. The new "IP_SWITCH_MEMBERSHIP" option in the IGMP host part

To make the new "IP_SWITCH_MEMBERSHIP" option available for ***setpsockop*****t()** calls, we modified the source code [27] and added a new case in the ***do_ip_setsockopt()*** function code. Branch 3 of Fig 8 summarizes the sequences of the called functions to finally send a unique message to switch between multicast channels.

Unlike the "IP_ADD_MEMBERSHIP" and the "IP_DROP_MEMBERSHIP" options, the new "IP_SWITCH_MEMBERSHIP" option needs simultaneous information about the joined and the left groups. Therefore we created a new ***ip_mreqn*** structure called ***ip_mreqn_switch*** carrying two address fields called ***imr_joinaddr*** and ***imr_leaveaddr*** instead if the one which is called ***imr_multiaddr*** in the standard case. The rest of the structure stays similar to the standard case.

```
struct ip_mreqn_switch {
        struct in_addr imr_joinaddr;    /* group to join        */
        struct in_addr imr_leaveaddr;   /* group to leave       */
        struct in_addr imr_interface;   /* interface to join on */
}
```

Like the joining/leaving sequences presented in branch 1 and 2 of Fig. 8, the ***do_ip_setsockopt()*** function calls a new *ip_mc_switch_group()* function. This function is the merge of the *ip_mc_join_group()* (line 1614:***igmp.c***) and the *ip_mc_leave_group()* (line 1692:***igmp.c***) functions. The goal of this new function is to update the structure of the properties of the socket in use. So the ***imr_joinaddr*** address is inserted at the beginning of the multicast groups list and the ***imr_leaveaddr*** address is removed from this group list. Additionally, the *ip_mc_inc_group()* (line 1659:***igmp.c***) and the *ip_mc_dec_group()* (line 1717:***igmp.c***) functions of the branch 1 and 2 of Fig. 8 are replaced by a new *ip_mc_inc_dec_group()* function. This function is used to update the elements of the physical device used by the current socket. It is the merging of *ip_mc_inc_group()* and *ip_mc_dec_group()* functions. The difference with the original functions is that this function needs to receive the two groups' information simultaneously.

Finally, the *igmp_group_added()* function of the standard case is replaced by a new *igmp_group_added_no_report()* function. In the original *igmp_group_added()* function, in addition to the settings of the IGMP protocol variables and timers there is a call to the *igmp_ifc_event()* (line 1091:***igmp.c***) function. This call triggers an IGMP-Report message. To avoid this and allow the further message to be sent as a 2 in 1 message, we create a new function called *igmp_group_added_no_report*() in which the *igmp_ifc_event()* is suppressed. The rest of the merged functions remains unchanged. That's it, at the end of the *ip_mc_inc_dec_group()* function a call to the *igmp_ifc_event()* function triggers an IGMP-Report message. This message will then contain both the new added group in the form of a (*group_to_join: change_to_include_mode{}*) group record and the deleted group in the form of a (*group_to_leave:change_to_exclude_mode{}*) group record.

As shown in Fig. 9, in the standard approach, the switching method from group 239.1.1.1 to group 239.1.1.2 made with the multimedia player VLC is expressed with IGMPv3 by two redundant messages. IGMP protocol robustness requires that each message is repeated twice. This requirement leads to a mixed Leave/Join message at the end of the switching process. Delta time are displayed in the *''Time''* column, we can see that the *CSD* is equal to 152 ms in this try.

The frames capture of Fig. 10 show the result of a ***setsockopt()*** call with the new defined "IP_SWITCH_MEMBERSHIP" option for a switching from group 239.1.1.2 to group 239.1.1.1. As in Fig. 9, the robustness variable value (equal to 2) makes that the message is repeated twice. The comparison of Fig. 9 and Fig. 10 shows us that now, only one messages is sent to switch from one group to another.

B. The new ''IP_SWITCH_MEMBERSHIP'' option in the IGMP router part

Because the version 3 of the IGMP protocol is designed to take into account IGMP-Report messages carrying several records of IGMP-Join/Leave information, there is no need to any modification in order to take into account this new option.

C. Backward compatibility

If we force the IGMP version of multicast hosts to version 2 by setting the kernel *force_igmp_version* parameter to 2 and run zapping tests with code using the "IP_SWITCH_MEMBERSHIP" option, we notice that our new implemented option stay compatible with version 2 of IGMP and two messages will be sent to switch between channels.

VI. SIMULATION AND RESULTS

As indicated at the end of section IV, to run our simulation based on equations (4) and (6) we went through two steps. In a real network, we stared by measuring the Join Latency (*JL*) of IPTV channels for a given values of *LL*

Time	Source	Destination	Info
0.000000	172.16.179.132	224.0.0.22	V3 Membership Report / Join group 239.1.1.1 for any sources
0.786027	172.16.179.132	224.0.0.22	V3 Membership Report / Join group 239.1.1.1 for any sources
6.758307	172.16.179.132	224.0.0.22	V3 Membership Report / Leave group 239.1.1.1
0.152416	172.16.179.132	224.0.0.22	V3 Membership Report / Join group 239.1.1.2 for any sources
0.589402	172.16.179.132	224.0.0.22	V3 Membership Report / Leave group 239.1.1.1 / Join group 239.1.1.2 for any sources

Figure 9: Whireshark screenshot of the channel switching scenario in the standard case when IGMPv3 is used

Time	Source	Destination	Info
0.000000	172.16.179.132	224.0.0.22	V3 Membership Report / Join group 239.1.1.2 for any sources
4.087784	172.16.179.132	224.0.0.22	V3 Membership Report / Join group 239.1.1.2 for any sources
3.199888	172.16.179.132	224.0.0.22	V3 Membership Report / Leave group 239.1.1.2 / Join group 239.1.1.1 for any sources
2.452044	172.16.179.132	224.0.0.22	V3 Membership Report / Leave group 239.1.1.2 / Join group 239.1.1.1 for any sources

Figure 10: Whireshark screenshot of the channel switching scenario in the proposed case when IGMPv3 is used

before running our simulations.

A. Measurement of the Join Latency in a real network

Vialis [28] is a small ISP based in Colmar (France) offering a triple play service over a Cable, FTTH-PON (FTTH-Passive Optical Network) and FTTH-P2P (FTTH-Point To Point) networks. In [1] we already measured the Join Latency and evaluated the bandwidth demand in Vialis PON network where only IGMPv2 was used. In this paper, to run additional simulations, we have chosen the FTTH-P2P Vialis network in which we measured the Join Latency, and so the Network Delay, in the standard and the proposed approach with each IGMP protocol version.

As presented in Fig. 11, Vialis IPTV network is based on 1 IPTV Broadband, 4 multicast routers, one Ethernet switch and two optical switches (one in the Head End and another in the last mile side). Each optical switch has 24 optical ports and all active equipments run IGMP Proxy function.

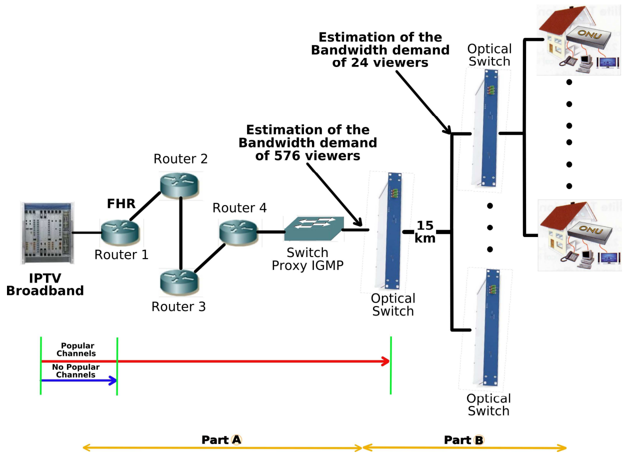

Figure 11: Vialis FTTH-P2P network

According to previous work [11], the most popular channels are directly available at the optical switch in the Head End while all others must be required further in the network. According to [7], we set the Leave Latency equal to 100ms (Max Response Time = 100ms, Robustness Variable =1).

We modified the software code of the IPTV device to test our proposed approach with IGMPv2 and IGMPv3. When IGMPv2 is used with our proposition, the Channel Switch Delay was measured equal to 10 ms. This value may be reduced when the software code modification is optimized

The time to setup a branch end to end between an IPTV device and the IPTV Broadband was measured equal to 125ms. For each version of IGMP protocol, this value is composed of 10 ms from the IPTV device to the optical switch in the Head End (branch A in Fig. 11) and, in accordance with [29], 115 ms from this optical switch to the IPTV Broadband through the 4 routers and the Ethernet switch (branch B in Fig. 11). To simplify our illustration, we assume that at any place of the network, the Join Latency does not vary with the network traffic load.

Table II summarizes the values of the parameters used in our simulation, 10% of the available channels are HD channels and 60 % of the users have STB to watch IPTV channels, the rest uses software solutions.

TABLE II
PARAMETERS VALUES OF OUR EXAMPLE

Symbol	Definition	Values
λ	Mean of Poisson distribution	5
T_{surf}	Channel surfing duration	300 seconds
n	Number of available channels	500 channels
α	Zipf exponent value	0.95
SD	Bit stream of Simple Definition channel	4 Mbps
HD	Bit stream of High Definition channel	15 Mbps
LL	Leave Latency	100 ms
CSD_1	Channel Switch Delay of STB (in the Standard Approach)	20 ms
CSD_2	Channel Switch Delay of computer software solution (in the Standard Approach)	200 ms
CSD_3	Channel Switch Delay (in the Proposed Approach)	10 ms
JL_1	Join latency of the 10 most popular channels	10 ms
JL_2	Join Latency of all channels except the 10 most popular one	125 ms

.

Now that we have the values of the simulation parameters, we will estimate, as a second step, the bandwidth demand and the maximum channels requested for a given number of viewers and channels during a commercial break with Scilab, a software for numerical computations [30].

B. Estimation of the bandwidth demand of 24 active viewers

At T=0, suppose that all active viewers are watching IPTV channels according to their popularity. The first simulation conducted with Scilab shows us in Fig. 12 that for a branch of 24 active viewers, the bandwidth demand varies from 51 Mbps to 175 Mbps both in the standard and the proposed approach. The big gap between the minimum and the maximum value of the bandwidth demand is caused by the number of the requested channels according to their popularity, and the channel overlapping when it happens. This means that, as we did, the channel switching scenario

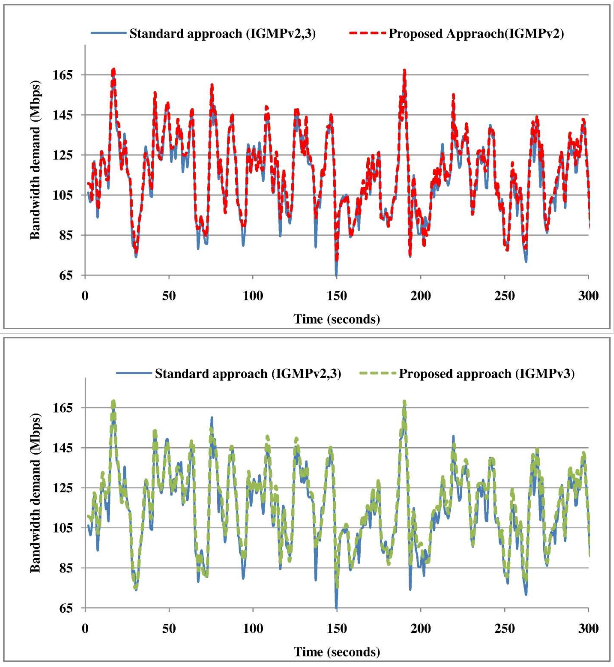

Figure 12: 3-moving average of bandwidth demand

must be taken into account to model the bandwidth demand of multicast streams.

Fig. 13 shows over a short period of time the difference between the bandwidth demand in the standard and the proposed approach for each version of IGMP protocol. We can see that our solution increases the bandwidth demand only during overlap periods.

Unlike the estimation of the maximum number of the requested channels at the same time, illustrating a real time bandwidth demand is not a good indicator to dimension multicast IPTV networks and know, at the top, how many channels will be requested at the same time.

C. Estimation of the maximum number of the requested channels by the active viewers

To evaluate the maximum number of channels requested at the same time in the multicast IPTV network, we use Monte Carlo approach [25]. Keeping the same parameter values of Table II, we vary the number of active viewers from 24 to 576 (24*24) and the number of available channels from 250 to 500. For each of those parameters value, we repeat the process 1,000 times. At first we do not vary the value of the Zipf exponent α and set it to 0.95.

As we can see in Fig. 14, for 24 active viewers in a single FTTH-P2P branch, our approach will not increase the maximum number of the requested channels regardless of the number of available channels in the network and the IGMP protocol version used. For each approach, the maximum number of the requested channel varies from 23 to 27 if 250 channels are available and from 24 to 28 if 500 channels are available. In 1000 runs, 24 and 25 are the numbers of the maximum channels which occurs the most if respectively 250 channels and 500 channels are available.

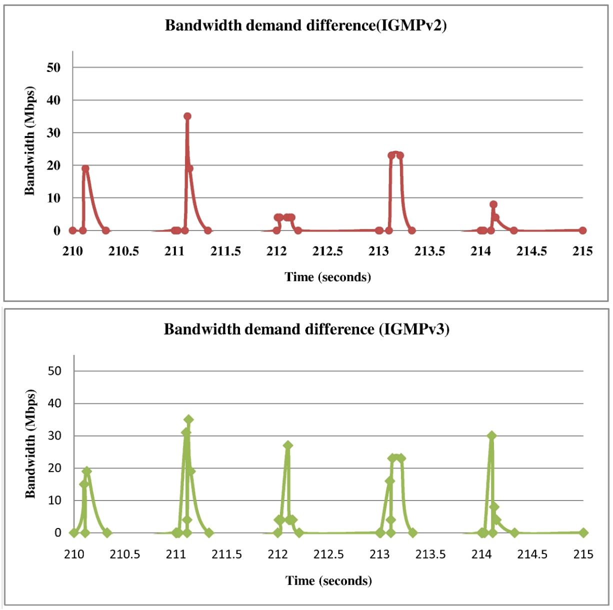

Figure 13: Bandwidth demand difference

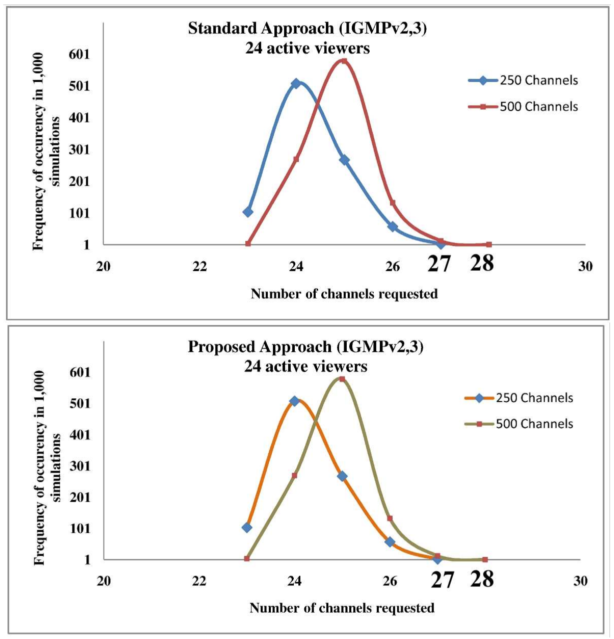

Figure 14: Maximum channels requested by 24 active viewers

However, to set the network load capability and bring the blocking probability close to zero, an IPTV service provider must estimate the maximum peak rate in the core network.

Since the 10 most popular channels are available at the optical switch in the Head End, to estimate the maximum number of the requested channels between the optical switch and the upper router, these channels will be taken into account regardless of viewer behavior in both approaches. At this point of the network, based on the values in Table II, the Join latency is equal to 115 ms ($JL_2 - JL_1$).

Fig 15 makes it clear that globally, as in the FTTH-PON network [1], for the chosen parameters values of Table II, our approach does not increase the bandwidth demand compared to the standard approach. In other terms, the multicast bandwidth necessary to provide IPTV service is the same in both standard and proposed channel switching scenario. This is due to the fact that, as it's showed in Table III, when the Join Latency is equal to 115 ms, the Channel Overlap Delay (*COD*) is equal to zero based on equation (2) and (5), in each approach, regardless of the IGMP protocol version.

The *COD* is not null in a branch of 24 users only when the popular channels are requested.

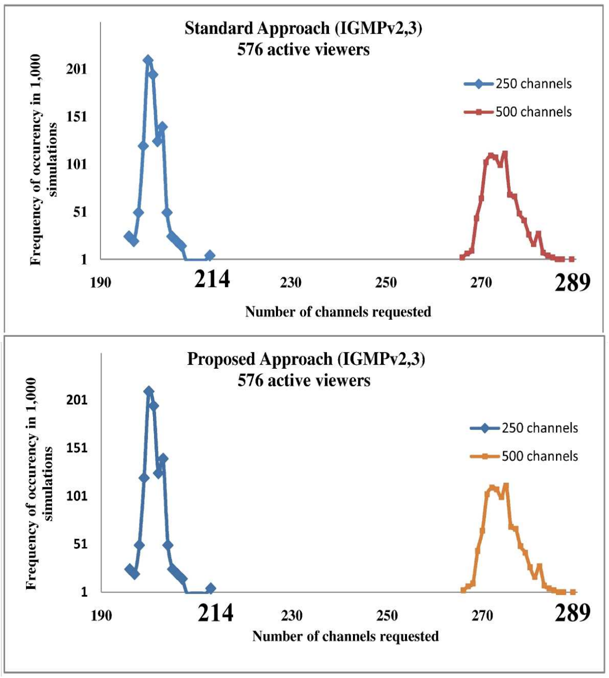

Figure 15: Maximum channels requested by 576 active viewers

TABLE III
VALUES OF THE CHANNEL OVERLAP DELAY (1)

Cases	Leave Latency (ms)	Channel Switch Delay (ms)	Join Latency (ms)	Channel Overlap Delay (ms)	Part in Fig. 11
Standard IGMPv2,3	100	20	10	70	B
	100	**20**	**115**	**0**	A
	100	20	125	0	A+B
	100	200	10	0	B
	100	**200**	**115**	**0**	A
	100	200	125	0	A+B
Proposed IGMPv2	100	10	10	100	B
	100	**10**	**115**	**0**	A
	100	10	125	0	A+B
Proposed IGMPv3	100	0	10	90	B
	100	**0**	**115**	**0**	A
	100	0	125	0	A+B

To see what happens if the Channel Overlay Delay exists in our approach, the Join Latency must be reduced to satisfy the following condition:

$$JL < CSD + LL$$

Therefore, only one multicast router in the Head End was necessary to remove to reduce the Join Latency and make channels overlap in the proposed approach. Table IV summarizes those new measured Join Latency values in each part of the network presented in Fig. 11. We will then compare our approach when channels overlap during a zapping process with a standard case in which channels don't overlap in the core network.

TABLE IV
VALUES OF THE CHANNEL OVERLAP DELAY (2)

Cases	Leave Latency (ms)	Channel Switch Delay (ms)	Join Latency (ms)	Channel Overlap Delay (ms)	Part in Fig. 11
Standard IGMPv2,3	100	20	10	70	B
	100	**20**	**85**	**0**	A
	100	20	95	0	A+B
	100	200	10	0	B
	100	**200**	**85**	**0**	A
	100	200	95	0	A+B
Proposed IGMPv2	100	10	10	100	B
	100	**10**	**85**	**25**	A
	100	10	95	15	A+B
Proposed IGMPv3	100	0	10	90	B
	100	**0**	**85**	**15**	A
	100	0	95	5	A+B

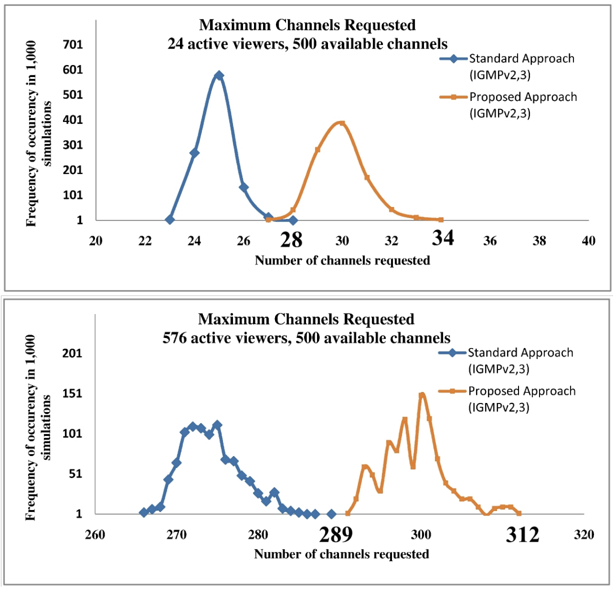

Figure 16: Maximum channels requested if channels overlap

As showed in Fig. 16, compared to the standard approach, our proposition may increase the maximum number of the requested channels when channels overlap during a zapping process. In our approach, if 500 channels are available in a branch of 24 active viewers, the maximum number of the requested channels varies from 27 to 34 with a mean of 30 channels. Compared to the standard approach, this increase is about 21%. In the network core when 576 viewers are active, the number of the maximum channels requested at the same time varies from 290 to 312 channels in our approach, which is an increasing of 8% compared to the standard approach.

Of course, depending on the values of the Zipf exponent, for a given number of active viewers, the maximum number of the requested channels may not be the same in each approach. Therefore, we evaluate the impact of channel popularity in Fig. 17 and Fig. 18 for a branch of 24 active viewers and then, for 576 active viewers.

Fig. 17 shows us, based on the values in Table IV, the maximum number of the requested channels in a branch of 24 active viewers for 3 different α values. As we can see, in the standard approach, this number goes from 21 channels, if we have special events (α=1.5), to 28 channels if the raking values of the most popular channel are small (α=0.5). In the proposed approach, the maximum number of the requested channels varies from 27 to 39. Also, compared to the standard approach, our approach increases in each branch of the FTTH-P2P network, the maximum number of the requested channels from 9% to 32% depending on α value.

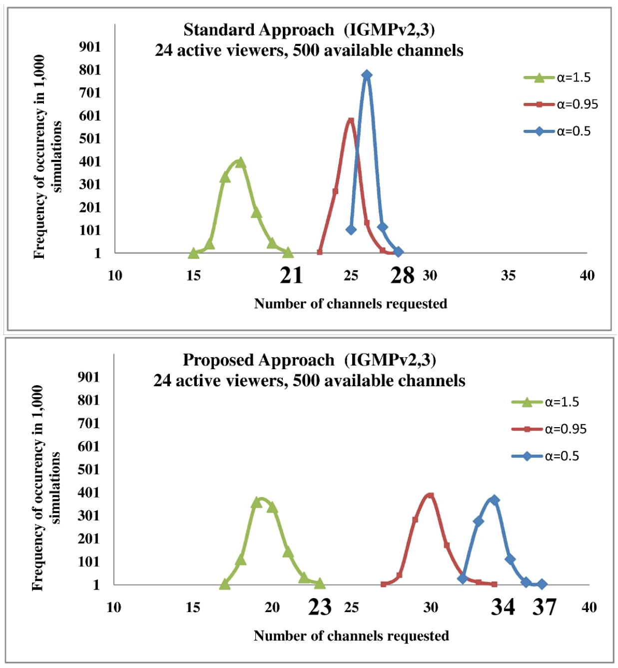

Figure 17: Maximum channels requested depending on *α* value

Fig. 18 shows us that in a multicast IPTV network of 576 active viewers our proposition increases the maximum number of the requested channel only by 8% the standard approach, regardless of the IPTV programs popularity and the IGMP protocol version used. This is because of the big number of active viewers who request generally the same IPTV channels during the surfing.

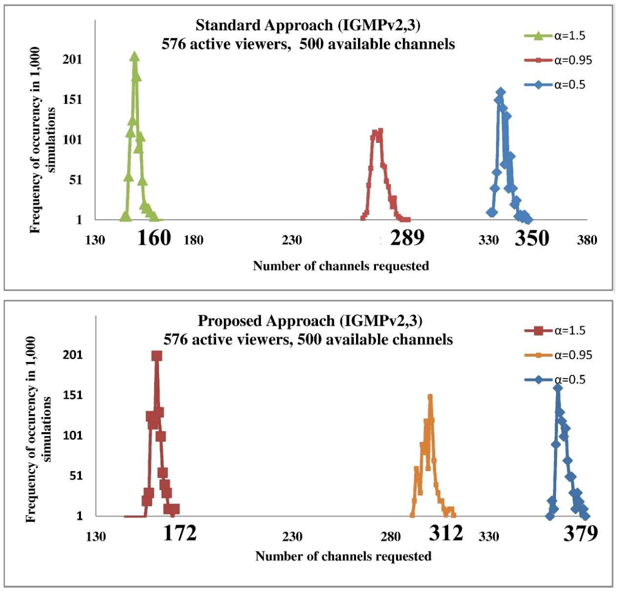

Figure 18: Maximum channels requested depending on *α* value

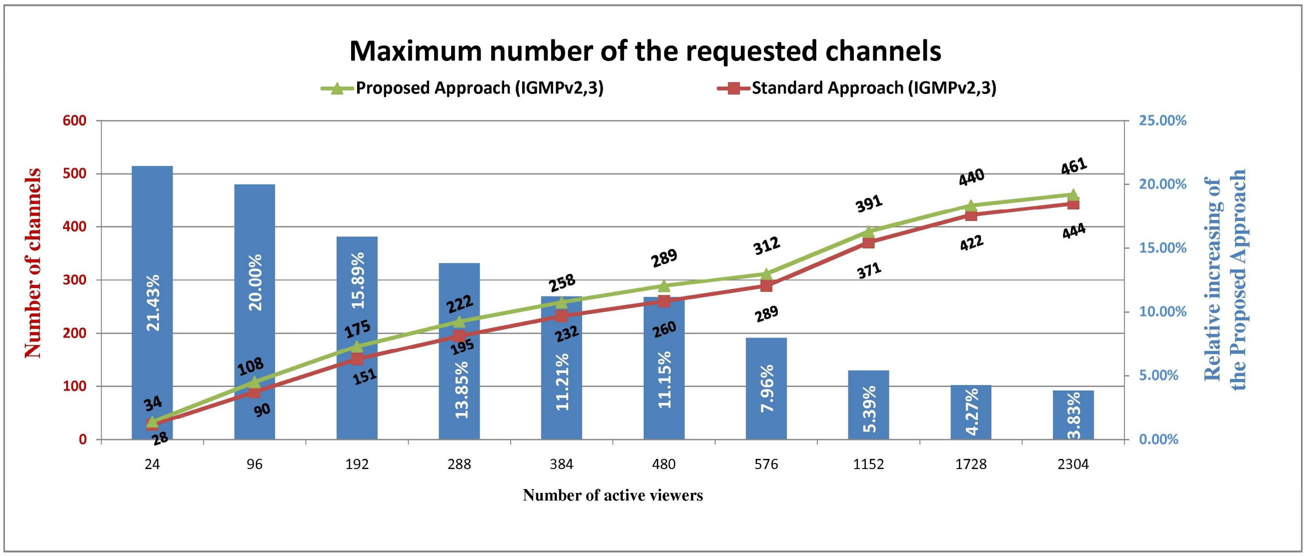

Figure 19: Maximum number of the requested channels

Our last simulations show that, compared to the standard approach, the relative increasing caused by our approach when channels overlap, is reduced when the number of viewers increases. As it's presented in Fig. 19, when α=0.95 and 500 channels are available, the impact of the channel overlapping has less effect when the number of active viewer is increasing. The biggest difference value of the maximum requested channels was measured when 480 viewer are active, this value decreased to be equal to 17 channels when 2304 active viewers are in the network.

D. Reduction of the Blackout time in the proposed approach

When a viewer requests a new channel, a blackout appears during the switching time. This is due to the decoding process delay and the time between the last packet of the currently watched channel and the first packet of the requested one.

Depending on the network parameter values (Leave Latency, Join Latency and Channel Switching Delay), our approach may introduce a channel overlapping. In the previous section, we showed that this may increase a little the bandwidth demand in the network core, but the good point is that it can also reduce the blackout time increasing the quality experienced by the viewer.

Based on the parameter values of Table II, during a period of 300 seconds, we summed, according to (4) and (6), the time interval in which the bandwidth demand is equal to zero. In a branch of 24 active viewers, the average of the measured value was measured equal to 6.5 seconds per viewer in the standard approach while in our approach, this value goes down to 0.6 second per viewer when IGMPv2 is used and 1 second when IGMPv3 is used to switch between channels.

VII. CONCLUSION AND FUTURE WORK

To reduce the zapping time, many solutions were proposed based on the same channel switching scenario, this standard scenario could be more optimized. Therefore, we proposed in this paper a novel scenario which stays working with the previous solutions. Unlike the standard approach, to switch between two channels, the solution we proposed consists in sending an IGMP-Join message for the requested channel before leaving the currently watched channel by sending an IGMP-Leave message. Our solution improves channel zapping time, especially when users are using computer software solution. However, it may create short overlaps period between channels during a zapping process increasing briefly the bandwidth demand.

We showed then that with some source code modifications, IGMPv3 becomes suitable for switching between channels with a unique message

To estimate the impact of the channel overlapping in terms of bandwidth consumption, we modeled the multicast bandwidth demand of viewers surfing during a commercial break and we prove that globally, depending on network parameters values and regardless of the used IGMP version protocol, our solution increase by 8% the maximum number of the requested channel in a core network where 500 channels are available and 576 viewers are active during a zapping process. Moreover, this relative increasing tends to decrease when the number of the active viewer becomes bigger.

We finally measured that the blackout time is reduced bringing an additional improvement of the viewer experience.

In this work, we acted mainly at the first step of the zapping process which is the network processing part. In further work, we will focus on the second step of the zapping process. For the same aim, we will propose some mechanisms to send secondary streams, based on the mean streams and the moment of the zapping, to reduce the Buffering Delay and the First I-Frame Delay at the same time.

REFERENCES

[1] M. Sarni, B. Hilt and P. Lorenz, “A Novel Channel Switching Scenario in Multicast IPTV Networks”, The Fifth International Conference on Networking and Services (ICNS 2009), April 2009.

[2] “Assuring Quality of Experience for IPTV”, HeavyReading Whitepaper, July 2006.

[3] Estrin D., Farinacci D., Helmy A., Thaler D., Deering S., Handley M., Jacobson V., Liu C., Sharma P., and L. Wei, “Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification”,4601, August 2006.

[4] W. Fenner, “Internet Group Manamagement Prtocol, version 2’’,RFC 2236, November 1997.

[5] B. Cain, S. Deering, I. Kouvelas, B. Fenner, A. Thyagarajan, “Internet Group Management Protocol, version 3” ,RFC3376, October 2002.

[6] L. Hart, ‘’Introduction to MPEG; MPEG-1, MPEG-2 and MPEG-4”, Althos Publishing, August 2006.

[7] J. Kim, H. Yun, M. Kang, T. Kim, and J. Yoo, “Performance evaluation of channel zapping protocol in broadcasting services over hybrid WDM-PON”, International Conference on Advanced Communication Technology, vol. 2, pp. 1152-1155, Feb. 2005.

[8] D. E. Smith, “IP TV Bandwidth Demande : Multicast and Channel Surfing”, International Conference on Computer Communications, vol. 26, pp. 2546 -2550, 2007.

[9] C. Saraki, A. Tagami, T. Hasengawa and S. ANO, “Rapid Channel Zapping for IPTV Broadcasting with Additional Multicast Stream,” International Conference on Communication, October 2008.

[10] Y. Berjerano and P. V. Koppol, “Improving Zap Response Time for IPTV”, International Conference on Computer Communication, March 2009.

[11] U. Jennehag and T. Zhang, “Increasing bandwidth utilization in next generation IPTV networks”, International Conference on Image Processing, vol. 3, pp. 335-340, July 2002.

[12] J. M. Boyce and A. M. Tourapis, “Fast efficient channel change [set-top-box applications]”, International Conference on Computers in Education Digest of Technical Papers, pp. 1-2, Jan. 2005.

[13] D. Banodkar, K. K. Ramakrshmann, S. Kalyanaraman, A. Gerber and O. Spatscheck, “Multicast Instant Channel Change in IPTV Systems”, 3rd International Conference on Communication Systems Software and Middleware, January 2008.

[14] Cisco System, “Optimizing Video Transport in Your IP Triple Play Network”, WhitePaper, 2006.

[15] H. Joo, H. Song, D. B. Lee, and I. Lee, “An effective IPTV channel control algorithm considering channel zapping time and network utilization”, in IEEE Transactions on Broadcasting, vol. 51, pp. 208-216, 2008.

[16] P. Sieber, T. N. M. Van Caenegem and M. Wagner, “Analysis and Improvements of Zapping Time in IPTV Systems”, IEEE Transaction on Broadcasting, p. 404-418, vol.55, n.2, June 2009.

[17] C. Cho, I. Han, Y. Jun and H. Lee, “Improvement of channel zapping time in IPTV services using the adjacent groups join-leave method”, International Conference on Advanced Communication Technology, vol. 2, pp. 971-975, 2004.

[18] S. M. Mandal, M. MBuru, “Intelligent Pre-fetching to Reduce Channel Switching Delay in IPTV Systems”, department technical report, Texas A & M University.

[19] J. Lee, G. Lee, S. Seok, B. Chung, “Advanced scheme to reduce IPTV channel zapping time”, in *LNCS 4773*, pp. 235-243, 2007.

[20] Y. Kim, J. K. Park, H. J. Choa, S. Lee, H. Park, J. Kim, Z. Lee and K. Ko, “Reducing IPTV channel zapping time based on viewer’s surfing behavior and preference”, International Symposium on Broadband Multimedia Systems and Broadcasting, 2008.

[21] S. H. Hsu, M. Wen, H. Lin, C. C. Lee, and C. H. Lee, “AIMED – A personalized TV Recommendation system”, in LNCS 4471, pp. 166-174, 2007.

[22] VLC, http://www.videolan.org/.

[23] MyFreeTV, http://myfreetv.sourceforge.net//

[24] D. T. vanVeen, M. K. Weldon, C. C. Bahr, and E. E. Harstead, “An Analysis of the Technical and Economical Essentials for Providing Video over Fiber-to-the-Premises Network”, Bell Labs Tech J., vol. 10, no. 1, pp. 181-200, 2005.

[25] J. Gong, D. A. Vivanco and J. Martin “Simulation Study of the Spectral Capacity Requirements of Switched Digital Broadcast”, Fourth International Conference on Broadband Communications, Networks and Systems, September 2007.

[26] W.R Stevens, B. Fenner, A. M Rudoff, “Unix Network Programming”, Addison Wesey, Nov. 21, 2003.

[27] IPv4 API, http://www.isovarvas.com/lxr/http/source/net/ipv4.

[28] Vialis website, www.calixo.net.

[29] V. Lucas, J. J. Pansiot, M. Hoerdt “Influence du temps d’adhésion sur le contrôle de congestion multicast’’, French symposium on engineering protocols, 2006.

[30] Scilab software, http://www.scilab.org/.