

CHANNEL SMURFING: MINIMISING CHANNEL SWITCHING DELAY IN IPTV DISTRIBUTION NETWORKS

*Fernando M.V. Ramos**, *Jon Crowcroft**, *Richard J. Gibbens**, *Pablo Rodriguez[‡]*, *Ian H. White**

*University of Cambridge [‡]Telefonica Research, Barcelona

ABSTRACT

One of the major concerns of IPTV network deployment is channel switching (or zapping) delay. This delay can add up to two seconds or more, and its main culprits are synchronisation and buffering. By analysing an extensive dataset - comprising 255 thousand users, 150 TV channels, and covering a 6-month period - we have observed that most channel switching events are linear: it is very common the user switching up or down to the next TV channel. This fact led us to the proposal, in this paper, of a simple mechanism to reduce channel switching delay. Our proposal is to send the neighbouring channels (i.e., channels adjacent to the requested one) to the Set Top Box (STB) during zapping periods. If the user switches to any of these channels the switching latency is virtually eliminated, not affecting therefore user's experience.

Notwithstanding the simplicity of this scheme, trace-driven simulations show that the zapping delay can be virtually eliminated for a significant percentage of channel switching requests. As an example, by sending the previous and the next channel concurrently with the requested one, for only one minute after a zapping event, switching delay is eliminated for around 45% of all channel switching requests. Furthermore, this simple scheme has a performance close to that of an ideal predictor, while the increase of bandwidth utilisation in the access link is negligible.

Keywords— IPTV, switching delay

1. INTRODUCTION

The offer of TV services over IP networks is very attractive as it presents a new source of revenues for network operators. IPTV offers network providers greater flexibility, while at the same time offering users a broad range of new applications. In order to compete in this market, IPTV operators have to at least guarantee the same QoS offered by cable networks or over the air broadcasts. In this respect, one of the major concerns of IPTV network deployment is channel switching (or zapping) delay. This delay can add up to two seconds or more (Section 2). It is known that this figure should be below 430ms to guarantee an acceptable user experience[10], so this is a clear handicap for IPTV providers. The main culprits for this high delay are stream synchronisation and buffering. For

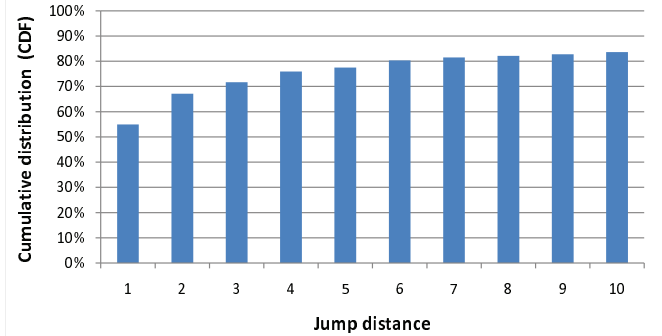


Fig. 1. Cumulative distribution of zapping jump distance

this reason, there is a good set of papers focused on this issue. Solutions to this problem exist both at the video coding and processing level and at the network level (see Section 7 for more details).

By analysing an extensive dataset comprising 255 thousand users, 150 TV channels, and covering a 6-month period (Section 4), we have observed that most channel switching events are linear: it is very common the user switching up or down to the next TV channel. This occurs despite the IPTV service in question including an Electronic Program Guide. Even when zapping is not linear, the “jump distance” is usually small (i.e., there is a high probability for the user to switch to one of the neighbouring channels). This can be observed in Figure 1. The probability of zapping linearly is close to 55%, and the probability of jumping to a close neighbour, for example not further away than 6 channels, is equal to 80%.

Based on all the above, we propose a simple mechanism to reduce channel switching delay. Our proposal (Section 5) is to send the neighbouring channels (i.e., channels adjacent to the requested one) to the Set Top Box (STB) during zapping periods. These neighbouring channels are synchronised and buffered concurrently with the requested one. Therefore, if the user decides to switch to any of these channels, the switching delay experienced is virtually zero.

One of the main advantages of this scheme is its simplicity. Notwithstanding its lack of sophistication, trace-driven simulations using the huge dataset mentioned before show

that the zapping delay can be virtually eliminated for a significant percentage of channel switching requests (Section 6). As an example, by sending only two channels concurrently (the previous and the next, respectively), for only one minute after a zapping event, switching delay is eliminated for around 45% of all channel switching requests. This figure jumps to 60% if we consider zapping periods only. Besides this, we also show that this simple scheme has a performance close to that of an ideal predictor.

It is important to stress that we assume the access network is able to accommodate the peak bandwidth needed to distribute several TV channels concurrently. Most systems today distribute Standard Definition (SD) TV channels using MPEG2, requiring 4 Mbps guaranteed bit rate per channel. We believe, however, that this is not a limiting factor to our scheme. In fact, in most OECD countries access networks already offer tens of Mbps of average download speeds. Japan and South Korea, for instance, have an average broadband speed close to 100 Mbps, with Japan already offering 1Gbps to some users, and it is expected other countries to follow this trend in the near future[8]. More importantly, by using our scheme the increase of bandwidth utilisation in the access link is negligible, since the concurrent channels are only kept in the STB during zapping periods. We also assume that the STB is able to process (i.e., synchronise and buffer) several TV channels in parallel. In this study we take both these limitations in consideration, and restrict the number of neighbouring TV channels sent in parallel.

Globally, the scheme we propose offers interesting results. However, we have realised that user behaviour can vary significantly: it is true that many users enjoy zapping up and down, but others seem to be more targeted, zapping less. Therefore, while some users would benefit tremendously from using our scheme, others would see only a relatively small improvement. Therefore, as ongoing work we are considering other schemes that may be beneficial to different types of users. Some ideas we are currently testing are discussed in our concluding section (Section 8).

2. SWITCHING DELAY COMPONENTS

Users zapping in IPTV usually experience a couple of seconds' delay [13]. Henceforth we will call this the *normal IPTV delay*. This delay can be divided in three broad components: network delay, STB processing delay, and synchronisation and buffering delay.

Network delay can be subdivided in multicast leave and join time delay, which is the delay a multicast receiver experiences to leave the current multicast session and join the new one, and access link propagation delay. The sum of these types of delay is usually below 100-200ms[14, 16, 9, 1, 2], so it is a relatively unimportant contributor to the overall delay. STB processing delay is also relatively low, usually not exceeding 50ms[14].

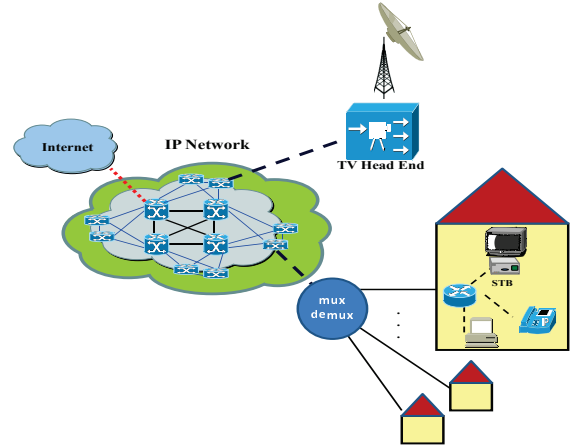


Fig. 2. IPTV network

The main contributors to IPTV switching delay are, then, stream synchronisation (waiting for anchor frames to allow decoding of the video signal) and buffering (needed to compensate for packet loss and jitter), adding up to 1 or 2 seconds on average[14, 16, 9]. The main concern in the industry and in the research community has been, in fact, to try to improve the performance on these two aspects [3].

As will be explained in detail in Section 5, in our scheme different channels are sent to the STB concurrently, hence synchronised and buffered together. Therefore, if the following request is for a channel already present in the STB, there is no network or synchronisation delay, and switching delay will be the result of STB processing only, thus virtually zero (i.e., way below the 430ms needed to guarantee an acceptable viewing experience).

3. DESCRIPTION OF AN IPTV NETWORK

At present, incumbent operator's IPTV networks are "walled gardens", well provisioned to guarantee the user experience required by TV viewers. Current networks use static IP multicast within a single network domain. The multicast protocol of choice is PIM-SM [7]. Also, all channels are distributed everywhere at all times, so the multicast flows are present at the edge of the network. This way, propagation delays for join requests are modest.

Figure 2 illustrates a typical IPTV service "walled garden" architecture. The TV head-end injects live TV streams encoded as IP packets to the network core. These packets travel through the network to routers that are connected to mux/demux equipment (e.g. DSLAMs in the case of a DSL network) that distribute traffic to several users and connect them to the high-speed backbone. The TV channels are distributed from the TV Head-end to these edge nodes through bandwidth-provisioned, static, multicast trees. The network is dynamic exclusively between these nodes and the customers,

where multicast trees are extended or pruned based on the channel switching signals. The STB translates a channel switching event from a user remote control into a pair of Internet Group Management Protocol (IGMP) messages: one to inform of the user's departure from the multicast group of a TV channel, and the other to join another multicast group of the new channel.

4. DESCRIPTION OF THE TRACES

The evaluation of the schemes presented in this paper is made by trace-driven simulations. The dataset used for this purpose is from a commercial, nationwide, IPTV service. The logs were collected over a six month period. The traces scale to 255 thousand users, 623 DSLAMs, 150 TV channels, and 11 regions.

The trace includes all switching events during this 6-month period, and each event comprises the following information:

- Timestamp in units of seconds;
- DSLAM ID;
- Set-top-box (STB) ID;
- IP address of the multicast group (TV channel);
- Type of switching event: join or leave channel.

With this data it is possible to obtain information on all switching events performed by one specific STB: when a user started watching a specific channel, and when it left it to watch another. Note that all data was anonymised for the purpose of this study. No information that could directly or indirectly identify individual subscribers was included.

5. PROPOSAL: CHANNEL SMURFING

Our proposal, **channel smurfing**, is very simple: judiciously Select an extra set of TV channels and send all these channels to the Set Top Box together (as if they were “Merged”), with the objective of providing an improved TV sURFING experience to the user.

The extra channels to send are the neighbouring channels (i.e., channels adjacent to the requested one), and we send them during zapping periods only. These neighbouring channels are synchronised and buffered concurrently with the requested one. Therefore, if the user decides to switch to any of these neighbouring channels, the switching delay is virtually zero, and user experience is not affected.

We consider two dimensions in our analysis. The first is the amount of time the neighbouring channels are sent concurrently to the STB. The scheme leading to the best results would be to send these channels at all times. However, this

would be inefficient, as it would unnecessarily increase access link bandwidth utilisation. Therefore, in the schemes proposed we maintain neighbouring channels in the STB during a small period, between 10 seconds and 2 minutes. The second dimension is the number of neighbouring channels to send concurrently. Considering the limitations of STB processing and of access link bandwidth, we decided to restrict the number of concurrent channels to a minimum of 2 and a maximum of 10. In fact, the gain of sending more channels would be negligible, as can be inferred from Figure 1.

5.1. Ideal predictor

In our scheme we predict the next channel to be one of the requested channel's neighbours. Sometimes the prediction is wrong, so it is important to understand how our scheme compares with an ideal predictor. An ideal predictor is one that knows to which TV channel the user will switch next. This will be used as benchmark for comparison.

6. EVALUATION

To evaluate the proposed scheme we have performed trace-driven simulations on the large dataset presented in Section 4. All the results presented in this section stem from the analysis of the full data set.

Some details need clarification beforehand. Sometimes, users zap linearly, from one channel to the next, in a very swift manner (in less than the *normal IPTV switching delay*, which we will consider in this paper to be 2 seconds, as explained in Section 2). In these cases, the STB may not have time to synchronise to the requested channel, or to any of its neighbours. However, all these channels are already in synchronisation mode. Therefore, if after this the user does switch to a channel already in the STB, the switching delay, albeit not being zero, will be less than the normal delay. In this case, we say the user experienced *partial delay* (some value between zero and the *normal delay*).

Recall that in our simple scheme we send the neighbouring channels together with the requested channel. These are synchronised and buffered together, but the neighbouring channels stay in the STB for a limited, predefined period. We call this the **concurrent channel time**. After this time, the STB sends IGMP leave requests and the neighbouring channels are removed. We evaluated our scheme for values of the *concurrent channel time* equal to 10, 30, 60, 120 seconds. For comparison purposes, we also consider the case where the neighbours are never removed from the STB (they are “always” there until the next switching event occurs). Since there may be limitations on the number of neighbouring channels the STB can process, or that the access network link can accommodate, we decided to restrict the **number of neighbour TV channels** to 2, 4, 6, 8, and 10 neighbours.

So, in summary, when the user switches to a particular TV channel, one of these things can occur:

1. If the time between two user switching events is above the concurrent channel time, no extra channel is in the STB, and therefore the user will experience the *normal switching delay*.
2. If the time between two user switching events is below the concurrent channel time, there are extra channels in the STB. So, one of three things can happen:
 - a If the user switches to a channel different from one of the neighbours present in the STB, it will experience the *normal delay*.
 - b If the user switches to one of the neighbours present in the STB, and if the time between two user switching events is above or equal 2 seconds, the user experiences virtually no delay. This is due to the fact that the channel is in the STB, and is already synchronised.
 - c If the user switches to one of the neighbours requested by the STB before, but the time between two user switching events is below 2 seconds, the user experiences *partial delay*. As explained, this is due to the fact that the channel was requested by the STB, but the user zapped rapidly, so it was not yet synchronised.

6.1. Results

Figure 3 illustrates the percentage of switching events that experienced virtually zero, or partial, delay. The main conclusion we can draw is that by using this very simple scheme we can reduce zapping delay to a significant number of switching events. For example, by sending only 2 neighbours, the previous and next channel, for only one minute, we reduce the delay to virtually zero to around 45% of the switching events (2 to 3% will experience partial delay).

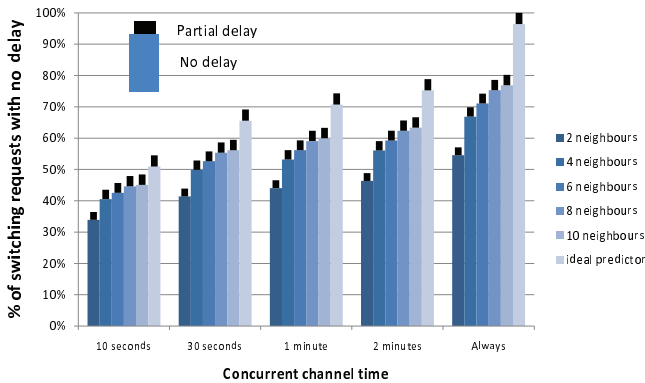


Fig. 3. Requests that experience no (or partial) delay

In Figure 4 we consider only zapping periods. We believe focusing in zapping periods in detail is important, because arguably it is in these periods that the user expects a swift zapping experience. We consider that a user is in “zapping mode” if the time between consecutive switching requests is less than one minute (in accordance with previous research [4]). Therefore, all events for which the switching time was above one minute were removed (for this reason, we logically do not include the results with *concurrent channel time* above one minute). We can see that, for example, by sending only 4 neighbours for one minute, more than 70% of the switching requests during zapping periods will experience virtually no delay.

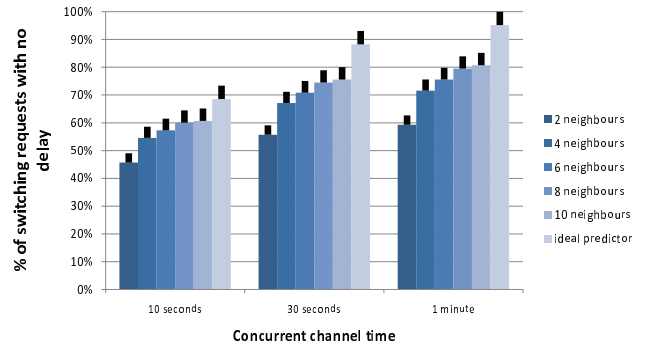


Fig. 4. Requests that experience no (or partial) delay during zapping only

It is important to note that an ideal predictor (i.e., one that would always know exactly to what channel the user would switch to), would not do much better than most of the schemes we tested, as can be attested from the previous figures. To make this comparison more clear, in Figure 5 we show the distance of each of our schemes to the ideal predictor. We define distance as the difference between the percentage of requests the ideal predictor would benefit and the percentage of requests each of our schemes can affect positively. It is interesting to note that our simple schemes produce results that are not very distant from the ideal case. An ideal scheme would not perform much better than a scheme that sends 6 or 8 neighbours, for instance.

Currently, IPTV service providers distribute TV content in SD format encapsulated as MPEG2 streams, so each TV channel needs 4Mbps of guaranteed bitrate. As explained before, it is important that the access link can accommodate the distribution of several TV channels concurrently. We assume this is the case. However, when several TV channels are sent in the access link other broadband applications (P2P, web browsing, etc.) will be affected, so it is important to quantify its impact. Therefore, in Figure 6 we illustrate the average bandwidth consumption, to understand the impact our scheme will have in this particular. We can observe that by limiting the *concurrent channel time* to zapping periods only, the av-

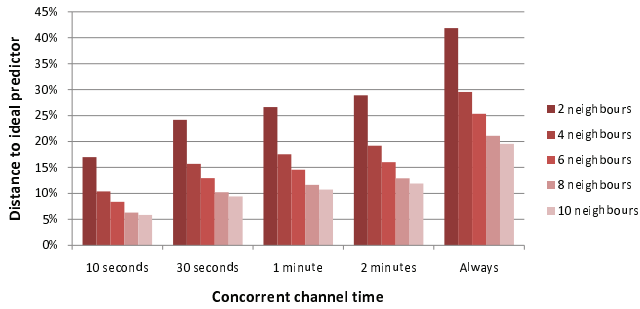


Fig. 5. Distance to an ideal predictor

erage bandwidth is very close to the 4 Mbps current IPTV services usually require. This is due to the fact that zapping periods are relatively rare events during the course of a normal day. So, even though the access link will have to accommodate peaks of high bandwidth consumption during zapping periods, these average out during the course of the day.

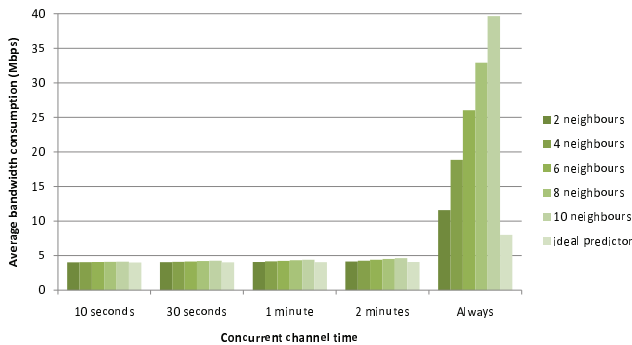


Fig. 6. Bandwidth consumption

7. RELATED WORK

In conventional analog broadcast all channels are received simultaneously, and the zapping times are typically less than 200 ms [2]. With such low latency, user experience is not affected. However, with the digitisation, compression, and encryption of content, zapping times have increased, and this is today one of the most significant concerns in IPTV systems. Therefore, there is a set of papers focused on this issue. Solutions to this problem exist both at the video coding and processing level [9] and at the network level [6, 13]. [15] proposes a channel change mechanism where users leave multicast groups and receive unicast streams at higher than usual bandwidth to mitigate the problem. Begen et al. [2, 1] also offer a nice overview of this problem and describe how the Real Time Protocol (RTP) can help reduce channel-change times. [14] presents an overview of all IPTV network elements that influence delay, and is also a nice survey on some of the techniques developed to reduce zapping delay.

In [5] the authors propose a similar scheme to the one presented here. In their scheme each STB simultaneously joins additional multicast groups along with the channel that is requested by the user. The main problem is that no performance evaluation was given. Also, this solution is not bandwidth-efficient, and hence other solutions to the same problem were presented more recently. Sun et al. [16] have analysed a multi-channel delivery system for IPTV, although with a focus on bandwidth demand and the effect on channel change time. They developed a mathematical model with many assumptions, some of which have proved wrong recently [12]: by analysing real IPTV datasets, these two studies proved that channel surfing behaviour can not be modelled with the Poisson models the authors of [16] used.

Anyway, in addition to having a different focus, we would like to emphasise that the main differentiating factor of the work we present here is that we performed trace-driven simulations on real data - a large scale dataset from a successful IPTV service - which offers us confidence in the results presented. We therefore believe our conclusions can motivate IPTV service providers to implement the proposed scheme.

8. CONCLUSIONS AND ONGOING WORK

In this paper we present a simple scheme to reduce channel switching delay in IPTV distribution networks. By making use of the fact that TV users enjoy zapping linearly - i.e., they tend to switch up and down using the remote control - we propose to send the neighbouring channels (channels adjacent to the requested one) to the Set Top Box (STB) during zapping periods, together with the one requested. Thus, in the event of the user switching next to any of these channels, switching latency is virtually eliminated.

To evaluate this scheme we have performed trace-driven simulations in a large dataset, comprising 255 thousand users, 150 TV channels, and covering a 6-month period. The main conclusion of our analysis is that by using our simple scheme the zapping delay can be virtually eliminated for a very significant percentage of channel switching requests. As an example, by sending the previous and the next channel concurrently with the requested one, for only one minute after a zapping event, switching delay is eliminated for around 45% of all channel switching requests. This figure jumps to 60% if we consider zapping periods only. Furthermore, we have also compared this simple scheme with an ideal predictor, and have realised that our proposal's performance is close to the optimal case. Finally, it is important to underline that the use of our scheme only negligibly increases the average bandwidth utilisation in the access link.

Globally, we believe the scheme we propose offers interesting results. However, we have realised that user behaviour can vary significantly. To attest this, we invite the reader to look at Figure 7. In here we show the results of using our scheme, but instead of the average, as before, we

now present the median, 5th and 95th percentile, to understand how our scheme would perform on a per-user basis. It is clear that some users would benefit tremendously from using our scheme (the “linear zapping fans”), but others would experience only a relatively small improvement (the “targeted” users).

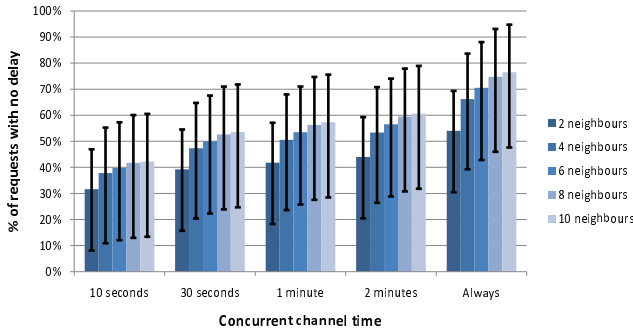


Fig. 7. Variance of the results

Therefore, as ongoing work we are considering other schemes that may be beneficial to different types of users. We are currently evaluating a scheme where the most popular channels are sent together with some neighbours. It is known [4, 11] that some channels are extremely popular, so the probability of a user switching to such channel is anticipated to be higher. More importantly, to try to reduce the variance in Figure 7 we believe it is necessary to devise more personalised schemes. One idea we are testing is to maintain, in each STB, information on both the probability of the user zapping linearly and the probabilities of him or her switching to particular channels. This information will allow us, for instance, to tag the user as “linear zapping fan” or “targeted”, and use diverse strategies accordingly.

9. REFERENCES

- [1] Ali C. Begen, Neil Glazebrook, and William Ver Steeg. Reducing channel-change times in IPTV with real-time transport protocol. *IEEE Internet Comput., Special Issue on IPTV*, 13/3:40–47, May/June 2009.
- [2] Ali C. Begen, Neil Glazebrook, and William Ver Steeg. A unified approach for repairing packet loss and accelerating channel changes in multicast IPTV. In *Proc. IEEE Consumer Communications and Networking Conf. (CCNC), Special Session on IPTV towards Seamless Infotainment*, 2009.
- [3] Yigal Bejerano and Pramod Koppol. Improving zap response time for IPTV. In *Proc. INFOCOM*, 2009.
- [4] M. Cha, P. Rodriguez, J. Crowcroft, S. Moon, and X. Amatriain. Watching television over an IP network. In *Proc. IMC*, 2008.
- [5] C. Cho, I. Han, Y. Jun, and H. Lee. Improvement of channel zapping time in IPTV services using the adjacent groups join-leave method. In *Proc. 6th International Conference on Advanced Communication Technology*, 2004.
- [6] N. Degrande, K. Laevens, D. D. Vleeschauwer, and R. Sharpe. Increasing the user perceived quality for IPTV services. *IEEE Communications Magazine*, 46(2):94–100, February 2008.
- [7] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei. Protocol independent multicast-sparse mode (PIM-SM): Protocol specifications. In *RFC 2362*, 1998.
- [8] Organization for Economic Cooperation and Development. <http://www.oecd.org/>.
- [9] H. Fuchs and N. Farber. Optimizing channel change time in IPTV applications. In *Proc. IEEE Int. Symp. Broadband Multimedia Systems and Broadcasting*, 2008.
- [10] R. Kooij, K. Ahmed, K. Brunnström, and K. Acreo. Perceived quality of channel zapping. In *proceedings of the fifth IASTED International conference Communication Systems and Networks*, pages 155–158. Citeseer, 2006.
- [11] T. Qiu, Z. Ge, S. Lee, J. Wang, Q. Zhao, and J. Xu. Modeling channel popularity dynamics in a large IPTV system. In *Proc. ACM SIGMETRICS*, 2009.
- [12] Tongqing Qiu, Zihui Ge, Seungjoon Lee, Jia Wang, Jun Xu, and Qi Zhao. Modeling user activities in a large IPTV system. In *Proc. ACM IMC*, 2009.
- [13] C. Sasaki, A. Tagami, T. Hasegawa, and S. Ano. Rapid channel zapping for IPTV broadcasting with additional multicast stream. In *Proc. IEEE Int. Conf. Communications (ICC)*, 2008.
- [14] Peter Siebert, Tom N. M. Van Caenegem, and Marcelo Wagner. Analysis and improvements of zapping times in iptv systems. *IEEE Transactions on Broadcasting*, 55(2):407–418, June 2009.
- [15] D. Smith. IPTV bandwidth demand: Multicast and channel surfing. In *Proc. INFOCOM*, 2007.
- [16] W. Sun, K. Lin, and Y. Guan. Performance analysis of a finite duration multichannel delivery method in IPTV. *IEEE Transactions on Broadcasting*, 54(3/1):419–429, September 2008.