A Unified Approach for Repairing Packet Loss and Accelerating Channel Changes in Multicast IPTV

Ali C. Begen, Neil Glazebrook and William Ver Steeg

Cisco Systems
San Jose, CA 95134 USA
{abegen, nglazebr, billvs}@cisco.com

***Abstract*—In multicast-based IPTV distribution networks, when an IPTV viewer tunes to a new channel, the IP set-top box (STB) joins a new multicast session. Upon join, the IP STB needs to acquire and parse certain key information before it can process any data sent in the multicast session. Depending on the join time, length of the key information repetition interval, size of the key information as well as the application and transport properties, the time lag before the IP STB can usefully consume the multicast data, which we refer to as the synchronization delay, varies and may be large. This is an undesirable phenomenon and degrades the quality of experience perceived by the IPTV viewers. In this study, we describe a unified standards-based approach that can be used both to repair lost packets in real time and reduce the synchronization delay.**

I. Introduction

In conventional analog broadcast where the tuners can locally lock to a signal in a single frame time and render it immediately, the zapping times are typically less than 200 ms and the zapping demand does not have any impact on the zapping times. With the introduction of digital cable and satellite broadcast, the zapping times have increased due to the compression and encryption applied to the content. In IPTV, the zapping times are affected by not only the compression and encryption operations, but also the network operations. These operations are needed to support several key components of an IPTV distribution network such as admission control and multicast. Depending on the load in the system, the network operations consume a variable delay (See [1, 2] for models that can be used to calculate the impact of channel changes in the network). Thus, in an IPTV network, the zapping times are also affected by the zapping demand. In order to preserve the viewers' quality of experience, IPTV providers have to take network characteristics into account and strive to offer short and near-constant zapping times despite the potentially high demand during peak channel change times, such as at the top of the hour.

There are currently several proposals from industry and academia to accelerate the channel changes for IPTV. Due to the lack of space, we only provide the references to these proposals in this study. These fall into several broad categories. There are solutions based on video coding and processing [3, 4] as well as solutions at the network level [5–9]. Hybrid proposals that combine video and network-level solutions also exist [10–13].

While the variety of solutions proposed so far shows that the problem of accelerating the channel changes in IPTV can be approached in many ways, the scalability and the operational aspects of the solution become crucial in large-scale IPTV deployments. IPTV providers require a scalable channel change acceleration solution that minimizes their capital and operational expenditures while providing a high quality of experience, thus reducing customer churn [14, 15].

With the scalability goal in mind, there is an ongoing effort to develop a standard solution for reducing the synchronization delay in multicast-based applications in the Audio/Video Transport (AVT) Working Group of the Internet Engineering Task Force (IETF). This solution uses the existing protocol mechanisms for repairing packet losses in multicast sessions. [16] explains the details of this solution and also provides the necessary signaling mechanisms for the Session Description Protocol (SDP) [17]. In this study, we introduce the concepts and motivation behind the feedback-based loss-repair methods, describe how this solution can be used to accelerate channel changes in IPTV and provide initial experimental results.

Let us start with the basics of the synchronization problem in a multicast session. Consider a single-source multicast (SSM) session where there is a source and one or more receivers that randomly leave and join the multicast session. Suppose that the multicast flow carries an MPEG2 Transport Stream (MPEG2-TS) that is encapsulated in a Real-time Transport Protocol (RTP) [18] stream. When a receiver joins the multicast session, it must first acquire certain key information before starting to process the data sent in the multicast session. This key information is conventionally sent periodically in the multicast session and usually consists of items such as a description of the data schema, encryption information including keys, as well as any other information required to process the data in the multicast flow.

Upon join, the receiver has no control over what point in the flow is currently being transmitted. The receiver might join the session right before the key information is sent in the session. In this case, the required waiting time is minimal. On the other hand, the receiver might join the session right after the key information has been transmitted. In this case the receiver has to wait for the key information to appear again in the stream before it can start processing any multicast data.

978-1-4244-2309-5/09/$25.00 ©2009 IEEE

The key information may not be contiguous in the flow but dispersed over a large period, which forces the receiver to wait for all of the key information to arrive before it can usefully consume the multicast data.

The overall effect of waiting for the key information and waiting for various buffers, *e.g.*, de-jittering, loss-repair and application-level buffers, to fill to the required level is that the receivers may experience large delays in data processing. We refer to the difference between the time a receiver joins the multicast session and the time the receiver acquires all the necessary key information as the synchronization delay. The synchronization delay may not be the same for different receivers; it varies depending on the join time, length of the key information repetition interval, size of the key information as well as the application and transport properties. The varying nature of the synchronization delay adversely affects receivers that frequently switch among multicast sessions. Note that this synchronization delay comprises a substantial portion of the channel change time in the IPTV application.

Suppose that either the multicast source retains the key information for a period of time after transmission, or an intermediary network element joins the multicast session, continuously caches the key information as it is sent in the session and acts as a feedback target [19] for the session. When a receiver wishes to join the same multicast session, instead of simply issuing an Internet Group Management Protocol (IGMP) [20] Join message, it sends a request to the feedback target address for the session asking for the key information. The feedback target starts a unicast retransmission RTP session and sends the key information to the receiver over that session. If there is spare bandwidth, the feedback target may also burst the key information at a faster than its natural rate. As soon as the receiver acquires the key information, it can join the multicast group and start processing the multicast data. This method potentially reduces the synchronization delay.

A primary design goal in this solution is to use the existing tools in the RTP and RTP Control Protocol (RTCP) family. This improves the versatility of the existing implementations and promotes faster deployment and better interoperability. To this effect, we use the unicast retransmission support of RTP [21] and RTCP signaling [22]. These facilities are conventionally used to recover lost packets for receivers already joined a multicast session. The receivers report losses using a negative acknowledgment (NACK) packet, and the feedback target responds with the missing data through unicast transmission of the missing data encapsulated in retransmission format packets. For rapid synchronization, we use this mechanism to provide the key information to receivers prior to joining the multicast session. In fact, a single RTP session is used for both rapid synchronization and loss repair. This helps the scalability of the IPTV network.

We continue our discussion with a description of the delay components in multicast flows and video systems in Sections II and III, respectively. We outline the rapid synchronization method in Section IV. Section V presents experimental results and we summarize our conclusions in Section VI.

II. Delay Components in Multicast Flows

In multicast delivery systems, there are three major elements that contribute to the overall synchronization delay when a receiver switches from one multicast session to another one. We briefly discuss them below.

Multicast switching delay is the delay that a multicast receiver experiences to leave the current multicast session (if there is any) and join the new multicast session. In typical systems, the multicast join and leave operations are handled by a group management protocol. For example, the receivers participating in a multicast session typically use IGMP [20]. In IGMP, when a receiver wants to join a multicast session, it sends an IGMP Join message to its upstream router and the routing infrastructure sets up the multicast forwarding state to deliver the packets of the multicast session to the new receiver. Depending on the proximity of the upstream router, the current state of the multicast tree, the load on the system and the protocol implementation, the join times may vary. Typical systems provide join latencies that are usually less than 200 ms. If the receiver had been participating in another multicast session before joining the new session, it needs to send an IGMP Leave message to its upstream router to leave the session. The leave times are usually smaller than the join times, however, it is possible that a Leave or Join message may get lost, in which case the multicast switching delay will inevitably increase.

Key information latency is the time it takes the receiver to acquire the key information. It is highly dependent on the proximity of the actual time the receiver joined the session to the next time the key information will be sent to the receivers in the session, whether the key information is sent contiguously or not, and how big it is. For some multicast flows, there is a little or no interdependency in the data, in which case the key information latency is nil or negligible. However, for multicast flows that carry compressed audio/video, there may be a high degree of interdependency. For these flows, the key information latency is usually quite large and a major contributor to the overall synchronization delay.

Buffering delays in the IP STB are driven by the way the application layer processes the payload. In many multicast applications, an unreliable transport protocol such as UDP [23] is often used to transmit the data packets, and the reliability, if needed, is usually addressed through other means such as Forward Error Correction (FEC) or retransmission. These loss-repair methods require buffering at the receiver side to function properly. In many applications, it is also necessary to de-jitter and reorder the incoming data packets before feeding them to the application. The de-jittering process also increases the buffering delays. Besides these network-related buffering delays, there are also specific buffering needs of the individual applications. For example, MPEG decoders require a significant amount of content to be available in the decoder buffers prior to starting to decode the content.

III. Delay Components in Video Systems

For well-engineered multicast-based video delivery systems, the multicast switching delay, *i.e.*, the time required to leave the previous multicast session and join the new session, is not the primary contributor to the overall synchronization delay. The multicast flows are typically already present at the edge or deep in the network, the propagation delays for join operations are modest, and the multicast routers can process the Join and Leave messages quickly. Even if the edge multicast router is not currently a member of the requested multicast session, the multicast routing control messages propagate through the network rapidly and trees are built without experiencing large delays. Even in cases where a number of tree branches need to be built to the edge multicast router, this cost is frequently amortized over a large number of receivers such that only the first receiver joining the group experiences the increased delay. Furthermore, this delay can be eliminated at the cost of extra bandwidth in the core network by having the edge routers statically join the set of sessions they expect receivers to be interested in. These techniques can provide a well-bounded multicast switching delay. Once the join operation completes and a receiver starts receiving media content for the first time in a multicast session, it often experiences a considerable amount of key information latency and buffering delays. We discuss the details of these delay elements in the context of MPEG2-TS.

A. Overview of MPEG2 Transport Streams

MPEG2-TS [24] is an encapsulation method that multiplexes digital video and audio content, together with ancillary metadata to produce a multiplexed stream with internal synchronization. It describes the schema of the audio and video content and the in-band control information, which are carried in their respective Elementary Streams (ES). MPEG2-TS is ubiquitous in broadcasting over both terrestrial and satellite networks, and can carry many kinds of media, including MPEG2 and MPEG4/AVC-encoded content.

Program Specific Information (PSI) consists of the metadata carried in the transport stream. PSI includes Program Association Table (PAT), Conditional Access Table (CAT) and Program Map Table (PMT). PAT has the information about all the programs carried in the transport stream. It lists the Program IDs (PID) for all the PMTs, associating them with the individual programs. CAT defines the type of the scrambling used, and identifies all the PID values of the TS packets that contain the Entitlement Management Messages (EMM). In addition to containing the PID values of each ES associated with a particular program, PMT also includes private data associated with the program such as the PID value of the packet containing the Entitlement Control Messages (ECM). The data contained in the EMM and ECM messages are vital in descrambling the encrypted content. Note that PSI is carried in clear and is never scrambled so that a receiver, which just started receiving the transport stream can process the PSI. The PAT, CAT and PMT tables must be parsed by the decoder in order to find the ES streams, private data as well as the encryption information for a given program.

B. Key Information Latency in Video Applications

1) PSI (PAT/CAT/PMT) Acquisition Delay: The video (and the audio as well) in an MPEG2-TS is self describing, so the receiver must parse certain control information in the PAT, CAT and PMT tables contained in the transport stream in order to know how to parse the rest of the stream.

Many video services employ content encryption, so the encryption keys must be parsed and provided to the decryption engine for decrypting the content. In order to enable various system elements to process video effectively, certain portions of the stream are left unencrypted. The PAT/PMT tables are always in the clear. The structure of the ECMs is also in the clear, although the ECM content which contains keying material is encrypted. The PSI information is repeated periodically in the transport stream, thus, when a receiver joins the multicast session, it needs to wait until the next time PSI is sent in the transport stream.

2) Random Access Point Acquisition Delay: Conventional MPEG2 video encoders structure their output as Groups of Pictures (GoP). Each GoP is typically encoded independently from other GoPs and starts with an intra-coded frame (I-frame) that does not have any reference to other frames, *i.e.*, an I-frame contains the representation of an entire picture and can be decoded independently. Thus, the start of an I-frame is said to be a Random Access Point (RAP). On the other hand, due to the temporal compression, rest of the frames in the same GoP may have references to the I-frame or other frames. Due to this interdependency among the frames, one generally has to receive certain elements of the GoP prior to decoding or rendering any part of the GoP. For example, the decoder can decode a frame that is dependent on two other frames only after these two frames are decoded. In commonly deployed systems today, GoP durations are between 500 ms and one second. However, more advanced codecs tend to use longer GoPs to gain from the encoding efficiency. When a receiver joins the multicast session, it needs to wait until the next RAP shows up in the multicast flow before it can start decoding. Since the frame that is currently being multicast does not depend on the join time, the average time a receiver waits for RAP, *i.e.*, the average RAP acquisition delay, is approximately half of the GoP duration. Hence, for longer GoPs, the RAP acquisition delay is proportionally longer.

AVC (also called MPEG4 part 10 or H.264) compression is very similar to MPEG2 compression, but offers more compression tools, including hierarchical GoPs. In a hierarchical GoP, the dependent frames of a GoP may reference the key frame at the start of this GoP and the key frame at the start of the next GoP. This additional dependency causes a longer RAP acquisition delay, as the decoder must receive two I-frames (spread between two logical GoPs) before decoding can commence. AVC also has the ability to insert Instantaneous Decoding Refresh (IDR) frames. Frames that follow an IDR frame cannot reference frames that precede an IDR frame.

IDR frames are useful for editing AVC streams, but typically do not appear often enough in streaming video to be useful for stream synchronization.

C. Buffering Delays in Video Applications

1) Network-Related Buffering Delays: In general, multicast-based video applications use an unreliable underlying transport protocol such as UDP to distribute the content to a large number of receivers. This is largely due to the fact that these applications are one way in nature and providing closed-loop reliability does not scale well when the number of receivers is large or the acceptable playout delay is small, or both. Rather, if there is a need for reliability, the applications may employ one or more loss-repair methods to recover the packets missing at each receiver (The Reliable Multicast Transport (RMT) Working Group has several standardized solutions for this problem. Refer to [25] for details). For example, FEC may be used proactively and/or on-demand to provide reliable transmission to a potentially very large multicast group in a scalable manner [26]. Similarly, retransmissions may be used in RTP-based SSM sessions where the retransmissions can be handled by local repair servers rather than the source itself. However, regardless of the type of the loss-repair method(s) adopted by an application, loss-recovery operations always require additional buffering at the receiver side. The amount of buffering increases with the FEC block size when FEC is used, and with the round-trip time between the receiver and the local repair server when retransmission is used.

Audio/video decoders demand almost jitter-free content. If any jitter is introduced during the transmission in the network or due to the loss-repair methods, the jitter has to be smoothed out before the content is fed to the decoder. This is called de-jittering and it usually adds to the buffering delay.

2) Application-Related Buffering Delays: The application buffering requirements for MPEG-encoded content are quite rigorous, particularly for the MPEG-based video applications. Video compression devices apply more bits to represent certain scenes than they do for other scenes. A very complex scene (individual picture) requires considerably more information than a simple scene. Furthermore, pictures that are entirely intra-coded, *e.g.*, I-frames, consume more bits compared to pictures that make use of predictive coding. Each scene is shown by the decoder at a certain fixed frame rate. Since some scenes are comprised more bits than other scenes, the output rate of the decoder buffer is usually variable. However, the network flow is typically constant bitrate (CBR) or capped variable bitrate (Capped VBR). The net effect is that the input rate to the decoder buffer is close to constant, but the output rate is highly variable.

The video encoders keep track of the decoder buffer size, and use this information to regulate the temporal compression. This forces the decoder buffer to *breathe*. In order to avoid underflow, the decoder buffer must fill up to a certain level prior to starting to decode and play the content. The decoder buffer size required to avoid underflow is dependent on the encoder, and the encoder signals the decoder buffering requirements in-band. Typical decoder buffer requirements for MPEG2 content range from one second to two seconds. However, MPEG4/AVC encoders usually tend to use more temporal compression, and thus require a larger buffer at the decoder side. This consequently increases the buffering delays.

IV. Rapid Synchronization

The flow diagram for rapid synchronization is sketched in Fig. 1, where we have an RTP Sender, RTP Receiver (RR) and a Retransmission Server (RS) that provides the support for rapid synchronization. RS semi-permanently joins the multicast session and receives the RTP streams it wishes to cache so that it can perform the retransmission functions. For the rapid synchronization support, it also parses the incoming streams, looking for the key information. Note that while a retransmission server close to the edge of the network offers several advantages and is usually preferred, it might also be physically co-located with the media source, or anywhere in the path from the media source to a subset of the receivers.

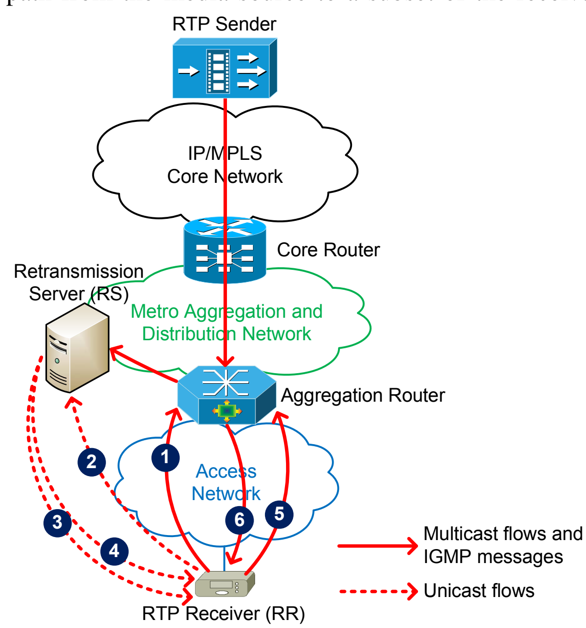

Fig. 1. Flow diagram for rapid synchronization.

We now outline the rapid synchronization algorithm step by step. For the encoding of the protocol messages, refer to [16].

1) RR sends an IGMP Leave message [20] to its upstream multicast router to leave the current multicast session.
2) RR sends a specific feedback message to the feedback target asking for rapid synchronization for the session it wants to join. In this feedback message, RR may also specify the minimum amount of data it requires from RS and optionally the maximum amount of data it can receive from RS. This information is used by RS to prepare a rapid synchronization that conforms to RR's buffer requirements.
3) RS receives the feedback message and decides whether to accept the rapid synchronization request or not. If

RS accepts the request, it sends a message to RR that describes the burst that RS will generate and send, including the indication when RR should switch to the multicast flow. If RS denies the request, it informs RR immediately so that RR can join the RTP multicast session.

4) If RS accepts the rapid synchronization request, it transmits the unicast RTP burst data and any additional message(s) needed to carry the key information that is not provided in the unicast RTP burst. The unicast burst continues at a higher than natural rate until the unicast burst catches up with the real-time multicast flow. The sustainable burst rate depends on the access network characteristics.
5) At the appropriate moment (as indicated or computed from the burst parameters), RR sends an IGMP Join message [20] to its upstream multicast router for the new RTP multicast session.
6) RR starts receiving the multicast RTP stream.

It is possible that RR may decide to switch to a new multicast session while an earlier rapid synchronization request is still pending or active. In that case, RR cancels the pending/active rapid synchronization operation before sending a request for the new multicast session.

It is vital to observe that the steps described above for rapid synchronization use the existing protocols [18, 19, 21, 22] that are already being adopted for providing loss-repair capabilities to IPTV distribution networks. With this approach, rapid channel change support can be deployed quickly, while maintaining interoperability with the existing protocols and infrastructure.

V. Experimental Results

In this section, we provide experimental results collected in two different IP STBs, where the first IP STB performed non-accelerated channel changes, and the second IP STB used rapid synchronization to accelerate its channel changes. In the experiments, the channel change time refers to the time difference between the channel change request and the time the new channel is played out on the screen.

The experiments were conducted in a setup similar to the one sketched in Fig. 1. In the experiments, we used two different video streams. Both streams were generated from the high-detail and high-motion scenes of a movie. We used an AVC encoder to encode the content in standard-definition format at 2 Mbps and 30 fps. One of the streams had an average GoP size of 60 frames, *i.e.*, two seconds. We refer to this stream as the long-GoP content. The other stream had an average GoP size of 15 frames, *i.e.*, half a second, and is referred to as the short-GoP content. Both source streams were transmitted in 1356-byte RTP packets, each carrying seven 188-byte MPEG2-TS packets. During rapid synchronization, we used 20% additional bandwidth to emulate heavily loaded access networks. In each IP STB, we also used a loss-repair buffer of 500 ms (See Section III-C).

The results for the short-GoP and long-GoP contents are plotted in Fig. 2 and 3, respectively. The comparisons of the non-accelerated and accelerated channel changes for both sets of content are also tabulated in Tables I and II. The results show that rapid synchronization reduces the average channel change times by about 65% and worst-case channel change times by about 53%. The standard deviation of the channel change times is also reduced by up to 60% when rapid synchronization is enabled.

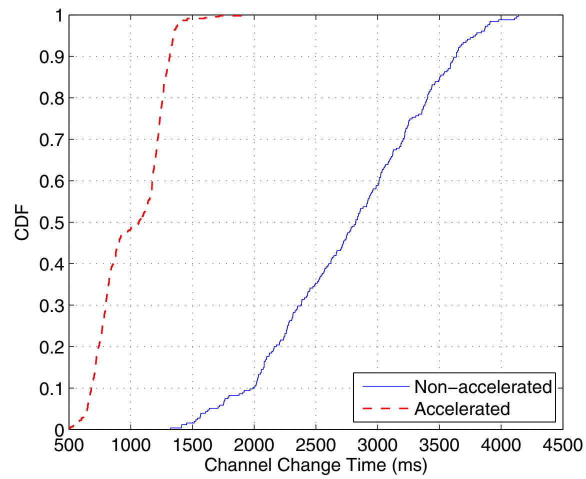

Fig. 2. Channel change time distributions for the short-GoP content.

TABLE I
Comparison of the non-accelerated and accelerated channel changes for the short-GoP content (In ms).

	Min	Mean	Standard Deviation	95^{th} Percentile	99^{th} Percentile	Max
Non-accelerated	1323	2785	645	3788	4101	4140
Accelerated	501	1009	260	1345	1457	1965

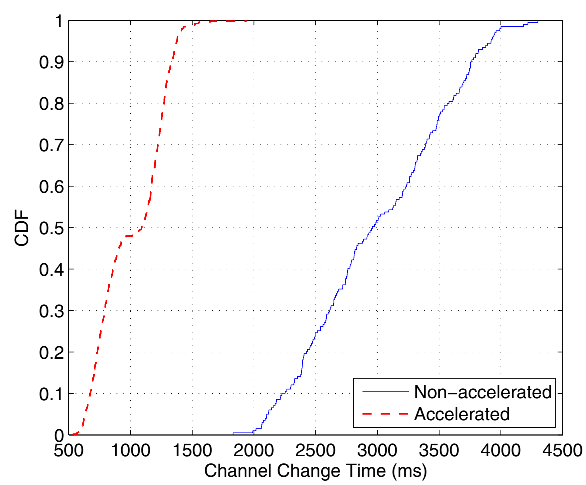

Fig. 3. Channel change time distributions for the long-GoP content.

TABLE II
COMPARISON OF THE NON-ACCELERATED AND ACCELERATED CHANNEL CHANGES FOR THE LONG-GOP CONTENT (IN MS).

	Min	Mean	Standard Deviation	95^{th} Percentile	99^{th} Percentile	Max
Non-accelerated	1831	3005	575	3920	4201	4300
Accelerated	536	1013	265	1377	1521	1937

We observe that the non-accelerated channel change times for the long-GoP content are larger than the ones for the short-GoP content. As explained in Section III-B, this is an expected result. However, the accelerated channel change times show a similar behavior for both contents. This is a major advantage of the rapid synchronization method: it can drastically reduce the RAP acquisition delay independent of the GoP size.

VI. CONCLUDING REMARKS

Two of the most important challenges an IPTV service provider faces today are recovering from packet loss in real time and reducing channel change times in a scalable manner. To address both challenges, this study describes a unified solution approach that is based on open protocols and standards. Early experimental results are encouraging and suggest that sub-second channel change times are achievable.

Due to the lack of space, we omitted a detailed mathematical analysis of rapid synchronization. However, in practice, the mechanism must minimize the number of duplicate packets that are received due to overlap of the unicast burst and multicast session while simultaneously avoiding any gaps in the received sequence numbers. Given the stochastic nature of the time it takes to process the IGMP messages as well as the time-varying nature of the available network resources, such algorithms are tricky to design and tune. Furthermore, shaping the unicast burst based on the receiver's requirements and the resource availability in the network path is also important. Our current work addresses these issues and we plan to report on that work in a future study.

Acknowledgments:

This work is based on the original architecture and technical approach designed by Dave Oran of Cisco Systems. We would like thank him for sharing his insights and comments as well as our colleagues in the VQE Team at Cisco Systems, especially Eric Friedrich and Yufeng Shan, for their constructive suggestions and contributions to this study.

REFERENCES

[1] D. E. Smith, "IP TV bandwidth demand: Multicast and channel surfing," in *IEEE Int. Conf. Computer Communications (INFOCOM)*, 2007.
[2] I. Kopilovic and M. Wagner, "A benchmark for fast channel change in IPTV," in *IEEE Int. Symp. Broadband Multimedia Systems and Broadcasting*, 2008.
[3] U. Jennehag, T. Zhang, and S. Pettersson, "Improving transmission efficiency in H.264 based IPTV systems," *IEEE Trans. Broadcast.*, vol. 53, no. 1, pp. 69–77, Mar. 2007.
[4] U. Jennehag and S. Pettersson, "On synchronization frames for channel switching in a GOP-based IPTV environment," in *IEEE Consumer Communications and Networking Conf. (CCNC)*, 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 *Int. Conf. Advanced Communication Technology*, 2004.
[6] D. boong Lee, H. Joo, and H. Song, "An effective channel control algorithm for integrated IPTV services over DOCSIS CATV networks," *IEEE Trans. Broadcast.*, vol. 53, no. 4, pp. 789–796, Dec. 2007.
[7] N. Degrande, K. Laevens, D. D. Vleeschauwer, and R. Sharpe, "Increasing the user perceived quality for IPTV services," *IEEE Commun. Mag.*, vol. 46, no. 2, pp. 94–100, Feb. 2008.
[8] C. Sasaki, A. Tagami, T. Hasegawa, and S. Ano, "Rapid channel zapping for IPTV broadcasting with additional multicast stream," in *IEEE Int. Conf. Communications (ICC)*, 2008.
[9] W. Sun, K. Lin, and Y. Guan, "Performance analysis of a finite duration multichannel delivery method in IPTV," *IEEE Trans. Broadcast.*, 2008, to appear.
[10] J. M. Boyce and A. M. Tourapis, "Fast efficient channel change [set-top box applications]," in *Int. Conf. Consumer Electronics (ICCE)*, 2005.
[11] H. Fuchs and N. Farber, "Optimizing channel change time in IPTV applications," in *IEEE Int. Symp. Broadband Multimedia Systems and Broadcasting*, 2008.
[12] Y.-H. Choi and Y. K. Seong, "Efficient channel navigation using circular buffering algorithm in digital TV," *IEEE Trans. Consumer Electron.*, vol. 54, no. 1, pp. 186–191, Feb. 2008.
[13] D. Banodkar, K. Ramakrishnan, S. Kalyanaraman, A. Gerber, and O. Spatscheck, "Multicast instant channel change in IPTV systems," in *Int. Conf. Communication Systems Software and Middleware (COMSWARE)*, 2008.
[14] R. Kooij, K. Ahmed, and K. Brunnstrm, "Perceived quality of channel zapping," in *IASTED Int. Conf. Communication Systems and Networks*, 2006.
[15] A. Takahashi, D. Hands, and V. Barriac, "Standardization activities in the ITU for a QoE assessment of IPTV," *IEEE Commun. Mag.*, vol. 46, no. 2, pp. 78–84, Feb. 2008.
[16] Internet Draft: draft-versteeg-avt-rapid-synchronization-for-rtp-00, "Unicast-based rapid synchronization with RTP multicast sessions." [Online]. Available: http://tools.ietf.org/html/draft-versteeg-avt-rapid-synchronization-for-rtp-00
[17] RFC 4566, "SDP: Session description protocol." [Online]. Available: http://www.ietf.org/rfc/rfc4566.txt
[18] RFC 3550, "RTP: A transport protocol for real-time applications." [Online]. Available: http://www.ietf.org/rfc/rfc3550.txt
[19] Internet Draft: draft-ietf-avt-rtcpssm-17, "RTCP extensions for single-source multicast sessions with unicast feedback." [Online]. Available: http://tools.ietf.org/html/draft-ietf-avt-rtcpssm-17
[20] RFC 3376, "Internet group management protocol, version 3." [Online]. Available: http://www.ietf.org/rfc/rfc3376.txt
[21] RFC 4588, "RTP retransmission payload format." [Online]. Available: http://www.ietf.org/rfc/rfc4588.txt
[22] RFC 4585, "Extended RTP profile for real-time transport control protocol (RTCP)-based feedback (RTP/AVPF)." [Online]. Available: http://www.ietf.org/rfc/rfc4585.txt
[23] RFC 768, "User datagram protocol." [Online]. Available: http://www.ietf.org/rfc/rfc768.txt
[24] "Generic coding of moving pictures and associated audio information: Systems," *ITU-T H.222.0*, 2006.
[25] Internet Draft: draft-ietf-rmt-pi-norm-revised-06, "NACK-oriented reliable multicast (NORM) protocol." [Online]. Available: http://tools.ietf.org/html/draft-ietf-rmt-pi-norm-revised-06
[26] A. C. Begen, "Error control for IPTV over xDSL networks," in *IEEE Consumer Communications and Networking Conf. (CCNC)*, 2008.