An Effective IPTV Channel Control Algorithm Considering Channel Zapping Time and Network Utilization

Hyunchul Joo, Hwangjun Song, Dai-Boong Lee, and Inkyu Lee

***Abstract*—This paper presents a simple but effective IPTV channel control algorithm that keeps channel zapping time in the tolerable range with high network utilization. The proposed algorithm controls channel zapping time by adjusting the number of broadcasting channels that are located close to users over IP networks and the number of additional I-frames inserted into each channel, based on the user's channel preference information. Finally, experimental results are provided to show the performance of the proposed algorithm.**

***Index Terms*—Broadcasting service system, channel zapping time, IPTV, network utilization.**

I. Introduction

THE DEMAND and interest of various services through the Internet have been increasing rapidly and the fast advance of broadband networking technology makes it feasible. Recently, Internet service providers launched the triple play service over the IP networks, which provides IPTV (Internet protocol television), VoIP (Voice over IP) and high speed Internet services simultaneously through a subscription line. This service model has the possibility of drawing new subscribers and increasing the average revenue per user through bundling data, voice, and video services together [1]–[3] and IPTV is already considered as one of key applications in the telecommunication market. In particular, it is believed that IPTV service presents an opportunity for telephone companies around the world to benefit from the video delivery over IP networks. Currently, there are active trials and commercial deployments across the world including North America, Europe, and Asia/Pacific [4].

IPTV service has several unique features compared to the traditional broadcasting services such as terrestrial, cable, and satellite broadcasting services in some aspects. In the traditional broadcasting services, the STB (Set Top Box) can immediately display the selected channel when user changes the watching channel because the STB receives all of the channels regardless of whether they are used or not as shown in (a) of Fig. 1.

Manuscript received October 23, 2006; revised December 12, 2007. This work was supported in part by MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment) (IITA-2006-C1090-0603-0045) and MIC and IITA through IT Leading R&D Support Project.

The authors are with the Department of Computer Science and Engineering, Pohang University of Science and Technology (POSTECH), Gyungbuk 790-784, Republic of Korea (e-mail: hwangjun@postech.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TBC.2008.915767

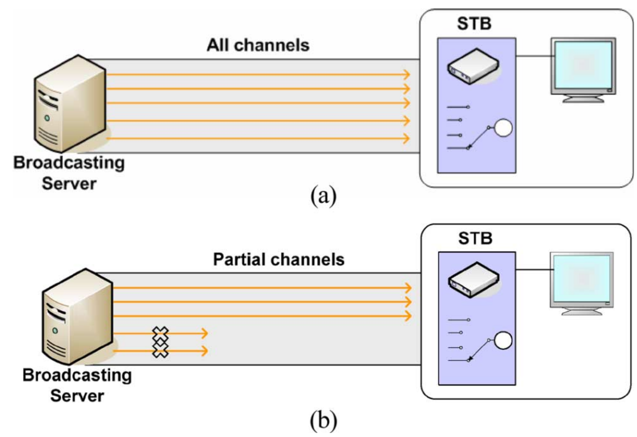

Fig. 1. Broadcasting service system architecture: (a) the traditional broadcasting service and (b) IPTV service.

However, IPTV service cannot transmit all the channels at the same time due to the lack of network bandwidth as shown in (b) of Fig. 1. Especially, it is expected that the occupancy of core network will greatly increase in the near future as the demand of various multimedia services such as VoD (Video on Demand), nPVR (network-based Personal Video Recorder), and other unicast video services significantly increases. Actually, some parts of core network are frequently congested. Therefore, IPTV service system must be designed to manage the bandwidth of the channels effectively because each channel necessitates high bandwidth.

IPTV service generally uses IP multicast technology to reduce the duplicate data transmission over the network, and thus computational overhead at router becomes larger as the number of broadcasting channels increases. Hence, only a part of channels is immediately available at STB and some delay is inevitable until the user watches the display of the selected channel when the selected channel is not available at STB, which is called as channel zapping time [5], [6].

Channel zapping time is considered to be one of the most important performance measures in IPTV service. Compared to the traditional broadcasting services, the relatively large channel zapping time is a big obstacle for the successful deployment of IPTV. Many research efforts have been devoted to find out how to reduce the channel zapping time [7]–[10]. In [7], Cho *et al.* presented the method of reducing the channel zapping time by sending the adjacent channels of the current channel in advance to the user. When the user requests an adjacent channel

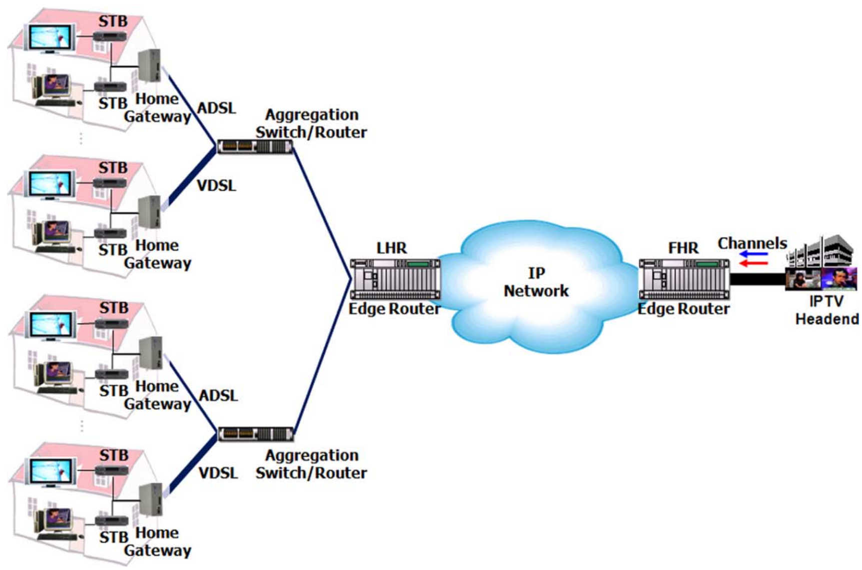

Fig. 2. IPTV service system architecture.

of the current channel, the user can watch the selected adjacent channel without network delay. In [8], the system is proposed to reduce the decoding delay and improve video coding efficiency at the same time. The encoder generates additional I-frames periodically encoded at a lower bit rate and transmits them with normal video frames. When the user requests a new channel, fast channel decoding is performed by the use of additional I-frames without waiting for normal I-frame. In [9], Kim *et al.* proposed an effective method to reduce channel zapping time by diminishing the GQI (General Query Interval) of IGMP (Internet Group Management Protocol) parameters. The smaller GQI makes fast channel change possible because the GQI is related to the join processing time of channel multicast group. SFCS (Synchronization Frames for Channel Switching) was proposed in [10] to increase bandwidth utilization by reducing the number of synchronization frames compared to the GOP (Group Of Picture) scheme. It can also decrease the decoding delay by increasing the frequency of synchronization frames.

In this paper, we present an effective IPTV channel control algorithm that statistically guarantees channel zapping time with high network utilization. One of unique features of the proposed algorithm is that we concurrently consider both broadcasting channel distribution state and video encoding structure as control variables to effectively guarantee channel zapping time. First of all, we present how to effectively reduce the network delay by positioning channels according to their popularity. Secondly, we consider how to efficiently decrease video decoding delay by adding extra I-frames to normal video frames. The rest of this paper is organized as follows. IPTV service system is studied in Section II, an effective channel control algorithm is presented in Section III, experimental results are provided in Section IV, and finally concluding remarks are given in Section V.

II. Review of IPTV Service System

The IPTV service structure under consideration is shown in Fig. 2. Users are connected to IPTV broadcasting system through STB, and the STB is attached to HG (Home Gateway). The HG plays as an agent between access network and home network, and controls the incoming IPTV channels as an IGMP proxy. Head-end receives all the broadcasting channels from external sources and multicasts them to users to improve the network utilization. LHR (Last Hop Router) is the closest edge router from home network and FHR (First Hop Router) is the closest edge router from head-end.

A. Channel Changing Mechanism

It is assumed that a user is watching channel #1 and wants to move to channel #2. Now, the user sends the channel change message to switch to channel #2 by using a remote controller. Then, STB sends an IGMP Leave message for channel #1 and an IGMP Join message for channel #2 to HG. As soon as HG receives the IGMP Leave message, it sends the IGMP group-specific Query message back to home network and waits for a while. If any response for channel #1 does not arrive, then HG leaves the multicast group for channel #1 via sending an IGMP Leave message to the upper-level router. When HG receives the IGMP Join message for channel #2, it immediately transmits channel #2 to the corresponding STB if already available. Otherwise, it sends an IGMP Join message to the upper-level router. These processes may increase channel zapping time. Now, we assume that every network device can support IP multicast over core network and IGMP snooping function [11], [12] over access network in the following. The above processes are summarized in Fig. 3.

B. Channel Zapping Time

Generally, the channel zapping time consists of command processing time, network delay, STB jitter buffer delay, and video decoding delay as shown in Fig. 3. Command processing time is a delay until the IGMP Join message is transmitted after user selects a new channel, network delay is the passing time that a requested stream is arrived after the transmission of IGMP Join message, STB jitter buffer delay is required to remove the unsmooth display caused by the delay jitter over the Internet,

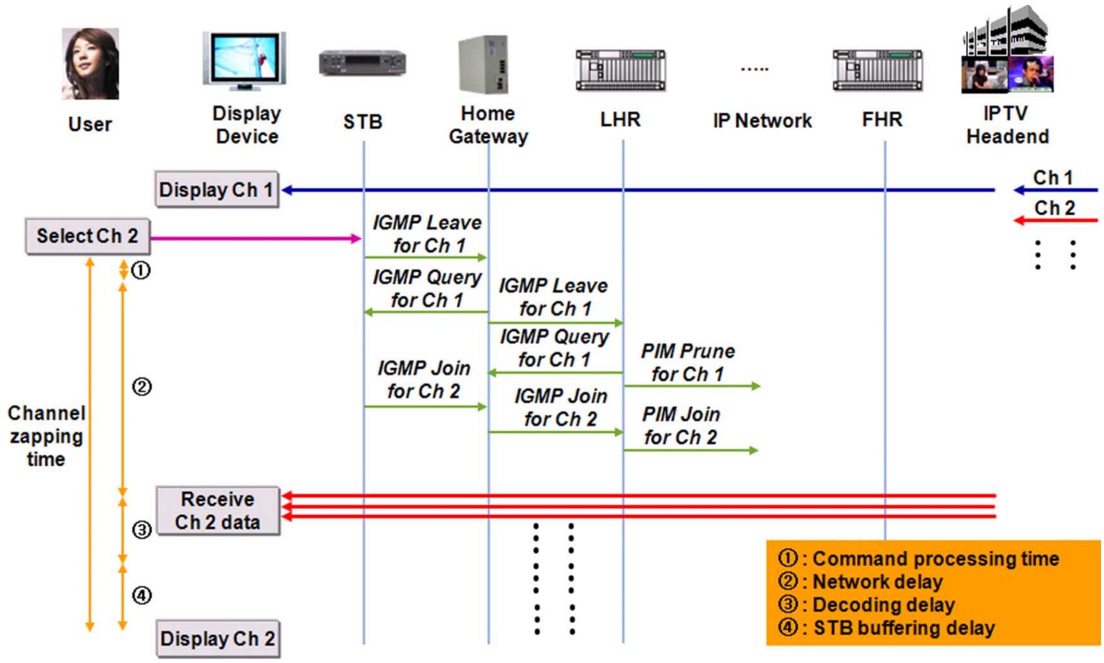

Fig. 3. IPTV channel changing scenario.

and video decoding delay is caused by the fact that compressed video cannot be decoded without I-frames.

Based on the network delay, channels can be classified into static channels and dynamic channels. Static channel is available at LHR, and thus the required network delay is relatively small but it wastes of network bandwidth when no one is watching the channel. On the other hand, dynamic channel is located at FHR and its network delay is big although the bandwidth waste is avoided. Video decoding delay is related to the encoding structure, and the maximum video decoding delay is the length of a GOP. To decrease the video decoding delay to be less than a GOP, additional I-frames must be transmitted through the fast channel change stream at the cost of the increased channel bandwidth.

III. Proposed Channel Control Algorithm

It is assumed that a broadcasting channel consists of normal stream and fast channel change stream as shown in Fig. 4. The proposed channel control algorithm is implemented to determine the channel distribution state and adjust the number of additional I-frames in the fast channel change stream to reduce the channel zapping time with high network utilization. In an engineering sense, the expected channel zapping time may be decreased when more popular channels are selected as static channels and more additional I-frames are inserted, but the network utilization is decreased. Hence, we pursue an effective trade-off between channel zapping time and network efficiency. The other factors of channel zapping time are beyond the scope of this paper.

A. Problem Description

We assume that IPTV service consisting of M channels is provided to N users in a cluster managed by a LHR, and users

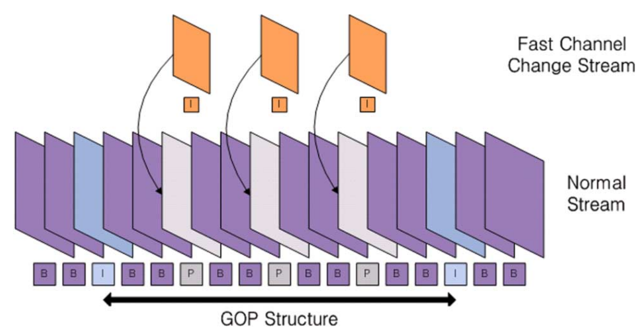

Fig. 4. Video encoding structure including fast channel change stream.

choose their channels independently. First of all, some symbol descriptions and problem formulation are given as follows.

$$\overrightarrow{BW} = (BW_1, BW_2, \ldots, BW_M),$$
$$\overrightarrow{BW}^I = \left(BW_1^I, BW_2^I, \ldots, BW_M^I\right),$$
$$\vec{p}_i = \left(p_i^1, p_i^2, \ldots, p_i^M\right),$$

where BW_j is the normal channel bandwidth required for the j_{th} channel, BW_j^I is the channel bandwidth for the fast channel change stream of the j_{th} channel, and $\vec{p}_i$ is the channel probability vector of the i_{th} user: p_i^j denotes the probability that the i_{th} user watches the j_{th} channel, which is determined to satisfy $\sum_{j=1}^{M}\sum_{i=1}^{N} p_i^j = 1$ and $\sum_{j=1}^{M} p_i^j = 1/N$. Now, the channel vector $\vec{s} = (s_1, s_2, \ldots, s_M)$ is defined by

$$s_j = \begin{cases} 1 & \text{if the } j_{th} \text{ channel is the static channel,} \\ \sum_{i=1}^{N} p_i^j & \text{otherwise,} \end{cases}$$

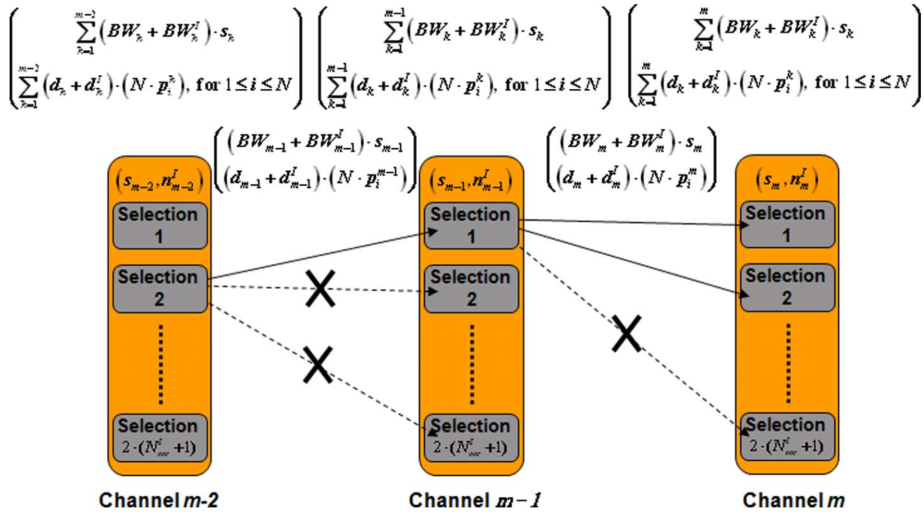

Fig. 5. Trellis for the optimal $\vec{s}^* = (s_1, s_2, \ldots, s_M)$ and $\vec{n}^* = (n_1^I, n_2^I, \ldots, n_M^I)$: the dot line does not satisfy the given delay constraint.

And network delay vector $\vec{d} = (d_1, d_2, \ldots, d_M)$ is characterized by

$$d_j = \begin{cases} d_{static} & \text{if the } j_{th} \text{ channel is the static channel,} \\ d_{dynamic} & \text{otherwise,} \end{cases}.$$

where $d_{dynamic}$ and d_{static} are average delay values of the dynamic channel and the static channel over the Internet, respectively (in general, $d_{dynamic}$ is much larger than d_{static}).

Now, the video decoding delay depends on the time interval of I-frames, but I-frames spend much higher bit rates than P- and B-frames. Hence, we treat the time interval of I-frames as a control variable. Under the assumption that I-frames are equally positioned in the fast channel change stream and encoded with the same bits R_I, the video decoding delay vector is described by

$$\vec{d}^I = (d_1^I, d_2^I, \ldots, d_M^I),$$
$$d_j^I = \frac{T_{gop}}{(n_j^I + 1)} \text{ for } 0 \leq n_j^I \leq N_{ccs}^I, \tag{1}$$

where n_j^I is the number of frames in the j_{th} fast channel change stream during a GOP, N_{ccs}^I is the maximum number of frames in each fast channel change stream during a GOP, and T_{gop} is the time interval of a GOP. The bandwidth for fast channel change stream is calculated by

$$BW_j^I = \frac{n_j^I \cdot R_I}{T_{gop}} \tag{2}$$

Then, we can formulate the given problem as follows.

Problem Formulation: Determine $\vec{s} = (s_1, s_2, \ldots, s_M)$ and $\vec{n} = (n_1^I, n_2^I, \ldots, n_M^I)$ to minimize

$$(\overrightarrow{BW} + \overrightarrow{BW}^I) * \vec{s}, \tag{3}$$
$$\text{subject to } (\vec{d}+\vec{d}^I)*(N\cdot\vec{p}_i) \leq D_{upper\_bound}$$
$$\text{for } 1 \leq i \leq N, \tag{4}$$

where $*$ denotes vector inner product and D_{upper_bound} is the tolerable upper bound of average zapping time. Practically, p_i^j may be approximately estimated by

$$p_i^j \approx \frac{t_i^j}{N \cdot T_{update}}, \tag{5}$$

where T_{update} is the fixed time interval and t_i^j is the time interval that the i_{th} user has watched the j_{th} channel for the fixed time interval.

B. Optimal Solution Based on Dynamic Programming

In this paper, Viterbi algorithm [13]–[15] is employed to obtain the optimal solution $\vec{s}^*$ and $\vec{n}^*$. The trellis of Viterbi algorithm is given in Fig. 5. As shown in the figure, each channel corresponds to a domain and each domain has $2\cdot(N_{ccs}^I + 1)$ selections. A selection (s_m, n_m^I) in the m_{th} domain denotes whether the corresponding channel is dynamic or static and the number of I-frames in the fast channel change stream. The number of possible paths is up to $\{2(N_{ccs}^I + 1)\}^M$.

When the cumulative state by the $(m-1)_{th}$ domain is $(\sum_{k=1}^{m-1}(BW_k + BW_k^I)\cdot s_k, \sum_{k=1}^{m-1}(d_k + d_k^I)(N\cdot p_i^k))$ for $1 \leq i \leq N$ and the selected control variable in the m_{th} domain is (s_m, n_m^I), the cumulative state by the m_{th} domain becomes $(\sum_{k=1}^{m}(BW_k + BW_k^I)\cdot s_k, \sum_{k=1}^{m}(d_k + d_k^I)(N\cdot p_i^k))$ for $1 \leq i \leq N$. For a fast pruning over trellis to reduce the unnecessary search, the delay constraint is checked at selected control variable during the process. If the interim delay $\sum_{k=1}^{m}(d_k + d_k^I)(N\cdot p_i^k)$ at the m_{th} domain over the trellis is already larger than D_{upper_bound} for each user, then the interim path is deleted. However, it still needs a considerable amount of computation. Thus, an effective fast channel control algorithm is studied in the following section.

C. Fast Channel Control Algorithm With a Low Computational Complexity

Here, we study how to obtain a near optimal solution $\vec{s} = (s_1, s_2, \ldots, s_M)$ and $\vec{n} = (n_1^I, n_2^I, \ldots, n_M^I)$ with lower computational complexity. $Delay\_over(\vec{s}, \vec{n})$ is defined as follows

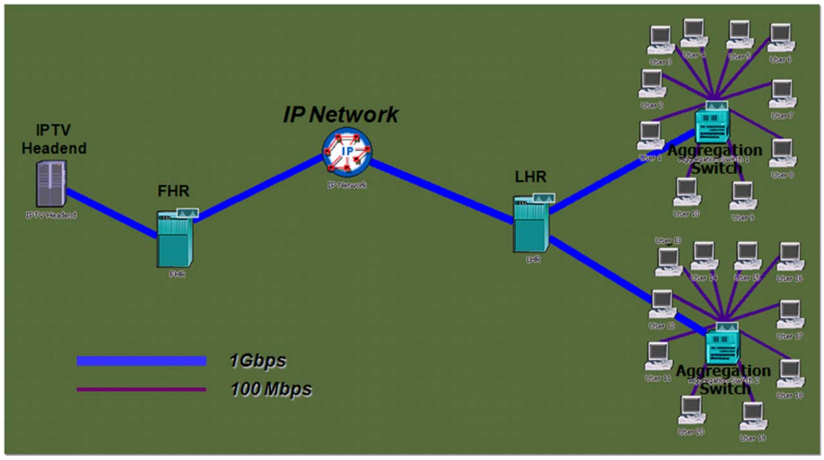

Fig. 6. OPNET simulation network configuration.

to consider the delay constraint of the above problem formulation.

$$Delay\_over(\vec{s},\vec{n}) = \sum_{i=1}^{N} MAX\left\{0, (\vec{d}+\vec{d}^{I})*(N\cdot\vec{p}_i) - D_{upper\_bound}\right\}.$$

The basic idea of the proposed fast algorithm is that channels are selected as the static or I-frames are inserted to decrease $Delay\_over(\vec{s},\vec{n})$ normalized by the increment of bandwidth by the largest amount. This process is repeated until (4) is satisfied. The proposed fast channel control algorithm is summarized as follows.

Step 1: At first, all channels are dynamic and n_j^I is set to zero for $1 \le j \le M$, and calculate $Delay\_over(\vec{s},\vec{n})$ when $s_j = \sum_{i=1}^{N} p_i^j$ for $1 \le j \le M$ and $\vec{n} = \vec{0}$.

Step 2: For the j_{th} channel $(1 \le j \le M)$, calculate the following function values.

$$Gain(\vec{s},\vec{n}) = \frac{\Delta Delay\_over(\vec{s},\vec{n})}{\Delta BW(\vec{s},\vec{n})}$$

$$\text{for } 0 \le n_j^I \le N_{ccs}^I \text{ and } s_j = \sum_{i=1}^{N} p_i^j \text{ or } 1,$$

where $\Delta Delay\_over(\vec{s},\vec{n})$ and $\Delta BW(\vec{s},\vec{n})$ denote the decrement of $Delay\_over(\vec{s},\vec{n})$ and the increment of required bandwidth compared with adjacent states $(\vec{s},\vec{n})$ in the increasing direction, respectively.

Step 3: Search for $(\vec{s},\vec{n})$ that make $Delay\_over(\vec{s},\vec{n})$ be zero. If solutions exist, then choose $(\vec{s},\vec{n})$ with the minimum $(\overrightarrow{BW} + \overrightarrow{BW}^{I}) * \vec{s}$ and stop. Otherwise, select $(\vec{s},\vec{n})$ with the largest $Gain(\vec{s},\vec{n})$ and then repeat step 1 until (4) is satisfied.

IV. Experimental Results

Experimental results are provided to demonstrate the performance of the proposed algorithms. During the experiment using OPNET [16] for simulation, the channel zapping time and the bandwidth usage over core network are employed as performance measures. The simulation conditions are set as follows:

1) Video stream of each channel is encoded at 30 fps (frames per second) and its encoding structure is IBBPBBPBBPBB (i.e. 1 GOP consists of 12 frames). Channels are classified into two groups: SDTV channel group and HDTV channel group whose target channel bandwidths are set to 4.12Mbps and 12.06Mbps, respectively. In the fast channel change stream, each I-frame is encoded with $R_I = 200000$ bits for both SDTV channels and HDTV channels and N_{ccs}^I is set to 3.
2) The network structure under consideration is shown in Fig. 6. In this figure, the link bandwidths of core network and access network are set to 1Gbps and 100Mbps, respectively. One-way delay over IP network is set to 0.6 sec., and the number of users in access network is 20.
3) It is assumed that channel change requests arrive according to a Poisson distribution with an expected inter-arrival time of $1/\lambda$, where λ is the request rate, which is set to 1/30 (the number of requests per second). Channel popularity is assumed to be uniformed or Zipf-like distributed [17]. In the case of uniformly distributed channel popularity, the probability of choosing the j_{th} channel is obtained by $p_i^j = (1/M \cdot N)$. When channel popularity is Zipf-like distributed, the probability of selecting an arbitrary channel is calculated by $p_i^j = (f_i^j / N \sum_{k=1}^{M} f_i^k)$, for $i = 1, \ldots, N$ and $j = 1, \ldots, M$, where $f_i^j = 1/j^{1-\theta}$, and θ is a parameter that specifies the skew factor (θ is set to 1.0 during the experiment). Each user has different channel preference to take into account the individual favorite, and the channel probability of HDTV is set to be higher than that of SDTV when the channel popularity is Zipf-like distributed.
4) The total simulation time is 400 seconds.

First, we compare the optimal solution and the proposed fast channel control algorithm in Section IV-A, and then present the performance of fast channel control algorithm with a larger number of channels in Section IV-B.

TABLE I
CHANNEL STATES ACCORDING TO THE TOLERABLE UPPER BOUND OF AVERAGE ZAPPING TIME (THE NUMBER IN PARENTHESIS INDICATES THE NUMBER OF ADDITIONAL I-FRAMES)

Measures \ D_{upper_bound}		1.8 sec	1.6 sec	1.3 sec	1.1 sec	0.8 sec	0.7 sec	0.5 sec	0.3 sec	0.2 sec
Optimal solution	# of static channels	2	2,4	2,3,7	2,3,5,6	2,3,5,6,7	2,3,4,5,6,7	2,3,4,5,6,7	1,2,3,4,5,6,7	1,2,3,4,5,6,7
	# of additional I-frames	1(1) 3(1) 4(1) 5(1) 6(1) 7(1)	1(1) 3(1) 5(1) 6(1) 7(1)	1(2) 2(1) 4(2) 5(2) 6(2)	1(3) 2(1) 4(3) 7(2)	1(3) 2(1) 3(1) 4(2) 5(1) 6(1)	1(2) 2(1) 3(1)	1(2) 2(2) 3(2) 4(2) 5(1) 6(1) 7(1)	1(2) 2(1) 3(1) 4(1) 5(1) 6(1) 7(1)	1(3) 2(3) 3(3) 4(2) 5(2) 6(2) 7(2)
Fast channel control alg.	# of static channels	7	2,3	2,3,4	2,3,4,5	2,3,4,5,6	2,3,4,5,6,7	1,2,3,4,5,6,7	1,2,3,4,5,6,7	1,2,3,4,5,6,7
	# of additional I-frames	1(1) 2(1) 3(1) 4(1) 5(1) 6(1)	1(3) 4(3) 5(3) 6(3) 7(3)	1(3) 2(1) 5(3) 6(3) 7(3)	1(3) 2(1) 3(1) 6(3) 7(3)	1(3) 2(1) 3(1) 4(1) 5(1) 7(3)	1(3) 2(1) 3(1) 4(1) 5(1)	2(1) 3(1) 4(1) 5(1) 6(1)	2(2) 3(2) 4(2) 5(2) 6(1) 7(1)	1(3) 2(3) 3(3) 4(2) 5(2) 6(2) 7(2)

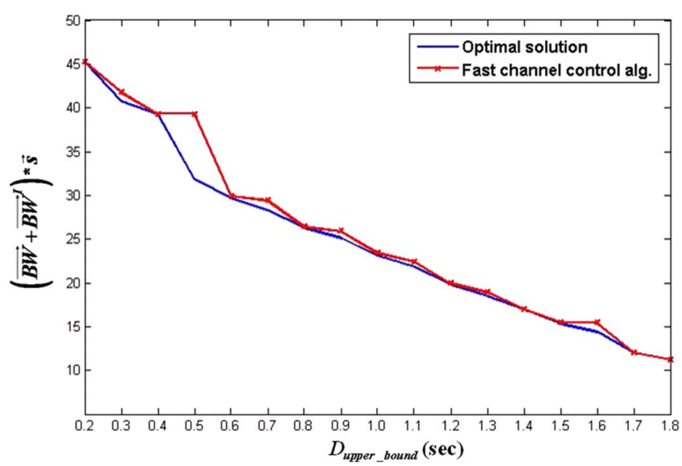

Fig. 7. Performance comparison between the optimal solution and the fast channel control algorithm according to the tolerable upper bound of average zapping time.

A. Performance Comparison Between Optimal Solution and Fast Channel Control Algorithm

In this section, we show that the proposed fast channel control algorithm can provide a near optimal solution with low computational complexity by using the optimal solution as a benchmarking. Since the computational complexity of dynamic programming increases exponentially as the number of channels becomes larger, the experiment is performed with a relatively small number of channels and uniformly distributed channel popularity: 1-HDTV channel (channel # 1) and 6-SDTV channels (channel # 2, 3, 4, 5, 6, and 7). The result is shown in Table I and Fig. 7 (x-axis and y-axis represent the tolerable upper bound of average zapping time D_{upper_bound} in (4) and $(\overrightarrow{BW} + \overrightarrow{BW}^{I}) * \vec{s}$, respectively).

As shown in Table I, the number of static channels and additional I-frames generally increase as D_{upper_bound} becomes smaller. It is observed that the performance difference between two solutions is small, although it becomes relatively large when D_{upper_bound} is 0.5 (it is expected that the difference becomes even smaller as the number of channels increases). On the other hand, the proposed fast channel control algorithm needs much smaller computational complexity than the optimal solution: the observed CPU time of the proposed fast channel control algorithm and the optimal solution are 0.001 sec. and 5.953 sec., respectively. In the following section, the fast channel control algorithm is examined with a large number of channels.

B. Performance of the Proposed Fast Channel Control Algorithm

The proposed fast channel control algorithm is compared with all-dynamic, all-static channel cases, and Adjacent channel algorithm [7] that transmits upward and downward channels of the current channel in advance to the user. Under the assumption that 5-HDTV channels (channel # 1, 5, 10, 20, 30) and 25-SDTV channels (the other channels) are serviced, the tolerable upper bound of average zapping time D_{upper_bound} is set to 1.0 second, and the user channel preference is randomly or Zipf-like distributed (the channel probability sum of each

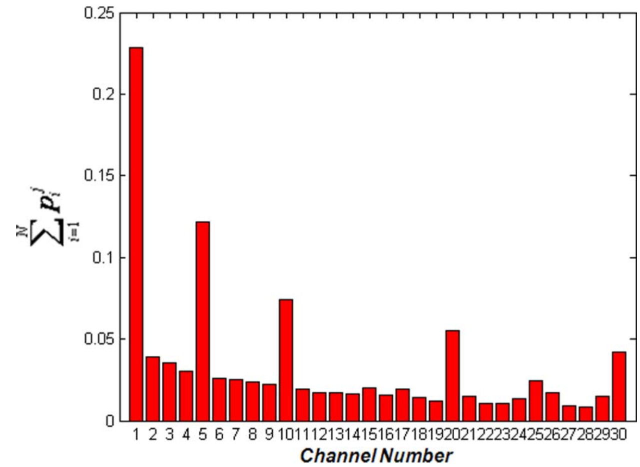

Fig. 8. Channel popularity sum of each user when the channel popularity is Zipf-like distributed.

TABLE II
PERFORMANCE COMPARISON WHEN USER CHANNEL POPULARITY IS UNIFORM DISTRIBUTED AND D_{upper_bound} IS SET TO 1.0 SECOND

Alg. \ Measures	Average bandwidth usage over core network	Maximum average channel zapping time	Average channel zapping time of all users
When all channel are dynamic with no additional I-frames	86.492 Mbps	1.177 sec.	0.794 sec.
When all channel are static with no additional I-frames	165.804 Mbps	0.271 sec.	0.199 sec.
The proposed fast channel control alg.	125.730 Mbps	0.939 sec.	0.493 sec.
Adjacent channel alg. with up/down 1 channel	149.729 Mbps	0.574 sec.	0.344 sec.
Adjacent channel alg. with up/down 2 channel	163.316 Mbps	0.363 sec.	0.233 sec.

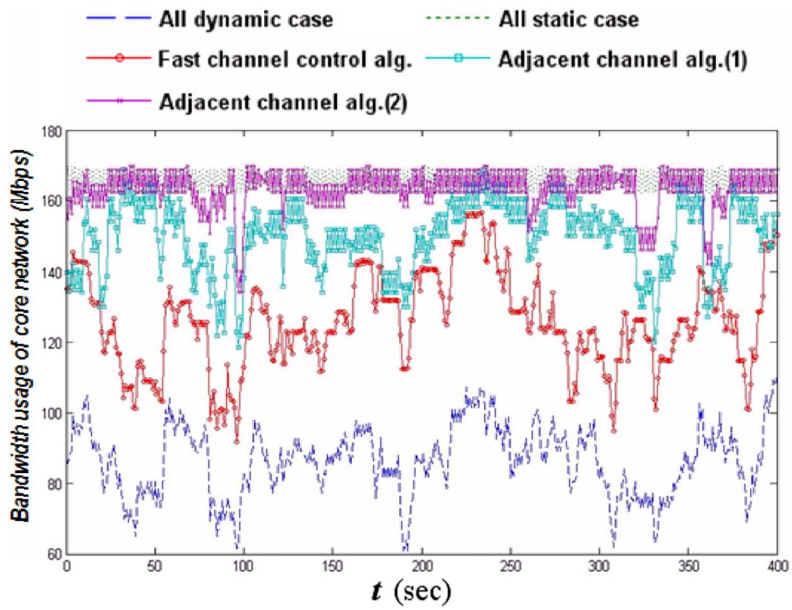

Fig. 9. Bandwidth plots when the channel popularity is uniform distributed (the number in parenthesis indicates the number of adjacent channels that is transmitted upward and downward).

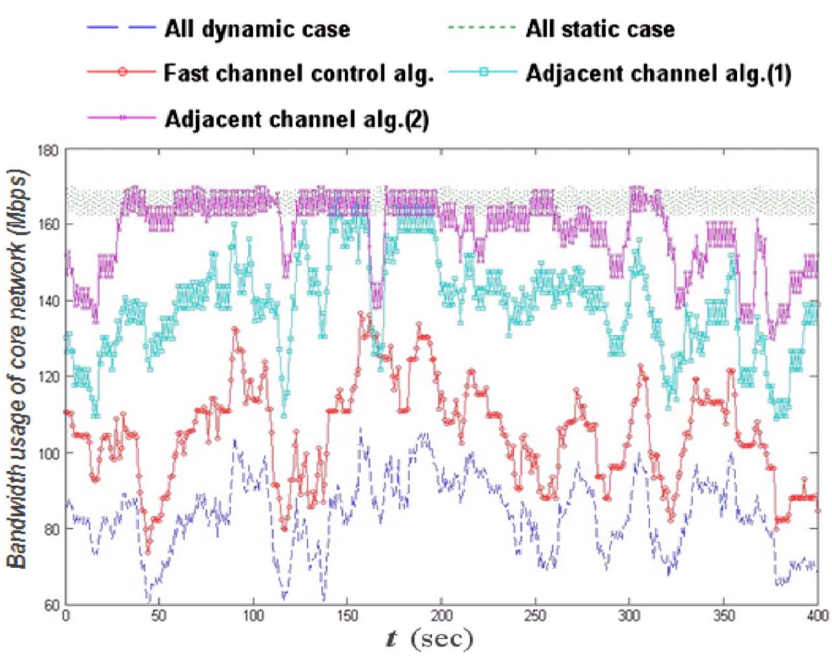

Fig. 10. Bandwidth plots when the channel popularity is Zipf-like distributed.

user $\sum_{i=1}^{N} p_i^j$ is shown in Fig. 8). The results are summarized in Tables II and III. As shown in these tables, the network utilization is the highest but the average channel zapping time of a few users is much larger than D_{upper_bound} when all channels are dynamic, while the channel zapping time is the smallest at the cost of network efficiency when all channels are static. By the way, the proposed algorithm can keep the average channel zapping time of each user below D_{upper_bound} by assigning a limited number of channels to the static channel and effectively inserting a small number of I-frames into the fast channel change stream. The Adjacent channel algorithm has a lower channel zapping time than the proposed algorithm at the cost of higher bandwidth. Also, it is observed that the channel zapping time is decreased and the corresponding bandwidth is increased as the number of adjacent channels increases. However, it needs to dynamically control the transmitted channels according to the channel selection of users, and thus the control overhead may be increased. When all channels have the same preference (i.e. uniform distribution), the results are given in Table II and Fig. 9. Compared to the case that all channels are static, the proposed algorithm reduces the required bandwidth by about 24 % with satisfying the given delay constraints. As shown in Fig. 9, the bandwidth of the proposed algorithm lies between all-dynamic and all-static channel cases. When the user channel

TABLE III
PERFORMANCE COMPARISON WHEN USER CHANNEL POPULARITY IS ZIPF-LIKE DISTRIBUTED AND D_{upper_bound} IS SET TO 1.0 SECOND

Measures / Alg.	Average bandwidth usage over core network	Maximum average channel zapping time	Average channel zapping time of all users
When all channel are dynamic with no additional I-frames	83.478 Mbps	1.133 sec.	0.672 sec.
When all channel are static with no additional I-frames	165.804 Mbps	0.272 sec.	0.206 sec.
The proposed fast channel control alg.	104.993 Mbps	0.917 sec.	0.500 sec.
Adjacent channel alg. with up/down 1 channel	138.423 Mbps	0.498 sec.	0.345 sec.
Adjacent channel alg. with up/down 2 channel	157.992 Mbps	0.389 sec.	0.252 sec.

popularity is Zipf-like distributed, the performance improvement is more obvious as shown in Table III and Fig. 10. It is observed that the proposed algorithm reduces the required bandwidth by about 37 % compared to the case that all channels are static, and the bandwidth plot of the proposed algorithm is closer to that of all-dynamic channel case.

V. CONCLUSIONS

In this paper, we have proposed an IPTV channel control algorithm pursuing an effective tradeoff between channel zapping time and network utilization. In the problem formulation, we have treated the channel distribution state and the video coding/transmission scheme as control variables. The optimal solution has been obtained by using Viterbi algorithm, and the fast channel control algorithm has been presented to reduce the computational complexity. The optimal solution has played a role as a benchmarking for the fast algorithm. It has been observed during the experiment that the proposed algorithm has achieved very good network efficiency, satisfying the delay constraint.

For a complete solution, we need to take into account not only core network but also access network even though the bandwidth of core is considered in this paper. In fact, the efficient management of access network is also one of critical factors for the successful deployment of IPTV service since access network has very limited resources compared to core network. This is under our consideration for future work.

REFERENCES

[1] J. Kim, J. H. Hahm, Y. S. Kim, and J. K. Choi, "NGN architecture for IPTV service without effect on conversational services," in *International Conference on Advanced Communication Technology*, Feb. 2006, vol. 1, pp. 465–469.

[2] W. Park, C. Choi, D. Kim, Y. Jeong, and K. Park, "IPTV-aware multi-service home gateway based on FTTH access network," in *International Symposium on Consumer Electronics*, Jun. 2005, pp. 285–290.

[3] S. Tekla, "The trial and travails of interactive TV," *IEEE Spectrum*, no. 4, pp. 22–28, 1996.

[4] A. Harris and G. Ireland, "Enabling IPTV: What carriers need to know to succeed," International Data Corporation, Ireland, May 2005, White paper.

[5] Cisco systems, "Managing delay in IP video networks version 1.0," 2005, white paper.

[6] N. Sinha and R. Oz, "The statistics of switched broadcast," in *Society of Cable Telecommunications Engineers Conference on Emerging Technologies*, 2005.

[7] 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 *International Conference on Advanced Communication Technology*, 2004, vol. 2, pp. 971–975.

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

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

[10] U. Jennehag and T. Zhang, "Increasing bandwidth utilization in next generation IPTV networks," in *International Conference on Image Processing*, Oct. 2004, vol. 3, pp. 2075–2078.

[11] Internet Engineering Task Force, "Considerations for IGMP and MLD snooping Switches," May 2004, Internet Draft.

[12] J. Wang, L. Sun, X. Jiang, and Z. Wu, "IGMP snooping: a VLAN-based multicast protocol," *High Speed Networks and Multimedia Communications*, pp. 335–340, July 2002.

[13] A. J. Viterbi and J. K. Omura, *Principle of Digital Communication and Coding*. New York: McGraw-Hill, 1979.

[14] A. Ortega, K. Ramchandran, and M. Vetterli, "Optimal trellis-based buffered compression and fast approximations," *IEEE Trans. on Image Processing*, vol. 3, no. 1, Jan. 1994.

[15] D. P. Bertsekas, *Nonlinear Programming*. Massachusetts: Athena Scientific, 1995.

[16] OPNET Modeler [Online]. Available: www.opnet.com.

[17] G. Zipf, *Human Behavior and the Principle of Least Effort*. Reading, MA: Addison-Wesley, 1994.

Hyunchul Joo received B.S. degree from Dept. Electrical Engineering and Computer Science, Hanyang University, Korea in February 2005. He received his M.S. degree from Dept. Computer Science and Engineer at POSTECH (Pohang University of Science and Technology) in February 2007. Currently, he is a Ph.D. student in POSTECH. His interests are in the areas of convergence networks, IPTV broadcasting system, overlay multicasting and image/video compression.

Hwangjun Song received the B.S. and M.S. degrees from Dept. of Control and Instrumentation (EE), Seoul National University, Korea in 1990 and 1992, respectively, and Ph.D. degree in Electrical Engineering-Systems, University of Southern California, Los Angeles, CA, USA in 1999. From 1995 to 1999, he was a research assistant in SIPI (Signal and Image Processing Institute) and IMSC (Integrated Media Systems Center), Univ. of Southern California. From 2000 to 2005, he was an assistant professor/vice dean of admission affairs at Hongik University, Seoul, Korea. Since Feb. 2005, he has been with Dept. of Computer Science and Engineering, POSTECH (Pohang University of Science and Technology), Korea. He received Haedong Best Paper Award from Korean Institute of Communication Science in 2005. His research interests include multimedia signal processing and communication, image/video compression, digital signal processing, network protocols necessary to implement functional image/video applications, control system and fuzzy-neural system.

Dai-Boong Lee received his B.S. and M.S. degree in 2002 and 2004, respectively. He is currently working toward his Ph.D. degree in Multimedia Communication System Lab., Dept. of CSE, POSTECH (Pohang University of Science and Technology). His research interests include multimedia communication, packet scheduling, QoS network, Int/Diffserv, network resource renegotiation algorithm, network management, visual information processing, IP-TV system, HFC networks, and DOCSIS QoS.

Inkyu Lee was born in Seoul, Korea in 1967. He received the B.S. degree (Hon.) in control and instrumentation engineering from Seoul National University, Seoul, Korea in 1990, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA in 1992 and 1995, respectively.

From 1991 to 1995, he was a Research Assistant at the Information Systems Laboratory, Stanford University. From 1995 to 2001, he was a Member of Technical Staff at Bell Laboratories, Lucent Technologies, where he studied the high-speed wireless system design. He later worked for Agere Systems (formerly, Microelectronics Group of Lucent Technologies), Murray Hill, NJ as a Distinguished Member of Technical Staff from 2001 to 2002. In September 2002, he joined the faculty of Korea University, Seoul, Korea, where he is currently an Associate Professor in the School of Electrical Engineering. He has published over 30 journal papers in IEEE, and has 20 U.S. patents granted or pending. His research interests include digital communications, signal processing, and coding techniques applied to wireless systems. He currently serves as an Associate Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS in the area of Wireless Communication Theory and has also been a Chief Editor for the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS (Special Issue on 4G Wireless Systems) in 2006. Dr. Lee received the IT Young Engineer Award as the IEEE/IEEK joint award in 2006. Also in 2006, he received the APCC'06 best paper award.