Modeling User Activities in a Large IPTV System*

Tongqing Qiu
Georgia Tech
Atlanta GA
tongqqiu@cc.gatech.edu

Zihui Ge
AT&T Labs – Research
Florham Park, NJ
gezihui@research.att.com

Seungjoon Lee
AT&T Labs – Research
Florham Park, NJ
slee@research.att.com

Jia Wang
AT&T Labs – Research
Florham Park, NJ
jiawang@research.att.com

Jun (Jim) Xu
Georgia Tech
Atlanta GA
jx@cc.gatech.edu

Qi Zhao
AT&T Labs – Research
Florham Park, NJ
qzhao@research.att.com

ABSTRACT

Internet Protocol Television (IPTV) has emerged as a new delivery method for TV. In contrast with native broadcast in traditional cable and satellite TV system, video streams in IPTV are encoded in IP packets and distributed using IP unicast and multicast. This new architecture has been strategically embraced by ISPs across the globe, recognizing the opportunity for new services and its potential toward a more interactive style of TV watching experience in the future. Since user activities such as channel switches in IPTV impose workload beyond local TV or set-top box (different from broadcast TV systems), it becomes essential to characterize and model the aggregate user activities in an IPTV network to support various system design and performance evaluation functions such as network capacity planning. In this work, we perform an in-depth study on several intrinsic characteristics of IPTV user activities by analyzing the real data collected from an operational nation-wide IPTV system. We further generalize the findings and develop a series of models for capturing both the probability distribution and time-dynamics of user activities. We then combine theses models to design an IPTV user activity workload generation tool called SIMULWATCH, which takes a small number of input parameters and generates synthetic workload traces that mimic a set of real users watching IPTV. We validate all the models and the prototype of SIMULWATCH using the real traces. In particular, we show that SIMULWATCH can estimate the unicast and multicast traffic accurately, proving itself as a useful tool in driving the performance study in IPTV systems.

Categories and Subject Descriptors

C.2.3 [**Computer System Organization**]: Computer-Communication Networks—*Network Operations*; C.4 [**Computer System Organization**]: Performance of Systems—*Modeling techniques*

General Terms

Measurement, Performance

Keywords

IPTV, Network Measurement, Modeling, Workload Generator

1. INTRODUCTION

In the past several years, there has been a global trend among telecommunication companies on the rapid deployment of IPTV (Internet Protocol Television) infrastructure and service, in which live TV streams are encoded in a series of IP packets and delivered to users through the residential broadband access network. This fast growth is motivated in part by commercial reasons – strengthening their competitiveness with so-called triple-play package that combines digital voice, TV and data service together. More importantly, this new technology provides the users with great interactive capability and functional flexibility, and creates tremendous opportunities for a broad range of new applications (e.g., CollaboraTV [11]), which may very well define a next generation of TV entertainment.

While the industry rushes into the IPTV era, what lags behind is a comprehensive understanding of the user activities, which directly dictate the flow of video streams and other bi-directional data (e.g., for user interactive sessions). Although commercial TV distribution networks (e.g., cable, satellite) have prevailed for decades, to the best of our knowledge, no detailed study on modeling individual users' TV watching activities is available in the literature. This might be partially because there has not been a strong need, as data flows in conventional TV networks are typically limited to the downstream direction from servers to set-top boxes (STBs), and user channel switching (with the exception of pay per view) has very little system-wide impact. In fact, even tracking the viewership of TV program – a statistic that bears significant commercial value – is typically done through a third party [1]. In IPTV systems, by contrast, an understanding of user activities is essential to many system design and engineering tasks such as evaluation of various design options, optimal system parameter tuning, improving customer care, and defining effective system care procedures to minimize service impact.

Without a realistic user activity model, the research community often has had to rely on some hypothetical user models when analyzing system performance [17, 19, 2, 20]. Unfortunately, such models are sometimes quite different from the reality and can potentially lead to incorrect estimation of the system performance. For example, while a constant-rate Poisson process is widely used as a workload model in other systems, it is incapable of capturing

*This work is supported in part by the NSF grant CNS-0905169, funded under the American Recovery and Reinvestment Act of 2009 (Public Law 111-5), and by the NSF grant CNS-0626979.

IMC'09, November 4–6, 2009, Chicago, Illinois, USA.

the high bursts of channel switches at around hour boundaries observed in our IPTV data. An alternative is to directly use actual IPTV trace data for the evaluation of system performance. However, such data, even when anonymized, can be highly sensitive, containing too much commercial and user private information to be publicly distributed. This creates a barrier for research community to perform system evaluation against real data traces. In this work, we bridge this gap by developing realistic models for user activities in a large IPTV system.

Our work in this paper is based on a large collection of data obtained from a nation-wide operational IPTV network, which includes the system logs from all of its subscribers' STBs, control plane signaling messages, network topology and configuration data, and TV channel information. Our approach starts with an in-depth investigation of the user activities, analyzing many intrinsic characteristics on attributes such as user viewing sessions, per-channel dwell time, and channel popularity. While some of our findings overlap with a previous study [6], we further abstract and generalize the chosen characteristics to enable *realistic workload generation*, which can be used for various stages of IPTV system design. Specifically, we develop a workload generator that faithfully mimics the user activities in real IPTV systems—this workload generator can turn a limited number of input parameters (published in the paper) into synthetic traces having similar statistical properties to realistic data traces.

We also consider this work a snapshot of user activity workload for the current IPTV system, which provides a feature set highly similar to that of conventional TV services such as cable and satellite. We envision that user viewing pattern evolves with more advanced IPTV features fundamentally changing the way users watch TV, and this work is used as a baseline to understand and to quantify such changes.

We make three major contributions in this paper. First, we present in-depth analysis results based on data traces from a nation-wide operational IPTV system (Section 3). In addition to the largest scale of such study (using more than a million STBs in four different time zones), we identify many interesting characteristics. For example, we find that user activities (such as channel switching) are often correlated, hence the aggregate activities are much more bursty than the outcome of a fixed-rate Poisson model that many previous studies assume [2, 20].

Second, we develop a series of models that capture these intrinsic characteristics on each of the attributes (Section 4). We use the mixture exponential distribution to model various session duration distributions. To characterize the time-varying nature of user activities, we apply Fourier Transform and model the periodically correlated events. We distinguish sequential-channel-scans and targeted-channel-switches and use Zipf-like and exponential distribution to characterize channel access popularity. We also adopt a mixture population model to capture the channel popularity dynamics observed at the finer time granularity.

Third, we combine these models and construct a workload generation tool, namely SIMULWATCH (Section 5), which takes a small number of parameters as input and outputs a series of synthetic user traces that mimic a set of real users watching IPTV. We also validate SIMULWATCH prototype by comparing the synthetic trace with a real data trace and show that they closely match even for some properties that we do not explicitly model. Specifically, we show that for a given number of STBs, we can accurately estimate the unicast and multicast traffic bandwidth based on the synthetic workload, which also illustrates how to use SIMULWATCH to drive the performance study in an IPTV system.

We also review related work in Section 6 and conclude our work in Section 7. In the following section, we first overview a typical IPTV system architecture and describe the data set we use in this study.

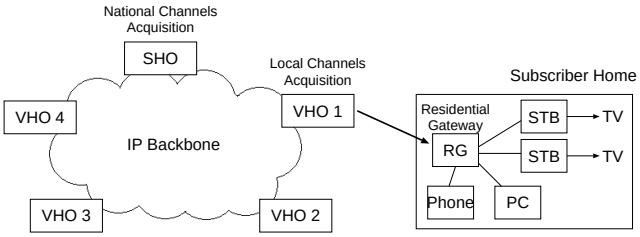


Figure 1: IPTV Architecture

2. BACKGROUND

2.1 Overview of IPTV Architecture

Figure 1 shows a typical IPTV service system [15]. The SHO (Super Hub Office), the primary source of television content, digitally encodes video streams received externally (e.g., via satellite) and transmits them to multiple VHOs (Video Hub Offices) through a high-speed IP backbone network. The VHOs, each responsible for a metropolitan area, in turn acquire additional local contents (e.g., local news), perform some further processing (e.g., advertisement insertion) and transmit the processed TV streams to end users upon request[1]. Inside a residential home, RG (Residential Gateway) connects to a modem and one or more STBs (Set-Top Boxes) with coaxial cable, receiving and forwarding all data, including live TV streams, STB control traffic, VoIP and Internet data traffic, into and out of the subscriber's home. Finally behind an STB, there connects a TV.

In order to leverage the one-to-many nature of IPTV traffic and ease the bandwidth requirement of the VHOs, video streams from VHOs to STBs are typically delivered using IP multicast. Depending on the TV channel and the codec used, the bit rate of each video stream varies widely from around 1.5Mbps (SDTV with H.264) to around 15Mbps (HDTV with MPEG2). The latency of channel switch is due to both the multicast group management using IGMP and the video decoding dependency (e.g., waiting for a next I-frame). They could add up to a few seconds, which might make the audience chafe at the bit. This limitation is likely to motivate IPTV users to perform more targeted channel switches than random or sequential channel scans compared to users from conventional TV systems. To address/alleviate this problem[2], some IPTV providers have adopted a fast-channel-switch mechanism in which a server in the VHO sends the STB a unicast video stream (often at a rate higher than multicast rate to avoid long decoding latency) while the STB catches up with the multicast TV stream [17].

Similar to conventional TV users, IPTV users use a vendor/provider customized remote controller to control the STB. For example, one may use *Up/Down* buttons to sequentially switch channels, use *Return* button to jump back to the channel previously watched, or enter a channel number to jump directly to a specific channel. On the other hand, IPTV providers often support additional features, some of which are not offered in conventional TV services. For example, many IPTV providers add the capability for

[1]Typically a VHO connects to RGs through a protected optical network using fiber-to-the-node (FTTN) or fiber-to-the-premises (FTTP) technologies. We omit showing the network elements in between as they are not the focus of our work.

[2]How to reduce this delay is an active research area.

a small number of user-defined *favorite channel list*, so that one can easily switch between or scan through the favorite channels. Furthermore, most STBs support the DVR (Digital Video Recording) feature, in which with the help of a local hard drive, a user can pause, rewind, fast forward (up to live play), and record the TV program being played. Some IPTV providers support one channel being recorded to DVR while another channel being played live on TV. Also depending on the IPTV provider, IPTV users can enjoy many advanced features such as Picture-In-Picture (PIP), on-line gaming and chatting, and personalized web services on their TVs.

2.2 Data Set

The data we use in this study are collected from a large scale IPTV provider in the United States, which has over one million subscribers and over two million STBs spread throughout four different time zones. As a privacy protection, only anonymous data was used in this study; no information that could be used to directly or indirectly identify individual subscribers was included. By combining data from the anonymous STB logs, control plane messages, network configuration data, and TV channel lists obtained from this service provider, we were able to model activities such as turning on/off STBs, switching channels, and playing live or recorded TV program. In particular, we associate each activity recorded in the anonymous STB logs with its origin STB and a timestamp (which is at the precision of one second). To account for different time zones, we map the STBs to their metropolitan area and convert the associated timestamps into their local time.

Although we have analyzed and validated data from multiple periods of time, in this paper, we only present the result based on the data collected in April 2009. More specifically, we use the 6 days' data from April 2nd to 7th for analysis and one day's data on April 8th for the validation of our model. Unless specified otherwise, we use the results based on data on April 2nd 2008 when we present the properties for a single day, while the results on other days are qualitatively similar.

There are a few limitations/caveats with the quality of the data in this study. First, the channel switch events capture user requests logged at the STB, with the timestamp indicating the time that the request is received at the STB. Note that this is different from the time when the request arrives at the VHO, and different from the time when the streaming content is received at the STB. Requests that are very rapidly followed by a subsequent request in time may not be recorded by the STB, hence are missing from our study. Second, since we do not have any data related to the remote controller, we can only determine the outcome of a user's action, but not the action itself. For example, when we observe a channel switching event, we do not know whether it is done by using the *Up/Down* button or by directly inputting the channel number, for which we have to rely on inference. Third, we do not have detailed TV program information when DVR is used. From STB logs, we know a recorded video is being played, but do not know what is played. In this paper, we use a special channel, 9999, to indicate user tuning into the DVR mode and playing some recorded content. Finally, our data regarding the advanced features such as online gaming are unfortunately incomplete. Therefore, we ignore all the activities with respect to advanced features in this paper.

3. ANALYZING USER ACTIVITIES

Recall that our objective is to define a mathematical process that mimics the activities of IPTV users and thus can produce realistic event series for tasks such as system performance evaluation. To accomplish this, we first need to understand how real users act in an operational IPTV system. We do so by studying various characteristics of our data traces. In particular, we focus on the aggregate properties regarding to users' turning STBs on and off, channel switches, and channel popularity.

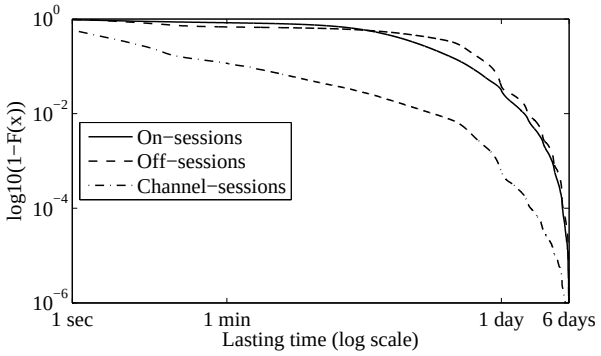


Figure 2: CCDF of the length for on-, off-, and channel-sessions

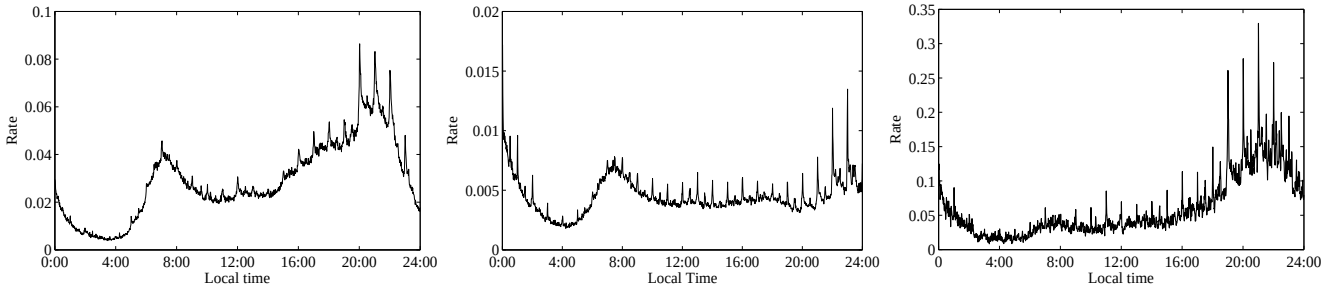
3.1 Turning STBs On and Off

We first focus on the length of STB on- and off-sessions. An *on-session* is defined as the duration from a STB being switched on till it gets switched off. Similarly, an *off-session* is the duration from the last time a STB was switched off till it gets switched on. We first examine the distribution function of the length of on- and off-sessions respectively.

Figure 2 shows the complementary cumulative distribution function (CCDF) of the length of on- and off-sessions. Using CCDF, we can better illustrate the tail property of the distributions. We first observe that both on-sessions and off-sessions exhibit a very long tail in their distributions – around 5% of the on-sessions and off-sessions are over 1 day in length. In fact, we believe the fast drop in both tails approaching the right end of the x axis is due to the limit of our dataset, which is 6 days in total. Comparing off-session and on-session, we find that the off-session has a heavier tail than the on-session. This matches our intuition since it is more likely that an IPTV user leaves the TV off for a long time (several days) than leaving the TV on. We also notice that the curve of off-session is below that of on-session for low session length. This is likely due to users' mistake in operating the remote controller – a user accidentally turning the STB off while watching a TV program may quickly switch the STB back on, producing a short off-session of a few seconds.

In Figure 3, we show the time series of the number of on-line STBs in one-minute precision (normalized by the average number of on-line STBs). We observe a very strong diurnal pattern, with daily peak at around 9PM, followed by a quick decrease in number, reaching daily minimum at around 4AM, and then steadily ramping up during the course of day. Note that there are a significant number of STBs left on over night.

As both on- and off-sessions are bounded by users' action in switching on and off the STBs, it makes sense to observe these event processes directly. Figures 4a and 4b show a one-day time series of the event rate for the *switching-on* and *switching-off* events respectively. Both plots are shown in one-minute precision. Here the event rate is the number of switching-on/-off events during the interval normalized by the total number of off-line/on-line STBs at the beginning of the interval. We make two observations. First, there is a strong time-of-day effect in both figures.The switching-on event rate has local peaks at around 7AM and around 9PM and the switching-off event rate has local peaks at around 7:30AM and around 12AM, both matching well with our intuition relating to



(a) Switching-on events (b) Switching-off events (c) Channel switching events

Figure 4: The normalized switching-on, switching-off, and channel switching events (one-minute granularity)

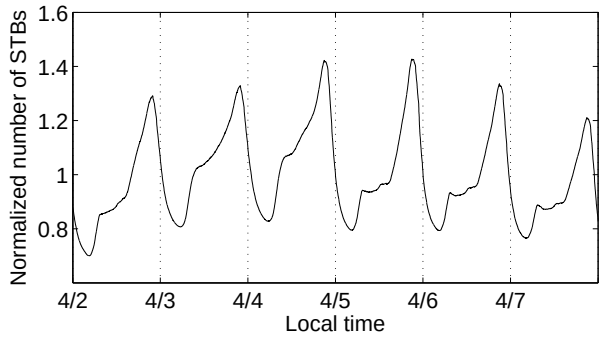


Figure 3: Number of on-line STBs

the daily living schedules of most people. More interestingly, we observe that both event rate series are very bursty, with significant spikes aligning closely with hour or half-hour boundaries (it is more pronounced in Figure 4b). This is due to the fact that most TV programs are aligned to hour boundaries. Many users may turn on TV in anticipating for a TV program or turn off TV after watching a TV program. This introduces significant correlations among users' activities, causing very strong bursts in the aggregate event rates.

3.2 Switching Channels

We now turn to channel switches. Figure 2 shows the distribution function of the length of channel-sessions, which we define as the duration from the time of a user's last channel switch (or turning on STB) till the next channel switch (or turning off STB). We find that this distribution also has a long tail, although not as heavy as those of on- and off-sessions.

Similar to those for switching-on and switching-off events, we also examine the aggregate event process for channel switches. Figure 4c shows the time-series of such event rate, which is defined as the total number of channel switches normalized by the number of on-line STBs. We note that the diurnal pattern in Figure 4c is quite representative of different days. Compared to switching-on/-off event rates, the channel switching events demonstrate even stronger spikes with the period of 30 minutes. This is again due to correlated user activities related to TV program alignment – many users switch channels together when a TV program ends, which may create temporarily high workload on the IPTV servers.

We next try to gain insight on how IPTV users switch channels. We classify channel switching events into two categories: *sequential-scanning* and *target-switching*. Sequential scanning represents the user in a channel-browsing mode by going through the available channels using the *Up/Down* button on the remote controller, while target switching represents the user intentionally switching to a specific channel of choice. We assume channel switches between adjacent channels being the sequential-scanning and the rest target-switching. To define the channel adjacency, we need to infer the list of available channels, which can be quite different from one user to another (e.g., due to different subscription plans). We keep track of all channels that a STB requests over an extended period (e.g., one month) and regard these channels as the complete list.

From the data we collected, we observe that 56% of channel switching events are sequential-scanning. This is a little bit lower than our expectation. We find that the high ratio of target switching can be attributed to many advanced features that the IPTV provider supports, including a user-customized favorite-channel list, a program menu where users can browse and switch channels by name, and an easy access to DVR. All of these help users find the TV program of interest easily and directly. To understand this effect better, we construct a user's favorite channel list using a heuristic (top ranked channels by watching time and frequency, e.g., watched in at least 4 days of a week) and find a large portion (46%) of the target-switching is toward such "favorite channels". Among sequential-scanning, we observe an unbalanced up- and down-channel-switches – 72% of them are up-channel-switches. It implies that more people prefer increasing channel number. Our analysis finds the ratio moderately stable over time, although we do not have a good intuition on why this is the case.

3.3 Channel Popularity

We now focus our attention on the properties of different channels. We first rank nearly 700 different channels that appeared in our data using two metrics: (1) the request count, which we call *channel access frequency*, and (2) total time STBs stay tuned in the channel, which we call *channel dwell time*. Figure 5 shows the cumulative distribution function of channel popularity ranked by the two metrics. We find that the distribution of channel dwell time is highly skewed – the top 100 channels account for around 63% of the total channel dwell time. As a comparison, the channel access frequency curve is less skewed. This is likely due to the large number of sequencial-scanning channel switch events. We observe similar level of skewness in the distribution of channel popularity when we examine different subsets of our data (such as by different time zones or by different date), although the ranking of the channels varies from one subset to another.

Figure 6 shows how the top 10/50/100 popular channels change in the two adjacent hours during a day. The change percentages are averaged over 6-day data. We observe that the channel popular-

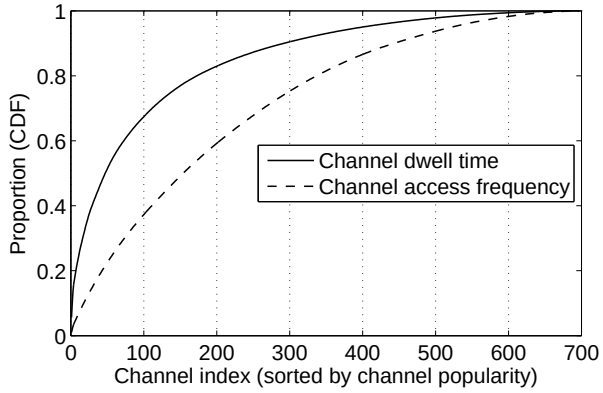


Figure 5: CDF of channels popularity

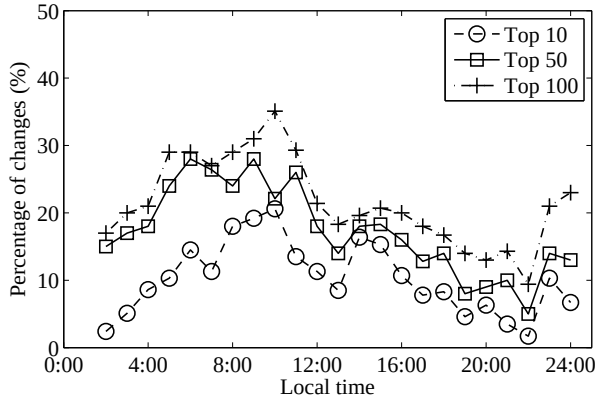


Figure 6: Ratio of change in popular channels, as seen over hours in a single day.

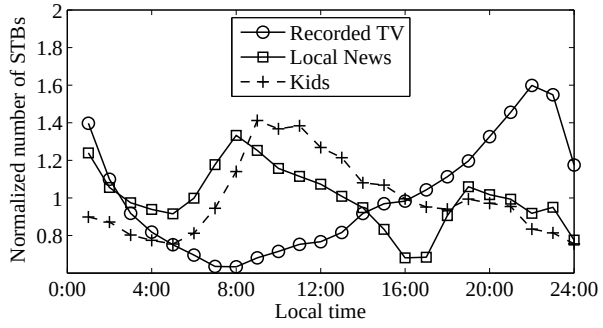


Figure 7: Channel popularity distribution change (hourly)

ity is relatively stable over time of day. For example, in Figure 6, among top 100 channels at 12pm, less than 20% of them did not belong to top 100 channels at 1pm, while more than 80% of them were among top 100 channels at both time periods. We find that the relative channel popularity changes the most during morning hours, but remain moderately stable for most part of the day. Figure 7 illustrates an example on the dynamics of channel popularity within a day, in which we compare normalized numbers of STBs of a top-ranked kids channel and a top-ranked local news channel against that of the "recorded TV" that people use to watch recorded contents (called DVR channel). We observe some interesting time-of-day trends – for example, the local channel peaks in the morning when people catch early news and weather forecast before going to work; the kids channel sharply loses popularity after 8PM when most kids go to bed. In comparison, the DVR channel has the most dramatic change in scale, which finds its peak late into night. The recent work [15] uncovered the reasons behind this behavior by grouping the whole user population into subgroups according to their preference. We will also integrate this into our workload generator in Section 4.4.

Table 1: Model parameters for session length distributions

	λ_1	a_1	λ_2	a_2	λ_3	a_3
On-session	1.3e-2	0.3	3.3e-3	0.66	2.3e-4	0.04
Off-session	3.2e-2	0.19	2.5e-3	0.75	2.4e-4	0.06
Channel-session	2.1	0.23	2.6e-2	0.64	3.2e-3	0.13

4. MODELING USER ACTIVITIES

In this section, we construct mathematical models to capture the observed characteristics of IPTV user activities. We need to model three different user activities – *switching-on*, *switching-off* and *channel-switch*. For each of them, we match their timing properties in both the session length distribution and the dynamics of the aggregate rate. For channel-switch, we also model channel popularity properties including popularity distribution and its temporal dynamics. We first present our models and then describe our methods to deriving the parameters of our model from the data traces.

4.1 Modeling Session Length

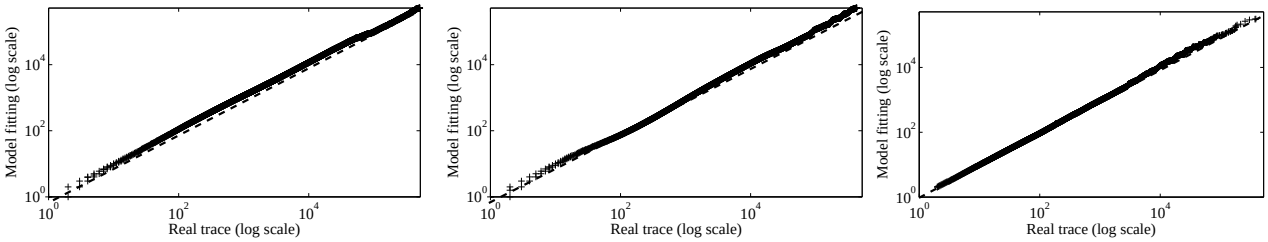
In order to capture the long tails exhibited in the empirical session length distributions (Figure 2), we adopt the mixture-exponential model [13] for on-, off-, and channel-sessions. The probability density function (PDF) of a mixture-exponential distribution is

$$f(x) = \sum_{i=1}^{n} a_i \lambda_i e^{-\lambda_i x} \quad (1)$$

where $1/\lambda_i$ is the mean of the i-th exponential distribution in the mixture and $\sum_{i=1}^{n} a_i = 1$. This model has been widely applied due to its simple form and its capability in approximating heavy-tailed distributions in a wide range [13].

To determine the model parameters that best describe the data trace we collected, we apply data fitting for on-, off-, and channel-sessions respectively. In the following, we use channel-sessions as an example while the procedure for fitting on- and off-sessions is essentially the same. We iteratively explore different values for the number of exponential distributions, n, in the mixture model. For a given n, we apply the Expectation Maximization (EM) algorithm [4] to find the maximum likelihood estimate (MLE) for the parameters λ_i and a_i. For the length distribution of channel-sessions, we identify the best tradeoff at $n = 3$, as it achieves a close match to the data while using a small number (i.e., 6) of model parameters. In Table 1, we report the parameter values that fit our trace. The QQ (quantile-quantile) plots in Figure 8 demonstrate good matches between our models and real traces collected.

Looking into the parameters, we gain tremendous insight on the process. For channel-sessions, the different λ_i corresponds to Poisson processes with average inter-arrival time of around 30 seconds, 40 minutes and 5 hours, representing an IPTV user in the state of channel-browsing, TV-program-watching, and being away-from-TV respectively. The likelihood of a user entering these modes



(a) The length of on-sessions (b) The length of off-sessions (c) The length of channel-sessions

Figure 8: QQ plots comparing models and real traces

Table 2: Modeling parameters for event rates

	k	μ	$p_{1\text{-hour}}$	$p_{30\text{-min}}$	$p_{15\text{-min}}$
Switching-on	0.0036	278	1.76	1.41	x
Switching-off	0.0316	233	4.43	7.85	x
Channel-switch	0.03840	293	4.23	5.34	4.53

is quantified by the a_i values. Similar observation can be made for on-sessions and off-sessions too.

4.2 Modeling Time-Varying Rates

The mixture-exponential models in the previous subsection imply a constant-rate stochastic process with the mean event rate equal to $1/(\sum_i a_i/\lambda_i)$. However, we have observed in Figure 4 that the aggregate event rate for switching-on, switching-off, and channel-switches are all highly variant, highlighted by many apparently-periodic spikes. The problem lies in a subtle underlying independence assumption (which has been commonly used in similar study without careful validation). The reality is that each individual user's activities are influenced by a common external process – the TV program schedules, and as a result, they become highly correlated to each other, breaking the independence assumption. In this subsection, we incorporate this impact from the external process through modeling the aggregate event rates in Figure 4.

As the aggregate event rates appear very complicated in the time domain – requiring an overwhelming number of parameters to characterize it, we decide to try a different angle and approach the problem from the frequency domain. We apply fast Fourier transform (FFT) to the event time series and present the result in Figure 9. The structure, in all cases, suddenly becomes very clear – there are a few of distinct spikes at frequencies that correspond to 1 hour, 30 minutes, 15 minutes etc., and an ambient gradual decrease in the power level (y-axis) from low to high frequencies. We next approximate the ambient power level by using the Weibull distribution. Its probability density function is:

$$f(x;k,\mu) = \frac{k}{\mu}\left(\frac{x}{\mu}\right)^{k-1} e^{-(x/\mu)^k}$$

where k and μ are model parameters. We choose the Weibull distribution since it can very well approximate a wide range of classes of functions including exponential, normal and lognormal only with two parameters. The model parameters that best match our data traces are reported in Table 2.

From only a small number of parameters (k, μ, and the values for the spikes in the frequency domain), we can now generate the frequency domain function and apply inverse FFT to reconstruct the time series that initially seemed highly complex. To determine the best trade-off between the number of spikes to explicitly include in the model and the quality of the match between the model and the empirical trace, we show in Figure 11 the discrepancy metric (we use mean squared error between the empirical trace and our modeling output) as a function of the number of spikes in the model. It is clear that the discrepancy becomes negligible when we choose 2 spikes for switch on/off and 3 spikes for channel switch. So they are the values we use in the rest of the paper. Figure 10 compares the result from the real trace (top) and the result from our model (bottom). We find that they match very well, even when we use only 13 parameters here (k, μ and the value for the 2-3 spikes in Table 2).

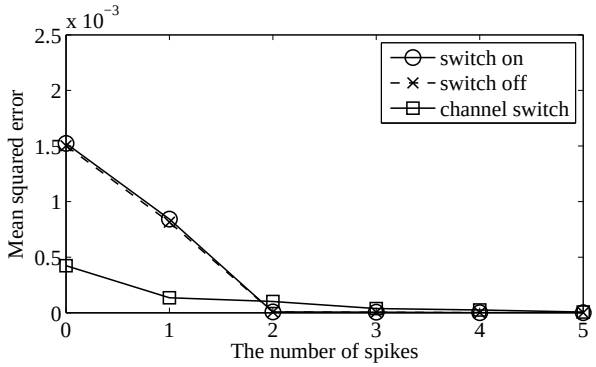
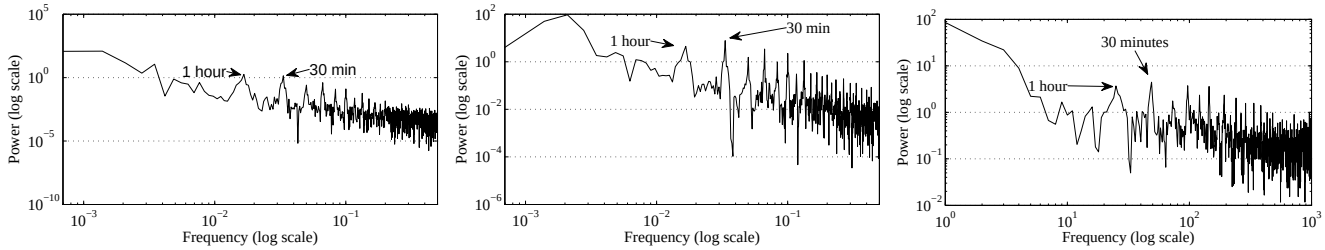


Figure 11: Find the optimal number of spikes

Finally, we define the time series function obtained from the aforementioned process as our *rate moderating function* $g(t)$, which models the impact of external TV program schedule to individual users' activity. Since $g(t)$ is constructed from data in a given window W, (in our example $W = 86,400$ seconds, or 1 day), we simply repeat $g(t)$ to make it a periodic function: $g(t+W) = g(t)$. Furthermore, we normalize $g(t)$ such that $\int_0^W g(t)dt = W$. Note that the periodic moderating function g will not impact the tail behavior of the session length distributions that we have modeled previously. However, it does change the shape of session length distribution at small durations. In particular, depending on the start time-of-day, the session length distribution varies.

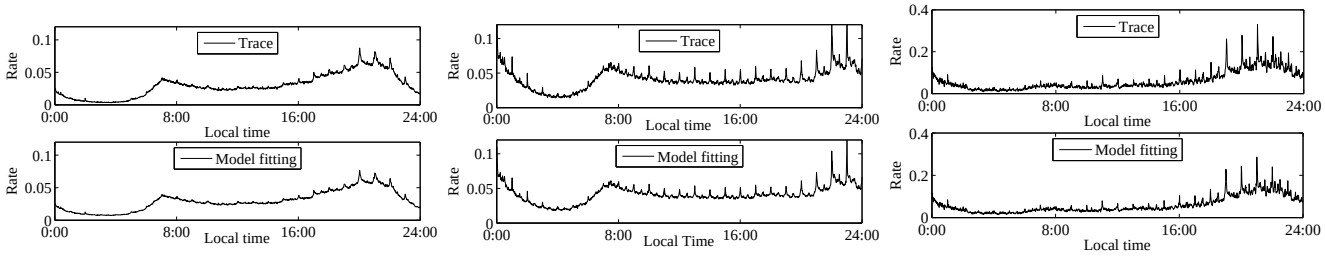
4.3 Modeling Channel Popularity Distribution

We have observed in Section 3 that the channel popularity is highly skewed. Motivated by the success of Zipf-like distribution in modeling skewed access frequencies of Web [3] and VoD systems [18, 21], we also examine the Zipf-like distribution in modeling channel access frequencies – for a channel of popularity rank



(a) Switching-on (b) Switching-off (c) Channel-switch

Figure 9: The time-varying rates in frequency domain



(a) Switching-on (b) Switching-off (c) Channel-switch

Figure 10: Modeling aggregate event rate

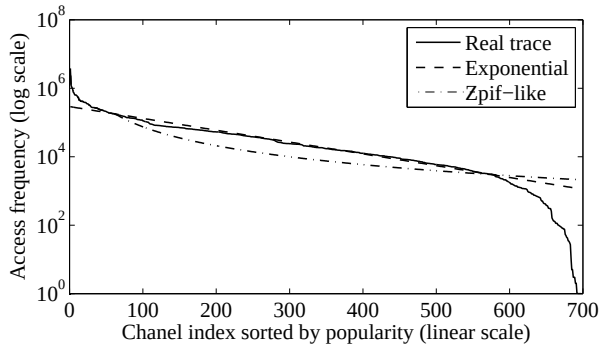


Figure 12: Fitting the channel popularity distribution

i, the access probability is a power function of its rank i. Figure 12 shows the channel access frequency as a function of the rank, along with the best-fit power law function and the best-fit exponential function. We find the Zipf-like distribution well captures the top 10% channels while the exponential function achieves a better fit for the large "body" part of the distribution function. The parameters for the Zipf-like distribution, $f_1(i) = C_1 i^{-\alpha}$, are $\alpha = 0.513$, $C_1 = 12.642$. The parameters for the exponential function, $f_2(i) = e^{-\beta+C_2}$, are $\beta = 0.006$, $C_2 = 2.392$. In the rest of the paper, we use a hybrid model – approximating the top 10% of the channel popularity distribution using the above Zipf-like power-law function and the remaining part using the exponential function. Particularly, the probability density function can be expressed as follows,

$$f_0(i) = \begin{cases} C_1 i^{-\alpha}/C_0 & i < 10\% \text{ of available channels,} \\ e^{-\beta+C_2}/C_0 & \text{others,} \end{cases}$$

where C_0 is the normalization factor such that $f_0(\cdot)$ is a well-defined probability density function.

The concatenated distribution function achieves a good match for the top 600 popular channels, which together account for over 97% of the channel-switches (as shown in Figure 5).

Channel popularity in terms of channel access frequencies is only applicable to target-switching. For sequential-scanning, the channel number simply increments or decrements. We define the probability of user entering target-switching mode as p_t, which is 0.44 in our data. The probability of user entering sequential-scanning mode is hence $1 - p_t$. When in sequential-scanning state, a user switches to a higher number with the probability of p_u (0.72 in our data), and to a lower number with $1 - p_u$.

To align the channel ID to the channel popularity, we adopt a simple random permutation method – we randomly shuffle the ranks of the channel popularity and use them as the channel ID. This however does not capture the subtle clustering effect in the commercial channel listing, such as music channels being next to each others. Depending on the application, a detailed modeling of such effects can be of interest.

4.4 Modeling Channel Popularity Dynamics

The channel popularity model described in the previous section captures popularity skewness, which have been found relatively stable at large time scale (e.g., daily [15]). However, we also observe from our data that channel popularity exhibits some temporal patterns over time-of-day (See Figure 7). While a stationary channel popularity model might be sufficient for many applications (for example network capacity planning analysis), we expect that some other applications (for example evaluating a P2P type content caching scheme for IPTV) may require a proper modeling of such channel popularity dynamics. One way of modeling such dynamics is to observe the differences across multiple smaller time intervals (e.g., hourly granularity) and model the channel popularity

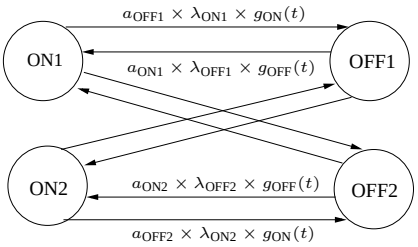


Figure 13: ON-OFF model

in each small interval separately. Alternately, we can try to understand the underlying structure producing such dynamics and model this underlying process. In fact, in our recent work [15], we demonstrate that the channel popularity dynamics can be well explained by groups of users that have intrinsically different channel preference and tend to watch TV at different time of day, as described next.

We divide STBs into multiple classes according to some feature. In [15], we have compared different choices for such feature. To model the daily dynamics of channel popularity, we choose *average daily watching time* as our classifier because (1) the resulting subgroups exhibit distinct and stable channel preference and (2) the STBs in each subgroup tend to affiliate with the same subgroup over time. Specifically, we classify STBs using two thresholds, and in our data, 28% of STBs are *heavy-watchers* (12 hours or longer average daily watching time), 36% of them *light-watchers* (1 hour or shorter average daily watching time), and the remaining 36% *medium-watchers*. We find that the channel preference among STBs in a particular subgroup stays stable (e.g., throughout a day), and the overall channel popularity dynamics is largely due to the change in the population mix of those groups [15].

We thus extend the model to a multiple-class population model. We first define the membership ratio for different subgroups using the numbers above. Next, we identify the channel preference within a subgroup (which is stable and follows Zipf-like distribution) and characterize the session lengths and moderating functions for each subgroup separately. This would capture the change of population mix over time. As demonstrated in [15], this method leads to a highly accurate model of channel popularity dynamics. We present more details and evaluations of this method in the next section.

5. SIMULWATCH: A WORKLOAD GENERATOR

Thus far we have constructed several models to characterize various aspects of IPTV user activities. In this section, we present our design of SIMULWATCH – user TV watching activity generator. We validate our tool by comparing its output with the real data traces. We also demonstrate how to use this tool to drive the network performance study in an IPTV system (e.g., estimate unicast and multicast traffic rates given the number of subscribers).

5.1 SIMULWATCH Design

For simplicity, we first describe the design of SIMULWATCH based on the single-class population model. Then we present the extension using the multi-class population model if the dynamics of channel popularity is of interest.

In the single-class population model, we first focus on generating switching-on and switching-off events matching both on- and off-session length distributions and the aggregate event rates. We define a closed-population ON-OFF model where both ON and OFF states comprise several sub-states, each of which corresponds to one of the mixture exponential distributions in Section 4.1. Figure 13 illustrates the structure of the ON-OFF model with 2 sub-states in each of the ON and OFF states. The transition rate between sub-states are constructed using the parameters in Sections 4.1 and 4.2. For example, the transition rate from ON_i to OFF_j state is

$$a_{\text{OFF},j} \times \lambda_{\text{ON},i} \times g_{\text{ON}}(t)$$

and similarly the reverse direction rate from OFF_j to ON_i is

$$a_{\text{ON},i} \times \lambda_{\text{OFF},j} \times g_{\text{OFF}}(t)$$

To drive the event simulation, assuming a STB arrives at state ON_i at time t, we can easily determine the edge of the next transition using the branching probabilities $a_{\text{OFF},j}$, and we can also determine the time of the next transition, $t+x$, using the following probability density function

$$\phi_{\text{ON},i}(x;t) = \lambda_{\text{ON},\,i} \times g_{\text{ON}}(t+x) \times e^{-\lambda_{\text{ON,i}} \int_t^{t+x} g_{\text{ON}}(y)dy}$$

We next focus on generating channel-switch events. It is not hard to see that the timing of channel-switch events can be determined in the same fashion as those of switching-on or switching-off events. There are two subtle details worth noting. First, we need to trigger the event generation for a next channel-switch event not only at the time of the previous channel-switch, but also when a new switching-on event takes place. Second, we need to cancel a pending channel-switch event if a switching-off event from the same STB takes place first.

In order to determine which channel to switch to, we keep track of, for each STB, the last channel watched. At the time of a scheduled channel-switch event, assuming the last channel watched is i with popularity rank r_i, we compute the probability that the next channel is j with rank r_j as follows.

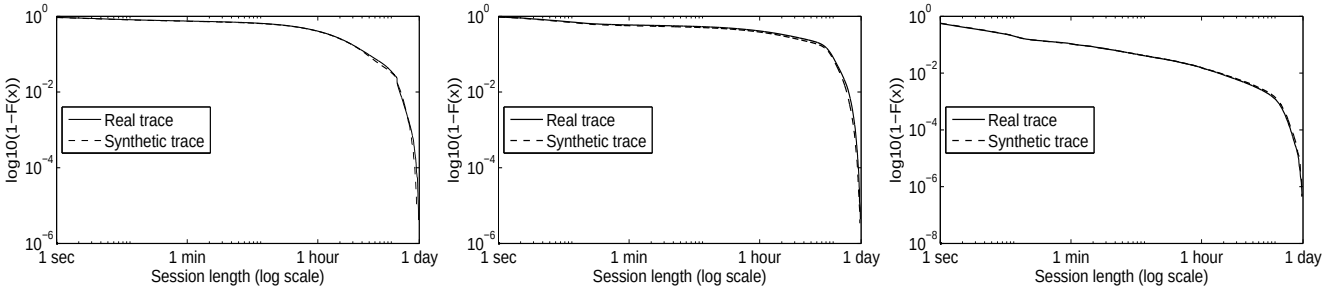
$$\text{Probability} = \begin{cases} (1-p_t)p_u + p_t f_0(r_j) & j = i+1, \\ (1-p_t)(1-p_u) + p_t f_0(r_j) & j = i-1, \\ p_t f_0(r_j) & |i-j| > 1. \end{cases}$$

The initial rank r_i is randomly assigned as described in Section 4.3. The definition of f_0 and all other parameters involved are defined in Section 4.3.

Now we have described the design of SIMULWATCH using the single-class population model. We will show that the above procedure simulating the channel switches cannot precisely generate the dynamics of channel popularity. To equip SIMULWATCH with this functionality, we add an extension of the multi-class population model as follows. Assume that we obtain N classes/groups, each of which consists of a fixed proportion, $p_i, i = 1, 2, ..., N$, of all STBs, where $\sum_{i=1}^{N} p_i = 1$. Then for each STB, we first determinate which group it belongs to based on the probabilities p_i's. Then in each group, the workload is generated using the same method as we described above using the single-class population model with the proper parameters. To generate the synthetic traffic to mimic the dynamics of channel popularity, we can determine the mapping between channel ID and its popularity rank within each group and then the channel popularity can be calculated by combining the results from all the groups.

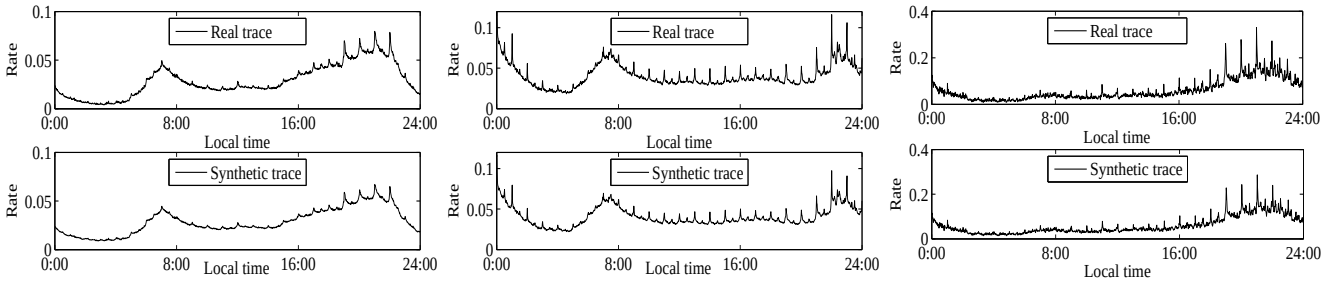
5.2 Evaluation

In this subsection, we will evaluate whether the synthetic traces generated by our SIMULWATCH mimic the real user activities very well. We do so by comparing the synthetic traces and real traces



(a) On-session (b) Off-session (c) Channel-session

Figure 14: Comparison of the session-length distribution. CCDFs for the real trace and generated workload closely match in all cases.



(a) Switching-on (b) Switching-off (c) Channel-switch

Figure 15: Comparison of the aggregate event rate. The real-trace results are on the top, and the workload results are on the bottom.

from three aspects – (i) properties that we explicitly model such as session length distribution, aggregate event rate, and channel popularity distribution, (ii) properties we do not explicitly model like channel popularity dynamics and numbers of on-line STBs, and (iii) a case study on estimating the bandwidth consumed by simultaneous unicast streams, and concurrent multicast channels at different time.

In our experiments, we generate synthetic user activities for two millions STBs and 700 channels based on model parameters listed in Section 4. Each STB starts from a random state at time 0 and we discard the initial part of the output until the system reaches a steady state. On a PC with 2.4GHz CPU and 4GB memory, it takes about 5 hours for our implementation of SIMULWATCH to generate one-day worth of data. We compare the synthetic trace against the real trace collected on a different date (April 8, 2009) than the dates from which we derive the model parameters. Since the single-class population based workload generator works reasonably well for many properties, we use the single-class population model unless specified otherwise for the interest of simplicity. We also use the multi-class population model when illustrating its capability in capturing the dynamics of channel popularity.

5.2.1 Properties explicitly modeled

Session-length distribution: Figures 14 shows the session-length distribution of different types of sessions, where we observe an exceptionally good match between the real trace and the synthetic trace from SIMULWATCH by visual inspection. In order to qualitatively measure the closeness of two distributions, we further compute the *goodness-of-fit*. In the chi-square goodness-of-fit computation, we divide the data into m bins and test

$$\chi^2 = \sum_{i=1}^{m} (O_i - E_i)^2 / E_i$$

where O_i is the observed frequency for bin i (generated by model) and E_i is the expected frequency for bin i (collected from the real trace). The smaller the value is, the better the model and trace match. First, we want to test whether observation O can be considered as arising from the same distribution as E. We represent it through associated one-sided chi-square P-value $P(\chi^2)$, i.e., the proportion of the time that a value of χ^2 or greater would be obtained if O and E were drawn from the same distribution. For a hypothesis testing at significant level P_0, we reject the null hypothesis (O and E are from the same distribution) if $P(\chi^2) < P_0$.

Table 3: Goodness-of-fit scores for session length and channel popularity distributions

Model	Session length			Channel popularity
	ON	OFF	Channel	
Single-class	0.147	0.132	0.132	0.083
Multi-class	0.099	0.089	0.091	0.067

Table 3 shows the goodness-of-fit for session length distribution (the bin size is 1 minute). Using a common significant level $P_0 = 5\%$, we see that in all cases the two session length distributions (synthetic trace and real trace) are *statistically* the same. In addition, multi-class population model yields smaller goodness-of-fit score, indicating that it can fit the real trace better than the single-class population model.

Table 4: RMSE when modeling the time-varying rate

Model	Switch-on	Switch-off	Channel-switch
Single-class	2.3e-3	2.4e-3	2.5e-3
Multi-class	1.8e-4	1.9e-4	2.4e-4

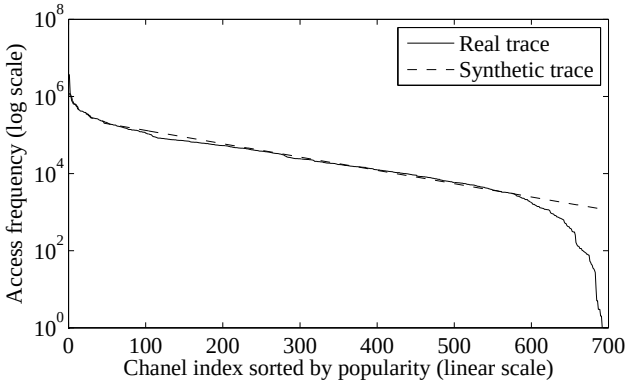


Figure 16: Channel popularity distributions for the real trace and the generated workload.

Aggregate event rate: Figure 15 shows the aggregate event rate of different types of events. Again, we observe a reasonably good match between the model and real trace from visual inspection. Table 4 shows the root mean square error (RMSE) between the model and real trace when modeling the time-varying rate. We find that the RMSE of multi-class population model is an order of magnitude smaller that that of single-class population model. For example, the RMSE of modeling switch-on events by using single-class population model is 2.3e-3 where the corresponding figure by using multi-class population model is 1.8e-4. The reduction in RMSE is due to the fact that much more (triple) parameters are used in the multi-class population model (i.e., more details have been modeled). Depending on the applications of SIMULWATCH, this may or may not be a desirable property.

Channel popularity distribution: Figure 16 shows the channel popularity distributions for both the real trace and synthetic trace. As expected, the popularity of top 600 channels matches very well. Table 3 shows the goodness-of-fit for channel popularity distribution (the bin size is 1 channel). Again, we observe that the multi-class population model yields smaller goodness-of-fit score, indicating that it can fit the real trace better than the single-class population model.

5.2.2 Properties not explicitly modeled

Number of on-line STBs: Figure 17 shows the average number of on-line STBs as a function of the time-of-day. We normalize both synthetic trace and real trace such that the value at their peak time is 1. This is a property that we do not model directly, however, we still find a decent match in their shape.

Channel popularity dynamics: Recall that in Section 3 we showed that the channel popularity distribution changes over time within a single day. Taking a kids channel as an example, we show the change of channel popularity in Figure 18. The change is normalized such that the maximum value is 1. We observe that the strength of using multi-class population model is that this model can capture the dynamics of channel popularity very well, while single-class population model fails. Note that the curve for the single-class population model is actually similar to the curve of

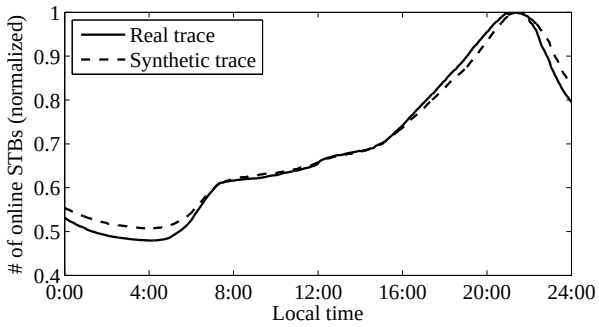


Figure 17: Number of on-line STBs over time. The results from real trace and workload closely match.

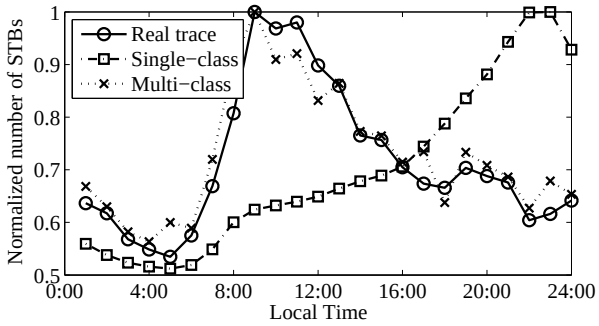


Figure 18: Multi-class population model captures the change of channel popularity over time (hourly)

online STBs over time in Figure 17 because the single-class population model treats every channel-switch event uniformly. Therefore, the changing rate of channel popularity is proportional to the changing rate of on-line STB population.

To better understand the reasons that a multi-class population model can better capture the dynamics of channel popularity, we drill down the above example. There are two factors which shape the popularity dynamics of a particular channel: i) the channel access probability defined as the number of STBs in a particular population (for multi-class population model) watching that channel divided by the number of online STBs in that population at that time period (e.g. one hour). It is actually the transition probability of that channel; ii) the population mix which consists of the proportions of different subgroups in the whole user population. The channel popularity dynamics is generated by the combinational effect (which can be viewed as the weighted sum of productions of subgroups) of these two factors.

Figure 19 shows the channel access probability of a popular kids channel from our multi-class synthetic trace. We display one curve for each group and an additional curve for all-STB cases (denoted by "All"). We observe that the curve for all-watchers shows a diurnal pattern, but all the other curves are quite stable. This means the change of the channel access probability in each group is very small and hence does not contribute to the dynamics of channel popularity in Figure 18. Figure 20 illustrates the population mix in our synthetic trace. By [15], the light-watcher group consists of the majority of watchers for kids channels. When we compare Figure 19 with Figure 20, it is clear that the increase bump (approximately 5:00 – 14:00) of light-watcher group in population mix contributes to the spike view of Figure 18.

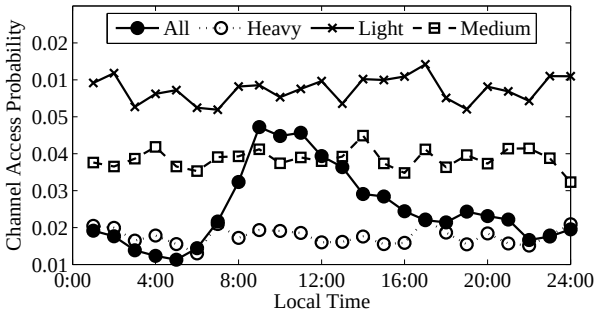


Figure 19: Time-of-day dynamics for a popular kids channel, based on multi-class synthetic trace

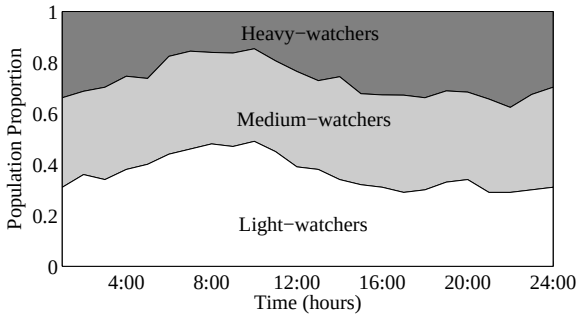
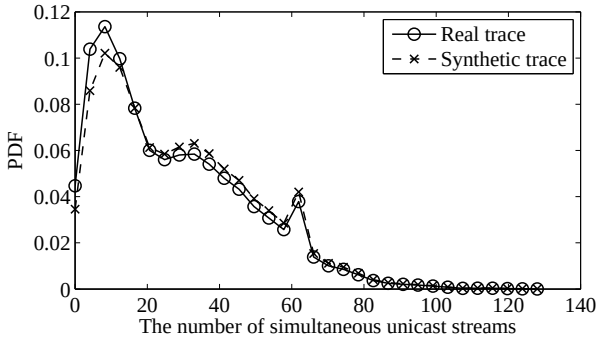
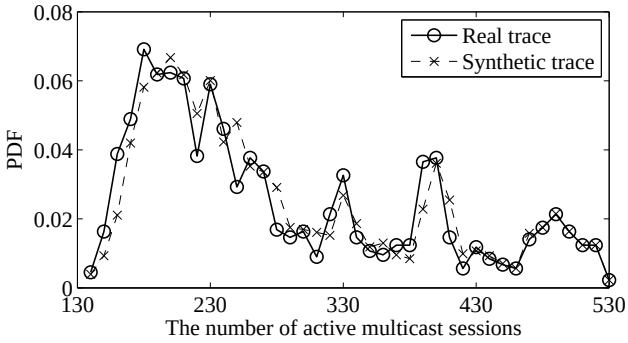


Figure 20: Population mix for each group, based on multi-class synthetic trace



(a) Simultaneous unicast stream count



(b) Active multicast channel count

Figure 21: Case study. The results from real trace and workload closely match.

5.2.3 Case Study

Since our ultimate goal is to use the synthetic trace from SIMULWATCH in evaluating the performance of different design of IPTV system, different system parameter settings, etc., we put SIMULWATCH to a final test by using it in a case study. In particular, we are interested in evaluating the bandwidth requirement to support fast-channel-switch. Recall that using fast-channel-switch, a short (x seconds) unicast stream is transmitted to the STB (in addition to a new multicast stream) when a user switches to a new channel. We focus on a single router in one of the VHOs in the IPTV network, which connects to $2,137$ downstream STBs. We evaluate different value of x (4, 8, 16, 32). We only present result for $x = 32$ seconds, while other results are quantitatively very similar.

We study the number of simultaneous unicast streams flowing downstream from the router of interest under the above settings. Figure 21a shows the distribution density function on the number of concurrent unicast streams when using either real trace or synthetic trace in evaluation. We observe that the two curves closely match. Both curves show that for around 4% of time there is no unicast stream in the system; with a small probability, there can be demand for more than 80 concurrent unicast streams, with the maximum being 128 in both cases; and interestingly there are two local peaks (at 10 and 60) in both distribution functions, the second of which may relate to the correlated channel switchings at hour boundaries. This result demonstrates that SIMULWATCH faithfully preserves the intrinsic characteristics of user activities that are essential to our evaluation.

We also examine the number of channels that these $2,137$ STBs collectively request. This value translates to the amount of multicast traffic involved to support live TV viewing for the users. In Figure 21b, we report the probability density function for the number of channels, in which we confirm that the result from synthetic trace closely matches that of real trace.

6. RELATED WORK

Traditionally, understanding users' TV viewing activities in the conventional TV systems relied on phone surveys or specialized monitoring boxes (e.g., by Nielsen Media Research [1]). The challenge with that approach is the difficulty conducting a large-scale survey or deploying monitoring boxes for the majority of TV users. In this paper, we analyze user activity data from more than one million commercial IPTV subscribers and present models that can be used to generate realistic user activity workload.

Many researchers recently have looked into various aspects of IPTV systems. The closest work to our study is the recent measurement study conducted by Cha et al. [6]. While some of our findings overlap with their study, our focus is to *model the user activities* based on the measurement study and *design a workload generator*, which can be used to evaluate different aspects of IPTV system design and performance with respect to realistic user workload. In our earlier work [15], we extensively study one aspect of IPTV system: channel popularity. In this paper, we model a wider range of aspects of user activities, and design a workload generator. Smith [17] analyzed bandwidth demand to support both multicast and unicast for fast channel change, where channel switching is modeled as a renewal process. However, the work is not based on actual traces, and such a study can benefit from our workload model and trace generator. Whereas our work focuses on IPTV services running on top of a provider backbone, there are a number of peer-

to-peer (P2P) based IPTV systems [22, 14] and the measurement study focusing on P2P IPTV systems [12, 16].

Some researchers have investigated user activity workload in other context. For example, Costa et al. [9] analyzed user activities in media streaming applications. Cherkasova et al. [7], Chesire et al. [8], and Tang et al. [18] built models for the workload of media streaming service. Yu et al. [21] studied the user activities to access a Video-on-Demand (VoD) system. Cha et al. [5] explored how users access videos in the YouTube system. Guo et al. [10] compared access patterns of different types of media content on the Internet including Web, P2P, VoD, and live streaming. These studies are complimentary to our work in that as IPTV providers offer more interactive video streaming and VoD services, we also need to consider these aspects in the system design.

7. CONCLUSION

In this paper, we have performed an in-depth analysis on several intrinsic characteristics of user activities in large IPTV systems, including durations for on-, off- and channel-sessions, time-varying rates of switching-on, switching-off and channel switching events, and channel popularity. We have also developed a series of practical mathematical models to capture these characteristics. Furthermore, we construct the first IPTV user activity workload generation tool SIMULWATCH, which can generate synthetic yet realistic activity traces of a large number of IPTV users. All the derived models and the implementation of SIMULWATCH have been validated using real traces collected from a large nationwide IPTV provider in the United States. In particular, we demonstrate that while not explicitly modeled, the estimation of unicast and multicast traffic demand based on SIMULWATCH trace closely matches the actual values from the real trace. We believe that SIMULWATCH will prove useful in many different aspects of IPTV system design and evaluation.

Our future work includes several extensions to our current model. We plan to include proper modeling for the use of advanced features in IPTV, such as PIP and DVR. We also expect that users' activities likely change over time as IPTV providers introduce more features. It would be also interesting to analyze such changes and evaluate the performance impact those new features impose on IPTV systems.

8. REFERENCES

[1] The Nielsen Company. `http://www.nielsenmedia.org`.

[2] Dakshi Agrawal, Mandis S. Beigi, Chatschik Bisdikian, and Kang-Won Lee. Planning and Managing the IPTV Service Deployment. In *10th IFIP/IEEE International Symposium on Integrated Network Management*, pages 353–362, 2007.

[3] Paul Barford and Mark Crovella. Generating representative web workloads for network and server performance evaluation. In *SIGMETRICS*, pages 151–160, 1998.

[4] Kathryn Jo-Anne Barger. Mixtures of exponential distributions to describe the distribution of poisson means in estimating the number of unobserved classes. Master's thesis, Cornell University, 2006.

[5] Meeyoung Cha, Haewoon Kwak, Pablo Rodriguez, Yong-Yeol Ahn, and Sue Moon. I Tube, You Tube, Everybody Tubes: Analyzing the World's Largest User Generated Content Video System. In *Proceedings of ACM IMC*, 2007.

[6] Meeyoung Cha, Pablo Rodriguez, Jon Crowcroft, Sue Moon, and Xavier Amatrianin. Watching Television Over an IP Network. In *Proceedings of ACM IMC*, 2008.

[7] Ludmila Cherkasova and Minaxi Gupta. Characterizing locality, evolution, and life span of accesses in enterprise media server workloads. In *NOSSDAV*, 2002.

[8] Maureen Chesire, Alec Wolman, Geoffrey M. Voelker, and Henry M. Levy. Measurement and analysis of a streaming media workload. In *USITS*, pages 1–12, 2001.

[9] Cristiano P. Costa, Italo S. Cunha, Alex Borges Vieira, Claudiney Vander Ramos, Marcus M. Rocha, Jussara M. Almeida, and Berthier A. Ribeiro-Neto. Analyzing client interactivity in streaming media. In *WWW*, 2004.

[10] Lei Guo, Enhua Tan, Songqing Chen, Zhen Xiao, and Xiaodong Zhang. The stretched exponential distribution of internet media access patterns. In *PODC*, pages 283–294, 2008.

[11] Chris Harrison and Brian Amento. CollaboraTV: Using Asynchronous Communication to Make TV Social Again. In *EuroITV*, 2007.

[12] Xiaojun Hei, Chao Liang, Jian Liang, Yong Liu, and Keith W. Ross. A measurement study of a large-scale p2p iptv system. *IEEE Transactions on Multimedia*, 9(8):1672–1687, 2007.

[13] Nicholas P. Jewell. Mixtures of Exponential Distributions. In *Annuals of Statistics*, 1982.

[14] Xiaofei Liao, Hai Jin, Yunhao Liu, Lionel M. Ni, and Dafu Deng. Anysee: Peer-to-peer live streaming. In *INFOCOM*, 2006.

[15] Tongqing Qiu, Zihui Ge, Seungjoon Lee, Jia Wang, Qi Zhao, and Jun (Jim) Xu. Modeling Channel Popularity Dynamics in a Large IPTV System. In *SIGMETRICS*, 2009.

[16] Thomas Silverston, Olivier Fourmaux, Kavé Salamatian, and Kenjiro Cho. Measuring p2p iptv traffic on both sides of the world. In *CoNEXT*, page 39, 2007.

[17] Donald E. Smith. IPTV Bandwidth Demand: Multicast and Channel Surfing. In *INFOCOM*, pages 2546–2550, 2007.

[18] Wenting Tang, Yun Fu, Ludmila Cherkasova, and Amin Vahdat. Medisyn: a synthetic streaming media service workload generator. In *NOSSDAV '03*, pages 12–21, 2003.

[19] J. Weber and J. Gong. Modeling switched video broadcast services. In *Cable Labs*, 2003.

[20] Young J. Won, Mi-Jung Choi, Byung-Chul Park, Hee-Won Lee, Chan-Kyu Hwang, and Jae-Hyoung Yoo. End-user iptv traffic measurement of residential broadband access networks. In *NOMS Workshops 2008*, 2008.

[21] Hongliang Yu, Dongdong Zheng, Ben Y. Zhao, and Weimin Zheng. Understanding user behavior in large-scale video-on-demand systems. In *EuroSys*, pages 333–344, 2006.

[22] Xinyan Zhang, Jiangchuan Liu, Bo Li, and Tak-Shing Peter Yum. Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming. In *INFOCOM*, pages 2102–2111, 2005.