Towards the Optimal Caching Strategies of Peer-Assisted VoD Systems with HD Channels

Le Chang, Jianping Pan
University of Victoria, Victoria, BC, Canada
Email: {lechang, pan}@uvic.ca

Abstract—In this paper, we propose a modeling framework to capture the major characteristics of peer-assisted video-on-demand systems offering standard and high-definition channels. Our framework can be extended to model a variety of caching strategies, including FIFO, passive caching, and active caching. We use the framework to prove that passive caching is sufficiently effective for stationary user behaviors, and generate the optimal caching solutions when the channels in the system demonstrate different popularity evolutions, i.e., with non-stationary behaviors. Simulation results verify the efficacy of our active-caching strategy and provide further insights into such systems that are gaining more popularity over the Internet.

Index Terms—Peer-to-peer, video on-demand (VoD), view-upload decoupling (VUD), bandwidth allocation, caching strategy

I. INTRODUCTION

Nowadays, popular peer-assisted video-on-demand (PA-VoD) systems attribute their great success to the vast amount of resources, such as the upload bandwidth and cache space, contributed by individual peers. These peer-contributed resources not only make P2P systems scalable, but also lower the server bandwidth consumption, and thus the cost of providing such services [1]–[3]. This trend boosts the emerging of large-scale multi-channel online VoD systems with higher video quality, e.g., offering high-definition (HD) movies.

However, peer-contributed resources vary dramatically between channels, especially when HD movies are offered. Such a great “resource imbalance” can arise due to a variety of reasons. First, HD channels usually require more bandwidth supply and cache space, which exceeds the capacity of the participating peers, while peers in standard-definition (SD) channels have bandwidth surplus [4], [5]. Moreover, a great challenge is presented concerning the newly released movies [6]. As a huge number of peers watching and few contributing, it imposes a heavy instantaneous bandwidth consumption at the server just after the movie is released.

To overcome the resource imbalance problem, it is desirable to utilize the surplus bandwidth of well-provisioned channels to help compensate the bandwidth deficit of other channels, i.e., view-upload decoupling (VUD). VUD strategy is originally for live-streaming systems [7], [8], where the peer upload bandwidth is allocated across channels, regardless of what a peer is currently watching. The objective is to minimize the server bandwidth consumption by equalizing the bandwidth demand-to-supply ratio for each channel, i.e., a “water-leveling” approach. However, VUD calls for a careful redesign in order to adapt to VoD systems with HD channels. First, VoD users are likely to watch different parts of a movie. The video chunks requested by peers can possibly spread over the entire movie and change frequently over time, which makes it difficult to achieve real-time resource provisioning for each chunk. Second, the playback rate of HD movies offered by VoD systems usually exceeds the critical threshold, the average peer upload capacity. Besides, the peer cache is limited and a single HD movie may occupy the entire cache space. As a result, the resource balancing for VoD systems involves two components, the management of peer upload bandwidth and cache space, and a proper coordination in between is required.

Concerning VUD in PA-VoD systems, existing work relies on relaxed assumptions to simplify the analysis. The stationary model is widely used to describe user behaviors, in which users form a closed Jackson queueing network and transit between channels (queues) with fixed probabilities, i.e., the popularity of each channel will never change [6], [8]–[10]. Although these models are applicable to a sufficiently stable system within a short period (e.g., several hours), they miss the dynamic nature of PA-VoD systems, and fail to reflect the influence of the evolution of movie popularity, especially when a new movie is released. Moreover, most existing work focuses on systems with homogeneous, moderate playback rates, e.g., SD channels [6], [9], [12], ignoring the pervasiveness of HD channels in real systems.

In this paper, we model a generic PA-VoD system with the coexistence of SD and HD channels under stationary and non-stationary scenarios, i.e., the arrival rate of users to a channel may change over time (i.e., a non-homogeneous arrival process). We let SD viewers help HD channels in terms of both upload bandwidth and cache space. We design caching strategies to determine what content to remove or fetch at a peer, and the corresponding bandwidth allocation to select peers to upload to or download from. We aim at answering the following questions concerning such systems.

• What is the demand-vs-supply relationship of an HD channel with non-homogeneous arrival processes?
• What is the best caching strategy in terms of server bandwidth consumption when the system is stable (i.e., stationary scenarios), and how to find the best caching strategy when a new movie is released and causing flash crowds (i.e., non-stationary scenarios)?

The main contributions of this paper are as follows.

• We build a modeling framework to capture the major characteristics of a PA-VoD system, e.g., the demand-vs-supply relationship. Our modeling framework can be easily modified or extended to model a variety of user behaviors and caching strategies in real systems.
A detailed model for multi-channel VoD systems is proposed based on the modeling framework. We use the detailed model to prove that passive caching achieves the best performance under stationary scenarios, and derive the server bandwidth consumption with passive or active caching strategies under non-stationary scenarios, as passive caching is not sufficient.

We formulate finding the best caching strategy under non-stationary scenarios as a mixed integer linear programming (MILP) problem. Despite the intractability of general MILP problems, many instances of our optimization problems are actually solvable and generate optimal solutions. We use simulation to verify the efficacy of the optimal (or suboptimal) solutions obtained from the optimization problem, with suitable bandwidth allocation strategies to match the proposed active caching strategy.

To the best of our knowledge, this is the first modeling work for PA-VoD systems with the presence of HD channels and non-stationary user behaviors. The remainder of this paper is organized as follows. Section II discusses the most related work to this paper. In Section III, we describe the modeling framework. Section IV presents the detailed model and derives the server bandwidth consumption with different caching strategies and user behaviors. We discuss the corresponding bandwidth allocation strategies in Section V. These strategies are then extensively evaluated in Section VI. Section VII discusses the future work with conclusions.

II. RELATED WORK

The benefits of utilizing peer-contributed resources have been verified for a long time through measurement studies of real systems [1]–[4], [13]. Although Huang et al. estimated that 95% of the server bandwidth consumption can be saved through a P2P approach [1], the resource imbalance problem often breaks such an optimistic expectation. To balance the bandwidth resource in a single channel, Parvez et al. built analytical models to compare different bandwidth allocation strategies and verified the optimality of the chain-based approaches [14], and Yang et al. proposed practical queuing techniques to achieve such a balance through simulation [15]. Ciullo et al. then used an approximate fluid model to study the chain-based algorithms under both stationary [11] and non-stationary scenarios [16]. For BitTorrent-like systems under a flash-crowd scenario, D’Acuntoa et al. characterized the tradeoff between injecting new video chunks and replicating existing ones [17]. Our work differentiates itself from these studies as we focus on the cooperation of multiple channels rather than the resource allocation within a single channel.

As to multi-channel systems, Wu et al. proposed VUD for P2P live streaming [7], [8]. Wang et al. built linear programming models to investigate the bandwidth allocation under several user-defined viewing behaviors [18]. The similar ideas have been applied to VoD systems [6], [9], [12], [19], [20]. However, existing models heavily rely on impractical assumptions, such as stationary user behaviors, moderate playback rates of video programs, and so on [6], [9], [12], [19]. Some simulation-based studies have also been conducted under non-stationary scenarios, without the mathematically guaranteed optimality of the caching strategies [6], [20]. To the best of our knowledge, modeling PA-VoD systems with HD channels under non-stationary behaviors is still not sufficiently studied, and thus it is the main focus of our work in this paper.

III. MULTI-CHANNEL PA-VoD MODELING FRAMEWORK

A. System Model

In P2P video streaming systems, if the playback rate of a channel is less than the average peer upload bandwidth, e.g., SD channels, the viewers/peers of the channel can adopt a proper bandwidth allocation strategy within the channel (e.g., chain-based algorithms) to achieve the download rate as the playback rate [6], [11], [14], [16]. This means that SD viewers can view the video program without interruption. Meanwhile, the bandwidth support from the server can be minimized, regardless of any kind of user dynamics, and thus such a channel is referred to as a “surplus channel”. On the other hand, there are channels with bandwidth deficit, i.e., the average peer upload bandwidth is less than the playback rate of the channel. Such channels are referred to as “deficit channels”. For instance, HD channels usually have a playback rate up to 1,000 Kbps, while the typical uplink bandwidth of residential Internet access links varies from 384 Kbps to 800 Kbps. Therefore, HD channel needs extra “help” from SD channels or the server(s). We call peers that are from SD channels, but uploading to HD channels as “bandwidth helpers”. To serve as such a “helper”, a peer needs to satisfy two conditions. First, it has unused upload bandwidth available (e.g., SD viewers). Second, the content of the corresponding HD movie has to be stored (“cached”) in its local cache.

Due to the limit of the peer cache space, the caching strategy plays an important role in PA-VoD systems. Cache replacement occurs when the local cache of a peer is full. In order to accommodate new video segments, the peer has to remove some existing ones from its local cache. A simple approach is first-in-first-out (FIFO), in which the earliest segment is removed when necessary. As the FIFO approach is extensively studied in our prior work [10], we focus more on the two caching strategies described as follows.

1) Passive Caching: A peer selects a segment to remove following the water-leveling criterion rather than FIFO, as to balance the resource provisioning to match the dynamic demand at either the movie or segment level. Some examples include least-recently used (LRU) or least-frequently used (LFU). In this paper, we simply let a peer remove early SD segments first that do not belong to the movie being watched, as SD segments are most likely to be well-provisioned.

2) Active Caching: The server can actively introduce more helpers for a particular HD channel, if the channel is observed as badly provisioned and passive caching alone is considered ineffective. Such helpers need to actively fetch the content of the HD channel, and are referred to as “active cachers”.

Intuitively, with a passive caching strategy, users can update their cached content slowly through a natural viewing process
B. Modeling Framework

Figure 1 shows our modeling framework. There are two queues considered in an HD channel $i$, the viewer queue and the helper queue, with the number of residing peers as $x_i$ and $y_i$, respectively. The arrival and departure rates are denoted as $\lambda_i$ and $\theta_i$ for the viewers, and $\eta_i$ and $\gamma_i$ for the helpers, respectively. Notice that such arrival and departure rates are fixed for stationary scenarios and can be time-dependent under non-stationary ones. The residence time of viewers is $t_{i,\text{view}}^r$. After $t_{i,\text{view}}^r$, a peer leaves the viewer queue, and may either become a helper, or leave the current channel. We denote the probability that a viewer becomes a helper as $p_{i,\text{view} \rightarrow \text{helper}}$, and that of leaving the channel as $p_{i,\text{view} \rightarrow 0}$. Once a peer becomes a helper, it stays there for time $t_{i,\text{helper}}^r$, and then makes a decision of whether to stay or leave. Such probabilities are denoted as $p_{i,\text{helper} \rightarrow \text{helper}}$ and $p_{i,\text{helper} \rightarrow 0}$, respectively. For active caching, denote the arrival rate of such active cachers as $\omega_i$. At last, denote the average viewer bandwidth deficit as $d_{i,\text{view}}^r$, and the average bandwidth supplied by helpers as $s_{i,\text{helper}}^r$. Other parameters are also marked in the figure, and the notations we often used in this paper are listed in Table I.

The framework captures the major characteristics of a PA-VoD system, such as the bandwidth deficit of viewers as $x_i(t_{i,\text{view}}^r)$ and the extra bandwidth supply of helpers as $y_i(t_{i,\text{helper}}^r)$ for each channel $i$. The parameters $p_{i,\text{view} \rightarrow \text{helper}}$, $p_{i,\text{view} \rightarrow 0}$, $p_{i,\text{helper} \rightarrow \text{helper}}$, $p_{i,\text{helper} \rightarrow 0}$, $t_{i,\text{view}}^r$, $t_{i,\text{helper}}^r$, and $\omega_i$ can be easily customized to model different caching strategies under a variety of scenarios, and $x_i$, $y_i$, $s_{i,\text{helper}}^r$ and $d_{i,\text{view}}^r$ are derivable after these parameters are specified. For instance, $\omega_i = 0$ indicates FIFO or passive caching, and $\omega_i > 0$ represents active caching. Moreover, for stationary scenarios, we can derive the steady-state metrics by assuming the arrival rate of each queue is equal to its departure rate. If $\lambda_i$ is dependent on time, i.e., non-stationary scenarios, the framework is able to capture the effect of the evolution of movie popularity or user dynamics, and thus the cost of accommodating such dynamics at the server. At last, the framework is not limited to a single HD channel. If the transition matrix on channels is given, the probability $p_{i,\text{view} \rightarrow \text{helper}}$ for any channel $i$ can be derived, and thus the server bandwidth consumption of all the channels.

IV. Multi-Channel PA-VoD: A Detailed Model

In this section, we customize our modeling framework in the previous section to model a multi-channel PA-VoD system with the coexistence of SD and HD channels. In order to facilitate further analysis, we first make several assumptions, most of which are based on existing measurement studies and also widely used by other modeling work in the literature.

A. Model Description and Assumptions

We assume two categories of the video channels in the system, SD channels with playback rate $r_S$, and HD channels with playback rate $r_H$. For simplicity, we set $r_H = 2r_S$, a typical case with $r_H = 1,000$ Kbps and $r_S = 500$ Kbps. The average peer upload bandwidth is assumed to be $r_S < \bar{u} < r_H$ as observed in real systems [4], [5]. Therefore, the bandwidth deficit of HD channels per viewer is $d_{i,\text{view}}^r = r_H - \bar{u}$, and an SD viewer has an average surplus bandwidth of $s_{i,\text{helper}}^r = \bar{u} - r_S$. We let HD viewers allocate their bandwidth within their viewing channels only, as these channels already suffer from the bandwidth deficit. SD viewers will perform the bandwidth

---

**TABLE I: Important Notations**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bar{u}$</td>
<td>The average upload capacity of all peers</td>
</tr>
<tr>
<td>$x_i(k), y_i(k)$</td>
<td>The number of viewers, helpers of channel $i$ in round $k$</td>
</tr>
<tr>
<td>$\hat{x}_i, \hat{y}_i$</td>
<td>The expected number of viewers, helpers of channel $i$</td>
</tr>
<tr>
<td>$\lambda_i(k), \theta_i(k)$</td>
<td>The arrival, departure rate of the viewers of channel $i$ in round $k$</td>
</tr>
<tr>
<td>$\eta_i(k), \gamma_i(k)$</td>
<td>The arrival, departure rate of the helpers of channel $i$ in round $k$</td>
</tr>
<tr>
<td>$\omega_i(k)$</td>
<td>The number of introduced active cachers of channel $i$ in round $k$</td>
</tr>
<tr>
<td>$N, M$</td>
<td>The total number of peers, channels in the system</td>
</tr>
<tr>
<td>$N_S, N_H$</td>
<td>The expected number of viewers in SD, HD channels in the steady state</td>
</tr>
<tr>
<td>$t_a^i, t_b^i$</td>
<td>The residence time of queue $a$ of channel $i$</td>
</tr>
<tr>
<td>$p_{i, a \rightarrow b}^r$</td>
<td>The transition probability from queue $a$ to $b$ of channel $i$</td>
</tr>
<tr>
<td>$p_{i, a}^r$</td>
<td>The probability of leaving channel $i$ from queue $a$</td>
</tr>
<tr>
<td>$d_{i,\text{view}}^r$</td>
<td>The average viewer bandwidth deficit of channel $i$</td>
</tr>
<tr>
<td>$s_{i,\text{helper}}^r$</td>
<td>The average helper bandwidth contribution of channel $i$</td>
</tr>
<tr>
<td>$r_S, r_H$</td>
<td>The playback rate of an SD, HD movie</td>
</tr>
<tr>
<td>$T$</td>
<td>The time duration of each movie</td>
</tr>
<tr>
<td>$F$</td>
<td>The size of the peer cache</td>
</tr>
<tr>
<td>$p_{i,\text{view} \rightarrow \text{helper}}$</td>
<td>The probability of leaving the VoD system from channel $i$</td>
</tr>
<tr>
<td>$p_{i,\text{view} \rightarrow 0}$</td>
<td>The transition probability from category $a$ to $b$</td>
</tr>
<tr>
<td>$p_{i,\text{helper} \rightarrow \text{helper}}$</td>
<td>The transition probability from channel $i$ to $j$</td>
</tr>
<tr>
<td>$p_{i,\text{helper} \rightarrow 0}$</td>
<td>The stationary popularity of channel $i$</td>
</tr>
<tr>
<td>$q_{i,\text{max}}$</td>
<td>The capacity of the server(s)</td>
</tr>
<tr>
<td>$SBC$</td>
<td>The expected total server bandwidth consumption at the steady state in stationary scenarios</td>
</tr>
<tr>
<td>$SBC_{i, t_a, t_b}$</td>
<td>The server bandwidth consumption for channel $i$ in round $k$</td>
</tr>
<tr>
<td>$D, S$</td>
<td>The total bandwidth demand, supply of peers</td>
</tr>
<tr>
<td>$R$</td>
<td>The total number of segments of an HD movie</td>
</tr>
<tr>
<td>$f$</td>
<td>The byte length of the video fetched by an active cacher</td>
</tr>
<tr>
<td>$s_{i,\text{helper}}$</td>
<td>The compensation cost of the server(s) for channel $i$</td>
</tr>
<tr>
<td>$N_{sub}$</td>
<td>The number of substreams for an HD movie</td>
</tr>
</tbody>
</table>
allocation within their watching channels first, and then use the surplus bandwidth to help HD channels.

Each peer contributes some local cache space of a finite size. To reflect the fact that HD movies occupy more cache space, we assume that the peer local cache can accommodate either one complete HD movie or two SD ones. The time duration of each movie is \( T \). With \( r_S = 500 \) Kbps and \( r_H = 1,000 \) Kbps, assume the size of the local cache in each peer is \( F = 1 \) GB (e.g., in PPLive [2]), the duration of any movie is 2.33 hours, which is typical for the duration of an on-line full-length movie. The movie file is also equally divided into small segments, and each segment has the same byte length (e.g., 20 segments for an HD movie and 10 for SD). Each SD viewer will use a half of the cache space to preserve the second half of the HD movie previously watched. The other half is used to store the SD movie that is being watched.

When watching a movie, each peer starts from the beginning of the movie and watches in its entirety for time \( T \), i.e., VCR operations are not considered in this paper. After that, the peer can select another movie to watch, or leave the VoD system with probability \( p_{1 \to \text{quit}} \). We also assume peers stream exactly at the playback rate of a movie, which is also a common assumption in [6], [11], [16], [17]. Such an exact-rate streaming can be easily implemented using a sliding window of a fixed size: all the segments within the window can be requested for downloading, while the priority is given to the segments closest to the current playback point; the window is moving forward as the segments are consumed by the media player. To model the user behaviors across channels, we assume two transition matrices, \( P^c \) at the category level and \( P^m \) at the movie level. \( P^c \) is a two-by-two matrix containing the transition probabilities between SD and HD categories, i.e., \( p_{SS}, p_{SH}, p_{HS} \) and \( p_{HH} \), where S represents SD movies and H represents HD movies. For instance, \( p_{HS} \) represents the probability that a peer finishes watching an HD movie and transits to watching an SD movie. An element \( p_{ij}^m \) in \( P^m \) is referred to as the transition probability between movie \( i \) and \( j \), which is also adopted in [6], [8], [9]. Note that \( P^c \) and \( P^m \) have to be consistent to each other.

There is a server (or a virtual cluster of servers) with capacity \( u_{\text{max}} \), which is able to help any peer at any time if the current bandwidth consumption does not exceed \( u_{\text{max}} \). The instantaneous server bandwidth consumption (SBC) at time \( t_k \) is denoted as \( \text{SBC}(t_k) \), and the expected SBC for the steady state is \( \overline{\text{SBC}} \) for stationary scenarios. In order to evaluate different caching strategies under non-stationary scenarios, we define the average server bandwidth consumption over a time duration of \( t_a \) to \( t_e \) as \( \text{SBC}_{t_a \to t_e} \), which is the aggregated data volume that the server injects into the system over the period \( t_e - t_a \). \( \text{SBC}_{t_a \to t_e} \) quantifies the cost to run the server from time \( t_a \) to \( t_e \). For instance, if a new movie is released at time \( t_a \) and the system becomes stable at time \( t_e \), \( \text{SBC}_{t_a \to t_e} \) can be interpreted as the cost of releasing the new movie.

With the assumptions above, we now customize all the critical parameters in our modeling framework to create the detailed model. An SD viewer is considered as a helper if it transits from an HD channel to an SD channel, as it has both extra bandwidth and the content of the previously watched HD movie, so \( p_{1 \to \text{view} \to \text{help}} = (1 - p_{1 \to \text{quit}}) p_{HS} \). Since a helper will preserve HD segments when it is watching SD movies due to passive caching, \( p_{1 \to \text{help} \to \text{help}} = (1 - p_{1 \to \text{quit}}) p_{SS} \). Each movie lasts for time \( T \), and each peer watches a movie in its entirety, after which it leaves the viewer/helper queue, i.e., \( t_{\text{view}}^i = t_{\text{help}}^i = T \). In stationary scenarios, we can calculate the stationary arrival rate \( \lambda_i \) for each channel using the transition matrix \( P^m \), and assume \( \lambda_i(t_k) \) is known for any \( t_k \) in non-stationary scenarios. \( \omega_i \), specially for the active caching strategy, can either be given to evaluate the system performance, or be the unknown variables in an optimization problem minimizing certain server cost, e.g., bandwidth.

A diagram of the detailed model for an HD channel (HD 2) is shown in Fig. 2. The arrows indicate the possible transitions of peers. If Viewer 1 decides to watch SD channel 5 (SD 5) after it finishes HD 2, it becomes a helper of HD 2, i.e., through the passive caching process. However, a viewer may select another HD movie, or leave the VoD system, as illustrated by the two choices of Viewer 2. A helper (e.g., Helper 1) may also choose to stay if it always watches SD movies, leave the channel if transiting to an HD channel, or leave the VoD system. Moreover, the server can actively introduce helpers through active caching, e.g., Helper 2. Note that HD movies are not necessarily cached in their entirety, due to the limited size of the local peer cache.

\[\text{Fig. 2: The PA-VoD system with passive/active caching.}\]

### B. Optimal Caching in Stationary Scenarios

In stationary scenarios, assume there are \( N \) peers and \( M \) channels in the system, let a vector \( \vec{p}^m = (p_1^m, p_2^m, \ldots, p_M^m) \) denote the stationary probability distribution that satisfies \( \vec{p}^m = \vec{p}^m P^m \). The expected number of viewers of channel \( i \) in the steady state is thus \( \pi_i = N p_i^m \). Now we prove that a simple passive caching strategy achieves the best (or minimum) SBC for the steady state of the system, i.e., \( \omega_i = 0 \), \forall i.

**Theorem 1.** Under stationary scenarios, assume the server has unlimited capacity, passive caching achieves the optimal
expected server bandwidth consumption \( \overline{SBC}_{\text{opt}} = (D - S)^+ \), where \( D = \sum_{i=1}^{M} x_i r_i \) and \( S = N \bar{u} \) are the total bandwidth demand and supply of all peers, respectively, \( r_i \) is the playback rate of channel \( i \), and \( a^+ := \max\{a, 0\} \).

**Proof:** The optimal SBC in the steady state is calculated as \( \overline{SBC}_{\text{opt}} = (D - S)^+ = (N_S \bar{p} + N_H \bar{h} - N \bar{u})^+ \), where \( N_S = N_p \bar{p} / (\bar{p} + \bar{h}) \) and \( N_H = N_p \bar{h} / (\bar{p} + \bar{h}) \) are the expected number of SD and HD viewers in all channels, respectively. In the steady state, the arrival rate of each queue in Fig. 1 is equal to the corresponding departure rate. Thus \( \lambda_i = \bar{x}_i / \tau_i = \bar{p} \bar{h}_i + \bar{p} \gamma_i = H_i / T = \gamma_i \), which leads to \( \bar{y}_i = (\bar{p} \bar{h}_i \bar{x}_i) / (1 - p_{\text{SS}}) \). Therefore, the expected server bandwidth consumption for this HD channel \( i \) is \( \overline{SBC}_i = (D_i - S_i)^+ = (\bar{p}_i (r_{\text{H}} - \bar{u}) - \bar{y}_i (r_{\text{S}} - \bar{u}))^+ = (\bar{p}_i (r_{\text{H}} - \bar{u}) - (p_{\text{SS}} \bar{p} \bar{h}_i (\bar{u} - r_{\text{S}})) / \bar{p}_{\text{SS}})^+ = (N_{\text{H}} (r_{\text{H}} - \bar{u}) - N_{\text{S}} (\bar{u} - r_{\text{S}}))^+ (\bar{x}_i / \bar{p}_{\text{SS}}) \).

**Remarks:** Theorem 1 indicates that if a PA-VoD system is sufficiently stable, no complicated caching strategy is required. Each peer can simply keep the HD content it watched before until it starts to watch another HD movie, without any global information or coordination. This property best fits the distributed nature of a P2P system. An intuitive explanation is that the system is “self-balancing”. After peers finish an HD movie, they serve as helpers for it. Thus, if there are more viewers in an HD channel, there will be more helpers that watched it before in the steady state. However, as the system is not always stable enough (especially with small populations), the actual instantaneous SBC will fluctuate around the expectation. The passive caching with stationary scenarios are comprehensively studied through simulation in our prior work [10].

### C. Optimal Caching in Non-Stationary Scenarios

In a non-stationary scenario, the arrival rate of viewers will change over time. Therefore we adopt the average server bandwidth consumption \( \overline{SBC}_{t_s \rightarrow t_e} \) over a monitoring period from \( t_s \) to \( t_e \). \( t_s \) and \( t_e \) can be defined by service providers for different purposes. If the long-term performance is of the most concern, a longer monitoring period \( t_e - t_s \) can be selected, otherwise a shorter duration is preferred. Moreover, in the case a new movie is released, \( t_s \) can be set to a time prior to the release time of the movie while the content is time locked [6]. Such a “pre-release” active caching can reduce the tremendous instantaneous server bandwidth consumption during the first several hours after the movie is released, when users start to watch the movie in flash crowds. This is especially useful when the server capacity is limited, i.e., with a small \( u_{\text{max}} \).

We discretize the time into small time slots or rounds. The duration of each round is \( \Delta t = T / R \), where \( R \) is the total number of segments of an HD movie. We set \( t_e - t_s = KR \Delta t \), i.e., \( K \times R \) rounds for the monitoring period. The numbers of the viewers and helpers during the \( k \)-th round after the start of the monitoring period are denoted as \( x_i(k) \) and \( y_i(k) \), respectively. Let \( \lambda_i(k) \) denote the number of user arrivals, and \( \theta_i(k) \) the departures at the end of (or during) round \( k \). Also let \( \delta_i(k) \) denote the number of viewers that turn into helpers, \( \omega_i(k) \) the number of new helpers introduced by the server through active caching, \( \eta_i(k) \) the number of all arrivals, and \( \gamma_i(k) \) the number of departures of the helper queue, respectively. For the purpose of analysis, we assume the arrival rate \( \lambda_i(k) \) at any round \( k \) is known during the monitoring period, and aim at evaluating the system performance with a series of \( \omega_i(k) \). In real systems, only the arrival rate in the past is known. To predict the arrival rate in the near future, control theory and learning techniques can be used [21]. However, they are out of the scope of this paper.

According to Little’s Law for non-homogeneous arrivals,

\[
x_i(k) = \sum_{j=1}^{R} \lambda_i(k - j), \quad y_i(k) = \sum_{j=1}^{R} \eta_i(k - j). \tag{1}
\]

As we assume the local cache of each peer can accommodate one complete copy of an HD movie (R segments), \( k - R \) represents the earliest round in which the arrived peers still reside in the view/helper queue in round \( k \). As a viewer/helper will stay in the viewer/helper queue for time \( T \) or \( R \) rounds and then leave the queue (a helper may return to the help queue with probability \( p_{\text{help} \rightarrow \text{help}} \) after “leaving”), the departure rate in round \( k \) is equal to the arrival rate \( R \) rounds before for both queues, so the following equations hold:

\[
\delta_i(k) = p_0 \theta_i(k) = p_0 \lambda_i(k - R), \tag{2}
\]

\[
\gamma_i(k) = \eta_i(k - R) = \delta_i(k - R) + \omega_i(k - R) + p_1 \gamma_i(k - R), \tag{3}
\]

where \( p_0 = (1 - p_{\text{quit}}) \bar{p}_{\text{HS}} \), and \( p_1 = (1 - p_{\text{quit}}) \bar{p}_{\text{SS}} \). At the start time \( t_s \), i.e., \( k = 1 \), there are already viewers and helpers residing in the queue network. They arrived at the system before the monitoring time, and the corresponding arrival rates also affect the number of viewers and helpers in the future. Therefore, we allow the round number \( k \leq 0 \), and note that \( \lambda_i(k), \delta_i(k) \), and \( \gamma_i(k) \) for any \( k \leq 0 \) are known. The arrival rate of the helper queue at round \( k \) can be derived as

\[
\eta_i(k) = \delta_i(k) + \omega_i(k) + p_1 \gamma_i(k) = p_1^{l+1} \gamma_i(k - lR) + p_1^l \delta_i(k - lR)
\]

\[
+ \sum_{j=1}^{l} p_1^{l-j} \omega_i(k - (l - j)R)
\]

\[
+ p_0 \sum_{j=1}^{l} p_1^{l-j} \lambda_i(k - (l - j + 1)R), \tag{4}
\]

where \( l = \lfloor (k - 1) / R \rfloor + 1 \) represents the batch number of time rounds, if we treat every \( R \) rounds from the start of the monitoring period as one batch.

The server bandwidth consumption at round \( k \) contains two components. First, if the total bandwidth deficit of the viewers exceeds the bandwidth surplus of the helpers, the server has to compensate for the difference, which we refer...
to as the compensation cost for these viewers. Moreover, in order to introduce new active cachers, the server has to upload specific content to the cachers in advance, which also consumes its upload bandwidth. We refer to the latter as the caching cost for the active caching. Assuming that each active cacher downloads the HD content of size $f$, the caching cost is calculated as $\omega_i(k)f/\Delta t$ during round $k$. Therefore, the server bandwidth consumption for channel $i$ at round $k$ is $SBC_i(k) = \left(\tilde{d}_i^{\text{view}}x_i(k) - \tilde{s}_i^{\text{help}}y_i(k)\right)^+ + \omega_i(k)f/\Delta t$, and thus the average SBC for channel $i$ over the monitoring period $t_a \rightarrow t_e$ can be calculated as follows:

$$
SBC_{i,t_a \rightarrow t_e} = \frac{1}{KR} \sum_{k=1}^{KR} \left(\tilde{d}_i^{\text{view}}x_i(k) - \tilde{s}_i^{\text{help}}y_i(k)\right)^+ + \omega_i(k)f/\Delta t
$$

$$
= \frac{1}{KR} \sum_{k=1}^{KR} \left(\tilde{d}_i^{\text{view}} \sum_{j=1}^{R} \lambda_i(k-j) - \tilde{s}_i^{\text{help}} \sum_{j=1}^{R} \eta_i(k-j)\right)^+ + \frac{KR}{KR\Delta t} \sum_{k=1}^{KR} \omega_i(k)f
$$

where $\tilde{d}_i^{\text{view}} = r_H - \bar{u}$, $\tilde{s}_i^{\text{help}} = \bar{u} - r_S$, and $\eta_i(k-j)$ can be obtained from Eqn. (4). Now we propose Theorem 2.

**Theorem 2.** Under non-stationary scenarios, during a period from $t_a$ to $t_e$, given the arrival rate of viewers $\lambda_i(k)$ and the number of active cachers $\omega_i(k)$ injected for an HD channel $i$ at each round $k$, the average server bandwidth consumption for channel $i$ over this period is bounded by Eqn (5).

Remarks: Equation (5) represents the lowest cost to run the server from time $t_a$ to $t_e$, if the caching strategy, i.e., $\omega_i(k)$ for any $k$, is given. When $\omega_i = 0$, it represents a special case as with passive caching only. Moreover, the equation quantitatively characterizes the tradeoff between injecting new content to active cachers and compensating the bandwidth deficit of the current viewers. A larger $\omega_i$ at a given time will lead to more helpers (a larger $y_i$) in the future and thus can reduce the compensation cost. However, it also results in a larger caching cost in terms of $\omega_i(k)f/\Delta t$. Thus we resort to optimization techniques to achieve the desired balance.

The objective of the optimization for a single HD channel $i$ is to minimize the average SBC from time $t_a$ to $t_e$. One constraint we consider is that the instantaneous server bandwidth consumption $SBC_i(k)$ at round $k$ must be bounded by the maximum bandwidth $u_i^{\text{max}}$ allocated to this channel by the server, where $\sum_{i \in \text{HD}} u_i^{\text{max}} = u^{\text{max}}$, $\omega_i(k)$ are the unknowns. Thus, the optimization problem can be formulated as

$$
\min \sum_{k=1}^{KR} \left(\tilde{d}_i^{\text{view}}x_i(k) - \tilde{s}_i^{\text{help}}y_i(k)\right)^+ + \omega_i(k)f/\Delta t
$$

$$\text{st: } \forall k, \left(\tilde{d}_i^{\text{view}}x_i(k) - \tilde{s}_i^{\text{help}}y_i(k)\right)^+ + \omega_i(k)f/\Delta t \leq u_i^{\text{max}};$$

$$\forall k, \omega_i(k) \in Z_0^+$$

However, the objective function above exhibits nonlinearity, which is difficult to solve. To make it tractable, we introduce a variable vector $u_i^f$ containing the compensation cost of the server for channel $i$ at each round, and convert the optimization problem to a mixed integer linear programming (MILP) problem. To achieve it, we add a constraint that the aggregated bandwidth supply from helpers and the server is no less than the total bandwidth deficit of these viewers at any time, i.e., $u_i^f(k) + \tilde{s}_i^{\text{help}}y_i(k) - \tilde{d}_i^{\text{view}}x_i(k) \geq 0$ for $1 \leq k \leq KR$. It is a necessary condition for all the viewers to watch the movie fluently. The optimization problem then becomes

$$
\min \sum_{k=1}^{KR} (u_i^f(k) + \omega_i(k)f/\Delta t)
$$

$$\text{st: } \forall k, u_i^f(k) + \tilde{s}_i^{\text{help}}y_i(k) - \tilde{d}_i^{\text{view}}x_i(k) \geq 0$$

$$\forall k, u_i^f(k) + \omega_i(k)f/\Delta t \leq u_i^{\text{max}};$$

$$\forall k, \omega_i(k) \in Z_0^+$$

where both $u_i^f(k)$ and $\omega_i(k)$ are unknown variables. The optimization becomes a MILP problem; although still NP-hard, many methods in the literature can generate “suboptimal” solutions in polynomial time. These suboptimal solutions can also provide insights and guidelines for the design of an active caching strategy. Moreover, the optimization is also applicable to the cases with multiple active-caching channels.

V. BANDWIDTH ALLOCATION STRATEGIES

The optimal caching strategies we obtained in the previous section assume that a helper is always able to help viewers. However, this is not true as such an HD movie cached on helpers is often incomplete due to the space limit. For instance, if a helper has Segment 10 to 20 of an HD movie, and an HD viewer is watching Segment 1, the helper cannot allocate its bandwidth to the viewer due to the content bottleneck. Therefore, we adopt a combination of the following two bandwidth allocation strategies: chain-based allocation and substream multiplexing. Our objective is to reduce the chance of content bottleneck as much as possible, which leads to a better utilization of the peer upload bandwidth.

1) **Chain-based Allocation**: Chain-based allocation, also referred to as “stratification” [6], [11], [14], [16], takes advantage of the temporal relationship between the peers within the same channel. Peers that arrived early in the system have cached a larger number of video segments already, and thus have the content that is of interest to late peers. Therefore, a simple approach is to treat the viewers within a channel as a chain, where a viewer streams from other viewers ahead of it. We extend the chain-based allocation of a single channel to support multi-channel systems with HD movies. The details can be found in our prior work [10].

Our chain-based allocation is composed of two parts: inner-chain and inter-chain allocation, where the former allocates the upload bandwidth of the viewers within a channel following the traditional chain-based allocation, and the latter manages the allocation between the viewers and their helpers. The allocation process includes the following three steps.

**Step 1:** SD channels perform the inner-chain allocation, in which each peer reserves a certain bandwidth $\bar{u} - r_S$ for helping
HD channels. In an SD channel \(i\), all the viewers are sorted based on their arrival time, i.e., forming a channel chain. The allocation bandwidth will be allocated to the nearest viewers following it until their bandwidth demands are all satisfied, or its remaining bandwidth is less than or equal to \(\bar{u} - r_S\). Then the following viewers start to allocate their bandwidth in the same manner until all the viewers have allocated their bandwidth or no more bandwidth is needed in the channel.

Step 2: The inter-chain allocation is invoked between an HD channel and its helpers. Recall that with passive caching, late HD segments are more likely to be cached, which are of the interests of viewers that arrive early in the system. We allocate such help bandwidth to early viewers first. The purpose is to facilitate the bandwidth allocation within that HD channel. If the bandwidth of the helper is allocated to late HD viewers first, the bandwidth of those late viewers will be never helpful for early viewers. Also, the helpers with fewer HD segments can only support a small set of HD viewers, and thus will select target viewers with priority.

Step 3: At last, the HD viewers perform an inner-chain allocation within the HD channel without reserving any upload bandwidth. After that, if there are still unsatisfied viewers, the server will compensate for the bandwidth deficit.

Remarks: The inter and inner-chain allocation aim at making the bandwidth “transferable” between peers, from SD to HD channels through the cooperation of viewers and helpers. It utilizes the cached content of the passive cachers.

2) Substream Multiplexing: Although the chain-based allocation is demonstrated to be efficient in stationary scenarios with passive caching in our prior work [10], it does not apply to non-stationary scenarios with active caching. For active cachers, to fetch a small size \(f\) of the HD content is preferred, as to reduce the caching cost \(f/\Delta t\). On the other hand, \(f\) indicates the portion of the HD movie to cache, where a larger \(f\) enables the helper to help a wider range of HD viewers. To address this concern, we adopt “substream multiplexing” to achieve the balance between these two kinds of cost, which originates from VUD for P2P live-streaming systems [7].

When introducing active cachers, we divide them into \(N_{\text{sub}}\) substreams. Each segment of the HD movie is further divided into smaller chunks, and the \(i\)-th substream is responsible for the distribution of the \(i\)-th chunk of every \(N_{\text{sub}}\) chunks in the HD movie. Figure 3 illustrates an example of substream multiplexing of active HD helpers. There are 20 substreams. Helper 1, Helper 2, etc form Substream 1, in which they only download and distribute the first chunk of every 20 chunks. Substream 2 is composed of some other helpers and distributes different chunks. Consequently, the caching cost \(f/\Delta t\) is reduced dramatically as each helper only needs to cache a small portion of the HD content, i.e., of size \(f = F/N_{\text{sub}}\), where \(F\) is the byte length of an HD movie. For the above case, \(N_{\text{sub}} = 20\), an active helper only downloads \(1/20\) of the entire movie. The bandwidth cost to achieve that in one round is \(F/20\Delta t = 1,000\) Kbps, which does not exceed the typical downlink bandwidth of broadband Internet access links. After the content is cached, the helper stays in the system, and continuously contributes the actively cached content with its available bandwidth until leaving the helper queue. Therefore, the initial short-term cost during the caching round brings a higher accumulating benefit in the near future. This underlies the efficacy of our active caching strategies.

VI. PERFORMANCE EVALUATION

In this section, we evaluate our caching strategies obtained from the optimization problem. We first follow a methodology to find the desired active caching strategy (i.e., \(\omega_i\)) using the TomLab optimization toolbox in MatLab, and then use a Java-based event-driven simulator to verify its efficacy.

A. System Setup

The peer cache size or the byte length of an HD movie is \(F = 1\) GB. An SD (HD) movie takes 512 (1,024) MB. They both last for 20 rounds in time. The upload bandwidth of peers follows the distribution listed in Table II, with an average of 600 Kbps. The playback rate of the SD (HD) movies are 500 (1,000) Kbps. Thus, on average one HD viewer has a bandwidth deficit of 400 Kbps, and four helpers (each has 100 Kbps bandwidth surplus) are needed to help one viewer. We focus on the scenario that one new HD movie \(i\) is released, and peers cram into the channel in a flash-crowd manner.

We set the monitoring period to include three stages: the pre-release stage, the flash-crowd stage and the steady stage. The pre-release stage refers to the time rounds before the new movie is released, when the arrival rate of viewers is 0. Viewers start to watch the movie in the flash-crowd stage following different arrival patterns. After the flash-crowd stage, we assume the arrival rate becomes a fixed constant, which is equal to the last round of the flash-crowd stage, and the system becomes stable afterward, i.e., the steady stage. The length of the first and third stages can be customized by service providers, but the flash-crowd stage is determined by users behaviors. Similar to [17], three arrival patterns during the flash-crowd stage are considered:
• Low intensity: the arrival rates follow an exponentially decreasing function, i.e., \( \lambda_i(k) = \lambda_i(1)e^{-\frac{k}{\tau_1}}, \) where \( \tau = 20 \) and \( \lambda(1) = 20. \) The number of arrivals \( \lambda_i(k) \) are rounded up to the smallest integers at each round \( k. \)
• Medium intensity: the arrival rate at round \( k \) is \( \lambda_i(k) = k, \) i.e., a linearly increasing function.
• High intensity: the arrival rate at any round is 20.

We set \( p_{SS} = p_{HS} = 0.8, \) which indicates that SD viewers account for 80\% of the total population, i.e., four helpers for one HD viewer on average, close to the observed ratio in [4]. This can be achieved through recommendation mechanisms. For example, after a user watched an HD movie, the system can recommend some related SD programs, e.g., the discussions, interviews, and talk shows about the movie, which have lower requirements in terms of the playback rate. The effect of such recommendation is verified in [22].

B. Obtaining the Optimal Caching Strategies

Under the system settings and the scenarios described above, we now present our methodology of obtaining the optimal caching strategies. Here we use the terminology “optimal” (instead of suboptimal) as many solutions we obtained are reported optimal by TomLab. We first investigate the system evolution under the three arrival patterns, and then decide proper configuration parameters, such as the length of the monitoring period, and the minimum server capacity that guarantee our optimal solution.

1) System Evolution: We set \( f = F/20, \) and relax \( u_{\text{max}} \) to \( \infty, \) which means the caching cost of an active cacher is quite small, and the server has unlimited capacity. Let \( \rho^{\text{quit}} = 0.2 \) for the medium-intensity case, and \( \rho^{\text{quit}} = 0 \) for the other two cases (i.e., no peer leaves). We also set the pre-release stage to 20 rounds, and the overall monitoring period to 80 rounds, i.e., \( KR = 80. \) We solve the optimization using TomLab and plot in Fig. 4 the evolution of the system, showing the expected number of viewers, helpers in active and passive cases and \( \omega_i \) (i.e., the number of new active cachers introduced) in each round in the channel under consideration.

We first observe the “mis-alignment” of the viewers and helpers with passive caching (denoted by \# Helpers, passive), with a duration gap greater than \( T = 20 \) rounds. This result is consistent with [16]. The number of helpers increases slowly after the new movie is released, which fails to provide sufficient bandwidth support for the viewers. Note that one viewer needs four helpers in terms of bandwidth supply. This is because the helper population starts to increase only after the first viewer finishes watching the HD movie and transits to SD channels (from round 41). In contrast, through active caching, the helpers are actively introduced by the server, which quickly catches up with the fast bandwidth demand increase of the viewers. Second, active caching occurs during a short period from round 21 to round 40 or 60 in the cases \( \rho^{\text{quit}} = 0 \) for the low or high intensity. This indicates: (1) the monitoring period does not need to be very long; (2) with unlimited server capacity, no pre-release caching is needed, as we can see \( \omega_i = 0 \) before the flash-crowd stage. Active caching starts right after the new movie is released. However, for the case \( \rho^{\text{quit}} = 0.2, \) peers may leave the system, resulting in an insufficient number of helpers. Thus a periodic active caching should be invoked, as shown in Fig. 4b. At last, from Fig. 4b and Fig. 4c, we may conjecture that for non-decreasing arrivals, a simple active caching strategy is to measure the number of helpers \( N_{\text{help}} \) and viewers \( N_{\text{view}} \) at a time instant, and inject the difference as \( 4N_{\text{view}} - N_{\text{help}} \) active cachers. This does not require the prediction of future arrivals.

2) Monitoring Period: We then investigate the desired length of the monitoring period when \( \rho^{\text{quit}} = 0. \) To reduce the cost of the server for such monitoring, a shorter period is better. However, the length of the monitoring period cannot be too short, either. Otherwise the optimization places more emphasis on the short-term benefits. For example, if the viewers are lack of 100 Kbps bandwidth in a particular round, we can either let the server compensate the bandwidth deficit at a cost of 100 Kbps, or introduce a new active cacher who has 100 Kbps extra bandwidth. The caching cost here is 1,000 Kbps. Obviously, for this round only, active caching is not optimal. However, once a new active cacher is introduced, it will keep helping the viewers for at least 20 rounds, with an aggregate bandwidth contribution of more than \( 100 \times 20 \) Kbps. As a result, the monitoring period needs to cover at least 20 rounds in order to reflect the long-term contribution of this active cacher. To this end, we vary the length of the monitoring period during the steady stage and plot the aggregate server bandwidth consumption, i.e., \( \sum_{k=1}^{KR} SBC_i(k). \) In addition, we keep 20 pre-release rounds. Figure 5a shows that such aggregate costs are increasing functions of the monitoring period until reaching their maximum. The cost below the maximum value (at length 40 or 60) is the short-term cost, as the monitoring period is too short to capture the long-term benefit, similar to the example we explained above. To conclude, the best choice in these cases is to monitor 60 more rounds after the movie is released, equivalent to 7 hours.

3) Cached Content: To explore the tradeoff between the compensation and caching cost, we vary the size of the actively cached content from \( f = F/20 \) to \( f = F/5, \) which we consider to be the maximum possible caching cost in real systems. The monitoring period is set to 80 rounds as suggested above (with 20 pre-release rounds). We plot in Fig. 5b the average server bandwidth \( SBC_{t_i,t_i+t_{r_i}} \) and the average compensation cost \( \text{mean}(u_i^c) \) allocated by the server through solving the optimization problem. With the increasing cost \( f \) of injecting active cachers, the compensation by the server rises only with the low-intensity arrival pattern. This is because from round 41, the number of viewers starts to decrease, which reduces the long-term bandwidth demand in the future. Thus, compensating the current deficit is preferred. With the other two arrival patterns, the number of viewers is always increasing, and so is the bandwidth demand. Thus the long-term benefit in the future still overwhelms the caching cost at present, and the server will always introduce new active cachers. i.e., no compensation at all.
consumption actually leads to a lower overall cost to run the server. The first star for the high-intensity pattern reaches zero because the optimization problem is MILP infeasible in this case, i.e., with \( u_{\text{max}}^i \) and \( \varepsilon \) for the maximum server bandwidth consumption per round from the optimization as \( 8 \times 10^4 \) Kbps for the three arrival patterns. Then we place constraints on the maximum server bandwidth as \( 8 \times 10^4 \) Kbps, which means that such a server can only inject a limited number of active cachers, the optimization also makes peers “pre-cache” the HD content before the movie is released, as illustrated in Fig. 6. However, some active cachers introduced during the pre-release stage may leave the channel system during the flash-crowd stage. The loss of the bandwidth of such active cachers results in a higher cost to run the server. The cost becomes a constant if \( u_{\text{max}}^i \geq 8 \times 10^4 \) Kbps, which means that such a \( u_{\text{max}}^i \) is the minimum instantaneous server bandwidth capacity that leads to the optimal active caching of this HD channel.

Moreover, although pre-release caching results in a higher average server bandwidth consumption over the period, on the other hand, it mitigates the high instantaneous bandwidth consumption at the server side. Thus the optimal choice depends on how VoD service providers pay for their server.
bandwidth consumption. Such pre-release caching strategies are extremely useful with a limited server capacity, which is also studied through simulations in [6].

C. Model Validation

To further validate the performance of our active caching strategies, we developed a Java-based event-driven simulator to emulate a multi-channel PA-VoD system. There are 10 HD channels (Channel 0 to 9) and 10 SD channels (Channel 10 to 19) at the beginning. Viewers start to join these channels following the popularity of movies, a Zipf distribution with parameter 1 for each category. The transition probabilities between categories are \( p_{SS} = 0.8 \) and \( p_{HS} = 0.8 \). Peers make decisions independently and locally when transitioning to other channels. They first decide which category to transit according to \( p_{SS} \) and \( p_{HS} \), and then select channels in the targeted category following the movie popularity. We set a 24-hour ramp-up stage to allow peers to replenish their local cache through passive caching. After 24 hours, we release a new HD movie in the system, i.e., Channel 20. A number of peers in the system are selected in each round to start watching Channel 20, forming the flash-crowd patterns aforementioned.

We measure the bandwidth consumption of each channel and select peers from the helpers of those over-provisioned HD channels. We then assign these peers to actively fetch \( f = F/20 \) of the new HD movie and help that channel. The number of such helpers follows the optimal \( \omega_i \) for each round we obtained from the optimization problem. After that, we first perform the inner and inter-chain allocation, and then use a group of active cachers to support the unsatisfied viewers through substream multiplexing. When adding helpers to substreams, we select the substream with the least bandwidth supply, i.e., a simple water-leveling approach.

We validate two things through our simulation: first, the evolution of the number of viewers \( x_i(k) \) and helpers \( y_i(k) \) derived using Eqn. (1), (2), (3) and (4); second, the efficiency of our bandwidth allocation approaches, i.e., inner and inter-chain allocation for passive cachers, and substream multiplexing for active cachers. Fig. 6b shows the simulation and analytical results under the low-intensity pattern with \( w^{\text{max}}_i = 3 \times 10^4 \) Kbps, while Fig. 6c demonstrates the server bandwidth consumption in the medium and high intensity patterns with unlimited server capacity. We also plot the server bandwidth consumption with passive caching for the medium intensity pattern, which increases dramatically after the new movie is released at Round 21. With our optimal active caching strategies, the server bandwidth consumption quickly decreases to zero, due to the sufficient bandwidth supply of the active helpers. These two figures illustrate that the simulation results agree with our analysis in all cases.

VII. CONCLUSIONS AND FUTURE WORK

This paper focused on the caching strategies in multi-channel PA-VoD systems with HD video channels. We proposed a modeling framework, which can capture the essential characteristic of a PA-VoD system, i.e., the bandwidth demand and supply of peers. The framework can be easily extended to reflect different user behaviors and evaluate a variety of caching strategies. We customized the framework to model a multi-channel PA-VoD system in detail and derived the server bandwidth consumption in both stationary and non-stationary scenarios, which leads to the optimal caching strategies for both cases. Our simulation verified the efficacy of our modeling framework and the active caching strategy.

Our future work will involve trace-driven simulation, and real-system implementation of the cache and bandwidth allocation algorithms. Moreover, as our scheme requires SD peers to contribute to HD peers, a proper incentive mechanism will be designed, modeled and evaluated.

Acknowledgment: The work is supported in part by NSERC of Canada, and Le Chang is also supported in part by a fellowship from China Scholarship Council.

REFERENCES