Serving Niche Video-on-Demand Content in a P2P Environment

E. Brosh, V. Misra, D. Rubenstein

Department of Computer Science, Columbia University, New York, US

{elibrosh,misra,danr}@cs.columbia.edu

Abstract: Current VoD services focus on providing access to popular content. Yet, the vast majority of content is heavy-tailed, subject to relatively low-demand, and is poorly supported by current approaches. In this paper, we study P2P VoD systems that include support for niche videos by reserving small portions of peers' storage and upload resources for such content. We demonstrate through analysis and simulation that simple weighted caching techniques can provide high streaming quality and low startup delay for niche videos.

Keywords: Video-on-demand, caching, stochastic modeling

1 Introduction

Video-on-demand (VoD) is an online service that allows users to view videos from a catalog of available choices at any time. Existing peer-to-peer (P2P) based VoD services focus on providing access to popular content. However, they are limited in their ability to support efficient streamed access to *niche content* that has relatively small demand. This limitation stems from the poor performance of P2P mechanisms when the number of peers sharing content is small [2]. P2P VoD providers can leverage known techniques to obtain wide-availability of niche content among peers. E.g., content can be explicitly pushed to the peers during off-peak hours [6], or implicitly by combining videos with diverse popularity into pushed bundles [4]. However, how to best serve that niche content is not yet well-understood.

Consequently, we study a system that extends the design of existing P2P VoD systems geared towards popular content by reserving a small portion of peer's storage and bandwidth resources for the purpose of serving niche content. We then address the following question: what may be the best and most practical strategies for content replication to enable efficient delivery of niche content? Our analysis reveals that when delivering a niche video, earlier parts of the video suffer more greatly from playback continuity gaps than later parts. A common approach to reduce such gaps is to allocate additional cache space for the videos' earlier portions, a technique known as prefix caching [5]. However, due to its storage overhead, applying prefix caching across the long-tail of niche content is too costly. Contrastingly, we demonstrate that the performance in servicing niche videos can be significantly improved without the need for additional storage space by optimizing the placement of pieces within the peers' caches according to a playback-point weighted caching strategy.

To design such strategy, we develop a stochastic model which characterizes the delivery of niche videos in P2P VoD environments (Section 2). Through simulations we show that a weighted caching strategy can reduce the startup delay requirement by a factor of 3 compared to a uniform strategy (Section 2.1). Moreover, in [1] we show that reserving a small portion of peers' resources to niche videos can significantly improve the system-wide playback performance.

2 Niche Content Delivery Model

We assume that each video is chopped into pieces. The pieces are distributed to the peers, who are then able to exchange them on demand. The set of peers exchanging video pieces is called a swarm. Our goal is to determine how to distribute the pieces of a niche video across peers' caches so as to optimize its playback performance. To this end, we develop a model that captures the piece-level playback performance of a single client viewing some target video, the common case when the video popularly exhibits a long-tail.

We study the system within the context of rounds. The duration of a round corresponds to the time taken to play a single piece. Each peer reserves a small portion of its upload bandwidth and storage resources for niche content delivery. To capture the peer's upload bandwidth sharing among popular and niche videos, we assume that in each round, when a peer is issued an upload request for niche video, it is *available* to serve that request with a probability $P_a < 1$.

An arriving client may buffer data for d rounds before commencing playback. In each round, the client requests the pieces closest to playback deadline (also called urgent pieces) from all the available peers that have those pieces in their cache. A peer is able to respond only to one request and selects the most urgent one. The unsatisfied requests are canceled and reissued again in the next round. The client departs the system when video playback is completed.

Consider a swarm for a video with m pieces. The swarm has a single downloading peer and N seeding peers (seeds) who are also participating in other swarms. We assume that when a piece is missing, playback stops until the piece is available. We use w_i to denote the time taken to complete the playback of piece i. This time consists of the time playback is stalled plus one round, the piece's own playback time. Suppose that the peer does not have piece i when it is first needed for playback. Then, playback is stalled until at least one seed with piece i becomes available. If r_i seeds have piece i in their cache, then the probability that at least one seed with piece i is available, denoted by $F(i)$, is $1-(1-P_a)^{r_i}$, and the time playback is stalled, $w_i - 1$, follows a geometric distribution with parameter $F(i)$.

As all peers with most urgent piece can be unavailable, a peer can download some non-urgent piece j, $j > i$, while waiting to obtain piece i. To capture out-of-order downloads, we use $p_s(i)$ to denote the probability that the downloader has piece i when piece s is first needed for playback. The average time taken to complete the playback of piece i, $E[w_i]$, can thus be expressed as:

$$E[w_i] = \sum_{k=1}^{2m} kPr\{w_i = k\} = 1 + \frac{1-p_i(i)}{F(i)} \qquad (1)$$

$$F(i) = 1-(1-P_a)^{r_i}$$

$$Pr\{w_i = k\} = \begin{cases} p_i(i) & \text{if } k=1 \\ (1-p_i(i))(1-F(i))^{k-2}F(i) & \text{otherwise} \end{cases}$$

The value of $p_i(i)$ can be computed recursively:

Proposition 2.1 *The probability a downloader has acquired piece i when piece $s > 0$ is needed for playback, $p_s(i)$, is given by:*

$$p_s(i) = \sum_{k=1}^{2m} Pr\{w_{s-1} = k\} p_{s-1}(i,k) \qquad (2)$$

$$p_s(i,k) = 1-(1-p_s(i))\prod_{j=1}^{k}(1-F(i)S_s(i,j))$$

$$S_s(i,k) = \prod_{j=s}^{i-1} 1 - \frac{F(j)}{p_a N}(1-p_s(j,k-1))$$

where $p_0(i) = 1-\prod_{j=0}^{d-1}(1-F(i)S_0(i,j))$, and $F(i)$ and $Pr\{w_{s-1} = k\}$ are defined in Eq. (1).

Proof Refer to [1].

The time taken to playback the video $E[T]$ is thus:

$$E[T] = d + \sum_{i=0}^{m-1} E[w_i] = d + m + \sum_{i=0}^{m-1} \frac{1-p_i(i)}{1-(1-P_a)^{r_i}}$$

where the last term in the equation above represents the total amount of time playback is stalled.

2.1 Weighted Caching

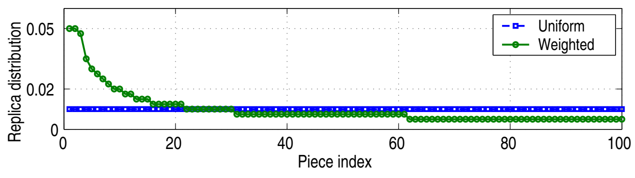
Using the model, we show in [1] that early pieces are highly sensitive to instantaneous fluctuations in seeds' service capacity availability; With a uniform cache distribution (i.e., $r_i = r_j$, for all i, j), earlier pieces are more likely to miss their playback deadlines than later pieces. A crucial issue is then to determine the right strategy of placing video pieces within the seeds caches. Our goal is to determine the number of replicas of piece i, r_i, so as to optimize the playback performance while respecting a storage capacity constraint nm (where n is the number of replicas of the video across the seeds caches) and a replication constraint $1 \le r_i \le N$.

$$\text{Minimize } \; E[T] \qquad (3)$$

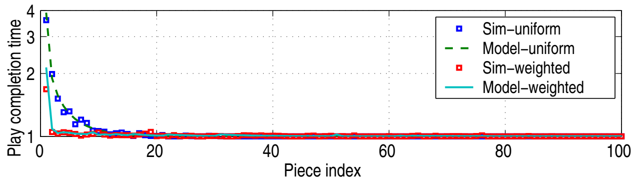
$$\text{subject to } \; \sum_{i=0}^{m-1} r_i = nm, \qquad 1 \le r_i \le N.$$

The optimal solution can be obtained numerically and typically yields a *front-weighted* cache distribution. For example, Figure 1(a) depicts the optimal piece replica distribution for a 100-piece video, $n = 4$, $N = 20$, $d = 0$, and $P_a = 0.1$. The optimal distribution replicates earlier pieces more heavily at the expense of later pieces to improve playback performance.

Figure 1(b) shows the predicted vs. measured time taken to complete the playback of each piece relative to the piece's duration for the two caching strategies, derived using simulations (see [1]). As shown, the weighted caching strategy reduces the time taken to complete the playback of the first 10 and 50 pieces of the video (by 30% and 20%, respectively), as well as the total time playback is stalled (by 75%). Using simulations, we observe that a weighted cache distribution can reduce the startup delay requirement by an average factor of 3.5 compared to a uniform design by making better use of the storage allocated for the video.



(a) Uniform vs. weighted cache distributions



(b) Performance of uniform and weighted distributions

Figure 1: Comparison of uniform and weighted caching.

References

[1] Brosh *et al.* Serving niche video-on-demand content in a managed p2p environment. Technical Report cucs-031-09, Columbia University, 2009.

[2] Dn *et al.* Dynamic swarm management for improved bittorrent performance. In *IPTPS*, Apr 2009.

[3] Huang *et al.* Challenges, design and analysis of a large-scale p2p-vod system. In *SIGCOMM*, Aug 2008.

[4] Menasche *et al.* Modeling content availability in peer-to-peer swarming systems. In *CoNext*, Dec 2009.

[5] Sen *et al.* Proxy prefix caching for multimedia streams. In *INFOCOM*, Mar 1999.

[6] Suh *et al.* Push-to-peer video-on-demand system: Design and evaluation. In *JSAC*, 25(9):1706–1716, 2007.