QoE-Driven Cache Management for HTTP Adaptive Bit Rate Streaming Over Wireless Networks

Weiwen Zhang, Yonggang Wen, *Member, IEEE*, Zhenzhong Chen, *Member, IEEE*, and Ashish Khisti, *Member, IEEE*

***Abstract*—In this paper, we investigate the problem of optimal content cache management for HTTP adaptive bit rate (ABR) streaming over wireless networks. Specifically, in the media cloud, each content is transcoded into a set of media files with diverse playback rates, and appropriate files will be dynamically chosen in response to channel conditions and screen forms. Our design objective is to maximize the quality of experience (QoE) of an individual content for the end users, under a limited storage budget. Deriving a logarithmic QoE model from our experimental results, we formulate the individual content cache management for HTTP ABR streaming over wireless network as a constrained convex optimization problem. We adopt a two-step process to solve the snapshot problem. First, using the Lagrange multiplier method, we obtain the numerical solution of the set of playback rates for a fixed number of cache copies and characterize the optimal solution analytically. Our investigation reveals a fundamental phase change in the optimal solution as the number of cached files increases. Second, we develop three alternative search algorithms to find the optimal number of cached files, and compare their scalability under average and worst complexity metrics. Our numerical results suggest that, under optimal cache schemes, the maximum QoE measurement, i.e., mean-opinion-score (MOS), is a concave function of the allowable storage size. Our cache management can provide high expected QoE with low complexity, shedding light on the design of HTTP ABR streaming services over wireless networks.**

***Index Terms*—Adaptive bit rate streaming, content cache management, optimization, quality of experience.**

I. Introduction

MOBILE video, owing to the rapid adoption of smartphones, is fueling a dramatic growth of mobile data traffic lately. Cisco VNI report [1] predicted that the mobile data traffic will increase 26 times between 2010 and 2015, among which the leading contributor is the video traffic generated by the mobile users worldwide. This growth of mobile video, however, is in tandem with a huge concern of the user experience, resulted from the inherent nature of stochastic wireless channels (e.g., multi-path fading and shadowing effects). As a result, service providers should aim to provide a high Quality of Experience (QoE) for the rising demand of video streaming over wireless networks.

Various techniques have been proposed to address real-time adaptation of multimedia contents over wireless channels for a better QoE (often resulting from a better Quality of Service). For example, rate-reduction transcoding design [2] was proposed as an integral part of wireless video streaming and [3] demonstrated by experiments that rate adaptation transcoding was effective for streaming high quality pre-encoded video. However, the real-time transcoding approach often entails a stringent requirement for computing resources. Recently, HTTP adaptive bit rate (ABR) streaming [4]–[7] has emerged as a main-stream solution to provide smooth playback experience with high quality of videos and improve the network resource utilization. In practical systems (e.g., Cisco's CDS-IS [8]) or the emerging media cloud platform, each content is transcoded into a set of media files with diverse playback rates, and appropriate files will be dynamically chosen in response to channel conditions and outlet forms [9]. The ABR solution is, however, stressed by the huge growth of user-generated contents. It has been observed in Cisco's deployment that transcoding a large set of video contents into files with different playback rates on the streaming engine can have the storage to be filled up rapidly. This observation dictates that either more storage space should be provisioned, resulting a higher cost, or intelligent algorithms should be designed to make the best use of the limited storage budget, providing a just-in-situ QoE requirement.

Taking the latter approach, we investigate the issue of optimal content cache management for HTTP ABR streaming. Previous studies on HTTP ABR streaming mainly focus on mechanisms to adjust the streaming bit rates for varying network conditions. In [5], [6], [9]–[11], various solutions were proposed to improve the performance of the HTTP ABR streaming services, with an objective to optimize the Quality of Service (QoS) (e.g., the reduction of end-to-end delay, better buffer management, bandwidth savings, or higher resource utilization). On a different track, research efforts [12]–[15] have been devoted to investigating QoE-aware adaptation schemes. These solutions aim to maximize content provisioning and network resources under the QoE requirement, or maximize the QoE under the bandwidth constraints. However, neither approach considers the storage aspect of HTTP ABR streaming. In this research, we extend the research scope of HTTP ABR streaming with QoE-driven content cache management.

In this paper, we aim to develop a QoE-optimal scheme for the individual content cache management in HTTP ABR streaming. Our design objective is to provide the best possible QoE for the mobile users, while avoiding the content storage

Manuscript received May 31, 2012; revised August 23, 2012; accepted November 22, 2012. Date of publication February 15, 2013; date of current version September 13, 2013. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Selcuk Candan.

W. Zhang and Y. Wen are with the School of Computer Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798 (e-mail: wzhang9@ntu.edu.sg; ygwen@ntu.edu.sg).

Z. Chen is with MediaTek USA Inc., San Jose, CA 95134 USA (e-mail: zzchen@ieee.org).

A. Khisti is with the School of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 1A1, Canada (e-mail: akhisti@comm.utoronto.ca).

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

Digital Object Identifier 10.1109/TMM.2013.2247583

to be filled up rapidly. Specifically, the content provided by the original server is transformed into HTTP streaming formats and cached on the streaming engine in advance. When a client consumes some content with a required streaming rate, the streaming engine will reply a content file with the closest bit rate to the requested, hoping that the distortion is acceptable by the client. In this case, the research problem is how to choose a set of video files with different playback rates, for a given storage budget, so as to maximize the expected QoE for a pool of mobile users.

We adopt a snapshot approach to the cache management problem in the system involved, which can be extended to the dynamic approach. Ideally, the cache management in streaming system should be dynamical and cache replacement should be in place for good performance. However, the complexity of implementing a dynamic system would be high, as one would consider all possible corner cases. As such, we take a snapshot of the inherent dynamic system and aim to shed some light by solving a snapshot optimization problem. The insight obtained can certainly facilitate to solve the dynamic problem. Indeed, in system development, the software architect normally prefers to solve a series of snapshot problems rather than the unstructured dynamic evolution of the underlined system. The QoE obtained by our cache management is close to the upper bound (i.e., 5), implying that the gap between our cache management and the dynamic cache management is small. Therefore, our cache management can provide good QoE for the users, while its complexity is much less than the dynamic one.

Our contribution in this research is multi-fold, detailed as follows:

- We present a logarithmic QoE model derived from our experimental results (cf. Section V-A), and formulate the content cache management for HTTP ABR streaming into a convex optimization (i.e., snapshot problem), thus giving this engineering problem an analytical framework.
- We apply the Lagrange multiplier method to solve the optimization problem and obtain numerical solution of the set of playback rates for an individual content cached in the stream engine. We characterize the optimal solution analytically and our investigation reveals a fundamental phase change in the optimal solution as the number of cached files increases.
- We provide three alternative search algorithms (i.e., exhaustive search, Dichotomous-based search, and variable step-size search) to find the optimal number of cached files, and compare their scalability under average and worst complexity metrics.
- Our numerical results suggest that under optimal cache schemes, the maximum QoE measurement, i.e., mean-opinion-score (MOS), is a concave function of the allowable storage size.

Our comprehensive investigation reveals insightful guidelines to provide HTTP ABR streaming services over commercially-available platforms, e.g., CDS-IS from Cisco.

The rest of this paper is organized as follows. Section III presents the system model and problem formulation. In Section IV, a mathematical solution is given for the optimal content cache management of the HTTP ABR streaming. Numerical simulations are provided in Section V. Finally, Section VI concludes the paper and suggests future work.

II. Related Work

There has been a line of research work studying the adaptive HTTP streaming [5], [6], [9]–[11], [16]. These works mainly consider the Quality of Service (QoS) for varying network conditions. [16] studied the multimedia-streaming systems and proposed packet scheduling algorithms over in-vehicle QoS-enabled wireless channels. [6] proposed an algorithm for adaptive HTTP streaming that detects bandwidth changes. [9] presented a solution to improve the performance of adaptive HTTP streaming services, with significant bandwidth savings and minimal decrease in quality. [10] conducted an experimental evaluation of three commercial adaptive HTTP streaming players, and investigated on how they react to the available bandwidth variations. [11] analyzed the delay components of adaptive HTTP streaming and considered how to minimize the end-to-end delay for live services. But few of them takes account into QoE.

Another line of research [17]–[20] investigated the QoE measurement and its relationship with QoS. In our work, however, we do not correlate QoE to QoS, but consider the QoE metric as MOS by capturing the user's subjective perception. We present a QoE model based on the distortion between the required bit rate and the actual streaming rate.

Moreover, [14] proposed a QoE adaptation scheme for video applications that maximizes content provisioning and network resources according to user's QoE requirement over resource constrained wireless and mobile networks. [12] developed a QoE-optimized packet scheduler that minimizes the overall distortion under the bandwidth constraints for the wireless links. [15] presented a Sender Bit rate (SBR) adaptation scheme at pre-encoding stage for video applications based on MOS. This scheme was shown to be responsive to available network bandwidth and congestion. [13] investigated the impact of transcoding and packet dropping on the video quality, and provided the optimal resource allocation, which maximizes the mean user satisfaction for network resources.

However, none of these works account for the content caching issue on the media cloud. This research discusses how to utilize the limited cache storage in the media cloud to achieve the optimal QoE. Basically, we focus on the snapshot problem. First, we verify the appropriate QoE function, by evaluating the scores from our experimental QoE study. Then, we formulate the content cache management for HTTP ABR streaming into a convex optimization problem. Following that, the mathematical solution and analysis are provided for the optimal bit rate streaming on the QoE-driven content cache management. In addition, we provide three alternative algorithms on finding the optimal cache management. Finally, the simulation results are in a great accuracy, and the design of bit rate streaming can provide insightful guidelines for the practical content cache management.

III. System Model and Problem Formulation

In this section, we first describe functional architecture for a generic HTTP ABR streaming system, modeled after a real

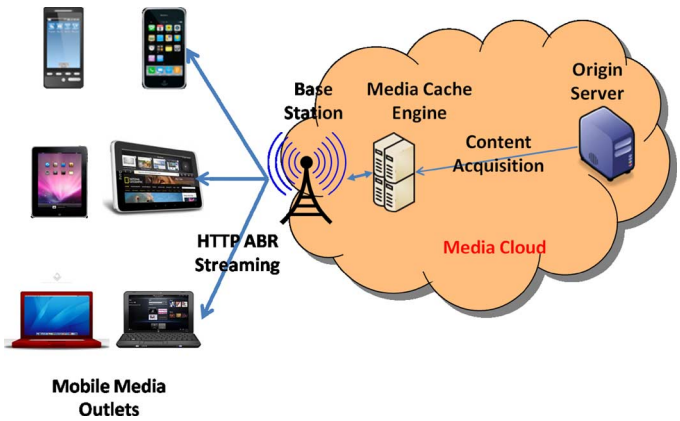


Fig. 1. A schematic diagram for HTTP ABR streaming system over the wireless network: contents acquired from original servers are transcoded into a set of HTTP ABR files with different playback rates, and cached in the streaming engine.

system deployment. Following that, we present various models for the HTTP ABR streaming system, including a user-request model, a QoE model and a content caching model. Using these models, we formulate the content cache management as a constrained optimization model.

A. System Architecture

This subsection presents our proposed ABR media streaming architecture for wireless networks. It is based on real experience with one of the leading vendors in the market.

The generic HTTP ABR streaming system, adapted from a real deployment, is illustrated in Fig. 1. It consists of three parts, including a content origin server, a media streaming engine with caching capability, and a pool of mobile media outlets. The content origin server stores media files in their original formats and transfers them to the intermediate media cache engine. In any media cache engine, the original content file is transformed into HTTP streaming format (e.g., SmoothHD, Adobe Zeri, MoveNet, etc.) for adaptive bit rate streaming. Specifically, a few files of the same content are created locally and stored at the content cache. Each file corresponds to a different streaming rate. Moreover, the format of each file includes two parts: i) a manifest file for meta data, and ii) a set of media content files, each of which contains video content of a fixed playback duration (e.g., 2 seconds). When the user requests some content, the streaming engine replies with a required streaming rate, which is determined by the physical capability of the media outlet and the network status. Based on the required streaming rate, the content engine will stream the content from a chosen file among all available copies cached locally. If a particular playback rate is not immediately available, the most logic approach is to stream the content with the closest rate from below.

B. System Model

1) User Request Model: User request from a mobile media outlet is characterized by a required playback rate, denoted as r. The required playback rate depends on both the physical capability of the media outlet (e.g., screen size) and the network channel condition (e.g., available bandwidth). Normally, r can be modeled as a random variable with a specific probability density function of $f_R(r)$ for $r \in [r_0, r_n]$, where r_0 and r_n are the lower and upper bound of r, respectively. In this paper, we assume the user request playback rate follows a uniform distribution, with the following probability density function,

$$f_R(r) = \frac{1}{r_n - r_0}, \quad r \in [r_0, r_n]\,. \tag{1}$$

A non-uniform distribution can be approximated with a flat line in its any local neighborhood, according to the basic principle in differential geometry [21]. Although a more realistic model can be assumed, it would render our mathematical approach intractable. The operating principle derived from the simple assumption can be made into software more reliable, with a limited performance penalty. Moreover, the results obtained can be extended to other random models, providing operational guidelines for practical HTTP ABR systems.

2) QoE Model: QoE measures the user's satisfactory of viewing a video. In the video streaming system, it not only depends on QoS, but also some other factors, for instance, the quality of the video content. In this paper, we present a simplified QoE model, in which the user's experience depends on two system parameters, including the required playback rate of r and the actual playback rate of r_i : $Q(r_i, r) = h(r_i, r)$. Practically, $r_i \leq r$. Moreover, we further assume that the QoE function[1] has the following properties, including

- $h(r_i, r)$ is always positive;
- $h(r_i, r)$ is concave against r_i;
- $h(r_i, r)$ is continuous and twice differentiable for r_i.

In [17], user experience follows the logarithmic law, and QoE function can be modeled in the logarithmic form for applications of file downloading and web browsing. As such, we adopt the QoE model as the logarithmic function between r_i and r in the HTTP ABR scheme, which is specified as

$$Q(r_i, r) = \alpha \ln \frac{\beta r_i}{r}, \tag{2}$$

where the constant parameters α and β are both positive, and can be different for various types of applications.

3) Content Caching Model: In a typical HTTP ABR streaming system, the cache engine conducts a transcoding process to convert the received media format into HTTP ABR format. Specifically, the received content file is transformed into a set of content files, each of which represents one streaming rate. We assume that n copies of HTTP ABR files are created in the cache, and each copy has a streaming rate of $r_i > 0$, $i = 0, 1, \ldots, n-1$. We call the vector of chosen playback rates as a rate profile, i.e., $\vec{r_i} = (r_0, r_1, r_2, \ldots, r_{n-1})$. Without loss of generality, we assume that $r_0 < r_1 < \cdots < r_{n-1}$. Moreover, the size of a content file with a streaming rate of r_i is given by $C_i = g(r_i)$, where $g(r_i)$ is a non-decreasing, continuous and twice differentiable function of the streaming rate.

[1]In this paper, we use QoE function and Mean-Opinion-Score (MOS) interchangeably.

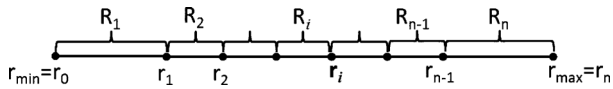


Fig. 2. Illustration of playback rates for cached media files and request playback rates.

In this case, the total storage capacity required at the content cache engine can be calculated as

$$C_{tot} = \sum_{i=0}^{n-1} g(r_i). \tag{3}$$

We model the storage size of the files as the affine function

$$g(r_i) = ar_i + b, \tag{4}$$

where ar_i represents the size of the media content and b represents the meta data stored on the media cache engine.

C. Problem Formulation

In real system development, the software architect normally prefers to solve a series of snapshot problems rather than the unstructured dynamic evolution of the underlined system. In this paper, we focus on the snapshot problem. Specifically, we consider how to maximize the expected QoE by optimally caching HTTP ABR content files in the media cloud. Obviously, it is beneficial to cache streaming files in different bit rates as many as possible for a higher QoE. However, in a real system deployment, the storage space in the content cache can be filled up rapidly, in order to meet different QoE requirements. Practically, it does not allow numerous files to be cached for streaming. As such, one should control the storage budget, by optimally choosing a subset of playback rates, for which a copy of the media content is cached.

Fig. 2 shows the scenario of the cached playback rates and the requested playback rates. There are n copies to be cached, i.e., $r_0, r_1, \ldots, r_{n-1}$. If the requested bit rate r falls within the region R_{i+1} (i.e., $r \in [r_i, r_{i+1})$), we will assign r_i as the replied bit rate. In this case, our task is to find $r_1, r_2, \ldots, r_{n-1}$ between the given minimal rate r_0 and the maximal rate r_n to maximize the average QoE, while respecting a given storage budget constraint. Mathematically, the problem can be formulated as the following constrained optimization problem,

$$\begin{aligned} \max_{n,\vec{r_i}} \quad & \mathbb{E}[Q], \\ \text{s.t.} \quad & C_{tot} \le C, \end{aligned} \tag{5}$$

where $\vec{r_i} = (r_0, r_1, r_2, \ldots, r_{n-1})$ is a rate profile, and the expectation of the QoE is taken over the distribution of user request playback rate (i.e., $f_R(r)$).

The entire QoE volume is a summation of all such regions $R_i = [r_i, r_{i+1})$, as shown in Fig. 2. The objective function hence is a sum of the individual QoE in every region. We integrate within each region because the requested bit rate r is continuous. Therefore, the optimization problem can be re-written as

$$\begin{aligned} \max_{n,\vec{r_i}} \quad & F = \sum_{i=0}^{n-1} \int_{r_i}^{r_{i+1}} Q(r_i, r) f_R(r) dr, \\ \text{s.t.} \quad & \sum_{i=0}^{n-1} (ar_i + b) \le C. \end{aligned} \tag{6}$$

Replacing the QoE function with its reverse, we transform the maximization problem into the following equivalent minimization problem,

$$\begin{aligned} \min_{n,\vec{r_i}} \quad & D = -\sum_{i=0}^{n-1} \int_{r_i}^{r_{i+1}} Q(r_i, r) f_R(r) dr, \\ \text{s.t.} \quad & \sum_{i=0}^{n-1} (ar_i + b) \le C. \end{aligned} \tag{7}$$

Notice that this optimization is different from the classical source coding problem in Information Theory [22], because in our formulation, each segment is represented by its lower bound, compared to the mid-point in the source-coding problem. As such, the classical Max-Lloyd algorithm [23], [24] cannot be used.

In Sections IV and V, we will provide the solution of snapshot problem, i.e., how to obtain the optimal set of streaming bit rate, n^* and $\vec{r_i^*}$, for the cache management. Notice that the formulation only concerns the cache management for individual content. In a real system, multiple contents could compete for storage space in the media cloud. It is beyond the scope of this paper to address the interplay among different contents. Nevertheless, the results obtained for individual content would be instrumental for the optimization problem involving multiple contents.

IV. Optimal QoE-Driven Cache Management for Snapshot Problem

In this section, we adopt a two-step process to solve the snapshot problem:

- *Deriving the optimal set of playback rates for a fixed number of cache copies*: in this step, we first identify the optimization problem of (7) as a convex optimization with respect to r_i for a fixed n, in Section IV-A. Then, we use the Lagrange multiplier method to solve this constrained optimization problem in Section IV-B, and characterize the optimal solution for content cache management in Section IV-C. Our investigation reveals a fundamental phase in exploring the available storage budget to maximize the offered QoE metrics. Moreover, we show that, as the increase of n, the optimal QoE function first increases monotonically and then decreases monotonically.
- *Searching for an optimal number of cache copies*: in this step, based on this property of the optimal QoE function, we present three algorithms to find n^* for the optimal QoE in the content cache management, and provide the analysis of their performance respectively, in Section IV-D.

A. Convexity of the Optimization Problem for a Fixed n

In this subsection, we first show that the optimization problem, for a fixed n, is convex. We consider the Hessian matrix H for the objective function D in (7), which is given as shown in the equation at the bottom of the page, where $m = (\alpha/r_n - r_0) > 0$.

The kth determinant minor of the matrix H can be derived inductively. Specifically, $|H_k|$ $(k = 1, 2, \ldots, n-1)$ is given by

$$|H_k| = \frac{\alpha}{(r_n - r_0)\prod_{i=1}^{k} r_i}\left[1 + r_{k+1}\sum_{i=1}^{k}\frac{1}{r_i}\right]. \quad (8)$$

To show this, we can first consider the cases of $k = 1$ and $k = 2$. For these two cases, we can trivially reach (8). Then, if we assume (8) holds for $k-2$ and $k-1$, $|H_k|$ can be calculated by

$$|H_k| = h_{k,k}|H_{k-1}| - h_{k-1,k}h_{k,k-1}|H_{k-2}|. \quad (9)$$

Following that, we can reach (8) for the case of k, which follows by induction.

Notice that each of the r_i is positive, thus the determinant of all the principal minors of H are greater than 0, i.e., $|H_k| > 0$. Therefore, H is positive definite, and the objective function D is strictly convex.

Also, the constraint in (7) is an affine, i.e., the constrained set is convex. Hence, the optimization problem of (7), for a fixed value of n, is a convex optimization problem. Since (6) and (7) are equivalent, (6) is also a convex optimization problem.

B. Optimal Rate Profile for a Fixed n

In this subsection, we use the Lagrange multiplier method to solve the optimization problem for a fixed value of n. Specifically, the Lagrangian function of the optimization problem (7) is given by

$$L(r_i, \lambda, \mu_i) = D + \lambda\left[\sum_{i=0}^{n-1}(ar_i + b) - C\right] + \sum_{i=0}^{n-1}\mu_{i+1}(r_i - r_{i+1}), \quad (10)$$

where

$$D = -\sum_{i=0}^{n-1}\int_{r_i}^{r_{i+1}} \alpha \ln\frac{\beta r_i}{r}\frac{1}{r_n - r_0}dr.$$

Since it is a convex optimization problem with respect to r_i $(i = 1, 2, \ldots, n-1)$, the KKT conditions are necessary and sufficient for a global minimum of D subject to the inequality constraints. Using the KKT conditions, we have the following equations,

$$\frac{\partial L}{\partial r_i} = -\frac{\alpha}{r_n - r_0}\left[\ln\frac{r_{i-1}}{r_i} + \frac{r_{i+1}}{r_i} - 1\right] + \lambda a + (\mu_{i+1} - \mu_i) = 0, \quad (11)$$

$$\lambda\left[\sum_{i=0}^{n-1}(ar_i + b) - C\right] = 0, \quad (12)$$

$$\mu_{i+1}(r_i - r_{i+1}) = 0, \quad (13)$$

$$\lambda \geq 0, \quad (14)$$

$$\mu_i \geq 0. \quad (15)$$

To solve the optimization problem, we introduce a set of slack variables z and d_i, following the approach in [25], to convert the inequality constraints into equality constraints. Therefore, the extended Lagrangian function can be written as,

$$L(r_i, \lambda, z, \mu_i, d_i) = D + \lambda\left[\sum_{i=0}^{n-1}(ar_i + b) + z^2 - C\right] + \sum_{i=0}^{n-1}\mu_{i+1}(r_i - r_{i+1} + d_{i+1}^2). \quad (16)$$

Then, we have

$$\frac{\partial L}{\partial r_i} = -\frac{\alpha}{r_n - r_0}\left[\ln\frac{r_{i-1}}{r_i} + \frac{r_{i+1}}{r_i} - 1\right] + \lambda a + (\mu_{i+1} - \mu_i) = 0, \quad (17)$$

$$\frac{\partial L}{\partial \lambda} = \sum_{i=0}^{n-1}(ar_i + b) + z^2 - C = 0, \quad (18)$$

$$\frac{\partial L}{\partial z} = 2\lambda z = 0, \quad (19)$$

$$m\begin{bmatrix}
\frac{1}{r_1} + \frac{r_2}{r_1^2} & -\frac{1}{r_1} & 0 & 0 & \ldots & 0 & 0 & 0 \\
-\frac{1}{r_1} & \frac{1}{r_2} + \frac{r_3}{r_2^2} & -\frac{1}{r_2} & 0 & \ldots & 0 & 0 & 0 \\
0 & -\frac{1}{r_2} & \frac{1}{r_3} + \frac{r_4}{r_3^2} & -\frac{1}{r_3} & \ldots & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \ldots & -\frac{1}{r_{n-3}} & \frac{1}{r_{n-2}} + \frac{r_{n-1}}{r_{n-2}^2} & -\frac{1}{r_{n-2}} \\
0 & 0 & 0 & 0 & \ldots & 0 & -\frac{1}{r_{n-2}} & \frac{1}{r_{n-1}} + \frac{r_n}{r_{n-1}^2}
\end{bmatrix}$$

$$\frac{\partial L}{\partial \mu_i} = r_{i-1} - r_i + d_i^2 = 0, \tag{20}$$

$$\frac{\partial L}{\partial d_i} = 2\mu_i d_i = 0. \tag{21}$$

Since $d_i \neq 0$, the variables μ_i must be zero. As a result, organizing these equations together, we obtain

$$\begin{cases} \frac{r_{i+1}}{r_i} - \ln \frac{r_i}{r_{i-1}} - 1 - \frac{\lambda a(r_n - r_0)}{\alpha} = 0, \ i = 1, 2, \ldots, n-1 \\ \sum_{i=0}^{n-1}(ar_i + b) + z^2 - C = 0 \\ 2\lambda z = 0 \\ r_{i-1} - r_i + d_i^2 = 0, \ i = 1, 2, \ldots, n. \end{cases} \tag{22}$$

Therefore, to solve the optimization problem (7), it is equivalent to find the solution of the system of nonlinear (22), in which there are $2n+1$ equations and $2n+1$ unknowns. Notice that only the real numbers are allowed for the unknowns.

Finally, to solve this system of nonlinear equations, we can minimize s, where s is defined as the sum of the square of functions on the left hand side of (22), as follows:

$$\min s = \sum_{i=1}^{n-1}\left(\frac{\partial L}{\partial r_i}\right)^2 + \left(\frac{\partial L}{\partial \lambda}\right)^2 + \left(\frac{\partial L}{\partial z}\right)^2 + \sum_{i=1}^{n}\left(\frac{\partial L}{\partial \mu_i}\right)^2. \tag{23}$$

It corresponds to an unconstrained optimization, which can be solved by the method of trust-region-dogleg [26]. In this paper, we leverage the solver (i.e., *fsolve*) provided by Matlab, which implements the trust-region-dogleg algorithm. We set the maximum iteration times to be 1000, and the order of magnitude for s to be 10^{-20} (this value can be increased for higher rate of convergence). That is, if the iteration times reaches 1000 and s is larger than 10^{-20}, then we claim that there is no solution for the set of equations. More details about the numerical evaluation will be given in Section V-B.

C. Characterization of Optimal Rate Profile

In this subsection, we characterize the optimal rate profile for the QoE-driven content cache management. It can be shown that a fundamental phase change exists in the optimal solution.

Considering the complementary slackness $2\lambda z = 0$ in (22), we can have two possibilities, each of which corresponds to a phase in the optimal solution.

1) Phase I ($\lambda = 0, z \neq 0$): This is the case when the available storage budget is not fully utilized, with the remainder of z^2.

The set of (22) can be transformed as follows,

$$\begin{cases} \frac{r_{i+1}}{r_i} - \ln \frac{r_i}{r_{i-1}} - 1 = 0, \ i = 1, 2, \ldots, n-1 \\ \sum_{i=0}^{n-1}(ar_i + b) + z^2 - C = 0 \\ r_{i-1} - r_i + d_i^2 = 0, \ i = 1, 2, \ldots, n. \end{cases} \tag{24}$$

Plugging the optimal conditions in (24) into the objective function in (6), we obtain

$$F^* = \alpha(\ln \beta + 1) + \frac{\alpha}{r_n - r_0}\left[r_1^* + r_n(\ln \frac{r_{n-1}^*}{r_n} - 1)\right], \tag{25}$$

where r_1^* and r_{n-1}^* are their optimal values.

In *Phase I*, we can characterize the optimal rate profile and the corresponding QoE with some nice properties, as detailed in the following three lemmas.

Lemma 1: From the first n equations in (24), there can be at most one solution for $\vec{r_i}$.

Proof: Assume that there are two solutions for the rate profile $\vec{r_i}$. Suppose $\vec{r_i} = (r_0, r_1, \ldots, r_{n-1})$, and $\vec{r_i'} = (r_0, r_1', \ldots, r_{n-1}')$ are two of qualified rate profiles. Then we have $r_1 \neq r_1'$, otherwise it will lead to $r_2 = r_2', \ldots, r_{n-1} = r_{n-1}'$. Without loss of generality, let $r_1 < r_1'$. Then, $(r_1/r_0) < (r_1'/r_0)$ and $-\ln(r_1/r_0) > -\ln(r_1'/r_0)$. Since $(r_2/r_1) - \ln(r_1/r_0) - 1 = (r_2'/r_1') - \ln(r_1'/r_0) - 1$, we have $(r_2/r_1) < (r_2'/r_1')$. In that case, $r_2 < r_2'$. Subsequently, from $(r_{n-1}/r_{n-2}) - \ln(r_{n-2}/r_{n-3}) - 1 = (r_{n-1}'/r_{n-2}') - \ln(r_{n-2}'/r_{n-3}') - 1$, we obtain $r_{n-1} < r_{n-1}'$. However, from $(r_n/r_{n-1}) - \ln(r_{n-1}/r_{n-2}) - 1 = (r_n'/r_{n-1}') - \ln(r_{n-1}'/r_{n-2}') - 1$, we obtain $(r_n/r_{n-1}) < (r_n/r_{n-1}')$, i.e., $r_{n-1} > r_{n-1}'$. Contradiction occurs. Therefore, there can exist at most one solution for $\vec{r_i}$. ■

Lemma 1 suggests that once we find the solution of $\vec{r_i}$, it must be optimal and unique rate profile for the streaming files.

Lemma 2: In *Phase I*, for a given n, the increase of storage size will still produce the same results of $\vec{r_i^*}$, and thus F^* remains the same.

Proof: Since C is irrelevant to the first $n-1$ equations in (24), the increase of cache size will produce the same results of $\vec{r_i^*}$. In addition, F^* in (25) depends on $\vec{r_i^*}$, but not the cache size C. Therefore, F^* remains the same. ■

Lemma 2 suggests that the expected QoE does not change for a specific n in *Phase I* even if there are more cache storage available.

Lemma 3: Consider two rate profiles, $\vec{r_i} = (r_0, r_1, r_2, \ldots, r_k)$ and $\vec{r_i'} = (r_0, r_1', r_2', \ldots, r_{k+1}')$ in *Phase I*. The rates satisfy the following relationships: $r_i' < r_i$ and $r_{i+1}' > r_i$, where $i = 1, 2, \ldots, k$.

Proof:

(1) Suppose $r_1' \geq r_1$. Then $-\ln(r_1'/r_0) \leq -\ln(r_1/r_0)$, which results in $(r_2'/r_1') \geq (r_2/r_1)$, i.e., $r_2' \geq (r_1'/r_1)r_2 \geq r_2$. Also, we have $-\ln(r_2'/r_1') \leq -\ln(r_2/r_1)$, which results in $r_3' \geq r_3$. Subsequently, it follows that $r_k' \geq r_k$ and $-\ln(r_k'/r_{k-1}') \leq -\ln(r_k/r_{k-1})$, which results in $(r_{k+1}'/r_k') \geq (r_n/r_k)$, i.e., $r_{k+1}' \geq (r_k'/r_k)r_n \geq r_n$. Contradiction occurs. Hence, $r_1' < r_1$. This will result $r_1' < r_1$, $r_2' < r_2, \ldots, r_k' < r_k$.

(2) Suppose $r_{k+1}' \leq r_k$. Then $(r_n/r_{k+1}') \geq (r_n/r_k)$, which results in $-\ln(r_{k+1}'/r_k') \leq -\ln(r_k/r_{k-1})$ and $(r_{k+1}'/r_k') \geq (r_k/r_{k-1})$, i.e., $r_k' \leq (r_{k+1}'/r_k)r_{k-1} \leq r_{k-1}$. Subsequently, $r_1' \leq r_0$. Contradiction occurs. Hence, $r_{k+1}' > r_k$. This will also result $r_k' > (r_{k+1}'/r_k)r_{k-1} > r_{k-1}$. Further, we have $r_{i+1}' > r_i$ for $i = 1, 2, \ldots, k$.

■

Lemma 3 indicates that as n increases, r_1 will decrease and r_{n-1} will increase.

Using these lemmas, we can derive the following theorem that describes the property of the optimal objective function F^* (i.e., the expected QoE) in *Phase I*.

Theorem 1: In *Phase I*, the optimal objective function F^* is increasing with n.

Proof: See Appendix A. ∎

Theorem 1 suggests that, in this phase, we should cache more copies with different bit rates, in order to improve the QoE.

Moreover, since $\sum_i r_i$ is increasing with n, the remainder z^2 will decrease with the increase of n, approaching to be zero eventually, which is *Phase II* ($\lambda > 0, z = 0$) in Sections IV-C and IV-D.

2) Phase II ($\lambda > 0, z = 0$)*:* The set of (22) can be transformed as follows,

$$\begin{cases} \frac{r_{i+1}}{r_i} - \ln \frac{r_i}{r_{i-1}} - 1 - \frac{\lambda a(r_n - r_0)}{\alpha} = 0, \ i = 1, 2, \ldots, n-1 \\ \sum_{i=0}^{n-1}(ar_i + b) - C = 0 \\ r_{i-1} - r_i + d_i^2 = 0, \ i = 1, 2, \ldots, n. \end{cases} \tag{26}$$

It corresponds to the case when the available storage budget is fully utilized. Plugging the optimal conditions in (26) into the objective function (6), we obtain

$$F^* = \alpha(\ln \beta + 1) + \frac{\alpha}{r_n - r_0}\left[r_1^* + r_n(\ln \frac{r_{n-1}^*}{r_n} - 1)\right] + \lambda^* a \sum_{i=1}^{n-1} r_i^*, \tag{27}$$

where r_i^* $(i = 1, 2, \ldots, n-1)$ and λ^* are their optimal values.

Similarly, for *Phase II*, we can characterize several properties of the optimal solution in the following lemmas.

Lemma 4: For the first n equations in (26) there can be at most one solution for $\vec{r_i}$ and λ.

Proof: We prove it by showing that for distinct λ, there cannot exist the rate profiles $\vec{r_i} = (r_0, r_1, r_2, \ldots, r_{n-1})$ and $\vec{r_i'} = (r_0, r_1', r_2', \ldots, r_{n-1}')$ satisfying the first n equations in (26).

We first assume it is true, i.e., there are two distinct rate profiles $\vec{r_i} = (r_0, r_1, r_2, \ldots, r_{n-1})$ with λ, and $\vec{r_i'} = (r_0, r_1', r_2', \ldots, r_{n-1}')$ with λ'. If $\lambda = \lambda'$, then $r_i = r_i'(i = 1, 2, \ldots, n-1)$ by Lemma 1. As a result, $\lambda \neq \lambda'$. Without loss of generality, let $\lambda' > \lambda$.

Suppose $r_1' \geq r_1$. Then $r_2' > r_2, \ldots, r_{n-1}' > r_{n-1}$. This will lead to $\sum_{i=0}^{n-1}(ar_i' + b) > \sum_{i=0}^{n-1}(ar_i + b)$. For different λ, the sum of the $ar_i + b$ cannot reach the same C. Contradiction occurs. Hence, $r_1' < r_1$.

Suppose $r_{n-1}' \geq r_{n-1}$. Then $(r_n/r_{n-1}') \leq (r_n/r_{n-1})$, which results in $-\ln(r_{n-1}'/r_{n-2}') \geq -\ln(r_{n-1}/r_{n-2})$, i.e., $(r_{n-1}'/r_{n-2}') \leq (r_{n-1}/r_{n-2})$. As such, $r_{n-2}' \geq (r_{n-1}'/r_{n-1})r_{n-2} \geq r_{n-2}$. Subsequently, we have $r_1' \geq r_1$. Contradiction occurs. Hence, $r_{n-1}' < r_{n-1}$.

Afterwards, suppose $r_{n-2}' \geq r_{n-2}$. It will lead to $r_{n-3}' \geq r_{n-3}, \ldots, r_1' \geq r_1$. Contradiction occurs. Hence, $r_{n-2}' < r_{n-2}$.

Similarly, it follows that $r_{n-3}' < r_{n-3}, \ldots, r_2' < r_2$. Since $\sum_{i=0}^{n-1}(ar_i' + b) < \sum_{i=0}^{n-1}(ar_i + b)$, for different λ, the sum of the $ar_i + b$ cannot reach the same C. Therefore, there exists at most one solution for $\vec{r_i}$ and λ. ∎

Lemma 5: In *Phase II*, as n increases, r_1 and r_{n-1} will monotonically decrease while λ will monotonically increase.

Proof: See Appendix B. ∎

Lemma 5 suggests that as n increases, r_1 will approach to r_0. As such, the system of equations will have no solution when r_1 is very close to r_0. Also, if there is no solution for $n = k$, there will be no solution for $n = k + 1$ since r_1 cannot go further as well.

Lemma 6: In *Phase II*, the optimal objective function F^* does not monotonically increase with n.

Proof: F^* depends on the variables r_1^*, r_{n-1}^* and λ^*, as indicated in (27). By Lemma 5, we notice that r_1^* and r_{n-1}^* will both decrease monotonically, while λ^* increases monotonically.

Hence, there can be two possibilities: (1) F^* monotonically decreases, or (2) F^* initially increases, and then decrease monotonically. Whichever the case it is, F^* is not monotonically increasing. ∎

Lemma 7: During the phase change, i.e., from *Phase I* to *Phase II*, there is a gain for the optimal objective function F^*.

Proof: This can be verified by Theorem 1. Consider two rate profiles in *Phase I* with $F^*(k)$, and *Phase II* with $F'^*(k+1)$, where k is the maximum number of cached files allowed in *Phase I*. As z^2 decreases, the optimal solution will move from *Phase I* to *Phase II*. This transition will result in $F'^*(k+1) > F^*(k)$, where $F'^*(k+1)$ can be viewed as a special case of *Phase I* when z^2 is extremely small. ∎

Then, based on the analysis above, we can have the following theorem that describes the properties of the optimal solution for rate profile.

Theorem 2: As the number of cached contents increases, the maximum expected QoE measurement first increases monotonically, and then decreases monotonically. Moreover, the maximum expected QoE measurement is achieved in *Phase II*, i.e., optimal QoE is achieved by fully using the storage budget.

This theorem suggests that it is desirable to cache a proper number of content files in the media cloud to maximize the expected QoE. In Section IV-D, we will investigate how to find the optimal value for the number of cached contents in the media cloud.

D. Algorithms of Searching an Optimal Value of n

In this subsection, we consider the problem of how to find the optimal number of copies n for the rate profile, in order to maximize the QoE function. In particular, based on the property of the optimal solution described in Section IV-C, we present three algorithms to find n^* and provide the performance analysis for these algorithms respectively.

Theorem 2 in Section IV-C indicates that the cache budget should be fully utilized for the optimal QoE. In other words, the rate profile under the partial utilization of the cache budget will provide a smaller QoE. As such, to obtain the optimal rate profile, we can ignore the cases when the cache size is partially utilized.

When the cache size is fully utilized, we have $\sum_{i=0}^{n-1}(ar_i + b) = C$. In addition, the rate profile $\vec{r_i}$ follows that $r_0 < r_1 < \cdots < r_{n-1}$. In this case, we have $n(ar_0 + b) < C < n(ar_n + b)$. Therefore, we can estimate a lower and an upper bound for n^*, given by $n_l = \lceil (C/ar_n + b) \rceil$ and $n_u = \lfloor (C/ar_0 + b) \rfloor$. In the following, we present three algorithms to find n^* among $[n_l, n_u]$.

The first brute-force one is an exhaustive searching algorithm that computes the QoE functions sequentially with a step size of one, starting from $n = n_l$. For each n, we solve the system of equations in (22), calculate the QoE function $F(n)$, and compare it with the previous one of $F(n-1)$. If $F(n-1) < F(n)$, the search continues. This process terminates when we reach $F(n) > F(n+1)$, and that n is the optimal number of the cached content copies in the media cloud. The pseudo-code description for the exhaustive search method is presented in Algorithm 1.

Algorithm 1 The Exhaustive Search Method to Find n^*

Require: n_l, n_u, $n = n_l$

Ensure: $n^* = n - 1$

calculate $F(n)$

$n = n + 1$

while $n \neq n_u$ **do**

calculate $F(n)$

if $F(n-1) > F(n)$ **then**

break

end if

$n = n + 1$

end while

The second one is a Dichotomous-based searching algorithm, illustrated in Algorithm 2. The Dichotomous-based search is based on *region elimination*. The search region of each iteration is $[lower, upper]$, in which initially the variable $lower$ is set to be n_l, and $upper$ to be n_u. In each search iteration, we examine the case of $n = mid$ (i.e., the half of $lower$ and $upper$), and update the variables $lower$ or $upper$ by the rules as follows. On one hand, if there exists no solution for $n = mid$, there will be no solutions for $n > mid$; hence, we can eliminate the region to $[lower, mid]$. On the other hand, if there exists a solution for $n = mid$, then we calculate $F(mid-1)$ and compare it with $F(mid)$. If $F(mid) < F(mid-1)$, there are no better solution for $n > mid$; hence, we can eliminate the region to $[lower, mid-1]$. If $F(mid) > F(mid-1)$, there are no better solution for $n < mid-1$; hence, we can eliminate the region to $[mid, upper]$. The algorithm stops when $lower$ and $upper$ are close enough, i.e., $lower + 1 == upper$ or $lower == upper$.

Algorithm 2 The Dichotomous-based Search Method to Find n^*

Require: n_l, n_u, $lower = n_l$, $upper = n_u$

Ensure: $n^* = mid$

while !($lower + 1 == upper$ or $lower == upper$) **do**

$mid = (lower + upper)/2$

if there has no solution for $n = mid$ **then**

$upper = mid$

else

calculate $F(mid)$

calculate $F(mid-1)$

if $F(mid) < F(mid-1)$ **then**

$upper = mid - 1$

else

$lower = mid$

end if

end if

end while

if $F(lower) > F(upper)$ **then**

$mid = lower$

else

$mid = upper$

end if

The third one is a variable step-size searching algorithm, as shown in Algorithm 3. The algorithm solves the system of equations for the case of n and $n+1$, and calculates the objective functions, $F(n)$ and $F(n+1)$, for each search iteration. Initially, n starts from n_l. In the following search iterations, n is increased or decreased by a step size, in response to the comparison between $F(n)$ and $F(n+1)$. If $F(n) < F(n+1)$, the tentative search direction is correct (feasible); hence, we can move forward and the step size, denoted as t_i, is increased exponentially, i.e., $t_i = 2t_{i-1}$, where i is the number of iterations and $t_0 = 1$. If $F(n) > F(n+1)$, however, the tentative search direction is incorrect (infeasible); hence, n should be returned back and decreased exponentially by the step size. We define another variable s_i to denote the step size for the case of back return, i.e., $s_i = 2s_{i-1}$, where i is the number of iterations and $s_0 = 1$. In addition, if there is no solution for n, then we move back to the previous n and continue the search. As such, n will fluctuate within the feasible set until it converges to the optimal one.

Algorithm 3 The Variable Step-size Search Method to Find n^*

Require: n_l, $n = n_l$, $t_0 = 1$, $s_0 = 1$, $i = 0$

Ensure: $n^* = n_k$

while there does not exist an n_k such that $f(n_k - 1) < f(n_k)$ and $f(n_k) > f(n_k + 1)$ **do**

if there has no solution for n **then**

//go back and start from the previous n

$n = n - t_{i-1}$

$t_i = 1$

$n = n + t_i$

$t_{i+1} = t_i * 2$

$s_{i+1} = 1$

else

calculate $F(n)$

if there has no solution for $n+1$ **then**

//go back and start from the previous n

$n = n - t_{i-1}$

$t_i = 1$

$n = n + t_i$

$t_{i+1} = t_i * 2$

$s_{i+1} = 1$

else

calculate $F(n+1)$

if $F(n) < F(n+1)$ **then**

//n is increased by t_i

$n = n + t_i$

$t_{i+1} = t_i * 2$

$s_{i+1} = 1$

else

//n is decreased by s_i

$n = n - s_i$

$s_{i+1} = s_i * 2$

$t_{i+1} = 1$

end if

end if

end if

$i = i + 1$

end while

We first present a preliminary analysis of the complexity of the three alternative searching algorithms for finding n^*. Numerical analysis will be detailed in Section V.

First, the exhaustive searching algorithm is the simplest one. It solves the system of (22) once in each iteration, and terminates when an n is found such that $F(n) > F(n+1)$. The average and worst search times to solve (22) are $O(n_u - n_l + 1)$.

Second, for the Dichotomous-based searching algorithm, the region is reduced by half in each iteration, which results in the average and worst search time complexity to be $O(log_2(n_u - n_l + 1))$. But if n_u is very large, there may be no solution for (22) starting with the mid point such that the trust-region algorithm to find the solutions will iterate for 1000 times, which is time-consuming. Notice that when there is a solution for (22), the iteration times can be much less, with only about 10 times for the convergence (cf. Section V-B).

Third, the variable step-size search method generally can have better performance. On one hand, the variable step-size search starts from n_l so that it does not have the problem of large n_u as the Dichotomous-based algorithm. On the other hand, n is increased or decreased by the step size, in response to the comparison between $F(n)$ and $F(n+1)$ in each iteration. Specifically, n would first increase exponentially if $F(n) < F(n+1)$, but will return back and decrease exponentially if $F(n) > F(n+1)$.

Moreover, the search times of the variable step-size algorithm can be analyzed as follows. On one hand, the average and worst search times of the variable step-size algorithm are no more than $n_u - n_l + 1$, because this algorithm can skip some iterations during the search, compared with the exhaustive search algorithm whose step size is always 1. On the other hand, the variable step-size algorithm cannot have less average and worst search times than and the Dichotomous-based algorithm, i.e., $log_2(n_u - n_l + 1)$. As such, the average and worst search times of the variable step-size algorithm are between $log_2(n_u - n_l + 1)$ and $n_u - n_l + 1$. However, we have to solve (22) twice in each search iteration with the variable step-size search algorithm.

V. Numerical Analysis and Results

In this section, we provide the numerical results of the optimal content cache management for HTTP ABR streaming over wireless networks. First, we verify our proposed QoE model, based on a set of experiments over an SVC video database, consisting of eight different video clips. Next, for a given n, we illustrate the phase change in our proposed algorithm. Finally, the three algorithms of how to find n^* are evaluated and their complexities are compared under two alternative complexity metrics.

A. Verification of QoE Model

In this subsection, we verify our QoE model based on the QoE database that consists of eight videos (i.e., harbour, ducks, parkjoy, city, crew, soccer, ice and oldtown, as shown in Fig. 3). These videos are saved in H.264/SVC format [27]. The presentation time of each video is 10s. The QoE of these videos can be measured by the MOS scaling from 1 to 5, where 5 represents the service is excellent and 1 represents bad. We have conducted an experimental QoE study for the adapted SVC bitstreams. The SVC bitstreams were adapted by the Bitstream Extraction in JSVM [27] for different bit rate and subjective tests were conducted accordingly.[2] In the subjective test, 22 non-expert viewers with normal or corrected-to-normal vision acuity participated in the single-stimulus test for evaluation based on the Adjectival Categorical Judgment Methods in [28]. The viewing

[2]Although the subject test is not conducted with real HTTP ABR streaming, the QoE model derived from the rate adaptation over SVC can be readily applied to HTTP ABR streaming.

Fig. 3. Eight types of videos. (a) Harbour. (b) Ducks. (c) Parkjoy. (d) City. (e) Crew. (f) Soccer. (g) Ice. (h) Oldtown.

TABLE I
DATA OF RATE(KBPS) AND MOS

Harbour		Ducks		Parkjoy		City		Crew		Soccer		Ice		Oldtown	
rate	MOS	rate	MOS	rate	MOS	rate	MOS	rate	MOS	rate	MOS	rate	MOS	rate	MOS
10378	5.0	19484	5.0	17163	5.0	5332.2	5.0	6520.8	5.0	5870	5.0	2302.6	5.0	2351.4	5.0
4142.4	4.0	8108.5	5.0	6946.8	5.0	2069.7	5.0	2428.8	4.0	2467.4	5.0	1133.3	5.0	1217.5	5.0
2029.9	4.0	2878.9	4.0	2354.7	4.0	896.7	4.0	1275	3.5	1188.4	5.0	573.0	4.0	554.5	3.0
895.3	3.0	1311.9	3.4	1053.6	3.5	437.9	3.5	622.3	3.0	600.4	4.0	336.7	3.0	284.0	3.0
783	3.0	1177.3	3.0	897.2	3.0	388.1	3.0	466.6	3.0	463	3.0	270.8	3.0	257.9	3.0
460.4	3.0	621.5	3.0	498.8	2.5	226.2	2.5	394.1	2.0	357.7	3.0	208.9	2.5	145.1	2.0
100.2	2.0	142.2	2.0	113.9	1.0	52.4	1.0	104.5	1.4	94.8	2.0	61.4	2.0	33.4	2.0
82.8	2.0	117	2.0	88.8	1.4	44.6	1.5	72.2	1.4	63.4	2.0	42.9	2.0	29.5	1.5
68.9	1.0	76.8	2.0	65.6	1.0	38.4	1.0	48.2	1.0	39.1	1.4	28.7	2.0	26.4	1.0

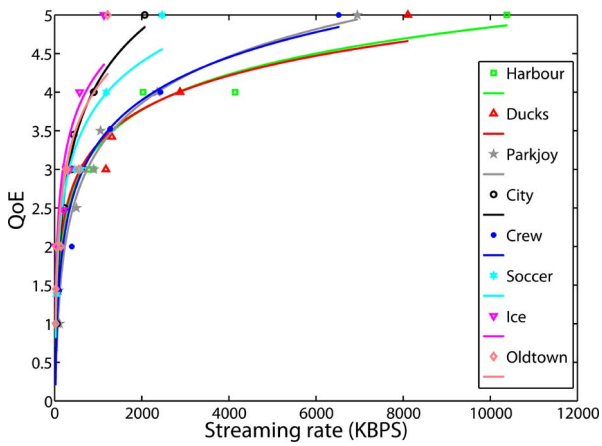


Fig. 4. Approximation of QoE function with streaming rates.

TABLE II
PARAMETERS FOR THE QOE FUNCTION

	α, β	r_0, r_n	$\frac{a(r_n-r_0)}{\alpha}$	$\frac{\|MOS-Q(.)\|_2^2}{N}$
Harbour	0.667, 1479.3	68.9, 10378	15467	0.0779
Ducks	0.634, 1554.8	76.8, 8108.5	12660	0.0514
Parkjoy	0.872, 290.3	65.6, 6946.8	7894	0.040
City	0.976, 143.2	38.4, 2069.7	2082	0.039
Crew	0.802, 419.6	48.2, 2428.8	2968	0.057
Soccer	0.777, 352.3	39.1, 2467.4	3126	0.080
Ice	0.765, 297.3	28.7, 1133.3	1443	0.161
Oldtown	0.787, 218.1	26.1, 1217.5	1515	0.2134

conditions, facility setup and data screening followed the ITU recommendations [28], [29]. Table I provides the rate-MOS results for these videos. These results are applied directly to verify the QoE function (2), which can be rewritten as, $Q(r_i, r) = \alpha \ln \beta + \alpha \ln(r_i/r)$.

Based on the QoE data set in Table I, we adopt a simple yet commonly-used linear regression[3] and obtain the QoE model parameters by minimizing $\|MOS - Q(\cdot)\|_2^2/N$, i.e.,

$$\arg\min_{\alpha,\beta} \frac{\sum_i (MOS_i - \alpha \ln \beta - \alpha \ln \frac{r_i}{r})^2}{N},$$

where i is the index of the fitting data, N is the total number of fitting data and MOS_i is the score in the subjective tests corresponding to r_i. Table II lists the values of α, β, r_0, r_n, $a(r_n - r_0)/\alpha$ and $\|MOS - Q(\cdot)\|_2^2/N$ for eight types of videos. It should be noted that videos with the similar feature (e.g., motion, temporal and spatial information) can have the same parameters of the QoE function model. Therefore, classifying videos into categories can reduce the efforts to model each video and address the scalability issues of modeling QoE function. Fig. 4 indicates that the fitting functions of QoE approximate well with the streaming rates for all the videos.

[3]A more sophisticated approach would be to adopt a weighed curve-fitting mechanism, resulting in complicated analysis. One might have to rely on numerical evaluation, as compared to the closed-form solution we obtained.

TABLE III
THE OPTIMAL RATE PROFILE $\vec{r_i}$ OF CONTENT CACHE MANAGEMENT FOR CITY VIDEO ($C = 3000$ KB)

	Phase I				*Phase II*				
n	2	3	4	5	6	7	8	9	10
z	48.98	40.93	30.24	11.78	2.5e-28	1.56e-23	0	3.0e-36	-1.32e-23
λ	1.5e-36	6.8e-30	-2.3e-34	5.3e-32	4.2e-5	6.8e-5	8.0e-5	8.5e-5	8.8e-5
r_0	38.4	38.4	38.4	38.4	38.4	38.4	38.4	38.4	38.4
r_1	561.9155	313.3511	220.5182	173.3575	115.1226	79.7274	59.4591	47.0031	38.9422
r_2	-	971.1587	605.9671	434.6588	251.6908	149.2230	95.3222	64.8498	46.5990
...	-	-	1218.5062	834.1998	470.7999	263.8175	156.1283	97.2356	63.4723
...	-	-	-	1378.0241	807.2227	451.3724	259.0700	153.8850	94.6768
...	-	-	-	-	1313.7640	757.4690	433.2547	251.8490	149.8223
...	-	-	-	-	-	1256.4907	727.9343	420.6284	245.9452
...	-	-	-	-	-	-	1226.4315	711.0542	412.7589
...	-	-	-	-	-	-	-	1210.5949	701.8355
r_{n-1}	-	-	-	-	-	-	-	-	1202.5478
s	5.2e-26	3.9e-31	2.6e-26	6.5e-27	4.0e-27	8.5e-27	1.8e-26	1.3e-26	2.4e-25
Iterations	13	14	14	14	14	13	12	12	14
QoE	3.7985	4.2230	4.4040	4.5036	4.5537	4.5673	4.5687	4.5663	4.5629

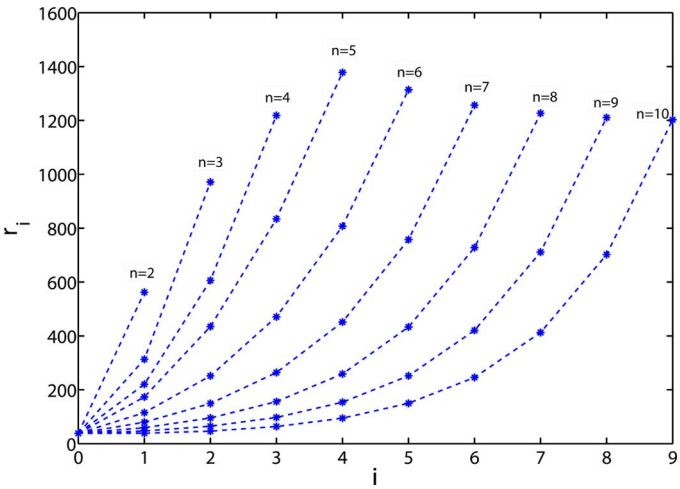


Fig. 5. Schematic illustrations of rate profile $\vec{r_i}$ for city video ($C = 3000$ KB).

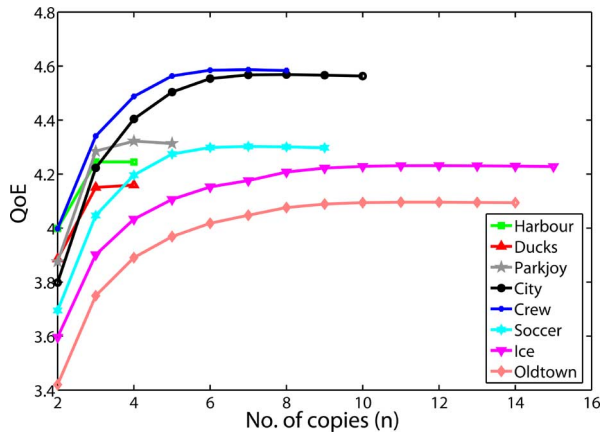


Fig. 6. Relationship of n and QoE ($C = 3000$ KB).

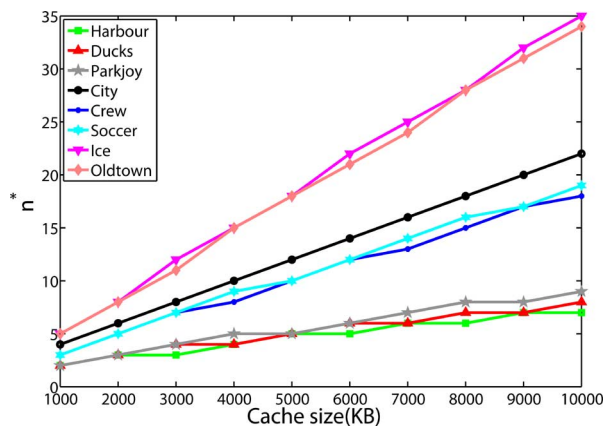


Fig. 7. Relationship of C and n^*.

B. Optimal Rate Profiles for a Fixed n

In this subsection, we present the numerical results of optimal rate profiles when the number of copies is fixed.

As an example, we consider the case of the city video with the cache size $C = 3000$ KB. We set the parameters in the content caching model (4) as $a = 1$ and $b = 0.5$. Given an n, we can find the optimal rate profile of $\vec{r_i}$ for the video. The optimal rate profiles for different n are given in Table III, with a great accuracy for the optimal solution (i.e., $s \approx 0$). It can be observed that, when $n < 6$, the constraint of the cache size is inactive, corresponding to the case of *Phase I*; and when $n \geq 6$, the storage budget is fully utilized, corresponding to the case of *Phase II*. The optimal QoE is achieved at $n = 8$. Therefore, it follows that a phase change occurs and the optimal QoE is achieved in *Phase II*, which agrees with the analysis in Section IV-C.

Fig. 5 gives a schematic illustration of rate profiles $\vec{r_i}$ for the city video. It can be observed that, r_{n-1} monotonically increases in *Phase I* and decreases in *Phase II*. That is, when the cache budget is partially utilized, r_{n-1} expands, trying to make more usage of the cache budget; when the cache budget is fully utilized, r_{n-1} shrinks back to allow other r_is to be fed into the cache.

Fig. 6 shows the relationship between n and QoE for eight videos, in which the cache size is 3000 KB. Notice that with the increase of n, the QoE function first increases, and then decreases upon a threshold. This agrees with Theorem 2.

C. Searching for Optimal n^*

In this subsection, for different videos and cache sizes, we find n^* to maximize the QoE objective functions. In addition,

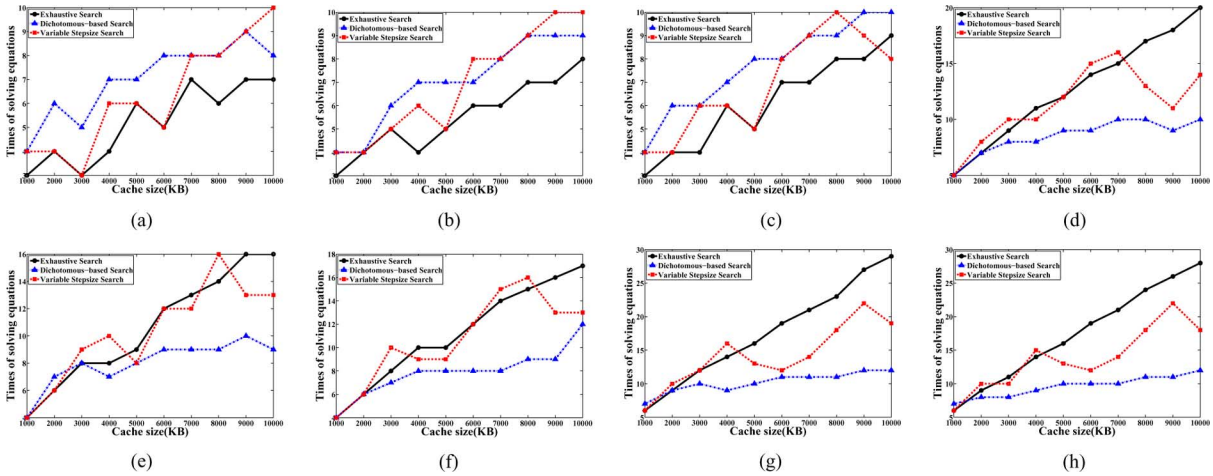


Fig. 8. Times to solve equations of search algorithms for three categories of videos. (a) Category 1: Harbour. (b) Category 1: Ducks. (c) Category 1: Parkjoy. (d) Category 2: City. (e) Category 2: Crew. (f) Category 2: Soccer. (g) Category 3: Ice. (h) Category 3: Oldtown.

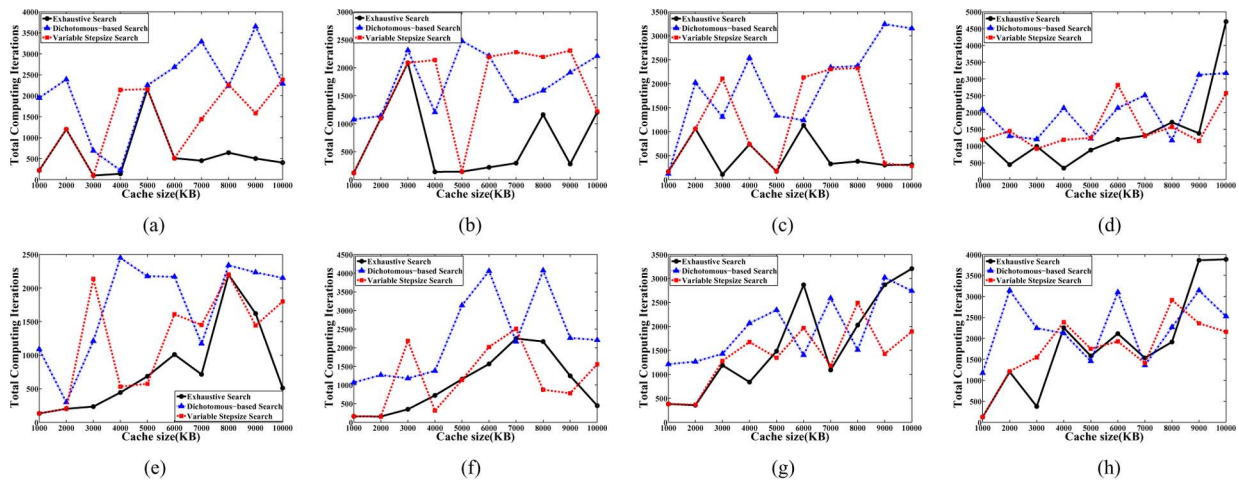


Fig. 9. Total computing iterations of search algorithms for three categories of videos. (a) Category 1: Harbour. (b) Category 1: Ducks. (c) Category 1: Parkjoy. (d) Category 2: City. (e) Category 2: Crew. (f) Category 2: Soccer. (g) Category 3: Ice. (h) Category 3: Oldtown.

we compare the scalability of the aforementioned three alternative algorithms. Numerical results suggest that scalability of the algorithm depends on the parameters of the QoE model for a specific video and the given cache budget.

First, Fig. 7 illustrates the optimal selection of n for various cache budgets. It can be observed that, some different types of videos can have almost the same of n, the number of optimal streaming copies. This indicates that some videos can be aggregated into a category, which can reduce the efforts to find the optimal solution and facilitate our design of the streaming on the cache engine in practice. Roughly, the videos can be divided into three categories in terms of the value of n^*: Category 1, with small n^* for videos of harbour, ducks and parkjoy, i.e., n^* increases slightly with the increase of the cache size; Category 2, with medium n^* for videos of city, crew and soccer, i.e., n^* increases gradually; and Category 3, with large n^* for the videos of ice and old town, i.e., n^* increases dramatically.

This observation can be understood as follows. Considering (22), we notice that the term $(a(r_n - r_0)/\alpha)$, dependent on the QoE model of a specific video, has an influence on n^*. For a given cache budget, if the term $(a(r_n - r_0)/\alpha)$ is larger, each rate in $\vec{r_i}$ will be larger. However, due to the cache constraint, n^* will be smaller. This is can be verified by the fact that Category 1 has large values of $(a(r_n - r_0)/\alpha) > (7000)$, and Category 2 and Category 3 have smaller values of $(a(r_n - r_0)/\alpha)$, with more than 2000 and about 1500, respectively, as shown in Table II.

Second, in Fig. 8, we consider the total number of times solving (22) as a metric for their algorithmic complexity. It shows that the exhaustive search has the smallest number of times solving (22) for Category 1 (with small n^*); Dichotomous-based search and the variable step-size method have a smaller number of times than the exhaustive search for Category 2 (with medium n^*) and Category 3 (with large n^*).

Third, we also evaluate the scalability of the search algorithms, by considering the total number of iterations used by the trust-region algorithm to solve the set of equations in (22). In Fig. 9, we compare the scalability of the three search algorithms of n^* for the three categories of videos with various cache budgets. The variable step-size search has the smallest total computing iterations for videos of Category 3, when the cache size is large. While the exhaustive search has the smallest total com-

TABLE IV
SEARCH ALGORITHMS

Algorithm	Characteristics
Exhaustive	solve a system of equations once for each iteration search sequentially average and worst search times are $O(n_u - n_l + 1)$ efficient when $\frac{a(r_n - r_0)}{\alpha}$ is large and C is small
Dichotomous-based	solve a system of equations twice for each iteration the region is reduced by half in each iteration average and worst search times are $O(log_2(n_u - n_l + 1))$ efficient when n_u is appropriately estimated
Variable step-size	solve a system of equations twice for each iteration step size is varying average and worst search times are between $log_2(n_u - n_l + 1)$ and $n_u - n_l + 1$ efficient when $\frac{a(r_n - r_0)}{\alpha}$ is small and C is large

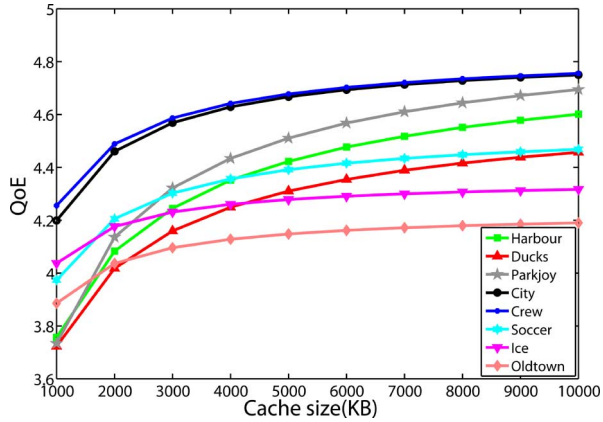


Fig. 10. Relationship of C and maximum QoE.

puting iterations for the videos of Category 1 and Category 2, when the cache size is small.

Moreover, we observe that, although the Dichotomous-based search has a small number of times to solve (22), its number of computing iterations is large. This is because the Dichotomous-based search has a high cost to solve (22) in the initial steps with large n_u, as discussed in Section IV-D. The approach to improve the performance of the Dichotomous-based search is to estimate n_u more accurately. One implication from Fig. 7 is that n^* increases with C. This indicates on how to find the optimal number of streaming files n^*, if some background information is given. Suppose n^* is known for a cache size C. Then, for a larger cache budget C' (i.e., $C' > C$), we can search n' starting from n^* (i.e., $n'_l = n^*$), because the optimal number of cached files for C', i.e., n'^*, should be no less than n^*; for a smaller cache budget C'' (i.e., $C'' < C$), we can set n''_u to be n^* (i.e., $n''_u = n^*$), because the optimal number of cached files for C'', i.e., n''^*, should be no greater than n^*. As a result, if the solution of a larger cache budget is known, we can reduce the value of n_u before using the Dichotomous-based search algorithm to find the optimal cache management, for a given cache budget.

Therefore, the scalability of the algorithms depends on the parameters of the QoE model for a specific video and the given cache budget. Specifically, the exhaustive search is efficient when $(a(r_n - r_0)/\alpha)$ is large and C is small, while the variable step-size search has the best performance when $(a(r_n - r_0)/\alpha)$ is small and C is large, where the term $(a(r_n - r_0)/\alpha)$ depends on the QoE model of the specific video. At the same time, the Dichotomous-based search is efficient only when the upper bound of n can be appropriately estimated. Combining the analysis above and the simulation results in this section, we summarize the characteristics of the algorithms in Table IV.

Finally, we plot the value of optimal QoE with n^* cached content copies as a function of the cache size C in Fig. 10. First, the optimal QoE is a concavely increasing function of the given cache size. As the cache size increases, the marginal benefit of a higher QoE diminishes. This characterization offers practical insights to choose a specific operating point for each video content within a video catalog, maximizing the overall QoE performance for an aggregate storage budget. Second, the expected QoE is larger than 4 for all of the videos if the given cache size is no less than 2000 KB. This indicates that the performance gap between our cache management and the upper bound (i.e., 5) is small.

VI. CONCLUSION

In this paper, we investigated the problem of how to cache a set of media files with optimal streaming rates, under HTTP adaptive bit rate streaming over wireless networks. We formulated this design as an optimization framework (snapshot problem), whose objective is to maximize the QoE objective function for a given storage budget. First, for a fixed number of content copies, we translated this problem into a convex optimization problem, for which we derived the mathematical solution and identified a phase change in the optimal solution. Second, we proposed three alternative search algorithms to find n^* for the optimal content cache management. Numerical results suggested that the scalability of the algorithm depends on the parameters of the QoE model for a specific video and the given cache budget. Finally, simulation results showed that our cache management can provide high expected QoE while requiring low complexity, which gives guidelines for practical design of HTTP ABR streaming services.

In the future, we will consider the model of user requests into a more realistic one (i.e., non-uniform distribution). In addition, we will extend the paper by considering the scenario of multiple distinctive content stored on the cache, coupled with the popularity of these content. Finally, we will also consider an elastic scheme of cloud storage, in which additional storage space can be acquired to meet the demands.

APPENDIX

Proof of Theorem 1: Consider two rate profiles, i.e., $\vec{r_i} = (r_0, r_1, r_2, \ldots, r_k)$ with F^*, and $\vec{r'_i} = (r_0, r'_1, r'_2, \ldots, r'_{k+1})$ with F'^*, where F^* and F'^* are the optimal QoE functions of $\vec{r_i}$ and $\vec{r'_i}$, respectively. To examine if the optimal QoE function will increase as n is larger, it is equivalent to check if the difference between F and F' is positive:

$$\begin{aligned}\Delta F^* &= F'^* - F^* \\ &= \frac{\alpha}{r_n - r_0}[r'_1 + r_n \ln r'_{k+1} - r_1 - r_n \ln r_k]. \quad (28)\end{aligned}$$

Since $\ln x > 1-(1/x)$ for any x ($x>0$ and $x\neq 1$), we have $\ln(r'_{k+1}/r_k) > 1-(r_k/r'_{k+1})$. As a result,

$$\Delta F^* > \frac{\alpha}{r_n - r_0}\left[r_n(1-\frac{r_k}{r'_{k+1}}) + r'_1 - r_1\right], \quad (29)$$

where

$$\begin{aligned} r_n\left(1-\frac{r_k}{r'_{k+1}}\right) &= r_n - r_n\frac{r_k}{r'_{k+1}} \\ &= r_k\left(1+\ln\frac{r_k}{r_{k-1}}\right) - r_k\left(1+\ln\frac{r'_{k+1}}{r'_k}\right) \\ &> r_k\left(1-\frac{r_{k-1}r'_{k+1}}{r_k r'_k}\right) \\ &= r_k - \frac{r_{k-1}r'_{k+1}}{r'_k}. \end{aligned} \quad (30)$$

Subsequently, we can have

$$\begin{aligned} r_n\left(1-\frac{r_k}{r'_{k+1}}\right) &> r_1 - \frac{r_0 r'_2}{r'_1} \\ &= r_1 - r_0\left(1+\ln\frac{r'_1}{r_0}\right) \\ &> r_1 - r_0\frac{r'_1}{r_0} \\ &= r_1 - r'_1. \end{aligned} \quad (31)$$

Hence, $\Delta F^* > 0$, i.e., the optimal QoE function F^* is increasing with n.

Proof of Lemma 5: Suppose that as n increases, λ will decrease. That is, $\lambda' \leq \lambda$, where λ is the multiplier for the rate profile $\vec{r_i} = (r_0, r_1, r_2, \ldots, r_k)$ and λ' is the multiplier for the rate profile $\vec{r'_i} = (r'_0, r'_1, r'_2, \ldots, r'_{k+1})$. Note that $r'_0 = r_0$. Then, we have

$$\frac{r_n}{r_k} - \ln\frac{r_k}{r_{k-1}} - 1 \geq \frac{r_n}{r'_{k+1}} - \ln\frac{r'_{k+1}}{r'_k} - 1. \quad (32)$$

It follows that

$$\frac{r_n(r'_{k+1}-r_k)}{r_k r'_{k+1}} \geq \ln\frac{r_k r'_k}{r_{k-1}r'_{k+1}} \geq 1 - \frac{r_{k-1}r'_{k+1}}{r_k r'_k}. \quad (33)$$

Suppose if $r_k \geq r'_{k+1}$, then

$$r_n\left(1-\frac{r_k}{r'_{k+1}}\right) \geq r'_{k+1}\left(1-\frac{r_{k-1}}{r'_k}\right). \quad (34)$$

It follows that $r_{k-1} \geq r'_k$. Subsequently, we obtain $r_{k-2} \geq r'_{k-1}, \ldots, r_0 \geq r'_1$. Contradiction occurs. Hence, $r_k < r'_{k+1}$. Similarly, it can be proved that $r_{k-1} < r'_k, \ldots, r_0 < r'_1$. Since, $a\sum_{i=0}^{k} r_i + (k+1)b = C$ and $a\sum_{i=0}^{k+1} r'_i + (k+2)b = C$, we have $\sum_{i=0}^{k+1} r'_i < \sum_{i=0}^{k} r_i$, which is inconsistent with $r_k < r'_{k+1}, r_{k-1} < r'_k, \ldots, r_0 < r'_1$. Therefore, $\lambda' > \lambda$.

Then suppose $r_1 \leq r'_1$. Given $\lambda' > \lambda$, we have

$$\begin{cases} \frac{r_{i+1}}{r_i} - \ln\frac{r_i}{r_{i-1}} - 1 < \frac{r'_{i+1}}{r'_i} - \ln\frac{r'_i}{r'_{i-1}} - 1, i = 1, \ldots, k-1 \\ \frac{r_n}{r_k} - \ln\frac{r_k}{r_{k-1}} - 1 < \frac{r'_{k+1}}{r'_k} - \ln\frac{r'_k}{r'_{k-1}} - 1. \end{cases} \quad (35)$$

By (35), we can obtain $r_2 \leq r'_2$, $r_3 \leq r'_3, \ldots, r_k \leq r'_k$, $r_n \leq r'_{k+1}$. Contradiction occurs. Hence, $r_1 < r'_1$. Similarly, we can also prove $r_k > r'_{k+1}$ by contradiction.

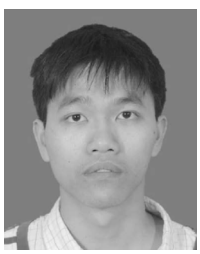
ACKNOWLEDGMENT

The authors would like to thank Dr. C. W. Chen from SUNY, Buffalo, Dr. D.-M. Chiu from the Chinese University of Hong Kong, Dr. X. Zhu from Cisco Systems, Dr. J. Cai from Nanyang Technological University, for their insightful comments and suggestions. Part of this work has been accepted by Globecom 2012. In this work, we present more new results, including the characterization of the property of optimal solution and three alternative algorithms to find n^* for the optimal content cache management.

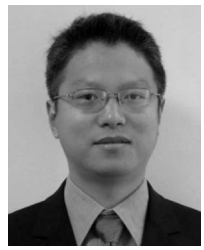
REFERENCES

[1] Cisco, "Cisco Visual Networking Index: Forecast and Methodology, 2010–2015," 2011.
[2] A. Vetro and C. W. Chen, "Rate-reduction transcoding design for wireless video streaming," in *Proc. 2002 Int. Conf. Image Processing*, 2002, vol. 1, pp. 29–32.
[3] Z. Lei and N. D. Georganas, "Adaptive video transcoding and streaming over wireless channels," *J. Syst. Softw.*, vol. 75, no. 3, pp. 253–270, 2005.
[4] A. C. Begen, T. Akgul, and M. Baugher, "Watching video over the web," *Internet Comput.*, vol. 15, pp. 54–63, 2011.
[5] I. Hofmann, N. Farber, and H. Fuchs, "A study of network performance with application to adaptive HTTP streaming," in *Proc. IEEE Int. Symp. Broadband Multimedia Systems and Broadcasting (BMSB)*, 2011, pp. 1–6.
[6] C. Liu, I. Bouazizi, and M. Gabbouj, "Rate adaptation for adaptive HTTP streaming," in *Proc. 2nd Annu. ACM Conf. Multimedia Systems*, 2011, pp. 169–174.
[7] W. Pu, Z. Zou, and C. W. Chen, "Dynamic adaptive streaming over http from multiple content distribution servers," in *Proc. IEEE Globecom 2011*, 2011, pp. 1–5.
[8] Cisco, "Cisco CDS Internet Streaming: Enabling New Video 2.0 Experiences," 2007.
[9] V. Adzic, H. Kalva, and B. Furht, "Optimized adaptive HTTP streaming for mobile devices," in *Proc. SPIE*, 2011, vol. 8135, pp. 81350T–81350T-10.
[10] S. Akhshabi, A. C. Begen, and C. Dovrolis, "An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP," in *Proc. 2nd Annu. ACM Conf. Multimedia Systems*, 2011, pp. 157–168.
[11] T. Lohmar, T. Einarsson, P. Frojdh, F. Gabin, and M. Kampmann, "Dynamic adaptive HTTP streaming of live content," in *Proc. IEEE Int. Symp. World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2011, pp. 1–8.
[12] A. B. Reis, J. Chakareski, A. Kassler, and S. Sargento, "Distortion optimized multi-service scheduling for next-generation wireless mesh networks," in *Proc. INFOCOM IEEE Conf. Computer Communications Workshops*, 2010, pp. 1–6.
[13] S. Thakolsri, W. Kellerer, and E. Steinbach, "QoE-based rate adaptation scheme selection for resource-constrained wireless video transmission," in *Proc. ACM Multimedia*, 2010, pp. 783–786.
[14] A. Khan, L. Sun, E. Jammeh, and E. Ifeachor, "Quality of experience-driven adaptation scheme for video applications over wireless networks," *IET Commun., Special Issue on Video Communications Over Wireless Networks*, pp. 1337–1347, 2010.
[15] A. Khan, I.-H. Mkwawa, L. Sun, and E. Ifeachor, "QoE-driven sender bitrate adaptation scheme for video applications over IP multimedia subsystem," in *Proc. IEEE ICC*, 2011, pp. 1–6.
[16] Q. Li, Y. Andreopoulos, and M. van der Schaar, "Streaming-viability analysis and packet scheduling for video over in-vehicle wireless networks," *IEEE Trans. Veh. Technol.*, vol. 56, no. 6, pp. 3533–3549, 2007.
[17] P. Reichl, B. Tuffin, and R. Schatz, "Logarithmic laws in service quality perception: Where microeconomics meets psychophysics and quality of experience," *Telecommun. Syst.*, pp. 1–14, 2011.
[18] M. Venkataraman and M. Chatterjee, "Evaluating quality of experience for streaming video in real time," in *Proc. IEEE GLOBECOM*, 2009, pp. 1–6.

[19] R. K. P. Mok, E. W. W. Chan, and R. K. C. Chang, "Measuring the quality of experience of HTTP video streaming," in *Proc. 2011 IFIP/IEEE Int. Symp. Integrated Network Management*, 2011, pp. 485–492.
[20] H. L. Kim and S. G. Choi, "A study on a qos-qoe correlation model for qoe evaluation on iptv service," in *Adv. Commun. Technol. (ICACT)*, 2010, pp. 1377–1382.
[21] M. P. do Carmo, *Differential Geometry of Curves and Surfaces*. Englewood Cliffs, NJ, USA: Prentice Hall, 1976.
[22] R. W. Hamming, *Coding and Information Theory*. Englewood Cliffs, NJ, USA: Prentice Hall, 1980.
[23] J. Max, "Quantization for minimum distortion," *IEEE Trans. Inf. Theory*, vol. 6, no. 1, pp. 7–12, 1960.
[24] S. P. Lloyd, "Least squares quantization in pcm," *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, 1982.
[25] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Belmont, MA, USA: Athena Scientific, 1999.
[26] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York, NY, USA: Operations Research, Springer, 1999.
[27] JSVM, "JSVM Software Manual," ver. jsvm 9.19, 2011.
[28] Methodology for the Subjective Assessment of the Quality of Television Pictures, 2002, ITU-R Rec. BT.500-11.
[29] Subjective Video Quality Assessment Methods for Multimedia Applications, 1999, ITU-R Rec. P.910.



Weiwen Zhang received his Bachelor's degree in software engineering and Master's degree in computer science from South China University of Technology (SCUT) in 2008 and 2011, respectively. He is currently a Ph.D. student in the School of Computer Engineering at Nanyang Technological University (NTU) in Singapore. His research interests include cloud computing and mobile computing.



Yonggang Wen (M'08) received his Ph.D. degree in electrical engineering and computer science from Massachusetts Institute of Technology (MIT) in 2008. He is currently an Assistant Professor with School of Computer Engineering at Nanyang Technological University, Singapore. Previously, he has worked in Cisco as a Senior Software Engineer and a System Architect for content networking products. He has also worked as Research Intern at Bell Laboratories, Sycamore Networks, and served as a Technical Advisor to the Chairman at Linear A Networks, Inc. His research interests include cloud computing, mobile computing, multimedia network, cyber security and green ICT.

Zhenzhong Chen (M'07) obtained his B.Eng. degree from Huazhong University of Science and Technology and Ph.D. degree from the Chinese University of Hong Kong, both in electrical engineering. He is currently the member of technical staff at MediaTek USA Inc. in San Jose, CA. Previously, he was a Lee Kuan Yew Research Fellow and a Principal Investigator with the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore. His research interests include visual perception, visual signal processing, and multimedia communications.

Ashish Khisti (M'08) received the B.A.Sc. degree from the Engineering Science program at the University of Toronto, Canada in 2002, and the M.S. and Ph.D. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 2004 and 2008, respectively. Since 2009, he has been an Assistant Professor in the Department of Electrical and Computer Engineering, University of Toronto. His research interests are in the area of information and coding theories and their applications to wireless communication systems, multimedia communication systems, inference, and security.