Towards Cost-Efficient Video Transcoding in Media Cloud: Insights Learned From User Viewing Patterns

Guanyu Gao, Weiwen Zhang, Yonggang Wen, *Senior Member, IEEE*, Zhi Wang, and Wenwu Zhu, *Fellow, IEEE*

***Abstract*—Video transcoding in an adaptive bitrate streaming (ABR) system is demanded to support video streaming over heterogenous devices and varying networks. However, it could incur a tremendous cost. Meanwhile, most viewers terminate viewing sessions within 20% of their durations; only a small fraction of each video is consumed. Built upon this user viewing pattern, we propose a *Partial Transcoding Scheme* for content management in media clouds. Particularly, each content is encoded into different bitrates and split into segments. Some of the segments are stored in cache, resulting in storage cost; others are transcoded online in the case of cache miss, resulting in computing cost. We aim to minimize the long-term overall cost by determining whether a segment should be cached or transcoded online. We formulate it as a constrained stochastic optimization problem. Leveraging Lyapunov optimization framework and Lagrangian relaxation, we design an online algorithm which can achieve the optimal solution within provable upper bounds. Experiments demonstrate that our proposed method can reduce 30% of operational cost, compared with the scheme of caching all the segments.**

***Index Terms*—Media cloud, partial transcoding scheme, user viewing pattern, viewer behavior.**

I. Introduction

VIDEO streaming has become one of the dominant contributors to Internet traffic. Only Netflix and YouTube make up 50.31% of the downstream traffic during the peak period [1]. Meanwhile, global mobile video will increase 16-fold between 2012 and 2017, which accounts for over 66% of total mobile data traffic [2]. However, limited and unstable bandwidth, and diversity of mobile devices inherently deteriorate the user experience, triggering a tussle between the growing demand of video traffic and quality of network service.

Manuscript received August 16, 2014; revised December 24, 2014 and May 08, 2015; accepted May 09, 2015. Date of publication June 01, 2015; date of current version July 15, 2015. This work was supported in part by the Singapore MOE under Grant MOE Tier-1, and in part by the Singapore EMA under Grant EIRP02. The work of G. Gao and Y. Wen was supported by the Singapore National Research Foundation under its IDM Futures Funding Initiative and administered by the Interactive & Digital Media Programme Office, Media Development Authority. The work of Z. Wang was supported by the NSFC under Grant 61402247. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Tommaso Melodia.

G. Gao, W. Zhang, and Y. Wen are with the School of Computer Engineering, Nanyang Technological University, Singapore 639798 (e-mail: ggao001@ntu.edu.sg; wzhang9@ntu.edu.sg; ygwen@ntu.edu.sg).

Z. Wang is with the Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, China (e-mail: wangzhi@sz.tsinghua.edu.cn).

W. Zhu is with the Computer Science Department, Tsinghua University, Beijing 100084, China (e-mail: wwzhu@tsinghua.edu.cn).

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.2015.2438713

Adaptive bitrate streaming [3] has been touted as an enabling technology to support growing media consumption over heterogenous devices and different network conditions. In ABR, by leveraging video transcoding [4], each video content is encoded into different bitrates and resolutions, which can allow users to receive appropriate bitrate stream to accommodate varying network conditions and different types of user devices. It was demonstrated in [5] that a video content could be encoded into more than 40 versions to meet the requirements of different kinds of user devices and network conditions. However, considering the sheer volume of ABR files, it consumes tremendous computing resource and storage resource to encode the video contents into different bitrates and store the encoded video contents. As reported by [6], it would cost millions of pounds for content providers to transcode and store a large number of contents. In contrast, it has also been observed that only 10% of the most popular videos account for almost 80% of total views [7], [8]; for 60% of the videos, only less than 20% of their duration is viewed, and most of users abort viewing within 40 seconds [9]–[12]. This user viewing pattern reveals that most of the video contents are seldom watched, and users consume only a small fraction of each video, which provide a good insight for video service providers to design content management system.

To address the problem that video transcoding for ABR could incur exorbitant cost, we propose a partial transcoding scheme for content management based on user viewing pattern. With the partial transcoding scheme, each original video content is split into a set of segments. Each segment has a fixed playback duration. When viewing videos, the user requests video segments in specified bitrates according to the current available bandwidth and device capacity. If the requested segment is available in storage, it would be consumed by the user immediately, resulting in storage cost; otherwise, a real-time transcoding is conducted, resulting in computing cost. The more segments are cached, the less transcoding tasks will be conducted, and vice versa. We aim to minimize the long term overall cost, including storage and computing cost, while satisfying the time average constraint on storage and computing consumption. We formulate the partial transcoding scheme as a constrained stochastic optimization problem.

To derive the solution of the partial transcoding scheme in real-time according to the current user request information, we apply the Lyapunov optimization framework to design an online algorithm which can minimize the long-term cost over the time span. Experiments demonstrate that: 1) our designed online algorithm can approximately approach to the optimal cost within provable upper bounds; 2) our proposed scheme can save 30% of operational cost, compared with the scheme of caching

all the segments; and 3) our proposed scheme is less sensitive to the change of user request rate than the full transcoding scheme without segmentation.

The contributions of our work are summarized as follows.

- We propose a partial transcoding scheme to reduce the operational cost by leveraging user viewing pattern. To the best of our knowledge, this is the first work to adopt the partial transcoding scheme under user viewing pattern to save operational cost in ABR system. Meanwhile, our proposed method can also increase the number of unique contents cached in the system.
- We apply the Lyapunov optimization framework to design an online algorithm which can derive solutions for the partial transcoding scheme.
- We conduct various of experiments under realistic settings to evaluate the performance of our algorithm. Numerical results demonstrate that our proposed method can reduce operational cost for content providers and provide adaptive control over content management.

The rest of this paper is organized as follows: Section II discusses related work. Section III introduces system architecture. Section IV presents system model and problem formulation. Section V provides our proposed online algorithm which can achieve the cost-efficient video transcoding by applying the Lyapunov optimization framework. Section VI presents verification and evaluation of our proposed method. Finally, Section VII concludes the paper.

II. Related Work

Adaptive bitrate streaming has become a prominent technology to improve the quality of video delivery over different network environments and support media consumption over heterogenous devices [13], [14]. User viewing pattern, which characterizes the user access behaviors when viewing videos, is also a key factor to affect the performance and the operational cost of the media system. There has been a large amount of work studying ABR and user viewing pattern. We will review each of them in the follows, respectively.

In terms of adaptive bitrate streaming, Shen *et al.* [15] proposed a transcoding-enabled proxy system which allows to perform content adaption according to the network environment. Zhang *et al.* [16] aimed to maximize the QoE for mobile users, by storing content copies transcoded in different bitrates under constrained storage budget. Ahlehagh *et al.* [17] introduced a joint video processing and caching framework to support adaptive bitrate streaming, which can improve network capacity and maintain QoE. Huang *et al.* [18] presented a SVC based video proxy to adapt to network dynamics by transcoding the original video into scalable codec at different bitrates. Yang *et al.* [19] proposed an online buffer fullness estimation method for adaptive video streaming. In these research, however, they mainly focus on improving user experience of the client side, without considering the operational cost of the content provider.

Another line of research characterizes the user viewing pattern. Liu *et al.* [5] investigated a top Internet mobile streaming service at the service side, and the measurements showed that a video content could be transcoded into more than 40 versions to deal with device heterogeneity. Cha *et al.* [7] analyzed the UGC VoD system of YouTube and Daum Videos, and found that video popularity followed the power-law distribution with an exponential cutoff, which indicates that a small fraction of videos account for most of the views. Finamore *et al.* [9] studied the Youtube traffic generated by mobile devices and common PCs, and the results revealed that 60% of videos are watched less than 20% of their duration. We can observe from the previous measurements that a video content needs to be transcoded into many versions in different bitrates to adapt the device heterogeneity and varying network conditions in a ABR system. As a result, considering the sheer volume of video files in different versions, it consumes huge amounts of computing resource to transcode videos into different bitrates, and storage cost to store different encoded versions of the video contents. However, as indicated in the previous work, only a small proportion of video contents are frequently consumed by users, most of them are seldom requested. As such, it is not cost efficient to transcode and store each version of the video contents.

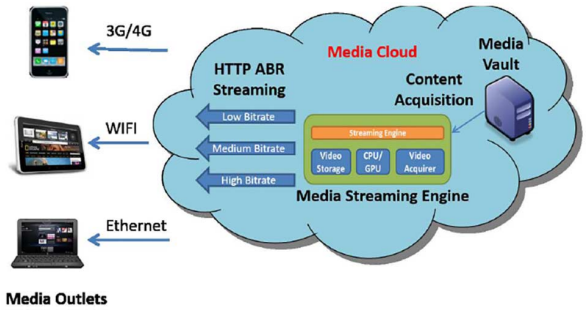


Fig. 1. Schematic diagram for adaptive bitrate streaming system. Contents are transcoded into a set of files in different playback rates and resolutions and cached in the streaming engine.

Our work is motivated by the real problems observed in the previous measurements. It differs from the previous work in that we aim to minimize the operational cost of the content provider by considering the factor of user viewing pattern. In our previous works [20], [21], we have considered it as a one-shot optimization problem. In this work, we further develop an online algorithm to derive the solution of minimizing the long run operational cost.

III. System Architecture

This section presents our proposed system architecture for cloud-based media streaming with enabled transcoding capability, serving a wide range of users. This referenced architecture is based on real experience with one of the leading video service providers in China. Our proposed architecture is illustrated in Fig. 1, which consists of three parts, including media vault, media outlet and streaming engine. The functionalities of each module are as follows.

Media Vault: It stores all of the original contents and their transcoded versions in alternative bitrates, however, it usually locates in only one geographical cloud site.

Streaming Engine: To deliver the video contents to users reliably and in time, streaming engines are deployed in different geographical regions to response for the user requests from media outlets. The video contents in the streaming engine can

be obtained from three alternative resources. First, in the local storage of streaming engine, a partial subset of video contents in various bitrates are stored and can be directly consumed. Second, the whole file of an original video content is stored in local storage, and can be transcoded into the user requested bitrate in a real-time manner by CPU/GPU in the streaming engine. Finally, in a rare case that the transcoding capacity is overloaded, the streaming engine can retrieve the transcoded version from the media vault. In this paper, we only focus on the first two cases.

Media Outlet: It negotiates with the streaming engine for a proper bitrate according to the current network status and its own physical capability, and pulls video segments from the streaming engine and displays them for viewers to consume the video content.

In the streaming engine, any original video content in this system is split into small multi-second segments. Each of the segments has a fixed playback duration (e.g., 10 seconds). These segments are further encoded into versions in different bitrates, supporting heterogenous media outlets and varying network conditions. There are two kinds of alternative cost incurred in the streaming engine. First, it stores a partial subset of transcoded video contents in its local storage, resulting in storage cost. Second, in case of cache miss, it transcodes the original video content into the requested bitrate, resulting in computing cost. It can be seen that the two parts of cost are competing with each other. On one hand, the larger storage capacity is allocated, the more transcoded video contents can be stored, minimizing the probability of cache miss and the opportunity of conducting online transcoding. On the other hand, the more computing resources are provisioned, the less transcoded video contents are needed to store statically, reducing the required storage space. Therefore, it demands a trade-off in optimizing the total cost of service. In the following, we mathematically formulate it into an optimization problem, aiming to minimize the long-run overall cost.

IV. System Model and Problem Formulation

In this subsection, we first present our mathematical models for the cloud-based ABR streaming system, including content management model, user request model and cost model, and then formulate it into a constrained stochastic optimization problem, along with key notations summarized in Table I.

A. System Model

1) Content Management Model: We model the content management as follows, including partial transcoding scheme and dynamic caching scheme.

Partial Transcoding Scheme: We assume that the system manages M video contents. For any video i ($i = 0, \cdots, M-1$), it is split into L_i segments, as illustrated in Fig. 2. Meanwhile, for each video i, it is transcoded into a set of versions in N bitrates. For each bitrate of $r_{i,j}$ ($j = 0, \cdots, N-1$), a partial subset of segments are cached in the streaming engine. We consider a discrete time slot model. In view of the user viewing pattern, we assume that the first $n_{i,j}(t)$ segments (i.e., from 0 to $n_{i,j}(t) - 1$) are cached at the time slot t and the rest of $L_i - n_{i,j}(t)$ segments (i.e., from $n_{i,j}(t)$ to $L_i - 1$), which are not present in the local storage, are transcoded on live by the streaming engine from the original video file cached in its local storage.

TABLE I
Key Parameters in the System Model

Notation	Definition
L_i	The number of segments for the ith video content.
$r_{i,j}$	The playback rate of the ith content in jth bitrate.
$n_{i,j}(t)$	The number of segments being cached for the ith content in the jth playback rate at the time slot t.
$F_i(k)$	The probability in which the user would watch the video up to the kth segment.
$x_{i,j,k}(t)$	The number of users watching the kth segment of the ith content in the jth playback rate at time slot t.
$\lambda_i(t)$	The number of arrivals requesting for the first segment of the ith content at time slot t.
$p_{i,j}$	The probability of the playback rate $r_{i,j}$ being requested when the user requests the ith content.
$B^S(t)$, $C^S(t)$	The storage consumption and storage cost for the M contents at the time slot t.
$B^W(t)$, $C^W(t)$	The computing consumption and computing cost for the M contents at the time slot t.
θ	The time average constraint on storage resource.
ρ	The time average constraint on computing resource.
V	Tunable parameter used to control the tradeoff between cost and queue stability.
$F(t)$	The virtual queue of the storage resource consumption.
$G(t)$	The virtual queue of the computing resource consumption.
$\Theta(t)$	The queue backlog of the virtual queues.
$\Delta(\Theta(t))$	The excepted change of queue backlog over one time slot.

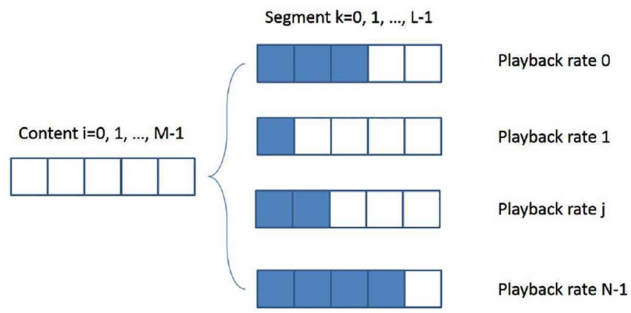


Fig. 2. Content management under the partial transcoding scheme. Shaded segments are cached, and unshaded segments are transcoded live.

Dynamic Caching Scheme: The user request rate for a video content is changing over time, which affects the computing cost for the contents transcoded online. As a result, the streaming engine needs to dynamically adjust the partial transcoding scheme in each time slot based on the user request information, to determine whether a segment should be transcoded online or locally cached to minimize the time average cost, while satisfying the resource constraints.

2) User Request Model: Most of viewers will not complete the whole video clip. Compared with the video duration, the length of a video actually watched by a viewer is rather short; specifically, 60% of videos are watched for no more than 20% of their duration [9], [10]. This user viewing pattern can be characterized by cumulative distribution function of $F_i(k)$, which denotes the probability of that the user would watch the ith video up to the kth segment. And it can be approximated by a truncated exponential distribution, resulting the following formula:

$$F_i(k) = \frac{1}{K_i}(1 - e^{-\mu k}), 0 \le k \le L_i - 1 \qquad (1)$$

where K_i is a factor for the ith content.

In practice, the user may repeat watching some parts of the video or jump over the first several minutes, which may lead to a higher request rate for the middle of a video than the beginning [22]. Since the video content are managed and streamed based on segment, all decisions for each segment can be made independently. As such, we measure the user request rate for each segment independently at the beginning of each time slot. Under the discrete time slot model, we denote $x_{i,j,k}(t)$ as the number of user requests for the kth segment of the ith content in the jth playback rate at time slot t. The user request rate for a video segment (i.e., $x_{i,j,k}(t)$) is changing over time.

3) Cost Model: There are two cost components incurred in streaming engine, including storage cost and computing cost.

Storage consumption: The storage consumption represents the storage space occupied by the video contents. Specifically, the storage consumption of the ith content is

$$B_i^S(t) = \sum_{j=0}^{N-1} n_{i,j}(t) f_i r_{i,j}, \tag{2}$$

where f_i denotes the scaling factor for video i and the file size is assumed to be proportional to its playback rate $r_{i,j}$. Thus, the total storage consumption to store all the contents is

$$B^S(t) = \sum_{i=0}^{M-1} B_i^S(t) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} n_{i,j}(t) f_i r_{i,j}. \tag{3}$$

Storage cost: It incurs storage cost in storing the cached video files. Specifically, for the ith content in the jth playback rate, the storage cost is $S_{i,j} = c_S f_i r_{i,j}$, where c_S denotes marginal price of storage space. Using this notation, the storage cost for the ith content at the time slot t is

$$C_i^S(t) = \sum_{j=0}^{N-1} n_{i,j}(t) S_{i,j}. \tag{4}$$

Thus, the total storage cost for the M contents is

$$C^S(t) = \sum_{i=0}^{M-1} C_i^S(t). \tag{5}$$

Computing consumption: The computing consumption of the ith content with playback rate of $r_{i,j}$ is $w_{i,j} = g_i r_{i,j}$, where g_i denotes the scaling factor for the ith content and the workload is assumed to be proportional to its playback rate. Using this notation, the total computing consumption for the ith content in the jth playback rate at the time slot t is

$$B_{i,j}^W(t) = \sum_{k=n_{i,j}(t)}^{L_i-1} x_{i,j,k}(t) w_{i,j}. \tag{6}$$

We assume that the computing cost of a particular video is proportional to the number of user requests. Thus, the total computing consumption for the ith content is

$$B_i^W(t) = \sum_{j=0}^{N-1} \sum_{k=n_{i,j}(t)}^{L_i-1} x_{i,j,k}(t) w_{i,j}. \tag{7}$$

The total computing consumption for the M contents is

$$B^W(t) = \sum_{i=0}^{M-1} B_i^W(t). \tag{8}$$

Computing cost: It incurs computing cost in real-time transcoding the video file into the requested playback rate. Specifically, the computing cost of transcoding the ith content into playback rate of $r_{i,j}$ is $W_{i,j} = c_W g_i r_{i,j}$, where c_W denotes marginal price of computing. Similar to the computing consumption, we can have the computing cost for the ith content, given by

$$C_i^W(t) = \sum_{j=0}^{N-1} \sum_{k=n_{i,j}(t)}^{L_i-1} x_{i,j,k}(t) W_{i,j}. \tag{9}$$

The total computing cost for the M contents is

$$C^W(t) = \sum_{i=0}^{M-1} C_i^W(t). \tag{10}$$

B. Problem Formulation

We aim to minimize the long-term operational cost for content management with the partial transcoding scheme. In a real system deployment, the storage space of content cache can be filled up quickly, if all different bitrate files of the video contents are cached to meet all the QoE requirements. In addition, if transcoding operations are conducted too frequently, it would overload the transcoding capacity and incur latency to the users. Therefore, we formulate the content management with partial transcoding scheme into the following stochastic optimization problem with time average constraints:

$$\min_{\vec{n}(t)} \quad \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{C^S(t) + C^W(t)\} \tag{11}$$

$$\text{s.t.} \quad \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{B^S(t)\} \le \theta \tag{12}$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{B^W(t)\} \le \rho \tag{13}$$

where $C^S(t)$ and $C^W(t)$ are the storage and computing cost in each time slot; $B^S(t)$ and $B^W(t)$ are the storage and computing consumption in each time slot; (11) represents the time average cost; (12) and (13) represent the time average storage and computing consumption with limited thresholds θ and ρ, respectively. With the partial transcoding scheme, we aim to derive policies in each time slot to minimize the cost over time span under the time averaged storage and computing capacity constraints.

V. Algorithms of Partial Transcoding Scheme

In this section, we derive the solution of the cost optimal transcoding problem. By taking the advantages of the Lyapunov optimization framework, we design an online algorithm which can be arbitrarily close to the optimal cost incurred over the whole time span. In particular, the Lyapunov optimization

framework allows us to minimize the cost incurred over time, while satisfying the resources consumption constraints.

A. Transformation by Lyapunov Optimization

To solve the optimization problem of minimizing the time average cost over T time slots subject to the time average constraints on storage resource and computing resource, we apply the Lyapunov optimization framework to design an online algorithm.

First, to transform the time average constraints on storage and computing resources, we define two virtual queues $F(t)$ and $G(t)$, which are bounded by F^{max} and G^{max}, respectively. Specifically, the update of these two virtual queues $F(t)$ and $G(t)$ are defined as follows:

$$F(t+1) = \max\{F(t) + B^S(t) - \theta, 0\}, \tag{14}$$

$$G(t+1) = \max\{G(t) + B^W(t) - \rho, 0\}. \tag{15}$$

Based on [23], the time average constraints of (12) and (13) can be transformed into the queue stability of the virtual queues in (14) and (15).

Then, we define the vector $\Theta(t) = (F(t), G(t))$ to represent the queue backlog and let the quadratic Lyapunov function $L(\Theta(t))$ measure the size of the queue backlog, which is defined as follows:

$$L(\Theta(t)) \triangleq \frac{1}{2}(F(t)^2 + G(t)^2). \tag{16}$$

The one-slot Lyapunov drift $\Delta(\Theta(t))$, which is the expected change of the queue backlog over one time slot, is given by

$$\Delta(\Theta(t)) \triangleq \mathbb{E}\{L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)\}. \tag{17}$$

In addition to stabilize the queue backlog to ensure the time average constraints on storage and computing resource, we also need to consider the cost given by the objective function of the original optimization problem. With the Lyapunov optimization framework, the original constrained optimization problem can be approximately solved by minimizing the drift-plus-penalty in each time slot, which jointly considers the virtual queue backlog and the incurred operational cost. Mathematically, we have the following one-slot optimization problem:

$$\min_{\vec{n}} \quad \Delta(\Theta(t)) + V\mathbb{E}\{O(t)|\Theta(t)\}. \tag{18}$$

The penalty $O(t)$ is the cost incurred at the time slot t (i.e., $O(t) = C^S(t) + C^W(t)$), and the tunable parameter V is used to control the tradeoff between the cost and the queue stability. If V approaches to infinite, the weight of the penalty function will decrease to zero, and the algorithm becomes to make policies of minimizing the cost incurred in each time slot, without considering the resource consumption. In this case, the solution derived by the online algorithm can be arbitrarily close to the optimal cost. On the contrary, if V is zero, only resource consumption is considered, and the algorithm becomes to make policies of minimizing the resource consumption, without considering the incurred operational cost.

Lemma V.1: The Lyapunov drift $\Delta(\Theta(t))$ satisfies the following inequation:

$$\Delta(\Theta(t)) \le B_1 + F(t)\mathbb{E}\{B^S(t)|\Theta(t)\} + G(t)\mathbb{E}\{B^W(t)|\Theta(t)\}$$

where $B_1 \triangleq \frac{1}{2}[(F^{max} - \theta)^2 + (G^{max} - \rho)^2]$.

Proof: Please see Appendix A for the detailed proof. □

Lemma V.1 provides the upper bound for the Lyapunov drift function (17). Under the framework of Lyapunov optimization, the strategy is to minimize the bound given on the right-hand-side of inequality in Lemma V.1 plus the penalty function in each time slot [23]. Since the constant component B_1 in the objective function will not affect the solution of the minimization problem, we take it out from the objective function. As such, we can transform the minimization of the drift-plus-penalty function (18) with the constraints into the following one-shot optimization problem:

$$\min_{\vec{n}} \quad F(t)B^S(t) + G(t)B^W(t) + V\{C^S(t) + C^W(t)\}, \tag{19}$$

$$\text{s.t.} \quad F(t) + B^S(t) - \theta \le F^{max}, \tag{20}$$

$$G(t) + B^W(t) - \rho \le G^{max}. \tag{21}$$

Equations (20) and (21) are to ensure the upper bound queue backlog of the virtual queue $F(t)$ and $G(t)$, respectively.

To this end, we have transformed the problem of minimizing the average cost over the time span (11) into a series of one-shot optimization problems (19). Following that, we propose an online algorithm for the partial transcoding scheme. In each time slot, it can be iteratively obtained by 1) observing the user request information at the beginning of each time slot, then 2) solving the one-shot optimization problem of (19) according to the current user request information and queue backlog size, and 3) updating the virtual queues $F(t)$ and $G(t)$. Our online algorithm can derive the partial transcoding solution in each time slot without requiring a prior knowledge of the user request information. The detailed online algorithm is given out in Algorithm 1.

Algorithm 1 An Online Algorithm for Partial Transcoding

Input:
Storage price C_S and transcoding price C_W.
Trade-off parameter V.
Scaling factors: f_i, g_i, $r_{i,j}$.
Output: The partial transcoding decision $m_{i,j,k}(t)$
1: Initialize $t = 0$, $F(0) = 0$, $G(0) = 0$
2: **while** the streaming engine is in service **do**
3: Observe the user request rate for each segment at the beginning of each time slot
4: Observe queue backlog $F(t)$ and $G(t)$.
5: Solve the one-shot optimization problem of (19) and derive the partial transcoding solution.
6: Perform the caching and transcoding decision for each segment according to the solution.
7: Update virtual queues $F(t)$ and $G(t)$ by (14) and (15), respectively.
8: $t \leftarrow t + 1$
9: **end while**

B. Solution of the One-Shot Optimization Problem

The one-shot optimization problem of (19) is an integer linear program which is NP-hard. We adopt Lagrange relaxation to obtain the approximate solution. First, we rewrite the above optimization problem by introducing binary variable $m_{i,j,k}(t)$ which denotes whether the kth segment of the ith video in the jth playback rate should be cached at the time slot t. Transforming the decision variable $n_{i,j}(t)$ into $m_{i,j,k}(t)$ can also make the decision for each segment independent of each other and unrelated to the order of the segments in a video content. Then, we rewrite (19), (20) and (21) as (22), (23) and (24) of the following optimization problems:

$$\min_{\vec{m}} \quad \sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\sum_{k=0}^{L_i-1}[A_{i,j,k}m_{i,j,k}(t)+B_{i,j,k}], \tag{22}$$

$$\text{s.t.} \quad \pi_1 \le 0, \tag{23}$$

$$\pi_2 \le 0,$$

$$m_{i,j,k} \in \{0,1\} \tag{24}$$

where $A_{i,j,k}$, π_1 and π_2 are parameters after rearrangement given as follows:

$$A_{i,j,k} = (F(t)+Vc_S)f_i r_{ij} - (G(t)+Vc_W)x_{i,j,k}(t)g_i r_{i,j} \tag{25}$$

$$B_{i,j,k} = (G(t)+Vc_W)x_{i,j,k}(t)g_i r_{i,j} \tag{26}$$

$$\pi_1(\vec{m}) = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\sum_{k=0}^{L_i-1} m_{i,j,k}(t)f_i r_{i,j} + F(t) - F^{max} - \theta \le 0 \tag{27}$$

$$\pi_2(\vec{m}) = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\sum_{k=0}^{L_i-1}(1-m_{i,j,k}(t))x_{i,j,k}(t)g_i r_{i,j} + G(t) - G^{max} - \rho \le 0. \tag{28}$$

We adopt Lagrange relaxation to obtain the approximate solution to the optimization problem of (22). First, we relax the constraints [(23) and (24)] by bringing them into the objective function (22) with associated Lagrange Multipliers [24]. Then, we can obtain the dual problem

$$L(\vec{\mu}) = \inf_{\vec{m}} \sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\sum_{k=0}^{L_i-1}[A'_{i,j,k}(\vec{\mu})m_{i,j,k} + \mu_1(F(t)-F^{max}-\theta) + \mu_2(G(t)-G^{max}-\rho+x_{i,j,k}(t)g_i r_{i,j})],$$

$$\text{s.t.} \quad m_{i,j,k} \in \{0,1\} \tag{29}$$

where $\vec{\mu}=(\mu_1,\mu_2)$ is the Lagrange multiplier, $\vec{m}$ is the decision vector consisting of $m_{i,j,k}$, and

$$A'_{i,j,k}(\vec{\mu}) = (F(t)+Vc_S+\mu_1)f_i r_{ij} - (G(t)+Vc_W+\mu_2)x_{i,j,k}(t)g_i r_{i,j}.$$

We can observe from (29) that, for a specified value of $\vec{\mu}$, $A'_{i,j,k}(\vec{\mu})$ becomes a constant. Then (29) can be minimized by obtaining optimal $\vec{m}$, which can be solved by setting $m_{i,j,k}$ to 0 if the corresponding $A'_{i,j,k}(\vec{\mu})$ is no less than zero, and setting $m_{i,j,k}$ to 1 otherwise. Thus, the original problem is evolved to find the optimal $\vec{\mu}$ and then solve the subproblem. We use the subgradient method to obtain the optimal value of $\vec{\mu}$ in an iterative manner. In each iteration, we have $[\pi_1(\vec{m}),\pi_2(\vec{m})]^T$ as a subgradient of $L(\vec{\mu})$, and use the following heuristic for selecting the step size [24]

$$\delta = \frac{\sigma(ub(\vec{m})-lb(\vec{m}))}{\pi_1^2(\vec{m})+\pi_2^2(\vec{m})}$$

where the current upper bound $ub(\vec{m})$ is the value of the objective function (22), and the lower bound $lb(\vec{m})$ is the value of Lagrangian function (29). The details of the subgradient method are illustrated in Algorithm 2.

Algorithm 2 Subgradient Method for One-shot Optimization

Input:
Initialize Scalar $\sigma \in (0,2)$, $\vec{\mu}=\mathbf{0}$, $s=0$,
Output:
repeat
  for all $i \in [0,M), j \in [0,N), k \in [0,L_i)$**do**
    if $A'_{i,j,k}(\vec{\mu}) >= 0$**then**
      $m_{i,j,k} \leftarrow 0$
    else
      $m_{i,j,k} \leftarrow 1$
    end if
  end for
  Update upper bound $ub(\vec{m})$ and lower bound $lb(\vec{m})$ by calculating (22) and (29).
  Update subgradient $[\pi_1(\vec{m}),\pi_2(\vec{m})]^T$ by calculating (27) and (28).
  Calculate step size:
    $\delta \leftarrow \frac{\sigma(ub(\vec{m})-lb(\vec{m}))}{\pi_1^2(\vec{m})+\pi_2^2(\vec{m})}, \vec{\mu} \leftarrow max(0, \vec{\mu}+\delta\vec{\pi})$
  $s \leftarrow s+1$
until $s = 200$

C. Optimality Analysis

The Lyapunov optimization framework allows us to design the above online algorithm which can ensure that, for any tunable parameter $V > 0$, it can approximately approach to the optimal cost within $O(1/V)$ and stabilize the virtual queue backlog, as presented in Theorem V.1.

Theorem V.1: For arbitrary user request arrival rates under system stability, the online algorithm can guarantee that the gap between the actual time average cost and the optimal time average cost is within B_1/V, i.e.

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\{C^S(t)+C^W(t)\} \le C^* + B_1/V \tag{30}$$

where C^* is the optimal time average cost.

Meanwhile, the online algorithm can stabilize the virtual queues over time, satisfying the following inequality:

$$\overline{Q} \le \frac{B_1+VC^*}{\epsilon} \tag{31}$$

where $\epsilon > 0$ and $\overline{Q}$ is the time average queue length, given as follows:

$$\overline{Q} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{G(t) + F(t)\}. \quad (32)$$

Proof: Please see Appendix B for the detailed proof. □

VI. Performance Evaluation of Partial Transcoding Scheme

In this section, we conduct experiments to evaluate the performance of our proposed partial transcoding scheme. We first describe the experimental settings. Following that, we evaluate the performance of the partial transcoding scheme under different system settings, and compare it with two alternative schemes.

A. Dataset Description and Experimental Setting

1) User Viewing Pattern: Video sessions have a very high probability to be terminated very soon, as observed in [9] that most are within 40 seconds. Similar observation was also given in [10], which indicates that the first several segments of a video content are more likely to be requested than the latter ones. We use Eq. (1) to approximately fit the data in [9], which describes the relative portions of watched videos for sessions, with K_i and μ equal to 0.98, 4.6, respectively.

2) Video Popularity Distribution: Most of videos are not frequently requested by users, as the measurements in [7] and [5] show that only 10% of the most popular videos account for almost 80% user requests, and the remaining 90% of video contents account for only 20% of the user requests. In our experiment, the user request rates for the video contents follow a power-law distribution with a shape parameter 1.2.

3) User Request Arrival Rate: The traces collected from YouTube indicate that user request rate seems relatively insensitive to video's age [10]. The same observation is also given by [25] that the videos on YouTube can keep getting views over the time span and the user request rate fluctuates around the average. In our experiment, we first perform the simulation assuming that the user requests for video content i arrive according to a random process of mean rate λ_i and variance δ_i, but without making any assumption of a priori knowledge of the probabilities associated with the user request rate. We also conduct the experiment with the real trace data given out in [26], which are the video view information collected from YouTube for 21 weeks.

4) Storage and Computing Cost: We adopt the Amazon S3 Standard Storage and Amazon EC2 On-demand Instance [27] to evaluate the storage cost and computing cost in our experiment. The storage price is \$0.03/GB/Month, and the computing price is \$0.07/Hour. The transcoding cost for a video content depends on its video duration and the targeted bitrate. If the video bitrate is less than 1 Mb/s, the transcoding cost is \$0.03 per hour (video duration); otherwise, the transcoding cost is \$0.05 per hour.

5) Video Duration and Bitrate: Most of online videos are short in terms of duration, with an average of 4.3 minutes and only less than 5% of videos last for more than 10 minutes [7]. In our experiment, for simplification, we assume that video duration has a uniform distribution from 0 to 10 minutes, and each segment has a fixed duration of 10 seconds. Meanwhile, each video content is transcoded into five versions in different bitrates. The video file in high bitrate corresponds to a high transcoding cost. The video versions and their corresponding transcoding cost are listed in Table II.

TABLE II
Bitrates and Computing Cost

Bitrate(Mb/s)	0.25	0.30	0.60	1.50	2.00
Computing Cost (\$/Hour)	0.03	0.03	0.03	0.05	0.05
Requested Probability	0.50	0.20	0.20	0.05	0.05

TABLE III
Approximation Error Under Different Dimensions

Dimension	10^2	10^3	10^4	10^5	10^6	10^7
Error Ratio(%)	0.0	2.4	2.6	2.8	2.7	2.7

B. Alternative Schemes for Comparison

We select two alternative schemes for comparison:

- *All Segments Stored (ASS)* scheme: it stores all segments in different bitrates for each content in a brute-force manner without performing transcoding tasks, which would result in a complete storage cost and no additional computing cost, since no transcoding operation is conducted.
- *Full Transcoding Without Segmentation (FTS)* scheme: it manages a tradeoff between the storage cost and the computing cost, which is similar to the partial transcoding scheme, but without file segmentation for content management. As such, the whole video file of a bitrate is either completely stored in local storage or transcoded into the user requested bitrate online.

C. Verification of the Approximate Solution of the One-Shot Optimization Problem

We compare the approximate solution of the one-shot optimization problem in Section V-B with the optimal solution to evaluate the approximation error ratio. The optimal solution can be obtained using the combination of Linear Programming (LP) Relaxation with the branch-and-bound method [28]. In this experiment, we obtain the optimal solution with CVX solver (e.g., Gurobi, MOSEK, and GLPK), which can solve the problem efficiently even when the number of decision variables is large (461 seconds for 10^7 variables[1]). We select the dimension of the variable $\vec{m}$ (i.e. segment number) from 10^2 to 10^7. As illustrated in Table III, the approximate solution is close to the optimal solution and the error ratio does not scale with the dimension of $\vec{m}$.

Verification of the Online Algorithm Optimality: In Fig. 3, we plot the time average cost and time average virtual queue length under partial transcoding scheme for different values of the control parameter V. From this figure, we can observe that with the increase of value V, the time average cost decreases and converges to the optimal value when V is large enough. However, as V increases, the time average virtual queue length also grows, demanding a tradeoff between the time average cost and system stability in making the partial transcoding decision. We can see that the result shown in Fig. 3 is consistent with Theorem V.1.

[1]Experiment configuration: CVX solver, Gurobi; Python 2.7.3; Computer Configuration, Intel i5-3470, 3.20 GHz, 8 GB RAM.

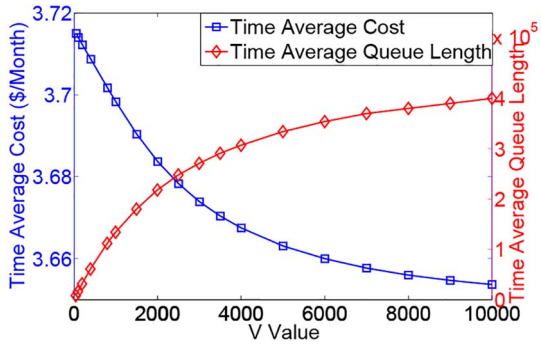


Fig. 3. Time average monetary cost and time average virtual queue length under different values of control variable V. $T = 100$.

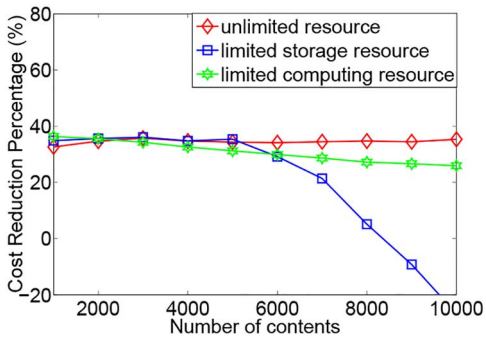


Fig. 4. Time average cost reduction percentage compared with ASS-based scheme under varying content number. $T = 100$.

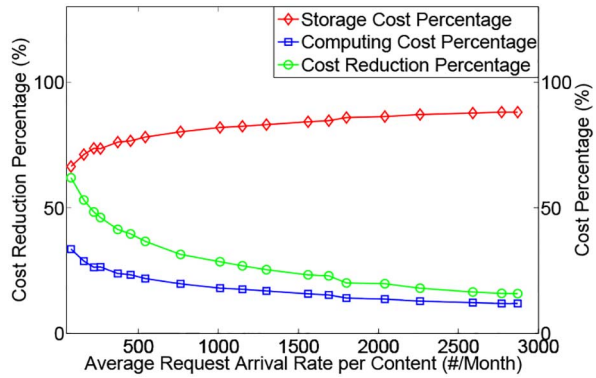


Fig. 5. Time average cost reduction percentage over ASS-based scheme under varying user request rate. $T = 100$.

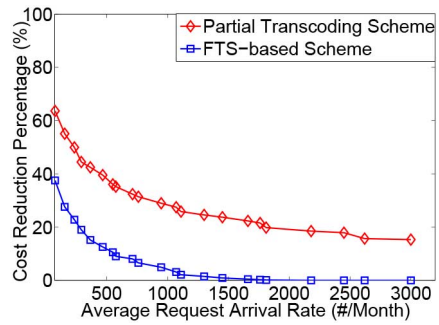


Fig. 6. Time average cost reduction percentage compared with FTS-based scheme under varying user request rate. $T = 100$.

D. Performance Comparison With Alternative Schemes

We compare the performance of the partial transcoding scheme with alternative schemes under different resource consumption constraints and varying user request rates.

1) Comparison With ASS-Based Scheme Under Different Resource Constraints: Fig. 4 demonstrates the time average cost reduction percentage of partial transcoding scheme over ASS-based scheme under different resource constraints. The partial transcoding scheme can reduce the cost for around 30% when both computing resource and storage resource are sufficient to satisfy the resource consumptions in each time slot. In another case that the computing resource is fixed, the cost reduction percentage drops slightly with the increase of video files introduced into the system, since more infrequently requested video segments need to be locally cached instead of transcoding on fly, due to insufficient computing resource. On the contrary, when the available storage resource is fixed, the cost reduction percentage drops dramatically as the number of video contents increases. Specifically, when there are around 6000 video contents, the constraint on storage consumption becomes active, indicating that the storage resource reaches its maximum utilization. When the number of contents is more than 8000, the partial transcoding scheme incurs more cost than the ASS-based scheme. This is because transcoding task needs to be conducted more frequently to satisfy the QoE requirement, under insufficient storage space for caching video segments. Therefore, one needs to strategically choose the storage capacity and computing capacity to design an effective partial transcoding scheme when planning system resource.

2) Comparison With ASS-based Scheme Under Varying User Request Rates: Fig. 5 demonstrates the time average cost reduction percentage of the partial transcoding scheme over ASS-based scheme under varying user request arrival rate. It can be seen that the partial transcoding scheme can outperform the ASS-based scheme in terms of operational cost under varying user request rate. However, the cost reduction percentage will decrease with the increase of user request rate. Particularly, with the increase of user request rate, the percentage of storage cost among the overall cost for the partial transcoding scheme increases, while the percentage of computing cost decreases. With the increase of user request rate, segments, if not locally cached, will be transcoded on fly more frequently, resulting in a higher computing cost. As a result, more segments need to be cached rather than transcoded on fly, resulting in a higher storage cost percentage and a less cost reduction percentage.

3) Comparison With FTS-Based Scheme Under Varying User Request Rate: Fig. 6 plots the time average cost reduction percentage of the partial transcoding scheme and FTS-based scheme over the ASS-based scheme. It demonstrates that compared with the FTS-based scheme, the partial transcoding scheme can save more operational cost under varying user request rates. Since the FTS-based scheme makes the caching or transcoding decision on file level, a video file needs to be wholly cached, even if only a small fraction of the file (e.g., the beginning part of a video) is

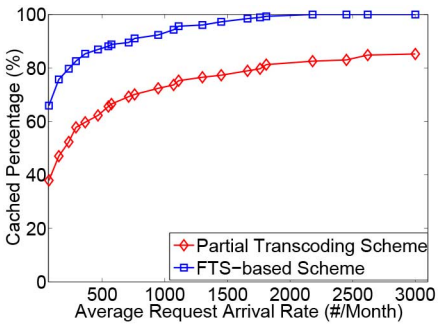


Fig. 7. Cached percentage of the partial transcoding scheme (segment) and FTS-based scheme (file) under different average user request rates. $T = 100$.

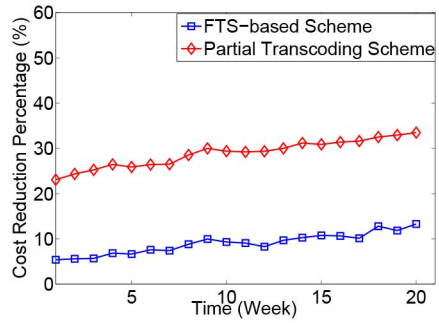


Fig. 8. Performance comparison of the three methods under real trace data.

frequently requested, resulting in more cost. In addition, it can be seen that the FTS-based scheme has a restricted cost reduction. With the increase of the user request rate, the cost reduction disappears when the user request rate approaches to 1500. In this case, all video files are cached, without any cost reduction compared with the ASS-based scheme.

4) Comparison of Segment Cached Percentage With FTS-Based Scheme: Fig. 7 plots the segment/file cached percentage of the partial transcoding scheme and FTS-based scheme, respectively. It can be seen that the cached percentage of video files under FTS-based scheme is much higher than the cached percentage of segments under partial transcoding scheme. Particularly, all the video files are cached under the FTS-based scheme when the user request rate reaches to around 1500, while in this case the partial transcoding scheme can still have a much lower cached percentage, which shows that the partial transcoding scheme is less sensitive to the change of the user request rate. Meanwhile, since less segments are cached compared with the ASS-based scheme and FTS-based scheme, the partial transcoding scheme consumes less storage resources for managing a number of video contents. As a result, the streaming engine can cache more unique video contents by leveraging the partial transcoding scheme, compared with using alternative schemes.

E. Performance Under Real Trace Data

In this subsection, we compare our proposed method with alternative schemes by using a real trace data.

The trace data [26] captures the user request information for a set of YouTube videos over 21 weeks. Since the number of views for these video contents are collected once a week, we can obtain the number of requests for a video content during one week. We evaluate the performance of each scheme in the case that the storage resource and computing resource are sufficient to achieve their best performance. We plot the cost reduction percentage of the partial transcoding scheme and FTS-based scheme over ASS-based scheme for every one week in Fig. 8. The ASS-based scheme has a constant storage cost over time, since all segments in different bitrates are all cached in the system and no transcoding operations are conducted. In contrast, the FTS-based scheme and Partial Transcoding Scheme, which make the caching and transcoding decision according to the current request rate, have a lower overall cost incurred over the time span. Compared among these three methods, our proposed method can save more operational cost in real environments.

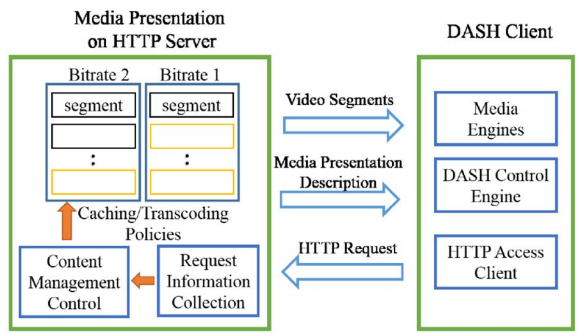


Fig. 9. System architecture for integrating the partial transcoding scheme with DASH standard.

VII. Conclusion and Future Directions

In this paper, we studied the problem of cost-efficient video content management in ABR system. We proposed a partial transcoding scheme for content management and formulated it into a stochastic optimization problem. We then applied the Lyapunov optimization framework to design an online algorithm which can derive the solution in each time slot and achieve the optimal solution within provable upper bounds. Results show that our proposed method can save around 30% of the operational cost, and can increase the number of unique contents cached in the system.

As the future work, we will consider integrating our method with the DASH standard to build a real content management system. As illustrated in Fig. 9, the system consists of two modules, including Request Information Collection (RIC) module and Content Management Control (CMC) module. The RIC is responsible for collecting the HTTP-based user request information for each segment over time; the CMC makes the caching/transcoding decisions for video segments in each time slot by obtaining the solutions of the online algorithm, according to the user request information. In this case, the system manages the video segments dynamically, supporting the DASH-based video streaming service at the minimum cost.

APPENDIX A
PROOF OF LEMMA V.1

Proof: Using the fact that $\max[q-b,0]^2 \le (q-b)^2$, we have

$$\frac{1}{2}[F(t+1)^2 - F(t)^2] \le \frac{\left(B^S(t)-\theta\right)^2}{2} + F(t)(B^S(t)-\theta),$$
$$\frac{1}{2}[G(t+1)^2 - G(t)^2] \le \frac{\left(B^W(t)-\rho\right)^2}{2} + G(t)(B^W(t)-\rho).$$

For a specified time slot t, $F(t)$ and $G(t)$ are constants and not less than zero, therefore, we can obtain

$$\begin{aligned}\Delta(\Theta(t)) \le \frac{1}{2}\{&\mathbb{E}\{(B^S(t)-\theta)^2|\Theta(t)\} \\ &+ \mathbb{E}\{(B^W(t)-\rho)^2|\Theta(t)\}\} \\ &+ F(t)\mathbb{E}\{B^S(t)|\Theta(t)\} \\ &+ G(t)\mathbb{E}\{B^W(t)|\Theta(t)\}.\end{aligned}$$

Since that $B^S(t)$ is bounded by F^{max}, $B^W(t)$ is bounded by G^{max}, we have

$$\frac{1}{2}[\mathbb{E}\{(B^S(t)-\theta)^2|\Theta(t)\} + \mathbb{E}\{(B^W(t)-\rho)^2|\Theta(t)\}] \le B_1$$

where $B_1 \triangleq \frac{1}{2}[(F^{max}-\theta)^2 + (G^{max}-\rho)^2]$.

Thus, the Lyapunov drift satisfies the following inequation:

$$\Delta(\Theta(t)) \le B_1 + F(t)\mathbb{E}\{B^S(t)|\Theta(t)\} + G(t)\mathbb{E}\{B^W(t)|\Theta(t)\}.$$

□

APPENDIX B
PROOF OF THEOREM V.1

Proof: Since we assume the time average computing and storage consumption are within the system capacity (i.e., $\mathbb{E}\{B^S(t)\} + \epsilon \le \theta$ and $\mathbb{E}\{B^W(t)\} + \epsilon \le \rho$), therefore, the drift-plus-penalty satisfies

$$\begin{aligned}&\Delta(\Theta(t)) + V\mathbb{E}\{O(t)|\Theta(t)\} \\ &\quad\le B_1 + F(t)\mathbb{E}\{(B^S(t)-\theta)|\Theta(t)\} \\ &\quad\quad + G(t)\mathbb{E}\{(B^W(t)-\rho)|\Theta(t)\} + VC^* \\ &\quad\le B_1 + VC^* - \epsilon(G(t)+F(t))\end{aligned}$$

where C^* is the target optimal time average cost and ε is a constant larger than zero. Thus, we have

$$\begin{aligned}&\mathbb{E}\{L(\Theta(t+1)) - L(\Theta(t))\} + V\mathbb{E}\{O(t)\} \\ &\quad\le B_1 + VC^* - \epsilon(G(t)+F(t)). \quad (33)\end{aligned}$$

Summing (33) over $\tau \in \{0,1,\ldots,T-1\}$, we have

$$\begin{aligned}&\mathbb{E}\{L(\Theta(T-1))\} - \mathbb{E}\{L(\Theta(0))\} + V\sum_{t=0}^{T-1}\mathbb{E}\{O(t)\} \\ &\quad\le (B_1 + VC^*)T - \epsilon\sum_{t=0}^{T-1}\{G(t)+F(t)\}. \quad (34)\end{aligned}$$

We rearrange the terms and divide Eq. (34) by T. Then, when $T \to \infty$, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\{O(t)\} \le C^* + \frac{B_1}{V}.$$

Similarly, we can obtain the upper bound on the time average queue length by taking the same rationale, which are given out as follows:

$$\epsilon\sum_{t=0}^{T-1}\mathbb{E}\{G(t)+F(t)\} \le (B_1 + VC^*)T + \mathbb{E}\{L(\Theta(0))\}. \quad (35)$$

Then, we divide (35) by T and ϵ

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\{G(t)+F(t)\} \le \frac{B_1 + VC^*}{\epsilon} + \frac{\mathbb{E}\{L(\Theta(0))\}}{T\epsilon}. \quad (36)$$

By taking a lim sup as $T \to \infty$, we can obtain

$$\overline{Q} \le \frac{B_1 + VC^*}{\epsilon}. \quad (37)$$

□

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their insightful comments.

REFERENCES

[1] "Global internet phenomena report," Sandvine, Waterloo, ON, Canada, Rep. 1H 2014, May 2014.
[2] "Cisco visual networking index: Forecast and methodology, 2012–2017" Cisco, San Jose, CA, USA, 2013.
[3] T. Stockhammer, "Dynamic adaptive streaming over http: Standards and design principles," in *Proc. 2nd Annu. ACM Conf. Multimedia Syst.*, 2011, pp. 133–144.
[4] A. Vetro and C. W. Chen, "Rate-reduction transcoding design for wireless video streaming," in *Proc. 2002 Int. Conf. Image Process.*, 2002, vol. 1, pp. 29–32.
[5] Y. Liu, F. Li, L. Guo, B. Shen, and S. Chen, "A server's perspective of internet streaming delivery to mobile devices," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 1332–1340.
[6] B. Whitelaw, "Almost all YouTube views come from just 30% of films," *The Telegraph* Apr. 2011 [Online]. Available: http://www.telegraph.co.uk/technology/news/8464418/Almost-all-YouTube-views-come-from-just-30-of-films.html
[7] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "Analyzing the video popularity characteristics of large-scale user generated content systems," *IEEE/ACM Trans. Netw.*, vol. 17, no. 5, pp. 1357–1370, Oct. 2009.
[8] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of YouTube network traffic at a campus network–measurements, models, and implications," *Comput. Netw.*, vol. 53, no. 4, pp. 501–514, 2009.
[9] A. Finamore, M. Mellia, M. M. Munafò, R. Torres, and S. G. Rao, "Youtube everywhere: Impact of device and infrastructure synergies on user experience," in *Proc. 2011 ACM SIGCOMM Conf. Internet Meas.*, 2011, pp. 345–360.
[10] L. C. Miranda, R. L. Santos, and A. H. Laender, "Characterizing video access patterns in mainstream media portals," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 1085–1092.
[11] S. S. Krishnan and R. K. Sitaraman, "Video stream quality impacts viewer behavior: Inferring causality using quasi-experimental designs," in *Proc. ACM Conf. Internet Meas.*, 2012, pp. 211–224.
[12] A. Balachandran, V. Sekar, A. Akella, and S. Seshan, "Analyzing the potential benefits of cdn augmentation strategies for internet video workloads," in *Proc. ACM Conf. Internet Meas.*, 2013, pp. 43–56.
[13] Y. Wen, X. Zhu, J. Rodrigues, and C. Chen, "Cloud mobile media: Reflections and outlook," *IEEE Trans. Multimedia*, vol. 16, no. 4, pp. 885–902, Jun. 2014.

[14] H. Hu, Y. Wen, H. Luan, T.-S. Chua, and X. Li, "Toward multiscreen social tv with geolocation-aware social sense," *IEEE Multimedia*, vol. 21, no. 3, pp. 10–19, Jul. 2014.
[15] B. Shen, S.-J. Lee, and S. Basu, "Caching strategies in transcoding-enabled proxy systems for streaming media distribution networks," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 375–386, Apr. 2004.
[16] W. Zhang, Y. Wen, Z. Chen, and A. Khisti, "QoE-driven cache management for HTTP adaptive bit rate streaming over wireless networks," *IEEE Trans. Multimedia*, vol. 15, no. 6, pp. 1431–1445, Oct. 2013.
[17] H. Ahlehagh and S. Dey, "Adaptive bit rate capable video caching and scheduling," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2013, pp. 1357–1362.
[18] Z. Huang, C. Mei, L. Li, and T. Woo, "Cloudstream: Delivering high-quality streaming videos through a cloud-based SVC proxy," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 201–205.
[19] J. Yang, H. Hu, H. Xi, and L. Hanzo, "Online buffer fullness estimation aided adaptive media playout for video streaming," *IEEE Trans. Multimedia*, vol. 13, no. 5, pp. 1141–1153, Oct. 2011.
[20] G. Gao, W. Zhang, Y. Wen, Z. Wang, W. Zhu, and Y. P. Tan, "Cost optimal video transcoding in media cloud: Insights from user viewing pattern," in *Proc. IEEE Conf. Multimedia Expo*, Jul. 2014, pp. 1–6.
[21] G. Gao, Y. Wen, W. Zhang, and H. Hu, "Cost-efficient and qos-aware content management in media cloud: Implementation and evaluation," in *Proc. 2015 IEEE Conf. Commun.*, to be published.
[22] F. T. Johnsen, T. Hafsoe, and C. Griwodz, "Analysis of server workload and client interactions in a news-on-demand streaming system," in *Proc. ISM*, 2006, vol. 6, pp. 724–727.
[23] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures Commun. Netw.*, vol. 3, no. 1, pp. 1–211, 2010.
[24] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ, USA: Prentice Hall, 1993.
[25] G. Szabo and B. A. Huberman, "Predicting the popularity of online content," *Commun. ACM*, vol. 53, no. 8, pp. 80–88, 2010.
[26] X. Cheng, C. Dale, and J. Liu, "Statistics and social network of YouTube videos," in *Proc. 16th Int. Workshop Quality Service*, Jun. 2008, pp. 229–238.
[27] H. Hu, Y. Wen, T.-S. Chua, and X. Li, "Toward scalable systems for big data analytics: A technology tutorial," *IEEE Access*, vol. 2, pp. 652–687, 2014.
[28] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. New York, NY, USA: Wiley, 1990.

Guanyu Gao received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2009, the M.S. degree from the University of Science and Technology of China, Hefei, China, in 2012, and is currently working toward the Ph.D. degree at the Interdisciplinary Graduate School, Nanyang Technological University, Singapore.

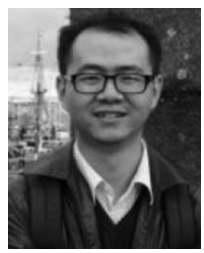
Weiwen Zhang received the B.S. degree in software engineering and the M.S. degree in computer science from the South China University of Technology, Guangzhou, China, in 2008 and 2011, respectively, and the Ph.D. degree in computer engineering from Nanyang Technological University, Singapore in 2015.

He was a Visiting Scholar with Purdue University, West Lafayette, IN, USA, in 2014. His current research interests include cloud computing, mobile computing, and green data center.

Yonggang Wen (S'00–M'07–SM'14) received the Ph.D. degree in electrical engineering and computer science (minor in western literature) from the Massachusetts Institute of Technology, Cambridge, MA, USA.

He is an Assistant Professor with the School of Computer Engineering, Nanyang Technological University, Singapore. He previously worked with Cisco, San Jose, CA, USA, where he lead product development in content delivery networks. He has authored or coauthored over 90 papers in top journals and prestigious conferences. His current research interests include cloud computing, green data center, big data analytics, multimedia networks, and mobile computing.

Dr. Wen serves on editorial boards for the IEEE TRANSACTIONS ON MULTIMEDIA, IEEE ACCESS, and Elsevier *Ad Hoc Networks*. He was the recipient of an ASEAN ICT Award (Gold Medal) in 2013 and an IEEE Globecom 2013 Best Paper Award.



Zhi Wang received the B.E. and Ph.D. degrees in computer science from Tsinghua University, Beijing, China, in 2008 and 2014, respectively.

He is currently an Assistant Professor with the Graduate School at Shenzhen, Tsinghua University, Shenzhen, China. His current research interests include online social networks, mobile cloud computing, and big-data systems.

Dr. Wang was the recipient of the Best Paper Award at ACM Multimedia 2012.

Wenwu Zhu (S'91–M'96–SM'01–F'10) received the Ph.D. degree in electrical and computer engineering from the New York University Polytechnic School of Engineering, New York, NY, USA, in 1996.

He is currently a Professor with the Computer Science Department of Tsinghua University, Beijing, China. His current research interests include multimedia cloud computing, social media computing, multimedia big data, and multimedia communications and networking.

Dr. Zhu has served on various editorial boards, including serving as Leading Editor of Computer Networks and Distributed Computing for the *Journal of Computer Science and Technology*; Guest Editor of the PROCEEDINGS OF THE IEEE, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS; and Associate Editor for the IEEE TRANSACTIONS ON MOBILE COMPUTING, the IEEE TRANSACTIONS ON MULTIMEDIA, and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY. He was the recipient of the Best Paper Award at ACM Multimedia 2012, the Best Paper Award for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY in 2001, and four other international Best Paper Awards.