This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

QoS-aware Resource Allocation for Video Transcoding in Clouds

Lei Wei, Jianfei Cai, *Member, IEEE,* Chuan Heng Foh, *Member, IEEE,* and Bingsheng He, *Member, IEEE*

***Abstract*—As the "biggest big data", video data streaming in the network contributes the largest portion of global traffic nowadays and in future. Due to heterogeneous mobile devices, networks and user preferences, the demands of transcoding source videos into different versions have been increased significantly. However, video transcoding is a time-consuming task and how to guarantee quality-of-service (QoS) for large video data is very challenging, particularly for those real-time applications which hold strict delay requirement such as live TV. In this paper, we propose a cloud-based online video transcoding system (COVT) aiming to offer economical and QoS guaranteed solution for online large-volume video transcoding. COVT utilizes performance profiling technique to obtain the different performance of transcoding tasks in different infrastructures. Based on the profiles, we model the cloud-based transcoding system as a queue and derive the QoS values of the system based on queuing theory. With the analytically derived relationship between QoS values and the number of CPU cores required for transcoding workloads, COVT is able to solve the optimization problem and obtain the minimum resource reservation for specific QoS constraints. A task scheduling algorithm is further developed to dynamically adjust the resource reservation and schedule the tasks so as to guarantee the QoS in runtime. We implement a prototype system of COVT and experimentally study the performance on real-world workloads. Experimental results show that COVT effectively provisions minimum number of resources for predefined QoS. To validate the effectiveness of our proposed method under large scale video data, we further perform simulation evaluation which again shows that COVT is capable to achieve economical and QoS-aware video transcoding in cloud environment.**

***Index Terms*-Video transcoding, cloud computing, resource allocation**

I. Introduction

With the explosive growth of the demands for online video streaming services [1], video service providers face significant management problems on the network infrastructure and computing resources. As reported in [1], the world-wide video streaming traffic will occupy approximately 69% of the total global network traffic in 2017. Therefore, the video data is becoming the "biggest" big data that contributes to a huge amount of IT investments such as networking, storage and computing. Besides, online real-time video streaming services such as online conferencing [2], live TV and video chat have been growing rapidly as the most important multimedia applications. In this paper, we refer the transcoding services that transcode live video data from source within a short delay as *online video transcoding*.

With the rapid growth of mobile market, increasing volumes of online videos are consumed by mobile devices. As a result, service providers often need to transcode the video contents into different video specifications (e.g., bit rate, resolution, quality, etc) with different QoS (e.g., delay, etc) for heterogeneous mobile devices, networks and user preferences. However, video transcoding [3], [4] is a time-consuming task, and how to guarantee acceptable QoS for large video data transcoding is very challenging, particularly for those real-time applications which hold strict delay requirement.

Cloud computing technology [5] holds many advantages on offering elastic and economical computing resource for online video applications. Compared to video service providers who invest on their own IT infrastructures, cloud-based video transcoding and streaming services are able to benefit from on-demand resource reservation, simpler system maintenance and lower investments. For service providers using their own data centers, they have to build up an infrastructure that satisfies QoS at the peak load. Such over-provisioning of resources is highly inefficient in terms of cost. In contrast, cloud-based transcoding systems only need to consider current workload amount and reserve suitable resources to offer predefined QoS.

Online transcoding in clouds brings new challenges. First of all, the key problem is that online video applications have strict delay requirement, which includes both transcoding delay and streaming delay. The streaming delay is mostly determined by the targeted transcoded video size, which is also determined in the transcoding phase. The second challenge is the resource reservation strategy that balances resource cost and QoS. If the reserved resource is less than demand, the video transcoding process in clouds will take long time, and thus the delay of video playback would be high. On the other hand, if provisioning too much resource, the unused resource is wasted. The third issue is brought by the heterogeneity of infrastructures. The transcoding time of video chunks is different on different servers. Thus, the hardware heterogeneity is an important factor that should be considered.

In this paper, we propose a cloud-based online video transcoding system (COVT) to handle the above challenges. COVT focuses on resource provisioning and task scheduling in order to provide economical and QoS guaranteed cloud-based video transcoding. Our research goal is to minimize the amount of resource (in terms of the number of CPU cores) for online video transcoding tasks given specific QoS constraints. In particular, we consider two QoS parameters: *system delay* and *targeted chunk size*. The system delay is

L. Wei, J.F. Cai, B.S. He are with the School of Computer Engineering, Nanyang Technological University, Singapore. E-mail: {weil0008, ASJFCai, BSHE }@ntu.edu.sg.

C.H. Foh is with the Centre for Communication Systems Research, University of Surrey, Guildford, Surrey, UK. E-mail: c.foh@surrey.ac.uk.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

defined as the time from the arrival of a video chunk to the completion of the transcoding, which consists of queuing time and transcoding time. The targeted chunk size is the average *file size* of output video chunks, which is the key indicator for streaming overhead. Here, video chunks are the segmented video clips with an equal playback length as video chunks.

Specifically, we allow different transcoding modes, which gives great flexibility to trade off between transcoding time and output chunk size. Faster modes produce larger output chunk size with short processing time, but slower modes use more transcoding time to get smaller chunk size. To facilitate the selection of optimal transcoding modes with different hardware and different video content, COVT performs performance profiling for each stream using history data. Based on the profiles, COVT designs a prediction model to find the optimal probability distribution of different transcoding modes while minimizing the required number of CPU cores. In the scheduling phase, COVT distributes the video transcoding tasks into the cloud cluster with a QoS guaranteed scheduling algorithm that dynamically reserves or releases CPU cores and determines the transcoding mode according to runtime QoS measurements.

The main contributions of this paper are threefold. Firstly, we propose a cloud-based platform for video transcoding of live streams with strict QoS and economical resource usage. Secondly, we model the resource provisioning problems with a queueing model that derives the relationship between the probability distribution of transcoding modes and resource usage with the QoS constraint. The model is able to effectively predict the required number of CPU cores for video streams as well as the transcoding mode distribution. Thirdly, we develop a runtime scheduling algorithm and adjust the resource reservation according to the runtime QoS measurements in order to avoid QoS broken due to burst workloads or prediction errors. We implement a prototype system of COVT to validate the effectiveness with real-world data set. The experimental results show that our proposed system effectively provisions suitable number of resources under predefined QoS constraints, which allows video service providers to offer high-quality services with minimum costs. Besides, we also perform simulation studies to validate the scalability of COVT under large scale data set.

The rest of this paper is organized as follows. We review the related work in Section II. Then we describe our system and give an overview in Section III. In Section IV, we introduce the performance profiling for video streams, followed by the proposed resource prediction model with QoS constraints in Section V. In Section VI, the task scheduling in COVT is introduced. The prototype system design and the experimental results are given in Section VII. Then, we simulate the system and evaluate its performance under large-scale data set in Section VIII. We conclude the paper in Section IX.

II. Related Work

Resource management in clouds is a typical and crucial topic and a number of works have been published [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17]. These works show their advantages on specific aspects of cloud resource management such as heuristic bin-packing of virtual machine (VM) placement in data centers [6], [7], balancing the cost and the deadline of jobs in clouds [8], [9], [10], [11], handling bursting workloads [12], dynamic cluster resizing [13], [15], [16], fairness between tenants in private clouds [17] and resource provisioning for heterogeneous workloads [14], [18]. All these works are designed for general workloads in clouds but not covering the unique system demands of video transcoding services. As introduced previously, online video transcoding is QoS-sensitive and time-consuming. To achieve stable and smooth service to video consumers, this paper proposes a cloud-based video transcoding system to guarantee the restrict QoS requirements for online video applications.

A number of attempts have been made on the problem of video transcoding in clouds [19], [20], [21], [22], [23], [24], [25], [2], [26], [27]. Garcia et al. [24], [25] and Kim et al. [19] proposed to use MapReduce parallel computing tools (e.g., Hadoop) to speed up the video transcoding process. These works make effort to enhance the performance of off-line video transcoding based on MapReduce or Hadoop. MapReduce is a general framework that splits a big task, dispatches to multiple virtual machines (VMs) and collects the results back. It has no support for QoS and it does not dynamically adjust computing resource. Since it is designed for general purpose, the overhead would be large and the QoS support for online video transcoding would be weak. MapReduce based methods are more suitable for offline video transcoding, where all the date are stored in local storages, but they are not appropriate for our considered QoS-aware online video transcoding scenario.

Pereira et al. [23] designed an architecture for processing videos in clouds including merge&split operations. Similarly, Zhuang et al. [22] also designed an architecture for video transcoding in content delivery networks. Ashraf et al. [20], [21] studied the admission control problem for video streams to prevent blockage of services and the cost-efficient resource provisioning problem for transcoding tasks in clouds. Jokhio et al. [26] studied the basic dynamic allocation and release of VMs and the decision making on whether performing transcoding tasks in advance so as to avoid excess storage on cloud servers. There are some recent relevant works including [28], [29] and [30] which proposed to adaptively transcode the videos according to cost or user preferences. The videos with high storage cost or low user preference can be processed with a transcode-on-request manner which only transcodes the videos when there are requests for them. Similarly, the videos with low storage costs and high preference can be transcoded into multiple versions in advance to improve both streaming quality and cost. These works addressed the task scheduling problem for local data with both online and offline transcoding, while we mainly focus on live video streams with online transcoding. Our effort is on how to request minimum cloud computing resource while still meeting the QoS requirements of video transcoding. Our work can be complement to [28], [29], [30], by outsourcing their online transcoding part from local servers to the cloud.

To the best of our knowledge, very few research has

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

studied the problem of provisioning the minimum resources while satisfying restrict QoS for cloud-based online video transcoding. One most related work is [31] where Zhang et al. designed an energy-efficient job dispatching algorithm in transcoding-as-a-service cloud. The video transcoding is viewed as services provided by transcoding engines in the clouds. As each video transcoding task consumes a portion of energy in cloud servers, the authors try to minimize the total energy consumption by intelligently dispatching transcoding jobs to service engines. Meanwhile, maintaining low delay is also a significant constraint factor. However, there is no real experimental evaluation of the proposed method. Since some assumptions like that the energy consumption of data center is determined by CPU speed might not be completely correct, the practical energy consumption may differ as predicted. Besides, the impact of workload dynamicity is not considered, which also has high chance to break the QoS for online transcoding. Thus, in this paper, we design a system that fully considers the feature of cloud environment by adopting infrastructure-aware performance profiling and dynamic task scheduling to guarantee the QoS. We also implement a prototype to validate our system design.

Performance profiling methodology is common in evaluating the cloud computing performance on general workloads [32], [33], [34], [35]. Nevertheless, the performance requirements in an online video transcoding system are significantly different from the previous works that mainly target at Mapreduce framework, lack of online QoS guarantee. Thus, the performance profiling method in our system COVT fully considers the unique system configurations of video transcoding tasks.

III. System Architecture

In this section, we introduce the system architecture and provide an overview of COVT. For better explanation, all the important notations and parameters used throughout this paper are listed in Table I.

Fig. 1 illustrates the system architecture of COVT which consists of three components: video consumer, video service provider and cloud cluster. Generally, video consumers request their favored videos from the service provider who is responsible to stream the transcoded video contents to consumers. The service provider reserves and manages computing resources from clouds to comprise a transcoding cluster. The cloud cluster consists of a number of VMs that transcodes the source videos into targeted videos with a certain video specification (including format, resolution, quality, etc) with some QoS constraints. The service provider is charged according to the amount of resource reserved in clouds. The detailed description of the three components in COVT is discussed as follows.

A. Video consumer

The source videos (or workloads) to be transcoded and forwarded to customers are a number of streams of video data, each of which is partitioned into video chunks with L seconds playback length. The video consumer includes all kinds of devices such as personal computers, mobile phones, tablets and televisions that request video contents from the video service provider. For different terminals, the desirable videos in terms of data rate, resolution, format are different due to heterogeneous network bandwidth, hardware ability and software functions. The delay tolerance of the video service is also different for different applications. For example, the online TV permits a relatively long delay of several minutes, but the delay for delay-sensitive applications such as online conferencing should usually be less than one or two seconds. Breaking the delay tolerance will result in poor playback quality in customers' devices.

We use two aspects of delay as the QoS constraints in COVT, namely *system delay (d)* and *targeted chunk size (s)*. The system delay is defined as the time from the arrival of a video chunk to the completion of transcoding, which consists of queuing time and transcoding time. The targeted chunk size is the average size of output video chunks which is the key indicator for streaming time in networks (although video streaming is not a main concern in this paper). We set two thresholds for system delay d and targeted video chunk size

TABLE I
Notations Used in the Transcoding System

K	Number of video streams
L	Length of video chunks (seconds)
V	Number of video types
$\vec{B}$	$\vec{B} = \{b_v \mid v = 1, 2, ..., V\}$, b_v is the proportion of v^{th} video types and $\sum_{v=1}^{V} b_v = 1$
M	Number of transcoding modes
$\mathbf{T}$	Array of profiles for average transcoding time, $\mathbf{T} = \{t_v^m \mid, m = 1, 2, ..., M, v = 1, 2, ..., V\}$, t_v^m is the average transcoding time of video type v using m^{th} transcoding mode
$\mathbf{W}$	Array of profiles for average targeted video chunks size, $\mathbf{W} = \{w_v^m \mid, m = 1, 2, ..., M, v = 1, 2, ..., V\}$, w_v^m is the average size of video type v using m^{th} transcoding mode
$\vec{P}$	$\vec{P} = \{p_m \mid m = 1, 2, ..., M\}$, p_m indicates the probability that the system should use m^{th} mode, $\sum p_i = 1$
$\vec{O}$	$\vec{O} = \{o_m \mid m = 1, 2, ..., M\}$, o_m indicates the estimated proportions of video chunks using m^{th} mode, $\sum o_i = 1$
N	Predicted number of CPU cores by probabilistic model
n	The reserved CPU cores in clouds
u	The actual number of CPU cores used in system, $u \leq n$
D_{max}	Constraint of average delay
d	Estimated average delay in system
S_{max}	Constraint of average size for targeted video chunks
s	Estimated average size of targeted video chunks in system
c	The number of CPU cores in the smallest VM
τ	The discretizing gap of the probabilities of transcoding modes

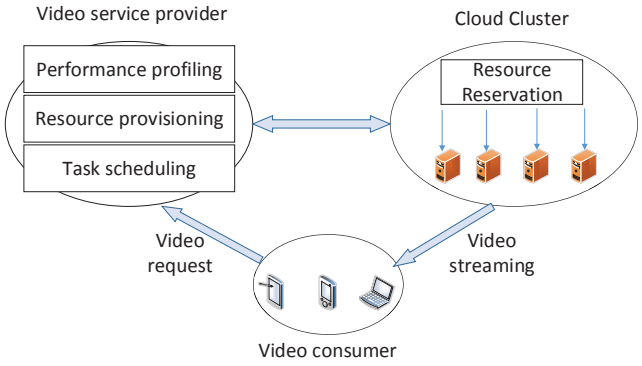


Fig. 1. System architecture of COVT.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

s as the QoS constraints that the system should comply with, denoted as D_{max} and S_{max}, respectively. The values of D_{max} and S_{max} are determined by the service provider according to the practical requirements of different applications.

B. Video service provider

On one hand, the video service provider is responsible for streaming the required targeted video contents to video consumers by reserving sufficient resources in clouds. On the other hand, the service provider tries to seek an economical solution for the transcoding system in order to save monetary costs. Thus, the service provider needs to find the optimal point in the trade off between costs and QoS. With these design goals, we introduce the system modules in the service provider including performance profiling, resource prediction and task scheduling as follows:

- We define the transcoding for a video chunk with L seconds playback length as a task in COVT. The performance profiling is a common way to obtain the performance of tasks in terms of transcoding time and targeted chunk size, which is important to guide the resource reservation and the task scheduling. The performance profiling module considers to record the transcoding time and the targeted chunk sizes of different video types in the past with different transcoding modes under specific hardware. The concept of transcoding mode is a configuration that controls the compression ratio of the output video in video transcoding process. There are usually several transcoding modes that can be used for different system requirements. A faster mode means shorter transcoding time but a lower compression ratio or a larger chunk size. In contrast, a slower mode produces a smaller targeted chunk size with longer transcoding time. With the profiles, COVT is able to determine the suitable distribution of different transcoding modes for workloads and further reserve appropriate number of resources for given QoS. The details of the profiling method is given in Section IV.
- Resource provisioning is used to predict the number of resources that is needed for the workloads given predefined QoS constraints D_{max} and S_{max}. The resource provisioning in COVT is a general method that is feasible for different resource types. In this paper, we use the number of CPU cores to be the units in resource provisioning. Other resource types (e.g., GPU) can be supported with specific profiling data for the considered resource types.

  In resource prediction, we model the transcoding system in COVT as an $M/G/N$ queue with Poisson arrivals of video chunks produced from the video source. The service rates are determined by the profiles from the performance profiling module. By solving the queuing model, the QoS values d and s can be computed given the number of CPU cores N and the distribution of transcoding modes $\vec{P} = \{p_m | m = 1, 2, ..., M\}$ ($\sum_{m=1}^{M} p_m = 1$). Then, it is feasible to find the minimum number of CPU cores by enumerating different transcoding modes. The detailed modeling process is introduced in Section V.
- Task scheduling module is responsible for distributing the large number of video chunks into the cloud cluster for transcoding processing. The scheduling policy is based on the transcoding modes distribution $\vec{P}$ generated in the prediction model. Our basic idea is to use slower transcoding modes as much as possible as long as the system delay d meets the QoS constraint. Let $\vec{O} = \{o_m | m = 1, 2, ..., M\}$ be the estimated value for $\vec{P}$ in the scheduling phase. If the estimated proportion of the slowest mode o_1 is less than a threshold, we increase the resource reservation for the subsequent time periods. On the other hand, if o_1 is greater than a threshold, we decrease the number of resource because there is space for optimizing. In this way, we are able to accommodate the mismatch between the prediction and the actual situations so as to minimize the cloud resource and guarantee the QoS constraints. The detailed scheduling algorithm will be discussed in Section VI.

C. Cloud cluster

Cloud cluster includes several working nodes (VMs) leased from the clouds which are responsible for transcoding the video chunks despatched to them and forwarding the targeted video chunks to video consumers in parallel. The service provider periodically reserves the resources from clouds according to the provisioning scheme obtained from the prediction model in Section V for given QoS constraints. In the runtime of the transcoding system, the service provider adjusts the reserved number of resources in the clouds with the scheduling algorithm discussed in Section VI according to the instant states of the system. It is common that the predicted amount of resources mismatches the preset QoS constraints in the runtime, which will be well compensated by the scheduling algorithm. By such manner, COVT is able to strictly guarantee the preset QoS constraints for online video transcoding services.

Note that we use the number of CPU cores as the units for resource provisioning in clouds without loss of generality. For a given number of CPU cores N predicted by the model, how to reserve VMs from clouds (e.g., whether to lease two 4-core VMs or four 2-core VMs for the transcoding tasks requiring eight CPU cores) is determined by the specific pricing model in clouds, which is not in the scope of this paper.

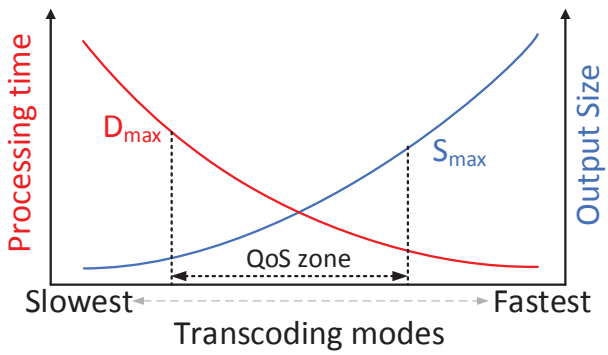


Fig. 2. The relationship between transcoding modes and QoS.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

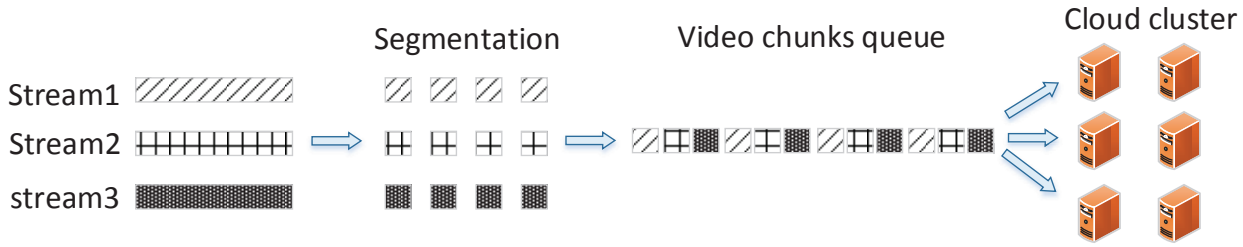


Fig. 3. The queuing model in COVT.

IV. Performance Profiling

In this section, we introduce the performance profiling for the video transcoding system aiming to assist the resource prediction for a targeted video specification (format, resolution and quality). As we discussed in Section III-B, in video transcoding, it is possible to produce different targeted video chunks with different sizes by using different transcoding modes. The design of different transcoding modes allows a flexible trade off between transcoding delay and targeted chunk size.

Generally, COVT recognizes M different transcoding modes (e.g., slow, medium, fast, ...) and V different video types (e.g., movie, news, sports, ...). We denote $\mathbf{T} = \{t_v^m | m = 1, 2, ..., M,\ v = 1, 2, ..., V\}$ and $\mathbf{W} = \{w_v^m | m = 1, 2, ..., M,\ v = 1, 2, ..., V\}$ to be the average transcoding time and the average output size of video chunks using the m^{th} transcoding mode for the v^{th} video type. We run all combinations of transcoding modes and video types to record the average transcoding time and output size of the history data (recent several hours or days). Then, the profiles obtained in the profiling ($\mathbf{T}$ and $\mathbf{W}$) are used as the input parameters for the prediction model.

Fig. 2 illustrates the relationship between the transcoding time and the output size with different transcoding modes (from slowest to fastest). We can see that the average processing time decreases as the transcoding mode varies from the slowest to the fastest, but the average output chunk size grows with the faster transcoding mode. Thus, there is a trade off between the processing time of transcoding tasks and the output video size. Since the system delay consists of transcoding time and queuing time, the transcoding time also contributes to the system delay. Therefore, the overall transcoding mode distribution needs to be located in an area where the conditions of $d \leq D_{max}$ and $s \leq S_{max}$ are satisfied. In next section, we discuss how to predict the minimum number of CPU cores that meet the conditions of QoS.

V. Resource Prediction Model

In this section, we introduce the prediction model for the given QoS requirement D_{max} and S_{max}. We formulate the problem using a queuing model and then develop an approximate solution for the proposed model.

A. Queuing model

In COVT, all the video chunks of the K video streams generated from the video source are maintained in a queue as shown in Fig. 3. We consider V video types and each video chunk belongs to one video type $v, v = 1, 2, ..., V$, with a playback length of L seconds. We partition the operating time of the system into multiple time slots and the resource prediction is performed at the beginning of each time slot. In this section, we focus on the resource prediction for one single time slot. The two QoS parameters of the system are denoted by d and s for the average system delay and the output chunk size, respectively. The goal of the resource prediction model is to provision the minimum N that meets the QoS requirements of $d \leq D_{max}$ and $s \leq S_{max}$. The distribution vector of transcoding modes $\vec{P}$ is also determined by the model when obtaining the minimum N.

We model the video transcoding process in the system as an *M/G/N* queue. Let l be the queue length that evolves with a video chunk arrival from a stream or a video transcoding task completion. A video chunk arrival at the queue increases the queue length by one, and the completion of processing a chunk decreases the queue length by one. The arrival rates follow Poisson distribution with an average value λ which is determined by the video generation speed in the video source and the number of streams. The service rates of the queuing system μ follow general distribution which are generated from the profiles obtained in the profiling module. Note that there are N CPU cores in system working in parallel, which means that the overall service rate in the model is μN.

In the queuing model, we study the relationship between the QoS values and different system settings including the transcoding mode distribution ($\vec{P}$), number of CPU cores (N) and the performance profiles ($\mathbf{T}$ and $\mathbf{W}$) that we have obtained from the profiling module. Denoting f and g as the functions for QoS values d and s, respectively. We formulate the provisioning issue of COVT as the following optimization problem:

$$\min_{\vec{P}} \ N \tag{1}$$

$$s.t. \quad d \leq D_{max} \tag{2}$$

$$s \leq S_{max} \tag{3}$$

$$d = f(N, \vec{P}, \mathbf{T}) \tag{4}$$

$$s = g(\vec{P}, \mathbf{W}) \tag{5}$$

To solve the above problem, we must first derive the functions f and g. The derivation of function g is simpler than function f because g has no relation with the resource amount N.

Markov Chain is a common technique to solve queuing problems, but it is not suitable here. In Markov Chain, states

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

are memoryless so that the transition from one state to another is independent from the other states. It means that inter-arrival and service time should be both exponentially distributed. However, the service time here follows general distribution in the $M/G/N$ queue in COVT. Thus, we cannot use Markov Chain to solve the model.

B. Solution

We observe that the delay of video chunks in the system can be divided into two parts: the waiting time in queue and the transcoding time in the cloud cluster, denoted as D_q and D_t, respectively. Thus, the average system delay of COVT d can be expressed as

$$E[d] \ = \ E[f(N, \ \vec{P}, \ \mathbf{T})] = E[D_q] + E[D_t], \tag{6}$$

where function E means the expectation function. Since $E[D_t]$ is just the average transcoding time of video chunks, we can get it through the profiles of transcoding time $\mathbf{T}$. Let μ be the average service (transcoding) rate of the queuing model which can be calculated by

$$\mu \ = \frac{1}{\sum_{m=1}^{M} p_m \cdot \sum_{v=1}^{V} b_v \cdot t_v^m}, \tag{7}$$

where $\vec{B} = \{b_v | v = 1, 2, ..., V\}$ is the proportion of different video types and $\vec{P} = \{p_m | m = 1, 2, ..., M\}$ is the probability distribution of transcoding modes. The average service rate is calculated by one over the average transcoding time of a video chunk in one CPU core. The average transcoding time is computed over different video types and different transcoding modes. Thus, the overall service rate of the cluster is μN since there are N CPU cores in the system. Accordingly, the average processing time of a video chunk in the system with N CPU cores is given by $\frac{1}{\mu N}$. Then Eq. (6) can be written as

$$E[d] \ = \ E[D_q] + \frac{1}{\mu N}. \tag{8}$$

The queuing delay D_q also consists of two parts, the remaining processing time of current transcoding tasks in the cloud cluster and the sum of the transcoding time of all the chunks in the queue, i.e.

$$E[D_q] \ = \ E[R] + \frac{E[l]}{\mu N}, \tag{9}$$

where $E[R]$ stands for the remaining processing time of video chunks in the cloud cluster and $E[l]$ is the average queue length. With Little's formula

$$E[l] \ = \ \lambda E[D_q], \tag{10}$$

we obtain

$$E[D_q] \ = \ \frac{E[R]}{1 - \frac{\rho}{N}}, \tag{11}$$

where $\rho = \frac{\lambda}{\mu}$ is defined for convenience of expression. Therefore, Eq. (8) becomes

$$E[d] \ = \ \frac{E[R]}{1 - \frac{\rho}{N}} + \frac{1}{\mu N}. \tag{12}$$

Now, the issue is to derive $E[R]$. Since the remaining processing time of video chunks in COVT does not follow the exponential distribution and the memoryless property, we derive it from the beginning by considering all the tasks (chunks) in the cloud cluster.

Considering a long time interval $[0, Z]$, we denote $\Gamma(z), z \in [0, Z]$ to be the remaining processing time of video chunks in the cloud cluster at time z. Then we can calculate $E[R]$ by

$$E[R] \ = \ \frac{1}{Z} \int_0^Z \Gamma(z) dz. \tag{13}$$

Assume there are totally $I(Z)$ tasks arriving at the system in time interval $[0, Z]$. Then, let Y_i be the processing time of the $i^{th}, i = 1, 2, ..., I(Z)$, transcoding task. To illustrate the processing time function $\Gamma(z)$ with the discrete video chunk arrivals, we show the evolving process in Fig. 4. As shown in the figure, the reminding processing time of $\Gamma(z)$ is equal to zero when there is no task in the cloud and set to Y_i as the task commences. Then the value of $\Gamma(z)$ decreases linearly with rate 1 till the task completion. It implies that the integration part in Eq. (13) is the sum of areas of all triangles under the curve $\Gamma(z)$, where the sides and heights of the triangles are both Y_i. Thus, for large Z, we derive

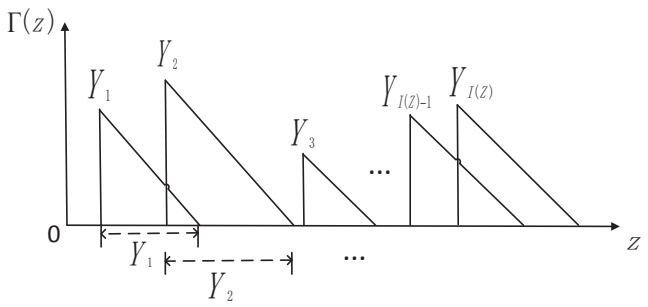


Fig. 4. Processing time of tasks.

$$E[R] \ = \ \frac{1}{Z} \int_0^Z \Gamma(z) dz \tag{14}$$

$$= \ \frac{1}{Z} \sum_{i=1}^{I(Z)} \frac{1}{2} (Y_i)^2 \tag{15}$$

$$= \ \frac{I(Z)}{2Z} \cdot \frac{1}{I(Z)} \sum_{i=1}^{I(Z)} Y_i^2 \tag{16}$$

$$= \ \frac{1}{2} \lambda \overline{Y^2}, \tag{17}$$

where $\overline{Y^2}$ is the second moment of the processing time Y_i. With the relationship between variance and second moment

$$\sigma^2 \ = \overline{Y^2} - \frac{1}{(\mu N)^2}, \tag{18}$$

where the σ^2 is the variance of Y_i, we obtain

$$E[R] \ = \ \frac{1}{2} \lambda (\sigma^2 + \frac{1}{(\mu N)^2}), \tag{19}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

where

$$\sigma^2 = \sum_{m=1}^{M} p_m \sum_{v=1}^{V} (\frac{b_v t_v^m}{N} - \frac{1}{(\mu N)})^2. \quad (20)$$

Finally, with Eq. (12) and Eq. (19), we derive the formula for the system delay $E[d]$ as

$$E[d] = \frac{N^2\lambda^2\sigma^2 + \rho^2}{2\lambda N(N-\rho)} + \frac{1}{\mu N}. \quad (21)$$

The average targeted output size s is given by

$$E[s] = \sum_{m=1}^{M} p_m \sum_{v=1}^{V} b_v \cdot s_v^m. \quad (22)$$

After we derive the models for the QoS parameters, we are able to find the optimal resource reservation (N) with respect to the transcoding mode distribution ($\vec{P}$) as shown in Eq. (1). However, it is difficult to solve the problem with a close-form solution since there are multiple unknown variables in $\vec{P}$. We seek an approximation solution for the optimization problem in Eq. (1), which is presented in Algorithm 1. In particular, considering the value of N must be an integer, we enumerate N starting from 1. For each p_m, we discretize the probability proportion by a gap $\tau, \tau < 1$, so that $p_m \in [0, \tau, 2\tau, ..., 1]$. In searching for the solution, we use the rule of selecting as many slower modes as possible as long as the QoS constraints are satisfied. The benefit of such rule is to reduce the chunk size if the delay constraint is met.

The complexity of Algorithm 1 is $O(N \cdot M^2/\tau)$, where N is provisioned number of CPU cores and M is the number of transcoding modes. Since N increases with the increased number of streams and M is a small positive integer, the complexity of the algorithm is quite low. Besides, the complexity is inversely proportional to the discretizing gap τ, for which we use a default value of 0.05 in our experiments.

VI. Task Scheduling

To ensure the QoS in the runtime, we develop a task scheduling algorithm that dispatches the tasks in the system queue to the cloud cluster based on the prediction of resource usage and transcoding mode distribution. It is inevitable that some mismatch exists between the predicted resource usage and the practical situation in the runtime due to dynamic workloads in the cloud-based transcoding system. Therefore, it is necessary to monitor and manage the QoS with dynamic adjustment in the task scheduling phase.

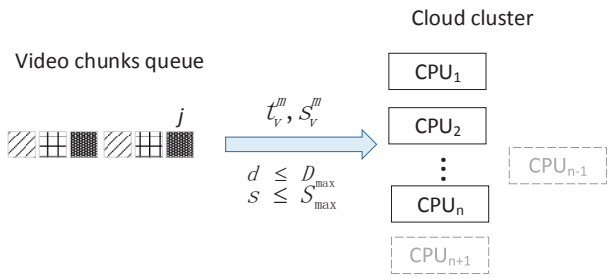


Fig. 5. Illustration of video transcoding tasks scheduling.

Algorithm 1 Resource prediction in a time slot

Input:
- $\mathbf{T}$: Profiles of average transcoding time
- $\mathbf{W}$: Profiles of average chunk sizes
- D_{max}: QoS constraint of system delay
- S_{max}: Qos constraint of chunk size

Output:
- $\vec{P}$: Predicted distribution of transcoding modes
- N: Predicted number of CPU cores
- d: Predicted average system delay
- s: Predicted average chunk size

```
N = 0
d = −1 and s = −1
while d > D_max or d < 0 or s > S_max or s < 0 do
    N = N + 1
    for m = 1 to M − 1 do
        if m == 1 then
            p_m = 1
        else
            p_m = 1 − Σ_{i=1}^{m−1} p_i
        for i = m + 1 to M do
            p_i = 0
        while p_m > 0 do
            d = f(N, P⃗, T) according to Eq. (21)
            s = g(P⃗, W) according to Eq. (22)
            if d > D_max or s > S_max then
                p_m = p_m − τ and p_i = p_i + τ
            else
                break
```

As shown in Fig. 5, the task scheduling function in COVT is responsible to distribute the video chunk at the top of the queue to be processed in the cloud, with consideration of the practical QoS values d and s. For each video chunk, the scheduler determines its transcoding mode by the principle of choosing the slower modes as much as possible. By such manner, the system transcodes the tasks with slower modes when the QoS values are very low and with faster modes when the QoS values are high (close to D_{max} or S_{max}). After each task completion, the system records and updates the QoS values and the estimated transcoding probability $\vec{O}$ which is the practical value for $\vec{P}$. Based on the estimated value of the transcoding mode distribution, we can infer the utilization of CPU cores in the cluster. Then, we dynamically adjust the resource reservation in clouds to conserve costs while guaranteeing QoS.

The detailed scheduling algorithm is given in Algorithm 2. At the beginning of each time slot, the number of CPU cores reserved in clouds is set to the prediction N, and practical delay d, targeted video size s and estimate transcoding mode distribution $\vec{O}$ are all set to zero. For each task j, we introduce the scheduling algorithm with the following steps.

Firstly, the system judges whether there is vacant CPU core in the cluster for the task in the queue top. If so ($u < n$), the system starts finding the suitable transcoding mode to process the task. The slower mode that satisfies the QoS requirements $d \le D_{max},\ s \le S_{max}$ is used for the task in scheduling.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

Algorithm 2 Task scheduling in a time slot

Input:

$\mathbf{T}$: Profiles of average transcoding time
$\mathbf{W}$: Profiles of average chunk sizes
$\vec{P}$: Predicted distribution of transcoding modes
N: Predicted number of CPU cores in each time slot

Output:

n: Provisioning result of the system
d: Actual average system delay
s: Actual average chunk size

$u = 0$ //The number of used CPU
$j = 0$ //Task number
$\vec{O} = \vec{0}$ //Estimated proportions of transcoding modes
$d = 0$, $s = 0$
Let v_j be the video type of task j, $v_j = 1, 2, ..., V$
Let α_j^m be the practical transcoding time of chunk j using m^{th} mode
Let β_j^m be the practical output size of chunk j using m^{th} mode
for each time slot **do**
  $n = N$
  for task j in the system **do**
    if $u < n$ **then**
      for $k = 1$ **to** M **do**
        if $(w_{v_j}^k + s \cdot j)/(j+1) \leq S_{max}$ and $(t_{v_j}^k + d \cdot j)/(j+1) \leq D_{max}$ **then**
          $m = k$
          Break
    else
      if $(1 - THR) \cdot p_1 \leq o_1 \leq (1 + THR) \cdot p_1$ **then**
        Wait for a while and $m = M$
      else
        Reserve one more CPU core in the cloud
        $n = n + c$
        $m = M$
      Transcode j with mode m
    $s = (\beta_j^m + s \cdot j)/(j+1)$
    $d = (D_q + \alpha_j^m + d \cdot j)/(j+1)$
    $u = u + 1$
    if $m == 1$ **then**
      $o_1 = (o_1 \cdot j + 1)/(j+1)$
    else
      $o_1 = (o_1 \cdot j)/(j+1)$
    if $o_1 < (1 - THR) \cdot p_1$ **then**
      $n = n + c$
    else if $o_1 > (1 + THR) \cdot p_1$ **then**
      $n = n - c$
    $j = j + 1$
  for a video chunk is finished transcoding **do**
    $u = u - 1$

Secondly, if there is no available CPU core immediately for the task, the system checks whether o_1 is within a reasonable range specified by THR, where o_1 is the estimated probability of the slowest mode used in the system and THR, $THR < 1$, is a preset threshold. $(1 - THR) \cdot p_1 < o_1 < (1 + THR) \cdot p_1$ means that the actual number of tasks using the slowest mode is neither too high nor too low. Thus, the lack of available CPU core is a temporary situation, and the system will let the task wait for sometime for available CPU core. But if o_1 is not in the range and there is no available CPU core, the system will reserve a smallest VM (with c CPU cores) from the cloud to alleviate the high utilization of resource to guarantee QoS. Then, the task in queue top will be processed with M^{th} (fastest) mode.

Thirdly, after the processing, the system updates the records of practical QoS values as well as the number of CPU cores utilized. Besides, the proportion of the slowest transcoding mode used in the system is also updated as an important indicator for resource utilization. Every time when the task in queue top is processed with the slowest mode, the system increases the value of o_1. On contrast, the system decreases the value of o_1 when the task is not processed with mode 1. Then, if o_1 is larger than $(1 + THR) \cdot p_1$, the system will reduce the number of CPU cores by shutting down one of the smallest active VMs in order to save cost. If o_1 falls below $(1 - THR) \cdot p_1$, the system adds one VM to meet the computing needs. By such manner, COVT is capable to dynamically reserve resources in clouds under different system states and strict QoS.

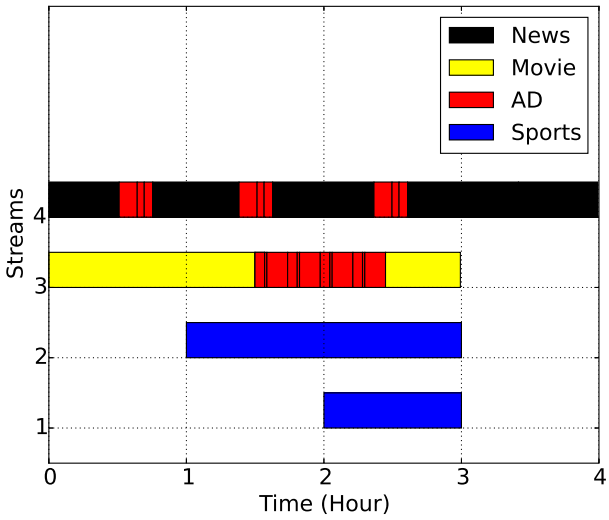


Fig. 6. Workloads in experiments.

VII. Testbed Experiments

A. Experiment setup

We implement a prototype of COVT and evaluate its performance on a cluster with six VMs that are hosted on a server with a six-core Xeon E5-1650 CPU and 16GB DRAM. Each VM is a transcoding worker (with one CPU core ($c = 1$) and two GB memory) that runs the transcoding algorithm for video chunks. Besides, we deploy another server as the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

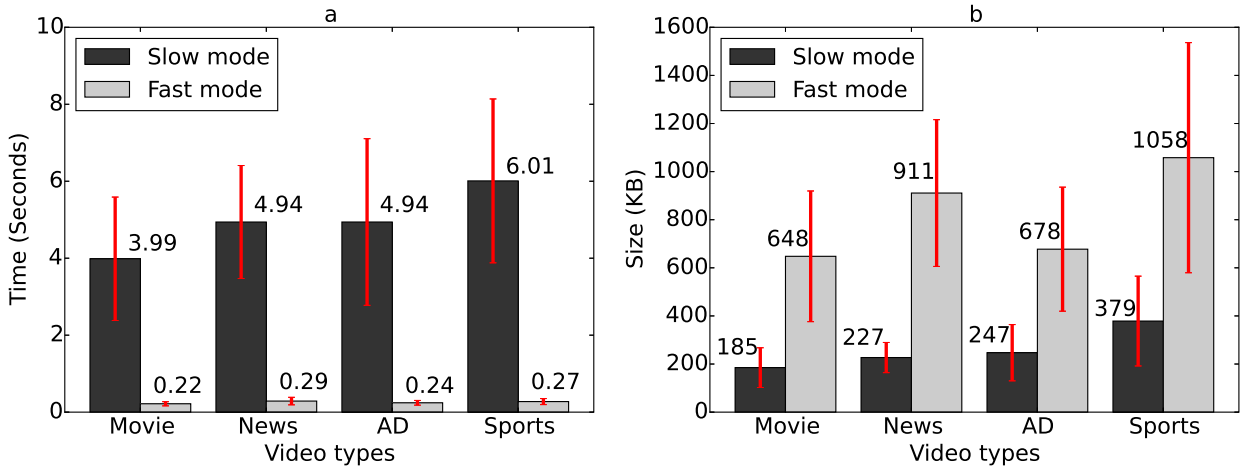


Fig. 7. Profiling results with two transcoding modes and four video types.

video service provider that is responsible for making resource scheduling decision and communicating with the cloud cluster. The whole system is built by python and the transcoder we use for video transcoding is an efficient video processing tool, FFmpeg. For convenience, we utilize two transcoding modes ($M = 2$) in the prototype system of COVT, namely *fast* and *slow* modes (note that they correspond to ultrafast and veryslow in FFmpeg, respectively). We consider four video types ($V = 4$) including movie, news, advertisement (AD) and sport. The threshold factor THR is set by default to 0.1. The parameter τ is set to 0.05. The default QoS constraints of delay and output size are 2 seconds and 500 KB, respectively.

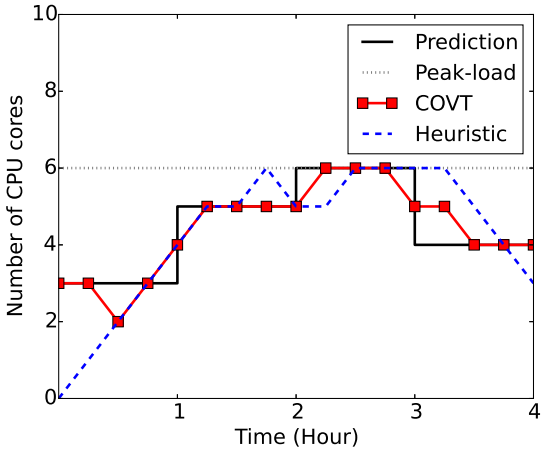


Fig. 8. Comparison of resource provisioning for different methods.

We use four video streams as the workloads for the cloud cluster in the experiments as shown in Fig. 6. The video data in stream 1 and 2 are a soccer game in World Cup 2014 and a table tennis game in Olympic game 2012, respectively. The data for stream 3 and 4 are from the famous TV station *Phoenix TV* in Hongkong. To show the performance under dynamic workloads, the streams are with different starting time and finishing time. We partition the total operating time into time slots with a length of 30 minutes and the resource provisioning is performed for each time slot. All the video contents are segmented into 5 ($L = 5$) seconds chunks in playback length with the container of MP4 and the resolution of 640x360. The videos are transcoded to H.264 with the container of AVI and the resolution of 320x240.

We compare COVT with two other schemes: peak-load provisioning (Peak-load) and heuristic provisioning (Heuristic). Peak-load is a static resource provisioning method that is used in many cloud applications as a basic one without any optimization. In Peak-load, it always reserves the number of resource that satisfies the QoS in the peak loads. Thus, we use the largest number of cores in COVT as the Peak-load provisioning. Heuristic provisioning is another popular method in clouds, while it dynamically provisions resource according to the resource utilization. Specifically, Heuristic increases resource when the resource utilization is too high (over 80%) and decreases resource when the resource utilization is too low (lower than 60%). For fair comparison, Heuristic also uses our predicted transcoding mode probability distribution.

B. Experimental Results

Profiling results. We firstly consider to obtain the profiles of the transcoding time and the targeted video chunk size. By running the video data one hour prior to the workloads in Fig. 6, we record the average transcoding time and the video chunk size for different video types and transcoding modes under the considered infrastructure, which are illustrated in Fig. 7. The bars represent average values and the red vertical lines show the corresponding 95% confidence intervals.

From Fig. 7, it can be seen that the transcoding time and the chunk size are significantly different for different transcoding modes. Specifically, the time for transcoding a video chunk using the slow mode is nearly 20 times of that using the fast mode, which offers a large space for the service provider to schedule the resource for a predefined QoS goal. Besides, the processing time of transcoding tasks for different types of video are closer using the fast mode. The case for the slow mode is more complicated since it depends on the data content. The average size of chunks with the fast mode is approximately triple of that with the slow mode. Thus, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

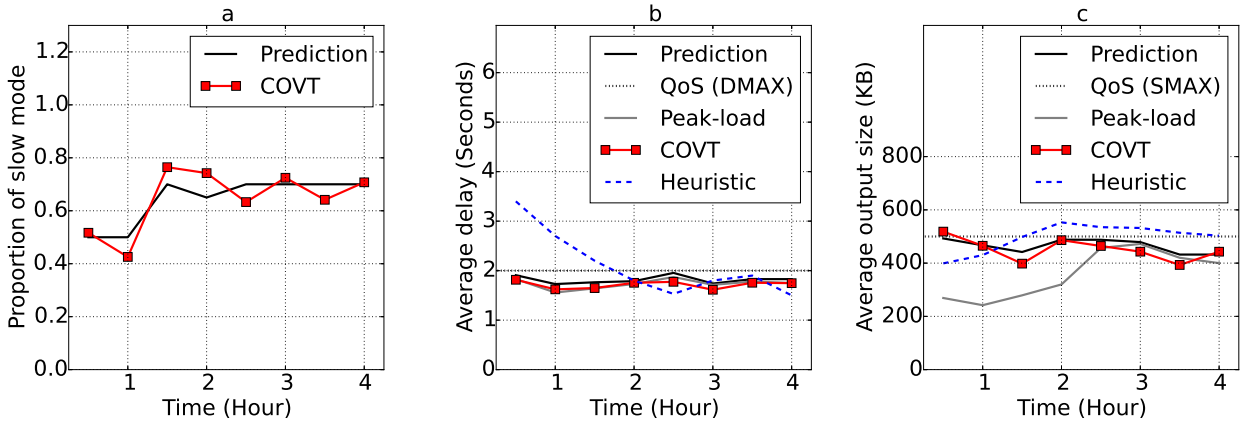


Fig. 9. Detailed results of slow mode proportion, delay and chunk size.

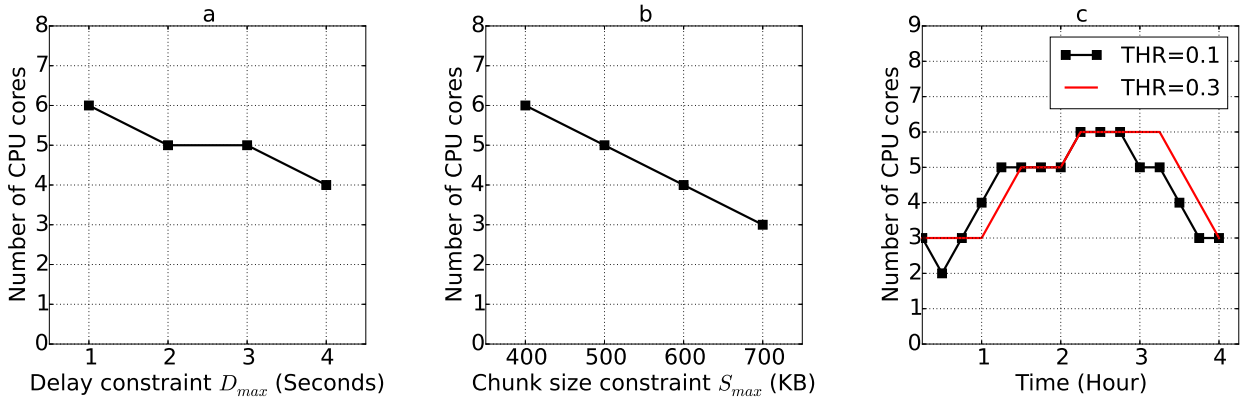


Fig. 10. Parameter studies of the testbed experiments.

slow mode produces smaller targeted video chunk size than the fast mode but takes longer transcoding time. Based on these profiling data on CPU cores under our experimental environment, COVT is able to predict the suitable number of cores for the workload.

Overall comparisons. Next, we present the overall comparisons of COVT with other methods in terms of resource provisioning for the online transcoding workloads in Fig. 8, where the provisioned numbers of CPU cores for the four hour period for Peak-load, Heuristic, model prediction and our proposed COVT are illustrated. We can see that the results of the prediction and COVT are quite close while the Heuristic approach differs in the beginning with climbing number of resources and in the end with falling resource provisioning. This is due to the deficiency of Heuristic that reacts slowly to the dynamic variation of workloads. Overall, COVT is able to conserve 25% resources in terms of CPU-hour compared with Peak-load for the workloads.

Together with Fig. 9 which draws the detailed information in the system runtime, we can see that Heuristic approach cannot meet the QoS requirements since it only passively reacts to the dynamic workloads. For example, at the beginning of the workloads, the provisioned CPU cores are lesser but the delay QoS is broken in Heuristic approach in Fig. 9 (b). Similar QoS broken can be viewed in Fig. 9 (c). In contrast, the results of COVT comply with the QoS constraints strictly, which demonstrates the effectiveness of COVT in provisioning QoS-sensitive video transcoding services. The 95% confidence intervals (8 time slots) of the results of COVT over multiple tests in Fig. 9 (a) , Fig. 9 (b) , Fig. 9 (c) are $[0.012, 0.018, 0.021, 0.023, 0.016, 0.019, 0.022, 0.018]$, $[0.043, 0.037, 0.029, 0.045, 0.056, 0.043, 0.049, 0.053]$ and $[167, 203, 151, 214, 176, 208, 142, 128]$, respectively.

Impacts of delay constraint. Fig. 10 (a) shows the parameter study of the delay constraint D_{max}. Fixing other parameters to their default values, we run the experiments with a set of D_{max} in $[1, 2, 3, 4]$ seconds. From the figure, we can see that the number of CPU cores reserved for the same workloads decreases when the delay constraint D_{max} increases. As the QoS constraint gets looser, less resource is needed to meet the delay requirement.

Impacts of chunk size constraint. The targeted chunk size is studied in Fig. 10 (b). Similarly, the required number of CPU cores decreases as S_{max} increases. The studies imply that both QoS parameters have a significant influence on the cloud resource and thus they need to be carefully selected based on specific application requirements.

Impacts of threshold factor. Fig. 10 (c) gives the results for varying the threshold factor THR in the task scheduling algorithm. As shown, a larger THR (0.3) reacts less frequently than a smaller one (0.1). A small THR can adjust the number of resource quickly according to bursty in workloads. On the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

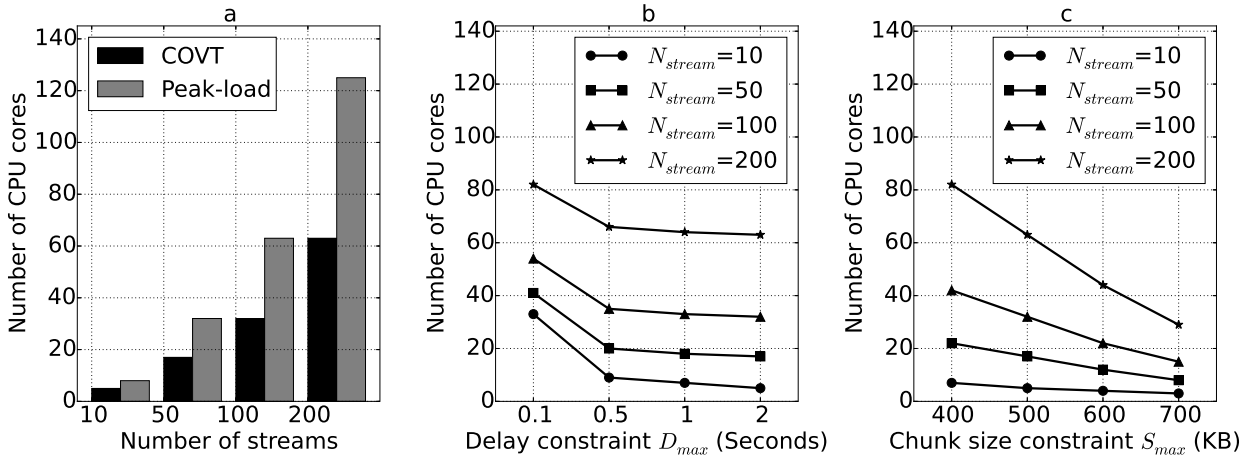


Fig. 11. Simulation results for large scale data set.

other hand, it should be limited by the minimum resource reservation period in the clouds.

VIII. Simulation Evaluation

The results of the testbed experiments reveal the effectiveness of COVT on provisioning online transcoding services with practical system setup in a small virtual cluster. In this section, we seek simulation studies for COVT in order to investigate the scalability beyond the limitation of the scale of the testbed infrastructure.

We develop discrete-time event simulation with simulated workloads. We consider four workloads with $10, 50, 100, 200$ as the maximum number of streams (also total time slots). For each workload, the number of streams increases from 1 in the first time slot to the maximum number with an increment 1. The simulation shares the same settings with the testbed experiments including the two transcoding modes and four video types as well as the profiling data. The proportion of different video types are equal in each time slot. The default QoS constraints of delay and output size are set to 1 second and 500 KB, respectively.

Simulation results. The simulation results with large scale data set are illustrated in Fig. 11. Based on the results of provisioned number of CPU cores under different workload scales for COVT and Peak-load approach in Fig. 11 (a), we make the following observations.

Firstly, the results reveal that our proposed system COVT is capable to work for large scale online video transcoding in clouds. As the workloads vary from 1 in the first time slot to 200 streams in the 200^{th} slot, COVT precisely provisions appropriate number of CPU cores for the predefined QoS.

Secondly, the advantage of using clouds as the processing platform is validated. As shown, the actual resource requirements of COVT is significantly less than Peak-load by approximately 47%. Thus, Peak-load solution for online video transcoding would waste a lot of investments on the IT infrastructure.

Thirdly, it is observed that the resource conservation rate is higher when the maximum number of video streams gets larger. It is more clearly illustrated in Fig. 12, which shows

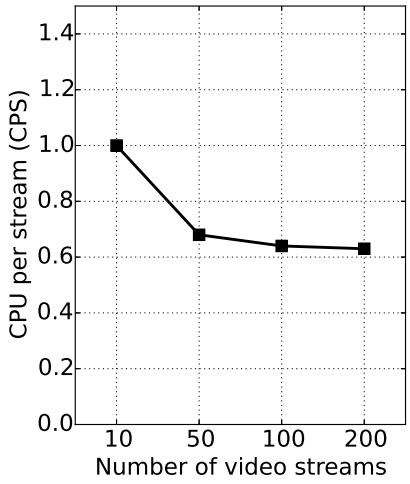


Fig. 12. Number of CPU cores per stream.

the number of CPU cores required for each video stream in the workloads, namely CPU per stream (CPS). CPS varies from greater than one in the testbed experiments to one in simulation using 10 streams and further decreases to less than one (approximately 0.63) when there are 200 streams. It implies that the larger the scale of the data set, the more benefit we can have for the proposed cloud based video transcoding.

Fig 11 (b) and Fig 11 (c) investigate the impacts of the QoS values D_{max} and S_{max} on the resource provisioning under large scale data set. They show stable effectiveness on different scales of workloads.

Overall, the simulation results validate the effectiveness of COVT for large scale online video transcoding applications. It shows the advantages in efficient resource conservation, elastic computing, QoS guarantee and high scalability.

IX. Conclusion

In this paper, we have proposed a novel method called COVT to address the problem of economical and QoS-aware online video transcoding in cloud environment. COVT considers two types of QoS: system delay and targeted video chunk size. As the transcoding time and the targeted chunk size

are different on different hardware using different transcoding modes, we perform profiling for the cloud cluster to assist resource prediction and scheduling. We partition video streams into video chunks with short playback time and schedule them as tasks in clouds. We model the video streams as a queue and derive the relationship between QoS values and number of resources, based on which we solve the minimum number of resources with the QoS constraints. With the resource prediction, we have also proposed a scheduling algorithm to schedule transcoding tasks into clouds with strict QoS guarantee.

We perform both testbed and simulation experiments to evaluate our method on real-world workloads and large-scale simulated workloads, respectively. The results show that our proposed solution is effective in terms of resource provisioning compared with the method that provisions physical resource according to the peak load. The number of CPU cores using COVT is up to 47% less than Peak-load in our experiments. It also showed that COVT outperforms Heuristic approach in terms of QoS guarantee. Overall, the experimental results validat our design goal that provisions minimum amount of resource while satisfying QoS constraints.

REFERENCES

[1] "Cisco visual networking index: Global mobile data traffic forecast update, 2012–2017," *Cisco Public Information*, 2013.

[2] R. Cheng, W. Wu, Y. Lou, and Y. Chen, "A cloud-based transcoding framework for real-time mobile video conferencing system," in *International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud'14)*. IEEE, 2014.

[3] A. Vetro, J. Cai, and C. W. Chen, "Rate-reduction transcoding design for wireless video streaming," *Wireless Communications and Mobile Computing*, vol. 2, no. 6, pp. 625–641, 2002.

[4] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: an overview," *Signal Processing Magazine, IEEE*, vol. 20, no. 2, pp. 18–29, 2003.

[5] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[6] D. Villegas, A. Antoniou, S. M. Sadjadi, and A. Iosup, "An analysis of provisioning and allocation policies for infrastructure-as-a-service clouds," in *Proc. of 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid'12)*. IEEE, 2012.

[7] K. Mills, J. Filliben, and C. Dabrowski, "Comparing vm-placement algorithms for on-demand clouds," in *Proc. of CLOUDCOM'11*, 2011.

[8] P. Marshall, H. Tufo, and K. Keahey, "Provisioning policies for elastic computing environments," in *Proc. of 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW)*. IEEE, 2012.

[9] L. Wang, J. Zhan, W. Shi, and Y. Liang, "In cloud, can scientific communities benefit from the economies of scale?" *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, no. 2, pp. 296–303, 2012.

[10] R. V. den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads," in *Proc. of IEEE CLOUD'10*, 2010.

[11] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, "Cost-and deadline-constrained provisioning for scientific workflow ensembles in iaas clouds," in *Proc. of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'12)*. IEEE Computer Society Press, 2012.

[12] A. Ali-Eldin, M. Kihl, J. Tordsson, and E. Elmroth, "Efficient provisioning of bursty scientific workloads on the cloud using adaptive elasticity control," in *Proc. of the 3rd workshop on Scientific Cloud Computing Date*. ACM, 2012.

[13] S. Niu, J. Zhai, X. Ma, X. Tang, and W. Chen, "Cost-effective cloud hpc resource provisioning by building semi-elastic virtual clusters," in *Proc. of International Conference for High Performance Computing, Networking, Storage and Analysis (SC'13)*. ACM, 2013.

[14] L. Wei, B. He, and C. H. Foh, "Towards multi-resource physical machine provisioning for iaas clouds," in *Communications (ICC), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3469–3472.

[15] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 24, no. 6, pp. 1107–1117, 2013.

[16] M. Mao and M. Humphrey, "Auto-scaling to minimize cost and meet application deadlines in cloud workflows," in *Proc. of SC'11*, 2011.

[17] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica, "Dominant resource fairness: fair allocation of multiple resource types," in *Proc. of USENIX NSDI 2011*, 2011.

[18] L. Wei, C. H. Foh, B. He, and J. Cai, "Towards efficient resource allocation for heterogeneous workloads in iaas clouds," *Cloud Comuting, IEEE Transactions on*, pp. In press, DOI: 10.1109/TCC.2015.2 481 400.

[19] M. Kim, Y. Cui, S. Han, and H. Lee, "Towards efficient design and implementation of a hadoop-based distributed video transcoding system in cloud computing environment," *International Journal of Multimedia and Ubiquitous Engineering*, vol. 8, no. 2, pp. 213–224, 2013.

[20] A. Ashraf, F. Jokhio, T. Deneke, S. Lafond, I. Porres, and J. Lilius, "Stream-based admission control and scheduling for video transcoding in cloud computing," in *Proc. of International Symposium on CCGrid'13*. IEEE, 2013.

[21] A. Ashraf, "Cost-efficient virtual machine provisioning for multi-tier web applications and video transcoding," in *Proc. of International Symposium on CCGrid'13*. IEEE, 2013.

[22] Z. Zhuang and C. Guo, "Building cloud-ready video transcoding system for content delivery networks (cdns)," in *Proc. of Global Communications Conference (GLOBECOM'12)*. IEEE, 2012.

[23] R. Pereira, M. Azambuja, K. Breitman, and M. Endler, "An architecture for distributed high performance video processing in the cloud," in *Proc. of CLOUD'10*. IEEE, 2010.

[24] A. Garcia, H. Kalva, and B. Furht, "A study of transcoding on cloud environments for video content delivery," in *Proc. of the 2010 ACM multimedia workshop on Mobile cloud media computing*. ACM, 2010.

[25] A. GARcIA and H. Kalva, "Cloud transcoding for mobile video content delivery," in *Proc. of the IEEE International Conference on Consumer Electronics (ICCE)*, 2011.

[26] F. Jokhio, A. Ashraf, S. Lafond, I. Porres, and J. Lilius, "Prediction-based dynamic resource allocation for video transcoding in cloud computing," in *2013 21st Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*. IEEE, 2013.

[27] F. Jokhio, A. Ashraf, S. Lafond, and J. Lilius, "A computation and storage trade-off strategy for cost-efficient video transcoding in the cloud," in *39th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 2013.

[28] Z. Wang, L. Sun, C. Wu, W. Zhu, and S. Yang, "Joint online transcoding and geo-distributed delivery for dynamic adaptive streaming," in *Proc. of the INFOCOM 2014*. IEEE.

[29] Z. Wang, L. Sun, C. Wu, W. Zhu, A. Zhuang, and S. Yang, "A Joint Online Transcoding and Delivery Approach for Dynamic Adaptive Streaming," *IEEE Transactions on Multimedia*, vol. 17, no. 8, pp. 867–879, 2015.

[30] A. Kathpal, M. Kulkarni, and A. Bakre, "Analyzing compute vs. storage tradeoff for video-aware storage efficiency," in *Proc. of the 4th USENIX Conference on Hot Topics in Storage and File Systems, HotStorage*, 2012.

[31] W. Zhang, Y. Wen, J. Cai, and D. O. Wu, "Toward transcoding as a service in a multimedia cloud: energy-efficient job-dispatching algorithm," *Vehicular Technology, IEEE Transactions on*, vol. 63, no. 5, pp. 2002–2012, 2014.

[32] D. Xie, N. Ding, Y. C. Hu, and R. Kompella, "The only constant is change: incorporating time-varying network reservations in data centers," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 199–210, 2012.

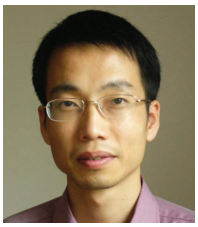
[33] H. Herodotou and S. Babu, "Profiling, what-if analysis, and cost-based optimization of mapreduce programs," *Proc. of the VLDB Endowment*, 2011.

[34] H. Herodotou, F. Dong, and S. Babu, "No one (cluster) size fits all: automatic cluster sizing for data-intensive analytics," in *Proc. of the 2nd ACM Symposium on Cloud Computing*. ACM, 2011.

[35] H. Herodotou, H. Lim, G. Luo, N. Borisov, L. Dong, F. B. Cetin, and S. Babu, "Starfish: A self-tuning system for big data analytics." in *Proc. of CIDR'11*, 2011.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2016.2589621, IEEE Transactions on Circuits and Systems for Video Technology

Lei Wei received his Ph.D degree in 2016 from School of Computer Engineering, Nanyang Technological University, Singapore. His research interests include Distributed computing and Big data processing.



Jianfei Cai received his Ph.D. degree from the University of Missouri-Columbia. He is currently an Associate Professor and has served as the Head of Visual & Interactive Computing Division and the Head of Computer Communication Division at the School of Computer Engineering, Nanyang Technological University, Singapore. His major research interests include visual computing and multimedia networking. He has served as the leading Technical Program Chair for IEEE International Conference on Multimedia & Expo (ICME) 2012 and the leading General Chair for Pacific-rim Conference on Multimedia (PCM) 2012. Since 2013, he has been serving as an Associate Editor for IEEE Trans on Image Processing (T-IP). He has also served as an Associate Editor for IEEE Trans on Circuits and Systems for Video Technology (T-CSVT) from 2006 to 2013 and a Guest Editor for IEEE Trans on Multimedia (TMM), ELSEVIER Journal of Visual Communication and Image Representation (JVCI), etc. He is a senior member of IEEE.

Chuan Heng Foh received his Ph.D. degree from the University of Melbourne, Australia in 2002. After his PhD, he spent 6 months as a Lecturer at Monash University in Australia. In December 2002, he joined Nanyang Technological University, Singapore as an Assistant Professor until 2012. He is now a Senior Lecturer at the University of Surrey. His research interests include protocol design and performance analysis of various computer networks including wireless local area and mesh networks, mobile ad hoc and sensor networks, 5G networks, and data center networks. He has authored or coauthored over 100 refereed papers in international journals and conferences. He is a senior member of IEEE.

Bingsheng He received the bachelor degree in computer science from Shanghai Jiao Tong University (1999-2003), and the PhD degree in computer science in Hong Kong University of Science and Technology (2003-2008). He is an Associate Professor in School of Computing, National University of Singapore. His research interests are high performance computing, distributed and parallel systems, and database systems.