Cloud-Based Interactive Video Streaming Service

Mohsen Amini Salehi
HPCC lab., School of Computing and Informatics,
University of Louisiana at Lafayette
Lafayette, Louisiana, USA
amini@louisiana.edu

ABSTRACT

A wide range of applications, from e-learning to natural disaster management are reliant on video streaming. Video streaming will construct more than 80% of the whole Internet traffic by 2019. Currently, video stream providers offer little or no interactive services on their streamed videos. Stream viewers, however, demand a wide variety of interactive services (*e.g.,* dynamic video summarization or dynamic transcoding) on the streams. Taking into account the long tail access pattern to video streams, it is not feasible to preprocess all possible interactions for all video streams. Also, Processing them is also not feasible on energy- and compute-limited viewers' thin-clients. The proposed research provides a cloud-based video streaming engine that enables interactive video streaming. Interactive Video Streaming Engine (IVSE) is generic and video stream providers can customize it by defining their own interactive services, depending on their applications and their viewers' desires. The engine enacts the defined interactive services through on-demand processing of the video streams on potentially heterogeneous cloud services, in a cost-efficient manner, and with respect to stream viewers' QoS demands.

KEYWORDS

Cloud Computing, Video Streaming, Resource Allocation, Real-time Processing

1 INTRODUCTION

Thanks to the high speed Internet, basic video streaming has become an ordinary service nowadays. However, what is offered currently is far from the higher level services that enable stream viewers to *interact* with the video streams. *Interactive video streaming* is defined as processing of a video stream upon viewersâĂŹ requests for that video. For instance, a viewer may request to watch a video stream with a particular resolution [4]. Another example, is a viewer who requests to view a summary of a video stream.

Current interactive video streaming services are very limited and often require preprocessing of the video streams. However, given the diversity of services offered in an ideal interactive video streaming and the long tail access pattern to the video streams [9], offering interactive video streaming based on lazy (*i.e.,* on-demand) processing of the video streams is required. Such computationally-intensive processing should be achieved in a real-time manner and guarantee specific QoS demands of the viewers.

Cloud services have provided an ideal platform for video streaming providers to satisfy the computational demands needed for interactive video streaming [4]. However, the common problem in utilizing cloud services [7, 8] for interactive video streaming is: *how to provide a robust interactive video streaming service through guaranteeing QoS desires of the viewers, while incurring the minimum cost for the cloud services*? Accordingly, the objective of this research is to present challenges, structures, and methods required to enable interactive video streaming that guarantee QoS in a cost-efficient manner. In particular, we present a framework for interactive video streaming called *Interactive Video Streaming Engine* (IVSE) that deals with the challenges of cloud-based interactive video streaming services and provides methods to address these challenges.

The reason that video streaming tasks need independent study is that they have unique characteristics. Video streaming tasks have individual deadlines that can be a hard deadline (in live streams [2]) or a soft deadline (in Video On Demand (VOD) [4]). Recent studies (*e.g.,* [6]) show that viewers often watch the beginning of video streams, as such, the quality of delivering the startup of video streams is of paramount importance. Accordingly, video streams have unique QoS demands that are defined as: minimizing missing tasks' individual deadlines and minimizing the startup delay of the streams.

Depending on the type of video stream content, their processing times (*i.e.,* execution time) vary on different types of processing services (*i.e.,* Virtual Machines) offered by cloud providers. Hence, to schedule video streaming tasks, we potentially deal with mapping tasks to heterogeneous cluster of Virtual Machines (VMs). In such a heterogeneous computing environment, predicting the execution time of of video streaming tasks is necessary to efficiently map tasks to VMs. Execution time prediction is viable thorough historic execution information for VOD streams, however, this is not the case in live streams, where video streaming tasks are generated and processed for the first time [1]. Processing performance of cloud VMs may vary over time or even VM failure can occur. In this case, all video streams assigned to those VMs cannot proceed with streaming. Hence, execution of video streams are required and failed tasks have to be rescheduled with a high priority to enable smooth video streaming. The access rate to video streams in a repository is not uniform. In fact, access patterns to video streams exhibits a long-tail pattern [9]. As such, caching methods are required to identify *hot* video streams and appropriately cache (store) them using different cloud storage services.

UCC'17, , December 5–8, 2017, Austin, TX, USA.

ACM ISBN 978-1-4503-5149-2/17/12.
https://doi.org/10.1145/3147213.3149451

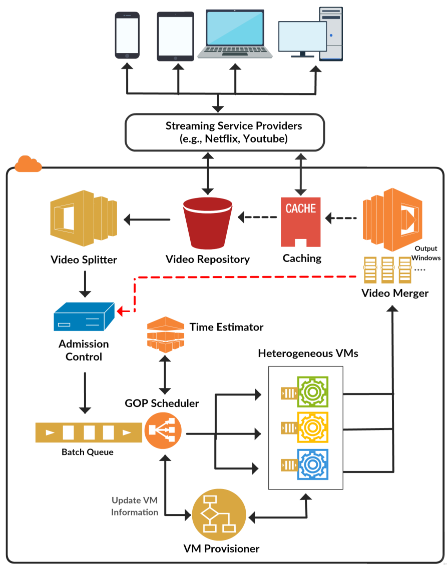

Figure 1: Cloud-based Interactive Video Streaming Engine (IVSE)

1.1 Interactive Video Streaming Engine (IVSE)

IVSE facilitates cost-efficient and QoS-aware interactive live or VOD streaming using cloud services for different type of subscribers. IVSE is extensible, meaning that the video stream provider is able to introduce new interactive services on video streams and the core architecture can accommodate the services while respecting the QoS and cost constraints of the video stream provider.

An overview of IVSE is presented in Figure 1. Upon receiving a streaming request, Video Splitter partitions the video into several Group of Pictures (*GOPs*) [3] that can be processed independently. Each GOP is treated as a task with an individual deadline which is the presentation time of the first frame in that GOP. The Admission Control component prioritizes dispatching of the GOPs to the scheduling queue. The VM Provisioner component allocates heterogeneous VM(s) from cloud to execute GOPs. Each VM is assigned a local queue to preload GOPs' data before execution. VM Provisioner component monitors the performance of VMs and adaptively configures the heterogeneity of the VM cluster based on the workload. Time Estimator provides predictive information on the affinity of GOP tasks with various VM types. The Scheduler component uses the estimation information for efficient allocation of tasks to VMs. Video Merger rebuilds the processed stream using an output window for each stream. In the event that a GOP is delayed (*e.g.,* due to failure), Video Merger asks the Admission Control to resubmit the GOP urgently. The Caching component decides if a part of, or the whole processed stream should be stored.

In summary, this this research project describes innovations in interactive video streaming particularly in the following areas:

- Robust, cost-efficient, and self-configurable VM provisioning policy: We explain novel methods to provision a dynamic VM cluster that conforms its heterogeneity according to the arriving requests (see [4] for further details).
- Heterogeneity- and QoS-aware scheduling method: It efficiently schedules streaming tasks on available heterogeneous VMs with the goal of minimizing both missing tasks' deadlines and their startup delays (see [4] for further details).
- Execution time prediction for video streaming tasks: We elaborate on the influential factors of the video streaming tasks execution times. In addition, we explain the way to model affinity exists between heterogeneous VMs and tasks while considering their cost difference (see [10] for further details).
- A priority-aware admission control method: That prioritizes submission of streaming tasks to minimize the startup delay. The method can also consider the viewer subscription priority, and network speed at the viewers' end.
- Cost-efficient caching methods: We will elaborate on the trade-off between computation versus storage for video streams. We also provide a formal way to measure the hotness of video streams and provide methods that perform caching based on the hotness measure (see [5] for further details).

2 CONCLUSIONS

Video streaming is one of the prominent services of the current and future Internet. Viewers increasingly request for more interactions on the streamed videos. In this research, we provide an Interactive Video Streaming Engine (IVSE) that leverages cloud computing in an efficient way to provide flexible interactivity for the streamed videos. IVSE can be easily extended with new video processing services and can support live and on-demand streaming.

REFERENCES

[1] Matin Hosseini, Mohsen Amini Salehi, and Raju Gottumukkala. 2017. Enabling Interactive Video Stream Prioritization for Public Safety Monitoring through Effective Batch Scheduling. In *Accepted in the 19th IEEE International Conference on High Performance Computing and Communications* (Bankok).

[2] Xiangbo Li, Mohsen Amini Salehi, and Magdy Bayoumi. 2016. VLSC: Video Live Streaming Using Cloud Services. In *Proceedings of 5th IEEE International Conferences on Big Data and Cloud Computing*. 595–600.

[3] Xiangbo Li, Mohsen Amini Salehi, and Magdy Bayoumi. Oct. 2015. Cloud-Based Video Streaming for Energy- and Compute-Limit Thin Clients. In *the Stream2015 Workshop at Indiana University*.

[4] Xiangbo Li, Mohsen Amini Salehi, Magdy Bayoumi, and Rajkumar Buyya. 2016. CVSS: A Cost-Efficient and QoS-Aware Video Streaming Using Cloud Services. In *Proceedings of the 16th IEEE/ACM International Conference on Cluster Cloud and Grid Computing (CCGrid '16)*.

[5] Darwich Mahmoud, Beyazit Ega, Salehi Mohsen Amini, and Bayoumi Magdy. 2017. Cost Efficient Repository Management for Cloud-Based On-Demand Video Streaming. In *In proceedings of 5th IEEE Internationa Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*. 1–6.

[6] Lucas CO Miranda, Rodrygo LT Santos, and Alberto HF Laender. 2013. Characterizing video access patterns in mainstream media portals. In *Proceedings of the 22nd International Conference on World Wide Web*. 1085–1092.

[7] Murali K. Pusala, Mohsen Amini Salehi, Jayasimha R. Katukuri, Ying Xie, and Vijay Raghavan. 2016. *Massive Data Analysis: Tasks, Tools, Applications, and Challenges*. Springer India, New Delhi, 11–40.

[8] Mohsen Salehi and Rajkumar Buyya. 2010. Adapting Market-Oriented Scheduling Policies for Cloud Computing. In *Algorithms and Architectures for Parallel Processing*. ICA3PP '10, Vol. 6081. 351–362.

[9] Navin Sharma, Dilip Kumar Krishnappa, David Irwin, Michael Zink, and Prashant Shenoy. 2013. Greencache: Augmenting off-the-grid cellular towers with multimedia caches. In *Proceedings of the 4th ACM Multimedia Systems Conference*. 271–280.

[10] Li Xiangbo, Joshi Yamini, Darwich Mahmoud, Landreneau Brade, Amini Salehi Mohsen, and Bayoumi Magdy. 2017. Performance Analysis and Modeling of Video Transcoding Using Heterogeneous Cloud Services. In *IEEE Transactions on Parallel and Distributed Systems*. 1–12.