Performance Benchmark of Transcoding as a Virtual Network Function in CDN as a Service Slicing

Ilias Benkacem[1], Tarik Taleb[1], Miloud Bagaa[1], and Hannu Flinck[2]
[1] Dept. of Communications and Networking, School of Electrical Engineering, Aalto University, Espoo, Finland
[2] Nokia Bell Labs, Espoo, Finland
Emails: [1]firstname.lastname@aalto.fi, [2]hannu.flinck@nokia-bell-labs.com

***Abstract*—Content delivery networks (CDNs) have been widely implemented to provide scalable cloud services. Such networks support resource pooling by allowing virtual machines to be dynamically running or stopping according to current users' demands. Recently, there has been an increasing interest in Network Function Virtualization (NFV) as an emerging technology that aims to reduce cost, enable scalability and flexibility by decoupling network functions from the underlying hardware. In this regard, this paper designs a novel architecture to provide CDN Slices as a Service and that is across multiple administrative cloud domains. The architecture is aligned with the NFV Management and Orchestration (MANO) models. The proposed platform consists of three virtual network functions (VNFs), namely virtual caches, virtual video streamers, and virtual video transcoders. Regarding the latter, the paper also proposes a scheme for load balancing the transcoding tasks of the uploaded videos over a distributed network of virtual transcoders. In this article, an extensive benchmark analysis is conducted in order to study the virtual transcoding behavior in different cloud environments. The experiment evaluations provides a solid knowledge base to predict the estimated transcoding time for an optimal workload management of videos, aiming to optimize the incurred efficient cost in terms of delivery time and latency.**

I. Introduction

Generally speaking, performance and reliability have become major factors that directly impact the user experience. In our highly connected world and extremely rising user demands, it becomes very important to reach clients whenever and wherever they are. The content is delivered to the end-users based on their geographic locations and availability of resources using the Content Delivery Network (CDN) as a distributed network of geographically dispersed servers According to the Cisco Visual Networking Index (2016–2021) report [1], it is expected that CDN traffic will carry 71% of all Internet traffic, most of which will consist of IP video traffic. Therefore, it becomes essential to satisfy the growing demands of users by distributing the video contents in an efficient manner. Although video contents can be transcoded into various formats, this requires greedy processes that need a large amount of computation for decoding and encoding. Moreover, efficient media delivery requires high performance transcoding using Video On Demand (VOD), as well as live broadcast platforms for various types of user devices.

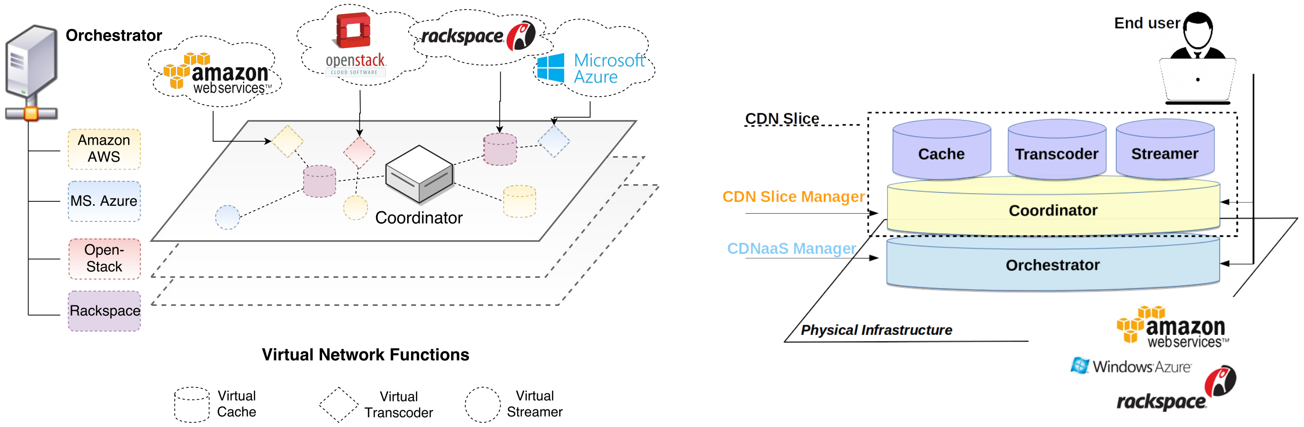
In fact, new challenges have emerged in terms of the transcoding and streaming process for delivering the video contents to end-users throughout the world. Moreover, with the growing number of the new device capabilities, there is a new concern on improving the Quality of Experience (QoE) and Quality of Service (QoS) while transcoding and delivering content to end-users. Hence, video transcoding is required to customize the service based on end-users' demands by taking into account the diversity of situations in terms of network speed/bandwidth limits, screen resolutions and video types supported by the new devices.

The existing body of research on network virtualization suggests video transcoding in the cloud [2], [3] in order to control the usage of virtual resources and to be close to users' locations. The CDN as a Service architecture, proposed in [4], ensures a fast response and delivery time of content due to reduced latency, therefore ensuring high QoS. In this architecture, a CDN slice mainly consists of a set of VNFs such as virtual streamers and virtual transcoders running across multi-administrative cloud domains. In CDNs and aiming for high scalability and high system responsiveness, load balancing is an important challenge as there is need to efficiently distribute jobs/tasks among servers distributed over the world. In this vein, and as an additional contribution to the large library of research work on CDN slicing leveraging NFV [5], this article introduces an extensive benchmark analysis to study the virtual transcoding behavior in different cloud environments. The experiment evaluations will provide a solid knowledge base to predict the estimated transcoding time for an optimal workload management of videos aiming to optimize the incurred cost in terms of delivery time and latency.

The rest of the paper is structured as follows. Section II summarizes the fundamental background topics and related research works. Section III gives a brief description of the proposed CDN as a Service architecture and presents the different components of the system. Section IV explains the problem and describes our proposed framework solution. Section V illustrates the transcoding performance over multi-cloud domains in order to construct a knowledge base for learning and to carry predictions of system performance. An extensive benchmark study is presented in Section V. Finally, the paper concludes in Section VI.

II. Related Work

In this section, we briefly summarize the literature research work relevant to our research topic. First, we start introducing a literature review related to network slicing, an important concept for our vision on CDN slicing over multi-cloud



(a) Multiple Fully-Fledged CDN Slices over Multiple Cloud Domains.

(b) Key Stakeholders of a CDNaaS Architecture.

Fig. 1. CDN as a Service Architecture

domains. Then, we present the research work carried out for enabling cloud-based video transcoding.

Regarding network slicing and MANO, there has been an important amount of research work conducted recently summarized in [6] [7]. Jose et al. [8] provide a comprehensive study of the architectural frameworks of both SDN and NFV as key enablers to achieve the realization of network slices. Peter et al. [9] provide the necessary flexibility and scalability associated with future network implementations. The authors propose 5G based on network slicing with the coexistence of dedicated as well as shared slices in the network. Bin et al. [10] present a network slice design for ultra high definition (UHD) video broadcast/multicast to achieve higher network efficiency and improved quality of experience. The authors in [11]–[13] improve the flexibility of network resource allocation and capacity of 5G networks based on network slicing and discuss the potential of network slicing to provide the appropriate customization and highlight the relevant technology challenges.

With regard to transcoding, there is a wide library of research work aiming to improve the performance of transcoding leveraging cloud computing. In [14], Guanyu et. al. implemented an open source cloud transcoding system and evaluated its performance in terms of reducing the resource consumption and achieving a higher profit compared with baseline schemes. In [14]–[16], the authors utilized cloud services for on-demand video transcoding in order to maintain a robust QoS for viewers and cost-efficiency for streaming service providers reducing the incurred cost. Hajiesmaili et al. expose the multi-party cloud video conferencing architecture to exploit rich computing and bandwidth resources in the cloud to effectively improve the video conferencing performance [17]. Here, the authors proposed NP-hard node assignment problems for the selection of suitable transcoding agents to perform transcoding tasks.

For developing a large-scale video transcoding platform, researchers have been looking into the potential of CDNs. On the other hand, the NFV concept has been gaining much attention in many research fields, including CDN slicing [4] [16]. In such virtual CDN slices, transcoding servers are considered as virtual network functions that can be hosted in virtual machines instantiated in different cloud domains. The cost associated with the deployment of these virtual transcoders (and other VNF types) can be optimized according to different factors, such as Quality of Experience as in [4].

III. CDN Slicing Across Multiple Administrative Cloud Domains

CDN as a Service (CDNaaS) is a VoD platform that allows the creation and life-cycle management of CDN slices running across multiple cloud domains [18]. The CDN slices consists of four main components, namely virtual transcoders, virtual streamers, virtual caches, and a CDN-slice-specific Coordinator for the management of the slice resources and the uploaded videos and that is across different private and public infrastructure as a service (IaaS) providers, such as Amazon AWS service, Microsoft Azure, Rackspace or on own data-centers administrated by OpenStack. A CDN slice consists of a number of VNFs running on virtual machines (or containers) hosted on multiple administrative cloud domains. Each CDN slice is administrated by only one coordinator that manages the different VNFs of the slice and ensures an effective communication among them. Fig.1 illustrates the architecture of the CDNaaS platform and its main components:

a) Orchestrator: The owner of a CDN slice logs into the orchestrator to manage the life-cycle of the CDN slice and its virtual resources, and perform actions including instantiation and termination of VNFs with desired flavors [19]. The orchestrator updates constantly the slice coordinator about any action made to VNFs under its control.

b) Slice-specific Coordinator: Every CDN slice has only one coordinator (i.e., VNF Manager [20]), functioning as its brain. It ensures the communication between VNF instances associated to a specific CDN slice. The owner of the slice

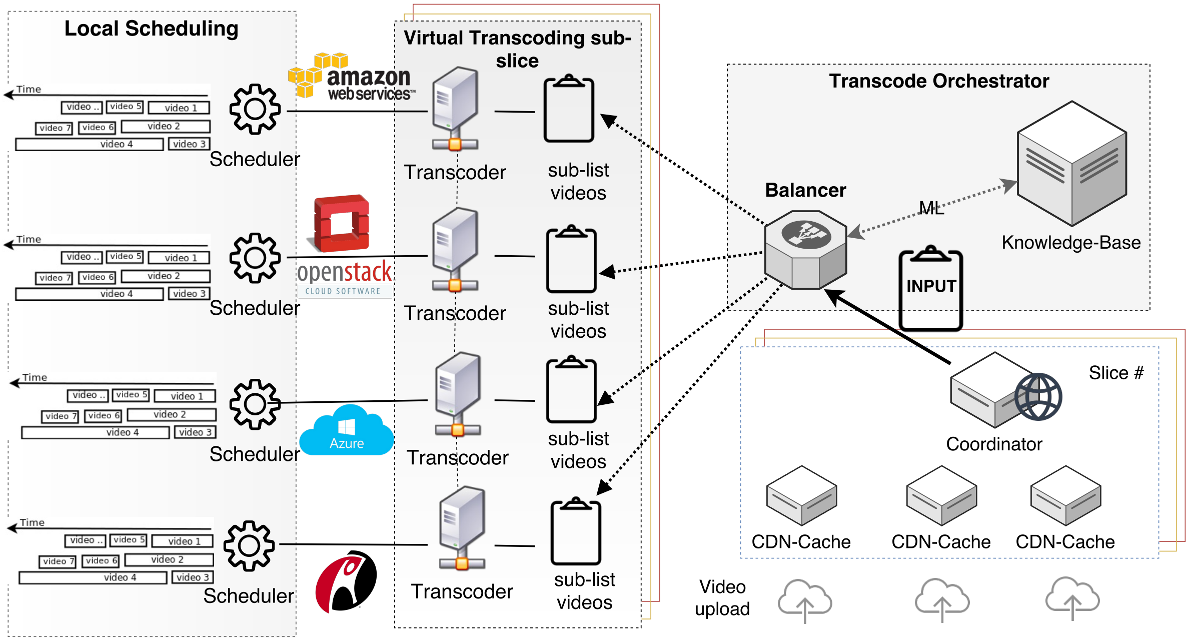


Fig. 2. Proposed Trasncoding Framework.

can manage the uploaded videos through the coordinator web interface, manages end-users and has access to the dashboard for monitoring the slice resources, content popularity and access statistics.

c) Virtual Cache: Basically, a CDN slice contains essentially a network of geographically dispersed cache servers. Each node caches static content and stores videos uploaded by users. A cache node stores the transcoding output files as well. When a user requests to watch a video at a desired resolution, the cache server nearest to the user will deliver the content, ensuring the shortest distance, therefore reducing the latency and providing the best QoS possible.

d) Virtual Transcoder: It is the server that is in charge of remote virtual transcoding. It consumes high computation resources. The transcoder server is always listening to the coordinator orders by the mean of a queuing system. It picks up the video from the relevant cache server, starts transcoding it and informs the coordinator about progress in the transcoding operation in real-time. The end-user will be notified if the operations are successfully completed.

e) Virtual Streamer: Essentially based on Nginx. It takes care of load balancing and receiving end-user requests for streaming specific videos [21] and redirecting the requests to proper cache servers to show the video content using available resolutions. The server also tracks the video accesses and sends them back to the coordinator for measuring statistics and analysis in order to improve the Business Intelligence of a CDN slice and understand more its customer needs and expectations. Traditionally, CDNs only push information down to the end users, operating and serving clients from the nearest edge server. However, in the envisioned CDNaaS platform, logs and access information are also pulled back from the edge server towards the respective CDN slice manager. This is essential to make delivery platform smarter in terms of $i)$ content popularity, $ii)$ VNF hit-ratio and $iii)$ Optimal VNF-placement [22].

IV. PROPOSED SOLUTION FRAMEWORK

This article examines potential stress points in the virtual transcoding system for the sake of improving the distribution of the workload associated with video transcoding across heterogeneous virtual transcoder nodes $T = \{t_1, t_2 \dots t_N\}$ in a CDN slice. Every VNF-Transcoder node is characterized by its flavor e.g, number of CPU cores, memory and internal queuing list of videos waiting to be transcoded. Looking at content creation and video encoding/transcoding and delivery, there are a number of factors that can dramatically affect the overall performance of the video streaming service. From one side, the profile of the data, itself, highly related to frame-rate, bit rate, quality, resolution and duration of videos adds complexity to potential bottlenecks in the system. From the other side, the performance depends on the behavior of the hosting virtual machines (VM) and that is based on their physical specifications such as CPU, memory and storage capacity.

Our proposed framework, depicted in Fig.2, provides management for transcoding high volumes of media files through heterogeneous transcoding nodes distributed over multiple cloud domains. The architectural components and operation logic of our proposed benchmarking system are depicted and elaborated in Fig. 2.

Knowledge Base (KB): Defined as a database used for knowledge sharing and management. It promotes the collection and retrieval of knowledge for artificial intelligence purpose. In this regards, our CDNaaS platform stores all previous transcoding results such as transcoding time, hosting VM resource usage and other information in order to construct a solid KB.

Transcode Orchestrator (TO): A component responsible for global optimization. It can be hosted on a separate virtual server dedicated for virtual transcoding orchestration. It consists of a KB and a Balancer. The main feature of this component is to distribute the arrival video uploaded to the CDN slice over multiple VNF-Transcoder nodes. The balancer agent retrieves the dataset from the KB, then generates a machine learning predictive model for future predictions (e.g., transcoding time, sojourn time). Basically, TO goes beyond standard load balancing mechanisms by optimally assigning transcoding jobs based on predictions of which node can most efficiently process a given job with an efficient use of virtual resources. The TO server tracks the performance of all transcoder nodes, learning which systems exhibit best performance and allocating a number of videos accordingly to optimize CPU and memory usages and reduce the estimated transcoding time.

Scheduler: A component responsible for local optimization. After receiving the assigned videos subset from the TO, the scheduler agent manages locally the scheduling of the transcoding process at the node level. The scheduler will also define the local transcoding process, either sequential or parallel transcoding or a combination of both. Then, it will predict the local transcoding time.

In order to achieve an optimal assignment, we first need to provide a detailed understanding of transcoding behavior through hundreds of benchmark experiments. We conduct our benchmark experiments in the following section with several heterogeneous cloud-based transcoder nodes, using various uploaded video durations.

V. Experimental Evaluations

In this section, we focus on benchmarking the performance of the VNF-Transcoder, based on FFMPEG software, by varying different parameters, i) Inputs: number of arrival videos and video durations, ii) Environment: processing capacity of the hosting VM. This benchmark results are stored in the Coordinator database forming a KB for future decisions. In the experiments, we fixed all video parameters, varying only the duration of the videos and the flavor of the hosting virtual machine. For this purpose, we have split the same large video file into several videos with different duration lengths (i.e., $5\,min$, $10\,min$, $15\,min$, ..., $90\,min$) to show the impact of the video duration on the performance. In this setup, four flavors were considered for our benchmarking study summarized in Table I. During our benchmark study we focused on transcoder performance in terms of processing time defined as the time required for transcoding a video, and sojourn time defined as the average time elapsed from the arrival of a video until its successful transcoding into outputs. Fig. 3 depicts the difference between processing time and sojourn time.

TABLE I
DEPLOYMENT FLAVORS

Flavor	Mini	Small	Medium	Large
# CPU	1	2	4	8
CPU (MHz)	4096	4096	4096	4096
RAM (GB)	3.86	3.86	3.86	3.86

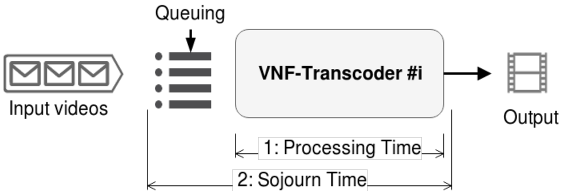


Fig. 3. Overview of the envisioned performance metrics.

A. Processing time of a video

1) Experiment and observations: The overall objective is to minimize the cost of video transcoding and to maximize the resource usage efficiency, to ultimately improve the affinity between QoE and price. We study the processing time of a set of newly arriving videos to our CDN slice. For the sake of optimization, both sequential and parallel transcoding are considered to gain a detailed understanding of the virtual transcoder node performance. We also track the CPU usage during the video transcoding. Then, the Scheduler should be able to define a local *schedule* of parallel/sequential transcoding at the transcoder node.

a) ***Sequential Transcoding****:* Fig. 4 shows that the transcoding time in case of the sequential approach is proportional to the duration of videos regardless the number of vCPUs of the hosting VM. Furthermore, the transcoding reduces while increasing the number of CPU cores. Hence, the transcoding time can be defined as a linear equation: $Transcoding\_Time = \lambda_i \cdot Video\_Duration$, whereby the slope λ is related only to the number of vCPUs.

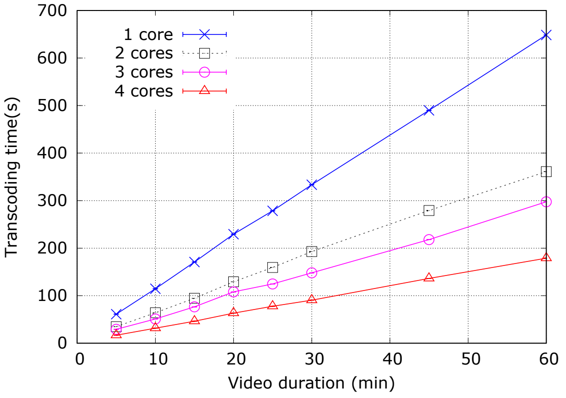


Fig. 4. Sequential transcoding operations in multi-core processors VNFs.

*b) **Parallel Transcoding**:* Fig. 7 shows the average processing time per video, whereby a set of videos with the same duration are transcoded simultaneously. From this figure, we can conclude two expected observations during parallel transcoding:

- i) The average processing time increases while increasing the number of CPU cores.
- ii) The average processing time increases proportionally with the duration of videos.
  Fig.7.b, with 4 CPU cores VM, shows that transcoding time y_4 can follow a linear regression $\Delta_{(4)}$ of video duration x_4, where: $\Delta_{(4)} : y_4 \simeq 5.x_4$.
  Fig.7.c, with 8 CPU cores VM, shows that transcoding time y_8 can follow a linear regression $\Delta_{(8)}$ of video duration x_8, where: $\Delta_{(8)} : y_8 \simeq 3.x_8$.
- iii) The average processing time is not predictable due to disturbance. Fig.7.a, with 1 CPU core VM, shows a disturbance while increasing the number of parallel videos. The figure does not show a relaxed (linear) system where the average processing time is nearly the same for a set of videos of the same durations. Which means that the 1 CUP core machine cannot handle 8 videos in parallel for transcoding.

*c) **Virtual resources usage**:* In Fig. 5, we plotted the average resource usages when using VMs with different number of CPU cores. The most trivial, yet interesting aspect of this figure is that when a hosting VM is powerful, it could handle a big number of videos. Accordingly, it uses almost the peak performance of computing resources. It is clear from Fig.5 that VM-8cores uses nearly 700% of CPU (equivalent to 100% of total processing resources) even while transcoding a big number of videos in parallel. While VM-1core decreases nearly to less than 50% when transcoding a big number of videos. Otherwise, transcoding only two videos in VM-1core machine, 95% of total processing resources are used.

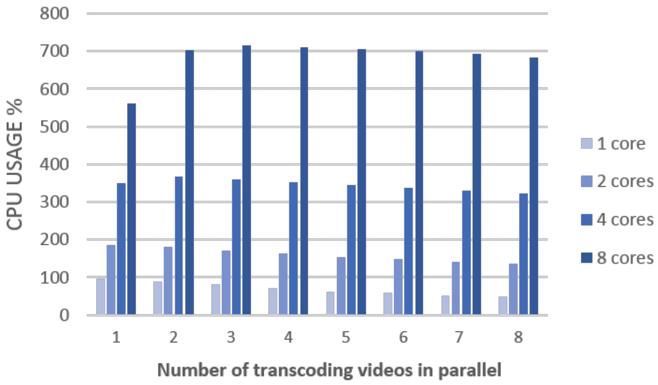


Fig. 5. CPU usage on multi-core processors VNFs

2) Discussions and findings: Firstly,in case of the sequential transcoding, we conclude that the processing time is linear and varies as a function of the video duration. Therefore, We are able to define linear equations and predict the estimated transcoding time based on the hosting machine resources and the video duration.

Secondly, in case of the parallel transcoding, we note that there is an optimal number of videos that can be simultaneously transcoded with the highest performance of the hosting VM. This optimal number depends on the VNF virtual resources. The transcoding time can be also predictable since it is approximately a linear function. However, if the videos exceed the optimal number, the CPU usage decreases and the transcoding time increases as well, as confirmed in both experiments Fig.7.a and Fig.5.

B. Mean Sojourn Time (MST) of video in the System

1) Experiment and observations: Fig. 6 shows the mean sojourn time of videos in the system in terms of video duration and the number of videos transcoded in parallel. There is a clear trend of increasing MST while increasing the video duration and number videos as well.

Comparing between the two transcoding approaches, the figure shows that the parallel transcoding approach (in blue) exhibits better performance than the sequential transcoding approach (in red) which means that videos during parallel transcoding pend less time in the the system as an average. Moreover, MST during sequential transcoding increases exponentially along with the duration of videos. However, it increases linearly with the duration of videos during the parallel transcoding.

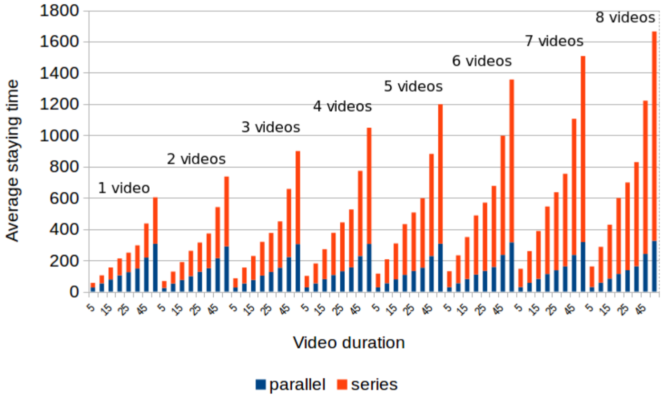
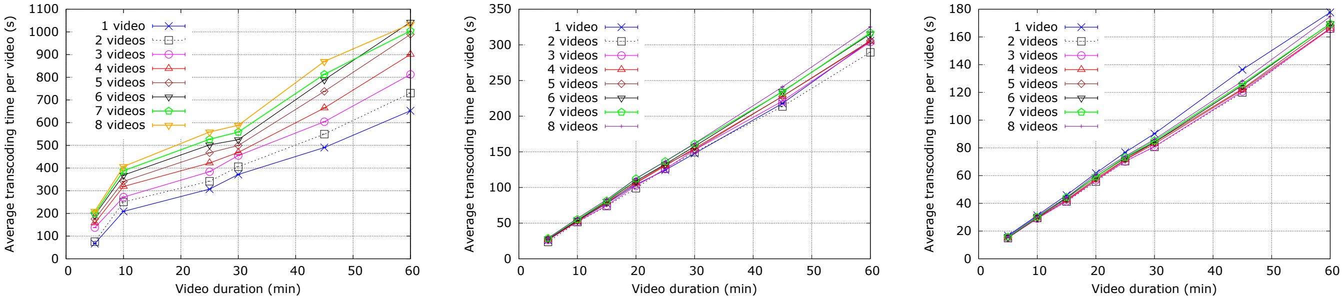


Fig. 6. Mean Sojourn Time per video in the System: (i) in sequence and (ii) in parallel

2) Discussions and findings: Parallel transcoding with an optimal number of videos that depends on the virtual resources of the hosting machine, provides a minimal sojourn time of videos in the system than a sequential transcoding. In the proposed framework solution, the TO will take into consideration these findings to find an optimal way of transcoding videos, either in a sequential or parallel with the an optimal local schedule avoiding any disturbance and maximizing the virtual resource usage. Based on the above results, the TO can assign a subset of videos over heterogeneous transcoder nodes. Then



(a) Average multi-transcoding time per video in 1-core machine

(b) Average multi-transcoding time per video in 4-cores machine

(c) Averge multi-transcoding time per video in 8-cores machine

Fig. 7. Experiment results: Average multi-transcoding time per video in multi-core processor machines.

the scheduler algorithm at the node level should be able to define an internal optimal schedule ensuring a minimal MST of videos in our distributed system. This shall reduce latency and the overall system's response time.

VI. Conclusion and Future Work

This paper introduced a CDNaaS platform that allows the creation of CDN slices across multiple administrative cloud domains and described the management and orchestration of VNFs. An extensive benchmark analysis was also conducted in order to study the virtual transcoding behavior in different cloud environments. The experiment results provided a knowledge base which continuously expands in every new transcoding process. Our future work consists on developing an efficient algorithm based on machine learning that uses the knowledge base to train the predictive model to find the optimal assignment of videos across heterogeneous transcoder servers with an optimal local schedule (sequential and parallel order). The main objective of the algorithm is to reduce the sojourn time of videos in a distributed system and improve the overall delivery time.

VII. Acknowledgment

This work was partially funded by the Academy of Finland Project CSN – under Grant Agreement No. 311654 and also partially supported by the European Unions Horizon 2020 research and innovation program under the 5G!Pagoda project with Grant Agreement No. 723172.

References

[1] Cisco Systems. (2017) Cisco VNI: 2016–2021.

[2] G. Gao, W. Zhang, Y. Wen, Z. Wang, and W. Zhu, "Towards cost-efficient video transcoding in media cloud: Insights learned from user viewing patterns," *IEEE Trans. Multimedia*, vol. 17, no. 8, pp. 1286–1296, 2015.

[3] G. Guanyu, Z. Weiwen, and Y. Wen, "Resource Provisioning and Profit Maximization for Transcoding in Clouds: A Two-Timescale Approach," *IEEE Transactions on Multimedia*, p. 19, Apr. 2017.

[4] S. Retal, M. Bagaa, T. Taleb, and H. Flinck, "Content Delivery Network Slicing: QoE and Cost Awareness," *Proc. IEEE ICC 2017*, May 2017.

[5] T. Taleb, A. Ksentini, and R. Jäntti, ""anything as a service" for 5g mobile systems," *IEEE Network*, vol. 30, no. 6, pp. 84–91, 2016.

[6] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[7] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies and solutions," *to appear in IEEE Communications Surveys and Tutorials*, 2018.

[8] O.-L. Jose, A. Pablo, and D. Lopez, "Network Slicing for 5G with SDN/NFV:Concepts, Architectures, and Challenges," *IEEE Communications Magazine*, May 2017.

[9] R. Peter and M. Christian, "Network Slicing to Enable Scalability and Flexibility in 5G Mobile Networks," *IEEE Communications Magazine*, Aug. 2017.

[10] T. Bin, W. Jun, and Y. Li, "Analog Coded SoftCast: A Network Slice Design for Multimedia Broadcast/Multicast," *IEEE Transactions on Multimedia*, Oct. 2017.

[11] A. Nakao, P. Du, Y. Kiriha, F. Granelli, A. A. Gebremariam, T. Taleb, and M. Bagaa, "End-to-End Network Slicing for 5G Mobile Networks," *Journal of Information Processing*, vol. 25, no. 1, pp. 153–163, Jan. 2017.

[12] T. Taleb, B. Mada, M. I. Corici, A. Nakao, and H. Flinck, "PERMIT: network slicing for personalized 5g mobile telecommunications," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 88–93, 2017.

[13] Z. Haijun, L. Na, and C. Xiaoli, "Network Slicing Based 5G and Future Mobile Networks: Mobility, Resource Management, and Challenges," *IEEE Communications Magazine*, Aug. 2017.

[14] G. Guanyu, H. Han, and W. Yonggang, "Resource Provisioning and Profit Maximization for Transcoding in Clouds: A Two-Timescale Approach," *IEEE Transactions on Multimedia*, Apr. 2017.

[15] L. Xiangbo and S. Mohsen Amini, "Cost-Efficient and Robust On-Demand Video Transcoding Using Heterogeneous Cloud Services," *IEEE Transactions on Parallel and Distributed Systems*, Oct. 2017.

[16] S. Dutta, T. Taleb, P. A. Frangoudis, and A. Ksentini, "On-the-Fly QoE-Aware Transcoding in the Mobile Edge," in *Proc. 2016 IEEE Global Communications Conference (GLOBECOM), Washington, DC USA*, Dec. 2016, pp. 1–6.

[17] H. Mohammad, T. M. Lok, and W. Zhi, "Cost-Effective Low-Delay Design for Multiparty Cloud Video Conferencing," *IEEE Transactions on Multimedia*, Dec. 2017.

[18] T. Taleb, "Toward carrier cloud: Potential, challenges, and solutions," *IEEE Wireless Communications*, vol. 21, no. 3, pp. 80–91, 2014.

[19] I. Afolabi, A. Ksentini, M. Bagaa, T. Taleb, M. Corici, and A. Nakao, "Towards 5g network slicing over multiple-domains," *IEICE Transactions*, vol. 100-B, no. 11, pp. 1992–2006, 2017.

[20] F. Z. Yousaf and T. Taleb, "Fine granular resource-aware virtual network function management for 5g carrier cloud," *IEEE Network Magazine*, vol. 30, no. 2, pp. 110–115, 2016.

[21] K. H. Tarik Taleb, "Ms2: A novel multi-source mobile-streaming architecture," *IEEE Trans. on Broadcasting*, vol. 57, no. 3, pp. 662–673, 2011.

[22] T. Taleb, M. Bagaa, and A. Ksentini, "User mobility-aware virtual network function placement for virtual 5g network infrastructure," in *Proc.2015 IEEE International Conference on Communications, ICC 2015, London, United Kingdom, June 8-12, 2015*, 2015, pp. 3879–3884.