Performance Comparison of Transcoding and Bitrate-aware Caching in Adaptive Video Streaming

Wenjie Li*, Sharief M.A. Oteafy†, and Hossam S. Hassanein*
*School of Computing
Queen's University, Kingston, ON, Canada
Email: {liwenjie,hossam}@cs.queensu.ca
†School of Computing
DePaul University, Chicago, IL, USA
Email: soteafy@depaul.edu

***Abstract*—Video traffic is growing in dominance in today's Internet, prompting new challenges in timely delivery of video content. As Dynamic Adaptive Streaming (DAS) is becoming the *de facto* paradigm for video delivery, there is growing evidence on how caching improves users' Quality of Experience (QoE) in DAS. However, there is no consensus on how to maximize the utilization of in-network caching resources. Specifically, there are conflicting proposals on the impact of caching based on its distance (in hops) from the network edge. That is, contrasting ubiquitous network-wide caching to edge-caching. Supporters of the ubiquitous caching paradigm propose bitrate-aware caching schemes for optimizing video streaming, while counter-proposals suggest that edge-caching only the highest bitrate with online transcoding, may offer superior performance to ubiquitous caching. In this paper, we answer a contentious question: Can transcoding at the edge outperform bitrate-aware ubiquitous caching for DAS? We devise an extensive simulation environment using NS-3 to contrast both paradigms, experimenting with different bandwidth fluctuation patterns, under the FESTIVE user-based bitrate adaptation protocol. Caching performance was evaluated under five established QoE metrics, gauging delivered video quality, playback freezing and bitrate oscillation. We further assume zero processing delay for online transcoding at the network edge, to contrast to an upper bound performance from the edge-caching paradigm. Our experiments demonstrate that neither transcoding nor bitrate-aware caching offer a silver bullet for all cases. We present our insights on networking scenarios where each model would dominate in performance, and present our concluding remarks on their development.**

***Index Terms*—In-network Caching; Edge Caching; Dynamic Adaptive Streaming; Performance Analysis; Guided Designs.**

I. Introduction

The rapid evolution of video dissemination and usage on the Internet is mandating novel paradigms to scale with the volume of traffic and heterogeneity of users. By 2021, video content is projected to dominate over 80% of global Internet traffic [1]. This dramatic growth in video traffic has motivated many developments in application-layer standards, yielding Dynamic Adaptive Streaming over HTTP (DASH) [2] as a leading contender in delivering video content over unstable channel conditions.

At its core, DASH adopts three fundamental features in its operation: video content is first partitioned into equal duration segments, all segments are encoded at multiple bitrates, and an adaptive control algorithm is applied to the consumer side to request the highest possible quality given estimates of real-time network bandwidth. To maximize consumers' satisfaction of video streaming, there has been a lot of research on modelling the behaviour of DASH [3] and improving this adaptation control mechanism [4]. However, DASH intrinsically is an application-layer solution, and thus cannot resolve the underlying network scalability problem or ease bottlenecks caused by massive video traffic.

A recent shift towards Information-Centric Networks (ICNs) [5] is a promising solution from a network design perspective. This new paradigm adopts a *Publish-Subscribe* model, along with salient features such as content-host decoupling, dynamic request forwarding and in-network caching. Among all these features, in-network caching received the most attention, as it has been shown in many studies [6]–[8], that it can significantly leverage users' Quality of Experience (QoE).

Although the importance of in-network caching is well recognized, there is no consensus on how to manage these resources to maximize their utilization. To date, two main paradigms have been explored, one where caches are distributed across the entire network or the other where caching nodes are only allocated at the network edge. Evidently, both models have their own merits. Ubiquitous caching on network intersections would effectively reduce traffic load in the core network, but miss the chance of satisfying requests closer to consumers. Edge caches, on the other hand, would cause the least access delay, but at the same time result in a higher degree of cache redundancy and require more caching capacity.

Ubiquitous and edge caching are both adapted to cater to ever-expanding video streaming applications. With recent advancements in mobile/edge computing, computational resources are moving from the Cloud to the network edge, which enables on-the-fly video transcoding at edge caches. For example, a representative transcoding paradigm [9] describes that network edge nodes may only cache the highest quality content and *transcode* requests to lower qualities when such video requests arrive. Ubiquitous caching, instead, relies on bitrate-aware caching schemes to enhance multi-bitrate streaming. For example, smooth playback can be achieved by safeguarding a

978-1-5386-8088-9/19/$31.00 ©2019 IEEE

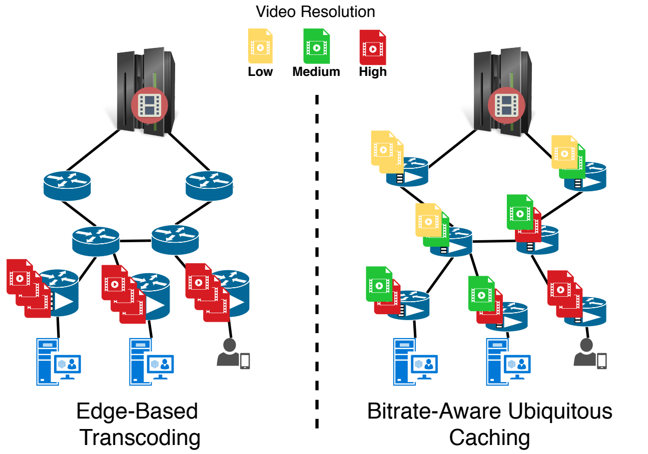


Fig. 1. Contrasting the caching behavior of ubiquitous bitrate-aware caching vs. Edge-based caching with transcoding, for DAS. In the latter paradigm, more caching resources need to be allocated at the edge to cater for storing most of the popular video segments at their highest bitrates.

network of caches for particular bitrates along each forwarding route [8]. However, deciding which one of these paradigms is superior in performance, has been a long-standing question. The operational difference between these two caching paradigms are depicted in Fig. 1.

The contributions of this paper are as follows: 1) We address the lack of comparison between the impact of transcoding and bitrate-aware caching on consumers' QoE. To highlight the potential gain of transcoding at the edge [9], we assume zero processing delay of transcoding, to cater for an upper bound performance that the edge caching paradigms may achieve. This performance is compared against *RippleFinder* [8] as the best known bitrate-aware caching scheme that delivers a comprehensive improvement on users' QoE. 2) We build an evaluation framework using NS-3 based simulator ndnSIM [10], where the leading FESTIVE protocol [11] is implemented to provide realistic consumer-side bitrate adaptation, and we evaluate consumers' QoE under different bandwidth fluctuation patterns. 3) Since neither paradigms achieve a landslide win for all scenarios, we provide recommendations on design decisions that aid the selection of caching paradigms, and highlight scenarios where each paradigm excels.

The remainder of this paper is organized as follows. In Section II we present pertinent background and related work, elaborating on state-of-the-art caching models, and the role of transcoding at the edge. To compare these two paradigms, we explain our experimental setup and performance evaluation in Section III. We build on our insights from these experiments and elaborate on design considerations for building caching models under either paradigm in Section IV, and we conclude in Section V and present potential directions moving forward.

II. Background & Related Work

A. Ubiquitous Network Caching

With the advent of next-generation routers that are capable of caching, there are a number of networking paradigms considering a wider-adoption of caching at its routers. While ICNs have this feature inherent in its design, the expansion of adaptive Content Distribution Networks, with more agile models for allocating caching resources across networks, are opening up new frontiers in network-wide caching. Ubiquitous caching [5] is a fundamental feature of ICN. Due to the decoupling of content and location in ICN naming mechanisms, information is not bound to a particular host and can be retrieved from anywhere in the network.

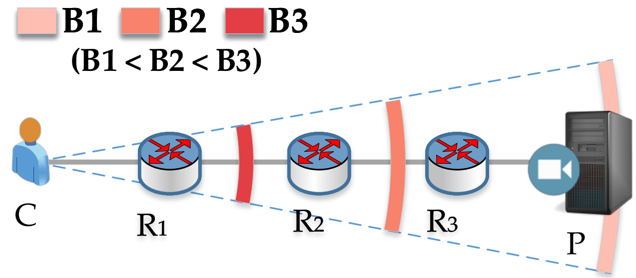


Fig. 2. Video content with different bitrates are safeguarded at different hop distances from a consumer, under the *RippleFinder* paradigm

The interaction between in-network caching and bitrate adaptation has attracted the attention of researchers, leading to studies on bitrate-aware caching schemes. Previous work that build on video throughput optimization [12]. As video throughput is a key metric of DASH adaptation, a higher video throughput would possibly trigger a video quality upgrade. Araldo et al. [13] attempted to build a more direct relationship between video quality and cache placement of multiple versions of video content. The work of *RippleFinder* [8] is a further step towards comprehensive improvement on users' QoE. As shown in Fig. 2, *RippleFinder* arranges a network of caches along the forwarding path to cooperatively cache different bitrates of video content. For example, the highest quality B_3 is catered at router R_1, bitrate B_2 at router R_2, and lowest quality B_1 at the farthest router R_3 from the consumer. It is demonstrated that by pushing low-bitrate content towards the core and safeguarding cache space for high-bitrate content at the edge, it is possible to achieve enhanced video quality, along with less video stream stalls and reduced bitrate oscillations.

RippleFinder is a leading caching scheme that takes into consideration the real-time interaction between consumer-side adaptation and in-network caches. In addition, *RippleFinder* also achieves improvement on multiple QoE metrics that are fundamental in QoE models [14]. Previous work (e.g., [7], [12], [13]) either focuses solely on improving video quality, or assume simplified user behavior that fails to mimic the realistic adaptive streaming scenarios. We choose *RippleFinder* as a benchmark and representative of bitrate-aware ubiquitous caching for comparison, as described further in Section III.

B. Transcoding on the Edge

Despite studies that support ubiquitous caching, recent proposals explored an alternative model. Fayazbakhsh et al. [15] discovered that edge caching deployment provides the same

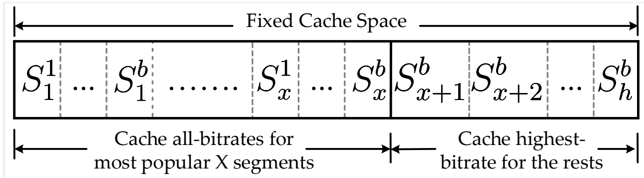


Fig. 3. Illustration of transcoding scheme [9]. All-bitrate versions are cached for the most popular x video content, while only the highest version is stored for the rest.

performance as ubiquitous caching. Their discovery was reinforced in [16], where an analytical model derived a similar result as [15]. However, these observations are constrained to simplistic Least Recently Used (LRU) cache replacement of non-video content. We argue that these results cannot be generalized for video-caching models that explore the effectiveness of edge caching under 1) varying video traffic and 2) popularity-based caching models.

The potential of edge caching shown in previous studies then motivates transcoding at the network edge to enhance adaptive streaming service. Jin et al. proposed the adoption of partial transcoding [9], as shown in Fig. 3, as a hybrid compromise. In this model, each edge node selectively caches all bitrate versions for a few popular video content, and only keeps the highest quality for the rest (as long as the cache capacity allows it). As a result, the partial transcoding ratio becomes a critical parameter which decides how to divide the caching space for these two different usages. This ratio is derived by minimizing the total *Cost* of caching system.

There are three types of *Cost*s: *Storage*, *Transcoding* and *Bandwidth*. The *Storage Cost* is charged by allocating cache space to each edge node. In our comparison, as we always allocate the same total cache space to both edge caching and ubiquitous caching, this type of cost is omitted in the remainder of this paper. The *Transcoding Cost* is incurred when the highest bitrate content is transcoded into any lower bitrate version. The *Bandwidth Cost* is triggered by retrieving content from a video producer when a cache miss occurs at the network edge.

The optimal partial transcoding ratio then depends on the trade-off between these three types of costs. However, different cost models, which focus on various performance metrics (such as energy consumption, access delay, or throughput), would result in far different results. For example, CPU usage becomes a significant component of *Transcoding Cost* in [17]. Finding a generic cost model for transcoding is still an open issue, as it varies from the underlying hardware, topology and so on. In this work, we assume zero transcoding cost/delay as a benchmark. With this assumption, the optimal partial transcoding ratio would be 1, i.e., using the entire caching space to store only the highest bitrate.

III. Transcoding vs. Bitrate-Aware Caching: Evaluation and Results

We compare the performance of caching under the ubiquitous and edge caching paradigms, via simulation. One of the important observations in our simulations is that conventional cache metrics (e.g., cache hit ratio) are not ideal for measuring streaming performance. Our evaluations are conducted on the Named Data Networking (NDN) [18], as a favored representative architecture that implements ICN primitives.

TABLE I
Simulation Parameters

NDN	
Number of video files	250
Number of video segments per file	40
Number of NDN nodes	16
Video segment playback time	4 sec
Number of video consumers	32
Request interval on video file/session (sec)	400
Skewness factor (α)	1.2
Content store size percentage (ω)	0.02
FESTIVE	
Drop Threshold	0.8
Combine Weight	8

A. Simulation Setup

We build an NDN environment using the NS-3 based simulator ndnSIM [10]. Caching space is either distributed evenly across all NDN nodes or only at edge nodes to represent both caching paradigms. We ensure equal cache capacity between edge and ubiquitous caching. This total capacity is allocated proportional to the size of all video content provided by the video producer, formally as $\sum_{n=1}^{N}\sum_{b=1}^{B} S_n^b * \omega$ where S_n^b is the size of a video segment with index n encoded at bitrate b. N is the number of video segments, and B is the number of video encodings. ω is a control parameter that enables us to examine the caching performance under different cache sizes.

Consumer-side adaptation is simulated via our implementation of FESTIVE [11]. FESTIVE is a highly-cited approach, and a representative of throughput-based adaptation control algorithm. A video session would be triggered following a Poisson process, where consumers' interests on video content are captured by a *Zipf* distribution (controlled via skewness parameter α). Once a video session is initialized, video segments within the requested video file are retrieved under control by FESTIVE. Video files are 160 seconds in duration and are divided into 4-second segments. Each video segment is prepared at 1, 2.5, 5, and 8 Mbps, which are recommended encoding bitrates by YouTube [19]. The simulation parameters are listed in Table I.

We evaluate *RippleFinder* [8] and the transcoding model in [9] (which we name as *Transcode*). *RippleFinder* is a cache placement scheme, where its caching decisions are updated until a steady state is reached. We assume zero transcoding cost/delay for *Transcode* to highlight the upper bound performance of transcoding at the network edge. In addition, we evaluate *Cache Everything Everywhere* (CE2) with Least Frequently Used (LFU) replacement. Although CE2 is not designed specifically for adaptive streaming, it is a widely used benchmark for ubiquitous caching. LFU caters to content

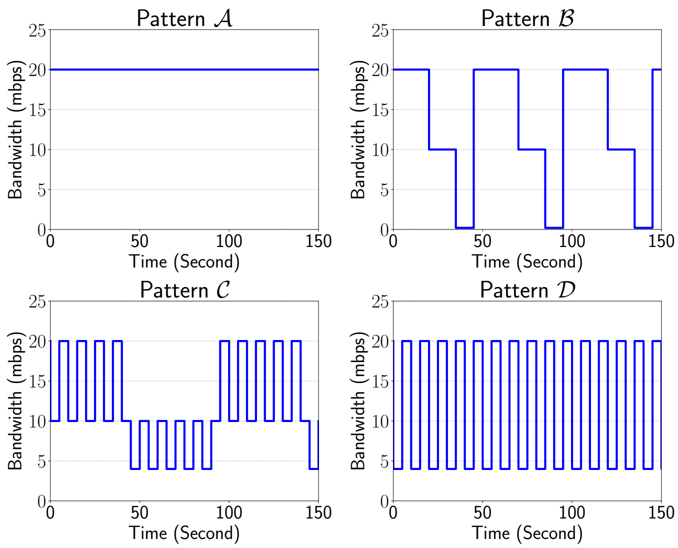


Fig. 4. 'Last-mile' bandwidth fluctuation pattern

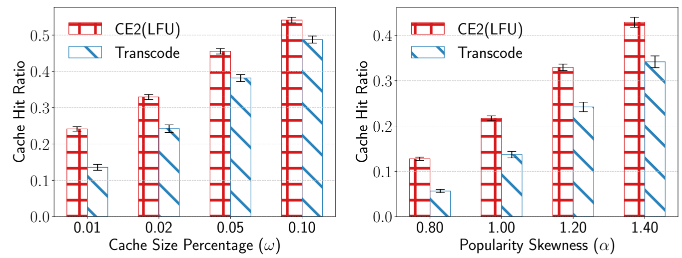


Fig. 5. Cache hit ratio under bandwidth pattern $\mathcal{A}$

popularity and outperforms LRU. We also test a variation of *CE2* where only edge nodes are allocated cache capacity, along with LFU for content replacement. We name this approach as *EdgeOnly* to represent generic edge caching. All results are presented at a 95% confidence level.

A random topology is generated by BRITE [20] to mimic a realistic streaming scenario [21]. In this topology, we carefully chose a video producer such that the hop distance between any consumer and the producer ranges from 3 to 6. This variation on hop distance would cause different video access delay by consumers. We choose in-network link capacity at 20 Mbps, and the 'last-mile' link bandwidth varies by fluctuation patterns as we detail in III-B. As a result, the highest bitrate (8 Mbps) cannot be retrieved directly from the producer and must be provided by caches. We choose this relatively small link capacity to examine the performance that is enhanced by caching policies.

B. Bandwidth Fluctuation Pattern

We investigate different bandwidth variation patterns on the 'last-mile' link between each consumer and their edge node, to mimic wired and wireless networks. Three patterns were discovered in studies on real measurements for mobile users [22], in addition to a stationary pattern for benchmarking. Thus, we adopted four variation patterns are evaluated as shown in Fig. 4. Throughout our experiments, we discovered that the variations in performance of caching schemes under Patterns $\mathcal{C}$ and $\mathcal{D}$ were not statistically significant. We thus opted to present results under Pattern $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ only.

C. Cache Hit Ratio

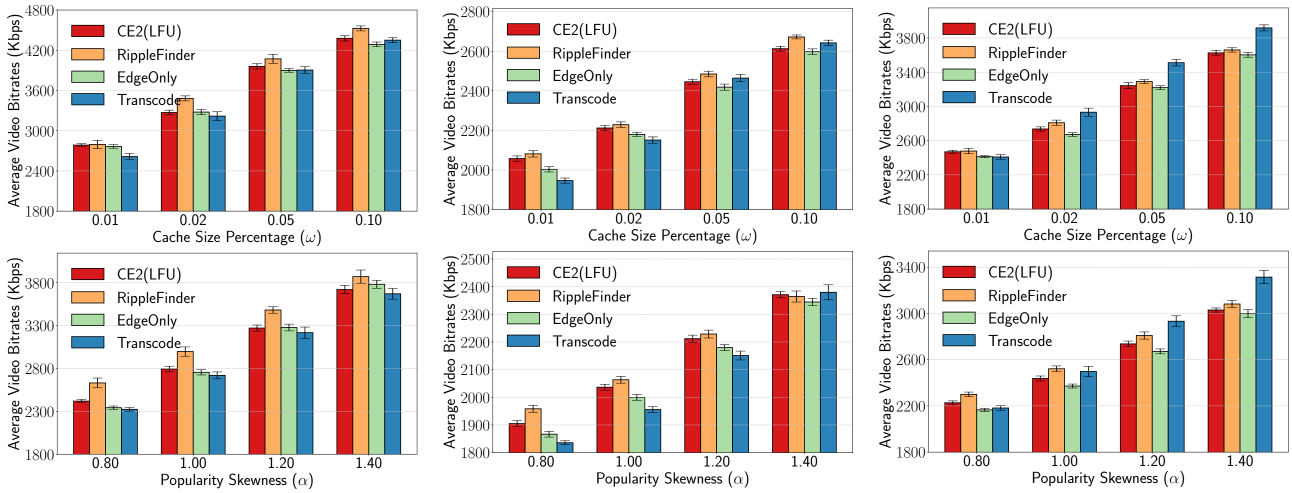
Cache hit ratio is a standard metric to evaluate the performance of caching schemes. As shown in Fig. 5, we observe even the baseline *CE2* with LFU outperforms *Transcode*, because of high cache redundancy caused by edge caching. This relationship remains the same no matter the bandwidth fluctuation pattern, cache capacity or content popularity skewness. However, our following observations on QoE contradict the current result, where *Transcode* outperforms *CE2* across almost every QoE metric that we examined. As the effectiveness of caching for video streaming must be verified by consumers' QoE, QoE metrics are thus more direct indicator of system performance. Cache hit ratio itself, as a conventional cache metric, has critical flaws when measuring schemes particularly for video streaming. As cache hit ratio cannot distinguish 'where' this hit occurs, cache hits at the edge or within the network can cause significantly different video throughput that alters the behaviour of consumer-side bitrate adaptation, impacting users' QoE.

D. QoE Metrics

DASH industry forum has published a standard set of QoE metrics [23]. In our experiments, we selectively adopt three metrics from the standard set, *Average Video Bitrate*, *Rebuffer Percentage* and *Bitrate Switch Count*. Other metrics, such as *Rebuffer Count* or *Bitrate Switch Rate* are also evaluated but not reported, since they either share a similar trend with presented metric or the performance difference (caused by caching) is insignificant.

1) Average Video Bitrate: This metric represents the average video quality that consumers request among all video sessions. Results are grouped by bandwidth fluctuation pattern, and in each group we present the performance across cache capacity and content popularity skewness. As shown in Fig. 6a, transcoding at the network edge has no advantage over ubiquitous caching with constant bandwidth at the 'last-mile' link (pattern $\mathcal{A}$). The video quality difference between *CE2* and *Transcode* was statistically insignificant. Instead, *RippleFinder* delivers the highest quality to consumers, which reinforces bitrate-aware caching as the superior paradigm under pattern $\mathcal{A}$. The reason for this is the case is that *Transcode* only maximizes its utilization of cached content when it assumes that ' requests to all bitrates are equally likely'. However, a constant link capacity fails to provide enough bandwidth fluctuation, and video throughput variations are mainly caused by in-network traffic congestion. Thus, this assumption is not always satisfied across all video segments, which diminishes the performance of *Transcode*.

Fig. 6b presents delivered video quality when there is intermittent connection failure. The performance trend is similar



(a) Pattern $\mathcal{A}$ (b) Pattern $\mathcal{B}$ (c) Pattern $\mathcal{C}$

Fig. 6. *Average Video Bitrate* across cache size and popularity skewness.

to pattern $\mathcal{A}$. *Transcode* still suffers from the fact that not all bitrates are frequently requested for each segment as requests for lower quality content are dominating under this pattern $\mathcal{B}$. Besides, it is noticeable that at low capacity (or low popularity skewness), *EdgeOnly* delivers even higher video quality than *Transcode*. This means online transcoding (even with zero processing delay) brings no benefits to system performance. It is because *Transcode* requires caching only the highest quality no matter what are the frequently requested bitrates. When requests for low quality content are dominating, *Transcode* forces edge caches to store the highest quality segments, which not only consume more caching space than needed (for lower quality content) but also reduce the amount of video content that can be served by the cache.

Transcode presents superior performance mainly under pattern $\mathcal{C}$ as shown in Fig. 6c. This is due to bandwidth fluctuation between 4 Mbps and 20 Mbps creating more chances for all encoding bitrates to be frequently requested, which boosts the performance of *Transcode*. At large cache volume, this enhancement by *Transcode* is significant since large caching space gains more advantage from the efficient cache utilization of *Transcode* that allows to cache only the highest quality video segment.

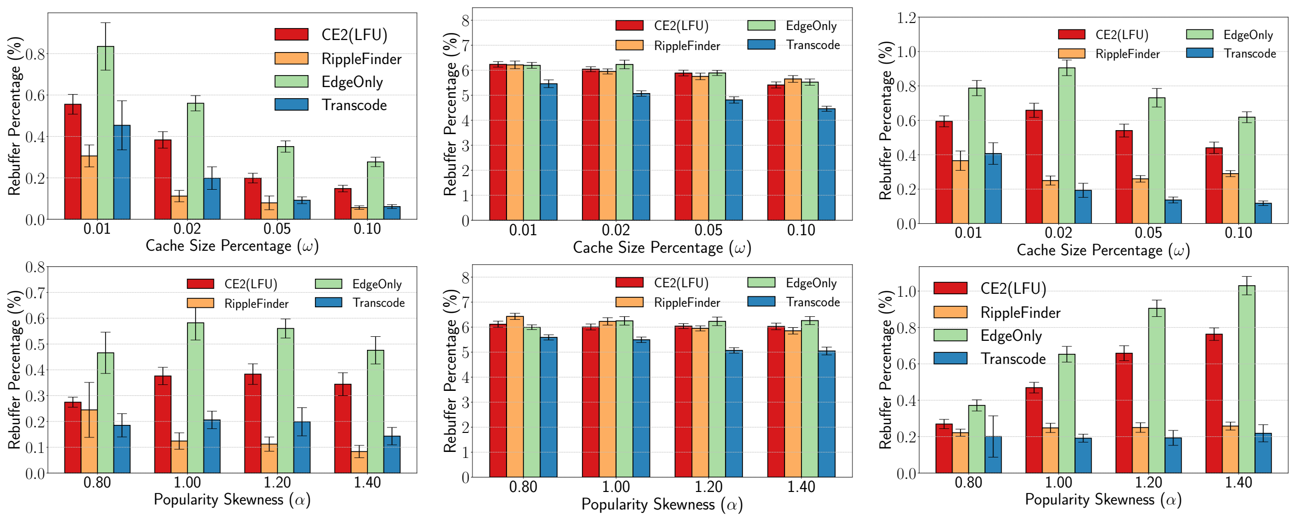
2) Rebuffer Percentage: This metric is defined as the time spent in a video freezing state over the active time of a video session. It is noticeable that *EdgeOnly* causes a higher chance of video freezing than ubiquitous caching scheme *CE2*. Intuitively, as a representative of edge caching, *EdgeOnly* would satisfy requests closer to the consumer, which should lead to less video access delay than *CE2*. This counter-intuitive result is affected by consumer-side bitrate adaptation. Cache hits on edge caches have a higher chance than in-network caches to trigger a video quality upgrade. However, this upgrade is harmful once high quality content is not sustainable. The follow-up cache misses would require consumers to retrieve content directly from the producer, which results in a even longer access delay and a higher chance of video stalling.

In contrast, *Transcode* performs better than *EdgeOnly*, since it can satisfy video requests for any version of the content, with a constant cost of caching space (by storing only the highest version). *RippleFinder* achieves less video freezing than *Transcode* under constant link capacity as shown in Fig. 7a. *Transcode* re-gains its advantage under fluctuated link capacity in Fig. 7c. Under an intermittent network connection, all tested schemes suffer from significant video stream stalls (although *Transcode* may perform slightly better). The impact of caches on video stalling is thus negligible under pattern $\mathcal{B}$.

3) Bitrate Switch Count: This metric is defined as the number of times the requested video bitrate changed during a video session. It is evident that *EdgeOnly* causes more bitrate oscillations than *Transcode*, which indicates that the edge caching paradigm alone is not the key contributor to smooth video playback. We suspect our assumption of zero transcoding delay is the main reason for such performance, as the same video throughput is guaranteed across all versions of popular video content. Thus, a higher degree of bitrate oscillation is expected when *Transcode* is applied under a realistic setting that factors in the inevitable processing delay, which varies as the highest quality version is transcoded to different bitrates. In addition, *RippleFinder* can still achieve a similar *Bitrate Switch Count* as *Transcode* under bandwidth pattern $\mathcal{A}$ and $\mathcal{C}$. As shown in Fig. 8a, *RippleFinder* even matches the upper bound performance of *Transcode* at high popularity skewness (e.g., at $\alpha = 1.2$ or 1.4). This result highlights the potential of bitrate-aware ubiquitous caching schemes in controlling bitrate oscillations.

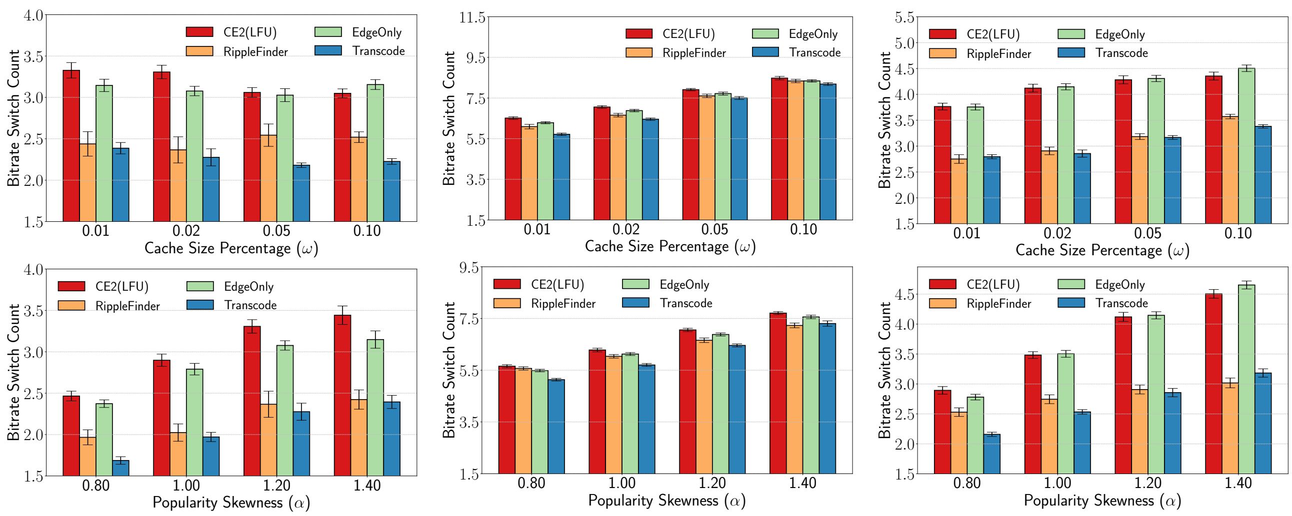
IV. Insights on Ubiquitous caching Vs Edge-transcoding

Throughout our experiments, we discover that neither transcoding nor bitrate-aware caching present blanket solutions across all bandwidth fluctuation patterns. When consumers are



(a) Pattern $\mathcal{A}$ (b) Pattern $\mathcal{B}$ (c) Pattern $\mathcal{C}$

Fig. 7. *Rebuffer Percentage* across cache size and popularity skewness.



(a) Pattern $\mathcal{A}$ (b) Pattern $\mathcal{B}$ (c) Pattern $\mathcal{C}$

Fig. 8. *Bitrate Switch Count* across cache size and popularity skewness.

dedicated to a fixed bandwidth or connect via a wired link, bitrate-aware caching is better suited to facilitate streaming services. *RippleFinder* outperforms even the upper bound performance of transcoding with regards to video quality and playback freezing, while almost matching *Transcode* in bitrate oscillation.

However, when consumers are connecting via mobile devices, transcoding has a great potential in overall QoE improvement. In reality, as the processing delay of transcoding exists and varies by case, it is necessary to evaluate against best-known bitrate-aware caching schemes to validate this advantage .

We also noticed the performance of *Transcode* diminishes at low cache capacity or low popularity skewness. Edge caching would cause a higher degree of cache redundancy and the online-transcoding assumption may not hold in many scenarios. Both of these factors undermine cache utilization, which impacts performance when the available cache capacity is limited. We thus emphasize the edge of transcoding based approaches when high cache capacities are guaranteed, coupled with minimal-contention on caching resources. Same reason applies to high popularity skewness, where few video content is popular and competes for cache.

In designing caching models, it is important to take into consideration the impact of transcoding on computing and networking resources. That is, simply assuming that edge nodes are computationally more equipped than core routers, does not warrant an assumption of superior performance that could handle both computationally-intensive transcoding along with edge computing requirements.

As we previously mentioned, transcoding also carries a significant storage and communication cost, which should be factored into the design. In addition, even if edge nodes have significant caching capacity, the notion that they can equate to the caching capacity found in ubiquitous caching models is not always true. Thus, relying on edge caching alone inherently sacrifices potential space that could benefit more content.

V. Conclusions and Future Work

In this work, we addressed the seldom investigated comparison between the impact of transcoding and bitrate-aware caching on consumers' QoE. We conducted extensive experiments to examine their performance across various bandwidth patterns, cache capacity, and popularity skewness measures. Our experiments demonstrated that conventional metrics, such as cache hit ratio, are not ideal indicators of video-related system performance, as they often contradict QoE performance benchmarks. We thus adopted industry-leading benchmarks in quantifying QoE, and accordingly contrasted the performance of both paradigms.

Even under the assumption of zero transcoding delay, we discovered that bitrate-aware caching can often match or even outperform the upper bound performance of transcoding. Based on our observations, bitrate-aware caching is more suitable to serve consumers with fixed and dedicated link capacity when cache resources are constrained. Online transcoding is more suitable to serve mobile consumers when there is a significant amount of caching space and computational power at the edge, in excess to the operational needs of the omnipresent edge computing architecture.

One of the important future directions in this quest to evaluate different caching models, is investigating user-centric video request patterns in edge networks. We are in need of more representative models for video behaviour in mobile environments that capture video viewing activity, along with inherited assumptions on omnipresent edge capabilities. Evidently, the co-existence of transcoding functionalities with other edge tasks is a problem that requires further investigation. This is especially important when edge computing architectures are tasked with significant offloading and migration requirements, which may hinder their responsiveness to time-sensitive video traffic management.

One of the other challenges that should be addressed is the evident impact of Edge caching on video stalling, especially under varying user request patterns. It is imperative to investigate the impact of consumer-side bitrate adaption on overall video delivery, especially when frequent cache-misses occur.

Acknowledgment

This work is supported by Natural Sciences and Engineering Research Council of Canada (NSERC), through grants RGPIN-2017-06902 and RGPIN-2014-06587.

References

[1] Cisco, "Cisco visual networking index: Forecast and methodology, 2016–2021," 2017.

[2] MPEG, "DASH," http://dashif.org/mpeg-dash.

[3] R. Jia, Z. Liu, X. Wang, X. Gan, X. Wang, and J. J. Xu, "Modeling dynamic adaptive streaming over information-centric networking," *IEEE Access*, vol. 4, pp. 8362–8374, 2016.

[4] J. Kua, G. Armitage, and P. Branch, "A survey of rate adaptation techniques for dynamic adaptive streaming over http," *IEEE Communications Surveys & Tutorials*, 2017.

[5] A. Ioannou and S. Weber, "A survey of caching policies and forwarding mechanisms in information-centric networking," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2847–2886, 2016.

[6] R. Grandl, K. Su, and C. Westphal, "On the interaction of adaptive video streaming with content-centric networking," in *IEEE International Packet Video Workshop (PV)*, 2013.

[7] D. Lee, C. Dovrolis, and A. Begen, "Caching in http adaptive streaming: Friend or foe?" in *ACM Network and Operating System Support on Digital Audio and Video Workshop*, 2014, pp. 31–36.

[8] W. Li, S. Oteafy, M. Fayed, and H. S. Hassanein, "Bitrate adaptation-aware cache partitioning for video streaming over information-centric networks," in *IEEE Conference on Local Computer Networks (LCN)*, 2018.

[9] Y. Jin, Y. Wen, and C. Westphal, "Optimal transcoding and caching for adaptive streaming in media cloud: An analytical approach," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 12, pp. 1914–1925, 2015.

[10] A. Alexander, I. Moiseenko, and L. Zhang, "ndnsim: Ndn simulator for ns-3," *Technical Report NDN-0005*, 2012.

[11] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive," in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*. ACM, 2012, pp. 97–108.

[12] W. Li, S. Oteafy, and H. Hassanein, "Rate-selective caching for adaptive streaming over information-centric networks," *IEEE Transactions on Computers*, vol. 66, no. 9, pp. 1613–1628, 2017.

[13] A. Araldo, F. Martignon, and D. Rossi, "Representation selection problem: optimizing video delivery through caching," in *IFIP Networking Conference*, 2016, pp. 323–331.

[14] Z. Duanmu, A. Rehman, and Z. Wang, "A quality-of-experience database for adaptive video streaming," *IEEE Transactions on Broadcasting*, vol. 64, no. 2, pp. 474–487, June 2018.

[15] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, and S. Shenker, "Less pain, most of the gain: Incrementally deployable icn," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, 2013, pp. 147–158.

[16] A. Dabirmoghaddam, M. M. Barijough, and J. Garcia-Luna-Aceves, "Understanding optimal caching and opportunistic caching at the edge of information-centric networks," in *Proceedings of the 1st ACM conference on information-centric networking*, 2014, pp. 47–56.

[17] H. Zhao, Q. Zheng, W. Zhang, B. Du, and H. Li, "A segment-based storage and transcoding trade-off strategy for multi-version vod systems in the cloud." *IEEE Trans. Multimedia*, vol. 19, no. 1, pp. 149–159, 2017.

[18] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang *et al.*, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[19] YouTube Help, https://support.google.com/youtube/answer/1722171?hl=en, [Online; accessed 20-September-2018].

[20] A. Medina, A. Lakhina, I. Matta, and J. Byers, "Brite: An approach to universal topology generation," in *IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, 2001, pp. 346–353.

[21] A. V. Ventrella, G. Piro, and L. A. Grieco, "Publish-subscribe in mobile information centric networks: Modeling and performance evaluation," *Computer Networks*, vol. 127, pp. 317–339, 2017.

[22] A. Sackl, P. Casas, R. Schatz, L. Janowski, and R. Irmer, "Quantifying the impact of network bandwidth fluctuations and outages on web qoe," in *IEEE International Workshop on Quality of Multimedia Experience (QoMEX)*, 2015.

[23] "Dash industry forum. dash-if position paper: Proposed qoe media metrics standardization for segmented media playback," (Date last accessed 25-October-2018). [Online]. Available: https://dashif.org/docs/ProposedMediaMetricsforSegmentedMediaDelivery-r12.pdf