Edge-assisted DASH Video Caching Employing RNIS in 5G Networks

Rasoul Behravesh

1 Background on DASH

Studies confirm that video content streaming has constituted the majority of traffic over the Internet in the last decades and still growing drastically with the novel graphical-reach video contents. The ever increasing video traffic over the Internet makes the quality of experience (QoS) assurance a challenging task for mobile network operators (MNOs) since the Internet was originally designed for best-effort, non-real-time data transmission. Regarding the dynamicity of today's network and the drastic increase in the network traffic, dynamic adaptive streaming over HTTP (DASH) emerged as an adaptive bitrate technique to improve bandwidth utilization and the user's perceived QoS [1, 2].

HTTP-based adaptive streaming (HAS)[1] utilizes HTTP as the application and TCP as the transport-layer protocol (see Fig. 1) and clients *pull* the data from a standard HTTP server, which hosts the media content. HAS techniques dynamically adapt the bitrate (video quality) concerning network varying conditions to provide a seamless streaming experience. A media file in the HAS technique is partitioned into multiple equal-size segments (also called chunks). Each video segment is available in multiple bitrates/resolution/quality. The server also generates a manifest file called media presentation description (MPD) for each video file, containing information about available representations on the server, the URL to identify the segments, and their availability time [2].

In a typical DASH system, the client first requests the MPD file containing the metadata for video, audio, subtitles, and other features from the server. After receiving the MPD file, the client continuously measures a certain set of parameters such as device buffer status, bandwidth condition, device screen size, and CPU level, etc. Based on these parameters the DASH client repeatedly requests the most suitable next segment among the available representations from the server. DASH intended to tackle several major concerns in the traditional streaming protocols [2]:

- It uses HTTP to deliver video segments, which simplifies the traversal through NATs and firewalls.
- At the server-side, it uses conventional web servers or caches available within the networks of internet service providers (ISPs) and content distribution networks (CDNs).

[1]The terms DASH and HAS are the same and can be used interchangeably

- A client requests and fetches each segment independently from others and maintains the playback session state, whereas the server is not required to maintain any state, hence, the client may download segments from different servers without impacting system scalability.
- It does not require a persistent connection between the client and the server, which improves system scalability and reduces implementation and deployment costs.

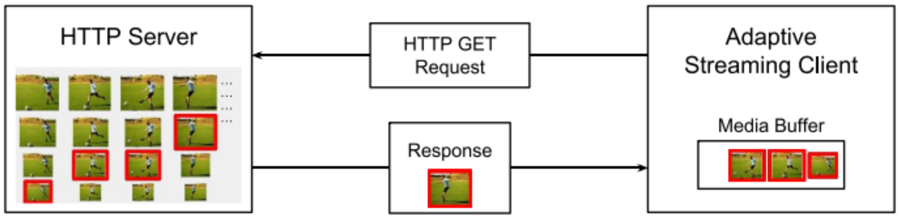

Figure 1: HTTP adaptive streaming (HAS). [2]

Today, HAS accounts for the majority of Internet video traffic. It has reached mainstream due to commercial solutions such as Microsoft's Smooth Streaming [3], Apple's HTTP Live Streaming (HLS) [4], Adobe's HTTP Dynamic Streaming (HDS) [5], Akamai's HD [6] and several open source solutions. To avoid fragmentation in the market, the Moving Picture Experts Group (MPEG) together with the 3rd Generation Partnership Project (3GPP) started working on HTTP streaming of MPEG media and HAS, respectively.

These efforts eventually resulted in the standardization of Dynamic Adaptive Streaming over HTTP (DASH) [7]. Unlike proprietary solutions, DASH provides an open specification for adaptive streaming over HTTP and leaves the implementation of the adaptation logic to third parties as shown in Fig. 2, where blue components are specified in the DASH standard, while red components are left unspecified or specified in other standards. The DASH server is essentially an HTTP server that hosts the media segments, which are typically two to ten seconds each or could be as long as hours for the entire content duration in presentation time. Each segment is encoded at multiple bitrate levels and listed in the manifest termed media presentation Description. The MPD is an XML document that provides an index for the available media segments at the server. At the client-side, DASH implements the bitrate adaptation logic, which issues timed requests and downloads segments that are described in the MPD from the server using HTTP (partial) GET messages. During the download, the DASH client estimates the available bandwidth in the network and uses information from the playback buffer to select a suitable bitrate for the next segment to be fetched. This behavior is called bitrate switching, where the client's goal is to fetch the highest-bitrate segments it can while keeping sufficient data in the playback buffer to avoid video stalls and thus achieve a good QoE trade-off.

There are various implementations of DASH players (client). For example, dash.js [8] is a JavaScript-based DASH client, which is the reference client from the DASH Industry Forum. Another JavaScript-based client is DASH-JS [9], which proposes a simple rate adaptation logic.

There are multiple adaptive bitrate (ABR) schemes including client-based adaption, server-based adaption, network-assisted adaption, and hybrid adaption.

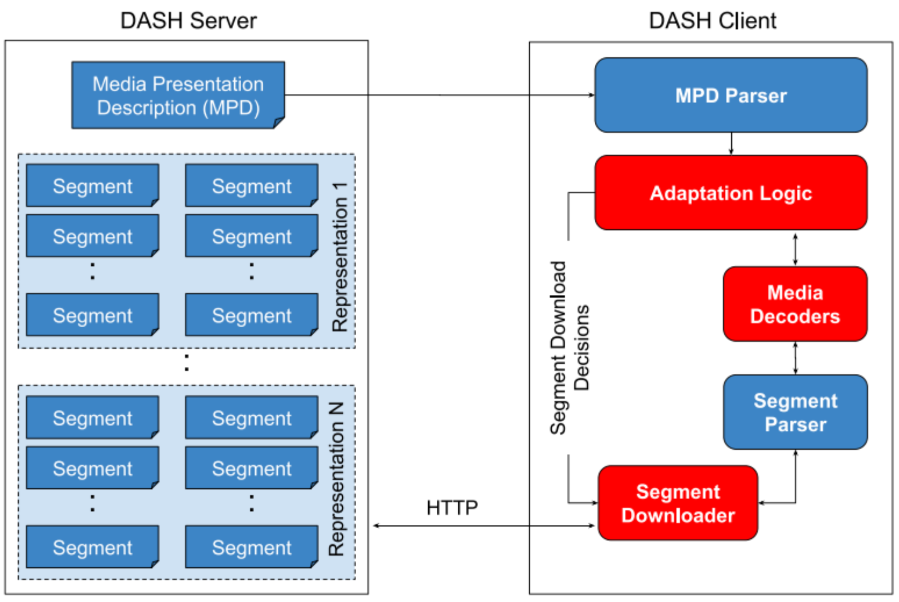

Figure 2: DASH architecture components. [2]

1.1 Client-based adaption

These schemes try to adapt to bandwidth variations by switching to an appropriate video bitrate according to one or more metrics such as the available bandwidth, playback buffer size, etc. These algorithms try to avoid streaming problems like video instability, quality oscillations and buffer starvation while improving viewer QoE.

The algorithms belonging to the client-based scheme strive to achieve:

- Minimal rebuffering events when the playback buffer depletes.
- Minimal startup delay especially in case of live video streaming.
- A high overall playback bitrate level concerning network resources.
- Minimal video quality oscillations, which occur due to frequent switching.

Some other characteristics of client-based adaptionn:

- Many adaptive streaming systems adopt a “client-pull” paradigm where the client device determines the quality level and the server accommodates the client’s requests by sending the appropriate video data.
- This requires a custom video player at the receiver (i.e., the client) to estimate the bandwidth.
- The playlist file at the server may also be requested periodically by the client so that the available video chunks can be selected based on the estimated bandwidth.

- If the bandwidth falls below a certain limit, chunks with higher quality levels (requiring higher bandwidth) will not be requested by the client.
- Depending on the number of quality levels, such a system may suffer from high startup delay and frequent quality switching, which leads to greater latency, especially with limited network bandwidth.
- In some instances, a drastic change in the video quality may occur from a severe fluctuation in the network bandwidth.
- The design of a bandwidth estimation algorithm with a proper switchover threshold is critical to the performance of client-based adaptive streaming systems but unfortunately the task is challenging.
- Some algorithms may constantly try to stream videos at the highest quality level. Others may choose to be conservative and stream at the lowest quality level when bandwidth for higher quality levels is available.

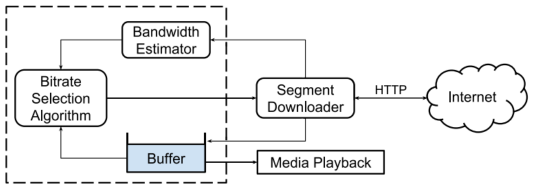

Figure 3: Client-based bitrate adaptation. [2]

1.2 Server-based adaption

Server-based schemes use a bitrate shaping method at the server side and do not require any cooperation from the client The switching between the bitrates is implicitly controlled by the bitrate shaper. The client still makes its own decisions, but the decisions are more or less determined by the shaping method on the server.

The algorithms belonging to the server-based scheme strive to achieve:

- Server to exert some control on quality switching to improve the adaptivity to network bandwidth variance and video content complexity.
- It is highly desirable for content or service providers because it allows the server traffic load to be managed, which reduces the possibility of a drastic change in the video quality that degrades the end-user quality of experience.
- In the limiting case, bandwidth estimation and quality switching are performed entirely by the server, leading to a “server-push” paradigm.
- The source-controlled system allows partial or full control by the server in minimizing quality switch activity, thereby maintaining consistent video playback.

- The server includes a rate shaper and a bandwidth estimator.
- The encoder determines the maximum number of quality levels based on a number of video encoding parameters such as the desired video resolution and bit rate.

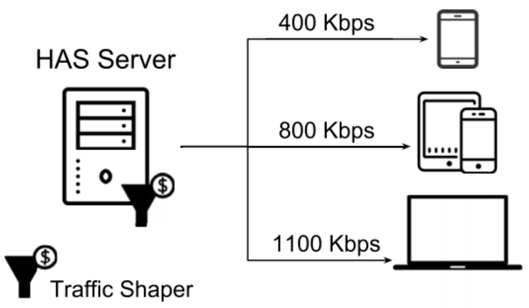

Figure 4: Server-based bitrate adaptation. [2]

1.3 Network-assisted adaption

The network-assisted approach allows the HAS clients to take in-network decisions during the bitrate adaptation process into consideration. This happens by collecting measurements about the network conditions while informing the clients on the suitable bitrates to be selected. The in-network process needs a special component (e.g., agent/proxy deployed in the network) to monitor the network status and conditions. It offers network-level information that allows the HAS clients to efficiently use network resources.

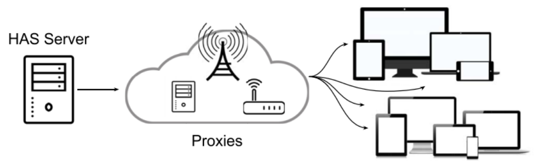

Figure 5: Network-assisted bitrate adaptation. [2]

1.4 Hybrid adaption

In hybrid bitrate adaptation, many networking entities collaborate together and collect useful information about network conditions that can help HAS clients in their bitrate selection. This type of technique consists of SDN-based and server-and-network-assisted adaptations.

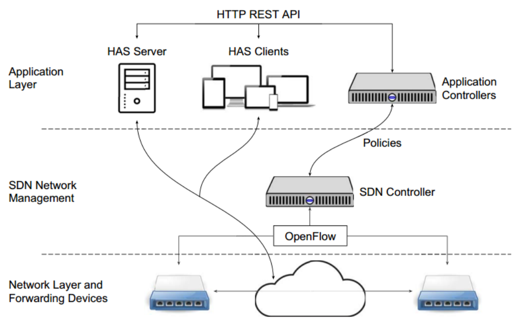

Figure 6: Hybrid bitrate adaptation. [2]

2 Cache Management and Delivery Techniques for 5G Systems [10]

While the demand for rich multimedia content has increased tremendously in recent years, the capacity of the wireless link, mobile radio network, mobile backhaul, and mobile core network cannot practically cope with the explosively growing mobile traffic due to the centralized nature of mobile network architectures.

An important portion of mobile multimedia traffic is due to duplicate downloads of a few popular contents (e.g., popular music videos) with large sizes. Therefore, researchers and engineers have been investigating effective ways to reduce the duplicate content transmissions by adopting intelligent caching strategies inside the mobile networks, and enabling mobile users to access popular content from caches of nearby MNO gateways (e.g., using selective IP traffic offload techniques). From the perspective of Internet service providers (ISPs), this also helps reduce traffic exchanged inter- and intra-ISPs, not to mention significant reduction in the response time required to fetch a content file. Thus, the impact of Internet traffic dynamics on variation of the response latency can be eliminated. Furthermore, reducing traffic load via intelligent caching of popular content would enhance the energy efficiency of 4G networks contributing to the evolution of green 5G networks effectively.

As the last-mile connectivity is constantly improving as 4G/LTE and fiber-optic broadband access is becoming popular1, the connection between the content server and the proxy is increasingly the bottleneck for large-scale video streaming applications. This phenomena that the "middle" of the network has become a limiting factor has also been observed by large-scale content distribution networks such as Akamai Technologies.

Caching in 3G mobile networks and caching in 4G LTE networks have both been proven to be able to reduce mobile traffic by one third to two thirds. Furthermore, from related studies, it has become apparent that the technical key issues fall into the following three questions:

Where to cache?

- The evolved packet core (EPC)

- The radio access network (RAN)

What to cache?

Caching aims to achieve a trade-off between the transmission bandwidth cost, which is usually expensive, especially for the inter-ISP traffic bandwidth, and the storage cost, which is becoming much cheaper. However, the scale of content acquired by content providers (CPs) is growing significantly and it is thus all but impossible to cache all content. It is hence important to decide what content to cache taking into account content popularity. It has proven that only a small amount of popular content is accessed by a large portion of mobile user requests, while a long tail of contents remains unpopular.

How to cache?

Caching policies, deciding what to cache and when to release caches, are crucial for overall caching performance. It is effectively important to estimate the gain behind a content by evaluating its current popularity, potential popularity, storage size, and locations of existing replicas over the network topology. Rather than adopting traditional caching policies, such as least recently used (LRU), least frequently used (LFU), and first-in first-out (FIFO), it is challenging to design cooperative caching policies for EPC and RAN caching to appropriately improve the cache hit ratio.

2.1 Caching within EPC

Both EPC caching and RAN caching can significantly reduce user-perceived latency as well as the transmission of redundant traffic over the network. Furthermore, caching at the edge of the network has the effect of smoothing traffic spikes and balancing the backhaul traffic over a long period of time. Current widely deployed caching functions mostly take place within the EPC, that is, at the P-GW forming the so-called mobile content delivery network (mobile CDN). However, downstream of the EPC (i.e., at the RAN), currently contents are transmitted via eNBs by GPRS Tunneling Protocol (GTP) with encapsulation, so it is technically easier to deploy content-aware caching at the EPC than at the RAN. By caching content at the mobile core net work of 3G [2] and 4G [4] systems, one to two thirds of mobile traffic can be reduced. Currently, there are generally two kinds of caching techniques:

- **Web caching**, which is object-oriented caching with content awareness,and consists of web caching based on uniform resource locator (URL) and prefix-based web caching. Caching of web objects have been widely deployed, however, the video objects are significantly larger in size than regular web objects. Hence, aggressive prefetching of videos could be very expensive, resulting in overloading the link between cache and the content server. In order to maximize the byte-hits using prefetching it is necessary to better predict the requests.
- **Redundancy elimination (RE)**, which is protocol-independent and flow-oriented or packet-oriented, and consists of chunk-level RE, TCP-RE, and packet-level RE.

2.2 Caching at the RAN

Moving application processing resources toward the network edge closer to mobile users will make it possible to simultaneously reduce network traffic and improve quality of experience. This can further optimize the most expensive part of the network operating cost of the various fiber leased lines that connect eNBs to EPC. However, most RAN caching solutions face the same implementation issues: eNBs establish "tunnels" between users and the EPC, while content files are packetized and then encapsulated by GTP tunneling, making it difficult to carry out object-oriented or contentaware caching.

Unlike centralized EPC caching, one important issue of RAN caching consists of the fact that the caching space at each eNB is practically small, and the number of users served by individual eNBs is usually small (i.e., unless in highly dense areas such as Times Square in New York City), resulting in low-to-moderate hit ratios. Therefore, intelligentcaching resource allocation strategies and cooperative caching policies among (neighboring) eNBs is mandatory for efficient RAN caching.

3 Open Issues and Challenges: Distributed Cache Resource Management and Cooperative Caching Policy

An important limitation of RAN caching stems from the fact that the caching space at individual eNBs is usually small and the user base is quite limited, which will potentially result in low hit ratios. Generally speaking, a RAN consists of thousands of eNBs interconnected via high-capacity links. A well tuned inter-eNB cooperative caching policy is therefore needed that considers the dynamics of local user demands, caching status of neighboring eNBs, and global optimization of caching resource utilization as well as the maximization of QoS of all users.

There is practically a trade-off between the diversity of content stored inside EPC and RAN and the redundancy of the replicas of popular content in eNBs.

What to consider to improve caching:

- In order to achieve efficient cooperation among nodes (devices) in the EPC and RAN, the caching status should be effectively shared among nodes (devices), most importantly incurring only a minimal signaling overhead.
- As there are always numerous new content coming, the underlying caching policy should also consider the dynamic changes of the content popularity and user demands so that adequate caching decisions are made online and in real time.
- Another important issues of RAN caching pertains to service continuity (i.e., upon a handoff operation, termination of content transmission from source eNB, and prefetching of content in the target attaching eNB). Mobility-aware caching and prefetching in the RAN are highly needed.

4 Cache Management for DASH at the Edge of 5G Networks

4.1 Motivation

The delivery of mobile content, in particular of ultra-high definition (UHD) videos with 4K resolution, has been envisaged as a prominent use case in the context of future 5G network development. It has been widely recognized that 5G is not only about increasing network capacity, but also about assuring service quality by enabling necessary network intelligence in complex and dynamic environments. With the recent development of new networking paradigms, in particular, multi-access edge computing it is envisaged that content awareness and intelligence can be embedded at the mobile network edge (e.g. at eNodeBs or eNBs in LTE networks) in order to achieve desirable user quality of experiences (QoE) against various dynamicity and uncertainty in the network ecosystem.

Regarding the dynamicity of today's network and the drastic increase in the network traffic, DASH video streaming emerged as an adaptive bitrate technique to improve bandwidth utilization and the user's perceived QoS. In conventional DASH video streaming applications, user devices (i.e., DASH clients) adaptively request different video qualities according to dynamic network conditions in order to maintain satisfactory user QoE. One way to improve the perceived QoS of DASH video streams for the users is to move the content as close as possible to the content consumers or cache them in close proximity to the users. Mobile-edge caching has two additional technological motivations compared to its CDN counterpart. First, caching videos close to eNBs and content servers relieves congestion at the network operators' backhaul. Second, caching also enables better control of the quality of experience (QoE) of streamed videos because end-to-end connections are confined within the LTE evolved packet system (EPS). Hence, they are fully under control of the mobile network operator, who can take full advantage of the existing EPS QoS mechanisms.

On challenge ahead of caching and contnet streaming at the edge stems from the fact that resources at the edge are scarce and so costly to provision especially for multimedia content that requires a considerable amount of storage resources; therefore, the intelligence existed at the network edge should be employed to make smart decisions on caching the most beneficial video segments to store at the edge.

4.2 Related Work

The study in [11] presents an integrated prefetching and caching proxy named iPac intending to maximize the byte-hit ratio for proxies through prefetching and CDNs. Authors trying to maximize the byte-hit ratio for proxies by prefetching video contents to meet the real-time constraints over the requested videos and the limited bandwidth of the network. They formulate the problem as the knapsack problem and then propose an online algorithm that can tackle the problem in real-time. This work assumes the future segment request to be in the same bitrate of the current segment; therefore, it prefetches video segments with the same bitrate for future requests. Moreover, the work does not consider any processing power existing at the edge for real-time transcoding of videos. Authors in [12], introduce a framework called adaption aware cache (AAC) to determine the video segments that are beneficial to be prefetched and stored in the cache in order o maximize the cache byte-hits. A component in the framework is responsible to measure the bandwidth of the active HTTP sessions at the cache server. The measurements acquired from the bandwidth estimator together with the information about the client-driven adaption scheme are used to predict the next segment request. The study presents a high-level view of the framework and no implementations are available. Moreover, the work s not specifically for cellular mobile networks and does not take the radio channel condition into account. The study in [13], formulate the problem of presentation selection as a MILP problem following by proposing a distributed algorithm capable of reaching a near-optimal solution in a shorter time scale. They consider the effect of multi-path on the transmission capacity in a multi-autonomous system environment. Authors in [14] suggest a MEC-enabled architecture for improved video delivery. Their proposed architecture includes an adaption algorithm at the edge, responsible for alleviating network congestion and improving user's perceived QoE. The work tries to improve the user's QoE by modifying the original MPD file using a service located at the MEC server. The work aims to change the original MPD file in a way that it matches the current wireless access network bandwidth. They assume the role of a proxy/redirection node between the end-user and the video servers. The study in [15] considers a heterogeneous network in which each client can switch between different wireless networks. A MEC application is introduced, capable of estimating the bandwidth and updating the client about the network condition. Therefore, it is considered as a proxy-based approach that helps the client to make proper decisions at the moment. Authors in [16] propose a MEC-enabled framework capable of utilizing RNIS to perform caching and updating the cache for DASH video services. They propose two kinds of popularity for the caching videos: (i) request popularity and (ii) expected popularity to improve video quality and reduce buffer time. Authors in [17] proposed a MEC-based DASH video caching in which always the highest quality representation of each segment will be stored at the edge. The work aims to employ the processing power available at the MEC nodes to transcode the video segment. The study presented in [18] proposes an adaptation aware hybrid client-cache (AAHCC) framework, a cache pre-fetching scheme that prefetch bitrate using an adaptation algorithm based on throughput measurements from the client and the forecasted throughput at the cache. They propose a hybrid client-cache driven pre-fetching scheme. Authors in [19] propose a learning-based edge with caching and prefetching (LEAP) to improve the online user QoE of adaptive video streaming. LEAP introduces caching into the edge to reduce the redundant video transmission and employs prefetching to fight against network jitters. To reduce the transmission redundancy, LEAP caches the most beneficial segments from the server according to the QoE-weighted popularity. The main challenge lies in the online decision of caching and prefetching. Moreover, they do not consider the possibility of transcoding. Authors in [19] present a context-aware approach that can use content context (e.g. segment popularity) and network context (e.g. RAN DL throughput) to make decisions on caching / replacement of video segment with aim of boosting the user's perceived QoE. The work not only takes the video segment popularity into account but the video representation as well.

4.3 Problem Definition

As mentioned earlier, DASH enables clients to switch between different video bit-rates in case of variations in network conditions. Allowing users to request their desired video bit-rate, helps to avoid stream interruptions, thus improving video QoE. However, when the clients share the same bottleneck link, as is the case for mobile networks, the DASH protocol shows some instabilities,

Table 1: Caching techniques for DASH video streaming

DASH Video Caching Parameters								
Ref.	BW Estimation	Client Collaboration	Server Collaboration	TCP/IP or ICN	RNIS	UE Mobility	Edge transcoding	Obj
[11]	Y	N	Y	TCP/IP	N	N	N	Max. byte-hit
[12]	Y	N	Y	TCP/IP	N	N	N	Max. byte-hit
[13]	Y	N	Y	General	N	N	N	Max. QoE
[14]	Y	Y	Y	TCP/IP	Y	N	N	Max. QoE
[15]	Y	N	Y	TCP/IP	Y	N	N	Max. MOS (QoE)
[16]	N	N	Y	TCP/IP	Y	N	N	Max. QoE
[17]	N	N	N	TCP/IP	N	N	Y	Max. QoE
[18]	Y	Y	Y	TCP/IP	N	N	N	Max. Byte-hit
[19]	Y	N	N	TCP/IP	N	N	N	Max. QoE
[20]	Y	Y	Y	ICN	Y	N	N	Max. QoE

unfairness, and bandwidth under-utilization. There is, therefore, a need to be aware of the end-user device context and network conditions to optimize video delivery.

DASH is particularly interesting in wireless and mobile access networks, which present unexpected and frequent such variations. Moreover, mobile users in these networks share a common radio access link and, thus, a common bottleneck in case of congestion, which may cause user experience to degrade. In this context, MEC gives new opportunities to improve DASH performance, by moving IT and cloud computing capabilities down to the edge of the mobile network.

The traditional scheme of caching is to store and forward videos where the video segments are cached in the requested quality and the same video may be cached on different MECs causing data replication. Since different users may request different bit-rates of the same video segment, cached video segments from one session are unusable to serve other users watching the same video. Therefore, a simple caching mechanism such as store and forward is not efficient for caching DASH videos. Moreover, regarding the fact that the coverage of a base station is much smaller than an EPC or server and the connectivity between base stations and the user is with high uncertainty, caching policy design at BSs is more challenging.

4.4 Approach

To overcome the limitations in the seamless streaming of DASH videos, we propose a smart caching strategy that utilizes the wireless channel condition information existed at the edge to make decisions on caching DASH video segments. Mainly two possible solutions were discussed in the literature for the caching problem of DASH videos: (i) store all segments of a video at the edge servers, (ii) store the highest representation of a video segment and transcode it when necessary. Each of the above-mentioned solutions has its disadvantages. The main issue with the first solution is the high storage requirement for storing all the videos at the network edge, which is difficult to find at the edge servers. Besides, the second solution also demands a high processing capacity which is very costly and scarce at the edge.

Regarding the constrained processing and storage resources at the edge, there is a high demand for having techniques that can intelligently decide how to use the resources to have the best resource utilization and the highest user satisfaction. Our first step towards solving the problem is to use a Machin Learning prediction method to predict the highest demand video segments by the users given the wireless channel condition. Therefore, we will have a method capable of predicting future video segment requests and based on that then devise an ILP method that can find the optimal location of videos. If the requested video bit-rate could be not found at the local edge servers, the system checks if the video with higher quality is available in the local server. If the same video with higher bit-rate could be found at the local edge server, then the video needs to be transcoded and then can be transmitted to the user. Otherwise, the requested version of the video should be fetched from the central MEC server, which contains all the versions of all the videos. Moreover, to tackle the scalability of the ILP model, we aim to propose a heuristic algorithm that can reach a near-optimal solution in a much smaller time scale.

Our contributions are listed below:

- We employ a machine learning method that can predict the next video segment that might be asked by the users given features such as traffic load on the base station, number of users, cell throughput, and user throughput.
- We propose an ILP method capable of finding the best solution for video segment placement and taking into account the possibility of transcoding videos at the edge using the processing capacity available at the edge nodes.
- Regarding the fact that a small number of users are attached to each base station and shifting the video segments to the edge result in a very low cache-hit, we consider the possibility of using video segments at the neighbor edge server and also tackle the problem of video duplication at the edge.
- Our method is a proactive caching method, meaning a video segment will be prefetched before being requested by the user. By this approach, we are trying to tackle the jitter problem and avoid buffer depleting of the users for the video segments requested for the first time.
- We consider users as mobile devices that can move while watching a video that is not considered in the literature.

Timeline.

Developing the idea: 15 March to 25 March 2020

Modeling and formulating of the problem: 25 March 2020 - 15 April 2020

Simulations and result validation: 15 April 2020 - 30 May 2020

Writing: 30 May 2020 - 15 Jun 2020

Conferences.

IEEE CNSM : Submission (Jun) - November

ACM CoNEXT : Submission (July) - September

IEEE NOMS: Submission (September) - April

IEEE WCNC: Submission (November) - April

IEEE CSCN: Submission (July) - October

References

[1] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia, "A survey on quality of experience of http adaptive streaming," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 469–492, 2014.

[2] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, "A survey on bitrate adaptation schemes for streaming media over http," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 562–585, 2018.

[3] "Microsoft Smooth Streaming, Microsoft, Redmond, WA, USA, 2015," Accessed on 09.02.2020. [Online]. Available: https://azure.microsoft.com/en-us/services/media-services/

[4] "Apple HTTP Live Streaming, Apple, Cupertino, CA, USA, 2015," Accessed on 09.02.2020. [Online]. Available: https://developer.apple.com/streaming/

[5] "Adobe HTTP Dynamic Streaming, Adobe, San Jose, CA, USA, 2015," Accessed on 09.02.2020. [Online]. Available: https://www.adobe.com/products/hds-dynamic-streaming.html

[6] "Akamai. (2015). Akamai HD," Accessed on 09.02.2020. [Online]. Available: https://www.akamai.com/us/en/resources/live-videostreaming.jsp

[7] "C. Timmerer. (2012). HTTP Streaming of MPEG Media." Accessed on 09.02.2020. [Online]. Available: https://multimediacommunication.blogspot.co.at/2010/05/httpstreaming-of-mpeg-media.html

[8] "Dash Industry Forum. (2017). DASH-264 JavaScript Reference Client," Accessed on 09.02.2020. [Online]. Available: https://github.com/Dash-Industry-Forum/dash.js/wiki

[9] B. Rainer, S. Lederer, C. Müller, and C. Timmerer, "A seamless web integration of adaptive http streaming," in *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*. IEEE, 2012, pp. 1519–1523.

[10] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. Leung, "Cache in the air: Exploiting content caching and delivery techniques for 5g systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 131–139, 2014.

[11] K. Liang, J. Hao, R. Zimmermann, and D. K. Yau, "Integrated prefetching and caching for adaptive video streaming over http: an online approach," in *Proceedings of the 6th ACM Multimedia Systems Conference*, 2015, pp. 142–152.

[12] P. Juluri and D. Medhi, "Cache'n dash: Efficient caching for dash," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4, pp. 599–600, 2015.

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

[14] Y. Li, P. A. Frangoudis, Y. Hadjadj-Aoul, and P. Bertin, "A mobile edge computing-based architecture for improved adaptive http video delivery," in *2016 IEEE Conference on Standards for Communications and Networking (CSCN)*. IEEE, 2016, pp. 1–6.

[15] ——, "A mobile edge computing-assisted video delivery architecture for wireless heterogeneous networks," in *2017 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2017, pp. 534–539.

[16] Y. Tan, C. Han, M. Luo, X. Zhou, and X. Zhang, "Radio network-aware edge caching for video delivery in mec-enabled cellular networks," in *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*. IEEE, 2018, pp. 179–184.

[17] S. Kumar, D. S. Vineeth *et al.*, "Edge assisted dash video caching mechanism for multi-access edge computing," in *2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. IEEE, 2018, pp. 1–6.

[18] S. K. Mehr, P. Juluri, M. Maddumala, and D. Medhi, "An adaptation aware hybrid client-cache approach for video delivery with dynamic adaptive streaming over http," in *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2018, pp. 1–5.

[19] W. Shi, Q. Li, C. Wang, G. Shen, W. Li, Y. Wu, and Y. Jiang, "Leap: learning-based smart edge with caching and prefetching for adaptive video streaming," in *Proceedings of the International Symposium on Quality of Service*, 2019, pp. 1–10.

[20] C. Ge, N. Wang, S. Skillman, G. Foster, and Y. Cao, "Qoe-driven dash video caching and adaptation at 5g mobile edge," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*, 2016, pp. 237–242.