Application of Device-to-Device Communication in Video Streaming for 5G Wireless Networks

Ala'a Al-Habashna and Gabriel A. Wainer

Department of Systems and Computer Engineering. Carleton University. 1125 Colonel By Dr. Ottawa, ON. K12 5B6. Canada.

Abstract We present one of the key-enabling technologies in 5G wireless networks, i.e., Device-to-Device (D2D) communication. Furthermore, we discus some of our work in utilizing D2D communication in one of the most bandwidth-demanding applications in 5G networks, i.e., video streaming. D2D communication, introduced by the LTE-Advanced (LTE-A) standard, allows direct communication between devices in cellular networks. We review some of the work in the literature on D2D communication and its applications in video streaming. We also discus an architecture that provides Dynamic Adaptive Streaming over HTTP (DASH) -based Peer-to-Peer (P2P) video streaming in cellular networks. The architecture employs Base-Station (BS) -assisted D2D video transmission in cellular networks for direct exchange of video contents among users. We evaluate the performance of the proposed architecture in terms of many video streaming Quality-of-Experience (QoE) metrics.

1 Introduction

With the advancement of smart devices and the evolution of cellular networks, video streaming service has become immensely popular. Video content is the main contributor to data traffic over cellular networks nowadays. According to [1], video traffic accounted for 60% of total mobile data traffic in 2016. Furthermore, over three-fourths (78%) of the world's mobile data traffic is expected to be video by 2021. Another reason for this increasing popularity of video streaming is the emergence of new platforms for video streaming such as YouTube and Netflix. As such, providing high Quality of Experience (QoE) video streaming services has become a main concern for cellular networks operators.

Due to the limited capacity of cellular networks, it is difficult to provide the users with the data rates needed to achieve high QoE video streaming, especially in the case of high user density. The continuous increase in resource-demanding video

applications is outpacing the improvements on the cellular network capacity. As per [2], video traffic is the main reason for congestion in mobile networks. The facts above made it necessary to develop new approaches that help serving the increasing video traffic over cellular networks and improve the QoE of video streaming.

In [3–5], we proposed two algorithms for BS-controlled progressive caching of video contents and Device-to-Device (D2D) video transmission in cellular networks. D2D communication, introduced by the LTE-Advanced (LTE-A) standard, allows direct communication between devices in cellular networks [6]. The algorithms are called Cached and Segmented Video Download (CSVD) and DIStributed Cached and Segmented video download (DISCS). The algorithms are employed by the BS to control progressive video content caching in selected User Equipment (UEs) in the cell, referred to as Storage Members (SMs). Furthermore, the algorithms are employed by the BS to control D2D communication between UEs in the cell for Peer-to-Peer (P2P) video content distribution to requesting UEs.

Here, we present an architecture to improve the QoE of video streaming in cellular networks. The proposed architecture employs the aforementioned cached and segmented video download algorithms. The architecture also employs Dynamic Adaptive Streaming over HTTP (DASH). DASH is an adaptive video streaming technique which allows changing the video bit rate during video streaming to adapt to the available throughput [7]. The proposed architecture is called DASH-based BS-Assisted P2P/D2D video STreaming in cellular networks (DABAST).

In [3–5], we investigated the performance of CSVD and DISCS in terms of the aggregate and average data rates. Results have shown that DISCS significantly improves the data rates over CSVD. Here, present some performance evaluation results for DABAST with CSVD (DABAST-CSVD) in various scenarios. Results have shown that DABAST-CSVD achieves significant gains and improves all the measured QoE metrics. We also show performance evaluation results for DABAST with DISCS (DABAST-DISCS) in terms of video streaming QoE metrics. We compare the results of both DABAST-CSVD and DABAST-DISCS to see if the improvement in the data rates achieved by DISCS would translate into significant improvements in terms of video streaming QoE. We provide analysis of the results and present the findings on DABAST with both algorithms.

We use the Discrete EVent System Specification (DEVS) formalism [8] to build a model for DABAST-CSVD and DABAST-DISCS. DEVS provides a formal framework for modeling generic dynamic systems. It has formal specifications for defining the structure and behavior of a discrete event model. We implement our DEVS model with the CD++ toolkit [8] and use this implementation for performance evaluation.

The rest of this chapter is organized as follows, in Section 2 we review the background and related work. In Section 3, we present CSVD, DISCS, and DABAST. We discus modeling of DABAST with DEVS in Section 4 and present our DEVS model. We present the simulation scenarios and results in Section 5. Finally, we conclude the chapter in Section 6.

2 Background

D2D communication is one of the main technologies in the Fifth Generation (5G) cellular networks [9] due to the improvements it provides. With D2D communication, two UEs within proximity of each other can exchange data over direct links, without the need to relay the traffic over the BS. This can improve the data rate between the two UEs due to transmission over one-hop and shorter distance. Moreover, the capacity of the cellular network can be increased by coordination of multiple short distance transmissions to achieve spatial frequency reuse. D2D communication can also extend the coverage area of the cell and improve the received signal for users at the cell edge. As such, much work has been conducted to develop applications for D2D communications in cellular networks and improve its performance [10–12]. D2D communication allows collaboration of users in cellular networks to share contents they have. However, approaches are needed to motivate participation of users in D2D communication. Incentivizing users to participate in D2D communication is a topic that has received much interest in the last couple of years. The interested reader in this area is referred to [13, 14].

Video streaming was considered ever since the early stages of the Internet, and nowadays, it is the most popular application on the web. According to [15], during the peak hours, YouTube traffic only accounts for 27% of the mobile downlink (DL) video traffic in North America. With video streaming, a user can start playing the video before the entire video file is downloaded. Most videos on the web nowadays are accessed via streaming. Contents such as movies, video news clips, and YouTube videos are watched by millions of people every day.

HTTP video streaming is the most popular form of video streaming nowadays, and it has been adopted by major video streaming solutions such as YouTube, Netflix, and Hulu. This is due to the convenience of using HTTP [16], which eliminates the need to install and use a dedicated streaming application and helps to get the streaming traffic past firewalls. HTTP video streaming works by breaking the overall video stream into a sequence of small HTTP-based file downloads, referred to

as video segments. Users progressively download these small segments, while the video is being played. Playout usually starts after receiving a certain "sufficient" number of video segments. The received segments are buffered in a video/application buffer. The application that plays the video is usually referred to as the client. The streaming client requests video segments from the server. Since each segment has a fixed duration, the size of the segment depends on its duration and the video bit rate. The client receives the pieces from the video buffer. The duration of video content available for playout is called the playout buffer length, measured in seconds of video. Furthermore, every second, one second of video is removed from the buffer and played to the user.

Bad network conditions (insufficient bandwidth, delay, etc.) may cause the playout buffer to get empty, as the video bit rate is higher than the video streaming rate, which causes video playout interruptions. These interruptions are referred to as video stalling or rebufferings. When stalling occurs, playout stops until sufficient data is buffered again.

Although HTTP video streaming provided a convenient way for video streaming over the Internet, it was still challenging to stream video to wireless and mobile devices due to the high bandwidth variability of the wireless links. DASH provided a promising technique to improve video streaming over networks with varying bandwidth [17], as it allows changing the quality of video streaming to adapt to network conditions.

DASH provides two features that helped improving video streaming. First, it breaks down the video into small, easy to download segments (for example 5-second chunks). Second, each segment is encoded at multiple bit rates, providing multiple quality levels for each segment, which allows adaptive streaming. Clients will choose between various bit rates to adapt to the network conditions. As such, DASH helps improving the bandwidth utilization and reducing the interruptions of the video playback, which results in a higher streaming quality. Due to these advantages of DASH over classical HTTP video streaming, DASH has been employed by big video streaming platforms, such as YouTube and Netflix, and it is being adopted by an increasing number of video applications.

There are various adaptation strategies that can be used to determine how the client selects the streaming quality to adapt to the varying network conditions. These strategies usually try to balance between two factors. They try to maximize the video quality by selecting the highest video rate the network can support, and at the same time minimize rebufferings. We refer to the component in the client that runs the adaptation strategy as the DASH controller.

With the increasing demand for video applications, providing high quality video service as perceived by the end user has become an important concern for cellular network operators. As such, quality measure has shifted from Quality of Service (QoS) to QoE. The ITU defines QoE as the overall acceptability of the service as perceived by the end user. Video streaming QoE is especially important because users pay their operator, and they expect to get video service with good QoE in return. If the user is not satisfied, they may look for other options and switch to another provider. As such, video streaming QoE must be considered in network design and management in order to maintain user satisfaction. There are many factors that are used to measure video streaming QoE, here we present the most important ones [18].

- Video stalling (rebuffering): the stopping of video playback as the playout buffer gets empty. Increasing video stalling decreases the QoE. Many studies [18] have shown that video stalling has the biggest impact on QoE, and thus, should be avoided as much as possible.
- Video continuity index: a measure of the extent by which rebuffering pauses are avoided [19]. The continuity index is measured as follows,

$$\eta_c = 1 - \frac{\Delta T_{rb}}{\Delta T}, \tag{1}$$

  where ΔT_{rb} is the total time the client remains paused due to rebuffering events and ΔT is the duration of the experiment (playing time and rebuffering time).
- Initial (startup) delay: the delay from the request to stream the video until the playback starts. Initial delay is always present as certain number of video segments should be received before decoding and playback starts.
- Video bit rate: it is a measurement of the amount of data in one second of the video. Video bit rate is determined by many quality factors of the video such as video frame rate, resolution, and quantization parameters. As the video bit rate increases, the video quality increases, which increases the QoE.

Direct communication between nodes in wireless networks has been studied in the context of wireless ad hoc networks [20]. However, it has not been considered in cellular networks until the emergence of D2D communications [10–12, 21]. The introduction of D2D communications in the LTE-A standard has opened the door for direct P2P communications between UEs in cellular networks [6]. With D2D

communication, two UEs that are within proximity of each other communicate directly without going through the BS or the core network.

D2D communication provides a promising solution to increase the capacity in cellular networks by allowing frequency reuse, and by increasing the data rates due to potentially improved transmission over shorter distance and fewer hops. As such, it has been investigated in the last few years [10–12, 21].

There has been some work on P2P video streaming in cellular networks. A protocol for P2P video streaming on mobile phones, called RapidStream, was proposed in [22]. It is similar to many of the P2P streaming protocols on wired networks that involve the dissemination of buffer maps and video chunks between peers. While such protocols work well in wired networks, they involve too much signaling and transmission (dissemination of buffer maps) to be appropriate for UEs that has limited power, processing, and transmission resources (especially on a large scale). In [23], multi-source video streaming was proposed where mobile users can connect through Wi-Fi direct to other users to get some of the video content. Such system requires the device to perform device discovery to find neighbors, and service discovery to find services offered by neighboring devices. These requirements along with the signaling needed to exchange content consume significant amount of resources.

In [24], a system, called MicroCast, was designed and evaluated using a testbed. MicroCast is used by a group of smart phone users who trust each other, are interested in watching the same video at the same time, and who are within proximity of each other. Users employ their cellular connection to download segments of the video and use their Wi-Fi connections to share among each other the downloaded content, to improve the streaming experience. While this could result in some improvement for a group of users, the scope of the system is limited. Furthermore, users usually do not use their cellular connection for downloading video segments when Wi-Fi is available.

In [25] the authors proposed a D2D communication system where multiple helpers collaborate to send a video segment to the requesting UE. The video, which is assumed to be in scalable video coding standard, is encoded by applying multiple description coding by each helper, and each helper sends a different description to the requesting UE. The authors analytically studied the problem of optimizing the number of transmitted descriptions to the requesting UE to maximize the video quality and efficiently consume the helpers' energy. However, the work only considers the energy consumed by the helpers to send the segments without considering the processing power and energy needed to encode the video segments. Encoding the video segments is a big favor to ask for, considering the limited energy and processing power of UEs.

None of the research studies above on P2P video streaming in cellular networks considers how the video segments are actually cached. When evaluating the performance, they consider that requested segments are already available in helper UEs. With DABAST, we provide a framework that takes care of video content caching and distribution [26]. Moreover, the employed CSVD and DISCS algorithms provide approaches for inter-cluster as well as inter-cell interference mitigation. Furthermore, the work above considers small-scale networks, i.e., up to 10 UEs including the helpers. We show that using clustering and BS assistance, the potential of collaborative D2D communication between UEs is significant.

Here, we present DABAST [27, 28] and discus its operation and implementation. We also discuss the performance evaluation of DABAST-CSVD under various scenarios, to provide quantitative evaluation of the impact of many parameters on the performance of DABAST. Furthermore, we compare the results of both DABAST-CSVD and DABAST-DISCS to see if the improvement in the data rates achieved by DISCS would translate into significant improvements in terms of video streaming QoE.

We used the DEVS formalism [29] to build a model for the DABAST architecture, and used that model to test and evaluate the performance of DABAST using various simulations. DEVS provides a formal framework for modeling generic dynamic systems. It has formal specifications for defining the structure and behavior of a discrete event model. A DEVS model is composed of structural (Coupled) and behavioral (Atomic) components, in which the coupled component maintains the hierarchical structure of the system, while each atomic component represents a behavior of a part of the system. The CD++ toolkit [8] was used to implement our model of DABAST. CD++ is an open-source simulation software written in C++ that implements the DEVS abstract simulation technique. The simulation engine tool of CD++ is built as a class hierarchy. C++ is used to develop the atomic components of the model. These components can be incorporated into the class hierarchy. Passive classes can be also used to model components of the system. Coupled models can be created using a language built in the simulation engine.

3 DABAST: The architecture

In this section, we discuss DABAST and its operation. Thereafter, we present the employed cached and segmented video download algorithms.

3.1 DABAST

By employing the cached and segmented video download algorithms [28], DABAST relaxes the bottleneck of the Radio Access Network (RAN), and hence, improves video streaming QoE for end users.

Fig. 1 illustrates the main structure for DABAST. At the bottom, we have the LTE-A network that provides the infrastructure for communication between the BS and UEs over cellular links, and the communication between UEs over D2D links where the UEs exchange data directly without going through the BS.

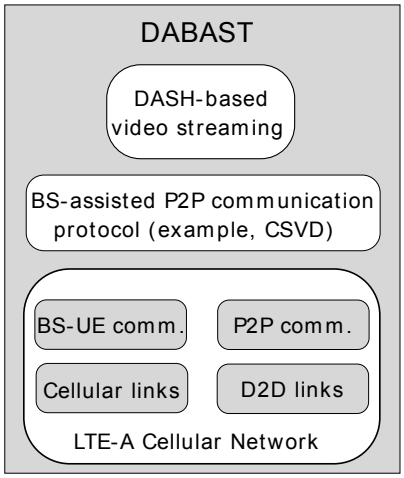

Figure 1 – DABAST: the architecture.

A BS assisted P2P/D2D communication protocol (e.g., CSVD) is implemented on top of that which uses both the cellular and D2D communication. DASH-based video streaming takes place on top of these layers, as the transmission of video segments is implemented as per the communication protocol at the layer below.

Figure 2 depicts the implementation of DABAST in cellular networks. A CSVD server/proxy is used in the RAN at the BS. This provides the processing and networking capabilities needed to implement CSVD or DISCS. The CSVD server can also be used to provide caching capabilities to store popular files at the BS. Streaming clients at the UEs send their requests asking for video segments. These requests are processed by the CSVD server. Based on the employed algorithm (CSVD or DISCS), the server decides whether to send the segment from the distributed cache over the D2D channel or get it from the content server and send it over the cellular

channel. If the video segment is to be delivered from the distributed cache, an assistance request will be sent to an SM. Otherwise, the request will be forwarded to the DASH server. Under high traffic load, the BS sends a video segment from the distributed cache (when found) even if the segment found in the distributed cache does not match the video bit rate requested by the UE. This is to maximize the exploitation of the distributed cache and D2D channel.

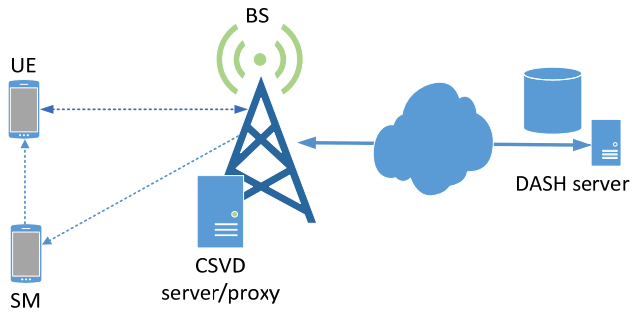

Figure 2 – Illustration of the implementation of DABAST.

Another feature of DABAST is that it can operate in proactive mode. In this mode, DABAST can send up to a certain number of video segments to the user when found in the distributed cache before the segments are requested. This reduces the signaling and latency between the BS and the UE and speeds up transmission of video segments. However, If the video segment is not available in the distributed cache, the request will be awaited.

A video bit rate adaptation strategy is used by DASH to determine how the client selects the streaming quality to adapt to the varying network conditions. Many video bit rate adaptation strategies are proposed for DASH [30, 31]. These strategies usually try to balance between two factors. They try to maximize the video quality by selecting the highest video rate the network can support, and at the same time minimize rebufferings. We refer to the component in the client that runs the adaptation strategy as the DASH controller. We adopt the buffer-based approach in [31]. This is because in our architecture, the UE could receive a video segment from the BS or from any SM in the cluster. As such, it would be difficult to estimate the throughput at which the next segment will be received. The adaptation algorithm used is a piecewise function, $f(B)$, that uses the length of the playout buffer, B, to determine the video bit rate [31].

3.2 The CSVD and DISCS algorithms

CSVD and DISCS relax the RAN bottleneck by providing BS-assisted progressive video content caching and P2P video segment distribution in cellular networks. When we use the term P2P here, we refer to direct transmission of video segments between UEs in the cell over D2D links.

With both algorithms, the cell is divided into clusters. To do that, the coverage area of the cell is divided into non-overlapping subareas by the BS. Each one of these subareas will be a cluster. The BS assigns UEs to clusters based on their locations, and it selects the UEs in the central area of each cluster as SMs of that cluster. SMs are UEs that are used as helpers in the cluster. To prevent inter-cluster as well as inter-cell interference, only the UEs in the central area of each cluster are selected as SMs.

After clustering, when a UE requests a video file from the BS, the BS processes the request and responds as follows:

- Send With Assistance (SWA): if the video (or parts of it) is available in any of the SMs of the cluster, the BS will ask the SMs to send the video segments to the requesting UE over D2D links.
- Send To an SM (STSM): if the requested video is not available in the distributed cache of the cluster, and the requesting UE is an SM, the BS will send the video to that UE over a cellular link, and it will ask the UE to cache it. This case allows the SMs to cache video files. These files will be available for UEs in the cluster when requested later.
- Distribute to SMs (DTSMs): this case is only used in DISCS. In this case, if a requested video is popular and it is not available in the distributed cache of the cluster, the BS will distribute the segments of the video among the SMs. The BS asks the SMs to cache the pieces (as the video is popular) and forward the received pieces to the requesting UE.
- Send To a UE (STUE): otherwise, the BS will send the video directly to the requesting UE over a cellular link.

The difference between CSVD and DISCS is the DTSMs case. The DTSMs case is only used in DISCS. The other 3 cases are used by both CSVD and DISCS. For further detail on the operation of CSVD and DISCS, the reader is referred to [5].

4 Modeling DABAST with DEVS

As discussed earlier, we used DEVS to build a model of DABAST in an LTE-A network. Fig. 3 shows the coupled DEVS model of the top-level architecture. As can be seen, we have defined a Cell coupled model that contains many UE coupled models, a BS coupled model, and a Transmission Medium coupled model. The Cell coupled model also contains Cell Manager and Log Manager atomic models.

A UE coupled model contains four atomic models: UE Queue, UE Controller, Streaming Client, and DASH controller. Messages received are buffered at the UE Queue. The UE Controller is where the UE part of the CSVD algorithm is implemented.

The DASH-based streaming client is implemented in the Streaming Client and DASH controller atomic models. The streaming client manages the video buffer. It adds video segments received to the video buffer and removes video segments that were played from the buffer. As the video buffer usually has a certain length that could be shorter than the video length, it is implemented as a sliding window. Video segments that were already played will be removed from the video buffer and the buffer slides to cover the next segments in the stream. The DASH controller implements the adaptation algorithm. It monitors the video playout buffer and updates the video bit rate accordingly. When the video bit rate is to be updated, a request is sent to the BS with the new video bit rate.

The BS coupled model includes four atomic models: BS Queue, BS Controller, Scheduler, and Transmitter. Received messages are buffered at the BS Queue. The BS Controller is where the BS part of the CSVD algorithm is implemented. The Scheduler schedules the messages to be transmitted in the next Transmission Time Interval (TTI), which is 1 ms. Every TTI, the BS Controller also asks the Transmitter to send messages that were scheduled for transmission during this TTI.

The Medium model simulates the transmission medium and the Cell Manager atomic model initializes and sets the parameters of the cellular DLs and uplinks (ULs) between the BS and the UEs, as well as the D2D links between the UEs. For further details on the communication models used for simulation of the LTE-A cellular links and D2D links, the reader is referred to [5]. In addition to the atomic models above, many other passive classes where developed to model other components of the system such as classes to model the cellular and D2D links, download sessions the BS has with UEs, etc.

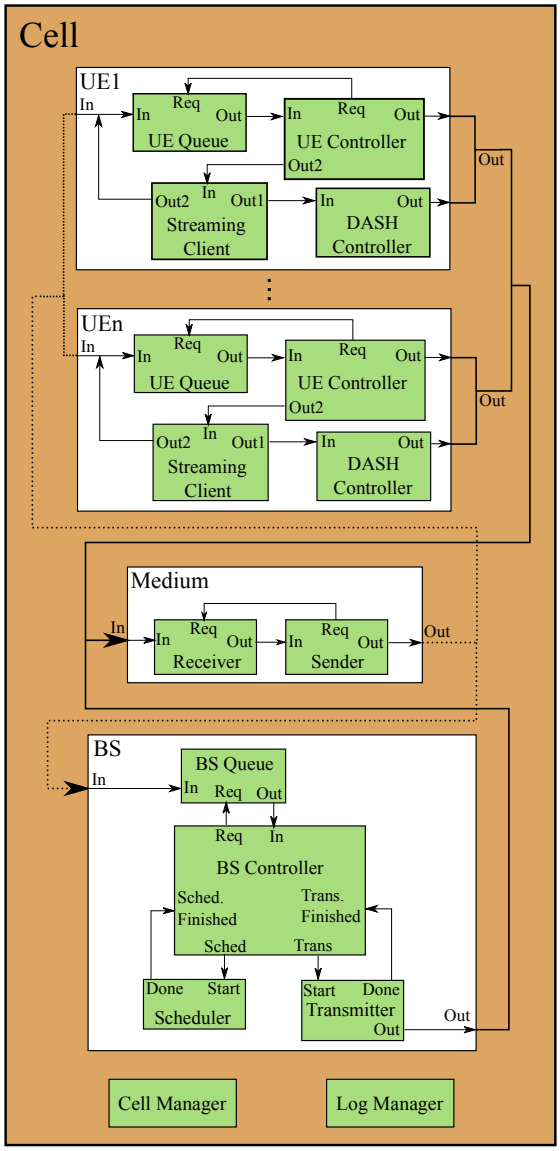

Figure 3 – Coupled DEVS model of DABAST.

5 Simulation scenarios and results

We executed simulations to evaluate the performance of DABAST in terms of the QoE metrics that were discussed in section 2. We ran various simulations to study the impact of many parameters on DABAST such as files' popularity, the number of UEs in the cell, etc. In this Section, we consider DABAST with CSVD (DABAST-CSVD). In the next section, we consider both DABAST-CSVD and DABAST-DISCS and compare their performance.

Table 2 – Simulation setup.

Parameter	Value
Cellular Channel BW (MHz)	10
Cell Range (m)	500
Number of clusters	9
BS antenna gain (dB)	12
BS transmission power (dBm)	43
UE antenna gain (dB)	0
UE transmission power (dBm)	21
Noise spectral density (dBm)	-174
Antenna height (m)	15
Transmission model	UTRA-FDD
DL Carrier frequency (MHz)	900
Number of requests by a UE	2
Area configuration	Urban
D2D Channel BW (MHz)	60
D2D Carrier frequency (GHz)	24
D2D transmitter TX Power (dBm)	23
D2D Large-scale fading std deviation (dB)	4.3
D2D Receiver noise figure (dB)	9
D2D TX/RX Height from Ground (m)	1.5
Segment length (second)	10
Number of buffered segments to start playout	4
Video bit rate levels (kbps)	384, 768, 2000, 4000
Videos length (second)	441

The simulation setup is shown in Table 1. The simulations consider a single LTE-A cell. The urban macro propagation model [32] was used for cellular links with a DL operating carrier frequency of 900 MHz, and a transmission bandwidth of 10 MHz. The D2D channel model at 24 GHz is used for D2D transmission [33].

In each iteration of the simulation, the UEs are uniformly distributed throughout the cell. Clustering takes place in the beginning in case of DABAST where the cell is divided into 9 clusters. The UEs then start requesting video streams. During each iteration of the simulation, each UE will request two video streams. A UE requests a video stream, and after finishing the playout, it will request a second video. The arrival of requests is generated according a Poisson arrival process. The popularity of videos is generated according to a Zipf distribution to simulate the variable popularity of the videos, as it has been established this is a good model for this purpose [34]. Using this distribution, some videos are requested more often than others. The length of the videos is 441 seconds, which is the mean length of a YouTube video [35]. Four video bit rate levels where used as shown in Table 1. These are adapted from the H.264/AVC video coding standard [36].

Regarding the DASH controller, the buffer-based approach in [31] was employed. This is because in our architecture, the UE could receive a video segment from the BS or from any SM in the cluster. As such, it would be difficult to estimate the bit rate at which the next segment will be received. The adaptation algorithm used is a piecewise function, $f(L)$, that uses the length of the playout buffer, L, to determine the video bit rate.

Table 2 – Playout buffer length to video rate mapping.

Playout buffer length (s)	Video bit rate (kbps)
$0 \leq L \leq 90$	384
$90 < L \leq 150$	768
$150 < L \leq 200$	2000
$200 < L$	4000

We assume that the video buffer is long enough to accommodate all received segments. The playout buffer to video rate mapping is shown in Table 2. We measured the number of rebufferings, video continuity index, initial delay, and video bit rate levels of the received video segments. Table 3 shows the mean values for these measurements. The results in Table 3 are for 500 UEs in the cell, Zipf exponent of 1.5, and 500 videos. The average value for each simulation run was calculated. The

values below show the mean of all the average values from 50 simulation runs. In addition to the mean, we show the Margin of Error (MoE) for 95% confidence interval.

Table 3 shows that DABAST achieves improvements over conventional DASH in terms of all the measured metrics above. Regarding the average number of rebufferings, DABAST achieved 50% decrease in the average number of rebufferings, which is a significant improvement. The continuity index is also improved with DABAST due to decreasing the average number of rebufferings as well as the rebuffering time. It is worth mentioning that the average initial delay for conventional DASH is high because in this scenario, there are 500 UEs in the cell requesting video streams and sharing fixed cellular frequency resources (10 MHz). This is also because video playout starts after receiving 4 video segments.

Table 3 – Simulation results.

	Conventional DASH		**DABAST**	
	Mean	MoE	Mean	MoE
Rebufferings	3.4448	0.0179	1.7272	0.0164
Cont. index	0.7447	0.0009	0.8699	0.0001
Initial delay(sec)	56.881	0.2200	28.654	0.4821
Video bit rate (kbps)	397.27	0.3267	430.16	1.3694

Table 3 shows that DABAST also achieves a 50% decrease in the average initial delay, which is also a significant improvement. In addition to the improvements above, DABAST also achieved an improvement in terms of the average video bit rate. Due to the increase in the transmission rates achieved by DABAST, video segments are delivered to UEs much faster than in the case of conventional DASH. This is because in the case of DABAST, the CSVD algorithm is employed, where video segments are sent to many UEs from both the BS (over cellular links) and SMs (over D2D links) as opposed to only from the BS. This reduces the transmission delay of the first 4 segments needed to start playout, and consequently, reduces initial delay. This also reduces the possibility of video buffer stalling, and hence, reduces the number of rebufferings. There is only a small improvement achieved by DABAST in terms of average video bit rate. This is due to two reasons. First, with both conventional DASH and DABAST, the DASH controller resorts to choosing a lower video bit rate level to increase the video playout buffer length and reduce

the number of rebufferings. This means that video bit rate is the last metric that is improved when transmission rate improves. As such, the improvement in terms of the number of rebufferings is usually achieved on the expense of video bit rate. Second, as mentioned in the previous section, under high traffic load, the BS will send a video segment from the distributed cache (when found) even if the segment found in the distributed cache does not match the video bit rate requested by the UE. This is to increase the utilization of the available D2D channel and to speed up the transmission of video segments, as sending from the distributed cache is faster. This means that although DABAST might increase the transmission rate and play-out buffer length for some clients (which increases the requested video rate), such users might still receive segments with low video bit rate (from the distributed cache).

Figure 4 shows the histogram of the number of rebufferings for both conventional DASH and DABAST. The figure shows that over 96% of the streaming requests have 3 or 4 rebufferings in the case of conventional DASH. With DABAST, on the other hand, half of the streaming requests have 0 rebufferings, and slightly less than half of the streams have 3 or 4 rebufferings. After videos accumulate in the clusters' caches, many video segments will be delivered from the distributed cache in the cell. These segments will be transmitted faster to the requesting UEs. Moreover, as many of the segments will be sent over D2D links, there will be more cellular resources available for segments that are transmitted over cellular resources, which also reduces the transmission delay for such segments and reduces the possibility of playout interruption, decreasing the number of rebufferings by 50%.

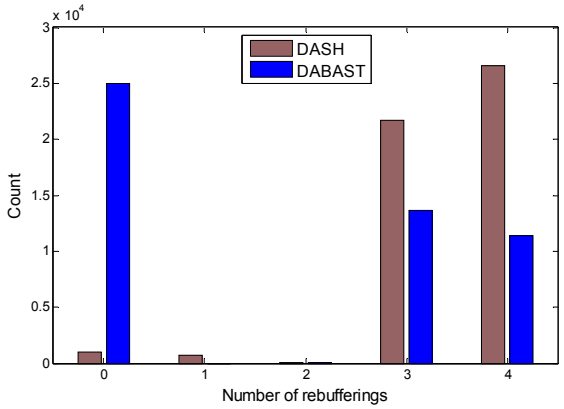

Figure 4 – Histogram of the number of rebufferings for conventional DASH and DABAST.

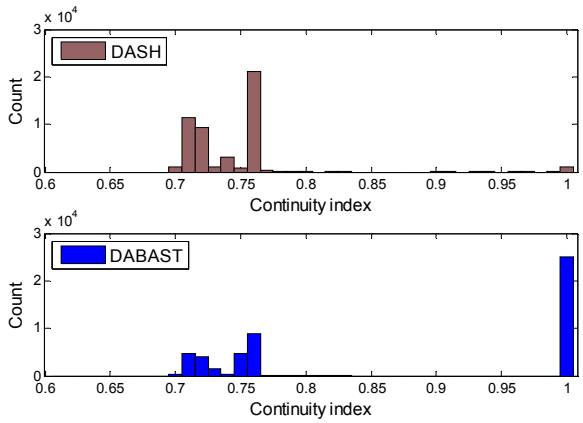

Figure 5 – Histogram of the continuity-index for conventional DASH and DABAST.

Figure 5 shows the histogram of the continuity index for both conventional DASH and DABAST. The continuity index results match these in Figure 4 for the number of rebufferings. Half of the requests with DABAST have a continuity index of 1, which corresponds to zero rebufferings. Less than half of the video streams have a continuity index less than 0.76 with DABAST (corresponds to 3 and 4 re-bufferings). However, in the case of conventional DASH, over 96% of the video streams have a continuity index less than 0.76.

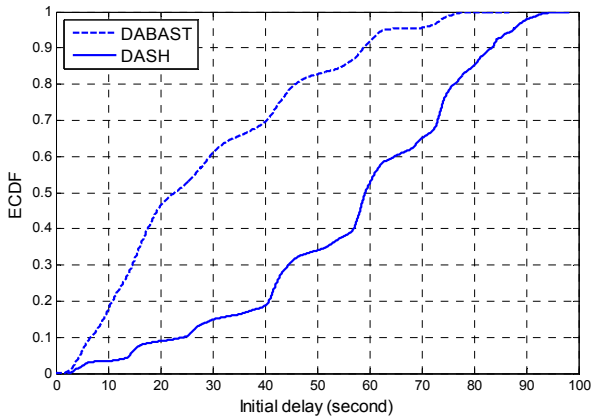

Figure 6 – ECDF of the initial delay for conventional DASH and DABAST.

Figure 6 shows the ECDF of the initial delay for both conventional DASH and DABAST. We can see that the ECDF of DABAST is always higher than that of conventional DASH. For example, the probability of having a stream with initial delay of 20 seconds or less is 0.46 with DABAST, and only 0.09 with conventional DASH. Figure 6 also shows that 50% of the streams have initial delay of 22.76 seconds or less with DABAST while 50% of the streams have 59.21 seconds or less with conventional DASH. As previously mentioned, with conventional DASH, all the UEs in the cell share the fixed frequency resources (10 MHz cellular channel), while with DABAST, the D2D channel is exploited for P2P communication in addition to cellular resources. The transmission of video segments from the distributed cache in the cell speeds up the delivery of video segments and significantly reduces initial delay.

Table 4 shows the count for the received video segments with each video bit rate level, for both conventional DASH and DABAST. The results show that with DABAST, fewer video segments with 384 kbps were received and more video segments with higher levels (768, 2000, and 4000 kbps) were received. This explains why the average video bit rate (Table 3) for DABAST is higher than that for conventional DASH. With DABAST, video segments are delivered faster to the requesting UEs as explained above. As such, clients will have more video segments in the playout buffer, i.e., higher playout buffer length. Consequently, the requested video bit rate will be higher in the case of DABAST.

The results presented in this section show that DABAST provides improvements over conventional DASH in terms of all the measured metrics, which significantly improves the QoE of video streaming in cellular networks. In the following, we investigate the impact of different parameters on the performance of DABAST.

Table 4 – Count of the received segments with each video bit rate.

	Count	
Video bit rate (kbps)	Conventional DASH	DABAST
384	2207508	2077111
768	31494	149365
2000	10998	19273
4000	0	4251

Figure 7 shows the average number of rebufferings for both conventional DASH and DABAST versus the number of UEs in the cell. Figure 7 shows that at 300 UEs, the average number of rebufferings is 0 for both conventional DASH and DABAST. This means that the cellular resources are enough with conventional DASH to avoid rebufferings for all the video streams. As the number of UEs increases, the available cellular resources will be shared by more UEs, which reduces the average data rate and increases the transmission delay of video segments. This increases the possibility of video buffer depletion and increases the average number of rebufferings.

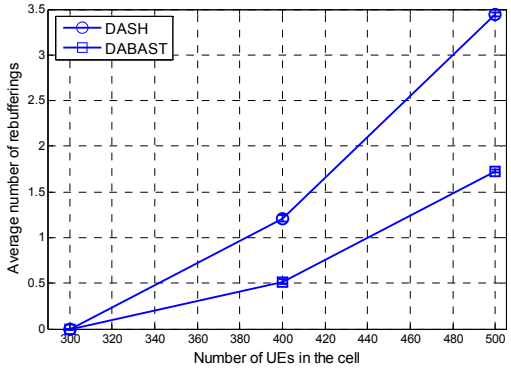

Figure 7 – Average number of rebufferings versus number of UEs in the cell. Zipf exponent = 1.5 and 500 videos.

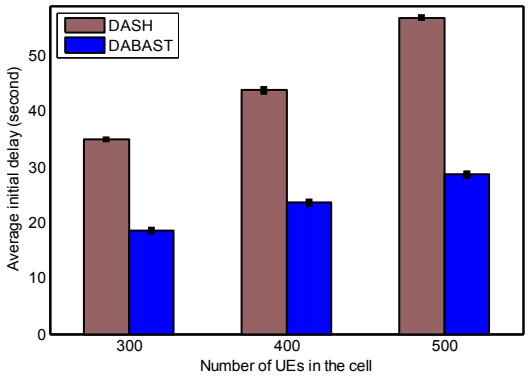

Figure 8 – Average initial delay versus number of UEs in the cell. Zipf exponent = 1.5 and 500 videos

Even with DABAST, the average number of rebufferings increases with increasing the number of UEs. This is because we study the case of progressive caching, where in the beginning there are no videos cached, and videos are cached as requested. Figure 7 shows that the improvement achieved by DABAST increases by increasing the number of UEs in the cell. Increasing the number of UEs increases the number of requests and also increases the number of SMs in each cluster. This increases the number of cached videos in a cluster and the percentage of requests that would be satisfied from the cluster's cache, increasing the improvement achieved by DABAST over conventional DASH.

Figure 8 shows the average initial delay for both conventional DASH and DABAST versus the number of UEs in the cell. Figure 8 shows that although the average number of rebufferings is zero at 300 UEs for both approaches (as in Figure 7) DABAST still achieves improvement in terms of the average initial delay at 300 UEs. Furthermore, it can be noticed that the gain achieved by DABAST over conventional DASH increases with the number of UEs. This is for the same reason explained above for the average number of rebufferings.

As mentioned above, the Zipf distribution was used to model the popularity of videos. The Zipf distribution has one parameter, namely the Zipf exponent. This exponent controls the relative popularity of the videos. When the value of the Zipf exponent increases, the relative popularity of some videos in the list increases. This increases the possibility of requesting these videos.

Figure 9 shows the average number of rebufferings versus the Zipf exponent. As can be seen, the average number of rebufferings decreases as the Zipf exponent increases. As the popularity of some videos increases, higher percentage of the requests will be found in the distributed cache and delivered over the D2D channel (rather than the cellular channel). This speeds up the transmission of many segments and decreases the possibility of playout buffer depletion, which decreases the average number of rebufferings.

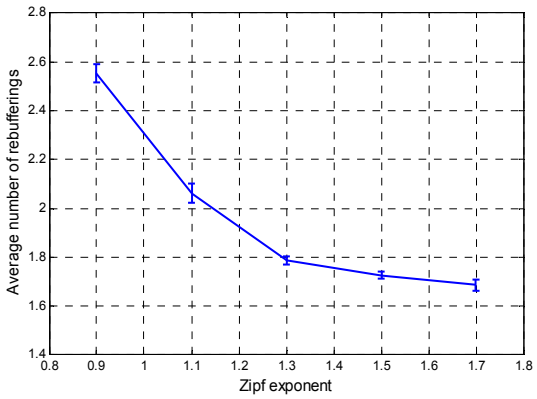

Figure 9 – Average number of rebufferings versus the Zipf exponent. 500 UEs, 500 videos.

Figure 9 shows that the improvement achieved with increasing the Zipf exponent eventually slows down. This is because in our scenario, each UE requests only two videos. Increasing the number of requests made by each UE further increases the exploitation of cached contents and increases the improvement achieved by DABAST. This is because cached videos will be further used by the later requests.

Figure 10 shows the average number of rebufferings for DABAST versus the number of videos available to request from. As can be seen, there is no considerable effect for the number of videos on the average number of rebufferings. As per the Zipf distribution, having more videos will not cause a noticeable impact on the probability of requesting the popular files. This means that for a certain Zipf exponent, although the number of videos increases, cached contents will still be exploited as long as there are popular videos.

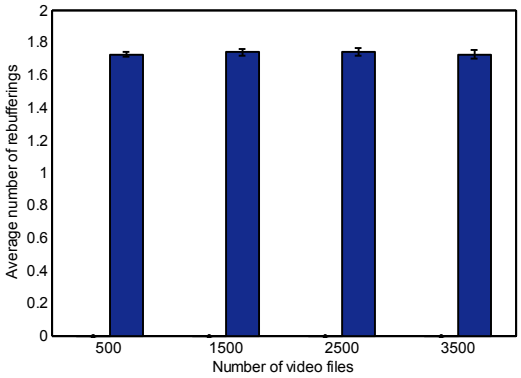

Figure 10 – Average number of rebufferings versus the number of videos 500 UEs and Zipf exponent = 1.5.

It is worth mentioning that DABAST should be even more beneficial when considering background traffic. To get a feel of the improvement achieved by DABAST in the case background traffic is present, we ran simulations with 300 UEs in the cell and with the same setup in Table 1. Unlike the previous simulations above, we consider that background traffic is present in the cell, and that the BS dedicates 5 MHz of the channel for background traffic and 5 MHz for video streaming traffic. Let us recall that in the case background traffic is absent and the whole channel is available for video streaming traffic (300 UEs), users experienced 0 rebuffering with both DASH and DABAST. On the other hand, in the case background traffic is present, the results have shown that with conventional DASH, the average number of rebufferings is 4.47, while with DABAST, the average number of rebufferings is 2.35. This shows that DABAST in this case has achieved 47.4% reduction in the average number of rebufferings. This shows how DABAST is even more beneficial and can achieve further gains in the case background traffic is present.

In [5], we have studied the performance of CSVD and DISCS in terms of data rates. Results have shown that DISCS achieves significant improvement over CSVD in terms of the average data rate. Furthermore, in the previous section, we have studied the performance improvement achieved by DABAST that employs CSVD (DABAST-CSVD) in terms of video streaming QoE. Results have also shown that significant improvements can be achieved using DABAST-CSVD in terms of all the measured QoE metrics. In this section, we study how much improvement DABAST can further achieve if DISCS is employed instead of CSVD, i.e., we want to study if the significant improvement achieved by DISCS over CSVD in terms of data rate translates into significant improvement in terms of QoE.

Remember that with DISCS, in addition to the 3 cases considered in CSVD, that are listed in 3.2, an additional case is employed (DTSMs). In this case, if a requested video is popular (requested *n* times) and it is not available in the distributed cache of the cluster, the BS will distribute the pieces among the SMs in the cluster. The BS asks the SMs to cache the received pieces (as the file is popular) and forward them to the requesting UE. This case helps speeding up accumulation of popular video files in the distributed cache of the cluster. It also allows for more parallelism and load balancing among SMs when sending video files from the distributed cache of the cluster. This should increase the utilization of the D2D channel and speeds up the transmission, and consequently increase the average data rate.

We executed simulations to evaluate the performance of DABAST that employs DISCS (DABAST-DISCS) in terms of the same QoE metrics. The simulation setup in the previous section was also used here. Table 5 shows results for both DABAST-CSVD and DABAST-DISCS. The results in Table 5 are for 500 UEs in the cell, Zipf exponent of 1.5, and 500 videos. The average value for each simulation run was calculated. The values below show the mean of all the average values from 50 simulation runs. In addition to the mean, we show the MoE for 95% confidence interval.

Before discussing the results in Table 5 for the measured QoE metrics, it is worth mentioning that the average data rates achieved with DABAST-CSVD and DABAST-DISCS are 3.89 and 8.32 Mbps, respectively. DISCS significantly improves the achieved average data rate because it speeds up video caching which increases the percentage of requests that are satisfied from the distributed cache which speeds up the transmission. This is also because in the case of DISCS, many files will be sent in parallel from multiple SMs (as opposed to one SM), thanks to the DTSMs case which distributes segments of a cached video file among multiple SMs. This causes further parallelism in sending video files and better load balancing among SMs, which speeds up the transmission of video files and increases the average data rate. From Table 5, we can see that as expected, DABAST-DISCS provides improvement over DABAST-CSVD in terms of the measured QoE metrics. Table 5 shows that DABAST-DISCS reduced the average number of rebufferings from 1.73 to 1.63, which increased the continuity index. With DABAST-DISCS, the initial delay is also reduced from 28.7 seconds to 21.2 seconds, which is a significant improvement. Furthermore, the average video bit rate increased by 18 kbps with DABAST-DISCS.

Table 5 – Simulation results.

	DABAST-CSVD		DABAST-DISCS	
	Mean	MoE	Mean	MoE
Rebufferings	1.7272	0.0164	1.6294	0.0169
Cont. index	0.8699	0.0001	0.8763	0.0011
Initial delay(sec)	28.654	0.4821	21.192	0.4163
Video bit rate (kbps)	430.16	1.3694	448.57	0.9607

Although the results above show that DABAST-DISCS improves the QoE metrics, one can see that the only significant improvement achieved by DABAST-DISCS is in terms of the initial delay. Only slight improvement is achieved in terms of the average number of rebuffering and continuity index. One would expect higher gains by DABAST-DISCS over DABAST-CSVD given the that DISCS achieves more than double the average data rate obtained with CSVD. In the following, we present further analysis of the above results to understand this behavior.

Figure 11 depicts the relative frequency histogram for the number of rebufferings of DABAST-CSVD and DABAST-DISCS. From the figure, one can see that the main difference is that with DABAST-DISCS, higher number of video streams have 3 rebufferings and fewer number of video streams have 4 rebufferings. This explains why the average number of rebufferings with DABAST-DISCS is less than that with DABAST-CSVD. The figure shows that with both DABAST-CSVD and DABAST-DISCS, 50% of the video streams have 0 rebufferings. With DABAST-CSVD, 27.3% of the video streams have 3 rebufferings, and 22.7% of the streams have 4 rebufferings. With DABAST-DISCS, 36.0% of the video streams have 3 rebufferings, and 13.3% of the streams have 4 rebufferings. This improvement in the number of rebufferings is expected, as DABAST speeds up video caching, and improves the rate at which video segments are delivered to requesting UEs. Figure 2 also shows that with DABAST-DISCS, a small percentage of the video streams (0.1%) have 5 rebufferings. This increase in the number of rebufferings experienced by a slight percentage of the streams is a result of the DTSMs case, where video segments are sent to the requesting UEs in two steps, i.e., the segment is sent to the SM first, and then sent to the UE by the SM. Despite of this increase to a small percentage of the UEs, DABAST-DISCS still achieves lower initial delay and number of rebufferings, on average. This means that the DTSMs case is beneficial to the cell, as expected.

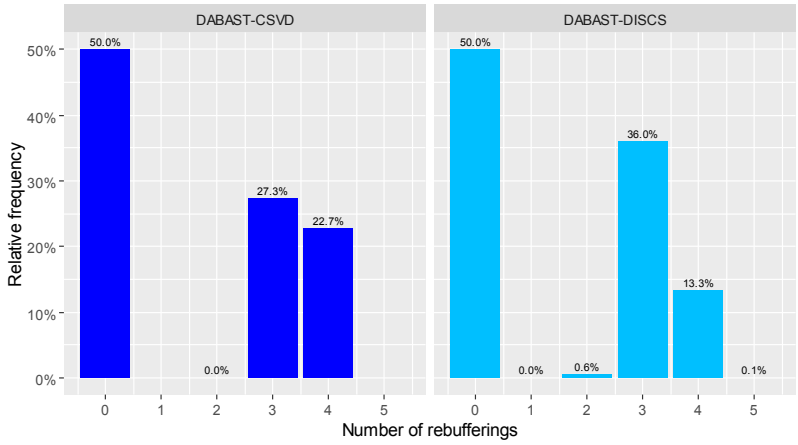

Figure 11 – Relative frequency histogram of the number of rebufferings for DABAST-CSVD and DABAST-DISCS.

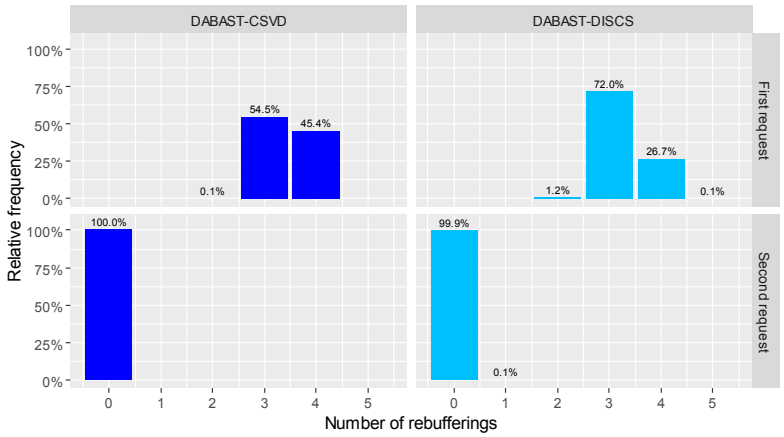

Figure12 – Relative frequency histogram of the number of rebufferings in each request for DABAST-CSVD and DABAST-DISCS.

Although the results show that DABAST-DISCS decreases the number of rebufferings, one can argue that DABAST-DISCS is expected to achieve higher gains in terms of the number of rebufferings, as it significantly improves the average data rate. To further investigate the results and explain this behavior, we separate the results for each request. As previously mentioned, each UE in the simulations makes

2 video streaming requests. After playout of the first video, a UE would stay idle for a random period of time, and then generates another request for a video stream.

Figure 12 shows the relative frequency histogram for the number of rebufferings of each request for both DABAST-CSVD and DABAST-DISCS. As with the previous figure, the histograms on the right side are for DABAST-CSVD, and the ones on the left are for DABAST-DISCS. In this figure, however, the histograms on the top are for the first requests, while the ones at the bottom are for the second requests. Figure 12 shows that for DABAST-CSVD, all the rebufferings take place during the first set of video streams. All the second video streams have 0 rebufferings. This is because by the time most of the video streams start, there are no video segments cached in the distributed caches. Hence, most of the video segments will be delivered over the cellular channel. As such, the limited cellular channel will be shared by the large number of users, which means the average data rate at which these segments are delivered is low and explains the high number of rebufferings. The figure shows that with DABAST-CSVD, all the second set of streams have 0 rebufferings. By the time the second set of streams starts, there will be many video segments cached in the clusters. Hence, many of the segments will be delivered over D2D links, which eliminates rebufferings for those streams. This also relaxes the bottleneck of the RAN because, at this time, only a portion of the video segments will be sent over the cellular channel. Because the cellular channel is now shared by a much lower number of UEs, the data rate will increase, and rebufferings will be avoided for these video streams as well.

With DABAST-DISCS, almost all rebufferings occur in the first set of video streams, as 99.9% of the second set of video streams have 0 rebufferings, and only 0.1% of the second set of video streams have 1 rebufferings. This is for the same reason all the rebufferings with DABAST-CSVD occur during the first set of video streams. Initially, there are no video segments available in the distributed cache, and hence, all video streams will experience multiple rebufferings. By the time the second set of streams starts, there will be many video segments cached in the clusters. Hence, many of the segments will be delivered over D2D links, which eliminates rebufferings for these streams, and relaxes the RAN bottleneck for segments delivered over cellular resources. The one difference here is that there is a small portion of the second set of video streams that still get its segments with the DTSMs case, which causes 1 rebuffering to 0.1% for the second set of video streams.

The results in Figure 12 also explain why DABAST-DISCS does not achieve significant improvement in terms of the average number of rebufferings over DABAST-CSVD. As discussed above, DISCS achieves significant improvement in terms of the average data rate over CSVD. This is because in the case of DISCS, many files will be sent in parallel from multiple SMs (as opposed to one SM). This

causes further parallelism in sending video segments and better load balancing between SMs, which speeds up the transmission of video files and increases the average data rate. However, we have seen from Figure 11 that video streams with segments delivered over D2D links already have 0 rebufferings, even in the case of DABAST-CSVD. As such, the increase in the average data rate achieved by DABAST-DISCS will not translate into reduction in the average number of rebufferings, as all rebufferings take place in the first set of streams when video segments are not cached. This means that the improvement in the average number of rebufferings gained by DABAST-DISCS over DABAST-CSVD is due to the fact that DISCS speeds up video caching and achieves better hit ratio, which increases the percentage of requests that are satisfied from the cluster's cache and speeds up the relaxation of the RAN bottleneck.

Figure 13 shows the histogram of the continuity-index for both DABAST-CSVD and DABAST-DISCS. As expected, 50% of the streams have a continuity index of 1. With DABAST-DISCS there is more concentration of the values around 0.76 and less concentration of values between 0.71 and 0.75. This agrees with the results for the number of rebufferings. However, with DABAST-DISCS, there are few continuity index values less than 0.7. These values correspond to streams with 5 rebufferings.

Regarding the average video bit rate, we can also see that DABAST-DISCS did not achieve a significant improvement over DABAST-CSVD (only 4.3% improvement). This can be explained as follows. As most of the cached video segments are downloaded during high traffic load. These segments are usually downloaded with low video bit rate. By the time the second set of video streams starts, there will be many segments available in the clusters' caches. However, most of these segments have low video bit rate. As DABAST sends available segments from the clusters' caches over D2D links (in the case an SM is available), most of the segments transmitted over the D2D channel will be sent from the distributed cache with low video bit rate. Although these segments are transmitted with higher data rates in the case of DABAST-DISCS, this does not increase the video bit rate for these segments. DABAST operates in this fashion to save the valuable cellular resources and exploit them for sending video segments that are not available in the clusters' caches to avoid rebufferings as much as possible.

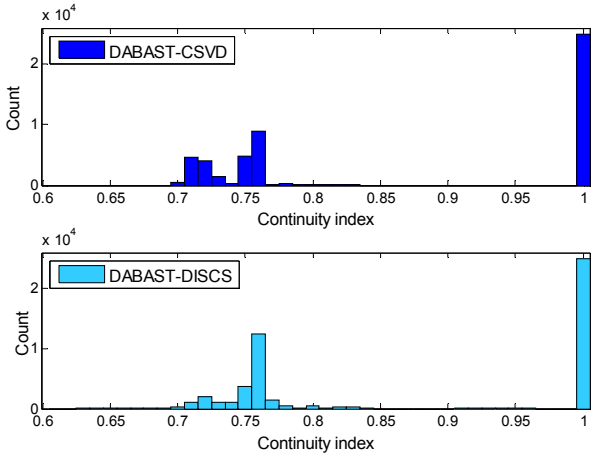

Figure 13 – Histogram of the continuity-index (DABAST-CSVD and DABAST-DISCS).

A cached video segment is sent over the cellular channel only when there are no SMs available to send the segment. These video segments that are sent over cellular resources (despite being cached) will usually be sent with high video bit rate as such segments are usually requested with high video bit rate. This is because streams with cached video segments usually have long playout buffer length, as most of their segments are sent over the D2D channel with higher data rates. Furthermore, such video streams have even longer playout buffer length in the case of DABAST-DISCS, as video segments are sent with higher data rate than these with DABAST-CSVD. Because of that, video segments that are sent over cellular resources (in spite of being cached) will be sent with higher video bit rate in the case of DABAST-DISCS when compared to DABAST-CSVD. This explains the small improvement achieved by DABAST-DISCS over DABAST-CSVD in terms of the average video bit rate.

This behavior, explained above, can be seen in Table 6, which shows the number of video segments received with each video bit rate, for DABAST-CSVD and DABAST-DISCS. Table 6 shows that in the case of DABAST-DISCS, fewer segments are received with video bit rate of 768 kbps, and more segments (about the double) are received with 2Mbps and 4Mbps video bit rates, when compared to DABAST-CSVD. While this is beneficial for the streams that receive video segments with high video bit rates, it increases the RAN bottleneck and decreases the average data rate for other UEs receiving video segments exclusively over cellular resources.

Table 6 – Count of the received segments with each video bit rate.

	Count	
Video bit rate (kbps)	DABAST-CSVD	DABAST-DISCS
384	2077111	2089649
768	149365	105874
2000	19273	46179
4000	4251	8298

6 Conclusion

In this chapter, we give a brief introduction on Device-to-Device (D2D) communication; a technology that allows direct communication between devices in cellular networks. Moreover, we review the work in literature on utilizing D2D communication in one of the most bandwidth-demanding applications in 5G networks, i.e., video streaming. We also discus an architecture we proposed that provides Dynamic Adaptive Streaming over HTTP (DASH) -based Peer-to-Peer (P2P) video streaming in cellular networks. The architecture employs Base-Station (BS) assisted D2D video transmission in cellular networks for direct exchange of video contents among users. We discuss in detail some performance evaluation results for the proposed architecture, which show that the proposed architecture can achieve significant performance gains in terms of video streaming Quality of Experience (QoE).

Acknowledgments

The authors would like to thank Professor Stenio Fernandes from the Federal University of Pernambuco (UFPE), Brazil, for his valuable input during this work.

References

1. Cisco (2017) Cisco visual networking index: Global mobile data traffic forecast update. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html. Accessed 28 Feb 2018.

2. Sandvine (2016) 2016 Global internet phenomena: Latin america and north america. Technical report.

3. Al-Habashna A, Wainer G, Boudreau G, Casselman R (2016) Cached and segmented video download for wireless video transmission. In: *Proceedings of the 49th Annual Simulation Symposium*. Pasadena, USA, pp 18–25.

4. Al-Habashna A, Wainer G, Boudreau G, Casselman R (2016) Distributed cached and segmented video download for video transmission in cellular networks. In: *2016 International Symposium on Performance Evaluation of Computer and Telecommunication Systems*. IEEE, Montreal, Canada, pp 473–480.

5. Al-Habashna A, Wainer G (2017) Improving video transmission in cellular networks with cached and segmented video download algorithms. *Mob Networks Appl* 23:543–559. https://doi.org/10.1007/s11036-017-0906-x.

6. Parkvall S, Astely D (2009) The evolution of LTE towards IMT-Advanced. *J Commun* 4:146–154.

7. IOS (2012) Information technology—Dynamic Adaptive Streaming over HTTP (DASH)-Part 1: Media presentation description and segment formats. ISO/IEC 23009-1:2012.

8. Wainer GA (2009) *Discrete-event modeling and simulation : A practitioner's approach*. CRC Press, Boca Raton.

9. Agiwal M, Roy A, Saxena N (2016) Next generation 5G wireless networks: A comprehensive survey. *IEEE Commun Surv Tutorials* 18:1617–1655. https://doi.org/10.1109/COMST.2016.2532458.

10. Asadi A, Wang Q, Mancuso V (2014) A survey on device-to-device communication in cellular networks. *IEEE Commun Surv Tutorials* 16:1801–1819. https://doi.org/10.1109/COMST.2014.2319555.

11. Kaufman B, Aazhang B (2008) Cellular networks with an overlaid device to device network. In: *the 42nd Asilomar Conference on Signals, Systems and Computers*. IEEE, Pacific Grove, USA, pp 1537–1541.

12. Doppler K, Rinne MP, Janis P, et al (2009) Device-to-Device communications; Functional prospects for LTE-advanced networks. In: *IEEE International Conference on Communications Workshops*. IEEE, Dresden, Germany, pp 1–6.

13. L. Duan, Gao L, Huang J (2014) Cooperative spectrum sharing: A contract-based approach. *IEEE Trans Mob Comput* 13:174–187. https://doi.org/10.1109/TMC.2012.231.

14. Zhang Y, Song L, Saad W, et al (2015) Contract-based incentive mechanisms for device-to-device communications in cellular networks. *IEEE J Sel Areas Commun* 33:2144–2155. https://doi.org/10.1109/JSAC.2015.2435356.

15. 3GPP (2016) Policy and charging control signaling flows and quality of service (QoS) parameter mapping. Technical report TS 29.213.

16. Li B, Wang Z, Liu J, Zhu W (2013) Two decades of internet video streaming. *ACM Trans Multimed Comput Commun Appl* 9:1–20. https://doi.org/10.1145/2505805.

17. Stockhammer T (2011) Dynamic adaptive streaming over HTTP: Standards and design principles. In: *Proceedings of the 2nd Annual ACM Conference on Multimedia Systems*. ACM Press, New York, USA, pp 133–144.

18. Seufert M, Egger S, Slanina M, et al (2015) A survey on quality of experience of HTTP adaptive streaming. *IEEE Commun Surv Tutorials* 17:469–492. https://doi.org/10.1109/COMST.2014.2360940.

19. De Cicco L, Mascolo S (2014) An adaptive video streaming control system: Modeling, validation, and performance evaluation. *IEEE/ACM Trans Netw* 22:526–539. https://doi.org/10.1109/TNET.2013.2253797.

20. Zhao J, Zhang P, Cao G, Das CR (2010) Cooperative caching in wireless P2P networks: Design, implementation, and evaluation. *IEEE Trans Parallel Distrib Syst* 21:229–241. https://doi.org/10.1109/TPDS.2009.50.

21. Doppler K, Rinne M, Wijting C, et al (2009) Device-to-Device communication as an underlay to LTE-advanced networks. *IEEE Commun Mag* 47:42–49. https://doi.org/10.1109/MCOM.2009.5350367.

22. Eittenberger PM, Herbst M, Krieger UR (2012) RapidStream: P2P streaming on android. In: *2012 19th International Packet Video Workshop*. IEEE, Munich, Germany, pp 125–130.

23. Siris VA, Dimopoulos D (2015) Multi-source mobile video streaming with proactive caching and D2D communication. In: *IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks*. IEEE, Boston, USA, pp 1–6.

24. Keller L, Le A, Cici B, et al (2012) MicroCast: Cooperative video streaming on smartphones. In: *the 10th International Conference on Mobile Systems, Applications, and Services*. New York, USA, pp 57–70.

25. Duong TQ, Vo N-S, Nguyen T-H, et al (2015) Energy-aware rate and description allocation optimized video streaming for mobile D2D communications. In: *2015 IEEE International Conference on Communications*. IEEE, London, UK, pp 6791–6796.

26. Al-Habashna A, Wainer G (2020) QoE awareness in progressive caching and DASH-based D2D video streaming in cellular networks. *Wirel Networks* 26:2051–2073. https://doi.org/10.1007/s11276-019-02055-x.

27. Al-Habashna A, Fernandes S, Wainer G (2016) DASH-based peer-to-peer video streaming in cellular networks. In: *2016 International Symposium on Performance Evaluation of Computer and Telecommunication Systems*. IEEE, Montreal, Canada, pp 481–488.

28. Al-Habashna A, Wainer G, Fernandes S (2017) Improving video streaming over cellular networks with DASH-based device-to-device streaming. In: *2017 International Symposium on Performance Evaluation of Computer and Telecommunication Systems*. IEEE, Seattle, USA, pp 468–475.

29. Zeigler BP, Praehofer H, Kim TG (2000) *Theory of modeling and simulation: Integrating discrete event and continuous complex dynamic systems*. Academic Press, San Diego.

30. De Cicco L, Caldaralo V, Palmisano V, Mascolo S (2013) ELASTIC: A client-side controller for Dynamic Adaptive Streaming over HTTP (DASH). In: *20th International Packet Video Workshop*. IEEE, San Jose, USA, pp 1–8.

31. Huang T-Y, Johari R, McKeown N, et al (2014) A buffer-based approach to rate adaptation: Evidence from a large video streaming service. In: *Proceedings of the 2014 ACM SIGCOMM*. ACM, New York, USA, pp 187–198.

32. 3GPP (2015) Evolved universal terrestrial radio access; RF system scenarios. Technical report TR36.942.

33. Al-Hourani A, Chandrasekharan S, Kandeepan S (2014) Path loss study for millimeter wave device-to-device communications in urban environment. In: *2014 IEEE International Conference on Communications Workshops*. IEEE, Sydney, Australia, pp 102–107.

34. Cha M, Kwak H, Rodriguez P, et al (2007) I tube, you tube, everybody tubes. In: *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*. ACM Press, New York, USA, pp 1–14.

35. Ahsan S, Singh V, Ott J (2014) Characterizing internet video for large-scale active measurements. arXiv Prepr arXiv14085777v1.

36. ITU-T (2012) Infrastructure of audiovisual servicescoding of moving video. ITU-T Recommendation H.264.

Author biographies

ALA'A AL-HABASHNA received his Master of Engineering degree from Memorial University of Newfoundland in 2010, and his PhD degree from Carleton University in 2018, both in Electrical and Computer Engineering. Currently, Dr. Al-Habashna is a Computer Vision Researcher at Statistics Canada and an Adjunct Professor at Carleton University, Ottawa, Canada. He received the fellowship of the School of Graduate Studies at Memorial University of Newfoundland in 2010. Furthermore, Dr. Al-Habashna has won multiple awards including excellence and best-paper awards. He worked as a reviewer for many conferences and journals including the IEEE ICC and Multimedia Tools and Applications journal. Dr. Al-Habashna is also the co-chair for the Communications and Networking Symposium (CNS) and a Technical Program Committee member in the International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS). His current research interests include discrete-event modeling and simulations, signal detection and classification, cognitive radio systems, 5G wireless networks, communication networks architecture and protocols, IoT applications, multimedia communication over wireless networks, and machine learning and computer vision.

GABRIEL A. WAINER, FSCS, SMIEEE, received the M.Sc. (1993) at the University of Buenos Aires, Argentina, and the Ph.D. (1998, with highest honors) at UBA/Université d'Aix-Marseille III, France. In July 2000, he joined the Department of Systems and Computer Engineering at Carleton University (Ottawa, ON, Canada), where he is now Full Professor and Associate Chair for Graduate Studies.

He has held visiting positions at the University of Arizona; LSIS (CNRS), Université Paul Cézanne, University of Nice, INRIA Sophia-Antipolis, Université de Bordeaux (France); UCM, UPC (Spain), University of Buenos Aires, National University of Rosario (Argentina) and others. He is one of the founders of the Symposium on Theory of Modeling and Simulation, SIMUTools and SimAUD. Prof. Wainer was Vice-President Conferences and Vice-President Publications and is a member of the Board of Directors of the SCS. Prof. Wainer is the Special Issues Editor of SIMULATION, member of the Editorial Board of IEEE Computing in Science and Engineering, Wireless Networks (Elsevier), Journal of Defense Modeling and Simulation (SCS). He is the head of the Advanced Real-Time Simulation lab, located at Carleton University's Centre for advanced Simulation and Visualization (V-Sim). He has been the recipient of various awards, including the IBM Eclipse Innovation Award, SCS Leadership Award, and various Best Paper awards. He has been awarded Carleton University's Research Achievement Award (2005, 2014), the First Bernard P. Zeigler DEVS Modeling and Simulation Award, the SCS Outstanding Professional Award (2011), Carleton University's Mentorship Award (2013), the SCS Distinguished Professional Award (2013), and the SCS Distinguished Service Award (2015). He is a Fellow of SCS.