Mobile Netw Appl (2018) 23:543–559
DOI 10.1007/s11036-017-0906-x

Improving Video Transmission in Cellular Networks with Cached and Segmented Video Download Algorithms

Ala'a Al-Habashna[1] · **Gabriel Wainer**[1]

Published online: 13 September 2017

Abstract Satisfying the increasing demand for high data rates has become a main challenge for cellular network operators. Moreover, the demand for video traffic in cellular networks has been continuously increasing. In 2016, wireless video accounted for more than half of the total mobile data traffic, and it is expected to further increase in the upcoming years. This will increase the amount of data to be transmitted in cellular networks, and further increase the challenge for cellular network operators. Device-to-Device (D2D) communication, introduced by the Long Term Evolution-Advanced (LTE-A) standard, is a new communication technique that allows direct communication between devices in cellular networks without going through the Base-Station (BS). We present two algorithms for improving the throughput of video transmission in cellular networks. The algorithms are called *Cached and Segmented Video Download* (CSVD), and *DIStributed, Cached, and Segmented video download* (DISCS). The algorithms send segments of video files to selected User Equipments (UEs) in the cellular network, to cache and forward them to requesting UEs using D2D communication. We study the performance of both algorithms analytically in terms of the hit ratio. Furthermore, we use the Discrete EVent System Specification (DEVS) formalism to build an LTE-A network model and use the model to study the performance of CSVD and DISCS in terms of the cell's aggregate data rate as well as the average data rate. Simulation results show that CSVD and DISCS achieve significant performance improvements in terms of the aggregate and average data rates.

Keywords Cellular networks · LTE-A · Hit ratio · D2D communication

1 Introduction

Cellular networks have witnessed an increasing demand for higher data rates due to the improvements on mobile devices, the services provided, and the increasing number of users [1]. Satisfying these demands has become a challenge for cellular network operators. As the data rates provided by radio links has been approaching their theoretical capacity, it is difficult to improve them further. Furthermore, the scarcity of the radio spectrum in cellular networks makes it difficult to provide all the users with enough resources to achieve the desired performance. Despite the emergence of new techniques to increase the spectrum utilization such as opportunistic spectrum sharing and cognitive radio, the radio spectrum is still considered scarce. Innovative communication techniques and algorithms are still needed to improve performance.

This situation is made worse by the increasing popularity of video applications and the improvements on smart devices. Wireless and mobile users increasingly use their phones and tablets to watch videos. This has increased the video traffic over cellular networks, which accounted for 60% of the total mobile data traffic in 2016 [2]. Video traffic is expected to account for 78% of the world's mobile data traffic by 2021 [2]. Hence, new techniques are needed to help serving the video traffic, which is becoming the main source of data traffic.

Device-to-Device (D2D) communication, introduced by the Long Term Evolution-Advanced (LTE-A) standard [3], is a new communication paradigm that allows direct

✉ Ala'a Al-Habashna
alaaalhabashna@sce.carleton.ca

Gabriel Wainer
gwainer@sce.carleton.ca

[1] Department of Systems and Computer Engineering, Carleton University, Ottawa, ON, Canada

communication between nearby User Equipments (UEs) without routing the traffic through the Base-Stations (BS) and the network infrastructure. This direct communication between UEs provides many benefits. Capacity gains can be achieved by sharing spectrum resources between cellular and D2D users. Data rate gains can be achieved due to proximity and potentially favorable propagation conditions. There have been various efforts to combine current standards with D2D communications in order to achieve such gains [4–7].

We present two algorithms for improving the throughput of video transmission in cellular networks [8–10]. Our algorithms are inspired by the architecture proposed in [11], which exploits D2D communications for BS-assisted D2D video transmission. The main idea is to cache popular video contents in the UE devices. If cached video files are requested, they will be sent to the requesting devices from the caching UEs over D2D links.

The first algorithm is called *Cached and Segmented Video Download* (CSVD) [9, 10]. In CSVD, the cell is divided into clusters. Selected UEs in each cluster are chosen as Storage Members (SMs). When an SM requests a video file, the BS will send segments of the video file to the SM over a cellular link. The SM will be requested to cache the downloaded file. When a cached file is requested by other UEs in the cluster, the segments will be sent over D2D links from the caching SM.

The second algorithm is called *DIStributed, Cached, and Segmented video download* (DISCS) [10]. In DISCS, a video file is also divided into segments, which are distributed over multiple SMs to be cached and forwarded to the requesting UE. This provides further parallelism when transmitting video files and further load balancing among SMs, which speeds up the transmission process. Furthermore, in DISCS, the SMs will be required to receive and forward pieces when asked for assistance (as opposed to just forwarding pieces they already have) which helps accumulating video files faster in the distributed cache.

We built a suite of models using the Discrete EVent System Specification (DEVS) formalism [12, 13] for an LTE-A cellular network that employs CSVD and DISCS. System level simulations were performed to evaluate the performance of CSVD and DISCS with different parameters and under different simulation scenarios. Simulation results show that the proposed algorithms can achieve significant improvements over conventional transmission methods in terms of both the cell's aggregate data rate as well as the average data rate per user.

In this paper, we extend our work and develop analytical models to evaluate the efficiency of caching video files in UEs with both algorithms in terms of the hit ratio. Analytical closed-form expressions were derived for the hit ratio of both algorithms. We also evaluate the hit ratio of both algorithms through computer simulations. Furthermore, we study the impact of UE mobility on performance, and provide an improved operation for the algorithms to avoid performance degradation that can be caused by UE movement.

In the following, we list the main contributions of this paper over our previous work,

- Analytical modeling of video caching with CSVD and DISCS and deriving closed-form expressions for the hit ratio of video caching with both algorithms
- Evaluating the hit ratio of both algorithms with computer simulation, and comparing simulation and analytical results
- Analyzing the impact of the number of requests on the performance of the algorithms
- Evaluating the overhead of the proposed algorithms
- Studying the impact of UE movement on performance
- Proposing an improved operation for CSVD and DISCS under UE movement and evaluating the performance with the proposed operation

The rest of this paper is organized as follows; in section 2 we provide an overview of the related work. In section 3, we present the CSVD and DISCS algorithms. In section 4, we describe modeling an LTE-A cellular network with DEVS. In section 5, we study the hit ratio of the proposed algorithms using both analytical modeling as well as the developed DEVS model. In section 6, we discuss the operation of the proposed algorithms in the case of UE movement, the implementation of the algorithms, and the power consumption of SMs. In section 7, we present simulation scenarios and results for the cell's aggregate data rate and average data rate.

2 Related work

Increasing the demand for video content in cellular networks has further increased the data rate requirements [2]. Decreasing the cell size is a solution that has been implemented to increase the frequency efficiency and provide higher data rates [14]. However, as the cell size decreases, it becomes more difficult to handle mobility and control interference. Moreover, decreasing the cell size increases the number of cells in a given area, and the cost of implementing a high-capacity wired backbone between the cells. As such, new solutions are required to increase the provided data rates and help serving video traffic that would take up a major portion of the overall data traffic.

Caching popular video files at the BSs or mobile switches has been employed to improve the transmission of video traffic in cellular networks [15]. When a popular file is cached at the BS, it will be available when requested by UEs, which eliminates the need for requesting the video from the content server and reduces the amount of traffic on the backhaul network. However, this solution does not reduce the amount of traffic between the BSs and UEs over cellular frequency links.

In [16], the authors proposed caching on the end-user devices the content expected to be requested in the future. Using

an adaptive popularity-based video caching strategy enables strong collaboration between users and the service environment. This can be exploited to provide higher quality of service to end-users. Video contents are dynamically cached in home-boxes following users' demands, allowing their delivery from optimal places. Although such approach helps reducing the amount of video traffic, cached contents can be only used locally by the caching devices, and cannot be exploited by others in the network.

Caching video content on central servers close to the users was proposed in [17]. Furthermore, the authors proposed employing simultaneous coded-multicasting by the central server, to satisfy the requests of several users with different demands with a single multicast stream. Nevertheless, such scheme requires a carefully designed placement phase in order to create coded-multicasting opportunities. As the content placement is performed before the actual user demands are known, it has to be designed carefully such that these coded-multicasting opportunities are available simultaneously for all possible requests.

Despite the improvement achieved by the approaches above, the increasing number of users and demand for video content in cellular networks make it challenging to serve video content with such client-server approaches. New and more scalable architectures are needed for video content distribution in cellular networks with high number of users and limited resources. As such, there has been a need for Peer-to-Peer (P2P) communication models. There has been some work on P2P communication in wireless ad hoc networks [18]. However, P2P communication has not been implemented in cellular networks until the recent emergence of D2D communications [4–7], as it allows direct communications between UEs in cellular networks. D2D communication depends on the participation of users for sharing contents. As such, it is important to find different approaches to motivate such user involvement. We do not consider incentive mechanisms here, as this is a different research area that is out of the scope of this paper. There has been much research on incentive mechanisms to motivate such user involvement in D2D communication. For more information about the research in this area, the interested reader is referred to [19–21].

Employing D2D communication has been previously proposed in the context of mobile content delivery networks [22]. Mobile content delivery networks are cellular networks that employ devices designated as caching servers to provide nearby users with cached contents on demand. Content delivery from the caching devices could take place over D2D links. While this technology could help improving the data rates in cellular networks, it is costly as these designated devices need to be placed throughout the network, configured, and maintained.

There has been some work on the use of D2D for P2P video communication in cellular networks [23–25]. Most of the previous work adapts algorithms that are similar to P2P streaming protocols on wired networks that involve the dissemination of buffer maps and video pieces between peers. While such protocols are suitable for communication on wired networks, they involve too much signaling and transmission (such as dissemination of buffer maps) to be appropriate for UEs with limited power and resources. Furthermore, the previous work considers small-scale networks (up to 10 UEs). The number of UEs of an LTE-A cell in urban areas is usually higher. Here, we show that using clustering and BS assistance, the potential of collaborative D2D communication between UEs is significant.

The architecture, proposed in [11], employs D2D communication to improve the throughput of video transmission and overcome the problem of rapidly increasing wireless video traffic. In this architecture, the cell is divided into clusters, and each cluster contains a group of nodes that can exchange information with each other using D2D links. The nodes in each cluster can cache video files. When a video file is requested by a UE, the BS will check to see if the file is stored in the virtual storage of that cluster. If the requested file is found, it will be transmitted from the UE that has the file to the requesting UE over a D2D link. The network model in [11] is oversimplified and the original architecture is limited; it assumes that the files are pre-cached in the nodes. As such, the architecture needs a complex and carefully designed placement phase. Since the content placement is performed before the actual user demands are known, it has to be designed carefully so that users make use of the cached content. The architecture also assumes that complete files are cached and exchanged between the network nodes. Furthermore, they did not define a messaging protocol between the UEs and BS to exchange such video files. Instead, a simple model was used to study the performance of the architecture analytically.

Here, we present two algorithms for improving video transmission over cellular networks, namely, CSVD and DISCS. The algorithms employ an improved architecture of the one described above. Instead of caching complete files, video files are treated as segments, and multiple copies of a file can be cached at multiple SMs. Segments of a video file can also be distributed among multiple SMs, as in DISCS. We assume that no files are cached in the beginning, and that files are stored upon request. The algorithms define how the files are cached and exchanged among the UEs. Only selected UEs in each cluster are used for caching to reduce inter-cell and inter-cluster interference. A complete detailed protocol has been defined, including a variety of messages necessary for this communication, and a complete definition of the protocol is described.

In CSVD, the cell is divided into clusters. Selected UEs in each cell are chosen as SMs. When an SM requests a video file, the BS will send segments of the video file to the SM over a cellular link. The SM will be requested to cache the downloaded video file. When a cached file is requested by other UEs in the cluster, the segments will be sent over D2D

links from the cluster's cache. In DISCS, a video file is divided into segments. The segments of a video file are distributed over multiple SMs to be cached and forwarded to the requesting UE. This provides further parallelism when transmitting video files and further load balancing among SMs, which speeds up the transmission process. Furthermore, in DISCS, the SMs will be required to receive and forward pieces when asked for assistance (as opposed to just forwarding pieces they already have) which helps accumulating video files faster in the distributed cache.

We developed analytical models to evaluate the efficiency of caching video files in UEs with both algorithms in terms of the hit ratio. Analytical closed-form expressions were derived for the hit ratio of both algorithms. We also used the DEVS formalism [12, 13] to build a model for an LTE-A network that employs CSVD and DISCS. DEVS provides a sound formal framework for modeling generic dynamic systems. DEVS includes hierarchical, modular, and component-oriented structure. It provides formal specifications for defining 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. This modular nature is a very useful property for modeling and simulating LTE-A networks. The network model can be built using different submodels; each one implements a different component of the wireless network such as the BS or the UE. Each one of these submodels can be tested and verified independently, and integrated into the whole model. These submodels can also be reused in other LTE-A network models. Furthermore, each submodel, such as the BS, can also be implemented using multiple submodels in a hierarchical manner. Each one of these submodels implements a certain subcomponent or functionality. This makes it easy to design, implement, and test LTE-A network models.

We used the CD++ toolkit [13] to implement our LTE-A network DEVS model. The developed model was used to study the performance of the CSVD and DISCS algorithms in terms of the cell's aggregate and average data rates, and measure the improvements achieved over conventional video transmission approaches by running various simulations. In the following section, we provide a detailed description of the CSVD and DISCS algorithms, and show how they operate.

3 The CSVD and DISCS algorithms

Both CSVD and DISCS are designed for scenarios where there is a high density of users in the cell, such as,

- Sport events in which users want to download instant replays from this event, or videos of other events taking place at the same time
- Live concerts with detailed video feeds of the arena
- Massive religious events (i.e., a Pope's Mass in St. Peter's Basilica in the Vatican)
- Large political events (i.e., election results or inauguration speeches)
- University convocations

Consider one cell in a cellular network, in which the BS is in the middle of the coverage area, as in Fig. 1. In this analysis, we consider the case where UEs are stationary. In section 6, we discuss the operation of the algorithms under UE movement. At the beginning, the BS starts by dividing the cell into clusters, as follows,

1) The BS logically divides the coverage area into non-overlapping subareas. Each one of these will be a cluster.
2) The BS sends a broadcast *Clustering* message telling the UEs that a cluster formation is about to start.
3) The UEs reply with a *Clustering Response* message indicating their GPS location.
4) The BS assigns UEs to clusters based on their locations. For instance, Fig. 1 shows that the cell is divided into 9 non-overlapping subareas (9 white squares). UEs that are located in the top left subarea will form the first cluster, while UEs in the top middle subarea will form the second cluster, and so on. This will result in 9 clusters. Furthermore, the BS selects the UEs in the central area

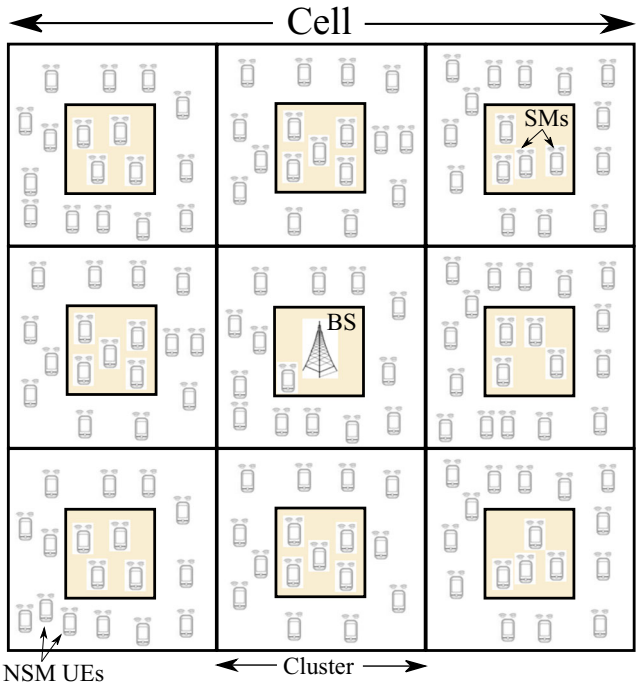

Fig. 1 Cell divided into 9 clusters

of each cluster (shaded square within each subarea in Fig. 1) as SMs of that cluster. Only UEs in the middle of each cluster are choosen as SMs, in order to prevent inter-cell and inter-cluster interference when the SMs transmit to other UEs in the same cluster over D2D links.

After dividing the cell into clusters, the transmission phase starts. The UEs send their requests to download video files to the BS. The BS processes a download request, and responds differently depending on the case. We consider four different cases. Three of the cases below apply to both CSVD and DISCS, and one of the cases is only used in DISCS. The cases are as follows,

- Send With Assistance (SWA): if the video file (or a part of it) is available in any of the SMs of the cluster, the BS will ask these SMs to send the pieces they have to the requesting UE over D2D links.
- Send To an SM (STSM): if the requested file is not available in the distributed cache (or more copies need to be cached), and the requesting UE is an SM, the BS will send the file to that SM over a cellular link, and ask the SM to cache the video file. Cached video files will be available for UEs in the cluster when requested later.
- Distribute to SMs (DTSMs): this case is only used in DISCS. 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 pieces (as the file is popular), and asks them to forward the received pieces to the requesting UE.
- Send To a UE (STUE): otherwise, the BS will send the file directly to the requesting UE over a cellular link.

In the following, we discuss the different cases described above in detail.

3.1 Send with assistance

In this case, the download process, which is shown in Fig. 2, takes place as follows:

1) The UE sends a *Download Request* message to the BS.
2) As this file has already been sent before to an SM to cache it, the BS has a *MetaInfo* file that describes the parameters for the download session of this video file. Table 1 shows the fields of the *MetaInfo* file.

   The fields in the *MetaInfo* file represent the parameters for this download session. The BS then sends a *Handshake* message to the requesting UE. The *Handshake* message contains the *MetaInfo* file. The *Handshake* message is sent to inform the UE that the *Download Request* was received and processed by the BS, and to provide the UE with the parameters of this download session that are included in the *MetaInfo* file (Table 1).
3) The BS will check its database to find out which of the SMs have pieces of the cached file. Then, the BS will send an *Assistance Request* message to these SMs asking for their assistance to send pieces to the requesting UE. The *Assistance Request* message has a field indicating the numberes and indexes of the pieces that the SM should send to the requesting UE.
4) The SMs will send a *Response* message. The SMs will indicate whether or not they are available to assist with this download session. The *Response* message also contains a field that indicates the maximum number of outstanding assists the BS should send, i.e., the maximum number of assists this SM can handle at a time.
5) The BS and the SMs start sending the pieces to the requesting UE. Each time the BS wants an SM to forward new piece(s) of the video file, it will send that SM an *Assistance Request* message indicating the piece(s) to

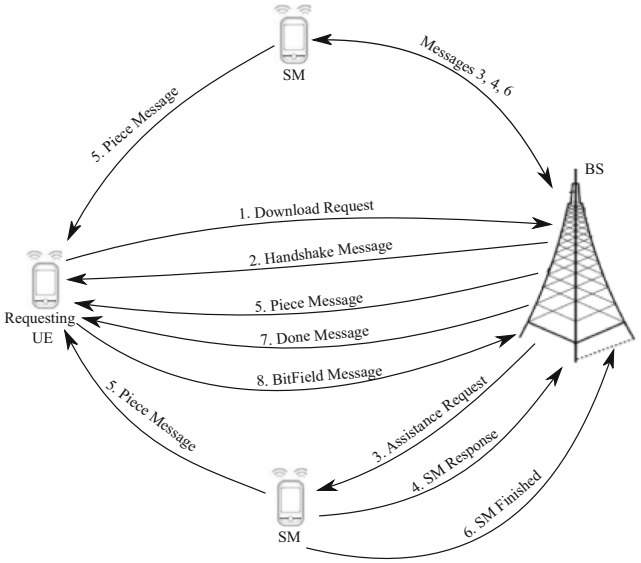

Fig. 2 SWA case

Table 1 MetaInfo file

Field	Description
File size	The file size in bytes
Number of pieces	The number of pieces
Piece size	The piece size in bytes
Last piece size	Last piece size in bytes
File name	A string representation of the file
Info	A dictionary that describe the file

forward. Each *Piece* message has an index that identifies that piece.

6) When an SM finishes sending piece(s), it will send an *SM_Finished* message to the BS, confirming the transmission of the piece(s).
7) When the BS receives *SM_Finished* for the pieces from the SMs participating, and when it finishes sending its pieces, it will send a *Done* message to the requesting UE.
8) When the requesting UE receives a *Done* message, it will send a *BitField* message to the BS indicating the pieces it has received.

The BS should keep in its database the following,

- A list of the clusters
- A list of the members and SMs of each cluster
- A list of the cached video files/segments, and caching nodes

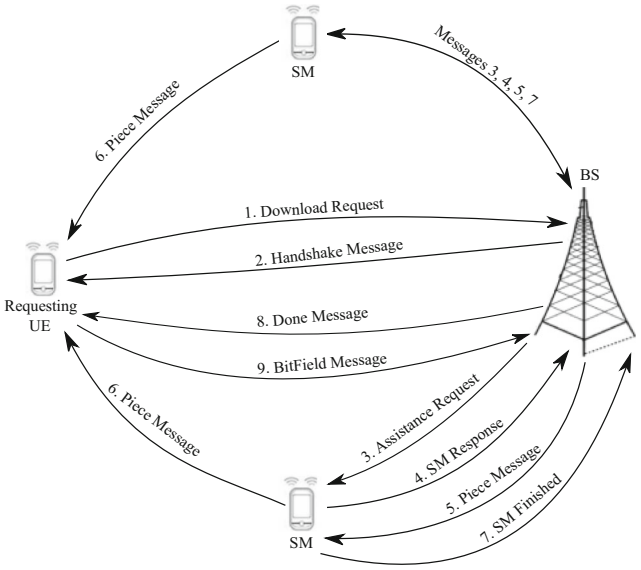

Fig. 3 DTSMs case

3.2 Send to an SM

In this case, the file transfer starts by the SM sending a *Download Request* message to download a video file. After receiving the request, the BS will start a session with this UE. If this is the first time an SM requests this file, the BS creates a *MetaInfo* file that contains information about this download. The *MetaInfo* file is the same as in Table 1. The BS also creates a *Handshake* message and sends it to the requesting SM. The BS then starts sending pieces directly to the SM over cellular link. A Save bit in the *Piece* message is always set to indicate that the SM should cache the received piece. The SM keeps a *BitField* to keep track of the received pieces. After sending all the pieces, the BS will send a *Done* message. When the SM receives the pieces and the *Done* message, it will send a message containing the *BitField* to the BS.

3.3 Distribute to SMs

As mentioned above, this is the main contribution of the DISCS algorithm. In this case, a popular video file (for instance, a video file that was requested *n* times) is requested by a non-SM (NSM) UE. The BS distributes the video file pieces over SMs and asks them to cache the pieces and forward them to the requesting UE (as the file is popular, and distributing it will be beneficial for the cluster). The download process (Fig. 3) is as follows,

1) The UE sends a *Download Request* message to the BS.
2) The BS creates a *MetaInfo* file that describes the parameters for the download session of this video file (as in Table 1). The BS then sends a *Handshake* message (containing the *MetaInfo* file) to the requesting UE.
3) The BS then sends *Assistance Request* messages to the SMs of the cluster asking their help to send the pieces to the requesting UE. There is a field in the message that is set to indicate that this is a "receive and forward" request, i.e., the SM is needed to receive the piece, cache it, and forward it to the requesting UE.
4) The SMs will send a *Response* message to indicate their availability for assistance, and to indicate the maximum number of outstanding assists the BS can send.
5) The BS then starts distributing the pieces over the SMs. Each *Piece* message has an index that identifies that piece.
6) When an SM receives a piece, it will cache it, and send it to the requesting UE over a D2D link.
7) When an SM finishes sending piece(s), it will send an *SM_Finished* message to the BS, comfirming the transmission of the piece(s).
8) When the BS receives *SM_Finished* for all the pieces from the SMs participating, it will send a *Done* message to the requesting UE.
9) When the requesting UE receives a *Done* message, it will send a *BitField* message to the BS indicating the pieces it has received.

This case helps speeding up accumulation of popular video files in the distributed cache of the cluster. It also increases the 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, speed up the transmission, and consequently increase the average data rate.

In addition to the data the BS needs to keep in its database for the CSVD (list of the clusters, list of the members and SMs of each cluster, and list of cached

video files/segments), the BS keeps track of the number of times video files were requested recently.

3.4 Send to a UE

In this case, the requesting UE is not an SM, and the file is not available in the cluster. Hence, the BS will transmit the file directly to the requesting UE over cellular links. This case is similar to STSM. However, in this case, the save bit is always zero in the *Piece* message so that the piece will not be cached.

3.5 The SVD algorithm

We refer to conventional download process as Segmented Video Download (SVD), as video files are sent in pieces. In SVD, file caching and D2D communications are not used. Instead, video files are always sent as in STUE.

4 Modeling an LTE-A network with DEVS

Fig. 4 shows the DEVS coupled model definition of an LTE-A network. At the top level, we have a *Cell* coupled model, which contains *BS*, *Transmission Medium*, and many *UE* coupled models. It also contains *Cell Manager* and *Log Manager* atomic models.

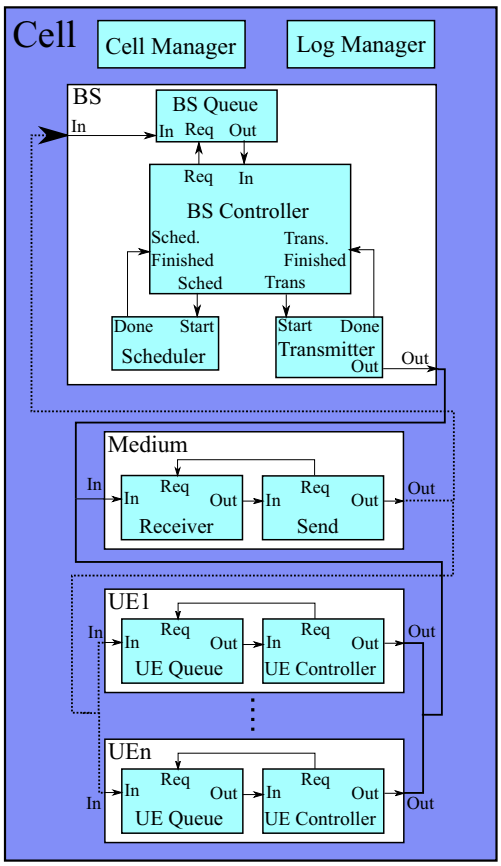

Fig. 4 DEVS model of the cellular network

The BS coupled model is in charge of modeling the BS in the cell. It is composed of four atomic models: *BS Queue, BS controller, Scheduler, and Transmitter*. Messages received by the BS are buffered at the *BS Queue* atomic model. The *BS Queue* checks the destination address of a received message. If it matches that of the BS, the message will be buffered, otherwise it will be discarded. The *BS Controller* controls the various BS components, processes received messages, and it implements the algorithms above (e.g., steps 2, 3, 5, and 8 in Fig. 3). Every Transmission Time Interval (TTI), which is 1 ms, the BS processes received messages and asks the *Scheduler* to schedule the messages to be sent in the next TTI. Every TTI, the BS Controller also asks the *Transmitter* to send messages that were scheduled for transmission during this TTI.

The UEs in the cell are modeled with the *UE* coupled models. A *UE* coupled model contains two atomic models: *UE Queue*, and *UE Controller*. Received messages are buffered at the *UE Queue*. The *UE Controller* is where the UE part of the algorithm is implemented (e.g., steps 1, 4, and 7 in Fig. 3).

The *Medium* model represents the transmission medium in the cell. It receives a message sent from the BS or any UE and broadcasts it to the other receivers (BS/UEs) in the cell. As mentioned above, the queue of the BS and the UEs will use the destination address to recognize their messages.

The *Log Manager* logs simulation events and record statistics during the simulations. The *Cell Manager* atomic model initializes and updates the parameters of the cellular DLs and uplinks (ULs) between the BS and the UEs, as well as the D2D links between the UEs. We consider Path loss and shadowing here. The urban macro propagation model [26] was used for cellular links with a DL operating carrier frequency of 900 MHz, and a transmission bandwidth of 10 MHz. According to [26], the propagation model (L) is given by,

$$L = 40 \times \left(1{-}4 \times 10^{-3} \times Dhb\right) \times log_{10}(d) \\ \text{-}18\ log_{10}(Dhb) + 21\ log_{10}(f) + 80dB, \quad (1)$$

where d is the BS-UE separation in kilometers, f is the carrier frequency in MHz, and Dhb is the BS antenna height in meters, measured from the average rooftop level. The path loss, PL, then can be calculated as,

$$PL = L + LogF, \quad (2)$$

where $LogF$ is a log-normally distributed shadowing with standard deviation of 10 dB. The received signal then can be calculated as,

$$P_{RX} = P_{TX}{-}MAX(PL{-}G_{TX}{-}G_{RX}, MCL), \quad (3)$$

where P_{RX} is the received signal power, P_{TX} is the transmitted signal power, G_{TX} is the transmitter antenna gain, and G_{RX} is the receiver antenna gain, and MCL is the minimum coupling loss.

Considering Additive White Gaussian Noise (AWGN), the link data rate, R, can be calculated as,

$$R = B \times \log_2\left(1 + \frac{P_{RX}}{P_n}\right), \tag{4}$$

where P_n is the noise power and B is the transmission bandwidth.

For D2D transmission, we used the D2D channel model at 24 GHz, defined in [27]. According to that model, the path loss for D2D links, PL_{D2D}, can be calculated as,

$$PL_{D2D} = 60.05 + 1.95^{*}\ 10\log_{10}(d) + LogF_{D2D}, \tag{5}$$

where d is the transmitter-received distance in meters, and $LogF_{D2D}$ is a log-normally distributed shadowing with standard deviation of 4.3 dB. The data rate is calculated considering AWGN.

We used the CD++ toolkit to implement our model. With CD++, atomic models are developed using the C++ programming language and can be incorporated into the class hierarchy. 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 DLs and ULs, D2D links, download sessions the BS has with UEs, cell clusters, exchanged message, etc.

5 Analyzing the hit ratio of the proposed algorithms

In this section, we analyze the hit ratio of the proposed algorithms, i.e., the fraction of video file requests that are found and satisfied from the cluster's cache. As mentioned before, the cell is divided into clusters. Only SMs in each cluster are used to cache video files. We assume that no video files are initially cached, and that video files are cached as requested. A hit occurs if a requested video segment is delivered from the cluster's cache (found in one of the SMs of that cluster). A miss occurs in the case the requested segment is not found in cluster's cache, and hence, sent from the BS. We provide analytical models for the network and develop closed-form expressions for the hit ratio of CSVD and DISCS. Furthermore, we run simulations using the DEVS model presented in the previous section to evaluate the hit ratios.

In the following, we present the analytical models and the derivation of the expressions for the hit ratio of CSVD and DISCS. Afterwards, we present the results obtained from the analytical expressions as well as the simulation results from the DEVS model.

5.1 Hit ratio of CSVD

Consider a cell with N UEs in the cluster. Furthermore, consider that time is divided into slots, each with a length T_s seconds. Each UE wants to request one video file, and each UE sends its video file request during any slot with probability, P_{req}, until success. The UEs generate requests to download videos from a list that contains F video files. The popularity of videos is generated according to a Zipf distribution to simulate a variable popularity of files, as it has been established that this is a good model for video files popularity [28]. Using this distribution, some files are requested more often than others. The Zipf exponent, β, controls the relative popularity of the files. According to the Zipf distribution, the relative popularity of the f^{th} file is,

$$P_f(\beta, F) = \frac{f^{-\beta}}{\sum_{i=1}^{F} i^{-\beta}}. \tag{6}$$

From now on, we will refer to the relative popularity of a file, i.e., the probability that a UE selects file f, as P_f. Without loss of generality, a file is assumed to be one piece. P_{SM} is the probability that a UE is an SM. This depends on the ratio of the central area of the cluster (see Fig.1) to the total cluster area. P_{NSM} is the probability a UE is an NSM. Consider that it takes R time slots for all the UEs to send their download requests.

The probability that a UE requests during slot r, P_r, follows a geometric distribution, as follows,

$$P_r = \left(P_{req}\right)\left(1-P_{req}\right)^{r-1}. \tag{7}$$

As requesting in slot r and choosing file f are independent events, the probability that a UE requests during slot r, and selects file f, $P_{r,f}$, can be given as,

$$P_{r,f} = P_r \times P_f. \tag{8}$$

Similarly, the probability that a UE is an SM, that requests in slot r, and selects file f, $P_{SM,r,f}$, can be given by,

$$P_{SM,r,f} = P_{SM} \times P_r \times P_f. \tag{9}$$

When a UE sends a request in slot r to download video file f, the request will be a hit if an SM has requested the same file in one of the previous slots. As such, the probability a certain request that takes place in slot r is satisfied from the cluster's cache, $P_{Hit,r,CSVD}$, is,

$$P_{Hit,r,CSVD} = \sum_{f=1}^{F} \Big(\Pr[\text{A UE } \textit{requests file } f \textit{ in slot } r] \times \Pr[f \textit{ is cached} \text{ by at } \textit{least} \text{ one SM}]\Big). \tag{10}$$

Recalling that a file is either cached in the cluster's cache or not, $P_{Hit,r,CSVD}$, can be also give as,

$$P_{Hit,r,CSVD} = \sum_{f=1}^{F} \Big(\Pr[\text{A UE } \textit{requests file } f \textit{ in slot } r] \times (1-\Pr[f \textit{ is} \text{ not } \textit{cached} \text{ by } \textit{an} \text{ SM}])\Big). \tag{11}$$

Given that a file, f, is cached with CSVD, if it was requested in a previous slot by an SM, $P_{Hit,r,CSVD}$ can be given as,

$$\begin{aligned} P_{Hit,r,CSVD} &= \sum_{f=1}^{F} \Big(\Pr[\text{A UE } \textit{requests file } f \textit{ in slot } r] \\ &\quad \times \Pr[\text{At } \textit{least} \text{ one SM } \textit{prevously requested } f] \Big) \\ &= \sum_{f=1}^{F} \Big(\Pr[\text{A UE } \textit{requests file } f \textit{ in slot } r] \\ &\quad \times (1 - \Pr[\text{No SM } \textit{prevously requested } f]) \Big) \\ &= \sum_{f=1}^{F} \Big[p_{r,f} \times \big(1 - P_{f \text{ not } cached|_r,CSVD}\big) \Big], \end{aligned} \tag{12}$$

where $P_{f \text{ not } cached|_r,CSVD}$ is the probability that a file, f, is not cached at time slot r, which can be give by,

$$\begin{aligned} P_{f \text{ not } cached|_r,CSVD} &= \Pr[\text{No SM } \textit{prevously requested file } f] \\ &= \prod_{u=1}^{r-1} \binom{N-1}{0} \big(P^0_{SM,u,f}\big) \big(1 - P_{SM,u,f}\big)^{N-1} \\ &= \prod_{u=1}^{r-1} \big(1 - P_{SM,u,f}\big)^{N-1}. \end{aligned} \tag{13}$$

Assuming that no files are cached in the beginning, only requests that take place at slot r = 2 or after can be satisfied from the cluster's cache. By substituting (13) in (12), and taking the summation over time slots 2 and after, the hit ratio for CSVD; the probability that a request is satisfied from the cluster's cache, $P_{hit,CSVD}$, can be given as,

$$P_{hit,CSVD} = \sum_{f=1}^{F} \sum_{r=2}^{R} P_{r,f} \times \left[1 - \prod_{u=1}^{r-1} \big(1 - P_{SM,u,f}\big)^{N-1} \right]. \tag{14}$$

5.2 Hit ratio of DISCS

For DISCS, consider n is the number of times a video file should be requested by UEs to be considered popular and to be distributed to SMs. In this case, a file f is delivered through the cluster's cache (SWA and DTSMs combined) if it was previously requested by an SM, or if it was previously requested by NSMs (n-1) times. As such, with DISCS, a file f is not cached at slot r, if f was not requested by an SM in the previous time slots, and it was not requested by n or more NSMs in the previous time slots. Without loss of generality, consider, n = 2. This means that if the video was requested once by an SM or was requested more than once by NSMs in the previous slots, it will be delivered from the cluster's cache (SWA). In this case, the probability that a file is not cached at time slot r,$P_{f \text{ not } cached|_r,DISCS}$, can be given by,

$$\begin{aligned} P_{f \text{ not } cached|_r,DISCS} &= \Pr[\textit{file was} \text{ not } \textit{prevously requested} \text{ by } \textit{an} \text{ SM}] \\ &\quad \times \Pr[\textit{file was} \text{ not } \textit{prevously requested} \text{ by } \textit{NSMs more than once}] \\ &= \prod_{u=1}^{r-1} \big(1 - P_{SM,u,f}\big)^{N-1} \times \Big[\prod_{u=1}^{r-1} \big(1 - P_{NSM,u,f}\big)^{N-1} \\ &\quad + \sum_{u=1}^{r-1} \binom{N-1}{1} \big(P^1_{NSM,u,f}\big) \big(1 - P_{NSM,u,f}\big)^{N-2} \times \prod_{\substack{j=1 \\ j \neq u}}^{r-1} \big(1 - P_{NSM,j,f}\big)^{N-1} \Big] \\ &= \prod_{u=1}^{r-1} \big(1 - P_{SM,u,f}\big)^{N-1} \times \Big[\prod_{u=1}^{r-1} \big(1 - P_{NSM,u,f}\big)^{N-1} \\ &\quad + \sum_{u=1}^{r-1} (N-1) P_{NSM,u,f} \big(1 - P_{NSM,u,f}\big)^{N-2} \times \prod_{\substack{j=1 \\ j \neq u}}^{r-1} \big(1 - P_{NSM,j,f}\big)^{N-1} \Big]. \end{aligned} \tag{15}$$

As with CSVD, the probability a certain request that takes place in slot r is satisfied from the cluster's cache, $P_{Hit,r,DISCS}$, is given by,

$$\begin{aligned} P_{Hit,r,DISCS} &= \sum_{f=1}^{F} \Big(\Pr[\text{A UE } \textit{requests file } f \textit{ in slot } r] \\ &\quad \times \Pr[f \textit{ is cached} \text{ by at } \textit{least} \text{ one SM}] \Big) \\ &= \sum_{f=1}^{F} \Big(\Pr[\text{A UE } \textit{requests file } f \textit{ in slot } r] \\ &\quad \times (1 - \Pr[f \textit{ is} \text{ not } \textit{cached} \text{ by } \textit{an} \text{ SM}]) \Big) \\ &= \sum_{f=1}^{F} \Big(\Pr[\text{A UE } \textit{requests file } f \textit{ in slot } r] \\ &\quad \times \big(1 - P_{f \text{ not } cached|_r,DISCS}\big) \Big). \end{aligned} \tag{16}$$

Assuming that no files are cached in the beginning, only requests that take place at slot r = 2 or after can be satisfied from the cluster's cache. By substituting (15) in (16), and taking the summation for slots 2 and after, the hit ratio for DISCS; the probability that a request is satisfied from the cluster's cache, $P_{hit,DISCS}$, can be given as,

$$\begin{aligned} P_{Hit,DISCS} = \sum_{f=1}^{F} \sum_{r=2}^{R} P_{r,f} \times \Big[1 - \Big(& \prod_{u=1}^{r-1} \big(1 - P_{SM,u,f}\big)^{N-1} \times \Big(\prod_{u=1}^{r-1} \big(1 - P_{NSM,u,f}\big)^{N-1} \\ & + \sum_{u=1}^{r-1} (N-1) P_{NSM,u,f} \big(1 - P_{NSM,u,f}\big)^{N-2} \times \prod_{\substack{j=1 \\ j \neq u}}^{r-1} \big(1 - P_{NSM,j,f}\big)^{N-1} \Big) \Big) \Big]. \end{aligned} \tag{17}$$

5.3 Hit ratio results

In this section, we present the results for the hit ratio of the CSVD and DISCS algorithms obtained from both the analytical model and the simulations performed using the DEVS model.

In each simulation run, UEs are placed throughout the cell with a uniform distribution. The central area of each cluster forms 1/4 of the total area of the cluster. Hence, roughly, one fourth of the UEs in each cluster will be SMs.

At the beginning of each time slot, which is 10 s, each UE tries to send a request with a probability, $P_{req} = 0.2$, or wait till the next time slot. Simulation ends when all UEs send their requests and download the video files. Each UE will request one video file. Simulations were performed with various numbers of UEs in the cluster. A library of 500 files was used in the simulations. 100 simulation runs were performed. In addition to the mean values for the hit ratios obtained from the simulations, we show the margin of error values with 95% confidence interval.

Figures 5 and 6 show the analytical and simulation results for the hit ratio of CSVD and DISCS, respectively. As can be seen from the figures, the simulation results are very close to the analytical results, which means that the analytical model accurately represents the transmission scenario used. This also verifies and validates our developed LTE-A network DEVS model.

One can also see that for both CSVD and DISCS, the hit ratio increases by increasing the number of UEs in the cluster. This is because, with both algorithms, increasing the number of UEs increases the number of requests for video files and the number of SMs in each cluster. This increases the number of cached files in a cluster and the number of requests that would be satisfied from the cluster's cache, which consequently increases the hit ratio.

Fig. 7 shows the hit ratio (analytical results) for both algorithms versus the number of UEs in the cluster. As can be seen, the hit ratio for DISCS is always higher than that for CSVD. This is because in DISCS, video files are cached with the DTSMs case, in addition to the STSM case. As such, more video files will accumulate in the cluster's cache faster, which increases the number of video files satisfied from the cluster's cache, and increases the hit ratio.

As mentioned above, the Zipf distribution is used to simulate variable popularity of video files. The Zipf distribution has one parameter, namely the Zipf exponent. This exponent controls the relative popularity of files. When the Zipf exponent increases, content reuse increases. This is because higher

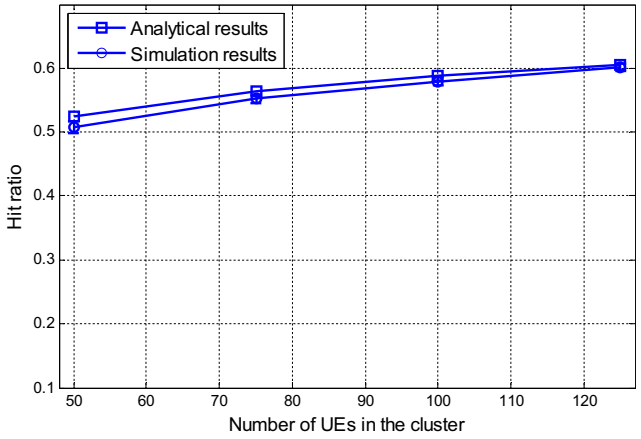

Fig. 6 Hit ratio of DISCS versus the number of UEs in the cluster. Zipf exponent = 1.5

exponent means that the popularity of the first files in the list will increase, and they will be requested more often.

Fig. 8 shows the hit ratio (analytical results) for both algorithms versus the Zipf exponent. As can be noticed, the hit ratio increases by increasing the Zipf exponent. As the number of popular files increases, content reuse increases because more files will be cached and consequently delivered later from the distributed cache rather than from the BS. This increases the portion of video files satisfied from the cluster's cache, which increases the hit ratio.

6 Operation with UE mobility and implementation

In section 3, we listed many scenarios where the proposed algorithms can be employed. Although users in such scenarios are usually static, some users (or all users sometimes) might be moving at pedestrian speed. For this reason, we discuss the operation of the proposed algorithms under user mobility in this section. As user mobility might reduce the performance

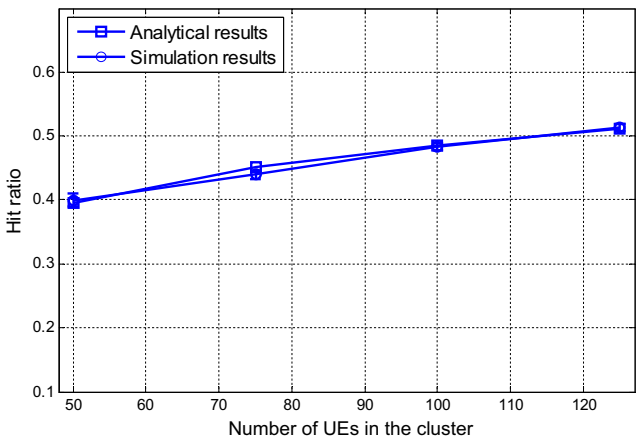

Fig. 5 Hit ratio of CSVD versus the number of UEs in the cluster. Zipf exponent = 1.5

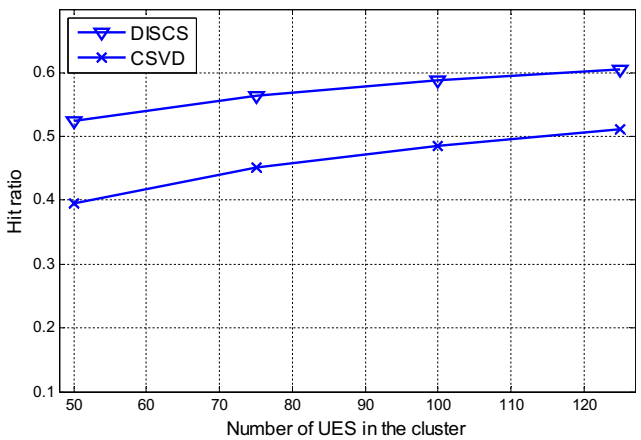

Fig. 7 Hit ratio of CSVD and DISCS (Analytical results) versus the number of UEs in the cluster. Zipf exponent = 1.5

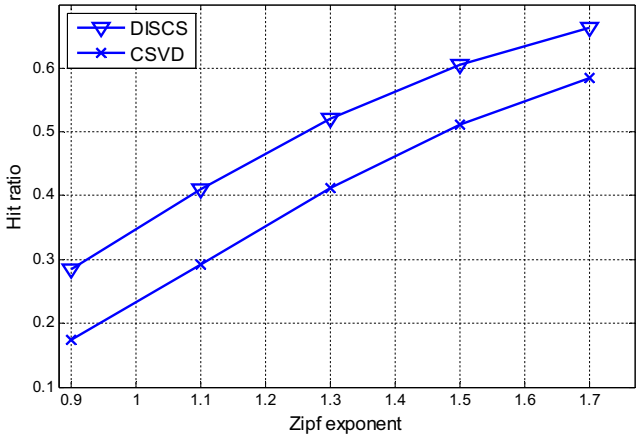

Fig. 8 Hit ratio of CSVD and DISCS (Analytical results) versus the zipf exponent. Number of UEs in the cluster = 125

improvement achieved by the proposed algorithms, we propose a minor update to the operation of the algorithms to be compatible with the case of user mobility. Afterwards, we discuss two important issues; the implementation of the algorithms in cellular networks, and the power consumption of SMs.

6.1 Operation of the algorithms with UE mobility

When UEs are moving, the BS needs to be aware of the location of the UEs. As such, moving UEs need to provide the BS with their updated location. This update can be sent periodically when the UE is continuously moving. As UEs are expected to be moving around average pedestrian speed, these updates can be sent to the BS periodically on a scale of hundreds of milliseconds or even on a scale of seconds. If the UE is not continuously moving, such update can be sent only when the UE changes its location by a certain threshold (for instance, a couple of meters). The BS will need such information to update the clusters in the case any of the following changes occur,

- An SM moves out of the central area of the cluster
- A non-SM moves into the central area of the cluster
- A UE moves from one cluster to another

A UE that is close to the cluster edge and moving towards the edge of the cluster will be marked as “transitioning”. UEs that are marked as transitioning will be sent video segments only through the BS (not from the cluster’s cache) until they transition to the new cluster.

The movement of UEs is expected to cause some degradation to the improvements achieved by the proposed algorithms. This is because under UE movement, SMs with cached content might now move out of the central area of the cluster and become NSMs.

To overcome this challenge, we propose a change to the operation of the proposed algorithms. Instead of setting the save bit to 1 only in the STSM and DTSMs cases, the save bit will always be set to 1. This means that every UE will be asked to cache received content. This is because when UEs are moving, any UE could be an SM (after entering the central area) and hence it would be very beneficial for any UE that becomes an SM to have cached content. This also increases the fairness of the algorithms, since in this case UEs that received content from the cluster’s cache could become SMs and provide others with video content later.

In section 7.3, we study the impact of UE movement on performance. Furthermore, we show how the improved operation overcomes this impact.

6.2 Implementing the algorithms in cellular networks

In this work, we study the potential gains that can be achieved in cellular networks when the CSVD and DISCS algorithms are employed. The proposed algorithms focus on the Radio Access Network (RAN), which is the main bottleneck in cellular networks with limited frequency resources that are shared among a large number of users.

The algorithms can be implemented in cellular networks by employing an entity at the BS, namely the CSVD proxy. The CSVD proxy checks the requests from the clients to the content server. The CSVD proxy stops the requests when the contents are found in the cell, as per the proposed algorithms. This entity also performs the BS part of the algorithm (for example, steps 2, 3, 5, and 8 in Fig. 3).

6.3 Power consumption of the SMs

Power consumption is an issue with D2D content sharing in general (not just with the proposed algorithms). This is due to the fact that receivers get content while helpers have to consume energy. The power consumption of the SMs should not be a major problem for the applicability of the proposed algorithms, for the following reasons,

1) The architecture employed by the proposed algorithms is designed to shorten the distance between the SMs and the requesting UEs. Due to clustering and selection of SMs in the central area of the clusters, SMs are relatively close to requesting UEs in the cluster. Because of this proximity between SMs and requesting UEs, SMs will not need to transmit with high transmission power, which reduces the power consumption.
2) Work in the literature on D2D communication usually assumes helpers are willing to participate. However, helpers in D2D communication may not comply with the requested assistance, especially considering power consumption. For this reason, providing incentive mechanisms for D2D communication is a topic that has been investigated by researchers. There are some existing

incentive mechanisms to motivate users' involvement (especially helpers) in D2D communication. For instance, some of these mechanisms assume that helpers are rewarded by the network operator, for example with money, better data plan, or other types of rawards. As prevously mentioned, we do not consider incentive mechanisms here, as this is a different research area that is out of the scope of this paper. The interested reader is referred to [19–21] for further information on this topic.

3) With the proposed algorithms, SMs in the cluster could request video content and get help from other SMs in the cluster, which can be an incentive for SMs to get involved. Furthermore, the improved operation increases the possibility of a requesting UE becoming a helper/provider in the future and sending content to another UE that was an SM.
4) Smart phones nowadays are equipped with batteries that last for a day, and can handle transmission of some video segments without losing much of their battery power. For instance, a simple test we performed with an iPhone 6s has shown that uploading a 1 min video twice, once over a cellular connection and once ovre a Wi-Fi connection consumes less than 1% of the battery power.
5) Different policies can be employed with the proposed algorithms to avoid over usage of the SMs power. For instance, an SM with a low power level can signal its unavailability via the *Response* message.

7 Simulation scenarios and results

System level simulations were performed to evaluate the performance of CSVD and DISCS, and compare them to SVD in terms of the DL cell's aggregate data rate and average data rate per user. We use more realistic transmission scenarios than the ones in section 5 to evaluate the data rates of the proposed algorithms in both the case of stationary UEs and the case of UE mobility.

7.1 Simulation scenarios

In the simulations, we consider a single LTE-A cell. The urban macro propagation model [26] was used for cellular links with a DL operating carrier frequency of 900 MHz, and a transmission bandwidth of 10 MHz. Table 2 shows the simulation parameters we used.

In the beginning of each iteration of the simulations, the UEs are uniformly distributed throughout the cell. The cell is divided into 9 clusters. According to their location in the cell, UEs are assigned to clusters as shown in Fig. 1. Furthermore, the UEs in the central area of each cluster are marked as SMs. The central area of each cluster forms 1/4 of the total area of the cluster. Hence, roughly, one fourth of the UEs in each cluster will be SMs. Each iteration in the simulations is divided to two phases; a transient phase, followed by a steady state phase. At the beginning of the transient phase, there are no files cached in the clusters. As UEs download videos during the transient phase, video segments will accumulate in the distributed cache of each cluster. At the beginning of the steady phase, there will be many pieces in the distributed caches of the clusters that were cached during the transient phase. During each phase, each UE sends 2 download requests in total (i.e., each UE downloads 2 video files). A UE sends one request at a time, and after downloading the video file, it generates another request. Before each request, a UE waits for a random period using a Poisson distribution with mean of 10 s. At the end of each phase, we calculate the cell's aggregate data rate and the mean of the average data rate per user. The mean of the cell's aggregate data rate and the mean of the average data rate from all the iterations are calculated at the end of the simulations. The results show the mean values based on 50 simulation runs along with the margin of error for 95% confidence interval.

Table 2 Simulation setup

Parameter	Value
Cellular channel BW (MHz)	10
Cell range (m)	500
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
Carrier frequency (MHz)	900
File size range (MB)	1–100
Area configuration	Urban
Piece size (KB)	512
Number of files	500
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
UE receiver noise figure (dB)	9
D2D TX/RX height from ground (m)	1.5

The UEs generate requests to download video files from a list. The popularity of videos is generated according to a Zipf distribution to simulate a variable popularity of files, as it has been established that this is a good model for video files popularity [28]. Using this distribution, some files are requested more often than others are. The Zipf exponent, β, controls the relative popularity of the files. The size of the video files will be generated according to a logNormal distribution as in [29].

Unless stated otherwise, the number of UEs is 500, the Zipf exponent (β) is 1.5, and the number of requests made by a UE during each phase is 2.

In section 7.2, we consider the case of stationary UEs. In section 7.3, we consider the case of moving UEs.

7.2 Simulation results

Fig. 9 shows the Cell's aggregate data rate versus the number of UEs in the cell, for the SVD, CSVD, and DISCS algorithms, respectively, in the steady state phase. Up to one copy of each piece of a file is cached in a cluster in the case of CSVD and DISCS. As Fig. 9 shows, CSVD and DISCS provide significant improvement over SVD. The maximum aggregate rate achieved using the SVD is around 130 Mbps, while with the CSVD and DISCS, aggregate rates of 490 Mbps and 693 Mbps can be achieved, respectively, at 700 UEs. This significant improvement on the aggregate data rate is due to having more resources, i.e., the D2D channel with large bandwidth (60 MHz) available in each cluster, and used for D2D communication.

Furthermore, DISCS achieves significant improvement over CSVD. This because DISCS speeds up video caching and achieves better hit ratio as discussed in section 5, which improves the utilization of D2D channel. This is also because in CSVD, when a video file is cached in a cluster, it is always cached in one SM, while in DISCS, a cached file is distributed over many SMs in the cluster in the case of DTSMs. As such, when video files are transmitted from the distributed cache, multiple SMs will be sending pieces in parallel to the requesting UE in the case of DISCS. As such, the D2D channel will be further utilized and the aggregate data rate will increase.

Fig. 9 also shows that with CSVD and DISCS, the aggregate data rate increases with increasing the number of UEs in the network. Increasing the number of UEs increases the number of requests for video files and the number of SMs in each cluster. This increases the number of cached files in a cluster and the number of requests that would be satisfied from the cluster's cache. Hence, the D2D channel will be further utilized and the aggregate data rate will increase. With SVD, the aggregate data rate does not increase with the number of UEs in the cell. In SVD, each cell has fixed cellular resources (10 MHz channel is used here) and as the number of UEs increases, the utilization of the cellular channel will increase, until it is fully utilized. As such, we can say from Fig. 9 that with SVD, at 100 UEs, the cell is overloaded and the cellular channel is fully utilized.

Fig. 10 shows the average data rate per user versus the number of UEs in the cell for SVD, CSVD and DISCS, respectively (steady state phase). Up to one copy of each piece of a file is cached in a cluster in the case of CSVD and DISCS. As Fig. 10 shows, CSVD and DISCS provide important performance gains due to the transmission of video segments from the BS and SMs (distributed cache), as opposed to only transmitting video files from one source (the BS). This speeds up the transmission process and increases the average data rate. In the SVD, the average data rate decreases faster with increasing the number of UEs. For instance, the average data rate decreases from about 2 to 0.63 Mbps when the number of UEs increases from 100 to 300 UEs. This is because the fixed available frequency resources are divided over higher number of UEs. The improvement achieved by the CSVD and DISCS over the SVD increases when the number of UEs increases. This is because increasing the UEs also increases the available SMs and requested and cached files. Thus, more data will be transmitted from the cluster caches over D2D links rather than being sent from the BS over cellular links. As such, increasing the number of UEs will cause less decrease in the average data rate per user than in SVD.

Fig. 10 also shows that DISCS achieves significant improvement over CSVD. This because DISCS speeds up video caching and achieves better hit ratio as discussed in section 5, which increases the percentage of requests that are satisfied from the cluster's cache and 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). This

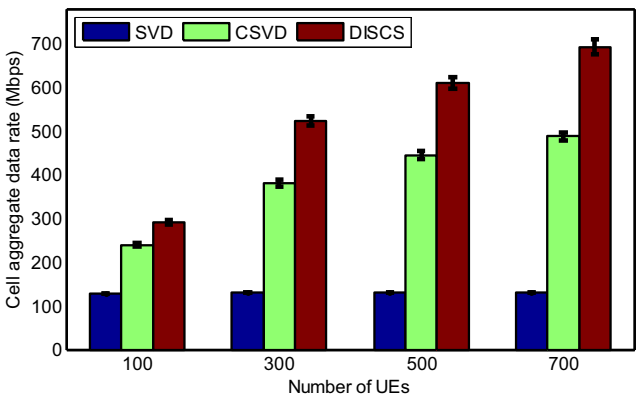

Fig. 9 Cell's aggregate data rate vs. number of UEs in the cell (steady state phase). 2 requests per user and Zipf exponent = 1.5

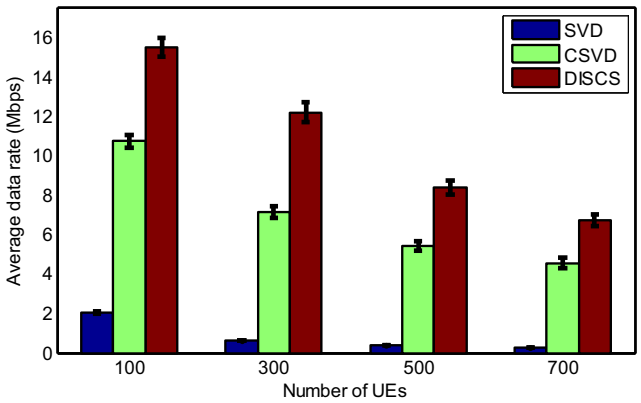

Fig. 10 Average data rate per user vs. number of UEs in the cell (steady state phase). 2 requests per user and Zipf exponent = 1.5

causes further parallelism in sending video files and better load balancing between SMs, which speeds up the transmission of video files and increases the average data rate.

As previously mentioned, the simulations were divided into two phases. The first phase is the transient phase that starts with no video files saved in the distributed caches of the clusters, and the pieces of the video files accumulate in the distributed caches during this phase as requested by UEs. At the beginning of the steady phase, there will by many files in the clusters that were cached during the transient phase. Fig. 11 and Fig. 12 show the aggregate data rates and average data rates, respectively, for the transient phase versus the number of UEs in the cell.

As expected, more improvement is achieved by CSVD and DISCS in the steady state phase. This is because in the steady state phase, there are more cached files in the clusters. Hence, more video files will be sent from the distributed cache, which increases the D2D channel utilization and speeds up the video transmission. However, good improvements are still achieved by both algorithms over SVD in the transient phase.

The impact of the Zipf distribution exponent on the performance of CSVD and DISCS is shown in Fig. 13. As can be seen, the average data rate increases by increasing the Zipf exponent. As the number of popular files increases, content reuse increases because more files will be cached and delivered later from the distributed cache rather than from the BS. This speeds up the transmission process and increases the average data rate. The increase in the average data rate will eventually slow down. This is because in our scenario, each UE requests only two video files.

Increasing the number of requests made by each UE further increases content reuse and improves the data rates. The effect of increasing the number of requests can be seen in Table 3, which shows the results for the average and aggregate data rates for CSVD and DISCS in the steady-state phase with 2 requests and 3 requests. As can be seen, increasing the number of requests increases the average and aggregate rates for both algorithms. This is because when the number of requests increases, the cached files will be further utilized by the later

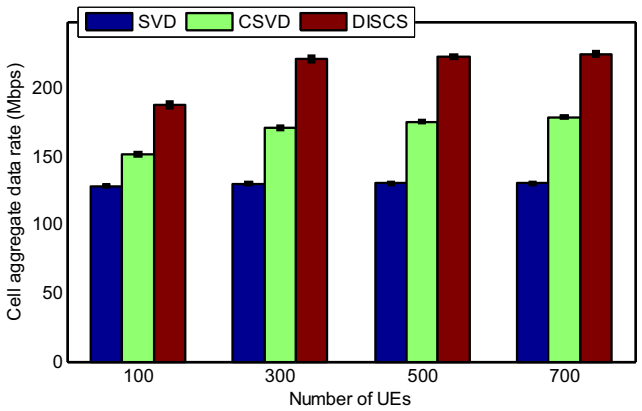

Fig. 11 Cell's aggregate data rate vs. number of UEs in the cell (transient phase). 2 requests per user and Zipf exponent = 1.5

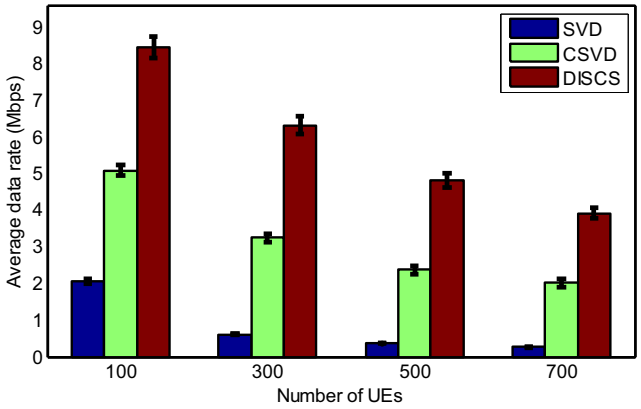

Fig. 12 Average data rate per user vs. number of UEs in the cell (transient phase). 2 requests per user and Zipf exponent = 1.5

requests. Hence, more requests will be satisfied from the cluster's cache, which improves the data rates.

To improve the average data rates achieved by the CSVD algorithm, more than one copy can be cached of each piece in the cluster. Of course, this would be on the expense of using more of the storage of the SMs. Fig. 14 shows that an improvement is achieved with the CSVD when the number of cached copies is increased from 1 to 3. This improvement is caused by having the popular files available in more SMs, which allows further parallelism when sending pieces, and allows more load balancing between SMs, which speeds up the transmission of video files to the requesting UEs.

After 3 cached copies, there is no improvement achieved by increasing the number of cached copies in the cluster, which means that 3 copies in the cluster are enough. This is explained in the following. As previously discussed, it is faster to send a video segment from the cluster's cache due the higher availability of frequency resources in the D2D channel when compared to the cellular channel. At the same time, it is beneficial for the whole cluster when more copies of the video segment are available in the cluster's cache as explained above. As such, a reasonable value for the number of cached

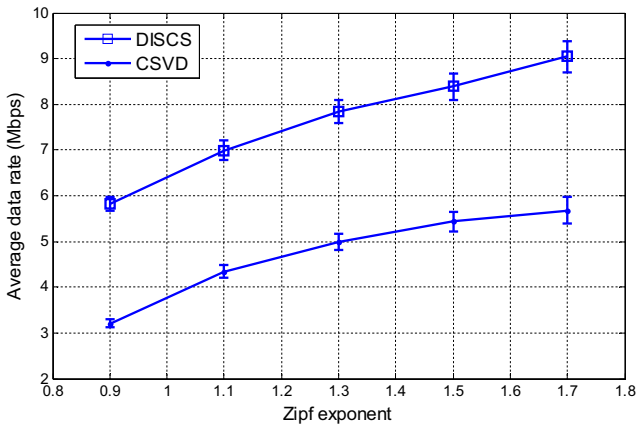

Fig. 13 Average data rate per user vs. Zipf exponent (steady state phase). 500 UEs in the cell and 2 requests per user

Table 3 Average and Aggregate data rates with 2 and 3 requests

	CSVD		DISCS	
	Average (Mbps)	Aggregate (Mbps)	Average (Mbps)	Aggregate (Mbps)
2 requests	5.43	445.60	8.39	611.24
3 requests	6.05	490.22	9.58	665.29

copies should be selected to balance between two factors. The first factor is sending more segments from the cluster's cache to increase the transmission rate of these segments and increase the offloading of video segments from the cluster's cache to make more cellular resources available for segments that need to be sent from the BS. The second factor is making more segments available in the cluster's cache to increase parallelism and load balancing among SMs in the cluster. With CSVD, the BS sends a requested segment to an SM directly (STSM) over a cellular link if the number of copies in the cluster is less than the intended number of cached copies. Otherwise, the piece will be sent to the SM from the distributed cache over D2D link. Hence, increasing the number of cached copies will increase the average data rate up to a certain point. After some point, increasing the number of cached copies beyond a certain value might cause a slight reduction in the average data rate, as more copies will need to be sent to SMs from the BS over cellular links even when enough copies are already cached in the cluster. This explains why the average data rate for CSVD with 5 cached copies is slightly less than that of CSVD with 3 and 4 cached copies.

One can notice that although the algorithms provide significant improvements, they introduce some overhead (Handshake message, Assistance Request message, etc.) needed to implement D2D distribution of video content. In the following we discuss the overhead of the proposed algorithms. The efficiency is measured as the ratio of transmitted data bits to the transmitted bits (data bits plus transmitted bits in the Handshake message, Assistance Request message, etc.). The overhead equals (1 - efficiency).

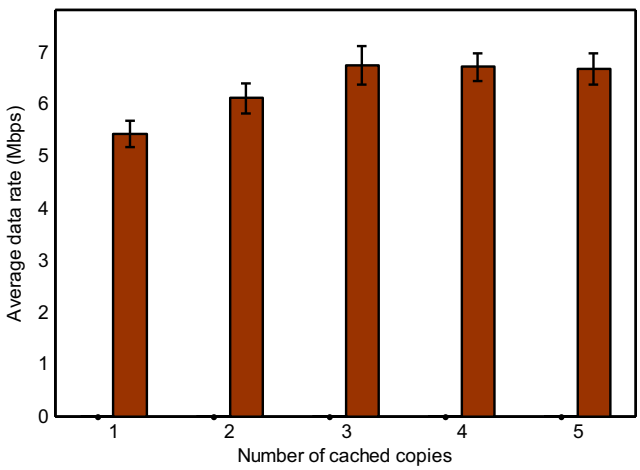

Fig. 14 Average data rate per user vs. number of cached copies (steady state phase) for CSVD. 500UEs in the cell and Zipf exponent = 1.5

With 512 KB piece message, the efficiency of CSVD in the steady state phase is 0.999936 (overhead of 0.000064) and the efficiency of DISCS is 0.999934 (overhead of 0.000066). The results show that the proposed algorithms have high efficiency (low overhead). The overhead for DISCS is slightly higher than that for CSVD. This is expected as more Assistance Request messages are sent by DISCS (due to the DTSMs case).

7.3 The impact of UE mobility

In this section, we analyze the effect of UE mobility. Furthermore, we show how significant performance improvement can still be achieved in the case of UE mobility by employing the proposed improved operation, i.e., requesting all UEs to cache received content. As previously mentioned, this also increases the fairness among users because it increases the chance of a requesting UE to be a helper/provider in the future.

The probability of movement, P_m, is a parameter that controls the probability that a UE in the simulation is moving. When P_m is 1, all the UEs in the simulation will be moving, and when P_m is 0, all the UEs in the simulation will be stationary. A moving UE will have a random destination in the simulation area. A moving UE walks with a speed of 1.34 m/s which is the average pedestrian speed. When a UE reaches its destination, it generates a new destination and starts moving towards the new destination and so on.

In addition to considering UE mobility, we updated the channel model to include fast fading in addition to path loss and shadowing. Rayleigh fading channel model was considered, where the employed Rayleigh fast fading model considers the speed of the UE, the subcarrier frequency, and the number of paths.

Fig. 15 shows the average data rate per user versus P_m, for CSVD (steady state phase). As we can see, the average data rate decreases by increasing P_m. When $P_m = 0$, all the UEs in the cell are stationary, which means that UEs with cached content will be always available as SMs in the central area. However, when P_m increases, some of the UEs will be moving. For instance, at $P_m = 0.5$, half of the UEs in the cell are moving, which means that about half of the SMs in the clusters will be moving as well. As some of these moving SMs (especially the ones with cached popular files) might leave the central area and become non-SMs, content reuse will be

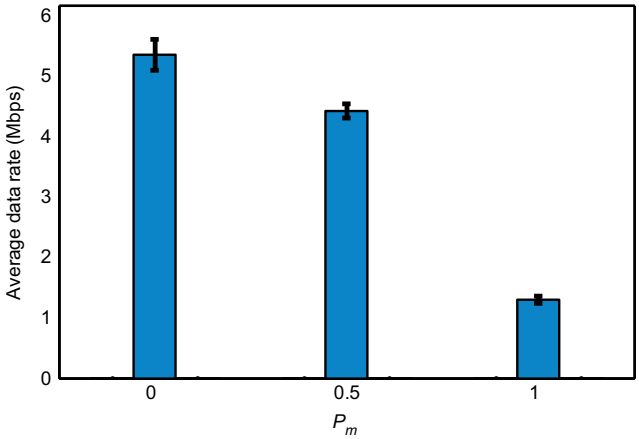

Fig. 15 Average data rate per user vs. P_m for CSVD (steady state phase). 500 UEs in the cell and Zipf exponent = 1.5

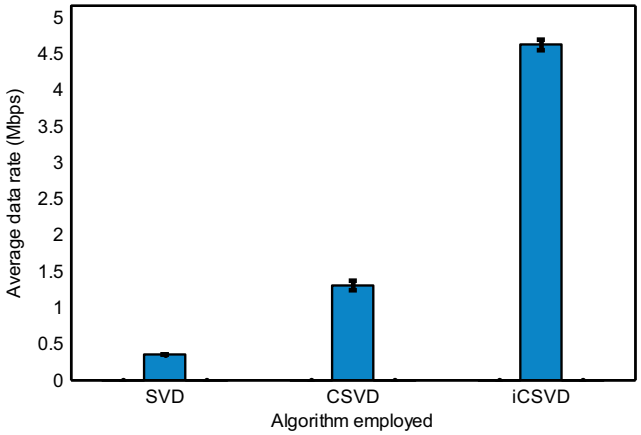

Fig. 16 Average data rate per user vs. the employed algorithm (steady state phase). 500UEs in the cell, Zipf exponent = 1.5, and P_m = 1

reduced. This is because such cached content will not be available for transmission. This reduces the number of video segments transmitted from the clusters' caches and hence reduces the average data rate.

At P_m = 1, all the UEs are moving, which means all SMs are moving. By the time the steady state phase starts, higher number of SMs would be out of the central area, and hence, their cached content will not be available for transmission. As such, this further reduces content reuse, and decreases the average data rate. From the above results, one can see that the movement of UEs has an impact on the performance of proposed algorithms with their conventional operation (especially when all UEs are moving). However, we will see next how the improved operation overcomes this challenge.

In section 6, we discussed an improved operation for the proposed algorithms to overcome the performance degradation that might be caused by the mobility of UEs. With the improved operation, instead of setting the save bit to 1 only in the STSM and DTSMs cases, the save bit will always be set to 1. This means that every UE will be asked to cache received content. This is because when UEs are moving, any UE could be an SM (after entering the central area) and hence it would be very beneficial (for the cluster) for any UE that becomes an SM to have cached content. This also increases the fairness of the algorithms, since in this case UEs that received content from the clusters' caches could become SMs and provide others with video content later.

Fig. 16 shows the average data rate per user for SVD, CSVD, and improved CSVD (iCSVD), with P_m = 1 (all UEs are moving). iCSVD is basically the CSVD with the improved operation where received video segments are always cached. As can be seen in the figure, although UE mobility has reduced the average data rate achieved by CSVD, CSVD still provides significant improvement over SVD. However, the proposed operation with iCSVD significantly improves the achieved average data rate and overcomes the impact of UE mobility, even at P_m = 1. This is because with the iCSVD, all UEs are caching received video segments. In this case, even if some SMs leave the central area and become non-SMs, other UEs with cached content enter the central area and become SMs with cached content, which increases content reuse and improves the achieved average data rates, when compared to CSVD.

8 Conclusion

In this paper, we present two algorithms for improving the throughput of video transmission in cellular networks. The algorithms are called *Cached and Segmented Video Download* (CSVD), and *DIStributed, Cached, and Segmented video download* (DISCS). The algorithms send segments of video files to selected User Equipments (UEs) in the cellular network, to cache and forward them to requesting UEs using Device-to-Device (D2D) communication.

We analytically model video caching with CSVD and DISCS and derive closed-form expressions for the hit ratio of video caching with both algorithms. We also study the performance of both algorithms in terms of the aggregate and average data rates. Our results show that CSVD and DISCS achieve significant performance improvements in terms of the aggregate and average data rates. Furthermore, the results show that DISCS achieves more improvement than CSVD. Results also show that the proposed algorithms are very efficient and have very small overhead.

Moreover, we study the impact of UE mobility on performance. We propose an improved operation for CSVD and DISCS under UE movement and show that the improved operation helps overcome the impact of UE movement.

In future work, we will study employing CSVD and DISCS in the context of video streaming over cellular networks, and evaluate the improvement achieved by these algorithms in terms of video streaming quality of experience.

Acknowledgements The authors would like to thank Gary Boudreau and Ronald Casselman from Ericsson Canada for their valuable assistance during this work.

References

1. ICT Data and Statistics Division, Telecommunication Development Bureau, ITU (2016) ICT facts and figures. Tech. rep. https://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2015.pdf
2. Cisco (2017) Cisco visual networking index: global mobile data traffic forecast update. Tech. rep. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html
3. Parkvall S, Astely D (2006) The evolution of LTE towards IMT-advanced. J Commun 4(3):146–154
4. Asadi A, Wang Q, Mancuso V (2014) A survey on device-to-device communication in cellular networks. IEEE Communications Surveys and Tutorials 16(4):1801–1819
5. Kaufman B, Aazhang B (2008) Cellular networks with an overlaid device to device network. In: Proceedings of Asilomar Conference on Signals, Systems and Computers, pp 1537–1541
6. Doppler K, Rinne MP, Wijting C (2009) Device-to-device communication as an underlay to LTE-advanced networks. IEEE Commun Mag 47(12):42–49
7. Doppler K, Rinne MP, Janis P (2009) Device-to-device communications: functional prospects for LTE-Advanced networks. In: Proceedings of IEEE ICC Workshops, pp 1–6
8. Al-Habashna A, Wainer G, Boudreau G, Casselman R (2015) Improving wireless video transmission in cellular networks using D2D communication. Provisional patent P47111
9. Al-Habashna A, Wainer G, Boudreau G, Casselman R (2016) Cached and segmented video download for wireless video transmission. In: proceedings of the 49th ANSS, pp. 18–25
10. Al-Habashna A, Wainer G, Boudreau G, Casselman R (2016) Distributed cached and segmented video download for video transmission in cellular networks. In SPECTS'16, pp. 473–480
11. Golrezaei N, Mansourifard P, Molisch AF, Dimakis AG (2014) Base-station assisted device-to-device communications for high-throughput wireless video networks. IEEE Trans Wirel Commun 13(7):3665–3676
12. Zeigler B, Praehofer H, Kim T (2009) Theory of modeling and simulation. Academic Press, San Diego
13. Wainer G (2009) Discrete-event modeling and simulation: a practitioner's approach. CRC/Taylor & Francis Group, Boca Raton
14. Chandrasekhar V, Andrews J, Gatherer A (2008) Femtocell networks: a survey. IEEE Commun Mag 46(9):59–67
15. Ahlehagh H, Dey S (2012) Hierarchical video caching in wireless cloud: Approaches and algorithms. In: Proceedings of IEEE ICC, pp 7082–7087
16. Chellouche SA, Négru D, Chen Y (2012) Home-box-assisted content delivery network for Internet video-on-demand services. In: Proceedings of IEEE ISCC, pp 544–550
17. Maddah-Ali MA, Niesen U (2015) Decentralized caching attains orderoptimal memory-rate tradeoff. IEEE/ACM Trans Networking 23(4):1029–1040
18. Zhao J, Zhang P, Cao G (2010) Cooperative caching in wireless P2P networks: design, implementation, and evaluation. IEEE Trans on parallel and distributed systems 21(2):229–241
19. Zhang Y, Song L, Saad W, Dawy Z, Han Z (2015) Contract-based incentive mechanisms for device-to-device communications in cellular networks. IEEE JSAC 33(10):1–12
20. Gao L, Huang J, Chen Y, Shou B (2014) Cooperative spectrum sharing: a contract-based approach. IEEE Trans Mob Comput 13(1):174–187
21. Duan L, Kubo T, Sugiyama K, Huang J, Hasegawa T, Walrand J (2014) Motivating smartphone collaboration in data acquisition and distributed computing. IEEE Trans Mob Comput 13(10):2320–2333
22. Kang HJ, Park KY, Cho K (2014) Mobile caching policies for device-to-device (D2D) content delivery networking. In: INFOCOM WKSHPS, pp 299–304
23. Keller L, Le A, Cici B, Seferoglu H, Fragouli C, Markopoulou A (2012) MicroCast: cooperative video streaming on smartphones. In: proceedings of MobiSys, pp 57–70
24. Eittenberger P, Herbst M, Krieger U (2012) RapidStream: P2P streaming on android. In: proceedings of IEEE International Packet Video Workshop, pp 125–130
25. Siris V, Dimopoulos D (2015) Multi-source mobile video streaming with proactive caching and D2D communication. In: IEEE WoWMoM, pp 1–6
26. [3GPP] TR36.942 (2015) Evolved universal terrestrial radio access; RF system scenarios. Tech. rep
27. Al-Hourani A, Chandrasekharan S, Kandeepan S (2014) Path loss study for millimeter wave device-to-device communications in urban environment. In: Proceedings of IEEE ICC, pp 102–107
28. Cha M, Kwak H, Rodriguez P, Ahn Y, Moon S (2007) I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system. In Proceedings of ACM SIGCOMM conference on Internet measurement, pp 1–14
29. Ahsan S, Singh V, Ott J (2014) Characterizing internet video for large-scale active measurements. arXiv preprint arXiv:1408.5777v1