DEVS-BASED MODELING OF CACHED AND SEGMENTED VIDEO DOWNLOAD ALGORITHMS IN LTE-A CELLULAR NETWORKS

Ala'a Al-Habashna
Dept. of Systems and Computer Engineering
Carleton University
Ottawa, ON, Canada
alaaalhabashna@sce.carleton.ca

Gabriel Wainer
Dept. of Systems and Computer Engineering
Carleton University
Ottawa, ON, Canada
gwainer@sce.carleton.ca

ABSTRACT

Cellular networks have witnessed increasing demands for higher data rates in the recent years. Satisfying these demands presents a challenge for cellular network operators. Video traffic plays a major role in this, as it accounted for more than half of the data traffic on cellular networks recently. Device-to-Device (D2D) communication, introduced by the Long Term Evolution-Advanced (LTE-A) standard, allows direct communication between User Equipments (UE) in the network. We proposed cached and segmented video download algorithms that employ D2D communication to improve the throughput of video transmission over LTE-A cellular networks. Here, we present the Modeling and Simulation (M&S) of an LTE-A network that implements the proposed algorithms. We used the Discrete Event System Specification (DEVS) formalism to model the network. Simulation results show that significant improvements are achieved by the proposed algorithms in terms of the average and aggregate data rates.

Keywords: M&S, DEVS, Cellular networks, LTE-A, D2D communication.

1 INTRODUCTION

Improving the data rates of cellular networks to meet the user demands has become one of the main challenges for cellular network operators. Recent studies have showed that wireless video is the major reason for the increase in the data traffic over cellular networks and the increasing demand for higher data rates. According to (Cisco 2016), video traffic accounted for 55% of the total mobile data traffic. Furthermore, it is expected that video will account for 75% of the total mobile data traffic by 2020. As such, it is necessary to develop new techniques that help serving the video traffic that is taking up a major portion of the overall traffic.

Device-to-Device (D2D) communication is a new technique introduced by the Long Term Evolution-Advanced standard (LTE-A) to improve data transmission in cellular networks. D2D transmission provides a direct mean of communication between two User Equipments (UEs) without going through the Base-Stations (BS) or the core network.

In recent years, we have seen various efforts combining current standards with D2D communications to improve performance (Asadi, Wang and Mancuso 2014; Kaufman and Aazhang 2008; Doppler et al. 2009). We proposed two algorithms based on the architecture in (Golrezaei et al. 2014), to exploit D2D communication for the transmission of video in cellular networks (Al-Habashna et al. 2015; Al-Habashna et al. 2016). The first algorithm is called Cached and Segmented Video Download (CSVD). The second algorithm is called DIStributed, Cached, and Segmented video download (DISCS). The proposed algorithms use selected UEs in the cell to cache segments of video files. If cached video segments are requested by UEs in the cell, they will be sent to the requesting devices from the caching UEs over D2D links.

SpringSim-CNS 2017, April 23-26, Virginia Beach, VA, USA

Here we will discuss the results of a Modeling and Simulation (M&S) study of an LTE-A network that implements the proposed algorithms. Although various network simulators exist (i.e., NS2, NS3, OPNET), recent studies shown that these simulators have some weaknesses (Koksal 2014; Kamoltham, Nakorn and Rojviboonchai 2012; Rajankumar, Nimisha and Kamboj 2014; Xian, Shi and Huan 2008), and that it is challenging to integrate those simulators with other models (for example, models of traffic of pedestrians holding UE). As such, we also want to use the Discrete Event System Specification (DEVS) formalism in order to avoid these problems. The objective is not to compare different M&S techniques, but to introduce a flexible technique for M&S of cellular networks. Hence, we built a suite of models with the DEVS formalism (Zeigler, Praehofer and Kim 2000) and the CD++ DEVS toolkit (Wainer 2009) to evaluate the performance of the proposed algorithms. The hierarchical and modular nature of DEVS was useful for modeling and simulating cellular networks. Our model was built using different submodels, in which each one implements a different component of the wireless network. Each of the submodels could be tested and verified separately, and later integrated into the whole model, and reused. This made design, implementation, and testing easier. System level simulations were performed to evaluate the performance of the proposed algorithms with different parameters. Simulation results show that the proposed algorithms achieve a significant improvement over the conventional transmission approach where D2D communication is not employed.

The paper is structured as follows: background on this work is provided in Section 2. The proposed algorithms are presented in Section 3. DEVS-based Modeling of an LTE-A network that implements the algorithms is discussed in Section 4. The implementation of our DEVS model with the CD++ toolkit is provided in Section 5. Simulation case studies are presented in Section 6.

2 BACKGROUND

As mentioned in the previous section, new techniques are needed to increase the data rates in cellular networks to handle the increasing user demands, and in particular, serve video traffic. The scarcity of the radio spectrum is a major cause for this challenge. As most of the licensed frequency bands are allocated, it is hard to give enough resources for users to enjoy high data rates. New techniques have emerged to increase the efficiency of radio spectrum exploitation such as cognitive radio (Wang and Liu 2011). However, the radio spectrum is still scarce, and novel techniques are needed to improve performance.

As video traffic is accounting for more than half the data traffic over cellular networks, much research has focused on improving its performance. In (Ahlehagh and Dey 2012), Caching popular video files at the BSs or at the mobile switches was proposed in order to reduce the traffic on the backhaul network and speed up the transmission of video traffic to the end user. However, such technique does not improve the transmission between UEs and BS over the radio access network.

Simultaneous coded-multicasting was proposed in (Maddah-Ali and Niesen 2014) to improve video transmission in cellular networks. In (Chellouche et al. 2012), using a virtual home-box layer is proposed for efficient and scalable internet on demand video streaming. These approaches use the client-server model, which does not scale well for video content distribution on a cellular network with high number of users and limited resources. As such, there have been efforts using Peer-to-Peer (P2P) communication.

D2D communication provides an efficient way to implement P2P between the UEs in the network. Much research has been conducted on exploiting D2D communication in cellular networks to improve their performance (Asadi, Wang and Mancuso 2014; Kaufman and Aazhang 2008; Doppler et al. 2009). In terms of improving video transmission, (Kang et al. 2014) proposed an architecture where devices that are dedicated as caching servers are used to provide UEs with cached contents on demand, and content delivery and D2D communication could be used for transmission from these devices to UEs. Although 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.

In (Al-Habashna et al. 2015, Al-Habashna et al. 2016), we proposed two algorithms based on the architecture in (Golrezaei et al. 2014). The algorithms exploit video files caching and D2D

communication to improve the throughput of video transmission and overcome the problem of rapidly increasing wireless video traffic. Instead of caching complete files, the files are split into pieces and multiple copies of a file can be cached at multiple SMs. 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 the selected UEs in each cluster are used for caching to reduce inter-cluster interference. A complete detailed protocol was defined, including the messages needed and a complete definition of the protocol, which is provided in the next Section.

Here, we focus on the M&S of the proposed algorithms. M&S is a widely used approach for evaluation of new standards and protocols. Testing the effectiveness of proposed methods and investigating different aspects of their operation is easier when using M&S. One can build a model for a network, simulate it, and test and evaluate the proposed approaches under different test configurations and scenarios. There is varied research on M&S for performance evaluation (Obaidat, Zarai and Nicopolitidis 2015; Ros, Martinez and Ruiz 2014; Qiu et al. 2009). For example (Qiu et al. 2009) considers testing as one of the recurring problems in LTE networks. An NS2-based LTE/SAE model was built for testing different parameters of the network. OPNET was used in (Koc et al. 2014) to investigate the coverage of LTE networks. Here, we want to explore a flexible technique for M&S of cellular networks based on the definition of DEVS-based models for an LTE-A network. DEVS has been proposed as a sound formal framework for modeling generic dynamic systems and includes hierarchical, modular and component-oriented structure and 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. The CD++ toolkit (Wainer 2009) was used to implement our LTE-A network DEVS model. CD++ is an open-source simulation software written in C++ implementing the DEVS abstract simulation technique. We present, as simulation case studies, results for the cell aggregate data rate and average data rate achieved with the proposed algorithms.

3 THE CSVD AND DISCS ALGORITHMS

CSVD and DISCS focus on scenarios where there is limited area with high density of users. In (Al-Habashna et al. 2016) we provide many examples of such scenarios (i.e., live concerts with detailed video feeds of the arena). Consider one cell in a cellular network, in which the BS is in the middle of the coverage area, as seen in Figure 1. The BS divides the cell into clusters. The BS assigns UEs to clusters based on their locations, and it selects the UEs in the central area of each cluster as Storage Members (SMs) of that cluster, as in Figure 1. Only UEs in the middle of the cluster are chosen as SMs, in order to prevent inter-cluster interference when the SMs transmit to other UEs in the same cluster using D2D.

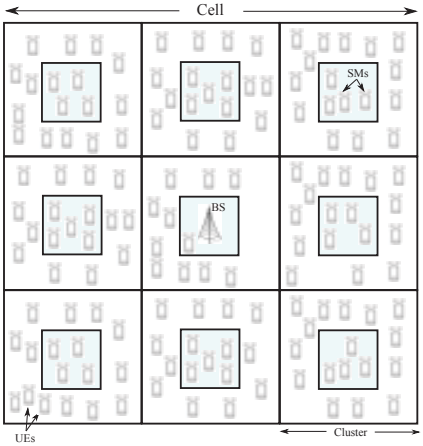

Figure 1: Cell after clustering.

After completing the clustering phase, the transmission phase can begin. 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:

- Send With Assistance (SWA): if the 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 a 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 a SM, the BS will send the file to that SM over a cellular link, and ask the SM to cache the video file. These files will be available for UEs in the cluster later.
- Distribute To SMs (DTSMs): this new case proposed by DISCS is as follows. If a requested video is popular and it is not available in the distributed cache of the cluster, the BS will distribute the pieces among the SMs. 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.

CSVD employs 3 of the above cases (does not employ DTSMs), while DISCS employ all of the above cases. Due to lack of space, we only present the message sequence of the DTSMs case. For further details on the other cases, the reader is referred to (Al-Habashna et al. 2016).

3.1 Distribute to SMs case

As mentioned above, this is the main contribution of the DISCS algorithm. In this case, a popular file (for instance, one requested n times) is requested by a non-SM 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 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 (file size, number of pieces, etc.). 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 *Piece* messages to 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 D2D link.
7. When an SM finishes sending piece(s), it will send an *SM_Finished* message to the BS, acknowledging 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 further helps accumulating 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.

3.2 The SVD algorithm

We call our implementation of the conventional download process the Segmented Video Download (SVD), as video files are sent in pieces. In SVD, we do not use file caching or D2D communications. Instead, the files will be always sent as in STUE.

4 DEVS-BASED MODELING OF THE LTE-A NETWORK

Figure 2 shows the structure of the LTE-A network DEVS model. The top level is called the Cell coupled model, that contains the BS, Transmission Medium, and many UE coupled models. It also contains the Cell Manager and Log Manager atomic models.

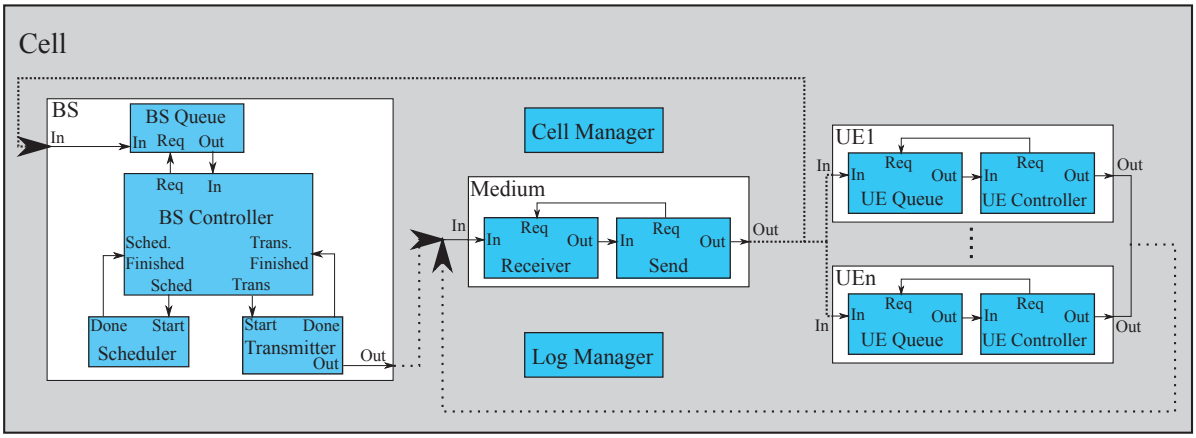

Figure 2: DEVS model of the cellular network.

The BS coupled model corresponds to the BS in the cell. It contains four atomic models: BS Queue, BS Controller, Scheduler, and Transmitter. The BS Queue atomic model is where received messages are buffered. The BS queue also checks the destination address of arriving messages. It buffers a message with destination address that matches that of the BS; otherwise, it ignores the message. The BS Controller processes received messages and operates as per the algorithms above (e.g., CSVD). 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 UE coupled models represent the UEs in the cell. A UE coupled model contains two atomic models: UE Queue, and UE Controller. The UE Queue is where received messages are buffered. The UE Controller is where the UE part of the algorithm is implemented. The Medium model receives a message sent from the BS or any UE and broadcasts it to the other receivers (BS/UEs) in the cell. Receivers recognize their intended messages by the destination address filed in the message. As mentioned above, this is done at the BS Queue and UE Queue models. 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 (e.g., path loss and received signal power). Path loss and shadowing is considered here. The urban macro propagation model (3GPP 2015) was used for cellular links and the D2D channel model at 24 GHz, defined in (Al-Hourani et al., 2014), was used for D2D communication. The Log Manager logs simulation events and record statistics during the simulations.

As previously mentioned, each atomic component represents a behavior of a part of the system. In the following, we discuss the behavior of the BS Controller and the UE Controller atomic models, as they are the most important components. The BS manages all the data transmission in the cell, the sessions with all the UEs, and allocates the radio frequency resources to the UEs. The BS Controller checks its queue for messages from UEs. The BS will process received messages and update the state of the corresponding download session. For instance, if the received message is a new download request, the BS will create a new *Session* object to that node. If the received message is *SM_Finished* message, the BS will update the statistics of that session (number of transmitted pieces, transmitted bits, etc.).

The BS Controller model has five states; *Inital, Check_Queue, Rcv_Msg, Schedule, and Send.* At the beginning of each iteration, the model is in the *Initial* state. Then it goes to the *Check Queue* state, during which, the BS sends requests to the queue asking for the messages from UEs. The queue will send the next message in line. The BS has a *Session* object for each active UE, that maintains the status and statistics of that download session. When the BS model receives a message, it goes to *Rcv_Msg* state. During this state, the BS processes the message, updates the status/statistics of the corresponding session, and then sends another request to the Queue. When no more messages are available to process, the queue sends *EMPTY_QUEUE* message to the *BS controller*. When the *BS controller* receives *EMPTY_QUEUE* message, it goes to the *Schedule* State. In this state, the BS sends a message to the *Scheduler*. The *Scheduler* goes through the sessions and schedules the messages to be sent during the next TTL. When the scheduler finishes scheduling (i.e., after allocating all the RBs in TTI or scheduling all the messages that should be transmitted in the next TTI), it sends a *SCHED_FINISHED* message to the *BS Controller* to indicate that scheduling is finished. When The *BS Controller* receives this message, it goes to the *Send* state. In this state, the model sends a message to the *Transmitter* model to start transmission. When the transmission is done, the *Transmitter* model sends a *TRANS_FINISHED* message to the *BS controller* to indicate that transmission is finished. Upon receipt of this message, the BS goes back to the *Check_Queue* state.

Figure 3 shows a state diagram representing a DEVS atomic of the *UE Controller*. Each UE starts at the *Idle* state as shown in Figure 3. Each time the UE is in the *Idle* state, it will wait for random Idle time. After the Idle time is expired, it will generate a download request for a certain video file.

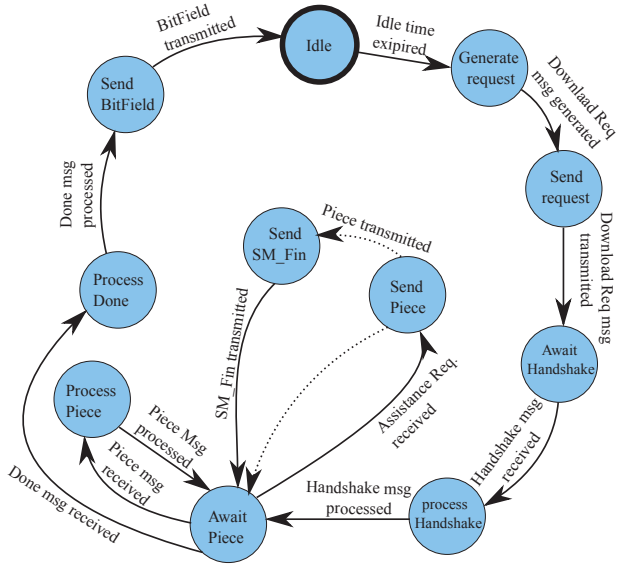

Figure 3: UE Controller state diagram.

After generating a request, a UE will wait for a *Handshake* message from the BS. After receiving the *Handshake* message, the UE will set the parameters for this session (file size, number of pieces, piece size, etc.). The UE will then wait for pieces of the video file. After receiving all the pieces, and a *Done* message, the UE will send a *Bitfield* message. When the file download is complete, the UE goes back to *Idle* state, and tries to generate another download request as described above. When a UE receives a *Piece* message while it is in *Await Piece* state, it updates the statistics for this session, and goes back to *Await Piece* state. If the UE is a SM and the *Save* bit in the *Piece* message is set, the UE will save the *Piece* message. The Save bit is always set for *Piece* messages in STSM and DTSMs cases (As described in section 3). When a SM UE is in the *Await Piece* state, and it receives an *Assistance Request* message, it will go to *Send Piece* state where it sends a *Piece* message to the requesting UE. Then it could go to back to *Await Piece* state, or could go to Send *SM_Finished* message to acknowledge the transmission of Piece(s), before going back to *Await Piece* state.

5 IMPLEMENTATION OF THE DEVS-BASED MODEL

We used the CD++ toolkit to implement our LTE-A network DEVS model, which is an open-source simulation software written in C++ and implements the DEVS abstract simulation technique. The simulation engine tool of CD++ is built as a class hierarchy. With CD++, atomic models are developed using C++ programming language and can be incorporated into the class hierarchy. Figure 4 shows a simplified UML diagram that shows the main classes of our model. As can be seen from Figure 4, the BS Controller atomic model is implemented with the BS class. The attributes of this class are used to contain the specifications of a BS, such as the location, maximum transmission power, antenna gain, etc. The class also has functions necessary to implement the functionalities performed by the BS Controller, as per the algorithms above.

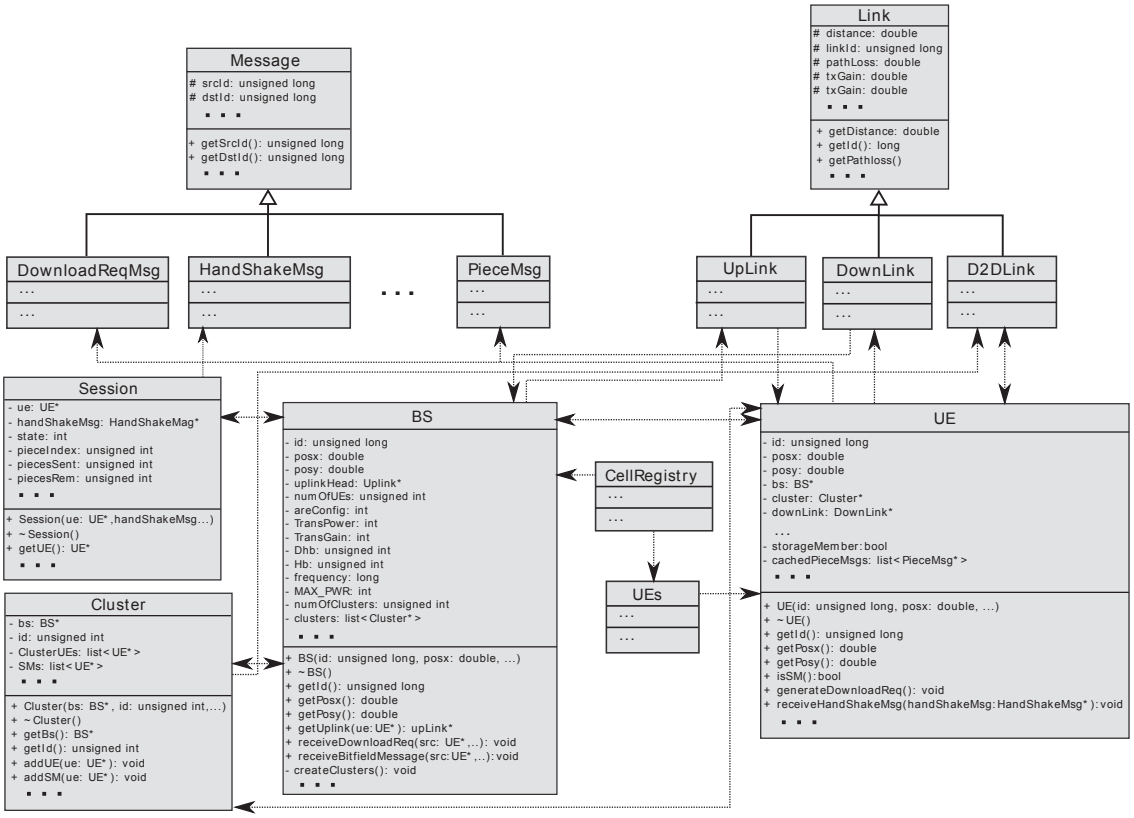

Figure 4: Simplified UML diagram of the DEVS model.

Similarly, the UE class is used to represent the UE Controller atomic model. It has many attributes to contain the parameters of the UE, such as the (location, transmission power, antenna gain, etc.) and functions to implement the functionalities of the UE.

Figure 4 shows that 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. The link base class is used to represent transmission links. It has many attributes to represent the transmission link parameters such as distance, path loss, received power, etc. The derived classes DownLink, UpLink, and D2DLink, are used to implement the DL, UL, and D2D links, respectively. These derived classes contain certain parameters and functions to compute the metrics (e.g., path loss) of the corresponding link type. Due to lack of space, many other classes are not shown here such as classes to implement the Queue, Transmitter, and Scheduler atomic models.

With CD++, Coupled models can be created using a language built in the simulation engine. Figure 5 shows a code snippet from the DEVS coupled model file. As can be seen from Figure 5, the coupled

model file is used to create the coupled model and maintain its hierarchical structure. For each coupled model, the components are listed and the links between these components are defined. For instance, the components for the top (Cell) coupled model are listed as: logManager, BS, UE, etc. A couple of defined links are shown. For instance, an out from the Medium coupled model is used as an input to the BS coupled model (as in Figure 2). The parameters for the atomic models are also listed in the file. For instance, the code shows some parameters for BS Controller atomic model.

```
[top]
components : logManager@LogManager  BS  Medium  UE1 ... UE100 cellManager @CellManager
...
Link : out@BS in@Medium
Link : out@Medium  in@BS
Link : out@Medium  in@UE1
Link : out@UE1 in@Medium
...
[BS]
components    :    BS1Queue@BSQueue    BS1Controller@BSController    scheduler@Scheduler
transmitter@Transmitter
[BS1Controller]
numberOfUEs : 100
BSId : 1
posX : 0
posY : 0
...
[UE1]
components : ue1Queue@UEQueue ue1Controller@UEController
...
```

Figure 5: Code snippet from the DEVS coupled model file.

6 SIMULATION CASE STUDIES

In this Section, we present simulation case studies for our LTE-A DEVS model. We used the developed model to evaluate the performance of the proposed CSVD and DISCS algorithms in terms of the cell's aggregate data rate and average data rate. We also evaluate the improvement achieved by these algorithms, when compared to conventional video download without video caching nor D2D communication (SVD). In the simulations, we consider a single LTE-A cell. The urban macro propagation model (3GPP 2015) was used for cellular links with a DL operating carrier frequency of 900 MHz, and a transmission bandwidth of 10 MHz. For D2D transmission, we used the D2D channel model at 24 GHz, defined in (Al-Hourani, Chandrasekharan and Kandeepan 2014). For further details on these models, the reader is referred to (Al-Habashna 2016). The simulation setup is shown in Table 1.

The UEs are uniformly distributed throughout the cell at the beginning of each simulation run, and the cell is organized into nine clusters. Each UE in the cell are identified by the BS as a SM or non-SM as in Figure 1. The central area of each cluster forms 1/4 of the total area of the cluster. As such, about 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. When the transient phase starts, the distributed cache in the clusters are empty. Video segments are cached during the transient phase as UEs download video segments. Each UE performs two requests during the transient phase. At the beginning of the steady phase, the distributed cache of the clusters contains many cached video file pieces. These were cached during the transient phase. Each UE download two video files during each phase. A UE sends one request at a time, and after downloading the video file completely, it generates another request. Before each request, a UE waits for a random period using a Poisson distribution with mean of 10 seconds. 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 40 simulation runs along with the margin of error for 95% confidence interval.

Table 1: 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 (Cha et al. 2014). Using this distribution, some files are requested more often than others are. The Zipf exponent, β, controls the relative popularity of the files. Many studies analyzed the distribution of the video files size and workload of YouTube, as it the most popular video sharing service (Abhari and Soraya 2010; Ahasan et al. 2014). These studies use heavy-tailed distributions to model the sizes of YouTube video files. Here, the sizes of the video files are generated according to a logNormal distribution as in (Ahasan et al. 2014). Unless stated otherwise, the number of UEs is 500, the Zipf exponent is 1.5, and the number of requests made by a UE during each phase is 2.

Figure 6 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.

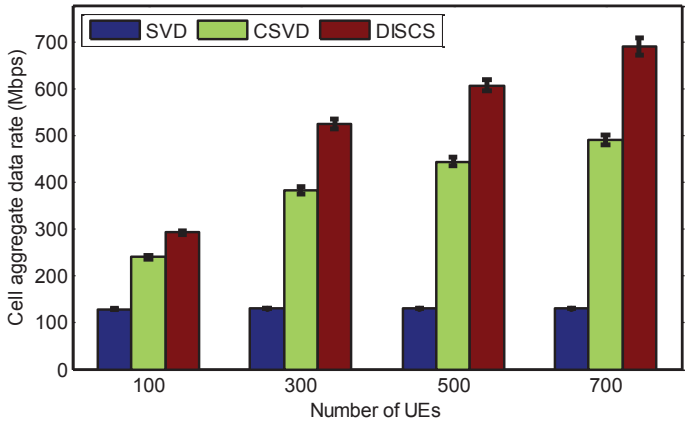

Figure 6: Cell's aggregate data rate vs. number of UEs (steady state phase). 2 requests per user, β = 1.5.

Up to one copy of each piece of a file is cached in a cluster in the case of CSVD and DISCS. As Figure 6 shows, CSVD and DISCS provide significant improvement over the 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 690 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 is 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.

Figure 6 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 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 (a 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 Figure 6 that with SVD, at 100 UEs, the cell is overloaded and the cellular channel is fully utilized.

Figure 7 shows the average data rate per user versus the number of UEs in the network 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 Figure 7 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 the SVD.

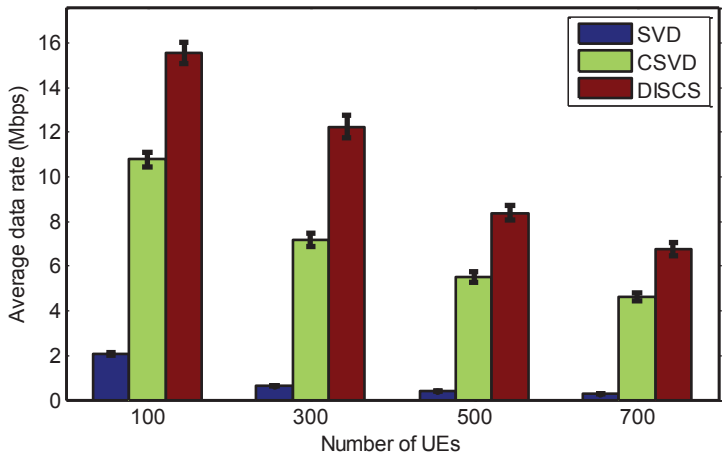

Figure 7: Average data rate per user versus the number of UEs (steady state phase). 2 requests per user, $\beta = 1.5$.

Figure 7 also shows that DISCS achieves significant improvement 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 files and better load balancing between SMs, which speeds up the transmission of video files and increases the average data rate.

7 CONCLUSION

In this paper, we used the Discrete Event System Specifications to build a model for an LTE-A network that implements two algorithms that we propose to improve video transmission in cellular networks. The algorithms are called Cached Segmented Video Download (CSVD), and DIStributed, Cached, and Segmented video download (DISCS). We present the different aspects for modeling the LTE-A network with DEVS. Furthermore, we present our implementation of the model with the CD++ toolkit. We used the developed model to run various simulations in order to evaluate the performance of the proposed CSVD and DISCS algorithms in terms of the cell's aggregate data rate and average data rate. We also evaluate the improvement achieved by these algorithms, when compared to conventional video download without video caching nor D2D communication (SVD). We present some simulation case studies and results here. Simulation results show that significant improvement can be achieved by the proposed algorithms when compared to conventional video download approach. Furthermore, Simulation results show that DISCS achieves significant improvements over CSVD.

ACKNOWLEDGMENTS

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

REFERENCES

Al-Habashna A., Wainer G., Boudreau G., and Casselman R. "Improving Wireless Video Transmission In Cellular Networks Using D2D Communication". Canada. Provisional patent P47111. May 2015.

Al-Habashna A., Wainer G., Boudreau G., and Casselman R. "Cached and Segmented Video Download for Wireless Video Transmission". In *Proceedings of the 2016 ANSS*. pp. 1-8. Pasadena, USA.

Al-Habashna A., Wainer G., Boudreau G., and Casselman R., "Distributed Cached and Segmented Video Download for Video Transmission in Cellular Networks". In *International Symposium on Performance Evaluation of Computer and Telecommunication Systems*. pp. 1-8. Montreal, Canada.

Al-Hourani A., Chandrasekharan S., and Kandeepan S., "Path Loss Study for Millimeter Wave Device-to-Device Communications In Urban Environment". In *Proceedings of the 2014 IEEE ICC*. pp. 102-107. Sydney, Australia.

Abhari A. and Soraya M., "Workload Generation For YouTube". *Multimedia Tools and Applications*, vol. 46, no. 1, pp. 91-118, 2010.

Asadi A., Wang Q., and Mancuso V., "A Survey On Device-To-Device Communication In Cellular Networks". *IEEE Communications Surveys and Tutorials*, vol. 16, no. 4, pp. 1801 - 1819, 2014.

Ahlehagh H. and Dey S., "Hierarchical Video Caching In Wireless Cloud: Approaches and Algorithms". In *Proceedings of the 2012 IEEE ICC*. pp. 7082–7087. Ottawa, Canada.

Ahsan S. et al., "Characterizing Internet Video for Large-scale Active Measurements". Submitted to the *Networking and Internet Architecture*, arXiv preprint arXiv:1408.5777v1, 2014.

Cisco. "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update". http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html. Accessed May 05, 2016.

Cha M. et al., "I Tube, You Tube, Everybody Tubes: Analyzing The World's Largest User Generated Content Video System". In *Proceedings of the 2007ACM SIGCOMM*. pp. 1–14. San Diego, USA.

Chellouche S. A. et al., "Home-box-assisted Content Delivery Network For Internet Video-on-demand Services". In *Proceedings 2012 IEEE ISCC*. pp. 544–550. Cappadocia, Turkey.

Doppler K. et al., "Device-to-device Communication As An Underlay To LTE-advanced Networks". *IEEE Communications Magazine*, vol. 47, no. 12, pp. 42–49, 2009.

Golrezaei N., Mansourifard P., Molisch A. F., and Dimakis A. G., "Base-Station Assisted Device-To-Device Communications For High-Throughput Wireless Video Networks". *IEEE Transactions on Wireless Communications*, vol. 13, no. 7, pp. 3665-3676, 2014.

Kamoltham N., Nakorn K. N., and Rojviboonchai K., "From NS-2 to NS-3 - Implementation and Evaluation". In *2012 Computing, Communications and Applications Conference*. PP 35-40. Hong Kong, China.

Kaufman B. and Aazhang B., "Cellular Networks With An Overlaid Device to Device Network". In *Proceedings of 2008 Asilomar Conference on Signals, Systems and Computers*. pp. 1537–1541. Pacific Grove, USA.

Kang H. J. et al., "Mobile Caching Policies For Device-to-Device (D2D) Content Delivery Networking". In *2014 INFOCOM workshops*. pp. 299-304. Toronto, Canada.

Koc A., Jha S., Vannithamby R., and Torlak M., "Device Power Saving and Latency Optimization in LTE-A Networks Through DRX Configuration," *IEEE Transactions on Wireless Communications*, vol. 13, no. 5, pp. 1-12, 2014.

Koksal M., "A Survey of Network Simulators Supporting Wireless Networks". http://user.ceng.metu.edu.tr/~e1595354/A%20Survey%20of%20Network%20Simulators%20Supporting%20Wireless%20Networks.pdf. Accessed Dec. 12, 2016.

Maddah-Ali M. A. and Niesen U., "Decentralized Caching Attains Orderoptimal Memory-rate Tradeoff". *IEEE/ACM Transactions on Networking*, vol. 23, pp. 1029 - 1040, 2015.

Obaidat M. S., Zarai F., and Nicopolitidis P., *Modeling and Simulation of Computer Networks and Systems: Methodologies and Applications*. San Francisco: Morgan Kaufmann Publishers Inc, 2015.

Qiu Q., Chen J., Ping L., Zhang Q., and Pan X., "LTE/SAE Model and its Implementation in NS2". In 2009 *IEEE International Conference on Mobile Ad-hoc and Sensor Networks*. pp. 299-303. Fujian, China

Rajankumar P., Nimisha P., and Kamboj P., "A Comparative Study and Simulation of AODV MANET Routing Protocol In NS2 & NS3". In *2014 International Conference on Computing for Sustainable Global Development*. pp. 889-894. New Delhi, India.

Ros F. J., Martinez J. A., and Ruiz P. M., "A survey On Modeling and Simulation of Vehicular Networks: Communications, Mobility, and Tools". *Computer Communications*, vol. 43, pp. 1–15, 2014.

Wainer G., *Discrete-event Modeling and Simulation: A Practitioner's Approach*. Boca Raton: CRC/Taylor & Francis Group, 2009.

Wang B. and Ray L. K. J., "Advances In Cognitive Radio Networks: A Survey". *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 1, pp. 5-23, 2011.

Xian X., Shi W., and Huang H., "Comparison of OMNET++ and Other Simulator for WSN Simulation". In *Proceedings of the 2008 IEEE Conference on Industrial Electronics and Applications*. pp. 1439-1443. Singapore, Singapore.

Zeigler B., Praehofer H., and Kim T., *Theory of Modeling and Simulation*. San Diego: Academic Press, 2000. 3rd Generation Partnership Project, "Technical Report 36.942, V12.0.0". 2014.

AUTHOR BIOGRAPHIES

ALA'A AL-HABASHNA is a PhD candidate in the Department of Systems and Computer Engineering at Carleton University, Ottawa, ON, Canada. His email address is alaaalhabashna@sce.carleton.ca.

GABRIEL WAINER is a Professor at the Department of Systems and Computer Engineering at Carleton University. He is a Fellow of SCS. His email address is gwainer@sce.carleton.ca.