On a Cloud-Controlled Architecture for Device-to-Device Content Distribution

Jiri Danihelka
ETH Zurich
Zurich, Switzerland

Domenico Giustiniano
IMDEA Networks Institute
Madrid, Spain

Bernhard Plattner
ETH Zurich
Zurich, Switzerland

ABSTRACT

It has been shown that the distribution of popular content can benefit from solutions that dynamically distribute copies of the content from the backend to a subset of subscribed users, and let these users spread the content with opportunistic communication. In this work, we study with an experimental analysis how cloud computing could help to disseminate the popular content for the above scenario. We design an architecture for cloud-based controller of content deliveries, and we investigate strategies for delivering the content with the cloud logic. We implement our system using Microsoft Azure cloud service and building a video application running on commodity smartphone. We experimentally compare different strategies and show that solutions controlled by the cloud are more efficient in terms of traffic offload than approaches without cloud logic, and that practical challenges are solved by our approach that were not considered in former analytical works.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless Communication*

Keywords

Mobile Data offloading; Opportunistic Networks; Cloud Computing; Design; Implementation; Evaluation

1. INTRODUCTION

A recent forecast study made by Cisco Inc. [1] shows that the mobile phone workload of cellular networks will be doubling or tripling every year. This dramatic traffic growth is driven by mobile video streaming, which will account for 72% of total mobile data traffic by 2019. Although the problem can be postponed by building the next generation of cellular networks, it is expected that the principal challenges will remain the same [2]. Due to this exponential growth of mobile data traffic and bandwidth-hungry applications, there is a demand for new approaches to access content. It has been shown that the devices can cooperate to download video content from the Internet and share it using their opportunistic wireless connection (e.g. Bluetooth), in the attempt to address the cellular bandwidth crunch problem [3]. Analytical work has further shown that dynamic decisions can be taken by the controller logic residing in the backend, such that the mobility of users and limited control overhead are jointly taken into account to decide to which users to send the content, and then let these users spread the content with opportunistic communication [4,5]. This concept intentionally introduces delays in the content delivery with the goal of reducing cellular data traffic [6], and it is of interest for *those applications where a local group of users carrying their own mobile device is interested in the same content*, like watching a popular Internet video [3], reading news feeds [7], and receive a flow of road traffic updates [4] and social data [8]. In this work, we focus on the design and implementation of solutions that address the problem of video content delivery in practice - by far the most challenging in terms of network resources among the set of use cases above. However, our ideas and concepts can be applied or reformulated for other use cases as well. By exploiting the similarity of interests among multiple mobile users, multiple contents can be also delivered using an extension of our architecture.

In the context of this work, it remains still unknown how to design and deploy a system architecture to deliver the popular video content. The specific question we address is how *cloud computing could tackle the overloading of the cellular networks and how it could help the network operators to alleviate the load of the network*. Second, users may be at one position or may move, and the *available bandwidth of device-to-device (D2D) communication* (and thus the offload opportunities) can greatly change over time. We then make the following contributions:

- We design a cloud service that coordinates D2D dissemination steps, based on low-signaling feedbacks from the opportunistic network.
- We investigate several strategies to reduce the cellular network load while taking into account realistic limitations of the opportunistic bandwidth.
- We implement our architecture using Windows Azure cloud service and a dedicated application running in smartphones, and we test our implementation in representative experiments.

We measure the performance of our methods with up to 16 smartphones, and further emulate environments where congestion of opportunistic D2D deliveries may occur. In our emulation environment conducted in a controlled setup, we achieve up to 71% of saved bandwidth with all the phones in D2D range. Our practical experiment with fewer phones in D2D range shows that users can save 39% on average and 53% in peak values.

2. SYSTEM ARCHITECTURE

We first introduce the scenario which is objective of investigation in this paper, and we then present our system architecture.

CHANTS'15, September 11, 2015, Paris, France.

DOI: http://dx.doi.org/10.1145/2799371.2799377.

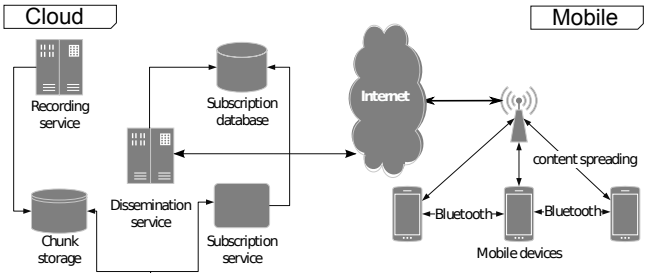

Figure 1: Schema of the cloud-based distribution system

2.1 Scenario

The system consists of a cloud component and a mobile component. A schema of the whole system is shown in Figure 1. The cloud part of the distribution system contains a media recording service, listening to online video streams on the Internet. Because mobile devices do not handle all video formats, the media recording service also converts the video content to a specific compression format and splits its length into fixed intervals. We refer to these short content files as *chunks*. The media recording service stores them afterwards in a cloud virtual drive called Chunk storage. A new mobile device can register to desired channels using the Subscription service, see Figure 1, and the channels per user are stored in the Subscription database.

We suppose that there are N users that are all subscribed to a common channel and want to fetch a content. In order to disseminate each chunk k of the entire content, each data chunk is initially delivered to m_k devices, that spread it throughout the opportunistic network to devices without the chunk for a time period T. When the time T has passed, there are m'_k devices without the data chunk yet. These devices request the chunk to the cloud. This allows for *guaranteed delays* of the content delivery and to use an application like video-streaming without interruptions in the service [4–6].

2.2 Dissemination service

The central component of the cloud is the Dissemination service. For each chunk, its main goal is to select the m_k devices that will be used for initial content spread. The initial spreading time of chunk k to m_k devices is denoted *spread period*, as shown in Figure 2. According to the specific strategy, the dissemination service might also provide additional information.

Once this injection from the cloud is concluded, the devices enter in the *dissemination period*, which is the time T designated for D2D dissemination of the chunk to other devices. In this work, we use Bluetooth for D2D communication. Each mobile device tries to connect to another device within its Bluetooth proximity. In case of successful establishment of a connection, the two devices compare their lists of chunks that have been previously downloaded. In order to increase the transfer rate, the dissemination periods of different chunks in a video sequence can overlap, so it is possible to share more than just the last chunk. The overlap of different chunks is shown in Figure 2. A random order for the delivery is defined for chunks present in only one of the two devices. The devices disconnect when there are no other chunks to be shared, or when their connection has been interrupted due to a communication failure. Each chunk that has not be fully downloaded due to communication failure is lost. Therefore, the size of the chunk should be sufficiently small to guarantee an efficient delivery, e.g. in presence of users with short contact duration δ. After that, both devices start searching for another device to connect to.

All deliveries from the cloud take place either before (m_k deliveries) or after (m'_k deliveries) the dissemination period. This has the

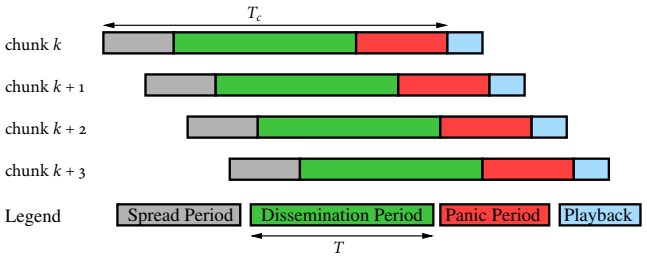

Figure 2: Timeline for content distribution. Each line represents the distribution of one data chunk. Data is sent from the cloud during the spread period and the panic period, and from opportunistic D2D communication during the dissemination period T. The chunks' dissemination period can overlap.

advantage of avoiding to monitor how information spreads during the dissemination period. We use the term *dissemination deadline* to denote a point in time when the opportunistic dissemination ends for a particular chunk k. We refer to the time interval after the dissemination deadline as *panic period* (see Figure 2), because the devices use their costly connection to download the chunk before its playback should start. The duration of the panic period has to be chosen to guarantee the delivery in the worst-case scenario of slow cellular connection to any of the m'_k devices.

Overall, the total number of data transmissions through cellular network is $M_k = m_k + m'_k$. *We want to design a system architecture and strategies for deliveries that leverages the cloud to control the opportunistic dissemination.* The goal is to reduce the traffic from the cellular network M_k as much as possible, while meeting the requirement that all the devices have received the content by the content's deadline T_c, defined as the sum of the spread period, the dissemination period T and the panic period (see Figure 2).

3. INITIAL SPREADING STRATEGIES

In this work, we investigate two general types of strategies that aim to reduce the traffic load of cellular network:

- *Initial spreading strategies*: for each chunk, it decides how many (m_k) and which devices should be selected for deliveries during the spread period (Section 3).
- *Dissemination strategies*: it determines when and which devices should establish D2D communication during the dissemination period, and which data they should share (Section 4).

We start investigating three strategies to inject m_k copies of the chuck k during the spread period, which can be divided in strategies with and without cloud logic. In the strategy without cloud logic, we simply randomly select m_k devices among the total set of N devices. In the strategies with cloud logic, we randomly select the m_k devices *per each connected component* of the opportunistic network to avoid the problem that all the devices of one connected component may all need to download the chunk from the cloud during the panic period. The topology of the network is estimated by the cloud based on the feedbacks received by each node of the opportunistic network. The report of each user provides the list of other mobile devices that are visible in opportunistic range. Note that the set of devices chosen by the cloud for chunk $k+1$ is independent of the set chosen for the previous chunk k.

Next, we describe how to select how many copies m_k of the chunk k have to be sent during the spread period.

3.1 Multi-armed bandit strategy

The dynamics of the content distribution system are subjected to changes. For instance, inter-contact time statistics may vary de-

pending on the time of the day [9]. Therefore, the initial spreading strategy should select m_k according to the history of the content distribution system. The first strategy we study in this work builds on the multi-armed bandit problem [10]. The multi-armed bandit problem has found applications in diverse fields, such as control, economics, statistics, or learning theory. In the original multi-armed bandit problem a gambler faces a row of slot machines and has to decide which of the N machines to play in each of a number of games. Each machine provides a random reward from a distribution specific to that machine. The gambler's objective is to maximize the sum of the rewards earned through a sequence of lever pulls. More formally, let $r_{k,1} \ldots, r_{k,N}$ be the rewards for each of the N levers at round k, and $\hat{r}_k$ the reward for the selected lever at round k. Let also $\mu_1, \ldots, \mu_N$ be the mean values associated with reward distributions of the N levers and $\mu^* = \max_k \{\mu_k\}$ the maximal reward mean. The regret ρ after R rounds is defined as $\rho = R\mu^* - \sum_{k=1}^{R} \hat{r}_k$, that is, the difference between the reward sum associated with an optimal strategy and the sum of the collected rewards. A strategy is defined "a zero-regret strategy", when the average regret per round ρ/R tends to zero with probability 1 when the number of played rounds tends to infinity.

We can identify an analogy between the multi-armed bandit problem and our content distribution problem. The number of games in the multi-armed bandit problem corresponds to the number of data chunks in the content, as they both represent the number of rounds in the problem. The number of levers N corresponds to the number of subscribed mobile devices N. The gambler's reward for chunk (round) k can be defined as the offloading ratio

$$\hat{r}_k = \frac{N - M_k}{N}$$

saved by selecting the lever m_k.

We implement a content dissemination algorithm that is based on the Epsilon-greedy method to solve our multi-armed bandit problem [10]. In this approach, the strategy will pick m_k that gives its maximal reward $\hat{r}_k$ (according to the current gambler's findings) for $1 - \epsilon$ of the time ($0 < \epsilon < 1$), and another lever is randomly selected for a proportion ϵ, which represents the likelihood of performing exploration moves. The algorithm then attempts to maximize the short-term reward based on current findings and to discover optimal parameters for long-term rewards in future (exploration vs. exploitation trade-off). This method is shown to provide a good approximation of the optimal decision [11].

For the algorithmic implementation, the empirical expectation of the reward function for m_k is stored in a dedicated array as well as the certainty (confidence) on each value. The confidence for a value decreases in each round by an aging factor γ. Values close to the actual m_k have the chance to be selected in proportion to their uncertainty. Therefore, the more uncertain we are about the correctness of a reward function, the more probable we will select the corresponding lever m_k for the next round in exploration phase.

3.2 Initial / deadline balance

This second strategy considers that a large number of downloads after the dissemination deadline indicate insufficient initial spread of chunks by the cloud, and vice versa. Recording in the dissemination service the number of chunks that were distributed in the last panic period, the objective of this strategy is to drive the system to the case where *the number of data chunks injected from the cloud during the spread period* is approximately equal to *the number of chunks injected during the panic period*, that is $m_k = m'_k$. Theoretical justifications on the initial/deadline balance strategy can be derived simplifying the assumptions presented in [5] for sake of simplicity of the system implementation, modeling the pair-wise contact event as an independent and memoryless Poisson process of rate λ and describing the chunk dissemination during the dissemination period T as a pure finite-state birth process $\{S(t), t \geq 0\}$[1]. For the selection of m_k with initial/deadline balance strategy, we implement an adaptive algorithm according to a control theoretic approach based on Proportional Integrator (PI) controller. In order to deliver one chunk, the controller monitors the system behavior (and in particular the output signal $m'_k - m_k$) given the value m_k that is currently used. Based on this behavior, it decides whether to increase or decrease m_k for the next chunk in order to drive the system to the reference value zero[2]. The parameters K_p and K_i of the PI controller are chosen as a trade-off between a stable and reactive system, using Ziegler-Nichols rules [12] and imposing the stability constraint on the closed-loop gain.

3.3 Fixed ratio spread

Finally, a simple distribution strategy for initial spread is to delivery the chunk to a fixed ratio m/N of the N subscribers. This strategy can also be implemented in scenarios that do not use distribution logic in a cloud component (and thus client-only). In this case, at the beginning of the initial spread period per chunk, a mobile device downloads the chunk with given probability.

4. DISSEMINATION STRATEGIES

We study three strategies to disseminate the content chucks during the dissemination period.

4.1 Client-only dissemination

After the initial spread period, each mobile device tries to connect to a random device within its Bluetooth proximity and download the missing chunks if available in the node in proximity. The client-only dissemination strategy uses the cloud only for downloading the chunks, both in the spread and the panic periods.

4.2 Cloud-based dissemination

Former work did not consider that congestion can happen also in the D2D communication [3, 5, 6]. In this strategy, each mobile device reports to the cloud service every chunk downloaded from D2D dissemination. When a mobile device completes its scanning of nearby Bluetooth devices, it downloads a list of the available chunks per device from the Dissemination service in the cloud. This knowledge ensures that the mobile device avoids connecting to other mobile devices that have no chunks available for sharing.

4.3 Adaptive cloud-based dissemination

As a limitation of Bluetooth technology, a device cannot accept a Bluetooth connection while it is (a) trying to connect to another device, or (b) scanning for nearby devices. We then consider that, when either (a) or (b) is concluded, the device waits for a random-length interval t for other incoming connections before actively scanning, where $t \in [t_1, t_2]$ and $(t_2 - t_1)$ is the listening period. In addition, this strategy reduces the frequency of communication attempts in cases there are too many connection failures, and that increases the random period when the devices are listening for incoming connections. In the event of an unsuccessful Bluetooth connection attempt the strategy will increase the listening period $(t_2 - t_1)$ by the multiplication-factor parameter α. In the event of

[1] This is a pure-birth Markov chain with N states, where state S_i corresponds to the case where i devices have the chunk.

[2] Note that the selection of m_k for chunk k is based on the last available feedback. Since we consider that the dissemination periods of chunks may overlap, this will be in general equal to $m'_{k-i} - m_{k-i}$, relative to chunk $k - i$, with $i \geq 1$.

Table 1: Setting used in the experimental evaluation

Parameters common to all strategies	
Video compression	MPEG
Video resolution	426 × 240 pixels
Content's deadline	T_c = 100 sec
Initial spread phase length	15 sec
Dissemination period length	T = 55 sec
Panic period length	30 sec
Playback time for each chunk	2 – 10 sec
Average chunk size	≈ 500 kB

Parameters specific to individual strategies	
Default listening period	t_1 = 5 sec; t_2 = 15 sec
Fixed spread ratio	25%
Adaptive factor α	1.15
Adaptive factor β	0.95
$\mathbf{K_p}$ (Initial/deadline balance strategy)	0.2
$\mathbf{K_i}$ (Initial/deadline balance strategy)	0.1176
Aging factor γ (multi-bandit strategy)	0.9

a connection success, the strategy will decrease the listening period by the multiplication-factor parameter β.

5. SYSTEM IMPLEMENTATION

In this section, we introduce the implementation details of our cloud-controlled system for D2D content distribution.

5.1 Cloud services

All our cloud services are hosted in the Windows Azure environment. Applications for Microsoft Azure are written in C# using .NET libraries. The Recording service and the Dissemination service run as worker role instances. Chunk storage is implemented like cloud blob storage. We use Azure Mobile Services technology for the Subscription service, which allows us to use seamless integration of cloud services into mobile client code with support from development tools. The subscription database is based on Azure SQL Database (the cloud version of the Microsoft SQL Server). A new device can register to desired video channels and the worker role of the Dissemination service is then notified about the new device and its content requests. It will then consider the new device in its future deliveries.

5.2 Mobile devices

We implement a video streaming application that runs on HTC8S Windows Phone 8 mobile devices. We choose Windows for better integration with Windows Azure cloud service. For the purposes of this paper, we use Bluetooth 3.1 for D2D communication. *The video in the application is transparent to the actual wireless interface used to receive the chunks*. Note that Wi-Fi Direct has limited support at the time of writing with Windows Phone 8. The application handles initial D2D discovery and it can send signaling messages to the cloud, if requested by the strategy.

Signaling For the strategies with cloud logic, signaling mechanism is controlled by the mobile device. When a new chunk is available, the device queries the Azure Mobile Service. The reply will state whether the chunk should be downloaded immediately or not from the cloud during the spread period. In the latter case, the device will attempt to get the content during the dissemination period. In case the device has not received the chunk yet by the end of the dissemination period, it will make a new query to the Azure Mobile Service to request the chunk. Therefore, at least one query and up to two queries per chunk are performed.

Overall, the size of the signaling (up to 400 Bytes) is such that it is largely negligible for reasonable chunk sizes, such as the ones of video content. The request from the device to the cloud (D2C) starts with a Strategy field, which indicates the strategy used. The last bit of the Strategy field is a flag, indicating whether the device is in spread period or panic period. Next, there is the chunk ID that the device is querying for download (Request Chunk). If requested by the Dissemination strategy, it will further send the list of chunks downloaded by the devices and available for D2D communication (that is, currently in dissemination period, Dissemination Chunks) as well as the devices in D2D range (Nearby Devices).

The first field of the response from the Azure mobile service is the Strategy field. As in the request, the last bit of the Strategy field is a flag. For a response, the flag indicates whether the device should download or not the chunk from the cloud. In case of positive reply, the uniform resource identifier (URI) is given to the chunk to identify the path in Chunk Storage (Chunk URI). If requested by the strategy and if the device did not indicate that it is in panic period, a list of nearby devices and their chunks can also be provided (Nearby Devices - Chunks per Device).

D2D communication The D2D communication is initiated by one device (device A) after an active scanning. In the event that a connection is successfully established, device A connects to another device (device B) which is in Bluetooth listening mode. The communication is composed by a handshake to exchange the set of chunks that are available and missing. The missing chunks in one device and available in the other device are exchanged according to a random order.

System setting The main parameters are summarized in Table 1. In our implementation, we use a deadline of T_c = 100 sec. For the purpose of performance measurements, we use prerecorded video streams. This approach circumvents any problems with the streaming source and the reproducibility of the tests. We use prearranged chunk files in Chunk storage of up to 500 kB, with recording interval between 2 and 10 sec (The time per chunk is greatly depending on the content of the chunk. Chunks with lively action deplete very quickly 500 kB of size). Therefore, multiple chunks may be exchanged during the dissemination opportunity interval of T = 55 sec. The total video size is approximately 50 MB and the total number of chunks per video is 99. The video is continuosly repeated once it is over. The total number of rounds R is equal to the total number of chunks of the entire experiment.

6. EXPERIMENTAL EVALUATION

We perform two different types of evaluation, first using a controlled setup, and then measuring the performance with real users.

6.1 Automated testing system

In order to compare and evaluate the strategies described in Section 3 and 4, we implement an automated module that tests the performance of the Dissemination service. The module repeatedly sends a short video stream (10 minutes) to mobile devices. After each test, the module changes the dissemination strategy that is used, or its parameters. The module can also place a mobile phone temporarily outside the dissemination process for one test by sending a command to turn off its Bluetooth connection. This step enables measurements to be made for groups between 2 and 16 phones. We run the experiments overnight to minimize the interference from other 2.4 GHz radio sources.

Results The results of automated testing are shown in Figure 3, that shows the saved bandwidth (in percentage) over the number of devices used in the experiments. The saved bandwidth is defined as

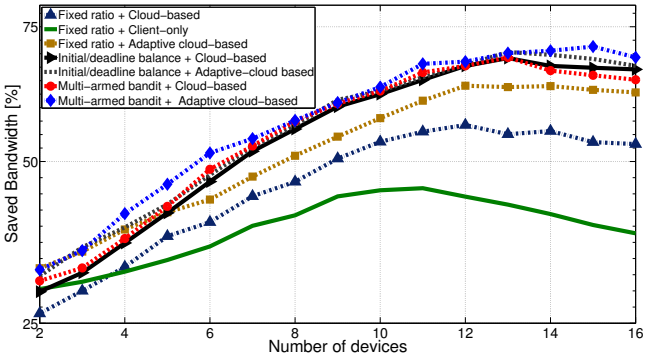

Figure 3: Comparison of saved bandwidth for different strategies. Each value is the average over three tests (each test is 10 minutes long).

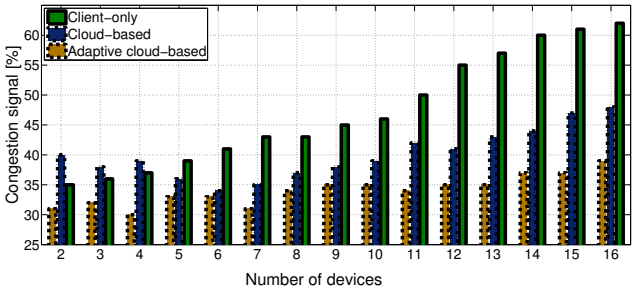

Figure 4: Congestion signals of D2D communication for the three different dissemination strategies.

the amount of data in byte downloaded over D2D communication over the total video size in byte downloaded for the video.

We first analyze the result of initial spreading strategy. As baseline strategy for the comparison, we use *cloud-based dissemination* as dissemination strategy. Results of initial spread strategy in Figure 3 show that multi-armed bandit strategy and the initial/deadline balance strategy achieve similar performance in terms of saved bandwidth. However, while the first one achieves slightly better performance for small number of devices in range, the opposite holds for higher number of devices. The reason why the initial/deadline balance strategy is slightly worst than expected is likely due to the fact that this strategy implicitly assume that transmission of a single chunk is instantaneous in both the cloud-to-device and the D2D communication [5]. In contrast, multi-armed bandit strategy does not rely on this assumption.

We further notice that a fixed ratio strategy achieves poor performance, and the performance decreases for more than 12 devices. This is because the bandwidth available for D2D communication becomes the bottleneck and thus more injections in absolute in the spread periods have only the effect to increase the failed attempts of D2D communication.

We then study the dissemination strategy, again referring to Figure 3. As baseline for the comparison, we use *fixed ratio spread* as initial spreading strategy and study which dissemination strategy helps to reduce the congestion of the D2D communication. We observe that cloud-based strategies outperforms the client-only one. Not shown in the figure, results with client-only get even worst with higher fixed spread ratio. For instance, using a fixed spread ratio of 50% rather than 25%, we report 31.43% of saved bandwidth with 16 devices rather than 36.66%. The best performing dissemination strategy is the adaptive cloud-based strategy, that can dynamically adjust the communication load of opportunistic network.

Finally, we compare the initial-deadline balance and multi-armed bandit, using the adaptive cloud-based as dissemination strategy.

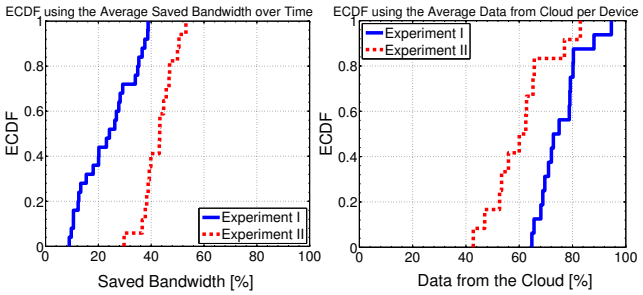

Figure 5: Comparison of Experiment I and Experiment II. On the left: ECDF of average saved bandwidth during the entire experiment. Each sample is the average saved bandwidth over a short time interval of ≈ 15 minutes. On the right: ECDF using the average data downloaded from the cloud per each device.

The plot in Figure 3 shows that the *combination of multi-armed bandit and adaptive cloud-based strategies generally outperforms any other strategies*, with up to 71% of saved bandwidth.

We analyze in more details the performance loss for high number of devices. We define the congestion signal as the number (in percentage) of unsuccessful opportunistic connections between mobile devices over the total number of attempts. Figure 4 shows that a Client-only strategy is greatly affected by a very high rate of congestion signals for increasing number of devices. The consequence is that the devices are not able to download the chunk using D2D communication, despite there are devices in range with a copy of it. In contrast, the adaptive cloud strategy shows a quite stable level of congestion signal, which explains why it outperforms other strategies in terms of saved bandwidth.

6.2 Evaluation with real users

To evaluate our system in practical scenarios, we organize two experiments in different days with volunteers that carry a mobile device with our application. The volunteers work in the same floor and different offices, and they carry the mobile device all the time. In Experiment I, we have 16 volunteers. In Experiment II, we have 12 volunteers. For Experiment I, we use *fixed ratio* as initial strategy and *client-only* as dissemination strategy. For Experiment II, we use the *multi-armed bandit* as initial spreading strategy and the *adaptive cloud-based dissemination* as dissemination strategy.

Both experiments are performed during working time when most of the volunteers are inside the building or in its proximity. The devices stream a video with a bit rate close to 400 kb/s (a usual bandwidth for a low-resolution YouTube video). The experiment attempts to emulate situations when multiple users want to access the same video content and may have the opportunity to be in D2D communication range with some of other volunteers. The users stay in the office or move according to their usual pattern.

Results We first measure there are more opportunities to share the chunks with D2D communication in Experiment I, which reports 3.53 devices in range on average (≈ 32% of the other users), while Experiment II gives 2.52 devices (≈ 17% of the other users). We then compare the Empirical Comulative Distribution Function (ECDF) of both experiments in Figure 5. On the left of the figure, we show the saved bandwidth, where each sample is the average bandwidth over a time interval of ≈ 15 mins. The figure shows that Experiment II, that uses a combination of initial spread and dissemination strategies (with superior performance in the automated testing system in the previous section), obtains significantly higher saved bandwidth than Experiment I. This results has been achieved despite there are more opportunities to share the chunks with D2D in Experiment I.

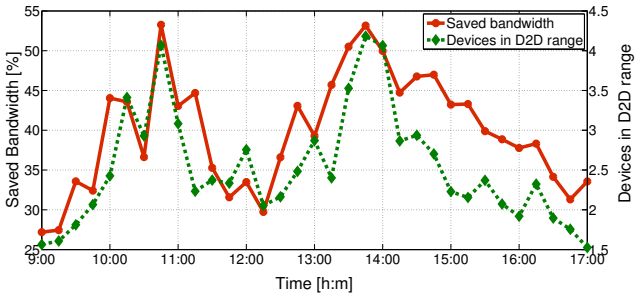

Figure 6: Bandwidth savings for Experiment II, and average number of nearby devices after Bluetooth scanning.

We then study the amount of data (in bytes) that each device downloads from the cloud over the total amount of data downloaded from both the cloud and D2D communication and compute how fair is the injection of chunks from the cloud using different strategies. For each device, we compute the average over the entire experiment, and plot the ECDF of data downloaded from the cloud on the right of Figure 5. We measure a Jain's fairness index of 0.989 for Experiment I and 0.967 for Experiment II. This results show there is a fair access to the costly wireless network interface. Concluding, a cloud-based approach is preferable in terms of saved bandwidth and it comes at negligible cost in terms of fairness.

In Figure 6, we finally depict the saved bandwidth, and the average number of nearby devices for Experiment II as a function of the time. The resulting saved bandwidth is 39% on average and 53% in two peak values at 10:45 and 13:45. The figure shows that the amount of saved bandwidth tends to increase and decrease according to the number of nearby devices in range, with a high Pearson correlation coefficient between the two variables of 0.79. As a result of this high correlation, we also observe that the strategy can adapt to variable conditions of the D2D network topology.

7. RELATED WORK

The principles of cooperative techniques to disseminate content have been presented in [13], motivating that cooperation will become one of the key technologies enabling better cellular networks. The authors of [14, 15] studied a cooperative technique for opportunistic podcast and feed distribution. However, they assumed that there is absence of cellular infrastructure, while we assume that this infrastructure exists, but the bandwidth does not suffice to disseminate popular content to all the subscribed users. [16] described a cooperative video streaming architecture with mobile devices sharing a single access point and D2D connectivity. [17] allowed a single video content to be received by multiple mobile devices using multicast from a common base station. Multicast increases the cost of complexity of the cellular base station deployment and it requires that traffic is sent at lower rate [3]. Differently from these works, we design strategies to disseminate the content under dynamic conditions and variable D2D network topology and congestion, and test these strategies with experiments and cloud services.

MicroCast [3] introduced a system for cooperative video streaming using cellular connection and Wi-Fi peer-to-peer connection. Their algorithm assigns the next segment of a video to be downloaded to a phone which has the smallest set of segments to download from the cellular network. In our system, we consider practical problems such as congestion of D2D communication, algorithms that can choose the exact number of devices such that the saved bandwidth is a high as possible, etc. [6] made a case study for cellular traffic offloading using opportunistic connections. They proposed heuristics to select the target set of users, in order to minimize the cellular data traffic. [4, 5] studies through heuristics and analytical study, respectively, the problem of injecting copies of the content to the users. However, none of those works looked at strategies for the distribution of D2D content during the dissemination period and the evaluation is solely done by means of simulations.

8. CONCLUSION

We have implemented and experimentally evaluated a novel architecture to disseminate popular video content to subscribed users. We have experimentally shown that cloud logic can help to alleviate the saturation of cellular network traffic and that an initial spreading strategy based on multi-armed bandit problem provides high offloading opportunities. We envision that the inherent scalability properties of the cloud can allow to deploy multiple instances of the media recording service for a large number of requested media streams and dynamically adapt the allocated network resources according to the number of subscribed users and the dynamics of D2D communication.

Acknowledgments

This work has been partially funded by Microsoft Innovation Cluster for Embedded Software (ICES) HyCloud project, and partially by the European Commission in the framework of the H2020-ICT-2014-2 project Flex5Gware (Grant agreement no. 671563). We thank Vincenzo Sciancalepore for his constructive feedbacks.

9. REFERENCES

[1] "Cisco visual networking index: Global mobile data traffic forecast update, 2014–2019," White Paper, Cisco, 2015.

[2] S. Y. Hui and K. H. Yeung, "Challenges in the migration to 4g mobile systems," *Communications Magazine, IEEE*, vol. 41, no. 12, pp. 54–59, 2003.

[3] L. Keller, A. Le, B. Cici, H. Seferoglu, C. Fragouli, and A. Markopoulou, "Microcast: cooperative video streaming on smartphones," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 2012, pp. 57–70.

[4] J. Whitbeck, M. Amorim, Y. Lopez, J. Leguay, and V. Conan, "Relieving the wireless infrastructure: When opportunistic networks meet guaranteed delays," in *Proceedings of IEEE WoWMoM*, 2011.

[5] V. Sciancalepore, D. Giustiniano, A. Banchs, and A. Picu, "Offloading cellular traffic through opportunistic communications: Analysis and optimization," *Accepted for publication at IEEE Journal on Selected Areas in Communications*, 2015.

[6] B. Han, P. Hui, V. Kumar, M. V. Marathe, G. Pei, and A. Srinivasan, "Cellular traffic offloading through opportunistic communications: a case study," in *Proceedings of the 5th ACM workshop on Challenged networks*. ACM, 2010, pp. 31–38.

[7] S. Ioannidis, A. Chaintreau, and L. Massoulie, "Optimal and scalable distribution of content updates over a mobile social network," in *Proceedings of IEEE INFOCOM*, 2009.

[8] B. Han, P. Hui, V. A. Kumar, M. V. Marathe, G. Pei, and A. Srinivasan, "Cellular traffic offloading through opportunistic communications: A Case Study," in *Proceedings of ACM CHANTS*, 2010.

[9] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on opportunistic forwarding algorithms," *Mobile Computing, IEEE Transactions on*, vol. 6, no. 6, pp. 606–620, June 2007.

[10] R. Sutton and A. Barto, *Reinforcement learning: An introduction*. Cambridge Univ Press, 1998, vol. 1, no. 1.

[11] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *CoRR*, vol. abs/1204.5721, 2012. [Online]. Available: http://arxiv.org/abs/1204.5721

[12] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamic Systems*, 2nd ed. Addison-Wesley, 1990.

[13] F. H. Fitzek and M. D. Katz, *Cooperation in wireless networks: Principles and applications*. Springer, 2006.

[14] V. Lenders, G. Karlsson, and M. May, "Wireless ad hoc podcasting," in *Sensor, Mesh and Ad Hoc Communications and Networks, 2007. SECON'07. 4th Annual IEEE Communications Society Conference on*. IEEE, 2007, pp. 273–283.

[15] G. Karlsson, V. Lenders, and M. May, "Delay-tolerant broadcasting," *Broadcasting, IEEE Transactions on*, vol. 53, no. 1, pp. 369–381, 2007.

[16] M. Ramadan, L. El Zein, and Z. Dawy, "Implementation and evaluation of cooperative video streaming for mobile devices," in *Personal, Indoor and Mobile Radio Communications, 2008. PIMRC 2008. IEEE 19th International Symposium on*. IEEE, 2008, pp. 1–5.

[17] S. Hua, Y. Guo, Y. Liu, H. Liu, and S. S. Panwar, "Scalable video multicast in hybrid 3g/ad-hoc networks," *Multimedia, IEEE Transactions on*, vol. 13, no. 2, pp. 402–413, 2011.