MDS-Coded Distributed Caching for Low Delay Wireless Content Delivery

Citation for the original published paper (version of record):
Piemontese, A., Graell i Amat, A. (2019). MDS-Coded Distributed Caching for Low Delay Wireless Content Delivery. IEEE Transactions on Communications, 67(2): 1600-1612.
http://dx.doi.org/10.1109/TCOMM.2018.2876921

(article starts on next page)

MDS-Coded Distributed Caching for Low Delay Wireless Content Delivery

Amina Piemontese, *Member, IEEE*, and Alexandre Graell i Amat, *Senior Member, IEEE*

***Abstract*—We investigate the use of maximum distance separable (MDS) codes to cache popular content to reduce the download delay of wireless content delivery. In particular, we consider a cellular system where popular files are cached in a distributed fashion in a limited number of the mobile devices using an MDS code and can be downloaded from them using device-to-device (D2D) communication. The base station controls the D2D communication and assists the requests that cannot be fully satisfied by the distributed caching (DC) network by providing the missing data. We consider a network model where the cell is divided into clusters, where D2D links can be activated. We derive an analytical expression for the delay incurred in downloading content from the wireless network assuming that devices roam in and out of clusters according to a Poisson random process. Our analysis allows to identify the parameters of the wireless network that mostly affect the performance and to compare different caching strategies in terms of delay. We show that DC using MDS codes can dramatically reduce the download delay with respect to the scenario where content is always downloaded from the base station and to the case of uncoded DC.**

I. Introduction

The proliferation of mobile devices and the surge of a myriad of multimedia applications has resulted in an exponential growth of the mobile data traffic. In this context, wireless caching has emerged as a powerful technique to overcome the backhaul bottleneck, by reducing the backhaul rate and the delay in retrieving content from the network. The key idea is to store popular content closer to the end users. In [1], a novel system architecture named *femtocaching* was proposed. It consists of deploying a number of small base stations (BSs) with large storage capacity, in which content is stored during periods of offpeak traffic. The mobile users can download content from the small BSs, resulting in a higher throughput per user.

In [2], it was proposed to store content directly in the mobile devices. Caching in the mobile devices has attracted a significant interest in the research community in the recent years [3]–[6]. In this scenario, users can then retrieve content from neighboring devices using device-to-device (D2D) communication or, alternatively, from the serving BS. Content may be stored using an erasure correcting code, which brings gains with respect to uncoded caching [1], [7]–[10]. The use of erasure correcting codes establishes an interesting link between distributed caching (DC) for content delivery and distributed storage for reliable data storage. The key difference is that in the wireless network scenario, data can be downloaded from the caching nodes (the small BSs or the mobile devices) but also from a serving macro BS, which has always the content available. Therefore, the reliability requirements in distributed storage for reliable data storage can be relaxed. In [9], the placement of content encoded using maximum distance separable (MDS) codes to small BSs was investigated and it was shown that a careful placement allows to significantly reduce the backhaul rate. In [7], for the scenario where content is stored directly in the mobile devices, the repairing of the lost data when a device caching data leaves the network was considered. Assuming instantaneous repair, the communication cost of data download and repair was investigated. In [8], [10], a repair scheduling where repair is performed periodically was introduced and analytical expressions for the overall communication cost of content download and data repair as a function of the repair interval were derived. Using these expressions, the overall communication cost entailed by caching content using MDS codes, regenerating codes [11], and locally repairable codes [12] was evaluated in [10] and it was shown that caching content using erasure correcting code can reduce the overall communication cost with respect to the scenario where content is downloaded solely from the BS.

Most previous works in the literature have focused on the cache hit probability and/or the communication cost and assume that the download of content is instantaneous. For the scenario where content is cached in small BSs, the expected file download delay is minimized over the cache content placement in [1], assuming that content is cached using *ideal* MDS rateless codes. However, while the mobile devices are spread randomly over an area, no mobility of the mobile devices is considered in the analysis. Mobility of the devices, assuming a random walk, is then considered in Monte Carlo simulations.

In this paper, we analyze the delay of retrieving content from the cellular network when content is cached directly in the mobile devices taking into account the mobility of the devices during the download process. In particular, we consider a similar cellular network scenario as the one in [7], [10], where content is cached in a number of mobile devices using an erasure correcting code. However, differently from [7], [10], we study a network model where the cell is divided into clusters where D2D links can be activated in order to increase the spatial reuse, and hence to reduce the latency. We assume that mobile devices roam in and out of a cluster according to a Poisson random process. We derive analytical expressions for the average file download

The paper was presented in part at the International Symposium on Turbo Codes & Iterative Information Processing, Brest, France, Sep. 2016.

The authors are with the Department of Electrical Engineering, Chalmers University of Technology, 412 96 Gothenburg, Sweden (e-mail: {aminap,alexandre.graell}@chalmers.se.

Amina Piemontese is supported by a Marie Curie fellowship (contract 658785-DISC-H2020-MSCA-IF-2014). This work was also was partially funded by the Swedish Research Council under grant #2016-04253.

delay where content is cached in the mobile devices using MDS codes and show that MDS-coded DC can significantly reduce the download delay with respect to the case where content is solely downloaded from the BS and the case where uncoded caching is used. This work is an extension of our previous work [13], where the download delay of a single file was analyzed by considering a simpler cell model. Here, as opposed to [13], we analyze a cluster-based cell model and consider that users may request files, of different popularity, from a library of files.

The remainder of the paper is organized as follows. The system model is introduced in Section II. The average file download delay incurred when MDS-coded DC is used is analyzed in Sections III and Section IV. Section V presents and discusses numerical results and finally some conclusions are drawn in Section VI.

Notation. The probability density function (pdf) of a random variable X is denoted by $f_X(\cdot)$ and the expectation with respect to X is denoted by $\mathbb{E}_X\{\cdot\}$. We denote by $\mathbb{N}_0 = \{0, \ldots, \infty\}$ the set of natural numbers. Probability is denoted by $\Pr\{\cdot\}$ and $\mathbf{1}_i$ represents a length-i vector of all ones. We denote by $\pi_m(\rho)$ the stationary distribution of an $\mathsf{M}/\mathsf{M}/\infty$ queueing system described by a Poisson birth-death process with arrival rate α and service rate per node δ; the probability that the system is in state m is [14]

$$\pi_m(\rho) = \frac{\rho^m}{m!} e^{-\rho}, \qquad m \geq 0\,, \tag{1}$$

where $\rho = \alpha/\delta$.

II. System Model

We consider a single cell in a cellular network where M mobile devices, referred to as nodes, can request files, each of size B bits, from a library of Z files. The files have different popularities and accordingly have a given probability to be requested. Depending on the placement strategy, some files are encoded and cached into a number of mobile devices, referred to as caching nodes, as described in detail in Section II-A below. A copy of each encoded file is also available at the BS serving the cell. A node requesting a file attempts to retrieve it from the caching nodes using D2D communication, and, if the file cannot be completely retrieved from the mobile devices, the BS assists in providing the missing data.

A. Data placement and Coding Strategy

We adopt a *uniform* content placement strategy where the $F \leq Z$ most popular files are cached in a distributed fashion in $n \leq M$ caching nodes in the cell using a single erasure correcting code. In particular, these files are partitioned into k packets, called symbols, of B/k bits each and encoded into n coded symbols using an (n, k) MDS erasure correcting code of rate $r = k/n$. Thanks to the MDS property, an encoded file can be reconstructed by accessing any k of the n encoded symbols. For each file, the n coded symbols are stored in n caching nodes chosen uniformly at random among the M mobile devices. For ease of language, for a given file, the set of nodes caching the file will be referred to as the DC network, and the nodes not caching the file will be referred to as *regular* nodes. Note that due to the random caching strategy, the DC network (and thus the set of regular nodes) may be different for different files. Overall, nF symbols are stored across the mobile devices and no two caching nodes store the same symbol. We model the popularity of the files in the library using the time-invariant Zipf distribution [15].[1] Accordingly, the probability that the ith file is requested is

$$z_i = \frac{1/i^\sigma}{\sum_{j=1}^{Z} 1/j^\sigma}\,, \qquad 1 \leq i \leq Z\,, \tag{2}$$

where parameter σ regulates the relative popularity of the files. In the following, the set of F files cached in the cache of the mobile devices will be referred to as the DC library.

While in the analysis in Sections III and IV we assume uniform content placement, the analysis applies in a straightforward manner to the more general case where files are cached using erasure correcting codes of different length and rate. In Section V, we give results for a *proportional* content placement, where the length and rate of the caching codes are selected according to the popularity of the files.

B. Network model

In order to increase the system efficiency, we allow multiple D2D communications to coexist if they are sufficiently far apart in space. Therefore, we divide the cell in C virtual clusters and assume that the size of the cluster and the transmit power are properly chosen such that every pair of nodes in the cluster can potentially communicate through a D2D link. A similar model is considered in [2], [5]. We allow only one D2D communication at a time in each cluster in order to avoid intra-cluster interference.[2] We assume that the D2D communication does not interfere with the communication between the BS and the mobile devices. This can be achieved by allowing overlay inband D2D links, i.e., part of the cellular spectrum is dedicated only to D2D communications [16], or outband D2D links, where D2D communication occurs in unlicensed spectrum [17]. The reader is referred to [18] for a tutorial discussion about this topic. Finally, we assume that multiple BS-to-node links can coexist as in classical cellular communications. We account for the inter-cluster interference by considering that one cluster must compete with the adjacent ones in order to use the resource devoted to D2D communication. In Fig. 1, we show an example of a cell composed by hexagonal clusters. In the case of a single frequency band for D2D communication in all clusters, a requesting user in the reference cluster (the blue one in the figure) can find the network *idle* if there is no active D2D communication in the reference cluster nor in the six adjacent ones. On the other hand, the inter-cluster interference can be avoided by using, for example, separate frequency bands for adjacent clusters. In the considered example with hexagonal clusters, it is easy to

[1] The popularity of the files in mobile data traffic does not change very rapidly, i.e., it can be considered constant during the day.

[2] Better results could be achieved by allowing multiple D2D communications to coexist in a cluster, but this would require a more involved communication scheme to avoid the interference among devices.

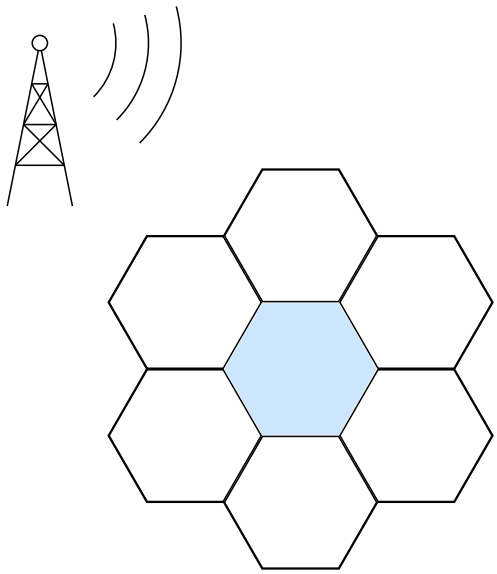

Figure 1. Part of a cell with hexagonal clusters. If we assume that all clusters employ the same frequency band for D2D communication, seven clusters must compete for accessing the DC network.

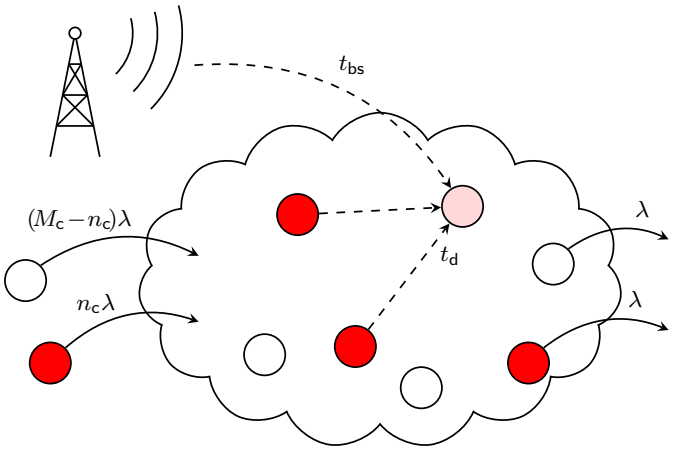

Figure 2. An example of cluster where nodes roam in and out according to a Poisson random process: we have on average M_{c} mobile devices, and n_{c} caching nodes among them (red circles), caching one different coded symbol for one of the most popular files. A device requesting the file (pink circle), must collect k symbols. It attempts to recover them by using the DC network and uses the BS to collect the symbols that it is not able to download from the devices. The download of a symbol from a caching node takes t_{d} t.u., and from the BS t_{bs} t.u..

show that only four orthogonal frequency bands are sufficient to avoid the interference.

C. Arrival-Departure Model

We consider a uniform spatial distribution of the nodes in the cell, and hence there are $M_{\mathsf{c}} = M/C$ devices per cluster on average and among them $n_{\mathsf{c}} = n/C$ caching nodes for a given file of the DC library. We focus on a single cluster in isolation. Mobile devices move inside the cell and thus roam in and out of the cluster. The arrival, departure and request model of the nodes are borrowed from [10] and are shown schematically in Fig. 2. In particular, nodes arrive to the cluster according to a Poisson random process with independent, identically distributed (i.i.d.) exponential random inter-arrival times T_{a} with pdf

$$f_{T_{\mathsf{a}}}(t) = M_{\mathsf{c}}\lambda e^{-M_{\mathsf{c}}\lambda t}, \qquad t \geq 0, \tag{3}$$

where $M_{\mathsf{c}}\lambda$ is the expected arrival rate of a node. We assume that the nodes stay in the cluster for an i.i.d exponential random lifetime T_{f} with pdf

$$f_{T_{\mathsf{f}}}(t) = \lambda e^{-\lambda t}, \qquad t \geq 0, \tag{4}$$

where $\lambda \geq 0$ is the expected departure rate per node and t is time, measured in time units (t.u.). This implies that the expected number of nodes in the cluster is M_{c}. The model corresponds to an $\mathsf{M}/\mathsf{M}/\infty$ queuing model and the probability that there are i nodes in the cluster is $\pi_i(M_{\mathsf{c}})$, defined in (1). The arrival of nodes caching a particular file of the DC library to the cluster can also be described as a Poisson random process. In particular, the inter-arrival times T_{s} of the set of caching nodes has pdf

$$f_{T_{\mathsf{s}}}(t) = n_{\mathsf{c}}\lambda e^{-n_{\mathsf{c}}\lambda t}, \qquad t \geq 0\,,$$

where $n_{\mathsf{c}}\lambda$ is the expected arrival rate of a caching node. The related lifetime is described by (4) and the probability that there are i caching nodes for a file in the cluster is $\pi_i(n_{\mathsf{c}})$.[3]

D. DC Network Update

We consider a network where the D2D communications are controlled by the BS. We assume that the nodes caching content that arrive to the cluster from neighboring clusters are not immediately available for download, but the BS serving the cell keeps track of them and periodically updates and broadcasts to all mobile devices the list of caching nodes in the cell every Δ t.u.. In the sequel, parameter Δ is referred to as the update interval and the set of caching nodes in the list broadcasted to the devices in the cluster by the BS as the DC list.

E. Data Delivery

Nodes request the file at random times with i.i.d. random inter-request time T_{r} with pdf

$$f_{T_{\mathsf{r}}}(t) = \omega e^{-\omega t}, \qquad \omega \geq 0, t \geq 0, \tag{5}$$

where ω is the expected request rate per node. We focus on the download process. If the requesting node of a given file is a caching node of the DC list for that file, it needs to download $k-1$ symbols, otherwise k symbols must be downloaded.

[3]The Poisson model is largely used in the case of uniform mobility and its popularity is also due to its tractability. However, we would like to remark that while it is able to capture the mobility in one cluster, this model does not guarantee that the total number of caching nodes for a file in the cell is constant and equal to n. More precisely, the only guarantee is that there are on average n_{c} caching nodes of a file per cluster, but there are no constraints on their instantaneous number, which can even exceed n. On the other hand, the probability of having a high number of caching devices in one cluster is generally very low. For $n_{\mathsf{c}} = 9$, we have $\pi_{18}(n_{\mathsf{c}}) = 3 \cdot 10^{-3}$, $\pi_{27}(n_{\mathsf{c}}) = 6.6 \cdot 10^{-7}$ and $\pi_{91}(n_{\mathsf{c}}) = 4 \cdot 10^{-48}$. The same consideration holds for the total number of mobile devices.

The node that requests a file attempts to retrieve it from the DC network using D2D communication. If the file cannot be retrieved from the DC network, the BS assists in providing the missing content. In particular, we consider that the download of a coded symbol from a caching node incurs t_{d} t.u.. Furthermore, the download of a symbol of a file of the DC library from the BS takes t_{bs} t.u.. In this case, only the cellular link is involved, since a copy of each encoded file is available at the BS. Finally, files that do not belong to the DC library are entirely downloaded from the BS in T_{bh} t.u.. In this case, the BS obtains the requested file from the core network through the backhaul link and transmits it to the requesting user through the cellular link.

We assume that a device cannot download in parallel from multiple caching nodes, but it serially tries to download the coded file symbols from the nodes in the DC list.[4] When a node requests the file, if the network is idle and the requested file belongs to the DC library, it randomly chooses one of the caching nodes from the list supplied by the BS present in the cluster (if any). After each downloaded symbol, the requesting node randomly chooses another caching node from the DC list and still alive.[5]

If the network is not idle, or if the download from the caching nodes can be only partially accomplished, the requesting node turns to the BS to collect the missing data. For the latter, we assume that a requesting node that has collected fewer than the k symbols necessary to reconstruct the file turns to the BS when all the reference caching nodes left from the cluster or when the download of a symbol fails, even if other caching nodes are available. To simplify the analysis, we assume that both cases (the failed symbol download and the absence of caching nodes) incur t_{d} t.u., even if the node could contact the BS earlier. Note that the download of a symbol through D2D communication fails if one of the two involved nodes leave the cluster in the middle of the download or due to link failure. In the case where D2D communication fails because the requesting node itself leaves the cluster, it could potentially download symbols from nodes in the new cluster. However, to make the analysis tractable, we do not consider this possibility and assume that download is completed from the BS. This is a negative assumption that reduces the benefits of DC.

We model the channel for D2D communication as a block-erasure channel with erasure probability ε, which is a special case of the block-fading channel, where transmission can be either totally lost, due to a deep fade, or noiseless. Regarding the download from the BS, we assume that t_{bs} and T_{bh} are average download times that account for the average number of necessary retransmissions.

[4]A more involved transmission/detection strategy would be required to allow a node to download in parallel from multiple devices. If multiple transmissions in parallel were possible, this would make our scheme even better, in the sense that the obtained gains would be higher.

[5]The requesting node uses the caching nodes alive at the moment of its request even if, during the download process, new caching nodes are included in the DC list of the cluster after the periodic restoration.

III. Average File Download Delay

In this section, we investigate the average time that is required to retrieve one file from the wireless network, referred to as the average file download delay. If a requested file is cached in the DC library, the requesting node attempts to retrieve it from the DC network using D2D communication. The network can be used only if it is idle, i.e., if there are no active D2D communications in the concurrent clusters.

In order to compute the average file download delay, it is useful to define the following binary random variables (RVs). We introduce the RV $H \in \{0,1\}$ which describes the hitting of the DC cache, i.e., $H = 1$ if a file of the DC library is requested and $H = 0$ otherwise. We use the RV $I \in \{0,1\}$ to describe the status of the DC network, i.e., $I = 1$ if the network is idle and $I = 0$ otherwise. Finally, we introduce the RV $R \in \{0,1\}$, which represents the type of request, i.e., $R = 1$ for requests originating from a node that is a caching node of the DC list for the requested file and $R = 0$ for the other requests. We define η as the average number of coded symbols downloaded per request using D2D communication and $\overline{T}_{\eta}$, referred to as the average D2D download delay, as the corresponding delay. From the discussion above, the average file download delay, $\overline{T}_{\mathsf{dw}}$, may be formalized as follows.

Proposition 1. *The average file download for the cellular network described in Section II where the F most popular files are cached in the mobile devices using an (n,k) MDS code is*

$$\begin{aligned}\overline{T}_{\mathsf{dw}} &= \Pr\{H=0\}T_{\mathsf{bh}} \\ &\quad + \Pr\{I=1\}\Pr\{H=1\}\Big(\overline{T}_{\eta} + (k - \Pr\{R=1\} - \eta)t_{\mathsf{bs}}\Big) \\ &\quad + \Pr\{I=0\}\Pr\{H=1\}(k - \Pr\{R=1\})t_{\mathsf{bs}}\,. \end{aligned} \tag{6}$$

Proof: When one node requests a file, we can distinguish the following three cases

- The file does not belong to the DC library. This happens with probability $\Pr\{H = 0\}$ and the file is entirely downloaded from the BS, involving the backhaul and cellular links. The download takes on average T_{bh} t.u..
- The file belongs to the DC library and the DC network is idle. This happens with probability $\Pr\{I = 1\}\Pr\{H = 1\}$. The requesting node attempts to retrieve it from the DC network using D2D communication for an average time $\overline{T}_{\eta}$, and turns to the BS to collect the missing data. With probability $\Pr\{R = 1\}$, the requesting node is a caching node for the file, and it needs to download $k-1$ symbols. Otherwise, k symbols must be downloaded. Therefore, the average number of missing symbols is $k - \Pr\{R = 1\} - \eta$. The download of a symbol of a file of the DC library from the BS takes on average t_{bs} t.u. and involves only the cellular link since a copy of each encoded file is available at the BS.
- The requested file belongs to the DC library and the DC network is not idle. This happens with probability $\Pr\{I = 0\}\Pr\{H = 1\}$. In this case, the download is served by the BS from the cellular link, and takes on average $(k - \Pr\{R = 1\})t_{\mathsf{bs}}$.

■

The computation of η, $\overline{T}_\eta$ and $\Pr\{R=1\}$ is addressed in Section IV. The probability of hitting the cache depends on the popularity distribution of the files since, according to the considered placement policy, the DC network caches the F most popular files. It can be expressed as

$$\Pr\{H=1\}=\sum_{i=1}^{F} z_i\,,$$

where the probabilities z_i are given in (2). It follows that $\Pr\{H=1\}=1$ if $F=Z$.

The next step is the computation of the probability that the DC network is idle. As discussed in Section II, the reference cluster must possibly compete with the adjacent ones that use the same frequency band for D2D communication in order to access the DC network. We therefore define κ as the number of concurrent clusters. For example, for the case of hexagonal clusters shown in Fig. 1, $\kappa=1$ if we use orthogonal bands for adjacent clusters and $\kappa=7$ if the neighboring clusters must compete for the same time-frequency resource. Let $I^{(\ell)}$ be the status of the network at the time of the ℓth request in the group of κ clusters. It follows that

$$\Pr\{I=1\}=\lim_{L\to\infty}\frac{1}{L}\sum_{\ell=1}^{L}\Pr\{I^{(\ell)}=1\}. \tag{7}$$

In order to compute $\Pr\{I^{(\ell)}=1\}$, we introduce the RV $W^{(j)}$ that denotes the time instant of the jth request. Also, let $T^{(j)}$ be the time during which the DC network is occupied by the jth request. The DC network is idle at the time of the ℓth request if none of the previous requests is still using D2D communication. Therefore, $\Pr\{I^{(1)}=1\}=1$ and

$$\Pr\{I^{(\ell)}=1\}=\prod_{i<\ell}\Pr\{W^{(\ell)}>W^{(\ell-i)}+T^{(\ell-i)}\},\ \ell>1. \tag{8}$$

Assuming that if the DC network is not idle at time $W^{(\ell)}$ is because of the $(\ell-1)$th request, the product in (8) reduces to the term involving the $(\ell-1)$th request only, i.e.,

$$\begin{aligned}\Pr\{I^{(\ell)}=1\}&\simeq\Pr\{W^{(\ell)}>W^{(\ell-1)}+T^{(\ell-1)}\} \qquad (9)\\ &=\int_0^\infty \Pr\{W^{(\ell)}>W^{(\ell-1)}+t\}f_{T^{(\ell-1)}}(t)dt\,.\end{aligned}$$

Since the requests are i.i.d. with inter-request time distributed as in (5) and on average there are M_{c} nodes in each of the κ concurrent clusters, we can compute

$$\Pr\{W^{(\ell)}>W^{(\ell-1)}+t\}=e^{-\omega\kappa M_{\mathsf{c}}t},\quad t\geqslant 0,\quad \ell>1\,,$$

and (9) can be written as

$$\Pr\{I^{(\ell)}=1\}\simeq\int_0^\infty e^{-\omega\kappa M_{\mathsf{c}}t}f_{T^{(\ell-1)}}(t)dt,\quad \ell>1. \tag{10}$$

As we will see in the next section, in order to avoid network congestion, we do not allow a requesting node to occupy the DC network for long time. For example, we assume that the requesting node turns to the BS if the download of a symbol fails even if there are other caching nodes in its vicinity. For this reason, the we can assume that the pdf $f_{T^{(\ell-1)}}(t)$ is small for large t. Therefore, we use the approximation $e^x\simeq 1-x$ that is valid for small x in equation (10) and we obtain

$$\begin{aligned}\Pr\{I^{(\ell)}=1\}&\simeq\int_0^\infty(1-\omega\kappa M_{\mathsf{c}}t)f_{T^{(\ell-1)}}(t)dt\\ &=1-\int_0^\infty\omega\kappa M_{\mathsf{c}}tf_{T^{(\ell-1)}}(t)dt\\ &=1-\omega\kappa M_{\mathsf{c}}\mathbb{E}\{T^{(\ell-1)}\}\\ &=1-\omega\kappa M_{\mathsf{c}}\Pr\{I^{(\ell-1)}=1\}\Pr\{H=1\}\overline{T}_\eta.\end{aligned} \tag{11}$$

In (11), we used the fact that the time spent in downloading from the DC network by the $(\ell-1)$th request is non zero only when the DC network is idle and the requesting file belongs to the DC library. We further assume that we can ignore boundary effects, i.e., the clusters in the group are not at the edge of the cell. We have then used the fact that the probability of hitting the cache is independent of the request index, as well as the average D2D download delay (if ℓ is sufficiently large), as it is proven in Corollary 1, in Section IV. It can be verified numerically that $\Pr\{I^{(\ell)}=1\}$ does not depend on the request index already for small values of ℓ. We now define $\xi\triangleq\omega\kappa M_{\mathsf{c}}\Pr\{H=1\}\overline{T}_\eta$. Assuming that the average D2D download delay is independent of the request index for any ℓ, it follows

$$\Pr\{I^{(\ell)}=1\}\simeq\sum_{j=0}^{\ell-1}(-1)^j\xi^j\,. \tag{12}$$

Let $I^{(\infty)}=\lim_{\ell\to\infty}I^{(\ell)}$. Recalling that we assume $\xi<1$, we obtain

$$\Pr\{I^{(\infty)}=1\}\simeq\frac{1}{1+\xi}\,. \tag{13}$$

Using (12) and (13) in (7), we finally obtain the following approximation for the probability that the DC network is idle,

$$\Pr\{I=1\}\simeq\frac{1}{1+\omega\kappa M_{\mathsf{c}}\Pr\{H=1\}\overline{T}_\eta}\,. \tag{14}$$

Equations (14) and (6) reveal an interesting trade-off. If from one hand an increase of the probability of hitting the cache increases the probability of avoiding the use of the backhaul link, on the other hand it causes an increase of the D2D traffic, that is the decrease of the probability of finding the DC network idle.

IV. Download From Caching Nodes

In this section, we address the computation of the average D2D download delay $\overline{T}_\eta$ and the average number of coded symbols η downloaded per request using D2D communication. To derive the average D2D download delay, we introduce three RVs describing the number of nodes of different type that are present in the cluster at the instant of the request of a file: with reference to the requested file, we define the number of caching nodes of the DC list, the total number of caching nodes (belonging or not to the list, the latter corresponding to the caching nodes that arrive to the cluster after the DC list update and that have not left the cluster at the time of the request), and the number of regular nodes , i.e., not caching

any symbol of the file. In particular, we denote by $X_1 \in \mathbb{N}_0$ the RV that describes the number of caching nodes in the cluster for the requested file of the DC list when the request arrives. We describe by the RVs $Q \in \mathbb{N}_0$ and $V \in \mathbb{N}_0$ the total number of caching nodes for the requested file and the number of regular nodes present in the cluster at the instant of a request, respectively. Moreover, we denote by $Y \in \mathbb{N}_0$ the RV that represents the number of nodes caching the requested file present in the cluster at the beginning of the update interval of length Δ. In the following three lemmas, we give a probabilistic description of the above RVs.

Lemma 1. *The probability that there are $x \geq 0$ caching nodes of the DC list for the requested file at the time of the request is*

$$\Pr\{X_1 = x\} = \frac{\sum_{y=0}^{\infty}\pi_y(n_{\mathsf{c}})\sum_{m=y}^{\infty}(1-e^{-m\omega\Delta})\pi_{m-y}(M_{\mathsf{c}}-n_{\mathsf{c}})\Pr\{X_1=x|Y=y\}}{\sum_{m=1}^{\infty}(1-e^{-m\omega\Delta})\pi_m(M_{\mathsf{c}})}, \tag{15}$$

where $\Pr\{X_1 = x|Y = y\}$ is the probability that X_1 is equal to x, given that $y \geq 0$ caching nodes for the file are in the cluster at the beginning of the update interval of length Δ, and is

$$\Pr\{X_1 = x|Y = y\} = \frac{1}{\Delta}\sum_{i'=x}^{y}\frac{1-p_{i'}}{\lambda_{i'}}\prod_{\substack{j=x\\ j\neq i'}}^{y}\frac{j}{j-i'} - \frac{1}{\Delta}\sum_{i'=x+1}^{y}\frac{1-p_{i'}}{\lambda_{i'}}\prod_{\substack{j=x+1\\ j\neq i'}}^{y}\frac{j}{j-i'}, \tag{16}$$

where $\lambda_{i'} = i'\lambda$ and $p_{i'} = e^{-\lambda_{i'}\Delta}$.

Proof. The proof is given in Appendix A. □

Lemma 2. *The probability that there are $q \geq 0$ caching nodes for the requested file in the cluster at the time of a request is given by*

$$\Pr\{Q=q\} = \frac{\sum_{m=q}^{\infty}(1-e^{-m\omega\Delta})\pi_{m-q}(M_{\mathsf{c}}-n_{\mathsf{c}})}{\sum_{m=1}^{\infty}(1-e^{-m\omega\Delta})\pi_m(M_{\mathsf{c}})}\pi_q(n_{\mathsf{c}}). \tag{17}$$

Proof. The proof follows the same lines as the proof of Lemma 1. □

Lemma 3. *The probability that there are $v \geq 0$ regular nodes in the cluster at the time of a request is given by*

$$\Pr\{V=v\} = \frac{\sum_{m=v}^{\infty}(1-e^{-m\omega\Delta})\pi_{m-v}(n_{\mathsf{c}})}{\sum_{m=1}^{\infty}(1-e^{-m\omega\Delta})\pi_m(M_{\mathsf{c}})}\pi_v(M_{\mathsf{c}}-n_{\mathsf{c}}). \tag{18}$$

Proof. The proof follows the same lines as the proof of Lemma 1. □

Based on the above lemmas, we can compute the probability that the request originates from a caching node for the requested file of the DC list and the probability of having a given number of caching nodes for the requested file in the DC list at the time of the request conditioned to the type of request.

Using Bayes' rule, the probability that there are $x \geq 0$ caching node for the requested file of the DC list alive at the time of the request, conditioned to the type of request, is given by

$$\Pr\{X_1=x|R=i\} = \Pr\{R=i|X_1=x\}\frac{\Pr\{X_1=x\}}{\Pr\{R=i\}}, i=0,1. \tag{19}$$

The probability $\Pr\{X_1 = x\}$ is given in Lemma 1. We now compute $\Pr\{R|X_1\}$ and $\Pr\{R\}$. With reference to the requested file, we compute the probability of having one request from the DC list conditioned to the number of caching nodes in the DC list at the time of the request. For $x > 0$, it can be written as

$$\Pr\{R=1|X_1=x\} = \sum_{v=0}^{\infty}\sum_{q=x}^{\infty}\frac{x}{q+v}\Pr\{V=v|Q=q,X_1=x\}\Pr\{Q=q|X_1=x\}.$$

Clearly, the condition $X_1 = 0$ implies that the request cannot originate from a caching node of the DC list, therefore $\Pr\{R = 1|X_1 = 0\} = 0$. We approximate the probability that there are q caching nodes of the requested file at the instant of the request, given the number of caching nodes for that file of the DC list, by using the steady state probability of a Poisson birth-death process with arrival rate $\lambda(\mathbb{E}_Q(Q)-\mathbb{E}_{X_1}(X_1))$ and departure rate λ. In particular, we compute

$$\Pr\{Q = q|X_1 = x\} \simeq \pi_{q-x}(\mathbb{E}_Q(Q) - \mathbb{E}_{X_1}(X_1)), \tag{20}$$

where the expectations $\mathbb{E}_Q(Q)$ and $\mathbb{E}_{X_1}(X_1)$ are obtained starting from the probabilities (17) and (15), respectively. The number of regular nodes V is independent of the number of caching nodes at the instant of the request, therefore we finally have

$$\Pr\{R = 1|X_1 = x\} = \sum_{q=x}^{\infty}\sum_{v=0}^{\infty}\frac{x}{q+v}\Pr\{V=v\}\pi_{q-x}(\mathbb{E}_Q(Q)-\mathbb{E}_{X_1}(X_1)),$$

where $\Pr\{V = v\}$ is given in Lemma 3. Note that in the expression above, with some abuse of notation, we used equal sign to avoid carrying all the way the approximation sign due to the approximation introduced in (20). Starting from this result, we compute the probability that the request originates from the DC list as

$$\Pr\{R = 1\} = \sum_{x=0}^{\infty}\sum_{q=x}^{\infty}\sum_{v=0}^{\infty}\frac{x}{q+v}\Pr\{V=v\}\pi_{q-x}(\mathbb{E}_Q(Q)-\mathbb{E}_{X_1}(X_1))\Pr\{X_1=x\}$$

where $\Pr\{X_1 = x\}$ is given in Lemma 1. The probability $\Pr\{R = 0|X_1 = x\}$ is easily computed as $1 - \Pr\{R = 1|X_1 = x\}$. Similarly, we have $\Pr\{R = 0\} = 1-\Pr\{R = 1\}$.

Following the same approach for the proof of Lemma 1, it is easy to show that $\Pr\{X_1^{(\ell)}=x|R^{(\ell)}=i\}$ and $\Pr\{R^{(\ell)}=i\}$ are independent of the specific request (when ℓ grows large), where $R^{(\ell)}$ is the binary RV describing the type of the ℓth request.

The case $\Delta = 0$ represents the case of instantaneous update, where the nodes contact directly the BS when they request a file and receive the list of the caching nodes through a dedicated link. For instantaneous update, the number of caching nodes at the instant of the request and the type of request is described by the following probabilities,

$$\begin{aligned}&\Pr\{X=x\}=\pi_x(n_{\mathsf{c}})\,,\\&\Pr\{R=1\}=\frac{n_{\mathsf{c}}}{M_{\mathsf{c}}}\,,\\&\Pr\{R=1|X=x\}=\sum_{m=x}^{\infty}\frac{x}{m}\pi_{m-x}(M_{\mathsf{c}}-n_{\mathsf{c}})\,.\end{aligned}$$

The probability that there are $x\geq 0$ caching nodes at the time of the request given the type of request can be computed by replacing the above probabilities in (19).

In order to describe the D2D download, let S_1 be the binary RV that describes the success of the download at the first attempt. More precisely, $S_1=1$ represents the successful download of the coded symbol from the first contacted caching node. If the download is not successful from the first contacted caching node, $S_1=0$. Similarly, we denote by S_j the binary RV describing the download at the jth attempt and we denote by $\mathbf{S}_{[i]}$, $i\geq 1$ the random vector $(S_1,...,S_i)$. In the following, in Lemmas 4, 5, and 6, we derive the probability that no symbols can be downloaded from the DC network, $\Pr\{S_1=0|R\}$, the probability that the content is fully recovered from the DC network, $\Pr\{\mathbf{S}_{[k-i]}=\mathbf{1}_{k-i}|R\}$, and the probability that it is only partially recovered, $\Pr\{\mathbf{S}_{[j]}=\mathbf{1}_j,S_{j+1}=0|R\}$, respectively.

Lemma 4. *The probability that no symbols are downloaded through D2D communication, conditioned to the type of request, is given by*

$$\begin{aligned}\Pr\{S_1=0|R=i\}=&1+e^{-\lambda t_{\mathsf{d}}}\Big(\Pr\{X_1=i|R=i\}\\&+\sum_{g=1}^{\infty}\sum_{d=0}^{g}\Big(\frac{d}{g}+\varepsilon\frac{g-d}{g}\Big)\Pr\{X_1=g+i|R=i\}\theta(d,g)-1\Big),\ i=0,1\end{aligned}$$

where $\Pr\{X_1|R\}$ *is given in (19) and*

$$\begin{aligned}\theta(d,g)=&\sum_{i'=g-d}^{g}e^{-\lambda_{i'}t_{\mathsf{d}}}\prod_{\substack{j=g-d\\j\neq i'}}^{g}\frac{j}{j-i'}\\&-\sum_{i'=g-d+1}^{g}e^{-\lambda_{i'}t_{\mathsf{d}}}\prod_{\substack{j=g-d+1\\j\neq i'}}^{g}\frac{j}{j-i'},\quad d\geq 0,g\geq 0\end{aligned}\tag{21}$$

with $\lambda_{i'}=i'\lambda$.

Proof. The proof is given in Appendix B. □

Lemma 5. *The probability that the file can be completely retrieved from the DC network, i.e., the probability that k symbols are downloaded through D2D communication when $R=0$, or $k-1$ when $R=1$, conditioned to the type of request, is given by*

$$\begin{aligned}&\Pr\{\mathbf{S}_{[k-i]}=\mathbf{1}_{k-i}|R=i\}=\\&e^{-(k-i)\lambda t_{\mathsf{d}}}(1-\varepsilon)\sum_{g=1}^{\infty}\sum_{d=0}^{g}\frac{g-d}{g}\gamma_{k-i}(g,d,i),i=0,1,\end{aligned}\tag{22}$$

where $\gamma_j(g,d,i)$ *is defined by the recursion*

$$\gamma_j(g,d,i)=\theta(d,g)(1-\varepsilon)\sum_{g'=1}^{\infty}\sum_{d'=0}^{g'-1}\frac{g'-d'}{g'}E(g,g',d')\gamma_{j-1}(g',d',i)\tag{23}$$

for $d,g\geq 0$ *and* $i=0,1$*, with initial condition*

$$\gamma_1(g,d,i)=\Pr\{X_1=g+i|R=i\}\theta(d,g)\,,\tag{24}$$

and where

$$E(g,g',d')=\begin{cases}1 & \text{if}\quad g=g'-d'-1\\0 & \text{otherwise}\,.\end{cases}$$

The function $\theta(d,g)$ *is given in (21), and* $\Pr\{X|R\}$ *is given in (19).*

Proof. The proof is given in Appendix C. □

Lemma 6. *The probability of consecutively download $j\geq 1$ symbols and to fail the download of the $j+1$th one is*

$$\begin{aligned}&\Pr\{\mathbf{S}_{[j]}=\mathbf{1}_j,S_{j+1}=0|R=i\}=\gamma_{j+1}(0,0,i)a_{j+1}\\&+\sum_{g=1}^{\infty}\sum_{d=0}^{g}\Big(\frac{d}{g}+\varepsilon\frac{g-d}{g}\Big)\gamma_{j+1}(g,d,i)a_{j+1}\\&+(1-\varepsilon)\sum_{g=1}^{\infty}\sum_{d=0}^{g}\frac{g-d}{g}\gamma_j(g,d,i)b_{j+1}\,,\end{aligned}$$

where $a_j=e^{-j\lambda t_{\mathsf{d}}}$, $b_j=e^{-(j-1)\lambda t_{\mathsf{d}}}(1-e^{-\lambda t_{\mathsf{d}}})$, *and* $\gamma_j(g,d,i)$ *is given in (23).*

Proof. The proof follows the same lines as the proof of Lemma 5. □

Finally, the average D2D download delay and the average number of downloaded symbols from the DC network are given in the following theorem.

Theorem 1. *Consider the network described in Section II, where an (n,k) MDS erasure correcting code is employed and where in the cluster there are on average n_{c} caching nodes for each cached file. Let t_{d} be the time to download a symbol through D2D communication. The average D2D download delay and the corresponding average number of downloaded symbols are given by*

$$\begin{aligned}\overline{T}_\eta=&\overline{T}_1p_{\mathsf{s}}+\overline{T}_0(1-p_{\mathsf{s}})\\\eta=&\eta_1p_{\mathsf{s}}+\eta_0(1-p_{\mathsf{s}})\end{aligned}$$

where $p_{\mathsf{s}} = \Pr\{R = 1\}$ is the probability that the request comes from a caching node of the DC list, and

$$\eta_i = (k-i)\Pr\{\mathbf{S}_{[k-i]} = \mathbf{1}_{k-i} | R = i\} + \sum_{j=1}^{k-1-i} j \Pr\{\mathbf{S}_{[j]} = \mathbf{1}_j, S_{j+1} = 0 | R = i\},$$

$$\overline{T}_i = t_{\mathsf{d}} \Big(\eta_i + c_{k,i} \Pr\{ S_1 = 0 | R = i\} + \sum_{j=0}^{k-1-i} \Pr\{\mathbf{S}_{[j]} = \mathbf{1}_j, S_{j+1} = 0 | R = i\} \Big), \qquad i = 0, 1$$

where $c_{k,i} = 1$ for $k - i > 0$ and $c_{k,i} = 0$ otherwise.

Proof. The average D2D download delay is obtained as the sum of the average D2D delays in the case of requests originated from nodes of the DC list caching symbols of the requested file and of requests originating from the other nodes, weighted by the probabilities p_{s} and $1-p_{\mathsf{s}}$, respectively. The same approach is used for the corresponding average number of downloaded symbols. According to our model, the requesting node completes the download of $k-i$ symbols from the DC network in $(k-i)t_{\mathsf{d}}$ t.u. with probability $\Pr\{\mathbf{S}_{[k-i]} = \mathbf{1}_{k-i} | R = i\}$, while the partial download of $j < k-i$ symbols happens with probability $\Pr\{\mathbf{S}_{[j]} = \mathbf{1}_j, S_{j+1} = 0 | R = i\}$ and incurs $(j+1)t_{\mathsf{d}}$ t.u.. For $k - i > 0$, in the computation of the average D2D download delay, we also consider the case where download from the DC network completely fails. The corresponding probability is $\Pr\{S_1 = 0 | R = i\}$ and the delay is t_{d}. When the request originates from the nodes of the DC list caching the requested file and $k = 1$, i.e., $k = i$, no symbols need to be downloaded, therefore $\overline{T}_1$ and η_1 are equal to zero. □

Corollary 1. *The average D2D download time for the ℓth request, $\overline{T}_{\eta}^{(\ell)}$, is independent of the specific request if the index ℓ is sufficiently large.*

Proof. Similarly to the average D2D download delay, $\overline{T}_{\eta}^{(\ell)}$ is

$$\overline{T}_{\eta}^{(\ell)} = \overline{T}_1^{(\ell)} \Pr\{R^{(\ell)} = 1\} + \overline{T}_0^{(\ell)} \Pr\{R^{(\ell)} = 0\},$$

where

$$\overline{T}_i^{(\ell)} = t_{\mathsf{d}} \Big((k-i) \Pr\{\mathbf{S}_{[k-i]}^{(\ell)} = \mathbf{1}_{k-i} | R^{(\ell)} = i\} + c_{k,i} \Pr\{ S_1^{(\ell)} = 0 | R^{(\ell)} = i\} + \sum_{j=0}^{k-1-i} (j+1) \Pr\{\mathbf{S}_{[j]}^{(\ell)} = \mathbf{1}_j, S_{j+1}^{(\ell)} = 0 | R^{(\ell)} = i\} \Big), i = 0,1$$

The Lemma follows from the fact that the probabilities in the expressions above are independent of ℓ, when ℓ grows large. □

V. Numerical Results

In this section, we evaluate the average file download delay when content is cached using MDS codes for a cluster with $M_{\mathsf{c}} = 30$ nodes on average, departure rate $\lambda = 1$, and request rate $\omega = 0.02$. According to the chosen parameters, the average lifetime of a node in the cluster is 1 t.u. and the probability that one node places more than one request in this time is low and equals the probability that the inter-request time T_{r}, distributed as in (5), is smaller that 1 t.u., that is

$$\Pr\{T_{\mathsf{r}} < 1\} = 1 - e^{-\omega} \simeq 0.02.$$

The infinite series involved in the computation of $\overline{T}_{\mathsf{dw}}$ are truncated to a given value t, chosen according to $argmin_{t>n_{\mathsf{c}}}\{\pi_t(n_{\mathsf{c}}) < 10^{-5}\}$ when involving the number of caching nodes and to $\text{argmin}_{t>M_{\mathsf{c}}}\{\pi_t(M_{\mathsf{c}}) < 10^{-5}\}$ when involving the number of nodes in general. We remark that the results depend on the parameter C through M_{c} and n_{c}.

Unless otherwise stated, we consider a uniform content placement (see Section II-A). The code parameters are chosen such that $k \leq n_{\mathsf{c}}$. In this way, the *average storage overhead* in a cluster, $n_{\mathsf{c}} - k$, is positive, which increases the probability that the content is downloaded through D2D communication only. Alternatively, we can use the same (n_{c}, k) MDS code for each cluster, but in this case the BS must continuously restore the initial state of reliability of the DC network when caching nodes leave the clusters [10]. In the following, we consider several MDS codes and an uncoded scenario where one caching node on average in the cluster stores the uncoded files. The system parameters used in the figures are summarized in Tables I and II.

Table I
System parameters common to Figs 3–9

M_{c}	λ	ω	$T_{\mathsf{bh}}/T_{\mathsf{bs}}$
30	1	0.02	2

Table II
Fixed system parameters used in Figs 3–8.

Fig.	Z/F	ε	κ	$t_{\mathsf{bs}}/t_{\mathsf{d}}$	Δ	σ
3	1	0	1	10		
4	1	0	1	100		
5	1	0	1	1000		
6	1	0	1	100		
7	10	0	1	100	0.5	
8	10		1,3,7	100	0.5	1.2

We first consider the special case where $F = Z$, therefore the probability of hitting the cache $\Pr\{H = 1\}$ is 1. This means that when the content is not downloaded through D2D communication, the BS satisfies the request by using the cellular link, without involving the backhaul channel, since the BS has a copy of each of the Z most popular files. We fix the ratio k/n_{c} to be $1/3$. Moreover, we assume that the channel does not introduce errors, i.e., $\varepsilon = 0$, and that nonoverlapping bands are used for adjacent clusters, i.e., $\kappa = 1$. We compare the average file download delay $\overline{T}_{\mathsf{dw}}$ of the considered network with MDS-coded DC with the delay of the scenario where the content is solely downloaded from the BS, denoted by $T_{\mathsf{bs}} = kt_{\mathsf{bs}}$ and with uncoded caching. In the following, with no loss of generality, we set $T_{\mathsf{bs}} = 1$ t.u.. In Figs. 3–5, we show the gain that can be achieved using MDS-coded DC, by reporting the ratio between T_{bs} and $\overline{T}_{\mathsf{dw}}$ as a function of the update interval Δ. In Figs. 3, 4, and 5, t_{d} is 10, 100, and 1000 times, respectively, smaller than t_{bs}. The solid lines in the plots correspond to the analytical closed-form expressions derived in the previous sections and markers correspond to simulation results. It is observed that the analytical expressions

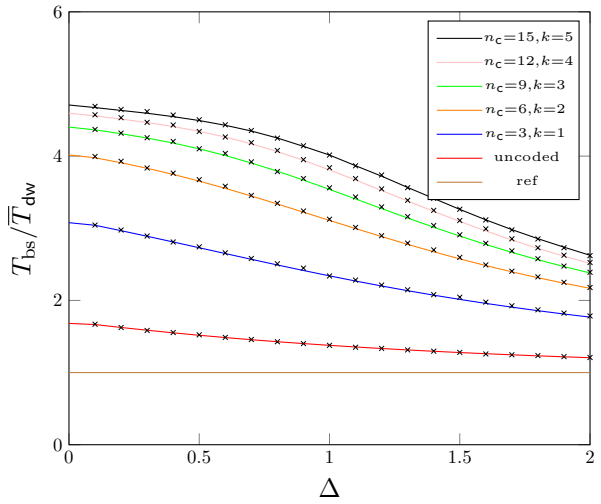

Figure 3. Ratio between the average file download delay without D2D communication and that of the scenario using MDS-coded DC when $Z = F$. $T_{bs} = kt_{bs}$, $t_{bs} = 10t_d$. Solid lines show analytical results and markers simulation results.

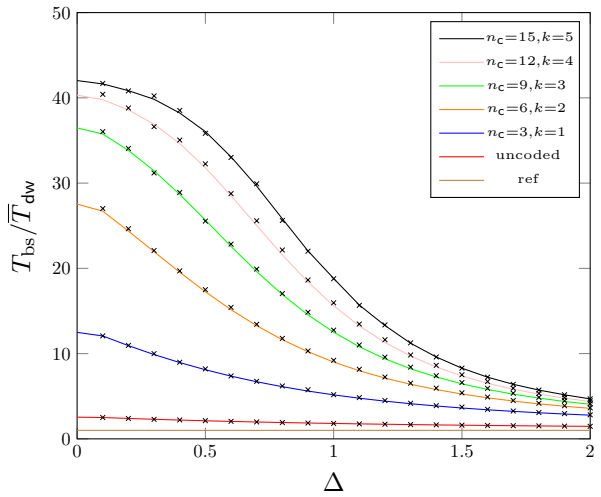

Figure 4. Ratio between the average file download delay without D2D communication and that of the scenario using MDS-coded DC when $Z = F$. $T_{bs} = kt_{bs}$, $t_{bs} = 100t_d$. Solid lines show analytical results and markers simulation results.

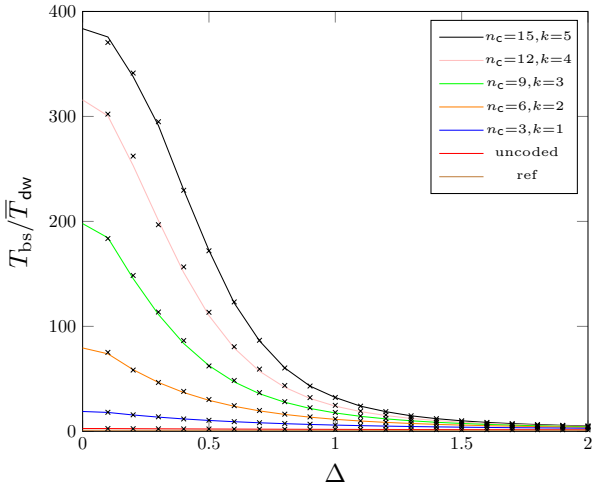

Figure 5. Ratio between the average file download delay without D2D communication and that of the scenario using MDS-coded DC when $Z = F$. $T_{bs} = kt_{bs}$, $t_{bs} = 1000t_d$. Solid lines show analytical results and markers simulation results.

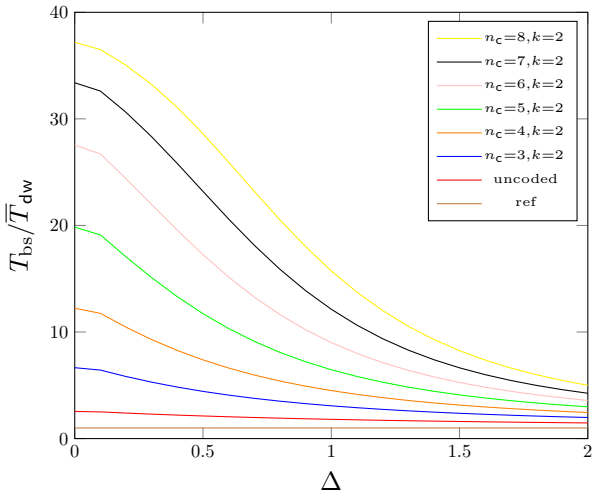

Figure 6. Ratio between the average file download delay without D2D communication and that of the scenario using MDS-coded DC when $Z = F$, for $k = 2$ and different n_c. $T_{bs} = kt_{bs}$, $t_{bs} = 100t_d$.

predict very well the actual performance, which shows the goodness of the approximations introduced in (14) and (20). The results clearly show that MDS-coded DC can greatly improve the performance in terms of content download delay with respect to the case where content is downloaded from the BS, provided that the update interval, Δ, is sufficiently small. For example, for $t_{bs} = 100t_d$ and $\Delta = 1$, a speed-up factor of around 19 in the download is achieved with respect to the case of downloading from the BS using a the MDS code with parameters $n_c = 15$ and $k = 5$. Interestingly, the results also show that the performance improves when k increases. In particular, simple replication (repetition coding with $k = 1$) is very inefficient and much better performance are achieved using larger MDS codes (of the same rate).

In Fig. 6, we show the gain as a function of the update interval for different average storage overhead. In particular, we fix $k = 2$, while the parameter n_c rages from 3 to 8. For comparison, we also report the uncoded case. The figure shows the advantage of MDS-coded DC for all considered cases.

We now consider the more general case where only part of the library of files is cached in the devices. We consider a library of size of $Z = 1000$ files and $F = 100$ files are cached in the DC network. This case takes into account the requests of files that are not cached in the DC library, that are satisfied by the BS through the backhaul and cellular link and that take on average T_{bh} t.u.. Due to the traffic congestion in the backhaul link, without loss of generality, we choose $T_{bh} = 2T_{bs}$. As explained in Section II, the n caching nodes are chosen uniformly at random and can be possibly different for each cached file. We compare the average file download delay $\overline{T}_{dw}$ of a network with MDS-coded DC with the delay of the classical scenario, where all the requests are satisfied by the BS through the backhaul link in T_{bh} t.u..

In Fig. 7, we show the download speedup factor $T_{bh}/\overline{T}_{dw}$ as a function of the parameter σ, which regulates the relative popularity of the files. For example, a large value of σ represents the case where few popular files are responsible for the majority of the download traffic. The figure refers to the case where $t_{bs} = 100t_d$, $\Delta = 0.5$, $\varepsilon = 0$ and $\kappa = 1$. The

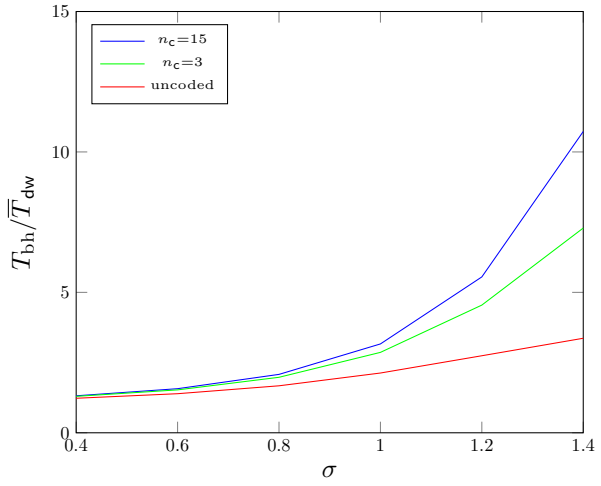

Figure 7. Ratio between the average file download delay of the classical scenario and that of the network using MDS-coded DC for $F < Z$. $T_{bh} = 2T_{bs}$, $t_{bs} = 100t_d$ and $\Delta = 0.5$.

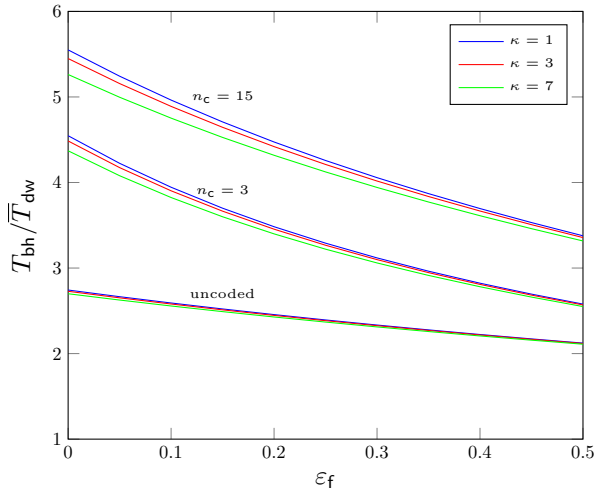

Figure 8. Ratio between the average file download delay of the classical scenario and that of the network using MDS-coded DC for $\sigma = 1.2$, $T_{bh} = 2T_{bs}$, $t_{bs} = 100t_d$ and $\Delta = 0.5$.

results confirm the gain that can be achieved by MDS-coded DC.

We now consider the effect of the D2D channel and of the number of competing clusters. In Fig. 8, we plot the download speedup factor $T_{bh}/\overline{T}_{dw}$ as a function of the channel erasure probability when we transmit the whole file, denoted as ε_f. The relationship between ε_f and the erasure probability of one symbol ε is

$$\varepsilon_f = 1 - (1-\varepsilon)^k .$$

We consider three MDS codes with $k/n_c = 1/3$, and for each code we plot three curves, corresponding to different number of competing clusters. Interestingly, the performance of the considered network is not very sensitive to the number of concurrent clusters. This observation suggests that the proposed network model, besides reducing the latency, has also a good spectral efficiency, since there is no need to allocate orthogonal frequency bands for adjacent clusters.

Finally, in Fig. 9 the proposed delay analysis is used to compare different caching placement strategies under an average cache size constraint per device of β files and different values of Δ. Besides the uniform content placement, we

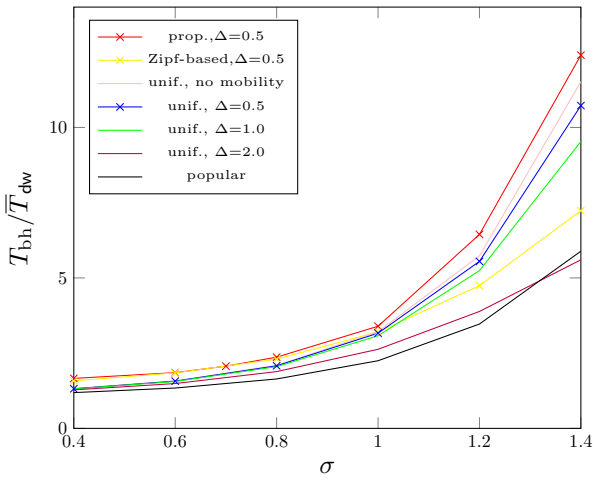

Figure 9. Ratio between the average file download delay of the classical scenario and that of the network using DC. Comparison of different caching placement schemes. For D2D communication, we assume $T_{bh} = 2T_{bs}$, $t_{bs} = 100t_d$.

consider three different caching strategies: i) The conventional popular content placement where all devices cache a copy of the β most popular files; ii) A content placement where a file is cached at any given node with a probability equal to its popularity probability, which we refer to as Zipf-based caching; iii) A proportional content placement where files are cached using MDS codes according to their popularity. In particular, we use the same k for each file and vary the code length $n_c \in \{0, k, \ldots, n_{max}\}$ (and thus code rate) proportionally to the file popularity. For Zipf-based caching and proportional content placement, the average file download can be computed using (6), where the quantities that depend on the code length are computed by averaging over the used codes. In the figure, we assume κ=1, $\beta = 10$, and ϵ=0. Furthermore, we choose k=5, n_c=15, and F=100 for the uniform placement, and k=3, n_{max}=15 and Δ=0.5 t.u. for the proportional content placement. Moreover, we set $T_{bh} = 2T_{bs}$ and $t_{bs} = 100t_d$. Note that the popular content placement does not foresee D2D communication, therefore the results are independent of Δ. As seen in the figure, for the considered scenario the proportional content placement yields the best performance (this is the case also for other values of Δ). The popular content placement has the worst performance, except for the case where both the parameter σ and the update interval are large, in which case popular content placement slightly outperforms the uniform content placement. Zipf-based caching performs as good as proportional content placement and better than uniform placement for small values of σ. However, for $\sigma > 1$, performs worse than uniform content placement and the loss is significant for larger values of σ.

For comparison purposes, we also plot in the figure the average file download delay for the case of no device mobility and uniform content placement (note that for popular content placement the curve for mobility and no mobility is the same). As expected, the device mobility reduces the performance gains with respect to the case of no mobility, and the loss is significant for large values of the update period. However, for $\Delta = 0.5$ and lower (the curve for $\Delta = 0.1$ is barely

better than that of $\Delta = 0.5$) the loss in performance gain with respect to the no mobility case is small. The same behavior is observed for the case of proportional allocation.

VI. Conclusions

We considered the cache of popular content in the mobile devices of a cellular network using maximum distance separable codes to speed-up content delivery. We considered a cluster-based network model that includes several practical aspects, such device mobility, inter and intra-cluster interference, and channel uncertainty. Moreover, we assumed that the network information is not instantaneosly available, but periodically broadcasted to the devices by the base station. For this scenario, we derived an analytical expression for the average file download delay as a function of the network parameters. The analysis shows that MDS-coded distributed caching can significantly reduce the download delay with respect to the traditional case where content is solely downloaded from the BS and the case where uncoded caching is used.

Interestingly, our analysis reveals that the gain with respect to uncoded and popular caching is bigger for longer codes. However, to realize the gains, the network update information must be broadcasted frequently enough. For small enough update period, the performance gains due to MDS-coded caching are close to those in the case of no mobility. We have also shown that the considered network has potentially a high spectral efficiency, since the performance only slightly degrades if we use the same frequency band for adjacent clusters with respect to the case where orthogonal bands are allocated.

The derived analytical expression for the download delay can be used to optimize the content placement through the optimization of the length and rate of the caching codes. Finally, an interesting research topic consists in extending our analysis by considering a more involved network model where the location of the devices is also accounted for, e.g., by considering different channel gains between the mobile devices depending on their relative positions.

Appendix A
Proof of Lemma 1

We denote by $X_1^{(\ell)}$ the number of caching nodes available for download at the time of the ℓth download request in the cluster, i.e., the number of caching nodes of the DC list that have not left the cluster at the time of the request. We compute $\Pr\{X_1 = x\}$ by averaging over an infinite number of requests,

$$\Pr\{X_1 = x\} = \lim_{L\to\infty} \frac{1}{L} \sum_{\ell=1}^{L} \Pr\{X_1^{(\ell)} = x\}\,. \tag{25}$$

Similarly, let $Y^{(\ell)}$ be the number of caching nodes at the beginning of the update interval wherein the ℓth request arrives, denoted by $\Delta^{(\ell)}$. We have

$$\Pr\{X_1^{(\ell)} = x\} = \sum_{y=0}^{\infty} \Pr\{X_1^{(\ell)} = x | Y^{(\ell)} = y\} \Pr\{Y^{(\ell)} = y\}\,. \tag{26}$$

In [10], it was shown that the probability $\Pr\{X_1^{(\ell)} = x | Y^{(\ell)} = y\}$ does not depend on ℓ (when ℓ grows large), and is given by (16). Its derivation is based on the observation that the number of caching nodes available for download in the update interval is described by a Poisson death process. The probability $\Pr\{Y^{(\ell)} = y\}$ can be written as

$$\begin{aligned}\Pr\{Y^{(\ell)} = y\} &= \Pr\{\tilde{Y}^{(\ell)} = y | \text{req. in } \Delta^{(\ell)}\} \\ &= \frac{\Pr\{\text{req. in } \Delta^{(\ell)} | \tilde{Y}^{(\ell)} = y\} \Pr\{\tilde{Y}^{(\ell)} = y\}}{\Pr\{\text{req. in } \Delta^{(\ell)}\}}\,,\end{aligned} \tag{27}$$

where in the second equality we used Bayes' rule. In (27), $\tilde{Y}^{(\ell)}$ is the number of caching nodes at the beginning of the update interval, which is described by a birth-death Poisson process, thus $\Pr\{\tilde{Y}^{(\ell)} = y\} = \pi_y(n_\mathsf{c})$. The probability $\Pr\{\text{req. in } \Delta^{(\ell)}\}$ is the probability that there is at least one request in $\Delta^{(\ell)}$. It depends on the inter-request time, which in turn depends on the number of nodes in the cluster. Therefore, we compute

$$\Pr\{\text{req. in } \Delta^{(\ell)}\} = \sum_{m=1}^{\infty} (1 - e^{-m\omega\Delta}) \pi_m(M_\mathsf{c})\,,$$

where $1 - e^{-m\omega\Delta}$ is the probability that the inter-request time is shorter than Δ when m nodes are present in the cluster. Similarly, we compute $\Pr\{\text{req. in } \Delta^{(\ell)} | \tilde{Y}^{(\ell)} = y\}$ as

$$\Pr\{\text{req. in } \Delta^{(\ell)} | \tilde{Y}^{(\ell)} = y\} = \sum_{m=y}^{\infty} (1 - e^{-m\omega\Delta}) \pi_{m-y}(M_\mathsf{c} - n_\mathsf{c}),$$

where $\pi_{m-y}(M_\mathsf{c} - n_\mathsf{c})$ is the probability that there are m nodes in the cluster, given that there are y caching nodes. Since these probabilities are independent of the specific request, we conclude that $\Pr\{Y^{(\ell)} = y\}$ is also independent of ℓ. Substituting (27) into (26), we observe that $\Pr\{X_1^{(\ell)} = x\}$ is also independent of ℓ and using (25) we prove the lemma.

Appendix B
Proof of Lemma 4

We compute the conditional probability that no symbols are downloaded by averaging over an infinite number of requests,

$$\Pr\{S_1 = 0 | R = i\} = \lim_{L\to\infty} \frac{1}{L} \sum_{\ell=1}^{L} \Pr\{S_1^{(\ell)} = 0 | R^{(\ell)} = i\}\,,$$

where $S_1^{(\ell)} \in \{0, 1\}$ is the RV describing the download of the first symbol for the ℓth request. We now consider the computation of $\Pr\{S_1^{(\ell)} = 0 | R^{(\ell)} = i\}$. The recovery of the first symbol fails with probability 1 if the requesting node leaves the cell before completing the download. It also fails if the requesting node stays in the cluster but no caching nodes are available, if it chooses to download from a caching node which departs before t_d t.u. from the start of the download, or if it chooses a node that does not leave the cluster but the channel causes the loss of the data.

Let $O^{(\ell)}$ be the departure time of the node which places the ℓth request and $W^{(\ell)}$ the time instant of the ℓth request in the cluster. We define $\mathcal{A}_1^{(\ell)} = \{O^{(\ell)} - W^{(\ell)} > t_\mathsf{d}\}$ as the event

that the node which places the ℓth request stays in the network for more than t_{d} t.u. from the start of the download. The corresponding probability does not depend on ℓ and is easily computed as $\Pr\{\mathcal{A}_1^{(\ell)}\} = e^{-\lambda t_{\mathsf{d}}}$. Similarly, the probability that the requesting node departs before t_{d} t.u. from the start of the download is $(1-e^{-\lambda t_{\mathsf{d}}})$. Therefore, the conditional probability that the ℓth request fails the first symbol download is

$$\Pr\{S_1^{(\ell)}=0|R^{(\ell)}=i\} = (1-e^{-\lambda t_{\mathsf{d}}}) + e^{-\lambda t_{\mathsf{d}}}\Pr\{S_1^{(\ell)}=0|\mathcal{A}_1^{(\ell)},R^{(\ell)}=i\}\,.$$

Let $G_1^{(\ell)} \in \{0,\dots,\infty\}$ be the number of caching nodes useful for download for the ℓth request, i.e., the number of caching nodes in the DC list that have not left the cluster at the time of the ℓth request, excluding the requesting node itself if it belongs to the DC list. Let $D_1^{(\ell)} \in \{0,\dots,\infty\}$ the number of departures in t_{d} t.u. among the $G_1^{(\ell)}$ caching nodes. The probability $\Pr\{S_1^{(\ell)}=0|\mathcal{A}_1^{(\ell)},R^{(\ell)}=i\}$ can be written as

$$\begin{aligned}
&\Pr\{S_1^{(\ell)}=0|\mathcal{A}_1^{(\ell)},R^{(\ell)}=i\} = \\
&= \sum_g \sum_d \Pr\{S_1^{(\ell)}=0|\mathcal{A}_1^{(\ell)},R^{(\ell)}=i,G_1^{(\ell)}=g,D_1^{(\ell)}=d\} \\
&\qquad \cdot \Pr\{G_1^{(\ell)}=g,D_1^{(\ell)}=d|\mathcal{A}_1^{(\ell)},R^{(\ell)}=i\} = \\
&= \sum_g \sum_d \Pr\{S_1^{(\ell)}=0|\mathcal{A}_1^{(\ell)},G_1^{(\ell)}=g,D_1^{(\ell)}=d\} \\
&\qquad \cdot \Pr\{G_1^{(\ell)}=g,D_1^{(\ell)}=d|R^{(\ell)}=i\}\,. \qquad (28)
\end{aligned}$$

(28) is obtained by observing that i) the probability that the download of the first symbol fails conditioned to $G_1^{(\ell)}$ and $D_1^{(\ell)}$ is independent of the type of request, and that ii) the number of useful caching nodes and the number of departures in t_{d} t.u. among them is independent of the departure of the requesting node. The probability $\Pr\{S_1^{(\ell)}=0|\mathcal{A}_1^{(\ell)},G_1^{(\ell)}=g,D_1^{(\ell)}=d\}$ is equal to 1 if

- there are no useful caching nodes, i.e., $g=0$,
- the requesting node chooses one of the d caching nodes that leaves the cluster in t_{d} t.u.,
- or the requesting node chooses one of the caching nodes that does not leave the cluster but the channel conditions prevent the download.

Otherwise, we have $\Pr\{S_1^{(\ell)}=0|\mathcal{A}_1^{(\ell)},G_1^{(\ell)}=g,D_1^{(\ell)}=d\} = 0$. We observe that the number of departures of useful caching nodes conditioned to their number is independent of the type of request and that the number of caching nodes useful for download $G_1^{(\ell)}$ is related to $X_1^{(\ell)}$ by

$$\Pr\{G_1^{(\ell)}=g|R^{(\ell)}=i\} = \Pr\{X_1^{(\ell)}=g+i|R^{(\ell)}=i\}\,,$$

since when the request originates from a caching node of the DC list, the requesting node itself is not counted among the useful caching nodes. Therefore, the probability $\Pr\{G_1^{(\ell)}=g,D_1^{(\ell)}=d|R^{(\ell)}=i\}$ can be written as

$$\begin{aligned}
&\Pr\{G_1^{(\ell)}=g,D_1^{(\ell)}=d|R^{(\ell)}=i\} = \\
&\qquad \Pr\{D_1^{(\ell)}=d|G_1^{(\ell)}=g\}\Pr\{X_1^{(\ell)}=g+i|R^{(\ell)}=i\}\,.
\end{aligned}$$

We denote by $\theta(d,g)$ the probability $\Pr\{D_1^{(\ell)}=d|G_1^{(\ell)}=g\}$, given in (21). Its derivation is similar to that of $\Pr\{X_1|Y\}$ [10]. The probability $\Pr\{X_1^{(\ell)}=g+i|R^{(\ell)}=i\}$ is independent of the specific request and is given by (19).

After simple manipulations, we finally obtain

$$\begin{aligned}
&\Pr\{S_1^{(\ell)}=0|R^{(\ell)}=i\} = 1+e^{-\lambda t_{\mathsf{d}}}\Big(\Pr\{X_1=i|R=i\} \\
&+\sum_{g=1}^{\infty}\sum_{d=0}^{g}\Big(\frac{d}{g}+\varepsilon\frac{g-d}{g}\Big)\Pr\{X_1=g+i|R=i\}\theta(d,g)-1\Big). \qquad (29)
\end{aligned}$$

Since the probabilities involved in (29) are all independent of ℓ, the lemma is proved.

Appendix C
Proof of Lemma 5

To evaluate the probability of complete download from the DC network, we start with the following limit,

$$\Pr\{\mathbf{S}_{[k-i]}=\mathbf{1}_{k-i}|R=i\} = \lim_{L\to\infty}\frac{1}{L}\sum_{\ell=1}^{L}\Pr\{\mathbf{S}_{[k-i]}^{(\ell)}=\mathbf{1}_{k-i}|R^{(\ell)}=i\}$$

where $\mathbf{S}_{[j]}^{(\ell)} = (S_1^{(\ell)},\dots,S_j^{(\ell)})$ and $S_i^{(\ell)}$ describes the successful symbol download at the ith attempt of the ℓth request. We consider the ℓth request in the cluster and, similarly to the proof of Lemma 4, we will find that this probability is independent of ℓ. We denote by the RV $G_j^{(\ell)} \in \{0,\dots,\infty\}$ the number of caching nodes useful for download at the time of the jth attempt of the ℓth request, i.e., the caching nodes of the DC list not yet contacted, excluding the requesting node if it belongs to the DC list. We denote by the RV $D_j^{(\ell)} \in \{0,\dots,\infty\}$ the number of departures in t_{d} t.u. among the $G_j^{(\ell)}$ nodes. We also denote by $\mathcal{A}_j^{(\ell)} = \{O^{(\ell)}-W^{(\ell)} > jt_{\mathsf{d}}\}$ the event that the node which places the ℓth request stays in the cluster for more than jt_{d} t.u. from the start of the download. The corresponding probability does not depend on ℓ and is given by $\Pr\{\mathcal{A}_j^{(\ell)}\} = e^{-j\lambda t_{\mathsf{d}}}$. The probability of complete download is zero if the requesting node departs before $(k-i)t_{\mathsf{d}}$ t.u. from the start of the download, therefore we can write

$$\begin{aligned}
&\Pr\{\mathbf{S}_{[k-i]}^{(\ell)}=\mathbf{1}_{k-i}|R^{(\ell)}=i\} = \\
&\qquad e^{-(k-i)\lambda t_{\mathsf{d}}}\Pr\{\mathbf{S}_{[k-i]}^{(\ell)}=\mathbf{1}_{k-i}|\mathcal{A}_{k-i}^{(\ell)},R^{(\ell)}=i\}\,.
\end{aligned}$$

The probability $\Pr\{\mathbf{S}_{[k-i]}^{(\ell)}=\mathbf{1}_{k-i}|\mathcal{A}_{k-i}^{(\ell)},R^{(\ell)}=i\}$ can be written as

$$\begin{aligned}
&\Pr\{\mathbf{S}_{[k-i]}^{(\ell)}=\mathbf{1}_{k-i}|\mathcal{A}_{k-i}^{(\ell)},R^{(\ell)}=i\} = \\
&= \sum_g\sum_d \Pr\{S_{k-i}^{(\ell)}=1|\mathcal{A}_{k-i}^{(\ell)},G_{k-i}^{(\ell)}=g,D_{k-i}^{(\ell)}=d\} \\
&\cdot \Pr\{G_{k-i}^{(\ell)}=g,D_{k-i}^{(\ell)}=d,\mathbf{S}_{[k-i-1]}^{(\ell)}=\mathbf{1}_{k-i-1}|\mathcal{A}_{k-i}^{(\ell)},R^{(\ell)}=i\} \\
&= (1-\varepsilon)\sum_{g=1}^{\infty}\sum_{d=0}^{g}\frac{g-d}{g}\gamma_{k-i}^{(\ell)}(g,d,i)\,. \qquad (30)
\end{aligned}$$

The last equality is obtained by observing that the probability $\Pr\{S_{k-i}^{(\ell)}=1|\mathcal{A}_{k-i}^{(\ell)},G_{k-i}^{(\ell)}=g,D_{k-i}^{(\ell)}=d\}$ is nonzero and equal to 1 if the requesting node chooses one of the caching nodes of the DC list that remains in the cluster and the channel does not cause data erasure. Moreover, we have

defined $\gamma_j^{(\ell)}(g,d,i) \triangleq \Pr\{G_j^{(\ell)} = g, D_j^{(\ell)} = d, \mathbf{S}_{[j-1]}^{(\ell)} = \mathbf{1}_{j-1} | \mathcal{A}_j^{(\ell)}, R^{(\ell)} = i\}$, that can be computed as

$$\begin{aligned}\gamma_j^{(\ell)}(g,d,i) &= \Pr\{D_j^{(\ell)} = d | G_j^{(\ell)} = g\} \\ &\cdot \sum_{g'} \sum_{d'} \Pr\{G_j^{(\ell)} = g | G_{j-1}^{(\ell)} = g', D_{j-1}^{(\ell)} = d', S_{j-1}^{(\ell)} = 1\} \\ &\cdot \Pr\{S_{j-1}^{(\ell)} = 1 | \mathcal{A}_j^{(\ell)}, G_{j-1}^{(\ell)} = g', D_{j-1}^{(\ell)} = d'\} \\ &\cdot \Pr\{G_{j-1}^{(\ell)} = g', D_{j-1}^{(\ell)} = d', \mathbf{S}_{[j-2]}^{(\ell)} = \mathbf{1}_{j-2} | \mathcal{A}_{j-1}^{(\ell)}, R^{(\ell)} = i\},\end{aligned}$$

for $j > 1$. We also define $E(g,g',d') \triangleq \Pr\{G_j^{(\ell)} = g | G_{j-1}^{(\ell)} = g', D_{j-1}^{(\ell)} = d', S_{j-1}^{(\ell)} = 1\}$, which is equal to one if $g = g' - d' - 1$ and $g' > d'$, and zero otherwise. The condition $g = g' - d' - 1$ follows from the fact that the number of useful caching nodes after a successful symbol download is equal to the number of useful caching nodes still alive, $g' - d'$, minus the caching node just used. The condition $g' > d'$ comes from the fact that the $(j-1)$th symbol download is assumed to be successful, i.e., $S_{j-1}^{(\ell)} = 1$.

It is easy to prove by induction that the probability in (30) does not depend on ℓ. By defining $\gamma_j(g,d,i) \triangleq \gamma_j^{(\ell)}(g,d,i)$, we obtain the recursion (23), with initial condition (24). The probabilities $\gamma_j(g,d,i)$, $j \geq 1$, are equal to zero for $d > g$.

REFERENCES

[1] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 8402–8413, Dec. 2013.

[2] N. Golrezaei, P. Mansourifard, A. F. Molisch, and A. G. Dimakis, "Base-station assisted device-to-device communications for high-throughput wireless video networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 7, pp. 3665–3676, Jul. 2014.

[3] M. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

[4] N. Golrezaei, A. G. Dimakis, and A. F. Molisch, "Scaling behavior for device-to-device communications with distributed caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 7, pp. 4286–4298, May 2014.

[5] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless D2D networks," *IEEE Trans. Inf. Theory*, vol. 62, no. 2, pp. 849–869, Feb. 2016.

[6] C. Yang, Z. Chen, Y. Yao, and B. Xia, "Performance analysis of wireless heterogeneous networks with pushing and caching," in *Proc. IEEE Int. Conf. Commun. (ICC)*, London, U.K., Jun. 2015, pp. 2190–2195.

[7] J. Pääkkönen, C. Hollanti, and O. Tirkkonen, "Device-to-device data storage for mobile cellular systems," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Atlanta, GA, 2013.

[8] J. Pedersen, A. Graell i Amat, I. Andriyanova, and F. Brännström, "Repair scheduling in wireless distributed storage with D2D communication," in *Proc. IEEE Inf. Theory Work. (ITW)*, Jeju Island, Korea, 2015.

[9] V. Bioglio, F. Gabry, and I. Land, "Optimizing MDS codes for caching at the edge," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, San Diego, CA, 2015.

[10] J. Pedersen, A. Graell i Amat, I. Andriyanova, and F. Brännström, "Distributed storage in mobile wireless networks with device-to-device communication," *IEEE Trans. Commun.*, vol. 64, no. 11, pp. 4862–4878, Nov. 2016.

[11] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network Coding for Distributed Storage Systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[12] D. S. Papailiopoulos and A. G. Dimakis, "Locally repairable codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 10, pp. 5843–5855, Oct. 2014.

[13] A. Piemontese and A. Graell i Amat, "MDS-coded distributed storage for low delay wireless content delivery," in *Proc. Int. Symp. Turbo Codes & Iterative Inform. Proc. (ISTC)*, Brest, France, Sep. 2016.

[14] S. L. Miller and D. Childers, *Probability and Random Processes*. Amsterdam, The Netherlands: Elsevier, 2004.

[15] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: evidence and implications," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, New York, NY, Mar. 1999.

[16] Y. Pei and Y. C. Liang, "Resource allocation for device-to-device communications overlaying two-way cellular networks," *IEEE Trans. Wireless Commun.*, vol. 12, no. 7, pp. 3611–3621, Jul. 2013.

[17] M. Ji, G. Caire, and A. F. Molisch, "Wireless device-to-device caching networks: Basic principles and system performance," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 1, pp. 176–189, Jan. 2016.

[18] A. Asadi, Q. Wang, and V. Mancuso, "A survey on device-to-device communication in cellular networks," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1801–1819, Apr. 2014.