Paakkonen, Joonas; Barreal, Amaro; Hollanti, Camilla; Tirkkonen, Olav

Coded Caching Clusters with Device-to-Device Communications

Published in:
IEEE Transactions on Mobile Computing

DOI:
10.1109/TMC.2018.2832636

Published: 01/02/2019

Document Version
Peer reviewed version

Please cite the original version:
Paakkonen, J., Barreal, A., Hollanti, C., & Tirkkonen, O. (2019). Coded Caching Clusters with Device-to-Device Communications. *IEEE Transactions on Mobile Computing*, *18*(2), 264 - 275. [8353772].
https://doi.org/10.1109/TMC.2018.2832636

Coded Caching Clusters with Device-to-Device Communications

Joonas Pääkkönen, Amaro Barreal, Camilla Hollanti, *Member, IEEE*, and Olav Tirkkonen, *Member, IEEE*

Abstract—We consider a geographically constrained caching community where popular data files are cached on mobile terminals and distributed through Device-to-Device (D2D) communications. To ensure availability, data files are protected against user mobility, or *churn*, with select caching and erasure coding methods. Communication and storage costs are considered, with an objective of minimizing the consumption of radio resources, given an available storage size. We focus on finding the coding method that minimizes the overall cost. Closed-form expressions for the expected consumption of radio resources incurred by data delivery and redundancy maintenance are derived. Closed form transmission costs in a circular caching community with a specific node density and caching method are calculated, when cost obeys a power law of distance. Our results are illustrated by numerical examples and verified by extensive computer simulations.

Index Terms—Device-to-Device Communications, Regenerating Codes, Wireless Caching, Markov Processes, Distributed Data Storage

1 Introduction

Recent years have seen an unprecedented growth in wireless data traffic and this growth is not slowing down. Compared to 2016, aggregate smartphone traffic is expected to increase almost tenfold by 2020 [3]. One promising technology to help meet the needs of heavily loaded future cellular networks is *Device-to-Device* (D2D) communications. The major benefit of D2D is that it allows for direct communication between proximate user equipment without the need of base stations, hence potentially offering higher data transfer speeds, lower latency, decreased interference, increased spectral efficiency and lower overall power consumption [4], [5], [6], [7], [8].

Another uprising technology is wireless *caching* – either directly on user terminals, base stations or both [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26]. Wireless D2D caching is an enticing future technology where data could be stored and distributed directly between mobile terminals – especially if the involved mobile terminals are geographically close to each other and can thus form D2D *clusters*. Geographically constrained caching is of particular interest since the popularity of data is highly location dependent [20].

Wireless content caching and data distribution through direct links have been proposed in several works such as [27], where delay-tolerant networking is considered for message dissemination and forwarding. In [28], a wireless peer-to-peer type of application is studied and it is shown that caching can greatly increase the application-level throughput. The potential of coded wireless D2D caching is investigated in [29], while [12] shows that D2D caching can improve the throughput of wireless video transmission. A method for minimizing the energy consumption of D2D caching nodes is analyzed in [30], whereas a joint transmission and caching policy that reduces both the total energy consumption at the base station and the economical cost for the operator is presented in [31]. In [32], the authors study cluster-centric D2D networks and demonstrate significant improvements in the network performance.

Joint use of caching and erasure coding for D2D clusters has been proposed in our previous work [1], [2] for instantaneous repairs. This work has been extended in [33] to efficiently scheduled repairs. Further work on distributed storage with D2D communications has been done in [34], where a combination of D2D and social networks is considered. In [1] we looked for a way to strictly minimize the amount of data traffic in caching clusters and found that repetition coding yields the best results for the considered system model. We then found in [2] that the optimal coding method, *i.e.*, the coding method that minimized a predetermined cost function, highly depends on the popularity of the file.

A clear drawback of wireless caching on mobile terminals is unconstrained user mobility, a coming-and-going phenomenon of mobile users here referred to as *churn*: when a caching node moves away from the caching cluster, its content is lost. To combat this, we introduce erasure coding to ensure data availability. The focus of this article is studying the performance of such coded caching clusters.

- *J. Pääkkönen and O. Tirkkonen are with the Department of Communications and Networking, Aalto University, Finland.*
- *A. Barreal and C. Hollanti are with the Department of Mathematics and Systems Analysis, Aalto University, Finland.*
  Emails: firstname.lastname@aalto.fi
- *Parts of this work were presented at IEEE GLOBECOM 2013 [1] and MACOM 2015 [2].*
- *The authors are financially supported by the Academy of Finland under Grants #276031, #282938, #283262, #284725 and #299916, the Finnish Funding Agency for Technology and Innovations under grant 2383/31/2014, as well as a grant from the Finnish Foundation for Technology Promotion. The support from the ESF COST Action IC1104 is gratefully acknowledged. The work of Joonas Pääkkönen is supported by TKK tukisäätiö.*

The aim of this work is to investigate under what kind of circumstances storage coding outperforms caching without coding. The aim is not to propose new coding methods or to promote certain caching schemes, but to study whether storage coding can improve the system performance compared to schemes without coding. The main contributions of this article can be summarized as follows:

- Closed-form expressions for the expected transmission cost based on signal attenuation of both uncoded and coded D2D caching methods are derived.
- We examine under which conditions coded caching outperforms uncoded caching without redundancy. We stress that we focus on investigating the maximum theoretical gains without considering the effect of interference or retransmissions.
- It is shown that coded caching can yield significant cost savings in terms of overall consumption of radio resources, *e.g.*, energy.

The rest of this article is organized as follows. In Section 2, we present the system model used throughout this work. In Section 3, we introduce the proposed caching methods. Analytical cost estimates are derived in Section 4, while simulation results are presented and compared with the analytical results in Section 5. Finally, conclusions are drawn in Section 6.

2 System Model

We begin by introducing the system model assumed throughout this work. We model a cluster of mobile terminals with data storage capabilities. Throughout this paper, a *cluster* is defined as a geographically constrained area consisting of a time-varying number of users, or *nodes*. The instantaneous number of nodes in the cluster follows the Poisson distribution. The expected total number of nodes present in the cluster is denoted by m. All nodes are assumed to be uniformly distributed inside the cluster according to a uniform Binomial Point Process (BPP) as in, *e.g.*, [22].

A single base station is located outside of the cluster. The base station can be contacted if the desired data file is not cached on the users. The base station also keeps track of what is stored and where. We assume that keeping track of this information yields a negligible cost compared to the considered data transmission costs. A graphical representation of the model is displayed in Figure 1. The nodes inside the cluster form a D2D caching community. New nodes arriving in the cluster do not have useful content cached upon arrival, which implies that our results are a lower bound on the achievable caching gains. We assume that each node knows about the content stored in every other node, and any two nodes can communicate data.

Comparing intra-cluster transmissions to cellular downlink transmissions, the protocols may be less reliable, meaning that for a given path loss between the nodes, more resources (energy, time, frequency) have to be used for reliable communication of an information bit. Also, more signaling overhead may be required for D2D caching than for conventional downlink transmissions. In contrast, due to the limited size of the clusters, interference from intra-cluster transmissions may spread less interference than cellular transmissions. These effects may lead to the overall consumption of radio resources to be a different function of path loss in the D2D community than in cellular downlink. We model this with an overall multiplicative factor Θ.

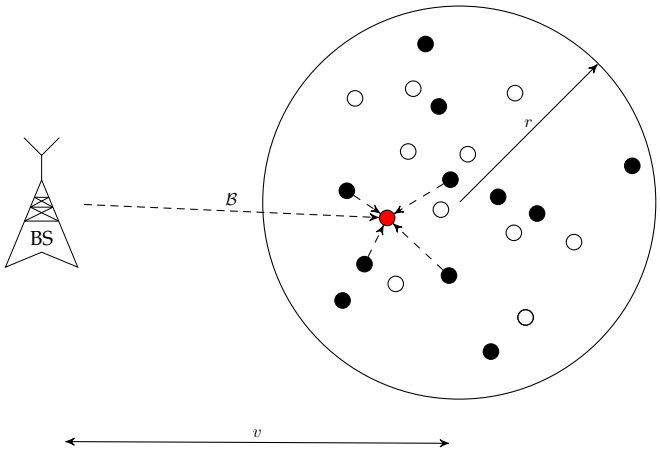


Fig. 1: D2D caching cluster system model example when the cluster is a disk of radius r. Instead of always having to contact a remote base station (BS), users in the cluster are able to communicate with each other through direct links. The cost of using the traditional downlink is $\mathcal{B}$. The distance between the BS and the center of the cluster is v.

The time dynamics of the system are modeled as follows. We assume that the time that an arbitrary node remains active in the cluster follows an exponential distribution with expected value T. We define a *failure* as the event when a node becomes inactive by leaving the system and denote the *node failure rate*, or *churn rate*, by $\lambda = 1/T$. We further assume that the nodes arrive in the cluster at random time instances according to a Poisson arrival process, which implies that the inter-arrival time of the nodes follows the exponential distribution. We assume that the expected value of this distribution is $1/(m\lambda)$, so the arrival rate is $m\lambda$.

Now by Little's law [35], the expected number of nodes in the cluster is $m\lambda T = m$, as we desire. With these parameters, we can model the instantaneous state of the system via an M/M/∞ Markov model (cf. Figure 2), which has been widely used to model wireless cellular systems with exponential[1] dwell times [36], [37], [38]. In this work, we only consider the steady state of the chain with m nodes in the cluster on average. In the parlance of queueing theory, here the arrival rate is $m\lambda$ and the service rate is λ. Hence, the probability that the system is in state j, *i.e.*, that there are j nodes in the cluster, can be written as [39]

$$\pi(j) = \frac{\left(\frac{m\lambda}{\lambda}\right)^j}{j!} e^{-\left(\frac{m\lambda}{\lambda}\right)} = \frac{m^j}{j!} e^{-m}. \quad (1)$$

We henceforth consider a single data file of unit size without loss of generality. Each user in the cluster can request the file anytime. The request interval of a user follows

1. Note that the results derived throughout this work are valid for the more general M/G/∞ queue where the service time need not be exponential. This is due to the fact that we only consider expected values.

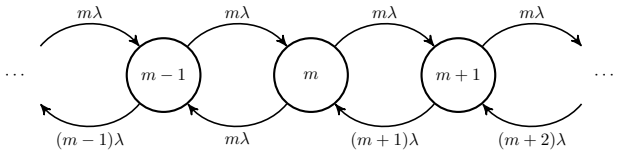


Fig. 2: M/M/∞ Markov chain. The state refers to the number of users in the cluster.

an exponential distribution with expected value $1/\omega$, where we call ω the *request rate* or, by a slight abuse of terminology, the *file popularity*. We concentrate on the case $\omega < \lambda$ as we assume that the vast majority of the users request the file only once during their visit to the cluster.

Throughout this paper, we assume instant repair after failures in order to avoid simultaneous node failures. In a practical caching system this would mean that cluster membership is soft in the sense that users moving into the border region are considered to be candidates for failing. With instant repair, no matter which coding method is used, there are always n caching nodes in the cluster as long as the Markov chain in Figure 2 never goes to a state lower than n, which we deem a valid assumption as we only investigate the case $n \ll m$, and thus the probability of finding the chain in states with a small number of members is extremely small[2].

3 Caching Methods

We consider four different caching methods.

3.1 Simple caching (SC)

A single node stores a full copy of the file. The file is not protected against storage node failures since no redundancy is enabled. As soon as the caching node leaves the system, the data file is lost from the caching community and the next requesting node needs to download the entire file from the base station. This node then automatically becomes the new caching node and, as long as it remains in the cluster, all file requests from other nodes are served by this node through D2D communications. Note that in this scheme, when a new user downloads the file, it caches the file only if nobody else is caching the file at the moment.

The simple caching scheme can be modeled with a Markov chain as depicted in Figure 3.

By performing a simple cut-based analysis of the chain, the steady state probabilities of the upper chain can be shown to become $\pi_j - \zeta_j$ and the lower chain ζ_j, where π_j are the M/M/∞ probabilities from (1), and ζ_j fulfil the recursion

$$\zeta_{j+1} = \left(\frac{m}{j} + \frac{\omega}{\lambda} + 1\right)\zeta_j - \frac{m}{j}\zeta_{j-1} - \frac{\omega}{\lambda}\pi_j \ ,$$

with $\zeta_0 = 0$. For the purposes of this article we do not need to find these steady state probabilities. Instead, in Section 4.2, we derive an approximation of the related performance metric. We use the chain of Figure 3 only to model the behavior of the system with computer simulations in order to empirically measure the performance of simple caching. This will be done later in Section 5.

2. For example, if $m = 100$ and $n = 6$, values which we will later use in our simulations, the probability that the number of nodes in the cluster drops to n or below is approximately 5.5×10^{-35}.

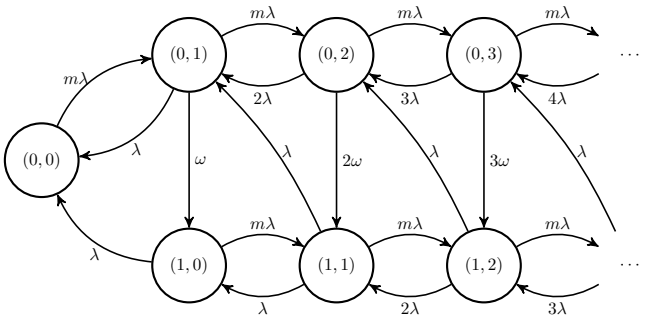


Fig. 3: Simple caching Markov chain state diagram. State (x, y) refers to having $x \in \{0, 1\}$ caching nodes and $y = 0, 1, 2, 3, ...$ empty nodes in the cluster.

3.2 Multicaching (MC)

This caching scheme is otherwise similar to simple caching, but here each user requesting the data file also caches it and thus provides additional redundancy. In other words, if there is one node already caching the file when another node requests the file, then the requesting node downloads the file from the single caching node in D2D mode and becomes a second caching node. If there are more than one user caching the file when a request takes place, then the requesting node contacts the nearest caching node and downloads the file from this node in D2D mode and becomes an additional caching node itself. Note that while this method increases the density of caching users, the drawback is the increased cumulative storage space consumption.

Due to the complex nature of analyzing this caching scheme, we do not provide mathematical analysis for this method, but only plot its simulated[3] performance. Nevertheless, the performance of this method is intuitively vese to that of simple caching when file popularity is low. On the contrary, when file popularity is high, the cluster starts to fill up with replicas and the performance of multicaching improves compared to simple caching. Indeed, we see that multicaching can outperform all the other methods when file popularity is very high as will be shown later in Section 5.

3.3 Replication (REP)

The most elementary way of adding redundancy to the system is simply to store multiple copies of the entire file on separate nodes. We refer to this strategy as *n-replication*, also known as *repetition coding*, where $n \ll m$ nodes store a replica of the file. When the system operates under this method, the file can be retrieved, or a lost node repaired, by contacting simply one of the storage nodes.

3. A Markov chain similar to that of Figure 3 can be constructed for the multicaching scheme. The multicaching Markov chain has an infinite number of states in both the horizontal and the vertical direction. The simulation program, the results of which are shown in Section 5, is based on this chain.

The obvious downside of replication is that it consumes more storage space than coded storage. Furthermore, the *repair bandwidth*, that is, the amount of data traffic that replacing a lost storage node incurs, is equal to the size of the entire file. Hence, the repair bandwidth is equal to the *reconstruction bandwidth*, which we define as the amount of data traffic incurred when a user downloads and reconstructs the data file.

3.4 Regenerating Codes (MBR and MSR)

We interpret the considered system as a *Distributed Storage System* (DSS) which is composed of $n \ll m$ storage nodes[4]. The original data file is encoded into n coded fragments of size α each. Storage nodes are assigned one of the coded fragments, and the entire file can be recovered by contacting any $k < n$ storage nodes, a feature also referred to as the *Maximum Distance Separability* (MDS) property of a code. This property is what allows the system to be resistant against arbitrary failure sequences.

To maintain redundancy, whenever a storage node fails, it is instantly replaced with a *newcomer* node that is randomly chosen from the empty nodes present in the cluster. This newcomer node contacts any $d \leq n-1$ storage nodes, downloads β units of data from each and stores α units of data. Note that the new content in the newcomer node does not need to be exactly the data that were lost in the failed node. Hence, we consider *functional repair*, which ensures that both the MDS property and the regeneration property hold after an arbitrary failure.

A DSS is determined by the tuple $(n, k, d, \alpha, \gamma)$, whereof the triple (n, k, d) consists of the *storage degree*, *reconstruction degree* and *repair degree*, respectively. In other words, reconstructing the data file requires contacting k out of total n storage nodes, while repairing the contents of a lost node requires contacting d nodes. In addition, the parameter pair (α, γ) consists of the fragment size α stored in each of the n storage nodes, and the *repair bandwidth* γ that is the total number of units of data that a newcomer needs to download for repairing a lost node. Note that when repairing, each storage node involved in the repair process transmits β units of data to the newcomer node, so that $\gamma = d\beta$.

A given tuple of parameters $(n, k, d, \alpha, \gamma)$ is *feasible* if a code with such α and γ exists. For a result on the existence of feasible parameter tuples, we refer to [40, Thm. 1]. More importantly, there is a natural tradeoff between α and γ given by a piecewise linear function. Codes lying on this tradeoff curve are called *regenerating codes*. Hence, regenerating codes offer an optimal tradeoff between storage space consumption and repair bandwidth, while maintaining the MDS property. Furthermore, any d nodes can be contacted to resurrect a lost node while maintaining these properties after repairs. Hence, regenerating codes are an attractive choice for our scenario.

In this work, we consider two types of regenerating codes: codes attaining one of the two extremal points, *i.e.*, the points where either the storage space consumption or repair bandwidth is minimized. These codes are known as *minimum storage regenerating* (MSR) codes and *minimum bandwidth regenerating* (MBR) codes, respectively. For a file of unit size, these points are achieved by the pairs [40]

$$(\alpha_{\text{MSR}}, \gamma_{\text{MSR}}) = \left(\frac{1}{k}, \frac{d}{k(d-k+1)}\right), \tag{2}$$

$$(\alpha_{\text{MBR}}, \gamma_{\text{MBR}}) = \left(\frac{2d}{k(2d-k+1)}, \frac{2d}{k(2d-k+1)}\right). \tag{3}$$

It has been shown that, in the typical case $k \leq d \leq n-1$ which we assume throughout this work, code constructions exists for both the MSR and the MBR point, see, *e.g.*, [41]. Note that the reason we do not consider traditional MDS erasure codes, such as Reed-Solomon codes, is that, for the purpose of this work, they are merely a special case of MSR codes with $k = d$.

4 Cost Estimates

The task of an entity managing the D2D storage community is to decide which files to store locally, and which caching method to use for each file. The objective is to reduce wireless traffic by exploiting available memory. By addressing the optimization of the use of radio resources for a population of files, we find that it is sufficient to understand the cost generated by an individual file, subject to both radio resource and storage costs. These can be directly calculated for the different caching methods.

4.1 Cost for Population of Files

First, we formulate an optimization problem in a setting where the network management decides the caching principle for a whole population of files, represented by a probability density function $f(\omega)$ of file popularities. For simplicity, all files are assumed to have the same size. This is achieved, *e.g.*, by segmenting larger files to a smallest size. The task is thus to find the caching method $g(\omega)$ as a function of file popularity ω, which minimizes the usage of radio resources

$$\mathcal{C}_{\text{tot}}[g] = \int f(\omega)\mathcal{C}\big(g(\omega), \omega\big)d\omega \tag{4}$$

for the file population, for a given available storage space. The caching selection function g takes values in the set of of caching methods discussed in Section 3, expanded with the *BS only* method. Note that $\mathcal{C}_{\text{tot}}$ is a functional of the categorical function g.

The underlying assumption is that $f(\omega)$ remains constant over a time which is long as compared to average node life time T. Accordingly, the caching decision is done for an ensemble of D2D caching nodes, not for an instance with certain individual nodes. Both the actual usage of radio resources, and the available amount of storage may vary according to the instance of storing nodes, but decisions have to be made on the statistical characteristics of the usage of radio resources and available storage in the ensemble.

The cost of radio resources $\mathcal{C}\big(g(\omega), \omega\big)$ will be called *transmission cost* for short. It depends, in addition to the file popularity and caching method, which determine the frequency and type of transmission, on the channel statistics, and the radio resource management principle used in

4. With a slight abuse of notation, we denote by n the number of nodes storing a replica in case of n-replication, and the length of an (n, k, d) MDS code used for the DSS. The meaning of n will always be clear from the context or clarified otherwise.

the cellular network and the D2D community. Statistical characteristics of user multiplexing, multiuser scheduling, multiantenna techniques, energy efficiency, and interference management have a role. We use ensemble averages to describe the statistics of costs. For transmissions within the D2D community, the cost of transmitting a unit of information to an arbitrary non-caching node in the cluster from its i^{th} nearest caching node when there are n caching nodes in total in the cluster is denoted by $\mathcal{C}(i,n)$. The expected cost of retrieving it from the base station is $\mathcal{B}$.

For simplicity, we formulate a caching optimization based on average memory consumption, and assume an upper limit S of reliable ensemble memory available in the community. With $\mathcal{S}\big(g(\omega),\omega\big)$ the average memory consumed by a file of popularity ω cached with method g, we have a storage constraint

$$h[g] = \int f(\omega)\ \mathcal{S}\big(g(\omega),\omega\big)d\omega - S \leq 0\ . \tag{5}$$

The optimization problem to be solved is thus

$$\begin{aligned} &\text{minimize} && \mathcal{C}_{\text{tot}}[g] \\ &\text{subject to} && h[g] \leq 0 \end{aligned} \tag{6}$$

and it is solved over all possible caching method selection functions $g(\omega)$. With a finite population of files, this problem can be formulated as a linear integer program, with assignment-type constraints binding the categorical variables g, and the complicating constraint $h[g]$ combining the optimizations of the files with different ω. Lagrangian relaxation can be applied to the complicating constraint [42], yielding the Lagrangian function

$$\chi_{\text{tot}}[g] = \mathcal{C}_{\text{tot}}[g] + \sigma h[g]\ , \tag{7}$$

where σ is the Lagrange multiplier. Inserting $\mathcal{C}_{\text{tot}}[g]$ from (4) and $h[g]$ from (5), the Lagrangian can be rewritten as

$$\chi_{\text{tot}}[g] = \int f(\omega)\chi(g,\omega)d\omega\ -\ \sigma S\ , \tag{8}$$

in terms of a cost function

$$\chi(g,\omega) = \mathcal{C}(g,\omega) + \sigma\mathcal{S}(g,\omega) \tag{9}$$

for a fixed file popularity ω and a caching method $g(\omega)$ selected for a file of this popularity. These file-specific costs will be computed in Section 4.2.

A tractable way to approach the optimization problem (7) is then to use Lagrangian duality. For this, we first find the dual function

$$\mathcal{X}(\sigma) = \min_{g} \chi_{\text{tot}}[g]\ , \tag{10}$$

and then find the price

$$\sigma^* = \arg\max_{\sigma\geq 0} \mathcal{X}(\sigma) \tag{11}$$

that maximizes the dual function. According to weak duality, the primal optimum $\mathcal{C}_{\text{tot}}[g^*]$, which is the smallest overall radio resource cost, achieved with the optimal caching policy g^*, is lower bounded by $\mathcal{X}(\sigma^*)$. If $\mathcal{C}_{\text{tot}}[g^*] > \mathcal{X}(\sigma^*)$, it indicates that with dual optimization, the constraint (5) is not fulfilled with equality, meaning that some storage space is left unused.

Applying duality considerably simplifies the overall cost minimization. When constructing the dual function (10), the discrete minimization over g can be separately performed for each file,

$$\min_{g(\omega)} \int f(\omega)\chi(g(\omega),\omega)\ d\omega = \int f(\omega) \min_{g} \chi(g,\omega)\ d\omega\ . \tag{12}$$

Thus instead of jointly optimizing over the caching policy $g(\omega)$ for all files with different popularity ω, it is sufficient to find the smallest file-specific cost $\min_g \chi(g,\omega)$ for each file with fixed popularity ω separately.

This file-specific optimization is performed with cost function (9), where the Lagrange multiplier σ appears as a *storage price* that can be seen as the cost of storing one unit of data for one unit of time. Accordingly, $\chi(g,\omega)$ can be seen as an effective cost combining transmission and storage cost for a single file. Joint optimization over the contributions of multiple files would then be performed in the dual maximization step (11). The role of the storage price σ is thus to couple the contributions of the different files in the overall optimization (7). Below, we concentrate on computing these file-specific costs, as functions of σ and ω.

4.2 Effective Cost for a File

The essential task for caching optimization is thus to find the expected cost for a file with popularity ω for the different storage methods, as a function of the storage price σ. We directly use the expected numbers of nodes to perform calculations. We later verify the validity of this approach with computer simulations in Section 5.

4.2.1 Simple Caching

The dynamics of the system under simple caching are modeled according to the Markov chain in Figure 3. Instead of a full steady state analysis of the chain, for the sake of simplicity, we derive an approximation for the expected cost.

When the file is cached, there is one node caching the entire file with no redundancy, so the cost of repair vanishes. There are, on average, $m-1$ nodes in the cluster generating requests as the single caching node does not need to download the file itself. Thus, the expected number of requests during the lifetime of the caching node is $(m-1)\omega T = (m-1)\frac{\omega}{\lambda}$. Once the caching node leaves the cluster, the next file request will be directed to the base station. The expected time in which this happens is approximately[5] $\frac{1}{m\omega}$, and an expected number of $(m-1)\frac{\omega}{\lambda}+1$ requests, including the local file retrievals in the cluster and the remote retrieval from the base station, are generated in time $T + \frac{1}{m\omega} = \frac{1}{\lambda} + \frac{1}{m\omega}$.

5. Strictly speaking, when the caching node has left the cluster, we should take the transient period in which the system returns back to steady state into account to find the exact expected value of the cost of simple caching. Since $\lambda < \omega$ and m is large, though, this approximation is accurate enough for our purposes as will be demonstrated later by the numerical results.

Using the approximation[6] $E\left(\frac{X}{Y}\right) \approx E\left(\frac{E(X)}{E(Y)}\right)$, where X, Y are two random variables and $E(\cdot)$ denotes expectation, the cost of simple caching can be approximated as

$$\chi(\text{Simple Caching}) \approx \frac{(m-1)\omega T \cdot \mathcal{C}(1,1) + \mathcal{B} + \sigma}{T + \frac{1}{m\omega}}. \quad (13)$$

The accuracy of this approximation, in the special cases considered in this work, is verified by numerical results in Section 5. Note that there is nothing to optimize in this caching method.

4.2.2 Replication

When replication is used, we assume n nodes storing an entire replica of the file. On average, there are $m-n$ empty nodes each generating file requests at rate ω. For reconstructing the file, the requesting node contacts the nearest storage node, so that the reconstruction cost is $\mathcal{C}(1,n)$. Thus, the reconstruction cost becomes

$$\chi_1 = (m-n)\omega\mathcal{C}(1,n).$$

To repair a failed node, a newcomer node contacts the nearest caching node out of the surviving $n-1$ storage nodes. The cost for a repair is hence given by $\mathcal{C}(1,n-1)$. There are n storage nodes each failing at rate λ, so the total repair cost becomes

$$\chi_2 = n\lambda\mathcal{C}(1,n-1).$$

The storage cost in this scenario is simply

$$\chi_3 = n\sigma\ ,$$

so the cost of replication is

$$\chi(\text{Replication}) = \chi_1 + \chi_2 + \chi_3\ . \quad (14)$$

The only parameter to be optimized for replication is the number of replicas n. Increasing n decreases the expected distances between the nodes and the number of empty nodes that request the file, but increases the total failure rate, and consequently the total repair cost, and the total storage cost. Note that similar observations have been made before for similar distance-dependent cost functions, see, *e.g.*, [11] and the references therein. To minimize the cost of replication it is crucial to find a suitable value of n, as will be demonstrated later in this work.

4.2.3 Regenerating Codes

In a system operating under this scheme, there are both storage nodes storing a fragment of the data file and empty nodes present in the cluster. We hence need to consider two types of requests. When one of the n storage nodes requests the file, it contacts $k-1$ out of the remaining $n-1$ storage nodes and downloads α units of data from each. The cost of this yields

$$\chi_4 = n\omega\alpha\sum_{i=1}^{k-1}\mathcal{C}(i,n-1).$$

When one of the empty nodes requests the file, k out of the n storage nodes need to be contacted. The cost of this is

$$\chi_5 = (m-n)\omega\alpha\sum_{i=1}^{k}\mathcal{C}(i,n)$$

since the expected number of empty nodes in the cluster is $m-n$.

When a storage node is lost, one of the empty nodes acts as the newcomer, contacts d of the remaining $n-1$ surviving nodes, and downloads β units of data from each, thus generating a total repair bandwidth of $\gamma = d\beta$. Thereby, the repair cost becomes

$$\chi_6 = n\lambda\beta\sum_{i=1}^{d}\mathcal{C}(1,n-1).$$

Since the storage cost using regenerating codes is simply

$$\chi_7 = n\sigma\alpha,$$

the total cost of using regenerating codes amounts to

$$\chi(\text{Regenerating}) = \chi_4 + \chi_5 + \chi_6 + \chi_7\ . \quad (15)$$

Here, α and β are functions of (k,d) given by (2) for MSR codes and (3) for MBR codes.

We immediately see that the same observations about varying the storage degree n that we made for replication apply to (15) as well. For regenerating codes we also need to choose between the MSR and MBR points, and find the optimal values of k and d, to minimize the cost. Maximizing the repair degree d minimizes both α and β for MBR and β for MSR, and maximizing the reconstruction degree k minimizes the amount of redundancy for MSR. However, high values of k and d imply that distant nodes need to be contacted. Therefore, we conclude that naively ignoring the distance-dependency and only optimizing the system with regard to the amount of data traffic does not necessarily imply the lowest cost.

4.3 Transmission costs in a disk-shaped cluster

To get a quantitative handle on the transmission cost, a model taking into account distances between nodes is needed. Instead of detailed analysis of system performance with scheduling, user multiplexing, power control, link adaptation, retransmissions and interference management, averaged over an ensemble of D2D storing nodes and conventional cellular usage, we consider average path loss as a catch-all measure indicating the radio resources needed for a transmission. When path loss is large, one either needs to increase transmit power, which consumes energy, and spreads interference to a larger domain, or to use more time/frequency domain resources. At low signal-to-noise ratio, transmit power can be directly exchanged for time/frequency resources.

We consider a special case of the system model in Figure 1. There is a cluster of nodes uniformly distributed in

6. This follows from the Taylor series expansion of $f(x,y) = \frac{x}{y}$ centered at the point $(E(X), E(Y))$ when y has support on $[0,\infty)$. This expansion can be truncated to $E(X/Y) \approx E(X)/E(Y) - \text{Cov}(X,Y)/E(Y)^2 + \text{Var}(Y)E(X)/E(Y)^3$ [43]. In the interest of space, instead of providing a full analysis of the error term, we will demonstrate the predictive ability of our estimate through numerical simulations, see the figures in Section 5.

a disk of radius $r \ll v$, and a base station located at a distance v from the center of the caching cluster. The path loss between two nodes is assumed to be

$$l = x^{\Gamma},$$

where x is the distance between the nodes and Γ is the path loss exponent. We consider two different pathloss exponents; for the downlink from the base station to the nodes in the cluster we use Γ_{BS}, for communications in D2D mode we use Γ_{D2D}.

We assume that there are n uniformly distributed storage nodes present in the disk of radius r. The transmission cost for cellular communication is given by the expected path loss between the base station and a node in the cluster.

This is found by integrating the complementary cumulative distribution function of the distance [47], [48]:

$$\mathcal{B} = \Gamma_{\text{BS}} \int_0^{v+r} x^{\Gamma_{\text{BS}}-1} \left(1 - \frac{A(x,r,v)}{\pi r^2}\right) dx. \tag{16}$$

Within the storage community, we are interested in the path loss between an arbitrary node in the disk and its i^{th} nearest caching node. To calculate it, we shall need the following geometric results.

Consider two circles of radii R and $r \leq R$ with centers separated by distance v. The chord connecting the cusps of the lens has length [44]

$$\mu \coloneqq \mu(R,r,v) \coloneqq \frac{1}{v}\sqrt{(r+R-v)(r-R+v)(-r+R+v)(r+R+v)}$$

and the area of the circular segment of a circle with radius x and chord length μ is [45]

$$\eta(x,\mu) \coloneqq x^2 \sin^{-1}\left(\frac{\mu}{2x}\right) - \frac{\mu}{2}\sqrt{x^2 - \frac{\mu^2}{4}},$$

where we have used equations (13) and (15) in [45] as well as the well-known identities $\sin(2x) = 2\sin(x)\cos(x)$ and $\sin^2(x) + \cos^2(x) = 1$. Figure 4 illustrates the overlapping circles.

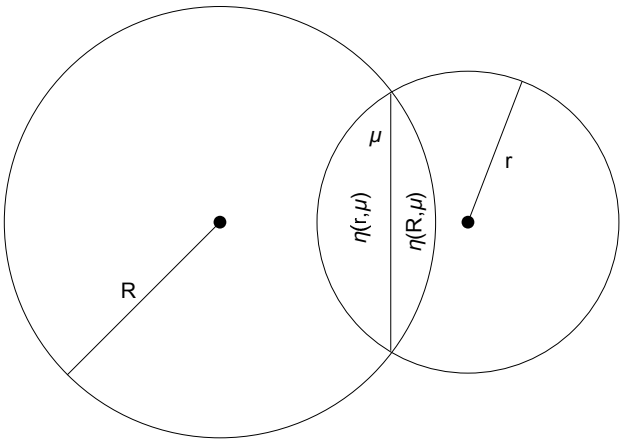


Fig. 4: Intersecting circles with radii R and $r < R$. The chord length is μ and the areas of the circular segments are $\eta(r,\mu)$ and $\eta(R,\mu)$.

If the smaller circle with radius r is completely inside the larger circle, the intersection area of the two circles is πr^2. If the smaller circle is mostly inside the larger one, the circular segments of the circles are overlapping and we need to subtract the area of the circular segment of the smaller circle from the total area of the smaller circle and then add the circular segment of the larger circle to find the total intersection area of the two circles. If only a small part of the small circles is inside the large circle, the overlapping area is simply the sum of the circular segments of the circles. If the circles are separated by a distance larger than the sum of their radii, they do not intersect.

The intersection area $A(R,r,v)$ of the two circles can thus be written as

$$A(R,r,v) = \begin{cases} \pi r^2 & \text{if } v \leq R - r \\ \pi r^2 - \eta(r,\mu) + \eta(R,\mu) & \text{if } R - r < v \leq \sqrt{R^2 - r^2} \\ \eta(r,\mu) + \eta(R,\mu) & \text{if } \sqrt{R^2 - r^2} < v \leq r + R \\ 0 & \text{if } v > r + R. \end{cases}$$

Note that $A(R,r,v)/(\pi r^2)$ is the probability that the distance between the base station and a node in the cluster is smaller than R.

Now consider a node in the cluster at distance t from the origin of the disk. Using the computed area of intersection, we can find the probabilities needed for our calculations. Of interest for our purposes is the expected distance between the node at distance t and its q^{th} nearest caching node when there are n caching nodes inside the disk in total.

The probability that the distance between the node and its q^{th} nearest neighbor is larger than x is [46, Eqn (1)]

$$\mathcal{I}(n,q,r,t,x) = \sum_{i=0}^{q-1} \binom{n}{i} \left(\frac{A(r,x,t)}{\pi r^2}\right)^i \left(1 - \frac{A(r,x,t)}{\pi r^2}\right)^{n-i}$$

if $0 \leq x \leq r$, and

$$\mathcal{I}(n,q,x,t,r) = \sum_{i=0}^{q-1} \binom{n}{i} \left(\frac{A(x,r,t)}{\pi r^2}\right)^i \left(1 - \frac{A(x,r,t)}{\pi r^2}\right)^{n-i}$$

if $x > r$. These results follow from the fact that when any i-subset, with $i \in [0, q-1]$, of the n nodes is inside a circle with radius x with its center a distance t away from the center of the disk, the q^{th} nearest neighbor of the node is at least distance x away from the node. Thus, the above equations represent the complementary cumulative distribution function of the distance. The results in [47], [48] can be used to find the expected value of the distance by integrating this function as

$$E(n,q,r,t) = \int_0^r \mathcal{I}(n,q,r,t,x)dx + \int_r^{r+t} \mathcal{I}(n,q,x,t,r)dx$$

and the results in [47], [48] can again be used to find the Γ^{th} power of the distance as

$$\mathcal{E}_{\Gamma}(n,q,r,t) = \Gamma\left(\int_0^r x^{\Gamma-1}\mathcal{I}(n,q,r,t,x)dx + \int_r^{r+t} x^{\Gamma-1}\mathcal{I}(n,q,x,t,r)dx\right).$$

The probability distribution function of the distance between the center of the disk and the node is $F_T(t) = \pi t^2/(\pi r^2) = t^2/r^2$ as the radius of the disk is r. The corresponding probability density function is thus $f_T(t) = 2t/r^2$. The expected intra-cluster transmission cost is then given by

$$\mathcal{C}(q,n) = \frac{2\Theta}{r^2}\int_0^r t\mathcal{E}_{\Gamma_{\text{D2D}}}(n,q,r,t)dt, \tag{17}$$

where we have added an overall factor Θ to reflect the possibly different value of radio resources in the base station, and the D2D cluster. There may, *e.g.*, be multiple D2D clusters in a macro cell, reusing the same time/frequency resources, while downlink users in a macro cell may be orthogonally scheduled. This would lead to $\Theta < 1$, reflecting increased spatial reuse of resources. On the other hand, managing a caching cluster may require considerable signaling overhead, as compared to straight forward downlink transmission. Also, link adaptation and other radio resource management protocols may be less effective in D2D than in cellular transmission, leading to increasing need for retransmission. Taking this into account may render $\Theta > 1$.

5 Numerical Results

In this section, we illustrate the performance of the five considered caching methods with respect to the derived performance metric with the help of numerical results. The parameter of replication n and the parameters of regenerating codes (n,k,d) are chosen so that the cost function is minimized. Further, we study the potential benefits of caching from an operator's point of view.

For all the cases in this section, we fix $m = 100$, $r = 1$, $v = 20$, $\Gamma_{\text{BS}} = 3.76$ [49] and $\Gamma_{\text{D2D}} = 4$ [49], while σ is varied. The user arrival rate $\lambda = 1$, if not otherwise stated. We choose $n \in [2,6]$ for replication, and $n \in [3,6]$ for regenerating codes, so that the cost is minimized for a given ω. When $\Theta = 1$, radio resources consumed in the D2D cluster are considered as valuable as cellular resources. When $\Theta < 1$, radio resources in the D2D cluster are considered less valuable than in cellular downlink; the D2D resources may be reused in multiple clusters inside the cell, while with $\Theta > 1$, signaling overhead makes storage-related transmissions in the cluster more expensive than cellular transmissions. The storage price is given by σ.

All the costs of the considered caching methods are compared to the method of retrieving data through the cellular downlink, called *BS only*. The cost function χ is given by (13) for simple caching, by (14) for replication, and by (15) for regenerating codes. In these, the expected intra-cluster transmission cost is given by (17), and the cost between the base station and a node in the cluster is (16). We choose (n,k,d) such that the values of χ for the respective caching methods are minimized.

The theoretical curves (solid lines) in Figures 5, 6, 7, 8, 9, and 10 are numerical values using the derived cost functions. In the figures, we plot the relative cost $C = \chi/\mathcal{B}$ compared to *BS only*, as a function of file popularity ω. Plots are double-logarithmic, with both C and ω expressed in dB-scale. Thus, the smaller C, the better the caching method performs compared to the benchmark *BS only*.

The simulated values (dots) are obtained by computing steady state averages of long event-based Monte Carlo simulations for the Markov chains depicted in Figure 3 for simple caching and Figure 2 for regenerating codes and replication, to verify the theoretical calculations. The initial number of nodes in the cluster is $m = 30$ or $m = 100$ and the file is cached when the simulation starts. In addition, multicaching is simulated.

The theoretical curves (solid lines) in the figures are numerical values using the cost functions. The simulation results (dots) are obtained by computing steady state averages of long event-based Monte Carlo simulations for the Markov chains in Figure 3 for simple caching and Figure 2 for regenerating codes and replication.

Figure 5 illustrates the costs of the caching methods when storage cost is relatively high. For low file popularities, caching is not useful and the traditional downlink ought to be used. Simple caching is preferred for a small interval in the figure around popularity $\omega \approx 10^{1.80}$. Furthermore, when storage cost is high, MSR is preferred, and provides drastic cost savings especially for high file popularities. Recall that MSR minimizes the storage space requirements while providing redundancy in the system with a relatively low repair bandwidth.

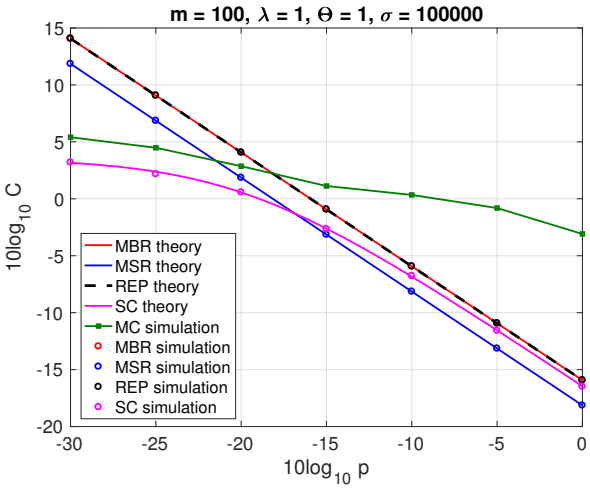


Fig. 5: Total costs when the storage price σ is high.

In Figure 6, the storage price is much lower. When the file popularity is low, replication yields the best results, while for popular files, multicaching is optimal. When the storage cost is low, transmission costs dominate the total cost and it is important to minimize the expected transmission costs. When replication is used, only the nearest caching node needs to be contacted, as is the case for multicaching. The additional benefit of multicaching is that, for high file popularities, the cluster fills up with replicas and the expected transmission distance can become lower than that of replication. Thus, for high file popularities, multicaching is preferred.

Compared to the setting of Figure 5, the D2D transmission cost Θ is much lower in the setting of Figure 7. This reflects radio resource management where the same D2D resources are used in multiple clusters in a cell, and signaling overhead due to D2D caching is low. We see how

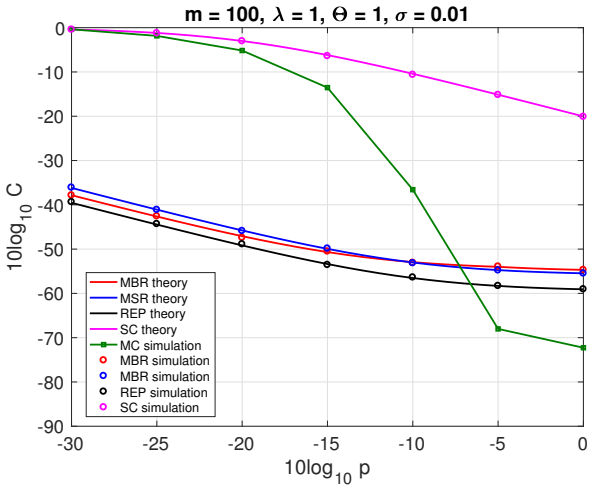


Fig. 6: Total costs for a low storage price σ.

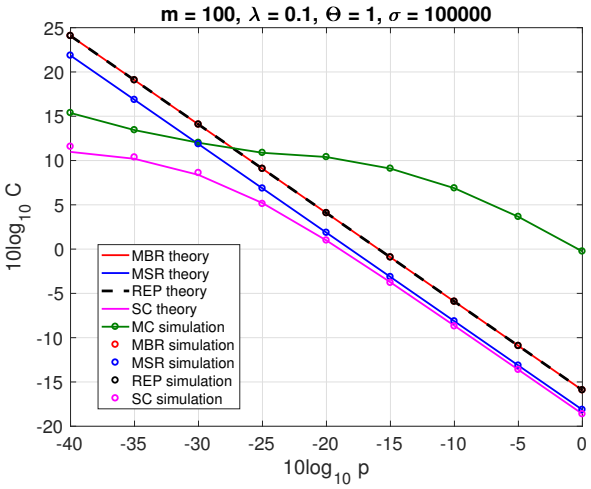


Fig. 8: Totals costs for a low churn rate λ.

simple caching and multicaching yield large cost savings even for very low file popularities. For higher popularities, MSR is optimal, and can offer even more than 60 dB of cost savings.

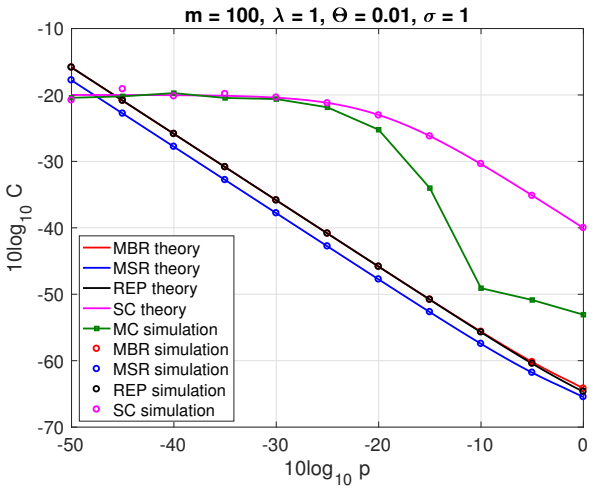


Fig. 7: Total costs for a moderate storage price σ when the relative D2D transmission cost Θ is low.

Compared to the setting of Figure 5, the node arrival rate λ is much lower in the setting of Figure 8. This corresponds to a case where the churn is much lower. We see that, for high file popularities, simple caching is the preferred method and not MSR as in the setting of Figure 5. We note that when churn is high, it is important to protect data against node failures with redundancy, while for low churn, simple caching suffices.

In Figure 9, the churn rate is even higher than in the setting of Figure 5. Now the improvement in cost achieved by MSR coding, especially compared to the performance of simple caching, is even more pronounced.

In Figure 10, we plot a case where the expected number of nodes in the cluster is low, that is, $m = 30$. The results are otherwise very similar to those of Figure 5 but we see that the performance of multicaching is less impressive than for the higher expected number of nodes $m = 100$.

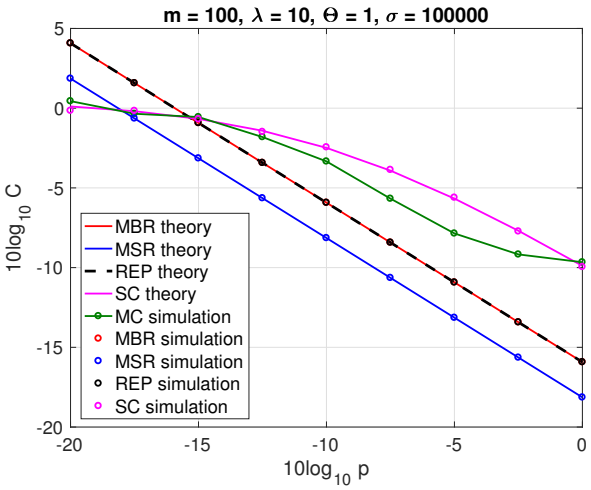


Fig. 9: Total costs for a high churn rate λ.

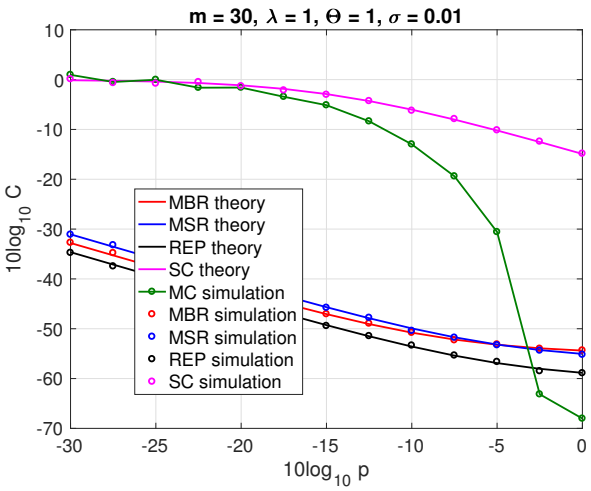
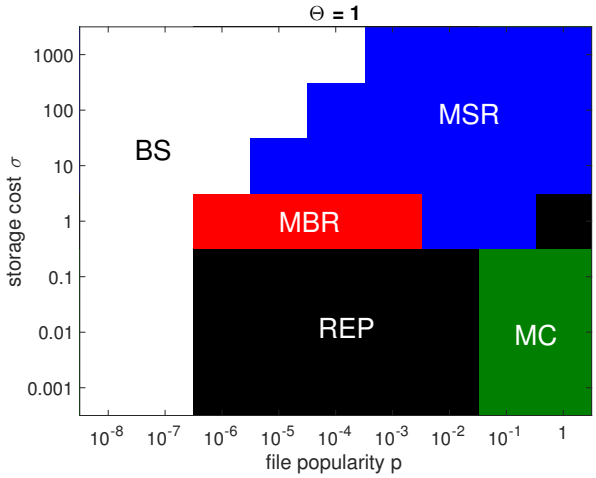
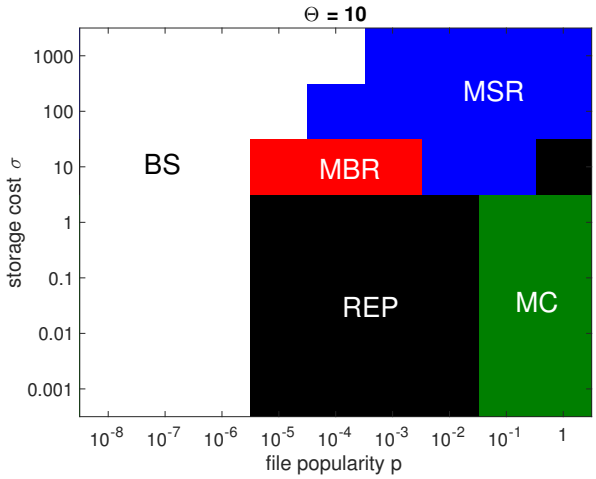


Fig. 10: Total costs for a low expected number of nodes in the cluster m.



a) Moderate relative D2D cost, $\Theta = 1$.



b) High relative D2D cost, $\Theta = 10$.

Fig. 11: Schematic view of the caching methods that minimize the total cost of transmission and storage for a given file-popularity/storage-price pair (ω, σ). Each of the methods Minimum storage regenerating (MSR) code, minimum bandwidth regenerating (MBR) code, replication (REP) and multicaching (MC), is useful for certain parameter values. When none of these methods is useful, the base station (BS) should be directly contacted.

Figure 11 present the optimal caching methods for transmission costs $\Theta = 1$ and $\Theta = 10$ with $\lambda = 1$. The latter represents a conservative estimate, where signaling and other protocol overhead is considered to make transmissions within the D2D cluster 10 times more expensive than cellular downlink transmissions for the same requirement of radio resources. For very low file popularities, caching is not useful at all. When the file popularity exceeds a certain threshold, caching with redundancy can be exploited. This threshold is a function of the transmission cost Θ and the storage price σ. Naturally, increasing the D2D transmission cost Θ increases this threshold.

Increasing the transmission cost Θ is equivalent to decreasing the storage price σ, except for simple caching. Therefore, for instance, multicaching outperforms MSR and replication already for $\sigma = 1$ in Figure 11 b), which does not occur in Figure 11 a).

When the storage cost is high, MSR is optimal. For moderate storage costs also MBR can be the most suitable option. When storage is not expensive, replication is preferred.

Note that in practice, caching is most useful for scenarios where file popularity is high and storage is inexpensive. In the figures we see that under such circumstances multicaching yields the lowest cost.

Finally, in Figure 12 the optimal MSR code parameters (n, k, d) are plotted for transmission cost $\Theta = 10$ and storage price $\sigma = 100$ with churn rate $\lambda = 1$ for the values for which MSR is the optimal caching method. This corresponds to the second horizontal line in Figure 11 b). The number of cached fragments is increased when the file popularity increases. This allows for shorted expected transmission distances at reconstruction and repair at the expense of a higher storage space consumption.

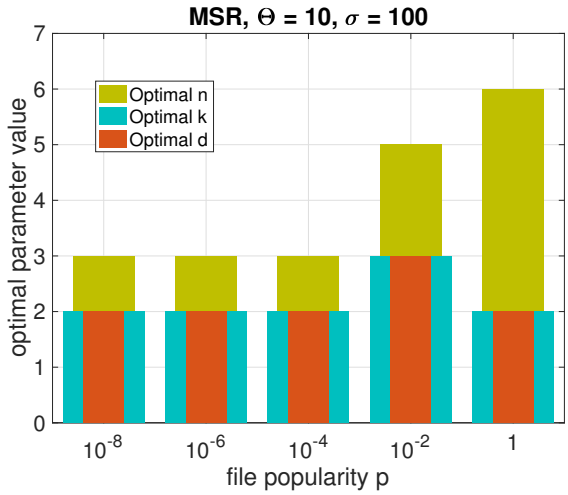


Fig. 12: Optimal MSR code parameters for $(\Theta, \sigma) = (10, 100)$ and $\lambda = 1$.

6 Conclusions

We have studied the prospective benefits of distributed storage coding in a D2D caching cluster, where communication cost grows with distance due to increasing pathloss. Our main objective has been optimizing the overall resource usage of the network. Sharing a limited storage size between multiple files yields a price for storage. For a given file, this has to be balanced against the saving of radio resources from using intra-cluster transmissions, to select the optimal caching method for a file. We have found that distributed storage coding can yield large savings in the usage of radio resources, *e.g.*, transmission energy, as compared to traditional downlink data transmission. Differences in spatial distribution of interference and related reuse of resources as well as in signaling overhead and protocol efficiency between D2D and cellular transmission have been taken into account by an overall cost multiplier.

We have derived an analytical method for choosing the optimal caching method in an environment that can be characterized by power-law transmission cost.

D2D caching is not beneficial when either file popularity is low or storage cost is high. On the contrary, when file popularity is high and storage cost is low, each user requesting a file should cache and distribute it. Actively maintaining redundancy with repetition coding or regenerating codes can also offer significant cost savings for most combinations of file popularity and storage cost. The file-specific optimization derived here can be directly used in multi-file storage optimization.

Acknowledgments

The authors would like to thank Majid Gerami, Ejder Baştuğ, Toni Ernvall, Pasi Lassila, and Lasse Leskelä for fruitful discussions.

References

[1] J. Pääkkönen, C. Hollanti, and O. Tirkkonen, "Device-to-Device Data Storage for Mobile Cellular Systems," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, pp. 671–676, December 2013.

[2] J. Pääkkönen, C. Hollanti, and O. Tirkkonen, "Device-to-Device Data Storage with Regenerating Codes," in *Proc. 8th International Workshop on Multiple Access Communications (MACOM)*, pp. 57–69, September 2015.

[3] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015–2020," White Paper, http://goo.gl/l77HAJ, 2014.

[4] P. Jänis, C.-H. Yu, K. Doppler, C. Ribeiro, C. Wijting, K. Hugl, O. Tirkkonen, and V. Koivunen, "Device-to-Device Communication Underlaying Cellular Communications Systems," *International Journal of Communication*, vol. 2, no. 3, pp. 169–178, June 2009.

[5] K. Doppler, M. P. Rinne, P. Janis, C. Ribeiro, and K. Hugl, "Device-to-Device Communications; Functional Prospects for LTE-advanced Networks," in *Proc. IEEE International Conference on Communications Workshops, 2009*, pp. 1–6, June 2009.

[6] N. Reider and G. Fodor, "A Distributed Power Control and Mode Selection Algorithm for D2D Communications," *EURASIP Journal on Wireless Communications and Networking*, pp. 1–25, August 2012.

[7] C.-H. Yu, O. Tirkkonen, K. Doppler, and C. Ribeiro, "On the Performance of Device-to-Device Underlay Communication with Simple Power Control," *Proc. IEEE Information Theory Workshop (ITW)*, pp. 1–5, September 2013.

[8] A. Asadi, Q. Wang, and V. Mancuso, "A Survey on Device-to-Device Communication in Cellular Networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 1801–1819, Fourthquarter 2014.

[9] S. Gitzenis, G. S. Paschos, and L. Tassiulas, "Asymptotic Laws for Joint Content Replication and Delivery in Wireless Networks," *IEEE Transactions on Information Theory*, vol. 59, no. 5, pp. 2760–2776, May 2013.

[10] M. A. Maddah-Ali and U. Niesen, "Fundamental Limits of Caching," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.

[11] E. Altman, K. Avrachenkov, and J. Goseling, "Distributed Storage in the Plane," in *Proc. International Federation for Information Processing (IFIP) Networking Conference*, pp. 1–9, June 2014.

[12] 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 Transactions on Wireless Communications*, vol. 13, no. 7, pp. 3665–3676, July 2014.

[13] E. Baştuğ, M. Bennis, and M. Debbah, "Living on the Edge: The Role of Proactive Caching in 5G Wireless Networks," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 82–89, August 2014.

[14] S-W. Jeon, S-N. Hong, M. Ji, and G. Caire, "Caching in Wireless Multihop Device-to-Device Networks," in *Proc. IEEE International Conference on Communications (ICC)*, pp. 6732–6737, April 2015.

[15] M. Gerami, X. Ming, and M. Skoglund, "Partial Repair for Wireless Caching Networks With Broadcast Channels," *IEEE Wireless Communications Letters*, vol. 4, no. 2, pp. 145–148, April 2015.

[16] B. Blaszczyszyn and A. Giovanidis, "Optimal Geographic Caching in Cellular Networks," in *Proc. IEEE International Conference on Communications (ICC)*, June 2015.

[17] C. Yang, Y. Yao, Z. Chen, and B. Xia, "Analysis on Cache-enabled Wireless Heterogeneous Networks", *IEEE Transactions on Wireless Communications*, vol. 15, no. 1, pp. 131–145, January 2016.

[18] Y. Guo, L. Duan, and R. Zhang, "Cooperative Local Caching and File Sharing under Heterogeneous File Preferences," in *Proc. IEEE International Conference on Communications (ICC)*, pp. 1–6, May 2016.

[19] J. Rao, H. Feng, C. Yang, Z. Chen, and B. Xia, "Optimal Caching Placement for D2D Assisted Wireless Caching Networks," in *Proc. IEEE International Conference on Communications (ICC)*, May 2016.

[20] M. Ji, "Fundamental Limits of Caching Networks: Turning Memory into Bandwidth," Doctoral dissertation, Faculty of the USC Graduate School, University of Southern California, 2015.

[21] M. Ji, G. Caire, and A. F. Molisch. "Wireless Device-to-Device Caching Networks: Basic Principles and System Performance," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 1, pp. 176–189, January 2016.

[22] M. Afshang and H. S. Dhillon, "Optimal Geographic Caching in Finite Wireless Networks," in *Proc. IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, July 2016.

[23] D. Malak, M. Al-Shalash, and J. G. Andrews, "Spatially Correlated Content Caching for Device-to-Device Communications," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, July 2016.

[24] Y. Pan, C. Pan, H. Zhu, Q. Z. Ahmed, M. Chen, and J. Wang, "On Consideration of Content Preference and Sharing Willingness in D2D Assisted Offloading," arXiv:1702.00209, 2017.

[25] R. Wang, J. Zhang, S. H. Song, and K. B. Letaief, "Mobility Increases the Data Offloading Ratio in D2D Caching Networks," arXiv:1702.05880, 2017.

[26] J. Zhang and P. Elia, "Fundamental Limits of Cache-Aided Wireless BC: Interplay of Coded-Caching and CSIT Feedback," *IEEE Trans. Inf. Theory*, vol. 63, no. 5, pp. 3142–3160, May 2017.

[27] J. Ott and M. Pitkänen, "DTN-based Content Storage and Retrieval," in *Proc. IEEE WoWMoM Workshop on Autonomic and Opportunistic Communications (AOC)*, pp. 1–7, June 2007.

[28] V. Lenders, G. Karlsson, and M. May, "Wireless Ad Hoc Podcasting," in *Proc. IEEE Conference on Sensor, Mesh, and Ad Hoc Communications and Networks (SECON)*, pp. 273–283, June 2007.

[29] M. Ji, G. Caire and A. Molisch. "Fundamental Limits of Distributed Caching in D2D Wireless Networks," in *Proc. IEEE Information Theory Workshop (ITW)*, pp. 1–5, September 2013.

[30] B. Chen and C. Yang, "Energy costs for traffic offloading by cache-enabled D2D communications", in *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–5, April 2016.

[31] M. Gregori, J. Gomez-Vilardebo, J. Matamoros, and D. Gündüz, "Wireless Content Caching for Small Cell and D2D Networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1222–1234, May 2016.

[32] M. Afshang, H. S. Dhillon, and P. H. J. Chong, "Fundamentals of Cluster-Centric Content Placement in Cache-Enabled Device-to-Device Networks," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, December 2015.

[33] J. Pedersen, A. Graell i Amat, I. Andriyanova, and F. Brännström, "Repair Scheduling in Wireless Distributed Storage with D2D Communication," in *Proc. Information Theory Workshop (ITW)*, pp. 69–73, October 2015.

[34] L. Wang, H. Wu, and Z. Han, "Wireless Distributed Storage in Socially Enabled D2D Communications," *IEEE Access*, March 2016, DOI: 10.1109/ACCESS.2016.2546685.

[35] A. O. Allen, *Probability, Statistics, and Queueing Theory: With Computer Science Applications*. Gulf Professional Publishing, p. 259, 1990.

[36] S. Tang and B. L. Mark, "Analysis of Opportunistic Spectrum Sharing with Markovian Arrivals and Phase-Type Service," *IEEE Transactions on Wireless Communications*, vol. 8, no. 6, pp. 3142–3150, June 2009.

[37] H.-N. Hung, P.-C. Lee, and Y.-B. Lin, "Random Number Generation for Excess Life of Mobile User Residence Time," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 3, pp. 1045–1050, May 2006.

[38] S. Thajchayapong, "Mobility Patterns in Microcellular Wireless Networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 1, pp. 52–63, January 2006.

[39] P. Harrison and N. M. Patel, "Performance Modelling of Communication Networks and Computer Architectures," *International Computer Science Series*, Addison-Wesley, 1992.

[40] A. G. Dimakis, P. B. Godfray, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network Coding for Distributed Storage Systems,"

IEEE Transactions on Information Theory, vol. 56, no. 9, pp. 4539–4551, September 2010.

[41] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal Exact-Regenerating Codes for Distributed Storage at the MSR and MBR Points via a Product-Matrix Construction," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5227–5239, August 2011.

[42] L.A. Wolsey, "Integer Programming", Wiley-Interscience, 1998.

[43] A. Stuart and K. Ord, *Kendall's Advanced Theory of Statistics*, vol. 1, pp. 351, Arnold, London, 1998.

[44] E. W. Weisstein, "Circle-Circle Intersection." From MathWorld – A Wolfram Web Resource. http://mathworld.wolfram.com/Circle-CircleIntersection.html

[45] E. W. Weisstein, "Circular Segment." From MathWorld – A Wolfram Web Resource. http://mathworld.wolfram.com/CircularSegment.html

[46] S. Srinivasa and M. Haenggi, "Distance Distributions in Finite Uniformly Random Networks: Theory and Applications," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 2, pp. 940–949, February 2010.

[47] P. Muldowney, K. Ostaszewski, and W. Wojdowski, "The Darth Vader Rule," *Tatra Mt. Math. Publ.*, no. 52, pp. 53–56, 2012.

[48] W. Feller, *An Introduction to Probability Theory and its Applications*, vol. 2, John Wiley & Sons, Inc. New York, 1966.

[49] C. Vlachos, V. Friderikos, and M. Dohler, "Optimal Virtualized Inter-Tenant Resource Sharing for Device-to-Device Communications in 5G Networks," *Mobile Networks and Applications*, pp. 1–10, 2017.