# Wireless Video Content Delivery through Distributed Caching and Peer-to-Peer Gossiping

Negin Golrezaei, Alexandros G. Dimakis, Andreas F. Molisch, Giuseppe Caire
Dept. of Electrical Eng.
University of Southern California
emails: {golrezae,dimakis,molisch,caire}@usc.edu

***Abstract*—We present a novel approach to handle the ongoing explosive increase in the demand for video content in wireless mobile devices. We show how distributed caching and collaboration between users and femtocell-like base stations *without* high-speed backhaul, which we call *helpers*, can greatly improve throughput without suffering from the backhaul bottleneck problem common to femtocells. We also investigate the role of collaboration *among* users - a process that can be interpreted as the mobile devices playing the role of helpers also. This approach allows an improvement in the video throughput without deployment of any additional infrastructure.**

**The efficiency of the caching approach depends on two key system properties: (i) the configuration of the "effective" distributed cache, i.e., which user can connect to which helpers and (ii) the popularity distribution of the video files. As a function of these properties, we consider the wireless distributed caching problem, i.e., which files should be cached by which helpers. For the uncoded helper problem, a strictly optimum solution of this problem is NP-hard, but good approximation algorithms can be found. Furthermore if distributed coding is used for the caching, exact optimization can be done with polynomial complexity.**

**For the configuration with mobiles-as-helpers problem, we adopt a simplified model wherein users are grouped into clusters within which communications is possible. A key question is the choice of the cluster dimension (collaboration distance), trading off spectral reuse with the probability of finding the desired video within the collaboration distance. Numerical optimization as well as general scaling laws are described.**

**Simulations show that our approach can improve the data throughput by as much as $400-500\%$ through addition of helpers, and more than an order of magnitude through the device-to-device communications.**

## I. Introduction

Over the past decade, the proliferation of smartphones, tablets, and other wireless devices has led to an explosive growth of wireless data traffic. While initial growth was mainly driven by web-browsing and associated applications, emphasis is now shifting to wireless delivery of video content, which is expected to lead to an increase in wireless data traffic by *two orders of magnitude* in the next few years. [1]. Current systems already have close-to-optimum technology for their physical link between transmitter and receiver. Thus, the most promising ways to achieve such an enormous increase in *area spectral efficiency* and thus data throughput, of such tremendous scale is a decrease in cell size, essentially bringing the content closer to the users. The common method of realizing this is the deployment of small base stations that enable high-density spatial reuse of communication resources [2]. Such pico- and femto-cell networks, which are usually combined with macrocells into a heterogeneous network, are receiving a lot of attention in the recent literature, (see e.g. [3] and references therein). However, a major practical obstacle to this approach is the requirement for high-speed backhaul to the cellular operator network [3].

In this paper we survey our on-going work [4]–[6] of using distributed storage and collaboration to address the backhaul bottleneck problem. Our main idea is to replace the femto-base stations by *s*mall base stations that have a (possibly wireless) low-bandwidth backhaul link but high storage capacity. These stations, which we henceforth call caching helpers, or simply *helpers*, form a wireless distributed caching infrastructure.

Further in our on-going work [6] we investigate using other mobile users as helpers: popular content is cached and through base station assisted device-to-device (BSA-D2D) gossiping is delivered to new users who request it. Distributed storage can be exploited at different levels and fueled by short-range high-throughput local connections. Predictability of the popular content and the massive distributed storage capacity yield that, with high probability, the video that any given user wishes at any given time is already in someone's cache in the same macro-cell. A novel wireless BSA-D2D wireless architecture takes care of delivering the desired content to the users within one short-distance hop through local D2D links. This architecture can lead to significant gains if there is sufficient content re-use, i.e., a significant fraction of the requests correspond to few popular files.

We now proceed to give an overview of our novel research contributions:

1) Femtocaching: How to optimize the file placement in helpers to maximize the probability of finding popular files locally in a physically distributed cache [4].
2) Coded Femtocaching: How the splitting of popular content into blocks and the encoding of these blocks allows the relaxation of an NP-complete problem into a convex optimization of linear program [5].
3) Device-to-Device Collaboration: How to turn other users into helpers by distributed caching of popular content in other mobile devices. The problem of finding the optimal collaboration distance and the scaling laws that can be achieved through this architecture [6], [7].

## II. FemtoCaching

We first consider the question how helper nodes can improve the video throughput. Figure 1 illustrates the system layout. The helpers form a wireless distributed caching infrastructure. More concretely, the helpers are placed in fixed positions in the cell and are assumed to have (i) large storage capacity, (ii) localized, high-bandwidth communication capabilities which enable high frequency reuse, and (iii) low- rate backhaul links which can be wired or wireless. They can cache popular files and serve requests from mobile User Terminals (UTs) by enabling localized communication. The key point is that *if there is enough content reuse, i.e. many users are requesting the same video content, caching can replace backhaul communication*. The helpers are working in conjunction with traditional macrocellular base stations,[1] which provide to the UTs the video files that cannot be obtained from the helpers. Clearly, the smaller the percentage of file requests that has to be fulfilled by the macrocell, the larger the number of UTs that can be served.

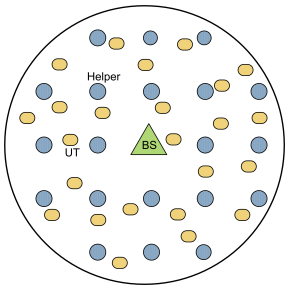


Fig. 1. An example of the single-cell layout. UTs are randomly distributed, while helpers can be optimally placed in the coverage region.

Let us now formulate the following *wireless distributed caching problem: for given popularity distribution, storage capacity in the helpers and wireless communication model for the BS to UTs downlink and helpers to UTs links, how should the files be placed in the helpers such that the average sum delay of all users is minimized?* For the limiting case that each UT can connect only to a single helper (because the distance between helpers is large), the solution is trivial: each helper should cache the most popular files, in sequence of popularity, until its cache is full. If the helper deployment is dense enough, UTs will be able to communicate with several such helpers and each sees a distributed cache that is the union of the helpers' caches. In this situation, the question on how to best assign files to different helpers becomes a much more complicated issue, because each UT sees a different, but correlated, distributed cache.

We consider in the following two versions of the distributed caching problem.

- In the first case, we assume that the video files are packetized and the corresponding packets are stored in the helpers' caches directly. We refer to this case as the *uncoded* distributed caching problem. Clearly, coding is used at the physical layer to achieve reliable communications. The uncoded nature of this problem lies in the fact that the raw data packets are directly stored in the caches.
- In the second case, we allow coding (i.e., using fountain or MDS codes), so that we can store in the helpers' caches parity symbols produced by encoding each video file, and each helper can store less than a complete video file. With fountain/MDS codes, the whole file can be recovered if enough parity symbols are received by the UT while the individual identity of the packets is not relevant. Rather, it is how many parity symbols of a given file are retrieved from the helpers in the reach of each UT. We call this case *coded* distributed caching problem.

In both cases, we consider the following system setup: there are $H$ helpers, $K$ user terminals, and a library of $N$ files, denoted by $\mathcal{F}$. The popularity distribution of the files conditioned on the event that a user makes a request is denoted by $P_n$, for $n = 1, \ldots, N$; often this popularity distribution can be approximated by a Zipf distribution [9]. The connectivity between users and helpers can be represented in a bipartite graph; one example is shown in figure 2. If there is an edge between helper $h$ and UT $k$, it means that UT $k$ can communicate reliably with helper $h$. We assume that the connectivity between UTs and helpers does not change during the transmission of a video file. This requires our users to be fairly static compared to the download time.

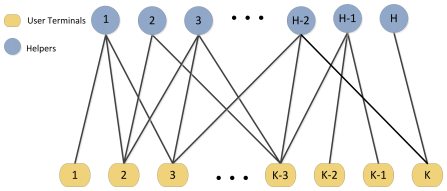


Fig. 2. Example of a connectivity bipartite graph indicating how UTs are connected to helpers

When a user requests some file $n$, it first asks its local helpers, i.e., helpers in the neighborhood of the user in the user-helper connectivity graph. If these local helpers have the requested file, they send it; otherwise, the BS handles the request, incurring a higher delay.

For the uncoded distributed caching problem, we show in [4] that finding the problem is NP-complete. Further, we express the problem as a maximization of a submodular function subject to matroid constraints. For the coded distributed caching problem, we show that the expected delay minimization problem can be formulated as a convex optimization problem [5]. By adding additional variables we can further reduce it to a linear program. The formulations are briefly sketched in Secs. II.A and II.B, respectively.

To verify the efficacy of our schemes in practical situations, in [4], [5] we present a detailed simulation of a university campus scenario covered by a single 3GPP LTE R8 cell and several helpers using a simplified 802.11n protocol. We

[1] for the sake of simplicity, we consider a single macro-BS, so that no inter-cell interference occurs.

simulate using the user request trace of YouTube videos from the Amherst campus [8] [9]. Our main finding is that there are very significant gains even when very simple caching algorithms are used.

### A. Uncoded Distributed Femtocaching

Due to the short distance between helper and UT, the data rate is very high, and the file transmission is approximately instantaneous. Thus, the average delay of UT $k$ is proportional to the probability of not finding the request file in the union of the helpers' caches in the reach of user $k$, i.e., the helpers in the neighborhood of user $k$ in the connectivity graph. With suitable normalization, the expected delay of user $k$ can be written as:

$$\bar{D}_k = \omega^k[1 - \sum_{n \in A_k} P_n], \tag{1}$$

where $\omega^k$ is the delay of downloading a file of fixed size from BS for user $k$ and $A_k$ is the union of the helpers' caches in the neighborhood of user $k$ in the connectivity graph of the network, i.e., $A_k = \cup_{h \in \mathcal{N}(k)} C_h$. $C_h$ is the set of files in the cache of helper $h$ and $\mathcal{N}(k)$ denotes the neighborhood of user $k$, i.e., all the helpers which user $k$ can reliably connect to. Consequently, $\sum_{n \in A_k} P_n$ is the probability that user $k$ requests one of the files that are accessible through its local helpers.

While the storage capacity of helper nodes can be large (on the order of Terabyte), it is not infinite, and certainly not all video files of possible interest to users can be stored. Rather, we wish to find out which files should be stored in which helpers such that the expected download time is minimized (notice that the same file can be placed several times in different helpers). Thus, the the optimization problem can be written as :

$$\begin{aligned} \text{maximize} \quad & \sum_{k=1}^{K} \omega^k \sum_{n \in A_k} P_n \\ \text{subject to} \quad & |C_h| \leq M, \quad \forall\, h \\ & A_k = \cup_{h \in \mathcal{N}(k)} C_h, \quad \forall\, k \end{aligned} \tag{2}$$

where the optimization is with respect to the sets $\{C_h \subseteq \mathcal{F} : \forall h\}$, $M$ is the cache capacity of each helper and $|C_h|$ is the cardinality of set $C_h$. The constraint indicates that no more than $M$ files are allowed to be placed in each helper's cache.

Solving the above optimization problem is NP-hard; we proved this in [4] by demonstrating that given a unit time oracle for our *Helper Problem* we can solve the *2- Disjoint Set Cover Problem* in polynomial time (a polynomial time reduction is denoted by $\leq_L$). In [11], it is proved that 2-disjoint set cover is NP-complete.

In [4] we furthermore formulate the problem as maximization of a monotone submodular function over matroid constraints. For this type of problem, algorithms exist that provide a solution that are provably within a constant factor from optimality. While the best of such algorithms come within $1/e$ of the optimum solution, they are rather complicated and might be difficult to implement in real-time. However, even a simple greedy algorithm is provably within $50\%$ of the optimum.

### B. Coded Distributed FemtoCaching

We now turn to the question of how to optimally place *fountain/MDS-encoded files* in the helpers' caches to minimize the total average delay. It turns out that by adding coding we relax the NP hard problem to a convex problem.

With fountain/MDS codes, user $k$ should receive $B$ coded symbols of file $n$ in the union of the caches of transmitters in $\mathcal{N}(k)$. If the first $j \leq |\mathcal{N}(k)|$ transmitters in set $\mathcal{N}(k)$ are enough for the recovering some file $n$ while just receiving the parity symbols of the first $j-1$ transmitters is insufficient, the delay for user $k$ because of file $n$ is equal to:

$$\bar{D}_k^{n,j} = \sum_{h \in \{h_1^k, \ldots, h_{j-1}^k\}} \omega_h^k \rho_{hn} + (1 - \sum_{h \in \{h_1^k, \ldots, h_{j-1}^k\}} \rho_{hn}) \omega_j^k \tag{3}$$

where $\omega_h^k$ is the average delay when user $k$ downloads file form its local helper $h$. $\rho_{hn}$ is the percentage of coded symbols for file $n$ in the cache of helper $h$. Clearly $\rho_{BSn}$ is equal to 1 for all $n$. $(1 - \sum_{h \in \{h_1^k, \ldots, h_{j-1}^k\}} \rho_{hn})$ is the amount of parity symbols that should be transmitted by the $j$th transmitter in $\mathcal{N}(k)$. The above expression can be used if $\sum_{h \in \{h_1^k, \ldots, h_{j-1}^k\}} \rho_{hn} < 1$ and $\sum_{h \in \{h_1^k, \ldots, h_j^k\}} \rho_{hn} \geq 1$. The matrix whose element in the $h$-th row and $n$-th column is $\boldsymbol{\rho} = [\rho_{hn}]$, and it is referred to as the placement matrix.

The delay $\bar{D}_k^n$ incurred by user $k$ because of downloading file $n$ can then be shown to be a pointwise maximum of affine functions and a convex function of $\boldsymbol{\rho}$ .

$$\bar{D}_k^n = \max_{j \in \{1,2,\ldots,|\mathcal{N}(k)|\}} \bar{D}_k^{n,j} \tag{4}$$

Thus, the placement optimization problem takes on the form:

$$\begin{aligned} \text{minimize} \quad & \sum_{k=1}^{K} \sum_{n=1}^{N} P_n \max_{j \in \{1,2,\ldots,|\mathcal{N}(k)|\}} \bar{D}_k^{n,j} \\ \text{subject to} \quad & \sum_{n=1}^{N} \rho_{hn} \leq M, \quad \forall\, h \\ & 0 \leq \rho_{hn} \leq 1, \quad \forall h, n \end{aligned} \tag{5}$$

where the optimization is with respect to $\boldsymbol{\rho}$. The first constraint indicates the cache capacity of each helper is equal to $M$ files. The second constraint shows the amount of parity symbols of file $n$ cached by a helper $h$ is normalized to the size of files. The objective function in (5) is convex function since it is positive weighted sum of convex piecewise linear functions [14]. Hence, we have minimization of convex function over linear constraints which can be solved using convex optimization.

## III. Base-station controlled D2D communications

We now consider the situation where there are no fixed helper nodes, i.e., no new (in addition to the standard macro-BS) infrastructure at all. Rather, to distribute popular content in

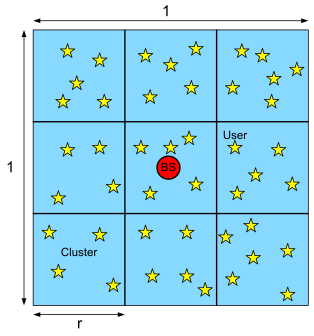


Fig. 3. Single-cell layout with clustering for D2D communications.

the cell and to bring content closer to users, we take advantage of increasing storage capacity of modern mobile handsets. Mobile users cache popular files and upon receiving request from their neighbors, if they have already cached the request file, they send it through D2D communication. If a UT does not find the desired file in the neighboring UTs, the BS handles the request. All D2D video sharing is controlled by the BS. In other words, the BS knows what each user caches and then, when it receives a request from a user, the BS directs the user to the closest user that has the desired file. The BS should also avoid interference between D2D links. To do this, we assume that the BS divides the cell into several virtual clusters. Then, it will allow only users in each cluster to collaborate with each other. Moreover, only one D2D link can be allowed in each cluster. For simplicity, we assume that there is no interference between adjacent clusters. Figure 3 illustrates the system layout and clusters structure. Each cluster is a square with edge $r$. We refer to $r$ as *collaboration distance*.

The main question that we study in [7] is to maximize the average spectral efficiency through D2D gossiping, i.e., how the collaboration distance should be chosen. The spectral efficiency is equal to number of *active* clusters (D2D links). A cluster is considered active if two users share a file via a D2D link. Increasing the collaboration distance decreases the spatial reuse, i.e., the number of potential D2D links. However, the collaboration distance cannot be very small since in this case users are surrounded by few neighbors which decreases the chance of finding the request file locally within the cluster.

The number of active clusters depends on what files each user stores. Ideally, the most popular files are made available locally. In other words, assume that each user wants to store only one file and there are $k$ users in the cluster, each of them should cache one of the $k$ most popular files without repetition. To implement this, the BS should track the user locations and the number of users within each cluster. Since most users are quite static while they are watching video, the overhead of storing files in users' cache is negligible. However, in our upcoming work [15], we consider distributed D2D in which users randomly and independently cache files.In either case, numerical results show video throughput improvements of one to two orders of magnitude.

In [7], we find the optimum collaboration distance for the finite number of users. The results show that the more redundancy in video requests, the smaller optimum collaboration distance is. Moreover, by increasing redundancy, the spectral efficiency improves. As the cell becomes more dense, the optimum $r$ decreases and the number of active D2D links increases. We also study the scaling behavior of the average number of active D2D links as the number users goes to infinity [6]. The results show that we can achieve linear scaling when there is enough redundancy in video requests.

## IV. Summary and Conclusions

In this paper we introduced a new method for increasing the throughput of wireless video delivery networks. The key idea is the use of a distributed cache, i.e., helper stations that store the most popular video files, and transmit them, upon request, via short-range wireless links to the user terminals. The caches are low-cost because storage capacity has become exceptionally cheap (according to recent prices, two TeraBytes cost approximately 100 dollars), while the loading of the files to the caches can occur through low-rate (and thus cheap and robust) backhaul links at low-demand times. We also showed that using UTs as caches that communicate through BSA-D2D communications is possible and highly efficient.

## References

[1] http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html.
[2] A.F. Molisch, *Wireless communications*. IEEE Press - Wiley, 2011.
[3] V. Chandrasekhar, J. G. Andrews, and A. Gatherer,"Femtocell networks: a survey," *IEEE Commun. Mag.*, 46(9):59 – 67, Sept. 2008.
[4] N. Golrezaei, K. Shanmugam, A.G. Dimakis, A.F. Molisch and G. Caire, "FemtoCaching: wireless video content delivery through distributed caching helpers", accepted in INFOCOM 2012.
[5] N. Golrezaei, K. Shanmugam, A.G. Dimakis, A.F. Molisch and G. Caire, "Wireless video content delivery through coded distributed caching", submitted for publication.
[6] N. Golrezaei, A.G. Dimakis and A.F. Molisch, "Asymptotic throughput of base station assisted device-to-device communications", to be submitted for publication.
[7] N. Golrezaei, A.G. Dimakis and A.F. Molisch, "Base station assisted device-to-device communications for high-throughput wireless video networks", to be submitted for publication.
[8] M. Zink, K. Suh, Y. Gu, and J. Kurose. Watch global, "cache local: YouTube network traffic at a campus networkmeasurements and implications," *Proc. 15th SPIEACM Multimedia Computing and Networking*, 2008.
[9] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of youtube network traffic at a campus network-measurements, models, and implications," *Computer Networks*, 53(4) : 501 – 514, 2009.
[10] S. Borst, "User-level performance of channel-aware scheduling algorithms in wireless data networks," *IEEE Proc. 22nd Joint Conf. IEEE Computer and Communications Societies*, v. 1, p. 321 – 331, 2003.
[11] M. Cardei and D.Z. Du, "Improving wireless sensor network lifetime through power aware organization," *Wireless Networks*, 11(3) : 333 – 340, 2005.
[12] A. Schrijver. *Combinatorial optimization: polyhedra and efficiency*, Springer Verlag, 2003.
[13] *http://traces.cs.umass.edu/index.php/Network/Network*.
[14] S,P. Boyd and L. Vandenberghe *Convex optimization*. Cambridge University Press, 2004.
[15] N. Golrezaei, A.G. Dimakis and A.F. Molisch, "Throughput of device-to-device communications with random caching", to be submitted for publication.