FEMTOCACHING AND D2D COMMUNICATIONS: A NEW PARADIGM FOR VIDEO-AWARE WIRELESS NETWORKS

Contributors

Giuseppe Caire
Department of Electrical Engineering,
University of Southern California

Andreas F. Molisch
Department of Electrical Engineering,
University of Southern California

Video is a main driver for the increased traffic in wireless data networks. This article gives a survey of a novel transmission paradigm that we have developed in the course of the Video-Aware Wireless Networks (VAWN) project. It is based on the following two key properties: (1) video shows a high degree of asynchronous content reuse, and (2) storage is the fastest-increasing quantity in modern hardware. Based on these properties, we suggest caching in helper stations (femtocaching) and/or devices, combined with spectrally efficient short-range communications to deliver video files. For femtocaching, we develop both optimum storage schemes and dynamic streaming policies that optimize video quality. For caching on devices, combined with device-to-device communications, we show that communications within clusters of mobile stations should be used. The cluster size can be adjusted to optimize the tradeoff between frequency reuse and the probability that a device finds a desired file cached by another device in the same cluster. We establish scaling laws that show that under some circumstances the throughput can increase linearly with the number of users, and also analyze the tradeoff between throughput and outage. We finally show that our scheme can be combined with coded multicasting. Simulations demonstrate that network throughput (possibly with outage constraints) can be increased by two orders of magnitude compared to conventional schemes.

Introduction and Motivations

Video transmission is the main driver of the explosive growth in the usage of wireless data transmission. Originally, wireless video mostly implied short video clips (YouTube* or news channels) on the very small screens of smartphones. The recent popularity of tablets and large-screen phones have enabled watching feature-length movies at high resolution on mobile devices, thus greatly increasing the amount of data that have to be transmitted. Numerous market predictions (for example, Cisco[1]) anticipate an increase in the amount of data transmitted each day by almost two orders of magnitude over the next five years. It will then be the dominant source of wireless traffic, by far. These developments, while opening new business models and potentially improving consumer satisfaction, threaten to clog up the already overburdened cellular networks.

Traditional ways of enhancing throughput in cellular systems suffer from significant problems: (1) *increasing the amount of available spectrum* is a drawn-out and costly process, and its effectiveness is limited because the amount of spectrum in the microwave range that can be rededicated to cellular/Wi-Fi* services is small; (2) *increasing the physical-layer capacity of wireless links* becomes difficult as 4G systems, employing MIMO-OFDM with capacity-achieving codes and interference coordination, are already close to the theoretical limits of what is practically feasible[2][3]; and (3) *decreasing the cell size* is viable but expensive. Deploying small base stations, thus creating pico- and femtocells that enable localized communication and high-density spatial reuse of communication resources, brings the (video) content closer to the users (see for example, Chandrasekhar et al.[4], Madan et al.[5], and references therein). However, one critical bottleneck is the cost of providing *backhaul connectivity* of the small base stations to the cellular operator network.[4] For this reason, new network structures have to be investigated that could provide higher per-user data rates at low cost.

In response to the Call for Proposals for "Video-Aware Wireless Networks"[6], we first proposed in 2010 principles of a new network structure, and from 2012–2014, sponsored by the Intel/Cisco/Verizon VAWN program, we elaborated, quantified, and refined these ideas considerably.[7–21] Our approach exploits a unique feature of wireless video, namely the high degree of (asynchronous) content reuse. Based on the fact that storage is cheap and ubiquitous in today's wireless devices, we suggest *replacing backhaul by caching*. We first consider the use of *femtocaching*, where dedicated "helper nodes" can cache popular files and serve requests from wireless users by enabling localized wireless communication. Such helper nodes are similar to femto-BSs, but they have two key differences: they have *added* a large amount of storage, while they *do not have or need* a high-speed backhaul.

"…we suggest replacing backhaul by caching."

We can achieve an even higher density of caching by using devices themselves as video caches—in other words, using devices as mobile helper stations.[8] Due to the tremendous increase in memory on wireless devices (32–64 gigabytes for tablets, and several hundred gigabytes for laptops), there is ample storage for caching available on wireless devices themselves. The simplest way of using this storage would have each user cache the most popular files (possibly with individual modifications based on the tastes of a particular user). However, this approach incurs inefficiencies due to the fact that many users are interested in similar files, and thus the same videos will be duplicated on a large number of devices. On the other hand, the cache on each device is too small to cache a reasonably large number of files. Thus, it is preferable that the devices "pool" their caching resources, so that different devices cache different files and then exchange them, when the occasion arises, through device-to-device (D2D) communications. It is furthermore advantageous that this exchange process is controlled by the cellular infrastructure, which keeps track of (i) which device has which files in its cache, and (ii) which devices have sufficient channel quality to achieve short-range, spectrally efficient, D2D communications. If a requesting device does not find the file in its neighborhood (or in its own cache), it obtains the file in the traditional manner from the controlling base station.

"We can achieve an even higher density of caching by using devices themselves as video caches…"

Notice that caching is a well-known solution in current content distribution networks (CDNs) over the (wired) Internet.[22] Although caching is a standard approach in CDNs, such off-the-shelf solutions do not translate immediately into efficiency gains in the wireless segment since CDNs are implemented in the Internet cloud while, as mentioned above, the bottleneck here is the wireless segment and the limited backhaul connecting dense small cells. In contrast, our proposed approach consists of caching directly at the wireless edge (through dedicated helper nodes) and/or in the user devices. Therefore, our approach is radically new and represents a major step forward with respect to conventional CDNs and yields higher spectrum spatial reuse at much lower deployment (CapEX) and operating (OpEX) cost.

"…our proposed approach consists of caching directly at the wireless edge…"

This article is a summary of our previously published work (for principles and overviews see[7][8], femtocaching descriptions[9][10][11][12][13][14][15], and device-to-device discussions[16][17][18][19][20][21]). It is organized as follows: "Content Reuse" describes the content reuse of video transmission, "Femtocaching" is dedicated to the principle of femtocaching in helper stations, while "Device-to-Device Communications for Wireless Video" describes the main results of caching on devices combined with D2D communications. The "Conclusions" section gives a summary and outlook for future work.

Content Reuse

Wireless video distinguishes itself from other wireless content through its strong content reuse. That is, the same content is seen by a large number of people. This fact was originally used in cellular systems for live streaming systems for exploiting the broadcast nature of the wireless channel.[23][24][25][26][27] Just like in live TV, the number of users tuned in to a particular channel hardly changes the resources required for transmission. However, attempts at practical implementation of such systems (such as, for example, the MediaFLO* system[28]) have failed because users do not want to be bound by predetermined starting times that are inextricably required for such distribution systems.

"Wireless video distinguishes itself from other wireless content through its strong content reuse."

Rather, the bulk of wireless video traffic is due to asynchronous *video on demand*, where users request video files from some cloud-based server (such as iTunes*, NetFlix*, Hulu*, or Amazon Prime*) at arbitrary times. As indicated in the previous section and expounded upon in the following two sections, the use of caching enables the exploitation of *content overlap*, even in the presence of *asynchronism of requests*. As a matter of fact, time-accumulated viewing statistics show that a few popular videos (YouTube clips, sports highlights, and movies) account for a considerable percentage of video traffic on the Internet. Numerous experimental studies have indicated that Zipf distributions have been established as good models to the measured popularity of video files.[29][30] Under this model, the frequency of the *i*th popular file, denoted by f_i, is inversely proportional to its rank:

$$f_i = \frac{\frac{1}{i^{\gamma_r}}}{\sum_{j=1}^{m} \frac{1}{i^{\gamma_r}}}, \ 1 \leq i \leq m. \tag{1}$$

The Zipf exponent γ_r characterizes the distribution by controlling the relative popularity of files. Larger γ_r exponents correspond to higher content reuse; that is, the first few popular files account for the majority of requests. Here, m is the size of the library, which is not the size of all possible files on the Internet, but rather the library of files that are of interest to the set of considered users. Thus, the library size can be a function of the number of considered users n. Let m increase like n^{α}, where $\alpha \geq 0$. The case of constant library size ($\alpha = 0$) occurs, for example, when a provider makes available only a small, regularly rotating set of files to the users. Then $\alpha = 1$, that is, linear increase of m with n, occurs when users have disjoint interests. $\alpha < 1$ corresponds to the case that users share some interests, and thus their file requests overlap. (Though, note that for fixed probability of overlap the total number of requested files m actually increases like $log(n)$.) The case of $\alpha > 1$ can be attributed to the effects of social networking, where the presence of more users spurs an increasing diversity of file requests.

"A further important property of the library is that it changes only on a fairly slow timescale…"

A further important property of the library is that it changes only on a fairly slow timescale (several days or weeks). It can furthermore be shaped by content providers, for example, through pricing policies, or through offering of a large but limited library of popular movies and TV shows (as currently done by NetFlix, Amazon Prime, and iTunes). It is thus possible for helper stations and devices to obtain popular content, for example, through wireless transmission during nighttime, so that they are available when mobile devices demand the content. Due to the steep price drop in storage space, 2 TB of data storage capacity, enough to store 1000 movies, costs only about USD 100 and could thus be easily added to a helper station. Even on mobile devices, 64 GB of storage could be easily dedicated to caching.

In order to avoid an excessively rosy scenario, we put forward a few disclaimers. The work reported here applies principally to a setting where a content library of relatively large files (such as movies and TV shows) is refreshed relatively slowly (for example, on a dally basis), and where the number of users consuming such a library is significantly larger than the number of items in the library. This may apply to a possible future implementation of a "wireless NetFlix," but it is hardly applicable to a "wireless YouTube" paradigm, where the library items are very short (a few minutes), and the library size is much larger than the typical number of users in a given geographic coverage area. In the latter case, the asynchronous content reuse which caching capitalizes on is hardly existent, and caching the whole YouTube library at the wireless edge would be clearly infeasible. In short, this article reflects a set of results and approaches that are relevant in the case where the caching phase (placement of content in the caches) occurs with a clear time-scale separation with respect to the delivery phase (the process of delivering video packets for streaming to the users), and where the size of the content library is moderate with respect to the user population. Further comments are provided under the "Main Conclusions" heading in the next section.

Femtocaching

In this section we illustrate our progress on a femtocaching architecture formed by a set of helper nodes (akin to small cell base stations, with large storage capacity and no or very weak backhaul) serving a set of user nodes, which request on-demand video streaming sessions. The two main problems to be addressed for such an architecture are (1) the generalization of dynamic adaptive streaming currently used in wireless video streaming applications-layer protocols such as Microsoft Smooth Streaming (Silverlight)*[31], Apple HTTP Live Streaming*[32] and 3GPP Dynamic Adaptive Streaming over HTTP (DASH)*[33], to the case of a network of multiple users and multiple helpers, and (2) the cache placement problem, that is, how to optimally place the video files into the helper caches. The following two subsections, "Dynamic Adaptive Streaming from Multiple Helpers" and "Optimal Centralized Cache Placement," illustrate some progress on these topics, respectively. "Testbed Experiments" briefly outlines an experimental Wi-Fi–based testbed setup that we have been developing for the sake of demonstration and validation.

"…we illustrate our progress on a femtocaching architecture formed by a set of helper nodes…"

Dynamic Adaptive Streaming from Multiple Helpers

Consider a discrete, time-slotted wireless network with user set $\mathcal{U}$ (requesting streaming) and helper set $\mathcal{H}$ (serving such requests). In general, every helper $h \in \mathcal{H}$ has a subset $\mathcal{F}(h)$ of files in the library $\mathcal{F}$ within its cache. Some "infrastructure" nodes (for example, cellular base stations) may be connected directly to a CDN in the core network and have access to the whole library $\mathcal{F}$. Also, because of radio-access technology (RAT) restrictions, not all users and helpers may communicate. Therefore, the set of possible communication links is (h, u).

Each user $u \in \mathcal{U}$ requests a video file $f_u \in \mathcal{F}$, formed by a sequence of chunks, which are independently decodable standalone units[34]. Chunks have a fixed duration T_{chunk} and must be reproduced sequentially at the user end. The streaming process consists of transferring N chunks from the helpers to the requesting users (for simplicity of exposition we assume that all files have the same duration $T_{\text{file}} = NT_{\text{chunk}}$) such that the playback buffer of each user contains the required chunk at the beginning of its playback time. Each file f is encoded at a finite number of different quality levels $m \in \{1, \ldots, N_f\}$. In VBR video coding[35], the quality-rate profile may vary from chunk to chunk in the same file, and across different files. We let $D_f(m, t)$ and $B_f(m, t)$ denote the video quality measure (for example, see Wang et al.[36]) and the number of bits for chunk t of file f at quality level m, respectively. We let $r_{hu}(t)$ denote the source coding rate (bit per chunk) of chunk t requested by user u to helper h. Hence, the streaming scheduler must also allocate the source coding rates $r_{hu}(t)$ satisfying

$$\sum_{h \in \mathcal{H}(u)} r_{hu}(t) = B_{f_u}(m_u(t), t), \quad \forall_u \in \mathcal{U} \tag{2}$$

Notice that this formulation applies also to the case of intrasession network coding or other forms of coding against packet erasures, where the requested data may not be the video-encoded bits but linear combinations thereof, suitably replicated throughout the network for the sake of robustness (see for example Pawar et al.[37]).

We represent the underlying wireless network physical layer through a certain long-term average rate region $\mathcal{R}(t)$. For example, we have considered[14] the region achievable by single-antenna helpers and users operating on the same frequency channel, treating interference as noise, and using intracell orthogonal access. However, our framework generalizes to virtually any arbitrary physical layer. For example, in the more recent work[15] we have considered multiuser MIMO helpers (for example, implemented by 802.11ac wave-2 access points, capable of multiuser spatial multiplexing).

We consider a *Network Utility Maximization* (NUM) formulation[14] in order to systematically design an adaptive dynamic streaming scheduler in the above described femtocaching network. Each helper h has a transmission queue pointing at its served users $\mathcal{U}(h)$, which evolves as

"We consider a Network Utility Maximization (NUM) formulationin order to systematically design an adaptive dynamic streaming scheduler…"

$$Q_{hu}(t+1) = \max\{Q_{hu}(t) - WT_{\text{chunk}} R_{hu}(t), 0\} + r_{hu}(t) \tag{3}$$

In words, at each time t, helper h serves $WT_{\text{chunk}}R_{hu}(t)$ information bits to user u by transmitting over WT_{chunk} physical layer dimensions a rate $R_{hu}(t)$ bits/channel use, where W is the system bandwidth. The input to queue $Q_{hu}(t)$ is formed by the newly requested $r_{hu}(t)$ video-encoded bits. Then, for large files formed by many chunks (in the limit for $N \rightarrow \infty$), the NUM problem is given by:

maximize

$$\phi(\bar{D}_u : u \in \mathcal{U}) \tag{4}$$

subject to

$$\lim_{t\rightarrow\infty} \tfrac{1}{t} \sum_{\tau=1}^{t-1} \mathbb{E}[Q_{hu}(\tau)] < \infty \; \forall (h, u) \in \mathcal{E} \tag{5}$$

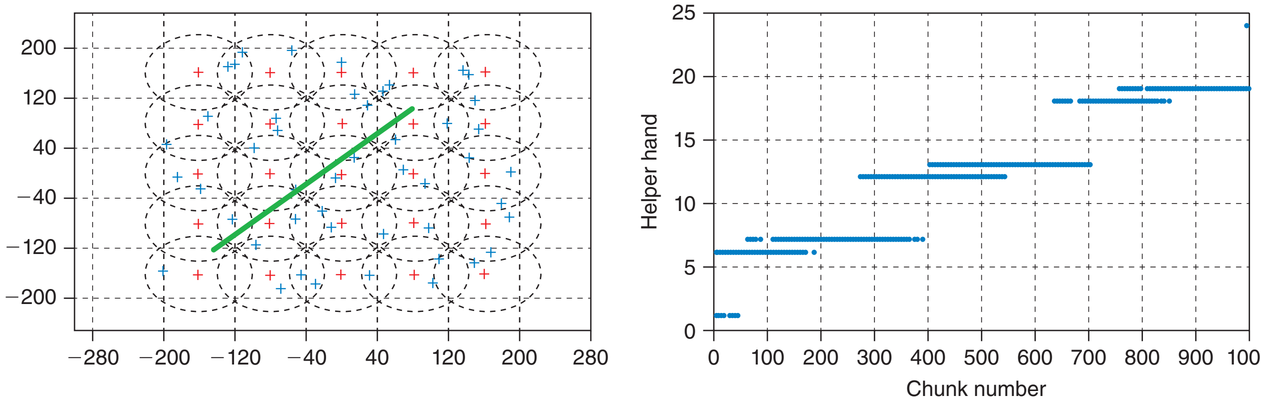
$$\alpha(\tau) \in A_{\omega(t)} \forall\, t, \tag{6}$$

where constraint (5) corresponds to the queues' *strong stability*, $\alpha(t)$ is the decision policy, including the video coding rate requests $\{r_{hu}(t)\}$, the feasible channel coding rate allocation $\{R_{hu}(t)\} \in \mathcal{R}(t)$ and the video quality selection decisions $\{m_{\mathrm{u}}(t)\}$, and $A_{\omega(t)}$ is the set of feasible policies for network state $\omega(t) = \{g_{hu}(t), D_{f_u}(\cdot, t), B_{f_u}(\cdot, t) : \forall (h, u) \in \mathcal{E}\}$. We solved problem (4)–(6) through the design of a dynamic policy based on the Lyapunov drift plus penalty (DPP) approach. The resulting policy is decentralized and consists of a distributed multiuser version of a DASH-like protocol, where users make adaptive decisions on which helper to request from and at which quality level each chunk should be requested.

In the more recent work[15] we have considered the case where each user maintains a single *virtual* request queue $Q_u(t)$ and dynamically requests only the chunk at the head of the line. This approach, referred to as "pull" strategy, prevents a user from receiving chunks out of playback order. We have also discussed[14][15] effective prebuffering and rebuffering schemes based on monitoring the maximum chunk delivery delay in a sliding window and setting the buffering time (in multiples of T_{chunk}) sufficiently larger than the maximum observed chunk delay.

"…we have considered the case where each user maintains a single virtual request queue…"

Figure 1 shows a numerical experiment based on a small-cell femtocaching layout, where we let one user move along a linear trajectory at speed 1 m/s (Figure 1(a)). Our dynamic adaptive streaming policy is able to implicitly "discover" new helpers as the user moves across the network, such that chunks are requested from favorable helpers along the streaming session (Figure 1(b)).



(a) Network layout

(b) Seamless helper hand-off

Figure 1: Simulation experiments of the dynamic adaptive streaming policy. (Source: Bethanabhotla et al., 2014.[15] (Copyright IEEE))

Optimal Centralized Cache Placement

In this section we consider the problem of optimal content placement in a femtocaching network where: (1) the network topology is known, (2) the long-term average link rates are known, and (3) the user demand distribution (file popularity) is known. Yet the realization of the users' demands is a random vector, and the caching placement must be done without *a priori* knowledge of the user requests, but only of their statistics.

We have considered the following problem.[12] With the same notation developed before, a femtocaching network is represented by the bipartite graph $\mathcal{G} = (\mathcal{H}, \mathcal{U}, \mathcal{E})$. Here, we are interested in minimizing the average file downloading time with respect to the case where all users download independently from the cellular base station (denoted as helper node 0 in the following). We simplify the physical layer and consider that each link has a fixed average downloading latency indicated by $\omega_{h,u}$, expressed in *time per information bit*. This scenario is consistent with that developed before, considering that the physical layer rate region $R(t)$ is time-invariant in this case, and that the adaptive dynamic scheduling chooses a fixed set of physical layer rates $\mathbf{R} \in \mathcal{R}$, such that $\omega_{h,u} = 1/R_{h,u}$.

The content placement problem treated in this work can be formulated as follows: *for a given file popularity distribution, helper storage capacity and network* *topology, how should the files be placed in the helper caches such that the average sum downloading delay of all users is minimized?*

"We distinguish between uncoded and coded content placement."

We distinguish between uncoded and coded content placement. In the uncoded case, video-encoded files are cached directly, with possible replication. In the coded case, we consider intrasession coding already mentioned before (for example, using the scheme presented by Pawar et al.[37]).

Uncoded cache placement: An uncoded cache placement is represented by a bipartite graph $\tilde{\mathcal{G}} = (\mathcal{F}, \mathcal{H}, \tilde{\mathcal{E}})$, such that an edge $(f, h) \in \tilde{\mathcal{E}}$ indicates that a copy of file f is contained in the cache of helper h. We let **X** denote the $|\mathcal{F}| \times |\mathcal{H}|$ adjacency matrix of $\tilde{\mathcal{G}}$, such that $x_{f,h} = 1$ if $(f, h) \in \tilde{\mathcal{E}}$ and 0 otherwise. By the cache size constraint, we have that the column weight of X is at most equal to the cache size M (expressed in file units).

The average delay per information bit for user u can be written as:

$$\bar{D}_u = \sum_{j=1}^{|\mathcal{H}(u)|-1} \omega_{(j)_u u} \sum_{f=1}^{|\mathcal{F}|} \left[\prod_{i=1}^{j=1} (1 - x_{f,(i)_u}) \right] x_{f,(j)_u} P_r(f)$$

$$+\omega_{0,u} \sum_{f=1}^{|\mathcal{F}|} \left[\prod_{i=1}^{|\mathcal{H}(u)|-1} (1 - x_{f,(i)_u}) \right] P_r(f). \qquad (7)$$

where $P_r(t)$ is the request probability distribution, and where the notation $(j)_u$ indicates the helper index in $H(u)$ with the jth smallest delay to user u. The minimization of the sum (over the users) average per-bit downloading delay can be expressed as the integer programming problem:

$$\text{maximize} \sum_{u \in \mathcal{U}} (\omega_{0,u} - \bar{D}_u)$$

$$\text{subject to} \sum_{f \in \mathcal{F}} x_{f,h} \leq M, \ \forall \ h$$

$$\mathrm{X} \in \{0,1\}^{|\mathcal{F}| \times |\mathcal{H}|}. \qquad (8)$$

We showed[12] that (8) is NP-complete. However, it can be formulated as the maximization of a monotone submodular function over matroid constraints, for which a simple greedy strategy achieves at least one half of the optimum value.

"Using intrasession coding, we can obtain a relaxed version of the problem."

Coded content placement: Using intrasession coding, we can obtain a relaxed version of the above problem. In particular, let $\rho = [\rho_{f,h}]$, where $\rho_{f,h}$ denotes the fraction of parity bits of file f contained in the cache of helper h. The delay to download a fraction of parity bits $\rho_{f,h}$ on the link (h, u) is given by $\rho_{f,h}\, \omega_{h,u}$. A file is entirely retrieved when a fraction larger than or equal to 1 of parity bits is downloaded, since by a property of Maximum Distance Separable codes, we have that with a number of parity bits equal to the number of information bits, then the latter can be exactly recovered. The average delay per information bit necessary for user u to download file f, assuming that it can download it from its best j helpers, is given by

$$\begin{aligned}\bar{D}_u^{f,j} &= \sum_{i=1}^{j-1} \rho_{f,(i)_u}\, \omega_{(i)_u,u} + \left(1-\sum_{i=1}^{j-1} \rho_{f,(i)_u}\right) \omega_{(j)_u,u} \\ &= \omega_{(j)_u,u} - \sum_{i=1}^{j-1} \rho_{f,(i)_u} (\omega_{(j)_u,u} - \omega_{(i)_u,u}). \qquad (9)\end{aligned}$$

Notice that file f can be downloaded by user u from its best j helpers only if $\sum_{i=1}^{j} \rho_{f,(i)_u} \geq 1$. In addition, since the cellular base station contains all files, we always have $\rho_{f,0} = 1$ for all $f \in F$, such that all users can always obtain the requested files by downloading the missing parity bits from the base station.

The delay $\bar{D}_u^f$ incurred by user u because of downloading file f is a piecewise-defined affine function of the elements of the placement matrix ρ, given by

$$\bar{D}_u^f = \begin{cases} \bar{D}_u^{f,1} & \text{if} \quad \rho_{f,(1)_u} \geq 1 \\ \vdots & \vdots \\ \bar{D}_u^{f,j} & \text{if} \quad \sum_{i=1}^{j-1} \rho_{f,(i)_u} < 1, \sum_{i=1}^{j} \rho_{f,(i)_u} \geq 1 \\ \vdots & \vdots \\ \bar{D}_u^{f,|\mathcal{H}(u)|} & \text{if} \quad \sum_{i=1}^{|\mathcal{H}(u)|-1} \rho_{f,(i)_u} < 1, \end{cases} \qquad (10)$$

We can show that $\bar{D}_u^f$ is a convex function of ρ. The average delay of user u is given by $\bar{D}_u = \sum_{f=1}^{|\mathcal{F}|} P_r(f) \bar{D}_u^f$. With some further manipulations, the coded placement optimization problem takes on the form:

$$\begin{aligned} &\text{minimize} \sum_{u=1}^{U} \sum_{f=1}^{|\mathcal{F}|} P_r(f) \max_{j \in \{1,2,\ldots,|\mathcal{H}(u)|\}} \{\bar{D}_u^{f,j}\} \\ &\text{subject to} \sum_{f=1}^{|\mathcal{F}|} \rho_{f,h} \leq M, \ \forall\, h \\ &\qquad \rho \in [0,1]^{|\mathcal{F}| \times |\mathcal{H}|}, \qquad (11) \end{aligned}$$

where the optimization is with respect to ρ. In general, the optimum value of delay obtained with the coded optimization is better than the uncoded optimization because any placement matrix with integer entries is a feasible solution to the coded problem. In this sense, the coded optimization is a convex relaxation of the uncoded problem.

"…beyond its theoretical interest, optimal cache placement is unlikely to be useful in practice…"

We conclude this section by mentioning that, beyond its theoretical interest, optimal cache placement is unlikely to be useful in practice since while the user demand distribution $P_r(f)$ may be well estimated and predicted, the network topology is typically time-varying with dynamics comparable with the streaming sessions. Therefore, reconfiguring the caches at this time scale is definitely not practical. However, further computer experiments have also shown that the cache distribution obtained when the mobile stations are in "typical" distances from the helpers also provides good performance for various realizations of random placement of nodes. Furthermore, distributed random caching turns out to be "good enough" as we shall see under the heading "D2D Throughput versus Outage" in the following section. Hence, comparing optimal placement with random caching yields useful insight into the potential performance gap lost by a decentralized approach. Interestingly, in any reasonable network configuration, it turns out that such a gap is very small.

Testbed Experiments

We have implemented a small Wi-Fi–based testbed to demonstrate the femtocaching idea. In particular, we have implemented a scheme reminiscent of the dynamic adaptive streaming scheme illustrated earlier under the heading “Dynamic Adaptive Streaming from Multiple Helpers,” where both helper and user nodes are implemented on an Android* mobile platform (see Figure 2). The helpers create their own hot spot using the tethering mode of Wi-Fi, such that they effectively act as base stations with cached content and no wired (DSL/Ethernet) backhaul.

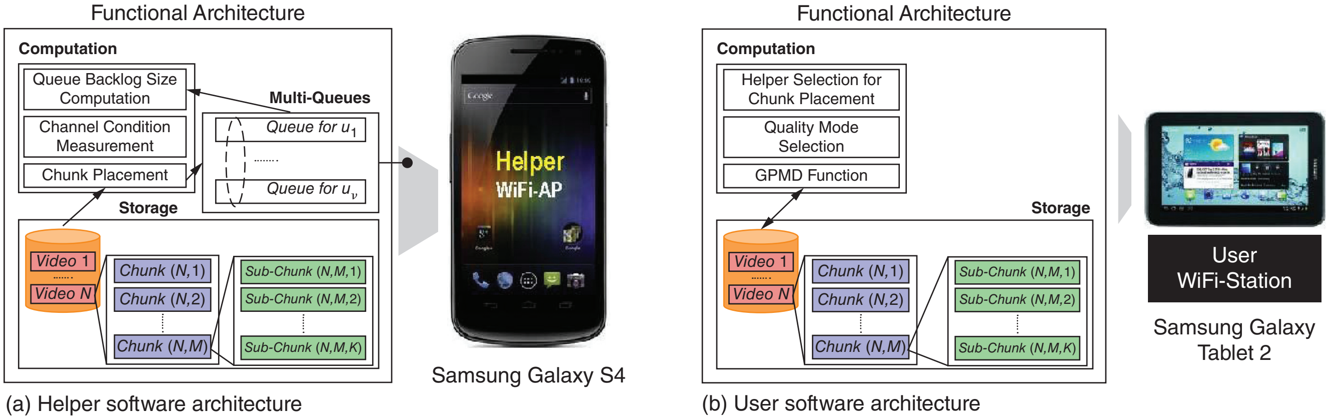


Figure 2: Device software architectures.
(Source: Kim et al., 2013.[13] (Copyright ACM))

A practical limitation of Wi-Fi is that users cannot be associated simultaneously with multiple helpers and therefore possibly receive multiple data packets from different helpers. Therefore, we have modified the scheme of the earlier section, implementing a heuristic strategy for user-helper *exclusive* association. In particular, we have implemented the “pull” strategy with a single request queue per user. Then, each user selects the helper node that has the next required chunk according to its own request queue and such that the selected helper has the shortest transmit queue among all helpers having said chunk. The chunk is requested at a quality level that depends on the helper-user Wi-Fi link rate. In this way, we take the helper transmit queue as a proxy for its current load, and this strategy implements a sort of implicit load balancing. It should be noted that this heuristic puts higher priority on receiving the chunk on time rather than obtaining high video-coding quality. In fact, with this strategy, a user always goes for the helper with the smallest transmit queue, even if the obtained link rate is smaller compared to some other helper. The implemented scheme is referred to as a *greedy pull for minimum delay (GPMD) strategy*.

The users can follow two approaches: (1) selecting a possibly different helper at each chunk request or (2) selecting a helper that should transmit a sequence of chunks (thus limiting the number of handoffs between helpers, which in practice contributes to a significant protocol overhead due to the inefficiency of Wi-Fi). In the first case, at each time slot, each user determines which helper should place/stream the next chunk. This is obtained by letting each user send a packet requesting the next video chunk to all neighbor helpers. Each helper replies back with the current queue backlog size. Then, the user selects the helper that has the smallest queue backlog size and sends a chunk download request to said helper. If there are multiple helpers achieving the same smallest queue backlog sizes, the user performs a random selection. The second approach is similar, but helper selection decisions are made less often. For example, in the case of low mobility users, while a user moves across the network, it detects when the current serving helper yields an unacceptably low per-link rate. In this case, it initiates a new request in order to determine a new helper from which to download.

Main Conclusions

This subsection summarizes a number of lessons learned, system design guidelines, and points for further investigations relative to our work on femtocaching summarized before.

"This subsection summarizes a number of lessons learned, system design guidelines, and points for further investigations relative to our work on femtocaching summarized before."

The first observation concerns the validity of the time-scale decomposition that we have implicitly assumed when separating the cache placement phase and the video delivery phase. This assumption is valid for the case of a video library formed by popular content such as movies and TV shows, which is refreshed on a daily or even weekly basis. In this case, we think of a sort of "wireless NetFlix" that pushes into the helpers' caches new content at off-peak times (for example, at night, exploiting the LTE cellular network, without requiring any wired backhaul). In our cache placement problem formulation, we have considered a single popularity distribution. As a matter of fact, the popularity distribution varies significantly with respect to the social group of users, the location, and the time of day. For example, we may imagine that in a train station (for example, Penn Station in New York) the morning commuters are interested in the news and stock market data, while people in a city park in the afternoon are interested in the latest episode of *Modern Family*. Mathematically, by conditioning with respect to a restricted location and time of day, the popularity distribution can have noticeable peaks as compared to averaging over all locations and times. Hence, demands can be more easily predicted and caches better utilized if such "high definition" information is available. Furthermore, the prediction of users' content demands can be further enhanced by taking into account the users' social network interconnections (for example, recommendation systems). All these considerations call for a systematic and coherent study of the problem of content demand prediction in space (with the resolution of the typical area of a small cell corresponding to a helper node) and in time (with the resolution of a few hours). Such prediction can be formulated as a large-dimensional Bayesian inference problem, for which machine learning techniques can be applied. This represents an interesting area for further research.

The second observation is that in our video delivery (streaming/scheduling) from multiple helpers we have assumed that users can request video chunks from multiple helper stations without paying a handover cost. As a matter of fact, if the underlying PHY and MAC wireless network are implemented with off-the-shelf Wi-Fi or Wi-Fi–direct, clients must associate to helper nodes and dynamic association on a per-chunk basis is infeasible because of the slow handover. This calls for a more efficient implementation of the PHY and MAC of the underlying small cell network, allowing for highly dynamic helper-user association.

Finally, a number of improvements to the dynamic streaming and link scheduling algorithms can be considered, as recently done by Bethanabhotla et al.[38] and Joseph and Veciana.[39] These schemes improve upon the basic scheme provided in this survey article, since they attempt a direct control of the playback buffer of the users, ensuring smooth streaming without interruptions, and make use of a single request "virtual queue" that avoids the annoying problem of chunks delivered out-of-order. This may occur in the case where each helper has its own individual transmit queue pointing to a requesting user such that chunks requested from different helpers may be subject to different queuing delays.

Device-to-Device Communications for Wireless Video

We now turn to networks where the devices themselves are caching video files, and transmitting them, upon request, to other devices via high-spectral-efficiency D2D links. For this type of network, we only consider the transmission of video *files*, not streaming, and also neglect the issue of video rate adaptation (these are topics for our future research). In this section, we first outline the principle and mathematical model we consider. We then provide the fundamental scaling laws, both for the sum throughput in the cell (disregarding any fairness considerations), and for the tradeoff between throughput and outage. Under the heading "D2D with Coded Caching," we then describe how D2D communications can be combined with coded multicast. Simulation results described in "Performance in Realistic Settings" illustrate important behaviors.

> *"We now turn to networks where the devices themselves are caching video files, and transmitting them, upon request, to other devices via high-spectral-efficiency D2D links."*

Principle and Mathematical Model

We consider a network where each device can cache a fixed number M video files, and send them—upon request—to other devices nearby. If a device cannot obtain a file through D2D communications, it can obtain it from a macrocellular base station (BS) through conventional cellular transmission. The BS also keeps track of which devices can communicate with each other and which files are cached on each device. Such BS-controlled D2D communication is more efficient (and more acceptable to spectrum owners if the communications occur in a licensed band) than traditional uncoordinated peer-to-peer communications.

Specifically, consider a network that consists of macrocells. For simplicity, each macrocell is assumed to have the shape of a square, with the side of the square having unit length, and with n users per cell. We assume that inter-cell interference from the device BS links is kept to a small level through appropriate cell/frequency planning.[40] There are n users in each cell. We consider either the case that the users are placed on a fixed grid such that the distance of the users is $1/\sqrt{n}$, or we consider a random placement of the nodes in the network, leading to a random geometric graph. Each user makes a request for a file in an i.i.d. (independent identically distributed) manner, according to a given request probability mass function $P_r(f)$.

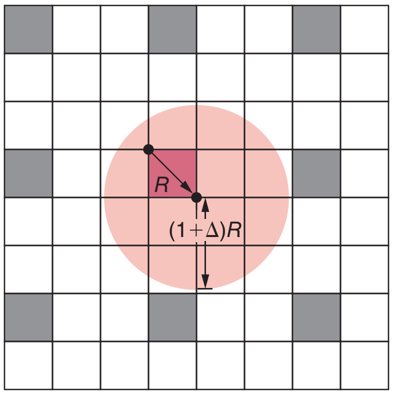


Figure 3: An example of the single-cell layout (Source: Ji et al., 2014.[21] (Copyright IEEE))

We furthermore subdivide the cell into equal-sized, disjoint groups of users that we call "clusters" of size (radius) r, with g_c nodes in it. To further simplify the mathematical model, we assume that only nodes that are part of the same cluster can communicate with each other. If a user can find the requested file inside the cluster, we say there is one *potential link* in this cluster; when at least one link is scheduled, we say that the cluster is "active." We use an *interference avoidance* scheme, such that at most one link can be active in each cluster on one time-frequency resource. Intercluster interference is avoided through a frequency reuse strategy as shown in Figure 3. In a simplified protocol model[41] nodes that are within the "reuse distance" cannot communicate at all due to interference (red disk), while nodes/clusters outside the reuse distance are not interfered with at all.

This model is, of course, a major simplification whose assumptions do not hold exactly in practice. Yet, it provides a first approximation to the exact solutions. Furthermore, many of the simplifications do not impact the scaling laws (that is, the functional form of the increase of throughput with number of users), though they do impact the absolute value of the throughput.

We furthermore have to determine which files should be cached by the devices. We consider here two strategies:

- *deterministic* caching, where the BS instructs the devices to cache the most popular files in a disjoint manner; that is, no file should be cached twice in devices belonging to the same cluster. This approach can only be realized if the device stays in the same locations for many hours (the time between refreshing of the cache, which is a rare event, and the time the files are requested by other devices). Performance obtained with the deterministic caching strategy also serves as a useful upper bound for more realistic schemes.
- *random* caching, where each device randomly and independently caches a set of files according to a common probability mass function. In our first papers, we assumed that the caching distribution is also a Zipf distribution, though with a parameter γ_c that is different from γ_r, and which has to be optimized for a particular γ_r and r. Since the Zipf distribution

is characterized by a single parameter, this description gives important intuitive insights into how concentrated the caching distribution should be.

We subsequently found[19] that the optimal caching distribution that maximizes the probability that any user finds its requested file inside its own cluster is given (for a node arrangement on a rectangular grid as described above) by

$$P_c^*(f)=\left[1-\frac{v}{z_f}\right]^+, f = 1, \dots, m, \tag{12}$$

where $v=\frac{m^*-1}{\sum_{f=1}^{m^*}\frac{1}{z_f}} z_f = P_r(f)^{\frac{1}{M(g_c-1)-1}}, m^* = \Theta\left(\min\left\{\frac{M}{\gamma_r}g_c, m\right\}\right)$ and $[\Lambda]^+ = \max\,[\Lambda, 0]$.

Besides the caching strategy, the main performance factor that can be influenced by the system designer is the cluster size. This is regulated through the transmit power (we assume that it is the same for all users in a cell, but can be optimized as a function of user density, library size, and cache size). Varying the cluster size trades off probability for finding the desired file in the cluster with the frequency reuse. Optimizing cluster size is an important task for the system design.

There are a number of different criteria for optimizing the system parameters. One obvious candidate is the total network throughput. It is maximized by maximizing the number of active clusters. We showed[18] that for deterministic caching, the expected throughput can be computed as

$$E\{T\} = \frac{1}{r^2}\sum_{k=0}^{n}\left(1-\prod_{i=1}^{k}(1-\left(P_{CVC}(k)-P_r(f_i)\right))\right)\Pr[K=k]. \tag{13}$$

where $P_{\mathrm{CVC}}(k)$ is the probability that the requested file is in the Common Virtual Cache (the union of all caches in the cluster), that is, among the k most popular files. $Pr[K=k]$, the probability that there are k users in a cluster, is deterministic for the rectangular grid arrangement, and

$$\Pr[K=k] = \binom{n}{k}(r^2)^k(1-r^2)^{n-k}, \tag{14}$$

for random node placement.

Scaling Laws for Throughput

We now turn to the scaling laws, which describe the functional behavior of the overall throughput T as a function of the user density. For this analysis, we concentrate on the case that each device make requests according to a Zipf distribution with γ_r, and randomly caches according to a Zipf distribution with parameter γ_c. We note, however, that deterministic and random caching show no fundamental difference in their scaling laws. We use the following notation: given two functions f and g, we say that: $f(n) = O(g(n))$ if there exists a constant c and integer N such that $f(n) \le cg(n)$ for $n > N$; $f(n) = \Omega(g(n))$ if $g(n) = O(f(n))$; $f(n) = \Theta(g(n))$ if $f(n) = O(g(n))$ and $g(n) = O(f(n))$.

"We now turn to the scaling laws, which describe the functional behavior of the overall throughput T as a function of the user density."

We established[16][17] the following lower and upper bounds:

Theorem 1: If the Zipf exponent $\gamma_r > 1$,

i) Upper bound: For any caching policy, $E[T] = O(n)$,

ii) Achievability: Given that $c_1\sqrt{\frac{1}{n}} \leq r_{opt}(n) \leq c_2\sqrt{\frac{1}{n}}$ (c_1 and c_2 are positive constants that do not depend on n) and using a Zipf caching distribution with exponent $\gamma_c > 1$ then $E\,[T] = \Theta(n)$.

This theorem shows that if we choose $r_{opt}(n) = \Theta\left(\sqrt{\frac{1}{n}}\right)$ and $\gamma_c > 1$, $E[T]$ can grow linearly with n. For the request distributions that are less concentrated γ_r < 1, we obtained the following result:

Theorem 2: If $\gamma_r < 1$,

i) Upper bound: For any caching policy, $E[T] = O\left(\frac{n}{m^{\eta}}\right)$ where $\eta = \frac{1-\gamma_r}{2-\gamma_r}$,

ii) Achievability: If $c_3\sqrt{\frac{m^{\eta+\epsilon}}{n}} \leq r_{opt}(n) \leq c_4\sqrt{\frac{m^{\eta+\epsilon}}{n}}$ and users cache files randomly and independently according to a Zipf distribution with exponent γ_c, for any exponent $\eta + \varepsilon$, there exists γ_c such that $E[T] = \Theta\left(\frac{n}{m^{\eta+\epsilon}}\right)$ where $0 < \varepsilon < \frac{1}{6}$ and γ_c is a solution to the following equation

$$\frac{(1-\gamma_r)\gamma_c}{1-\gamma_r + \gamma_c} = \eta + \varepsilon.$$

The main conclusion from the scaling law is that for highly concentrated demand distribution, $\gamma_r > 1$, the throughput scales linearly with the number of users. Equivalently, the per-user throughput remains constant as the user density increases; the number of users in a cluster also stays constant. For heavy-tailed demand distributions, the throughput of the system increases only sublinearly, as the clusters have to become larger (in terms of number of nodes in the cluster), to be able to find requested files within the caches of the cluster members.

D2D Throughput versus Outage

"…focusing on throughput only is not enough in a D2D network…"

We noticed[20] that focusing on throughput only is not enough in a D2D network, especially under the protocol "collision" model. In fact, from the network viewpoint, the throughput is maximized by allowing only nearest neighbor communication for the users whose request can be satisfied by a neighbor, and dropping all other users. This, however, yields a very large probability that a user request is not satisfied by the network. A more complete picture is provided by the throughput-outage tradeoff, defined by Ji et al.[20], where we focused on *max-min fairness.* That is, our aim is to maximize the minimum average throughput $\overline{T}_{\min}$ per user, subject to a constraint on the average outage probability p. In other words, the fraction of users that are not served by the system (either because their request is not found in the caches, or because the scheduling policy denies service to these users). The *Throughput-Outage Tradeoff* of a D2D caching network is generally defined as the region of all achievable throughput-outage pairs (T, p). In particular, letting $T^*(p) = \sup\{T: (T, p) \in \mathcal{T}\}$ be the maximum achievable min-throughput per user for given outage probability not larger than p, we have that $T^*(p)$ is the result of the optimization problem:

$$\begin{aligned} &\text{maximize } \overline{T}_{\min} \\ &\text{subject to } p_o \leq p, \end{aligned} \qquad (15)$$

where p_o indicates the average (over the users) outage probability, and the maximization is with respect to the decentralized cache placement and transmission policies.

Letting $|\mathcal{U}| = n$ (number of users) and $|\mathcal{F}| = m$ (number of files), our main result is that in the regime of n → ∞ and library size m at most linear in n, for any strictly positive p, the optimal throughput of a D2D caching network with one-hop direct communication between sources and destinations scales as $T^*(p) = \Theta\left(\max\{\frac{M}{m}, \frac{1}{n}\}\right)$. The outage probability affects only the multiplicative constant of the leading term, which can be tightly characterized via inner and outer bounds.[20] Interestingly, this scaling law is identical to what can be achieved by *network-coded multicasting* from a single base station[42], and what can be achieved by network-coded D2D delivery (see "D2D with Coded Caching"). Here, this provably optimal (in an information theoretic sense) scaling law is achieved by using simple direct delivery (no intrasession network coding) and simple decentralized cache placement. The related D2D link scheduling is also extremely simple, namely the scheme described earlier under the heading "Principle and Mathematical Model" and also assumed in "Scaling Laws for Throughput". Again, the "magic" of this scheme consists of choosing appropriately the cluster size (see also Golrezaei et al.[11][18]). If the cluster is too small, the spectrum reuse is large but the probability of not finding the desired content in the cluster is also large, resulting in a unacceptably high outage probability. If the cluster size is too large, the content can be found with high probability, but the spectrum spatial reuse is too low. Balancing the tension between spectrum reuse and outage probability, we arrive at the order-optimal result. Interestingly, the scaling $1/n$ corresponds to orthogonal access from a single broadcasting base station with the full library and can be regarded as representative of current conventional systems. In contrast, when $nM \gg m$, the throughput increases linearly with M. In this regime, *offloading the video on demand traffic to the D2D local links by exploiting the storage capacity at each node is a very attractive approach, since storage space is much "cheaper" than scarce resources such as bandwidth or dense base station deployment.*

"…offloading the video on demand traffic to the D2D local links by exploiting the storage capacity at each node is a very attractive approach…"

D2D with Coded Caching

Recently, a *network coded multicasting* scheme exploiting caching at the user nodes was proposed by Maddah-Ali and Niesen.[42][43] In this scheme, the files in the library are divided into blocks (packets) and users cache carefully designed subsets. (In this context, the packetization used for coding may not coincide with the chunk units used by the streaming process.) Then, for a given set of user demands, the base station sends to all users (multicasting) a common codeword formed by a sequence of packets obtained as linear combinations of the original file packets. For the case of arbitrary (adversarial) demands, the scheme of Maddah-Ali and Niesen[42] is shown to be information theoretic near-optimal in the sense that the number of network coded packet transmissions necessary to satisfy any user demand is minimal within a small bounded gap. Both articles by Maddah-Ali and Niesen[42][43] consider one-hop transmission from the base station only, and it is assumed that all users can receive (at the same rate) one network coded packet per unit time. (In fact, content delivery from a single transmitter (base station) to multiple users with caching is a special case of index coding[44][45][46] where the demands are arbitrary but the "side information" formed by the cached packets is explicitly designed or generated at random with a specific statistical distribution.)

The number of required multicast network coded packets that must be transmitted to satisfy any user demand achieved by the scheme of Maddah-Ali and Niesen[42] is given by

$$N(n,\, m,\, M) = n\left(1 - \frac{M}{m}\right)\frac{1}{1 + \frac{Mn}{m}}.$$

This yields the min per-user throughput $T_{\mathrm{CM}} = \frac{c_0}{N(n,\, m,\, N)}$ where C_0 is the common multicasting rate that any user in the system must be able to decode. (For simplicity of exposition, we neglect here the effect of a nonzero outage probability. See Ji et al.[21] for more details.) One can see immediately that for large m, n, and finite M, the scaling of the symmetric throughput per user is given by $\Theta\left(\max\{\frac{M}{m}, \frac{1}{n}\}\right)$. As mentioned above, somewhat surprisingly, this is the same scaling behavior of our D2D scheme outlined earlier in "D2D Throughput versus Outage."

At this point, a natural question to ask is whether the gain of D2D local transmission and the gain of network-coded multicasting will be compounded. We considered[20] a network-coded multicasting scheme that involves only local D2D communication (no base station). The caching and delivery scheme is best explained by the example shown in Figure 4. This scheme can be generalized to any n, m, M, and it can be shown that without any spatial reuse (that is, any transmission is heard by all users in the cell, and one transmission per time-frequency slot is allowed), this subpacketization caching and distributed network-coded delivery scheme delivers one packet to all requesting users in $\frac{m}{M}\left(1 - \frac{M}{m}\right)$ time slots. This is almost the same as the centralized scheme using the base station by Maddah-Ali and Niesen[42] and achieves the same fundamental scaling law. We also showed that no further scaling law order gain can be obtained if spatial reuse is also exploited. Intuitively, since network coding makes a single (coded) packet useful for as many requesting users as possible, it is better if such a packet is received by all users in the cell, while D2D transmission gets its efficiency from restricting each transmission to a small cluster of nodes. Nevertheless, reducing the transmission range and applying our scheme in clusters, with reuse, yields simpler caching subpacketization and allows transmissions at a higher rate (bit/s/Hz). Hence, the benefits of network-coded multicasting applied to a D2D caching network are reflected in the actual throughput and coding complexity, rather than in the throughput scaling order.

"…the benefits of network-coded multicasting applied to a D2D caching network are reflected in the actual throughput and coding complexity…"

Performance in Realistic Settings

We now present some examples based on simulations in the above-described settings. We first consider the case of a single square cell with $n = 500$ users and 1000 cells, using centralized caching and a protocol model. Figure 5

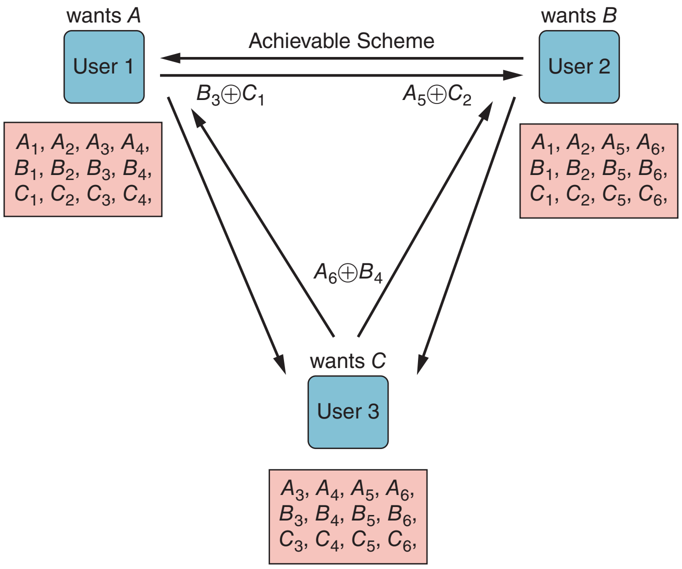

wants A
User 1
Achievable Scheme
wants B
User 2
$B_3 \oplus C_1$
$A_5 \oplus C_2$
$A_1, A_2, A_3, A_4,$ $B_1, B_2, B_3, B_4,$ $C_1, C_2, C_3, C_4,$
$A_1, A_2, A_5, A_6,$ $B_1, B_2, B_5, B_6,$ $C_1, C_2, C_5, C_6,$
$A_6 \oplus B_4$
wants C
User 3
$A_3, A_4, A_5, A_6,$ $B_3, B_4, B_5, B_6,$ $C_3, C_4, C_5, C_6,$

Figure 4: An example of subpacketized caching and network-coded delivery for D2D caching networks. We consider a network with $n = 3$ users, $m = 3$ files (A, B, C) and storage capacity is $M = 2$ files. We divide each file into 6 subpackets (e.g. A is divided into $A_1, \cdots, A_6$.) We let user 1 request A; user 2 requests B and user 3 requests C. The cached subpackets are shown in the rectangles under each user. For the delivery phase, user 1 transmits $B_3 \oplus C_1$; user 2 transmits $A_5 \oplus C_2$ and user 3 transmits $A_6 \oplus B_4$. The normalized number of transmissions is $3 \cdot \frac{1}{6} = \frac{1}{2}$. (Source: Ji et al., 2013.[20] (Copyright IEEE))

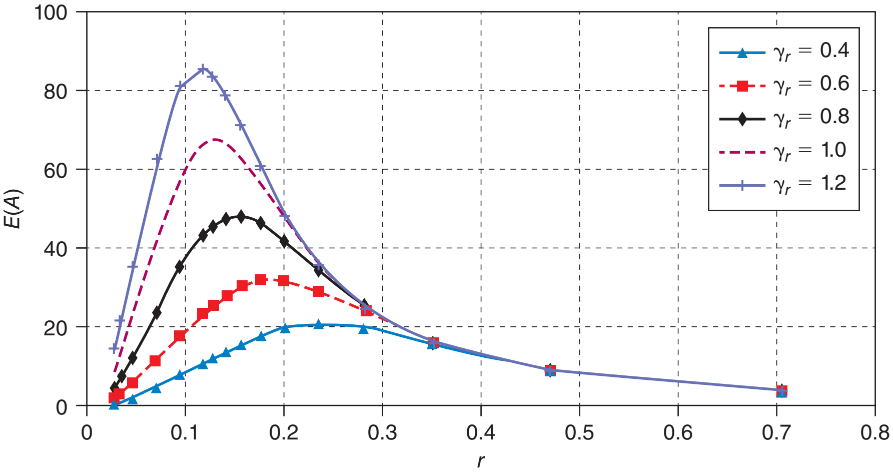


Figure 5: The average number of active clusters versus the collaboration distance for deterministic caching with $n = 500$ and $m = 1000$. (Source: Golrezaei et al., 2012.[18] (Copyright IEEE))

shows the average number of active clusters versus the collaboration distance *r* (neglecting intercluster interference). We see that the larger γ_r, that is, the more concentrated the request distribution, the smaller the cluster size should be. This is logical, since the probability of finding a desired file even in a small cluster increases with γ_r. Simulations with a more realistic setting (including shadowing and intercluster interference) provided very similar optimal cluster sizes.

We next consider the case of random caching, using a Zipf caching distribution. Figure 6 shows the average number of active clusters versus the collaboration distance *r* for different values of the exponent of the caching distribution, γ_c. Further simulations showed that increasing γ_r increases the optimum γ_c, since the first few popular files account for the majority of requests and thus to satisfy the users' requests. There is littler need to cache less popular files.

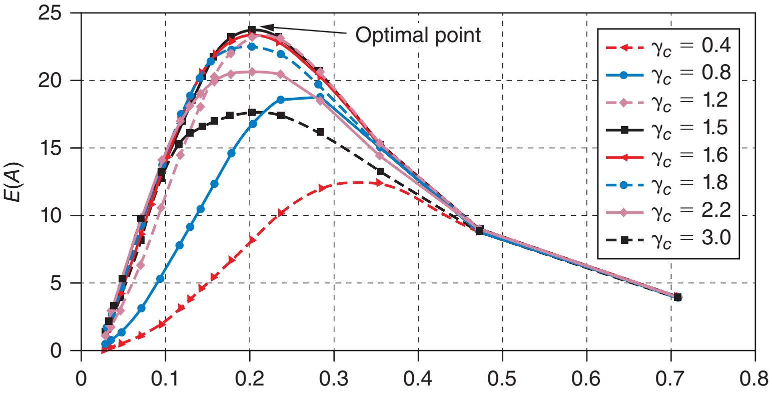


Figure 6: The average number of active clusters versus r for random caching for different γ_c, with $\gamma_r = 0.6$, $n = 500$ and $m = 1000$. (Source: Golrezaei et al., 2012.[18] (Copyright IEEE))

We also performed extensive simulations in more realistic scenarios. We first demonstrate that the throughput of the D2D scheme is markedly (an order of magnitude at low outages) higher than the standard broadcasting from the base station, harmonic broadcasting[47][48][49], and network-coded multicasting.[42] Figure 7 shows an example of such throughput-outage tradeoff performance in a realistic propagation and network topology scenario. We considered indoor office and hotspots environments. Winner models[50] for these wireless propagation channels in these environments were enhanced by models for body shadowing (for more details see Ji et al.[21]), and the capacity for the D2D links was computed based on Shannon's capacity equation. Even in such realistic conditions, the D2D solution provides competitive performance and is significantly simpler to implement than coded multicasting. It is remarkable that while the scaling laws for the coded multicast and D2D schemes are the same, in practical situations the higher capacity of the short-distance links plays a significant role, and a good throughput-outage tradeoff can be achieved even without the use of a BS connection. The good performance of the D2D scheme is tied to the inherent diversity.

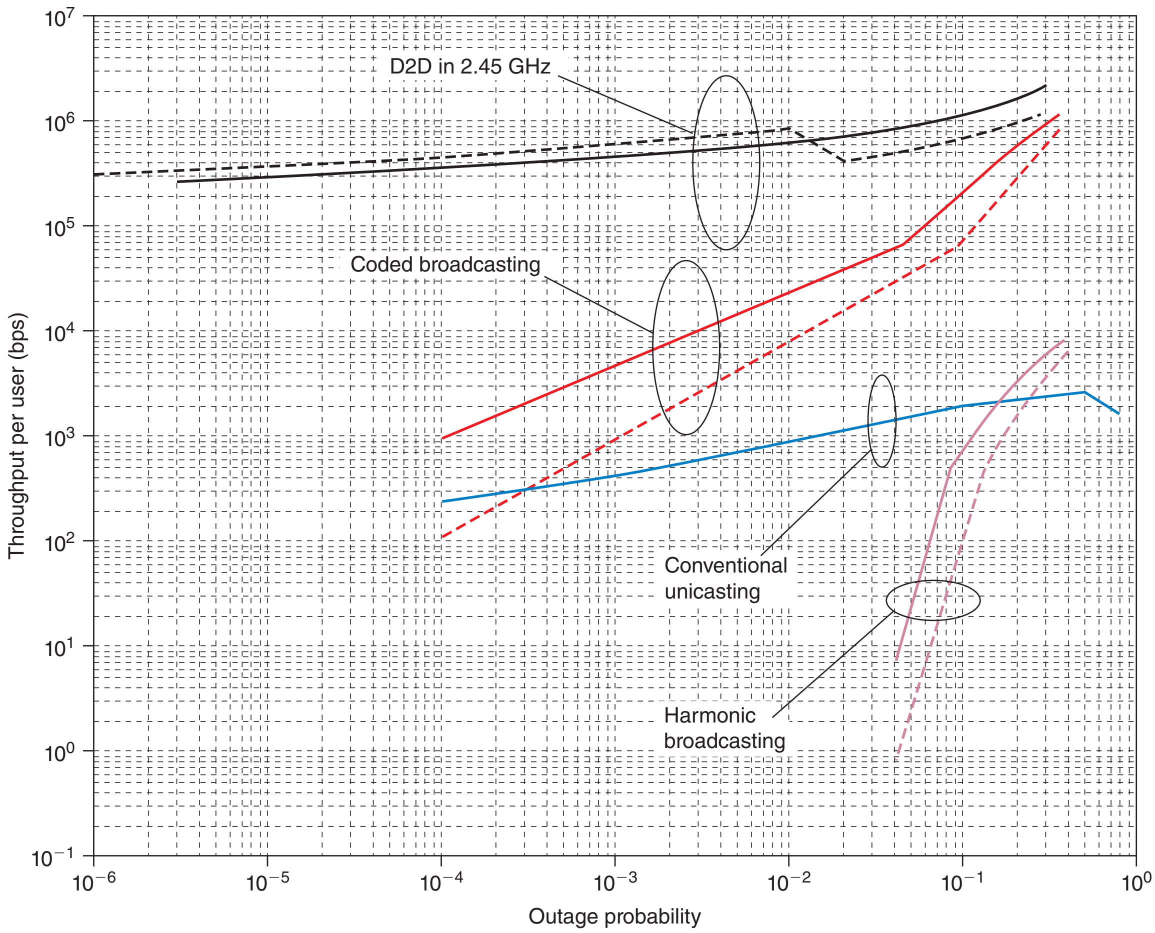


Figure 7: Simulation results for the throughput-outage tradeoff for conventional unicasting, coded multicasting, harmonic broadcasting, and the 2:45 GHz D2D communication scheme under both indoor office and indoor hotspot channel models. Solid lines: indoor office; dashed lines: indoor hotspot.
(Source: Ji et al., 2012.[21] (Copyright IEEE))

The theoretical derivations discussed in "D2D Throughput versus Outage" state that the throughput-outage tradeoff does not depend on the number of users or user density as long as n and m are large and $Mn \gg m$. However, the throughput-outage scaling behavior was obtained for the protocol model, where the link capacity for a (feasible) link does not depend on SNR, and thus distance. In practice, the dependence of capacity on distance makes the user density also an important parameter for the system performance. Figure 8 shows that there is a tradeoff between the user density and the throughput, which is caused by two effects: (1) a higher user density allows a smaller cluster size, in turn resulting in shorter links and higher SINR; (2) a small cluster size increases the probability for having LOS interference, which can degrade the performance of the system significantly.

Main Conclusions

From our detailed mathematical investigations, we can draw a number of important conclusions for actual implementation. Firstly, D2D

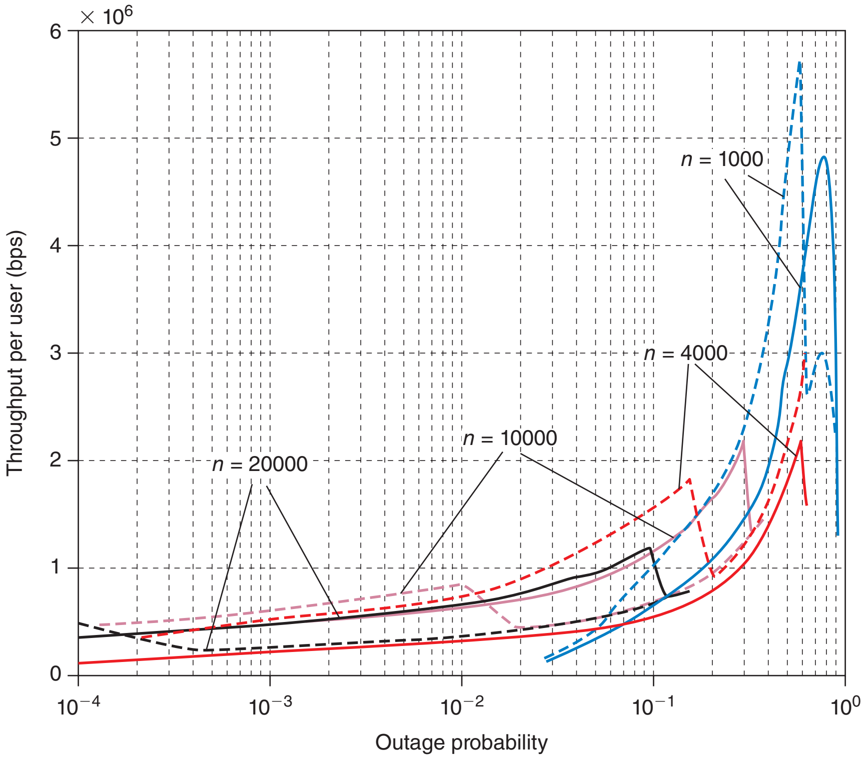


Figure 8: The throughput-outage tradeoff for different user densities. Solid lines: indoor office; dashed lines: indoor hotspot.
(Source: Ji et al., 2012.[21] (Copyright IEEE))

communications with caching on the local devices offers a dramatic increase in the capacity. The actual increase is critically determined by the reuse statistics of the videos. When the library considered by the users has a compact size, then the video throughput per user is independent of the total number of users in a cell. The approach is less useful when the demand distribution is heavy-tailed.

While the mathematical framework allows a detailed analysis, we conjecture that most benefits can be reaped if the "most requested videos" can be stored on 30 devices. From our analysis of upper bounds and achievable schemes, we can conclude that a very simple approach—random caching according to a given probability density function—is scaling-law optimum, and furthermore our numerical simulations showed that the actually achievable numerical values are close to those that can be achieved with an idealized, centralized caching scheme. This tells us not only that a low-complexity implementation is feasible, but also that on-the-fly modifications of caches (for instance, when demand distribution changes) can be done in a simple and straightforward manner. By itself, the D2D scheme might suffer from unacceptably high outage. However, through a suitable combination with transmission from the BS, a very high reliability can be achieved—as a matter of fact, much higher than for a BS alone. Users at a cell edge, who are normally the "problem case," benefit from the possibility of obtaining files from another nearby device. Our simulations have shown dramatic increase of both reliability and throughput not only with respect to traditional multicast, but also compared with the recently introduced coded multicast, once realistic propagation conditions are taken into account.

A final point that deserves attention is how users can be incentivized to act as caches. In other words, what is the answer to a user asking "why should I use *my* battery to help somebody else get a video more quickly?" In one possible approach, the video file exchange would happen on the basis of reciprocity: only users who provide files (on a time-averaged basis) are also allowed to obtain files. Similar principles have been successfully applied in peer-to-peer file exchanges. Secondly, operators can incentivize users: for example, they might state that files obtained through D2D communications do not count towards the data quota of a particular user (this would be in the interest of operators, since D2D transmissions relieve the pressure on the cellular infrastructure).

Conclusions

Supported by the VAWN research initiatives, we have developed a comprehensive framework for caching in wireless networks targeted to on-demand video streaming, which is a killer application for 4G cellular networks, and the root cause of the predicted hundredfold increase in wireless traffic demand in the next five years. We considered two related network architectures: caching in wireless helper nodes (femtocaching), such that users stream videos from helpers in their neighborhood, and caching directly in the user devices, such that users stream videos from other users via D2D connections. In both cases we have shown large potential gains and solved key problems in the design and analysis of such networks.

"…we have developed a comprehensive framework for caching in wireless networks targeted to on-demand video streaming…"

Besides the problems discussed in this article, a number of interesting problems constitute topics for future research:

- More advanced PHY schemes in femtocaching networks. For example, helper nodes may have multiple antennas and use multiuser MIMO and advanced interference management schemes, beyond the simple Wi-Fi–inspired schemes considered so far.
- More advanced PHY schemes for D2D networks, beyond the simple spectrum reuse and interference avoidance clustering scheme used so far. The optimum scheduling of users is connected to the maximum-independent-set problem, an important (and hard) problem in computer science.
- (Limited) multi-hop in D2D caching networks. Instead of restricting the communication to single-hop, rarer files can be reached through D2D communications without excessive increase in cluster size.
- Transmission schemes that take into account the limitations of existing standards for D2D such as Wi-Fi Direct.
- Analyzing the impact of nonuniform and nonergodic user distributions. Neighbor discovery and channel estimation can be optimized, which is a prerequisite for optimal scheduling of users.

In addition, there is also a large number of interesting problems revolving around the determination and prediction of user request probabilities, the question of incentivizing users to let devices be used for caching purposes, and data authentication and prevention of piracy. While all of these problems seem eminently solvable, as discussions with industry representatives have indicated, further research is needed. We can thus safely say that while our research has shown the feasibility and enormous promise of femtocaching and D2D communications, much remains to be done for a more complete understanding.

References

[1] Cisco, “The Zettabyte Era-Trends and Analysis,” 2013.

[2] Dahlman, E., S. Parkvall, and J. Skold, *4G: LTE/LTE-Advanced for Mobile Broadband: LTE/LTE-Advanced for Mobile Broadband*. Academic Press, 2011.

[3] Dohler, M., R. Heath, A. Lozano, C. Papadias, and R. Valenzuela, “Is the phy layer dead?” *Communications Magazine, IEEE*, vol. 49, no. 4, pp. 159–165, 2011.

[4] Chandrasekhar, V., J. Andrews, and A. Gatherer, “Femtocell networks: a survey,” *Communications Magazine, IEEE*, vol. 46, no. 9, pp. 59–67, 2008.

[5] Madan, R., J. Borran, A. Sampath, N. Bhushan, A. Khandekar, and T. Ji, “Cell association and interference coordination in heterogeneous lte-a cellular networks,” *Selected Areas in Communications, IEEE Journal on*, vol. 28, no. 9, pp. 1479–1489, 2010.

[6] “Video-Aware Wireless Networks” http://software.intel.com/en-us/articles/video-aware-wirelessnetworks.

[7] Golrezaei, N., A. G. Dimakis, A. F. Molisch, and G. Caire, “Wireless video content delivery through distributed caching and peer-to-peer gossiping,” in *Signals, Systems and Computers (ASILOMAR), 2011 Conference Record of the Forty Fifth Asilomar Conference on*. IEEE, 2011, pp. 1177–1180.

[8] Golrezaei, N., M. Ji, A. F. Molisch, A. G. Dimakis, and G. Caire, “Device-to-device communications for wireless video delivery,” in *Forty Sixth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*. IEEE, 2012, pp. 930–933.

[9] Golrezaei, N., A. F. Molisch, A. G. Dimakis, and G. Caire, “Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution,” *Communications Magazine, IEEE*, vol. 51, no. 4, pp. 142–149, 2013.

[10] Golrezaei, N., K. Shanmugam, A. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless video content delivery through distributed caching helpers," in *INFOCOM, 2012 Proceedings IEEE*, 2012, pp. 1107–1115.

[11] Golrezaei, N., A. Dimakis, and A. F. Molisch, "Wireless device-to-device communications with distributed caching," in *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 2781–2785.

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

[13] Kim, J., F. Meng, P. Chen, H. E. Egilmez, D. Bethanabhotla, A. F. Molisch, M. J. Neely, G. Caire, and A. Ortega, "Adaptive video streaming for device-to-device mobile platforms," in *Proceedings of the 19th annual international conference on Mobile computing & networking*. ACM, 2013, pp. 127–130.

[14] Bethanabhotla, D., G. Caire, and M. Neely, "Joint transmission scheduling and congestion control for adaptive video streaming in small-cell networks," *arXiv preprint arXiv:1304.8083*, 2013.

[15] Bethanabhotla, D., G. Caire, and M. J. Neely, "Adaptive video streaming in MU-MIMO networks," submitted to ISIT 2014, Jan. 2014.

[16] Golrezaei, N., A. G. Dimakis, and A. F. Molisch, "Wireless device-todevice communications with distributed caching," in *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 2781–2785.

[17] Golrezaei, N., A. G. Dimakis, and A. F. Molisch, "Device-to-device collaboration through distributed storage," in *Global Communications Conference (GLOBECOM), 2012 IEEE*. IEEE, 2012, pp. 2397–2402.

[18] Golrezaei, N., A. F. Molisch, and A. G. Dimakis, "Base-station assisted device-to-device communications for high-throughput wireless video networks," in *Communications (ICC), 2012 IEEE International Conference on*. IEEE, 2012, pp. 7077–7081.

[19] Ji, M., G. Caire, and A. F. Molisch, "Fundamental limits of distributed caching in D2D wireless networks," in *Information Theory Workshop (ITW), 2013 IEEE*, 2013, pp. 1–5.

[20] Ji, M., G. Caire, and A. F. Molisch, "The throughput-outage tradeoff of wireless one-hop caching networks," *arXiv preprint arXiv:1312.2637*, 2013.

[21] Ji, M., G. Caire, and A. F. Molisch, "Wireless device-to-device caching networks: Basic principles and system performance," *arXiv preprint arXiv:1305.5216*, 2012.

[22] Nygren, E., R. K. Sitaraman, and J. Sun, "The akamai network: a platform for high-performance internet applications," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 3, pp. 2–19, 2010.

[23] Luo, F.-L., *Mobile Multimedia Broadcasting Standards: Technology and Practice*. Springer Verlag, 2008.

[24] Reimers, U., "Digital video broadcasting," *Communications Magazine, IEEE*, vol. 36, no. 6, pp. 104–110, 1998.

[25] Ladebusch, U. and C. Liss, "Terrestrial dvb (dvb-t): A broadcast technology for stationary portable and mobile use," *Proceedings of the IEEE*, vol. 94, no. 1, pp. 183–193, 2006.

[26] Bursalioglu, O., M. Fresia, G. Caire, and H. Poor, "Lossy multicasting over binary symmetric broadcast channels," *Signal Processing, IEEE Transactions on*, vol. 59, no. 8, pp. 3915–3929, 2011.

[27] Li, Y., E. Soljanin, and P. Spasojevi´c, "Three schemes for wireless coded broadcast to heterogeneous users," *Physical Communication*, 2012.

[28] Gao, Q., M. Chari, A. Chen, F. Ling, and K. Walker, "MediaFLO technology: FLO air interface overview," in *Mobile Multimedia Broadcasting Standards*. Springer, 2009, pp. 189–220.

[29] Cha, M., H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system," in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '07. ACM, 2007, pp. 1–14.

[30] "http://traces.cs.umass.edu/index.php/network/network."

[31] Zambelli, A., "Iis smooth streaming technical overview," *Microsoft Corporation*, vol. 3, 2009.

[32] Begen, A., T. Akgul, and M. Baugher, "Watching video over the web: Part 1: Streaming protocols," *Internet Computing, IEEE*, vol. 15, no. 2, pp. 54–63, 2011.

[33] Stockhammer, T., "Dynamic adaptive streaming over http–: standards and design principles," in *Proceedings of the second annual ACM conference on Multimedia systems*. ACM, 2011, pp. 133–144.

[34] Sánchez de la Fuente, Y., T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. De Vleeschauwer, W. Van Leekwijck, and Y. Le Louédec, "idash: improved dynamic adaptive streaming over http using scalable

video coding," in *Proceedings of the second annual ACM conference on Multimedia systems*. ACM, 2011, pp. 257–264.

[35] Pancha, P. and M. El Zarki, "Mpeg coding for variable bit rate video transmission," *Communications Magazine, IEEE*, vol. 32, no. 5, pp. 54–66, 1994.

[36] Wang, Z., A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *Image Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 600–612, 2004.

[37] Pawar, S., N. Noorshams, S. El Rouayheb, and K. Ramchandran, "Dress codes for the storage cloud: Simple randomized constructions," in *Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on*. IEEE, 2011, pp. 2338–2342.

[38] Bethanabhotla, D., G. Caire, and M. J. Neely, "Adaptive video streaming in mu-mimo networks," *CoRR*, vol. abs/1401.6476, 2014.

[39] Joseph, V. and G. de Veciana, "Nova: Qoe-driven optimization of dash-based video delivery in networks," *CoRR*, vol. abs/1307.7210, 2013.

[40] Molisch, A. F., *Wireless Communications*. Second Edition, IEEE Press – John Wiley & Sons, 2011.

[41] Gupta, P. and P. Kumar, "The capacity of wireless networks," *Information Theory, IEEE Transactions on*, vol. 46, no. 2, pp. 388–404, 2000.

[42] Maddah-Ali, M. and U. Niesen, "Fundamental limits of caching," *arXiv preprint arXiv:1209.5807*, 2012.

[43] Maddah-Ali, M. and U. Niesen, "Decentralized caching attains order-optimal memory-rate trade-off," *arXiv preprint arXiv:1301.5848*, 2013.

[44] El Rouayheb, S., A. Sprintson, and C. Georghiades, "On the index coding problem and its relation to network coding and matroid theory," *Information Theory, IEEE Transactions on*, vol. 56, no. 7, pp. 3187–3195, 2010.

[45] Sun, H. and S. A. Jafar, "Index Coding Capacity: How far can one go with only Shannon Inequalities?" *ArXiv e-prints*, Mar. 2013.

[46] Shanmugam, K., A. G. Dimakis, and M. Langberg, "Local Graph Coloring and Index Coding," *ArXiv e-prints*, Jan. 2013.

[47] Juhn, L. and L. Tseng, "Harmonic broadcasting for video-on-demand service," *Broadcasting, IEEE Transactions on*, vol. 43, no. 3, pp. 268–271, 1997.

[48] Pâris, J.-F., S. Carter, and D. Long, “Efficient broadcasting protocols for video on demand,” in *Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 1998. Proceedings. Sixth International Symposium on. IEEE,* 1998, pp. 127–132.

[49] Engebretsen, L. and M. Sudan, “Harmonic broadcasting is bandwidth-optimal assuming constant bit rate,” *Networks*, vol. 47, no. 3, pp. 172–177, 2006.

[50] WINNER-II, “D1. 1.2, WINNER II channel models,” 2007.

Author Biographies

Giuseppe Caire was born in Torino, Italy, in 1965. He received a B.Sc. in Electrical Engineering from Politecnico di Torino (Italy) in 1990, a M.Sc. in Electrical Engineering from Princeton University in 1992, and a PhD from Politecnico di Torino in 1994. He has been assistant professor at the Politecnico di Torino, associate professor at the University of Parma, professor with the Department of Mobile Communications at the Eurecom Institute, Sophia-Antipolis, France, and he is currently a professor of Electrical Engineering at the University of Southern California, Los Angeles and an Alexander von Humboldt Professor at the Technical University of Berlin, Germany. He received the Jack Neubauer Best System Paper Award from the IEEE Vehicular Technology Society in 2003, and the IEEE Communications Society & Information Theory Society Joint Paper Award in 2004 and 2011. Giuseppe Caire has been a Fellow of IEEE since 2005. He has served in the Board of Governors of IEEE Information Theory and was the Society President of the IEEE Information Theory Society in 2011. His main research interests are in the field of communications and information theory. He can be reached at caire@usc.edu.

Andreas F. Molisch is professor of Electrical Engineering and Director of the Communication Sciences Institute (CSI) at the University of Southern California. He previously was at TU Vienna, AT&T (Bell) Labs, Lund University, and Mitsubishi Electric Research Labs. His research interests are novel cellular architectures, wireless propagation channel measurements and modeling, ultrawideband localization and communication, and multi-antenna systems. He has authored 4 books, 16 book chapters, 170 journal papers, and numerous conference papers as well as 70 patents and 60 standards contributions. He is a Fellow of IEEE, AAAS, and IET, and a member of the Austrian Academy of Sciences, as well as recipient of numerous awards. He can be contacted at molisch@usc.edu.