Wireless Caching Helper System with Heterogeneous Traffic and Random Availability

Ioannis Avgouleas, Nikolaos Pappas, and Vangelis Angelakis

Abstract—Multimedia content streaming from Internet-based sources emerges as one of the most high demanded services by wireless users. In order to alleviate excessive traffic due to multimedia content transmission, many architectures (e.g., small cells, femtocells, etc.) have been proposed to offload such traffic to the nearest (or strongest) access point also called “helper”. The deployment of more helpers is not necessarily beneficial due to their potential of increasing interference. In this work, we evaluate a wireless system in which we distinguish between cachable and non-cachable traffic. More specifically, we consider a general system in which a wireless user with limited cache storage requests cachable content from a data center that can be directly accessed through a base station. The user can be assisted by a pair of wireless helpers that exchange non-cachable content as well. Packets arrive at the queue of the source helper in bursts. Each helper has its own cache to assist the user’s requests for cachable content. Files not available from the helpers are transmitted by the base station. We analyze the system throughput and the delay experienced by the user and show how they are affected by the packet arrival rate at the source helper, the availability of caching helpers, the caches’ parameters, and the user’s request rate by means of numerical results.

I. INTRODUCTION

Wireless video has been one of the main generators of wireless data traffic. It is expected to originate 75% of the global mobile traffic by 2020 [1] and inevitably contribute to networks’ congestion and delays. One of the most promising technologies to cope with such issues is caching popular files in helper nodes that constitute a wireless distributed caching network that assists base stations by handling requests for popular files [2], [3].

Wireless caching helpers can store a number of popular files and transmit them to the requesting users more efficiently, considering that helpers have been deployed in such a way that the wireless channel between helpers and users is better than the one between users and base stations. Wireless networks with caching capabilities can significantly reduce cellular traffic and delay as well as simultaneously increase throughput [4], [5].

In this paper, we study a wireless system in which traffic is distinguished in cachable and non-cachable. A user with limited cache storage requests cachable content from a data center using a base station which has direct access to it through a backhaul link. Two wireless nodes within the proximity of the user exchange non-cachable content and have limited cache storage. Therefore, they can act as caching helpers for the user by serving its requests for cachable content. Files not available at the helpers can be fetched by data center through the base station. Additionally, the source helper is equipped with a queue whose role is to save the excessive traffic with the intention of transmitting it to the destination helper in a subsequent time slot. Packets arrive in bursts. Concerning caching, we assume the content placement is given and hierarchical.

A. Related Work

Various content placement strategies have been studied in scientific literature e.g., caching the most popular content everywhere [2], probabilistic caching [6], [7], cooperative caching [8], [12], or caching based on location e.g., geographical caching [13].

Additionally, several different performance metrics have been considered. In earlier studies of wireless caching, cache hit probability (or ratio) [13], and the density of successful receptions or cache-server requests [6], [13] have been commonly investigated as a means of evaluating the performance of wireless caching systems. Furthermore, there are several studies regarding energy efficiency or consumption of the different caching schemes [13]–[16] as well as taking into account the traffic load of the wireless links [17], [18]. Methods that reduce traffic load by optimizing the offloading probability or gain can be found in [19]–[21].

More recently, a considerable amount of research works analyze wireless caching systems by considering throughput [7], [8] and/or delay [22]. Regarding the latter, the majority of research works cope with mitigating the backhaul or transmission delay under the assumption that traffic or requests are saturated. However, there are works that take into account stochastic arrivals of requests at different nodes [23], [24].

Caching has been applied to several different network realizations e.g., FemtoCaching [3] in which the so-called Femto Base Station (FBS) serve a group of dedicated users with random content requests while simultaneously the non-dedicated users might be served with delay due to cache misses or no FBS availability. The coded/uncoded cached contents are stored in multiple small cells, the so-called femtocells. Given the file requests distribution and the cache size of each femtocell, the content placement is studied such that the downloading time is minimized.

The advent of vehicular networks necessitates the use of caches in order to reduce the latency of content streaming and increase the offered Quality of Service (QoS) [25], [26]. Supporting vehicle-to-everything connections urges the exploration of alternative data routing protocols in order to avoid incurring excessive end-to-end delay and backhaul resources allocation. On the contrary, moving computational and storage resources to the mobile edge computing seems
This can be done e.g., by employing a new paradigm known as local area data network [30], or other advances in radio access networks (RANs) for Internet of Things (IoT) [31].

Many contemporary works consider to jointly optimize the problems of content caching (or placement), computing, and allocating radio resources. They usually consider and solve separately these important issues by formulating the computation offloading or content caching as convex optimization problems with different metrics e.g., service latency, network capacity, backhaul rate etc. [32], [33]. Works that simultaneously address the aforementioned problems together and propose a joint optimization solution for the fog-enabled IoT or cloud RANs (C-RANs) can be found in [34] and [35], respectively.

For some applications e.g., broadcast or multicast applications, single transmissions from the base station to more than one user is useful. The authors in [36] propose a content caching and distribution scheme for smart grid enabled heterogeneous networks, in which each popular file is stored in multiple service nodes with energy harvesting capabilities. The optimization of the total on-grid power consumption, the user association scheme, and the radio resource allocation improves the reliability and performance of the wireless access network. The evolution of 5G mobile networks is going to incorporate cloud computing technologies. The authors in [37] propose the concept of "Caching-as-a-Service" (CaaS) based on C-RANS as a means to cache anything, anytime, and anywhere in the cloud-based 5G mobile networks with the intention of satisfying user demands from any service location with high QoS. Furthermore, they discuss the technical details of virtualization, optimization, applications, and services of CaaS in 5G mobile networks.

A key distinction among research papers in wireless caching is the assumption regarding the availability of caching helpers. Many papers consider that caching helpers can serve users requests whenever the requested file is cached while others adopt the more realistic assumption that caching helpers might be unable to assist user requests when, for example, serve other users of interest [3], [13]. To the best of our knowledge, the proposed wireless caching model has not been studied in literature. The work of [8] analyzes a wireless network with one caching helper that does not consider hierarchical caching.

B. Contribution

In this paper, we study a wireless system in which traffic is distinguished in cachable and non-cachable. A user with limited cache storage requests cachable content from a data center connected to a base station through a backhaul link. Two wireless nodes within the users proximity exchange non-cachable files and have limited cache storage. Therefore, they can act as caching helpers for the user by serving its requests for cachable content. Files not available at the helpers can be transmitted by the base station. The source helper is equipped with an infinite queue whose role is to save packets intended for the destination helper for transmission in a subsequent time slot. Packets arrive in bursts.

We analyze the system throughput by assuming that the wireless helper nodes have random access to the channel and there is no coordination between them. First, we characterize the system throughput concerning the case in which the queue at the source helper is stable as well as unstable. Moreover, we formulate a mathematical optimization problem to optimize the probabilities by which the helpers assist the user to maximize the system throughput. Subsequently, we characterize the average delay experienced by the user from the time of a local cache miss until it receives the requested file. Finally, we provide numerical results to show how the packet arrival rate at the source helper, the availability of caching helpers, random access to the channel, caching parameters, and the user’s request rate affect the system throughput and delay.

C. Organization of the paper

In Section II, we present the system model comprising the network, the caching, the transmission, and the physical layer model. Section III provides the analytical derivation of throughput for the cases of stable and the unstable queue at the source helper. The average delay performance is given in Section IV. In Section V, we numerically evaluate our theoretical analysis of the previous sections. Finally, Section VI concludes our research work.

II. SYSTEM MODEL

A. Network Model

We consider a network system with four wireless nodes: a pair of caching helpers S and D, a random user U within the coverage of the helpers and a base station (BS) node connected to a datacenter (DC) through a backhaul link as depicted in Fig. 1. We consider slotted time and that a packet transmission takes one time slot.

Helper S is equipped with an infinite queue and the packet arrivals in its queue follow a Bernoulli process with average
arrival rate $\lambda$. It transmits packets to the destination helper $D$. In each time slot, user $U$ requests for a file in its own cache. In case $U$'s cache miss, which happens with probability $q_U$, it requests the file from external resources i.e., the caching helpers or the data center (through $BS$). The data center stores the whole library and, hence, every file that $U$ may request.

Requesting a file directly from the $BS$ is not necessarily the best policy since there might be congestion at the link connecting $BS$ and $U$. Consequently, limited throughput or increased delay might be experienced using this link instead of requesting the file from one of the caching helpers. Moreover, the $BS$ is not always available to help $U$; this happens with probability $\alpha$ in each time slot. Therefore, it is preferable to $U$ when it is served by the caching helpers.

The flowchart of user $U$’s operation with respect to its request content search is shown in Fig. 2. The operation of caching helpers $S$ and $D$, represented as flowcharts, are depicted in Figs. 3 and 4 respectively.

B. Cache Placement and Access

We assume the content placement is given and hierarchical i.e., when the user node requests for a file that is not stored in its most popular files, it first probes the closest caching helper which stores the next most popular files. If this probe fails, then the second caching helper is probed for the requested file. If it also cache misses, then the file can be found in the data center. Additionally, the source helper is equipped with a queue whose role is to save the excessive non-cachable traffic with the intention of transmitting it to the destination helper in a subsequent time slot.

Furthermore, the user device $U$ and the caching helpers $D$ and $S$ have cache capacity to $M_U$, $M_D$, and $M_S$ files, respectively, and $M_U \leq M_D \leq M_S$ holds. We also consider the Collaborative Most Popular Content (CMPC) policy. According to CMPC, user $U$ stores the first $M_U$ most popular files in its own cache, helper $D$ stores the next most $M_D$ popular files, and stores the next most $M_S$ popular files. Following CMPC requires exchange of information among devices e.g., the cache size of each device and the content placement in each device. We assume that this information exchange is negligible.

C. Transmission Model

In each time slot, $S$ will attempt transmissions of non-cachable content to $D$ with probability $q_S$ (if its’ queue is not empty), and is available for $U$ with probability $1 - q_S$. We assume that the caching helpers assist $U$ only when specific conditions apply: $D$ will attempt transmission to $U$ with probability $q_D$, and $S$ will help $U$ only when it is not transmitting to $D$. When the source caching helper $S$ is transmitting to helper $D$ and the user $U$ requests a file from external resources, then $U$ can be served by $D$ or by $DC$. In that case, there are two parallel transmissions one from $S$ to $D$, and one from $D$ (or $DC$ respectively) to $U$. If the caching helper $S$ is available for $U$, then there is no parallel transmissions since only one of $S$, $D$, or $DC$ can help $U$ at the same time slot.

Regarding $DC$, we model its availability with a probability $\alpha$ to model the fact that it is not always available to serve $U$ due to serving other users or failure. If the $DC$ is always available to $U$, then $\alpha = 1$. On the other hand, if the $DC$ is not available for $U$, then $\alpha = 0$. We summarize the aforementioned events and notation in Table I. Additionally, the operation of $U$, $S$, and $D$ as flowcharts can be found in Figs. 3-4.

D. Physical Layer Model

We assume Rayleigh fading for the wireless channel and that a packet transmitted by node $i$ is successfully received by node $j$ if and only if $SINR(i,j) \geq \gamma_j$, where $\gamma_j$ is a threshold characteristic of node $j$. Therefore, the received power at node $j$ when $i$ transmits is $P_{tx}(i,j) = A(i,j)h(i,j)$, where $A(i,j)$ is exponentially distributed and the received power factor is:

$$h(i,j) = \frac{P_{tx}(i)}{r(i,j)p},$$

where $P_{tx}(i)$ is the transmit power of node $i$, $r(i,j)$ is the distance in $m$ between node $i$ and node $j$, and $p \in [2,6]$ is the path-loss exponent.

By assuming perfect self-interference cancellation, the success probability of link $i \rightarrow j$, with $T$ denoting the set of transmitting nodes, is given by [38]:

$$P_{i \rightarrow j \mid T} = \exp(-\frac{\gamma_j n_j}{v(i,j)h(i,j)}) \times \prod_{k \in T \setminus \{i,j\}} \left(1 + \frac{\gamma_j v(k,j)h(k,j)}{v(i,j)h(i,j)}\right)^{-1},$$

where $v(i,j)$ is the parameter of the Rayleigh fading r.v., and $n_j$ is the noise power at receiver $j$. To simplify notation, we use $P_{i \rightarrow j}$ to denote the success probability of link $i \rightarrow j$ if node $i$ is the only transmitting node.

III. THROUGHPUT ANALYSIS

In this section, we analyze the throughput of the system depicted in Fig. 1. We are interested in the weighted sum of the throughput that helper $S$ provides along with the throughput

| Notation | Event |
|----------|-------|
| $q_U$ | $U$ requests file from external resources (U cache miss) |
| $p_{h,D}$ | Cache hit at D (CMPC policy) |
| $p_{h,S}$ | Cache hit at S (CMPC policy) |
| $q_S$ | S attempts to transmit to D if $Q \neq 0$ (or S is available to serve D) |
| $q_C$ | S attempts to transmit to U |
| $q_D$ | D attempts to transmit to U |
| $P_{s \rightarrow j}$ | Success probability of link $s \rightarrow j$, when $s$ is transmitting |
| $P_{s \rightarrow j/k}$ | Success probability of link $s \rightarrow j$, when $s$ and $k$ are transmitting |
| $P(i \Rightarrow j)$ | Queued node $i$ attempts to transmit to $j$ |
Operation of U

S will transmit to U
Q=0
Is S attempting transmission to U?

Redirect to DC
N
Y
U will retry in a subsequent time slot

Is content cached at D?
N
Y
D will transmit to U

Is S attempting transmissions to D?
N
Y
Is S attempting transmission to U?

Is content cached at S?
N
Y
S will transmit to U

Is DC available?
N
Y
Is the transmission successful?

U successfully receives content

Fig. 2: Operation of U in the described protocol.

Operation of S

Q=0
Is S attempting transmission to U?

Is content cached at S?
N
Y
Is D attempting transmission to U?

S will transmit from its queue to D

Is S attempting transmissions to D?
N
Y
End

Y
D successfully receives from S
D will retry in a subsequent time slot

U successfully receives from S
U will retry in a subsequent time slot

U will transmit to U

Is the transmission successful?
Y
D will retry in a subsequent time slot

U successfully receives from S

Fig. 3: Operation of S when user U requests a file f from external resources.

Operation of D

U externally requests content

Is D attempting transmission to U?

Is content cached at D?
N
Y
Is the transmission successful?

Y
U successfully receives from D
U will try again in a future time slot

N
D will transmit to U

Y
U successfully receives from D

N
End

Fig. 4: Operation of D when user U requests a file f from external resources.
realized by user $U$. By denoting the former with $T_S$ and the latter with $T_U$, the weighted sum throughput is given by:
\[
 wT_S + (1 - w)T_U, \text{for } w \in [0, 1].
\] (1)
The average service rate of caching helper $S$ is:
\[
 \mu = q_S(1 - q_U)P_{S \to D} + q_Uq_SP_{D \to D} + \alpha q_S P_{S \to D/DC} + \alpha q_U P_{S \to D}.
\] (2)

As a corollary of the Loynes theorem [39], we obtain that if the arrival and the service process of a queue are strictly jointly stationary and the queue’s average arrival rate is less than the queue’s average service rate, then the queue is stable. Thus, in our model, the queue at helper $S$ is stable if and only if $\lambda < \mu$. Finite queueing delay is a ramification of a stable queue, and, hence, by adding the aforementioned constraint we can enforce finite queueing delay on our wireless system. Moreover, the stability at $S$ also implies that packets arriving at the queue will eventually be transmitted.

The throughput from $S$ to $D$, denoted by $T_S$, depends on the stability of the queue at $S$ and is $T_S = \lambda$ if the queue is stable or $T_S = \mu$ otherwise. Consequently:
\[
 T_S = \mathbb{1}(\lambda < \mu)\lambda + \mathbb{1}(\lambda \geq \mu)\mu,
\] (3)
with $\mathbb{1}(\cdot)$ denoting the indicator function.

The throughput realized by $U$, denoted by $T_U$, depends on whether the queue at $S$ is empty or not. The former happens with probability $P(Q = 0)$ and the latter with probability $P(Q \neq 0)$. Therefore:
- If the queue at $S$ is empty and $U$ requests a file from external resources. In this case, $U$ will be served: (i) by $D$ with probability $q_D$, or (ii) by $S$ with probability $q_C$ in case of $D$’s failure, or (iii) by the data center with probability $\alpha$ in case both helpers fail.
- If the queue at $S$ is non-empty and $U$ requests a file from external resources, then there are two cases: either (i) helper $S$ attempts transmission to the destination helper $D$ (which happens with probability $q_S$) or (ii) helper $S$ is available to serve $U$. In the first case, $U$ will be served by $D$ with probability $q_D$ or by the data center in case $D$ fails to serve $U$. In the second case, $U$ will be served by $D$ with probability $q_D$, or by $S$ with probability $q_C$ in case $D$ fails, or by the data center in case both helpers fail to serve $U$.

Considering all the details above, the throughput realized by user $U$ is:
\[
 T_U = P(Q = 0)q_U [q_D p_{D \to U} + (1 - q_D p_{D \to U}) q_C p_{S \to U} + (1 - q_D p_{D \to U}) (1 - q_C p_{S \to U})] + P(Q \neq 0) q_U q_S [q_D p_{D \to U/0} + (1 - q_D p_{D \to U/0}) q_C p_{S \to U/0} + (1 - q_D p_{D \to U/0}) (1 - q_C p_{S \to U/0})] + P(Q \neq 0) q_U (1 - q_S) [q_D p_{D \to U} + (1 - q_D p_{D \to U}) q_C p_{S \to U} + (1 - q_D p_{D \to U}) (1 - q_C p_{S \to U})].
\] (4)

where we have to differentiate cases of stable/unstable queue due to different $P(Q = 0)$ and $P(Q \neq 0)$ for each case. When the queue at $S$ is stable, the probability that $Q$ is not empty is given by: $P(Q \neq 0) = \lambda/\mu$.

In case the average arrival rate is greater than the average service rate i.e., $\lambda > \mu$, then the queue at $S$ is unstable and can be considered saturated. Consequently, we can apply a packet dropping policy to stabilize the system and the results for the stable queue can be still valid.

If the queue at $S$ is unstable, the throughput realized by $U$ is:
\[
 T_U' = q_U q_S [q_D p_{D \to U/0} + (1 - q_D p_{D \to U}) \alpha p_{D \to U/0}] + q_U (1 - q_S) [q_D p_{D \to U} + (1 - q_D p_{D \to U}) q_C p_{S \to U} + (1 - q_D p_{D \to U}) (1 - q_C p_{S \to U})].
\] (5)

We formulate the following mathematical optimization problem to optimize the probabilities $q_S$, $q_C$, and $q_D$ that maximize the weighted sum throughput when the queue at helper $S$ is stable:
\[
\begin{align*}
\text{max.} & \quad w\lambda + (1 - w)\lambda T_U \\
\text{s.t.} & \quad 0 \leq \lambda < \mu \\
& \quad 0 \leq q_D, q_C, q_D \leq 1
\end{align*}
\] (6a) (6b) (6c)

The first constraint ensures the stability of the queue at helper $S$ and the second one defines the domain for the decision variables. To solve the aforementioned problem for the case in which the queue at $S$ is unstable, we have to drop the first constraint and replace the expressions for $\lambda$ and $T_U$ with the ones for $\mu$ and $T_U'$, respectively. In Section IV, we provide results for maximizing the weighted sum throughput for some practical scenarios.

### IV. DELAY ANALYSIS

Delay experienced by users is another critical performance metric concerning wireless caching systems. In this section, we study the delay that user $U$ experiences when requesting cachable content from external sources until that content is received. Let $P(S = D) = p_s |P(Q \neq 0) = 0, P(S = D) = 1 - P(S = D)$ and $\bar{q}_i = 1 - q_i$.

The average delay that user $U$ experiences to receive a file from external resources is:
\[
 D_U = p_{D \to U}[P(S = D)[(1 - q_D) D_{DC,1,1} + q_D P_{D \to U/S} + q_D P_{D \to U/S}],
\] (7)
where $D_{S_i}$ is the delay to receive the file from $S$ given $D$ misses it:

$$D_{S_i} = q_CP_{S ightarrow U} + q_CP_{S ightarrow U}(1 + D_{S_i}) + q_DDC_{DC,0,S}, \quad (8)$$

and $D_{S_2}$ is the delay to receive the file from $S$ given $D$ caches it but does not attempt i.e., $q_D = 0$, transmissions to $U$:

$$D_{S_2} = q_CP_{S ightarrow U} + (1 - q_CP_{S ightarrow U})(1 + D). \quad (9)$$

We also need to compute delay caused by the data center $DC$:

$$D_{DC} = P(S \Rightarrow D)[\alpha P_{DC ightarrow U/S} + (1 - \alpha P_{DC ightarrow U/S})(1 + D_{DC})] + P(S \neq D)[\alpha P_{DC ightarrow U} + (1 - \alpha P_{DC ightarrow U})(1 + D_{DC})]. \quad (10)$$

Additionally, we need to calculate the following:

$$D_{DC,0,S} = \alpha P_{DC ightarrow U} + (1 - \alpha P_{DC ightarrow U})(1 + D_{S_i}), \quad (11)$$

$$D_{DC,1,S} = \alpha P_{DC ightarrow U/S} + (1 - \alpha P_{DC ightarrow U/S})(1 + D_{S_i}), \quad (12)$$

$$D_{DC,0,D} = \alpha P_{DC ightarrow U} + (1 - \alpha P_{DC ightarrow U})(1 + D_{D}), \quad (13)$$

$$D_{DC,1,D} = \alpha P_{DC ightarrow U/S} + (1 - \alpha P_{DC ightarrow U/S})(1 + D_{D}), \quad (14)$$

and:

$$D_{D} = P(S \Rightarrow D) \times$$

$$\times \{q_D|P_{D ightarrow U/S} + P_{D ightarrow U/S}(1 + D_{D})| + q_D D_{DC,1,D} \} +$$

$$+ P(S \neq D)\{q_D|P_{D ightarrow U} + P_{D ightarrow U}(1 + D_{D})| +$$

$$+ q_D(p_{h}S_{D_2} + \bar{p}_{hS}D_{DC,0,D})\} \quad (15)$$

As one can observe, (7) - (15) are recursively defined. After some basic manipulations, (10) becomes:

$$D_{DC} = (\alpha \{P(S \Rightarrow D)|P_{DC ightarrow U/S} - P_{DC ightarrow U}| + P_{DC ightarrow U}) \}^{-1}. \quad (16)$$

Assuming that $q_CP_{S ightarrow U} - qC \neq 1, (8)$ becomes:

$$D_{S_i} = (qC P_{S ightarrow U} + \alpha(1 - qC)\alpha P_{DC ightarrow U}^{-1} \quad (17)$$

Assuming that $qC P_{S ightarrow U} + (1 - qC)\alpha P_{DC ightarrow U} \neq 0$ and using (17), (11) and (12) become:

$$D_{DC,0,S} = 1 + \frac{1 - \alpha P_{DC ightarrow U}}{qC P_{S ightarrow U} + (1 - qC)\alpha P_{DC ightarrow U}}, \quad (18)$$

$$D_{DC,1,S} = 1 + \frac{1 - \alpha P_{DC ightarrow U/S}}{qC P_{S ightarrow U} + (1 - qC)\alpha P_{DC ightarrow U}}. \quad (19)$$

Using (9), (13), (14) and applying the regenerative method, we get:

$$D_{D} = q_D P(S \neq D)q_Cp_{h}S_{P_{S ightarrow U}} +$$

$$+ q_D \alpha[P(S \Rightarrow D)P_{DC ightarrow U/S}|P(S \neq D)|p_{hS}P_{DC ightarrow U}| +$$

$$+ q_D|P_{D ightarrow U} + P(S \Rightarrow D)|P_{D ightarrow U/S} - P_{D ightarrow U}]. \quad (20)$$

Substituting (20) to (9), (13), and (14) yields expressions for $D_{S_2}, D_{DC,0,D}$, and $D_{DC,1,D}$, respectively, that are functions of link success probabilities (see Table III and II) and cache parameters (see Table III) only.

V. Numerical Results

In this section, we present numerical evaluations of the analysis in the previous sections. The parameters we used for the wireless links between wireless nodes can be found in Table II. The helpers apply the CMPC policy as described in Section II-B. We consider a finite content library of files, $\mathcal{F} = \{f_{1}, \dots, f_{N}\}$, to serve users requests. For the sake of simplicity, we assume that all files have equal size and that access to cached files happens instantaneously. The i-th most popular file is denoted as $f_{i}$, and the request probability of the i-th most popular file is given by: $p_{i} = \Omega / i^\delta$, where $\Omega = (\sum_{i=1}^{N}j^{-\delta})^{-1}$ is the normalization factor and $\delta$ is the shape parameter of the Zipf law which determines the correlation of user requests.

Therefore, the probability that user $U$ requests a file that is not located in its cache is:

$$q_{U} = 1 - \sum_{i=1}^{M_{U}} p_{i}, \quad (21)$$

the cache hit probabilities at the caching helper $D$ and $S$ are respectively given by:

$$p_{hD} = \sum_{i=M_{U}+1}^{M_{U}+M_{D}} p_{i}, \quad (22)$$

$$p_{hS} = \sum_{i=M_{U}+M_{D}+1}^{M_{U}+M_{D}+M_{S}} p_{i}. \quad (23)$$

In the following results, we study the maximum weighted sum throughput which is defined as $T_{w} = wT_{S} + (1 - w)T_{U}$ or $T_{w} = wT_{S} + (1 - w)T'_{U}$ when the queue is stable or unstable, respectively. The expressions for $T_{S}, T_{U}$, and $T'_{U}$ are given by (13) - (15) in Section III. In order to maximize the weighted sum throughput, we solved the optimization problem (6a) - (6c) using the Gurobi optimization solver and report the results.

A. Maximum Weighted Sum Throughput vs. Average Arrival Rate $\lambda$

We consider a scenario where the wireless links parameters follow the values in Table III. The cache sizes and cache hit probabilities are set as per Table III for two different values for the variable $\delta$ of the standard Zipf law for the popularity distribution that the cached files follow.

### Table II: Wireless links parameters.

| Parameter     | Value  |
|---------------|--------|
| $P_{tx}(S)$   | 1 mW   |
| $P_{tx}(D)$   | 0.5 mW |
| $P_{tx}(DC)$  | 10 mW  |
| $n$           | $10^{-11}$ W |
| $r_{S ightarrow D}$ | 50 m |
| $r_{D ightarrow U}$ | 50 m |
| $r_{S ightarrow U}$ | 40 m |
| $r_{DC ightarrow U}$ | 80 m |
| $r_{DC ightarrow D}$ | 100 m |

| Parameter     | Value  |
|---------------|--------|
| $P_{S ightarrow U}$ | 0.963     |
| $P_{D ightarrow U}$ | 0.607     |
| $P_{DC ightarrow U}$ | 0.849     |
| $P_{DC ightarrow U/S}$ | 0.779      |
| $P_{S ightarrow D}$ | 0.779      |
| $P_{S ightarrow D}/D$ | 0.779     |
| $P_{S ightarrow D}/DC$ | 0.223      |
| $P_{D ightarrow U/S}$ | 0.029      |
| $\gamma_1$     | 0 dB    |
| $\gamma_2$     | 0 dB    |
TABLE III: Caches parameters and hit probabilities for different values of $\delta$.

| Parameter | Value | Parameter | Value |
|-----------|-------|-----------|-------|
| $M_U$     | 200   | $\delta$  | 0.5   |
| $M_D$     | 1000  | $\delta$  | 1.2   |
| $M_S$     | 2000  | $p_{h,U}$ | 0.865 |
| $F$       | 10000 | $p_{h,S}$ | 0.221 |

In Fig. 5 the maximum weighted sum throughput versus the average arrival rate $\lambda$ at helper $S$ is presented for three different values of $w$ when the queue at $S$ is stable. We chose: (i) $w = 1/4$ as a representative case in which $T_U$ is more important than $T_S$, (ii) $w = 2/4$ to equalize the importance of $T_U$ and $T_S$, and (iii) $w = 3/4$ to put more emphasis on the importance of $T_S$ versus $T_U$.

In case $w = 1/4$, the maximum weighted sum throughput is a decreasing function of $\lambda$ when $\delta = 0.5$ (see Fig. 5(a)), but increasing when $\delta = 1.2$ (see Fig. 5(b)). When $w = 2/4$, the maximum weighted sum throughput is almost constant for any value of $\lambda$ when $\delta = 0.5$ and increases with $\lambda$ for $\delta = 1.2$. Regarding $w = 3/4$, the maximum weighted sum throughput is an increasing function of $\lambda$ for any $\delta$ value since $T_S$ clearly dominates $T_U$ in this case.

![Figure 5: The maximum weighted sum throughput vs. $\lambda$ for different values of $w$.](image)

Furthermore, it is observed that the maximum weighted sum throughput is achieved when $q^*_U = 1$ for any value of $w$ and $\lambda$ when the queue at $S$ is stable (see Table IV), but this is not the case when the queue is unstable i.e., the average arrival rate $\lambda$ is greater than the average service rate $\mu$ (see Table V for different values of $\delta$). In case the queue at $S$ is unstable, it is optimal for helper $S$ to avoid transmissions i.e., $q^*_S = 0$, to $U$ when $\delta = 1.2$ for any values of $w$ and $\lambda$. However, this is not the case when $\delta = 0.5$ since helper $S$ must always attempt transmissions to $U$ i.e., $q^*_S = 1$, when $w \in \{1/4, 2/4\}$ to achieve the maximum weighted sum throughput.

TABLE IV: The values of $q^*_U$, $q^*_S$, $q^*_D$ for which the weighted sum throughput is maximized and the queue at $S$ is stable for $\alpha = 0.7$, $M_U = 200$, $M_D = 1000$, and $M_S = 2000$.

| $w$  | $T^*_U$ | $q^*_S$ | $q^*_D$ | $T^*_U$ | $q^*_S$ | $q^*_D$ |
|------|---------|---------|---------|---------|---------|---------|
| 1/4  | 0.423   | 0.029   | 1       | 0       | 0.187   | 0.988   |
| 2/4  | 0.286   | 0.978   | 1       | 1       | 0.358   | 0.985   |
| 3/4  | 0.392   | 0.996   | 1       | 1       | 0.536   | 0.999   |

TABLE V: The values of $q^*_U$, $q^*_S$, $q^*_D$ for which the weighted sum throughput is maximized and the queue at $S$ is unstable for $\alpha = 0.7$, $M_U = 200$, $M_D = 1000$, and $M_S = 2000$.

| $w$  | $T^*_U$ | $q^*_S$ | $q^*_D$ | $T^*_U$ | $q^*_S$ | $q^*_D$ |
|------|---------|---------|---------|---------|---------|---------|
| 1/4  | 0.430   | 0       | 1       | 0       | 0.189   | 1       |
| 2/4  | 0.280   | 0       | 1       | 0       | 0.363   | 1       |
| 3/4  | 0.399   | 0.024   | 1       | 0.537   | 1       |

B. Maximum Weighted Sum Throughput vs. Cache Size $M_U$

In this section, we study how the cache size $M_U$ affects the maximum weighted sum throughput. Recall that $q^*_U$ decreases as $M_U$ increases. We consider two different values for $\delta$, same as previously, to examine how $\delta$ affects maximum weighted sum throughput given different values for $M_U$.

In Fig. 6 the maximum weighted sum throughput versus $M_U$ is presented for $\alpha = 0.7$, $M_D = 1000$, $M_S = 2000$ and $\lambda = 0.4$ for which the queue at $S$ is stable. We observe that as the cache size at $U$ increases, the maximum weighted sum throughput remains almost constant when $\delta = 0.5$ and slightly decreases when $\delta = 1.2$. This is expected since increasing cache size at $U$ results in fewer requests for files from external results. Moreover, the maximum weighted sum throughput is higher when the value of $\delta$ is lower since, for a given cache size e.g. $M_U = 200$, the probability of requesting content from external resources decreases as $\delta$ is increased.

In Fig. 7 the maximum weighted sum throughput versus $M_U$ is presented for the same parameters as in the Fig. 6 but unstable queue at $S$. The maximum weighted sum throughput is an increasing function of $M_U$ for every value of $\delta$ when $w \in \{2/4, 3/4\}$. This is expected since, for these values of $w$, the throughput achieved by $D$ i.e., $T_S$, dominates $T_U$, and $T_S$ is increasing due to the decrease of requests to external content by $U$ (recall that as $M_U$ increases, $q^*_D$ decreases).

When $w = 1/4$, the maximum weighted sum throughput is
almost constant ($\delta = 1.2$) or decreases ($\delta = 0.5$) as $M_U$ increases. The latter decrease can be attributed to the fact that $T_U$ i.e., the dominant term in the maximum weighted sum throughput, decreases as $q_U$ decreases (and $M_U$ increases).

The values of $q_S^*, q_C^*$, and $q_D^*$ that achieve the maximum weighted sum throughput are given in Table VII when the queue at $S$ is stable and unstable respectively.

In case $\delta = 0.5$ and the queue at $S$ is stable, the maximum weighted sum throughput $T_w$ is achieved for $q_C^* = 1$ and $q_D^* = 1$ for every value of $M_U$ and $w$. This means that, for the aforementioned parameters, user $U$ should always be assisted by both $S$ and $D$ to achieve maximum weighted sum throughput $T_w$. This is not the case for $\delta = 1.2$ while the queue at $S$ is stable and $M_U \geq 400$. For every value of $w \in \{1/4, 2/4, 3/4\}$, user $U$ should only be assisted by $S$ to achieve the maximum weighted sum throughput $T_w$ since $q_C^* = 1$ and $q_D^* = 0$. We also observe that, in this case, $S$ should more frequently assist $D$ since $q_S^*$ has almost always a higher value compared to $\delta = 0.5$.

In case the queue at $S$ is unstable, the values of $(q_S^*, q_C^*, q_D^*)$ for which the maximum weighted sum throughput $T_w$ is achieved can be found in Table VII. We observe that neither helper $S$ should serve helper $D$ ($q_S^* = 0$) nor the latter should assist $U$ ($q_D^* = 0$) to maximize $T_w$ when (i) $\delta = 0.5$ and $w = 1/4$ for any cache size $M_U$ or (ii) $\delta = 0.5$, $w = 2/4$ and user $U$’s cache can hold $M_U = 200$ files.

Moreover, when $\delta = 0.5, M_U \geq 400$ and $w \in \{2/4, 3/4\}$, helper $S$ should only serve helper $D$ and the latter should assist user $U$ since $(q_S^*, q_D^*) = (1, 1)$. However, helper $S$ should slightly assist $U$ in some cases when e.g., $M_U = 400$ or 600. When $\delta = 1.2$, helper $S$ should only serve the destination helper $D$ and the latter should assist user $U$ for any value of $M_U$ and $w$. Additionally, helper $S$ should not assist user $U$ for any cache size $M_U$ but 400.

Furthermore, it should be noted that, for any value of $M_U$, the maximum weighted sum throughput is decreasing as $\delta$ increases when $w = 1/4$ and increases as $\delta$ increases when $w \in \{2/4, 3/4\}$.

C. Maximum Weighted Sum Throughput vs. Average Arrival Rate $\lambda$ when $M_D = 0$

We consider a scenario where the system parameters are the same as in Section V-A (see Tables II and III), but helper $D$ cannot assist user $U$ since its cache cannot hold any files i.e., $M_D = 0$. Consequently, $q_D = 0$ and $p_{hD} = 0$ as well. This scenario will allow the study of the maximum weighted sum throughput versus $\lambda$ when only one of the two helpers, the least powerful, is unable satisfy $U$’s needs for content from external resources.
TABLE VI: The values of \((q^*_2, q^*_2, q^*_T)\) that maximize the weighted sum throughput \(T_w\) for different values of \(M_U\) when \(\alpha = 0.7\) and the queue at \(S\) is stable.

| \(M_U\) | \(\delta = 0.5\) | \(\delta = 0.5\) | \(\delta = 0.5\) |
|---------|----------------|----------------|-----------------|
|         | \(w = 1/4\)   | \(w = 2/4\)   | \(w = 3/4\)   |
| 200     | \(0.227\)     | \(0.285\)     | \(0.342\)     |
|         | \((0.999, 1.1)\) | \((0.999, 1.1)\) | \((0.999, 1.1)\) |
| 400     | \(0.224\)     | \(0.283\)     | \(0.341\)     |
|         | \((0.801, 1.1)\) | \((0.824, 1.1)\) | \((0.769, 1.1)\) |
| 600     | \(0.221\)     | \(0.281\)     | \(0.340\)     |
|         | \((0.756, 1.1)\) | \((1.1, 1)\) | \((1.1, 1)\) |
| 800     | \(0.219\)     | \(0.279\)     | \(0.340\)     |
|         | \((1.1, 1)\) | \((1.1, 1)\) | \((0.746, 1.1)\) |
| 1000    | \(0.217\)     | \(0.278\)     | \(0.339\)     |
|         | \((0.732, 1.1)\) | \((1.1, 1)\) | \((1.1, 1)\) |

TABLE VII: The values of \((q^*_2, q^*_2, q^*_T)\) that maximize the weighted sum throughput \(T_w\) for different values of \(M_U\) when \(\alpha = 0.7\) and the queue at \(S\) is unstable.

| \(M_U\) | \(\delta = 0.5\) | \(\delta = 1.2\) | \(\delta = 1.2\) |
|---------|----------------|----------------|----------------|
|         | \(w = 1/4\)   | \(w = 2/4\)   | \(w = 3/4\)   |
| 200     | \(0.145\)     | \(0.230\)     | \(0.315\)     |
|         | \((0.713, 1.1)\) | \((0.713, 1.1)\) | \((0.713, 1.1)\) |
| 400     | \(0.135\)     | \(0.223\)     | \(0.312\)     |
|         | \((1.1, 0)\) | \((0.999, 1.0)\) | \((0.999, 1.0)\) |
| 600     | \(0.129\)     | \(0.220\)     | \(0.310\)     |
|         | \((1.1, 0)\) | \((1.1, 0)\) | \((1.1, 0)\) |
| 800     | \(0.126\)     | \(0.217\)     | \(0.309\)     |
|         | \((1.1, 0)\) | \((1.1, 0)\) | \((1.1, 0)\) |
| 1000    | \(0.123\)     | \(0.216\)     | \(0.308\)     |
|         | \((1.1, 0)\) | \((1.1, 0)\) | \((0.540, 1.0)\) |

In Fig. 8 we plot the maximum weighted sum throughput versus \(\lambda\) when the queue at \(S\) is stable and \(M_D = 0\). We observe that, when \(\delta = 0.5\), the maximum weighted sum throughput (i) is a decreasing function of \(\lambda\) for \(w = 1/4\), (ii) slightly decreases for \(w = 2/4\), and (iii) increases for \(w = 3/4\). Recall that, by definition, in the first case \(T_{S}\) dominates \(T_{S}\), in the second case both throughput terms contribute equally, and in the third case \(T_{S}\) dominates \(T_{U}\). When \(\delta = 1.2\), the maximum weighted sum throughput is increasing with \(\lambda\). We observe that higher values of \(w\) yield steeper increases in the maximum weighted sum throughput.

When the queue at \(S\) is stable, the maximum weighted sum throughput is always achieved when \(q^*_S = 1\) for any \(w, \delta\), and \(\lambda\) using the system parameters we quoted before. However, in case \(\delta = 0.5\), helper \(S\) should nearly always assist \(U\) since \(q^*_S \in \{0.977, 0.999\}\). When \(\delta = 1.2\), helper \(S\) should always assist \(U\) as Table VIII depicts.

On the other hand, when the queue at \(S\) is unstable and \(\delta = 0.5\), helper \(S\) should only assist \(U\) when \(w \in \{1/4, 2/4\}\) and only assist \(D\) when \(w = 3/4\) (see Table IX). This is expected since in the latter case, \(T_{S}\) dominates \(T_{U}\) and, hence, it is preferable that \(S\) always serves \(D\) to maximize the contribution of \(T_{S}\). In this case, if user \(U\) requests content from external resources, it will be only served by the data center. Moreover, when \(\delta = 1.2\), it is optimal that helper \(S\) serves only \(D\) for any value of \(w\).
TABLE VIII: The values of $q_S^*$ and $q_D^*$ for which the weighted sum throughput is maximized when the queue at $S$ is stable, $\alpha = 0.7, M_U = 200, M_D = 0$, and $M_S = 2000$.

| $M_D = 0$ | $\delta = 0.5$ | $\delta = 1.2$ |
|-----------|----------------|----------------|
| $w$       | max. $T_w^*$   | $q_S^*$        |
| 1/4       | 0.445          | 0.994          |
| 2/4       | 0.300          | 0.977          |
| 3/4       | 0.348          | 0.999          |

TABLE IX: The values of $q_S^*$ and $q_D^*$ for which the weighted sum throughput is maximized when the queue at $S$ is unstable, $\alpha = 0.7, M_U = 200, M_D = 0$, and $M_S = 2000$.

| $M_D = 0$ | $\delta = 0.5$ | $\delta = 1.2$ |
|-----------|----------------|----------------|
| $w$       | max. $T_w^*$   | $q_S^*$        |
| 1/4       | 0.451          | 0.187          |
| 2/4       | 0.301          | 0.350          |
| 3/4       | 0.349          | 0.531          |

D. Maximum Weighted Sum Throughput vs. Average Arrival Rate $\lambda$ when $M_S = 0$

Here, we study the maximum weighted sum throughput versus the average arrival rate $\lambda$ when node $S$ is not equipped with cache i.e., $M_S = 0$, and, hence, $q_C = 0$ and $p_{0S} = 0$. The parameters of helper $D$'s cache and the wireless links can be found in Tables III and II respectively.

In Fig. 9 we plot the maximum weighted sum throughput versus $\lambda$ for which the queue at $S$ is stable when $M_S = 0$ for different values of $w$. Regarding $\delta = 0.5$, when $w = 1/4$, the maximum weighted sum throughput is decreasing with $\lambda$. When $w \in \{2/4, 3/4\}$, and $\delta \in \{0.5, 1.2\}$ or $w = 1/4$ and $\delta = 1.2$, the maximum weighted sum throughput is an increasing function of $\lambda$.

TABLE X: The values of $q_S^*$ and $q_D^*$ for which the weighted sum throughput is maximized when the queue at $S$ is stable, $\alpha = 0.7, M_U = 200, M_D = 1000$, and $M_S = 0$.

| $M_S = 0$ | $\delta = 0.5$ | $\delta = 1.2$ |
|-----------|----------------|----------------|
| $w$       | max. $T_w^*$   | $q_S^*$        |
| 1/4       | 0.383          | 0.993          |
| 2/4       | 0.286          | 1.1            |
| 3/4       | 0.479          | 1.1            |

In Table XI we present the values of $q_S^*$ and $q_D^*$ that achieve maximum weighted sum throughput when the queue at $S$ is stable for different values of $w$. Recall that, in this specific scenario, $q_C^* = 0$ since helper $S$ has no cache and, thus, it cannot assist $U$. Therefore, $S$ is only useful to helper $D$. We observe that the maximum weighted sum throughput is lowered compared to the case when $M_D = 0$ for $\delta = 0.5$ and slightly higher for $\delta = 1.2$ (compare with Table VIII). Additionally, helper $S$ should almost always serve $D$ and the later should always assist user $U$ to achieve the maximum weighted sum throughput. In Table XI we present the values of $q_S^*$ and $q_D^*$ that achieve maximum weighted sum throughput when the queue at $S$ is unstable for different values of $\lambda$. In order to maximize the weighted sum throughput, helper $S$ should always serve $D$ for any values of $\delta$ and $w$ apart from the case in which $w = 1/4$ and $\delta = 0.5$ for which $S$ should remain silent since $q_S^* = 0$. Furthermore, helper $D$ should always assist $U$ requests for every value of $\delta$ and $w$ we used. The maximum weighted sum throughput is higher compared to the case in which $M_D = 0$ (compare with Table XI) for every value of $\delta$ and $w$ apart from the cases in which $\delta = 0.5$ and $w \in \{1/4, 2/4\}$.

E. Average delay at user $U$

Here, we present the numerical results of the average delay experienced by user $U$ to receive content from external sources. The delay analysis can be found in Section IV.

In the following plots, we study how the average arrival rate $\lambda$, the data center’s random availability $\alpha$, the probability that $S$ attempts transmissions to $D$, $q_S$, the probability that $D$
attempts transmissions to $U$, $q_D$, and the cache size at $U$, $M_U$ affect the average delay at $U$. The wireless links characteristics can be found in Table III. The cache sizes were set to hold $M_S = 2000$ and $M_D = 1000$ files at $S$ and $D$, respectively and we used two different values for $\delta$ to examine its effect on the realized average delay. Hence, the values of $q_U$, $p_{hD}$, and $p_{hS}$ were given by (21) - (23) depending on $\delta$. Also, we set $q_c = 0.5$.

In Fig. 10, the average delay versus the arrival rate at helper $S$ is depicted for $q_S = 0.9$, $q_D = 0.8$, $\alpha = 0.7$ and $M_U = 200$. We observe that the delay increases with the arrival rate and the increase rate is steeper when $\delta = 0.5$ compared to $\delta = 1.2$. As we explained in Section II-B, higher values of $\delta$ yield more requests for a few most popular files. Therefore, for a given $M_U$, the higher the $\delta$, the lower the $q_U$ i.e., user $U$ requests fewer files from external sources with lower probability, as well as lower value for cache hits $p_{hD}$ and $p_{hS}$ (for given $M_D$ and $M_S$). Fewer requests for files from external resources require fewer transmissions to $U$ and, hence, less interference is realized. Consequently, less average delay is experienced at $U$.

In Fig. 11 we present the average delay at $U$ versus data center’s availability for two cases of arrival rate $\lambda = 0.2$ and $\lambda = 0.4$. We observe that the delay is lower when $\lambda = 0.2$ since a higher average arrival rate is more likely to create a congested queue at $S$ and, consequently, a higher delay. In case $\lambda = 0.2$, the delay is decreased with the increase of $\alpha$ and the queue at helper $S$ is stable for any $\alpha \in [0.2, 1]$. Additionally, the decrease is steeper with $\alpha$ when $\delta = 1.2$. When $\lambda = 0.4$ and $\delta = 0.5$, the queue at $S$ remains stable for $\alpha \in [0.2, 0.8]$ and the delay has the non-monotonic behavior of Fig 11 (b). For $\alpha \in [0.8, 1]$, the average delay starts decreasing with $\alpha$ and the queue at $S$ is unstable. When $\delta = 1.2$, the queue at $S$ is stable for every value of $\alpha$ and the delay is decreased with the increased availability of the data center.

In Fig. 12 we plot the average delay at $U$ versus $q_S$ for $\lambda = 0.2$ and $\lambda = 0.4$. We observe that as long as the queue at $S$ is unstable, the delay increases with the $q_S$ increase. This is expected since as $q_S$ increases, helper $S$ attempts more transmissions to helper $D$ and, consequently, it is not only less likely to assist $U$ but also $U$’s probability to find an available helper is decreased (since the $S-D$ pair communicates). Regarding the case in which the queue at $S$ is stable, increasing $q_S$ does not contribute to delay’s improvement. Moreover, a lower value of $q_S$ is required to achieve queue stability at $S$ when $\lambda = 0.2$ compared to $\lambda = 0.4$. This is expected since a higher average arrival rate requires a higher average service rate to maintain queue stability.

In Fig. 13 we demonstrate the average delay at $U$ versus $q_D$ for $\lambda = 0.2$ and $\lambda = 0.4$. In the former case, the delay is slightly decreased with the increase of $q_D$. This can be attributed to helper $D$’s increased assistance that yields more transmissions to $U$ and, hence, potentially decreased delay. When $\lambda = 0.4$, the average delay decreases considerably with $q_D$ when $\delta = 0.5$ due to the increased assistance of helper $D$, but decreases slightly in case $\delta = 1.2$. This is expected, as we previously explained, since higher values of $\delta$ create more requests for a few most popular files, and, thus, $U$’s request for external content is decreased. As a result, the average delay at $U$ is decreased compared to lower $\delta$ values.

In Fig. 14 we show the average delay at $U$ versus the cache size at $U$ for $\lambda = 0.2$ and $\lambda = 0.4$. The cache size $M_U$ affects the request probability for external content, $q_U$, and as it increases, the $q_U$ decreases. In any case, the queue at $S$ is stable. When $\lambda = 0.2$, the effect of $M_U$ on the average delay at $U$ is minor. However, when average arrival rate $\lambda$ is increased, then increasing cache size at $U$ decreases the average delay.
especially when $\delta$ is lowered.

VI. CONCLUSION

In this paper, we studied the effect of multiple randomly available caching helpers on a wireless system that serves cachable and non-cachable traffic. We derived the throughput for a system consisting of a user requesting cachable content from a pair of caching helpers within its proximity or a data center. The helpers are assumed to exchange non-cachable content as well as assisting the user’s needs for cachable content in a random manner. We optimized the probabilities by which the helpers assist the user’s requests to maximize the system throughput. Moreover, we studied the average delay experienced by the user from the time it requested cachable content till content reception.

Our theoretical and numerical results provide insights concerning the system throughput and the delay behavior of wireless systems serving both cachable and non-cachable content with assistance of multiple randomly available caching helpers.

REFERENCES

[1] “Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017-2022 White Paper,” 2019. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.html

[2] G. Paschos, E. Bastug, I. Land, G. Caire, and M. Debbah, “Wireless caching: technical misconceptions and business barriers,” IEEE Communications Magazine, vol. 54, no. 8, pp. 16–22, Aug. 2016.

[3] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, “FemtoCaching: Wireless Content Delivery Through Distributed Caching Helpers,” IEEE Transactions on Information Theory, vol. 59, no. 12, pp. 8402–8413, Dec. 2013.

[4] G. S. Paschos, G. Iosifidis, M. Tao, D. Towsley, and G. Caire, “The Role of Caching in Future Communication Systems and Networks,” IEEE Journal on Selected Areas in Communications, vol. 36, no. 6, pp. 1111–1125, June 2018.

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

[6] Z. Chen, N. Pappas, and M. Kountouris, “Probabilistic Caching in Wireless D2D Networks: Cache Hit Optimal Versus Throughput Optimal,” IEEE Communications Letters, vol. 21, no. 3, pp. 584–587, Mar. 2017.

[7] E. Batu, M. Kountouris, M. Bengtis, and M. Debbah, “On the delay of geographical caching methods in two-tiered heterogeneous networks,” IEEE 17th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), pp. 1–5, July 2016.

[8] N. Pappas, Z. Chen, and I. Dimitriou, “Throughput and Delay Analysis of Wireless Caching Helper Systems With Random Availability,” IEEE Access, vol. 6, pp. 2856–2867, May 2014.

[9] J. Ma, J. Wang, and P. Fan, “A Cooperation-Based Caching Scheme for Heterogeneous Networks,” IEEE Access, vol. 5, pp. 14 013–14 020, 2017.

[10] Y. Wang, X. Tao, X. Zhang, and Y. Gu, “Cooperative Caching Placement in Cache-Enabled D2D Underloaded Cellular Network,” IEEE Communications Letters, vol. 21, no. 5, pp. 1151–1154, May 2017.

[11] Z. Chen, J. Lee, T. Q. S. Quek, and M. Kountouris, “Cooperative Caching and Transmission Design in Cluster-Centric Small Cell Networks,” IEEE Transactions on Wireless Communications, vol. 16, no. 5, pp. 3401–3415, May 2017.

[12] M. Naslcheraghi, M. Afshang, and H. S. Dhillon, “Modeling and Performance Analysis of Full-Duplex Communications in Cache-Enabled D2D Networks,” IEEE International Conference on Communications (ICC), pp. 1–6, May 2018.

[13] Z. Chen and M. Kountouris, “D2D caching vs. small cell caching: Where to cache content in a wireless network?,” IEEE 17th International
Fig. 14: The average delay at $U$ vs. $M_u$ for $\delta \in \{0.5, 1.2\}$. 

Workshop on Signal Processing Advances in Wireless Communications (SPAWC), pp. 1–6, July 2016.

[14] D. Liu and C. Yang, “Energy Efficiency of Downlink Networks With Caching at Base Stations,” *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 4, pp. 907–922, Apr. 2016.

[15] S. Lin, D. Cheng, G. Zhao, and Z. Chen, “Energy-Efficient Wireless Caching in Device-to-Device Cooperative Networks,” *IEEE 85th Vehicular Technology Conference (VTC Spring)*, June 2017.

[16] B. Chen, C. Yang, and A. F. Molisch, “Cache-Enabled Device-to-Device Communications: Offloading Gain and Energy Cost,” *IEEE Transactions on Wireless Communications*, vol. 16, no. 7, pp. 4519–4536, July 2017.

[17] J. Rao, H. Feng, C. Yang, Z. Chen, and B. Xia, “Optimal caching placement for D2D assisted wireless caching networks,” *IEEE International Conference on Communications (ICC)*, pp. 1–6, May 2016.

[18] D. Malak, M. Al-Shalash, and J. G. Andrews, “Spatially Correlated Content Caching for Device-to-Device Communications,” *IEEE Transactions on Wireless Communications*, vol. 17, no. 1, pp. 56–70, Jan. 2018.

[19] W. Wang, R. Lan, J. Gu, A. Huang, H. Shan, and Z. Zhang, “Edge Caching at Base Stations With Device-to-Device Offloading,” *IEEE Access*, vol. 5, pp. 6399–6410, 2017.

[20] B. Chen, C. Yang, and Z. Xiong, “Optimal Caching and Scheduling for Cache-Enabled D2D Communications,” *IEEE Communications Letters*, vol. 21, no. 5, pp. 1155–1158, May 2017.

[21] Y. Chen and H. Zhang, “Exploiting Transmission and Caching Diversity in Cache-Enabled User-Centric Network: Analysis and Optimization,” *IEEE Access*, vol. 7, pp. 65934–65943, 2019.

[22] K. Thar, N. H. Tran, S. Ullah, T. Z. Oo, and C. S. Hong, “Online Caching and Cooperative Forwarding in Information Centric Networking,” *IEEE Access*, vol. 6, pp. 59679–59694, 2018.

[23] F. Rezaei and B. H. Khalaj, “Stability, Rate, and Delay Analysis of Single Bottleneck Caching Networks,” *IEEE Transactions on Communications*, vol. 64, no. 1, pp. 300–313, Jan. 2016.

[24] D. Bethanabhotla, G. Caire, and M. J. Neely, “Adaptive Video Streaming for Wireless Networks With Multiple Users and Helpers,” *IEEE Transactions on Communications*, vol. 63, no. 1, pp. 268–285, Jan 2015.