Dynamic Cache Rental and Content Caching in Elastic Wireless CDNs

Jeongho Kwak†, Georgios Paschos‡ and George Iosifidis†
†School of Computer Science & Statistics, Trinity College Dublin, Ireland
‡France Research Center, Huawei Technologies, France
E-mail: jeongho.kwak@tcd.ie, georgios.paschos@huawei.com, george.iosifidis@tcd.ie

***Abstract*—With elastic CDNs, content providers can rent cache space on demand at different cloud locations in order to enhance their offered quality of service (QoS). This paper addresses a key challenge in this context, namely how to invest an available budget in cache space in order to match spatio-temporal fluctuations of file demand and storage price. Specifically, we consider jointly *dynamic cache rental*, *file placement*, and *request-cache association* in a wireless scenario in order to provide a just-in-time CDN service. The objective is to maximize the benefit in average download delay obtained by the rented caches, while ensuring that the time-average rental cost is less than a fixed budget. We leverage a Lyapunov drift-minus-benefit technique to transform our infinite horizon problem into day-by-day subproblems which can be solved without knowledge of distant future file popularity and transmission rates. For the case of non-overlapping small cells (also wired case) we provide an efficient subproblem solution, referred to as *JCC*. However, in the general overlapping case, the subproblem becomes a mixed integer non-linear program (MINLP). In this case, we employ a dual decomposition method to derive a scalable solution, namely the *JCCA* algorithm. Finally, via extensive simulations, we reveal that the proposed JCCA algorithm attains 82.66% higher delay benefit than existing static cache storage-based algorithms when available average cache budget is 20% of entire file library; moreover, the benefit becomes higher as the average cache budget gets tighter.**

I. Introduction

Background and Motivation. The growing traffic in wireless networks has popularized the use of content delivery networks (CDNs) for improving quality of service and reducing traffic costs. A large portion of the Internet traffic today is handled by CDNs owned by large content companies like Google and Netflix. Such deployments require a significant—often prohibitive for newcomers—investment for the cache servers and the associated control systems. For a smaller size Content Provider (CP) an alternative is to purchase service from a CDN provider such as Akamai. However, this can be very costly and impractical for some CPs since the leases are on long-term basis, prices are fixed and catalogue-dependent, and the content placement decisions are made by the CDN provider.

A disruptive solution has recently emerged known as content delivery as-a-service or *elastic CDN* (eCDN) [1]. The eCDNs rely on commonly available cloud computing infrastructures and exploit the network function virtualization (NFV) technology to instantiate virtual caches [2]. Solutions like Akamai Aura [3] and Huawei uCDN [4] allow dynamic cache scaling, e.g. to support a sudden traffic surge. The eCDNs therefore enable a novel business model, where small CPs can dynamically rent storage and instantiate virtual CDNs to meet customer demand just-in-time and space, i.e., whenever and wherever caching is needed. This model arrives with a flexible pricing scheme: the first market solutions already offer a fine-grained *pay-as-you-go* service [5]. Clearly, eCDNs can benefit CPs with tight monetary budgets, volatile demand and/or seeking fine-grained caching control over their storage management and caching decisions.

At the same time, this model raises technical and economic questions. In particular, the CP must decide *(i)* how much storage to lease at each location in order to meet user demand and *(ii)* which content items (files) to cache at each of these storage reservoirs. Furthermore, these decisions need to be updated regularly, often on per-day or per-hour basis, in order to accommodate the time-varying nature of the content demand. At each round, the CP encounters an investment dilemma: a larger cache lease will improve service quality but will also increase expenditure. With a given operational expenditure budget, the investment decisions are inherently coupled across different rounds. Overspending in one round improves the current performance but restraints subsequent decisions and limits future opportunities.

Our goal in this work is to *model and optimize the storage management and caching decisions that a content provider needs to make when leasing storage from an eCDN system.*

Scenario and Contributions. We study the more challenging but increasingly relevant scenario where the eCDN is owned by a mobile network operator, and the storage resources are at small cell base stations (SBSs).[1] These *wireless edge caching* architectures were addressed in several papers, e.g., [7]–[9] and they are expected to play a key role in 5G+ networks where low-latency or data-demanding services can be supported by caching-at-the-edge.

In our eCDN mobile network, a geographical area is covered by several SBSs with possibly overlapping coverage. The SBSs have caches which can be dynamically rented by the CP. When a file is served from a leased cache, there is a delay benefit as opposed to be served from a remote CP server, which is attributed to the proximity of the cache. Therefore, the CP can invest in cache space in different time slots and locations, decides which files to place in the rented space, and then enjoys a service delay benefit for the requests that were served from the caches. In this paper, our business model is that the CP gives an average cache rental budget to the mobile operator where all operations including cache scaling, content caching and wireless routing decisions are entrusted to the mobile operators. It would be beneficial to both of the CP and the mobile operator by jointly manipulating

[1]For example, AT&T have envisioned using their own CDN, namely *Telco CDN* which integrates content delivery with traffic engineering [6].

cache scaling, content caching and wireless routing since it enables to reduce backhaul congestions and enhance the quality of service for end users. Then, the objective is to *select investment, placement, and request association in order to maximize the service delay benefit for a given time-average budget*. Our contributions can be summarized as follows.

1) We formulate an important optimization problem for elastic wireless CDNs, which involves storage leasing, file caching and routing decisions. Our generic analysis applies to both wired and wireless systems, and provides a way to match fluctuating demand with cache size scaling.
2) We show that the overall proposed scheme is shown to converge to the maximum achievable service quality, subject to the provided budget. Moreover, the proposed scheme has the advantage that it does not require knowledge of distant future file popularity and average delay.
3) We show that the subproblem admits a simple solution in the non-overlapping SBS case (applies also to wired scenarios), while it becomes a mixed integer non-linear program (MINLP) in the overlapping case. For the latter, we propose a Lagrangian decomposition method which provides the optimal solution of its linear relaxation.
4) Simulations demonstrate the effectiveness of our approach, which attain 82.66% higher average delay benefit than existing static cache-based algorithms when average available cache budget is 20% of entire file library.

The remaining of this paper is organized as follows. We first describe the related work in Section II. Next, we present the system model in Section III. Then, we state the joint cache rental, file caching and region association problems and propose JCC and JCCA algorithms in Section IV. In Section VI, we evaluate the proposed algorithms by extensive simulations. Finally, we conclude this paper in Section VII.

II. Related Work

Cache dimensioning in traditional CDNs. Related to our work are the extensive studies on CDN server placement [10]–[13]. For example, Bektas *et al.* [10] formulated the joint problem of server opening, file placement, and transfer cost minimization, and solved it using Bender's decomposition, while Li *et al.* [12] used dynamic programming. Differently from server placement problems, Laoutaris *et al.* [13] formulated a storage budget allocation problem in a hierarchical file distributed system, and proposed heuristic solutions. Contrary to the above, and other static approaches in the context of traditional CDN, our problem is of dynamic nature, requiring to consider these decisions jointly across multiple time epochs.

Wireless caching in heterogeneous cellular networks. Since the seminal work on femtocaching [7], several techniques, e.g., [14], [15] have emerged for deciding the file placement when access to multiple caches is possible in wireless network environments. We mention the extension of [16] which studies caching policies under spatio-temporal variation of file popularity. Our work generalizes femtocaching to considering cache size scaling over multiple time slots.

Cache management in cloud CDNs. There is a number of recent works studying the scenario of deploying caches as virtual functions on clouds [17], [18], [20]. For example, Presti *et al.* [17] proposed a heuristic server placement algorithm based on traffic dynamics, while Llorca *et al.* [18] proposed a distributed solution for dynamic in-network caching. To the best of our knowledge, none of these works jointly optimizes cache rental, file placement, and area-SBS association in cloud/elastic CDNs without knowledge of full environmental information.

III. System Model

Our system consists of a macro base station (MBS) s and several small base stations (SBSs) $\mathcal{J}$, which together provide coverage to a geographical area as shown in Fig. 1. SBS $j \in \mathcal{J}$ offers *storage for lease* to be used for file caching and reduce file download delay for requests emanating from the geographical area. We discretize the geographical area into $\mathcal{I}$ non-overlapping smaller areas indexed by i and denote with $\mathcal{J}_i$ all the *reachable* small cells from location i. The macro cell is considered always reachable.

The time evolves in days $k = 0, 1, \dots,$ and each day is further divided in $T = 24$ hour slots. We consider a set of files $\mathcal{F}$, and denote with $\lambda_{i,f}^k(t), t = kT, \dots, (k+1)T - 1$ the traffic demand for file f emanating from location i in slot t (belonging to day k). The traffic demand reflects the file popularity in time and space, and therefore it will be crucial to adjust caching decisions over time. It is reasonable to assume that the daily demand profile $\boldsymbol{\lambda}^k$ can be predicted accurately only near its actual occurence [21], and this is a one-day realization of a random variable $\boldsymbol{\lambda}$; hence we assume that this information is revealed to the decision maker at the beginning of the day and it is otherwise random and i.i.d. over days with mean $E[\boldsymbol{\lambda}^k]$. We emphasize here that the decision maker does not know *a priori* the value of the mean $E[\boldsymbol{\lambda}^k]$.

When a user requests a file, there is an associated download delay $d_{ij}^k, j \in \mathcal{J}_i \cup \{s\}$, which is location-specific and depends on whether the file is cached or not.[2] The delay benefit of retrieving the file from the cache of SBS j is therefore expressed as $d_{is}^k - d_{ij}^k$, and often this benefit can be significant since d_{is}^k involves contacting a remote server. SBS $j \in \mathcal{J}$ leases storage at a fluctuating price h_j^k. The price variations are justified by a spot market where the cache owner sells its left-over storage, which depends how traffic and storage demand change from day to day. From the point of view of a small content provider, it is natural to think of an available *average budget* B_{avg} (dollars per day) to be spent on cache leasing. That is, there is day-to-day elasticity on the amount of money the provider can use, but certain operational expenditure (OpEx) constraints must be met. With these economical aspects in mind, we are willing to address the content provider question of *what is the best cache leasing strategy to optimize average delay performance*. Note that this question is very challenging for the following reasons: (i) the content provider does not know the actual traffic in future days, and since larger traffic will yield more return of cache investment, deciding how much budget to invest in each day is complicated, (ii) the delay benefit of caching at a SBS changes over time, and therefore the distribution of daily budget to different caches must be carefully designed.

[2]Note that if SBS j does not cache requested file, the file is retrieved from the remote server via backhaul links and the MBS s.

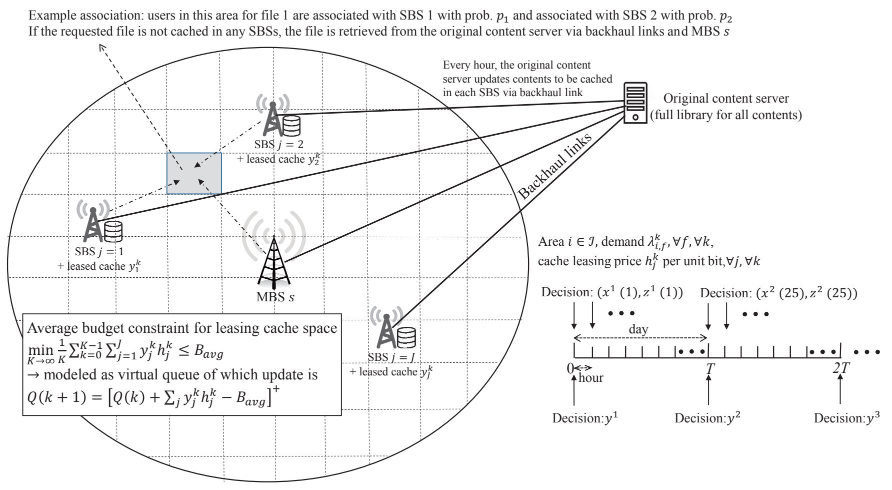

Example association: users in this area for file 1 are associated with SBS 1 with prob. p_1 and associated with SBS 2 with prob. p_2
If the requested file is not cached in any SBSs, the file is retrieved from the original content server via backhaul links and MBS s
Every hour, the original content server updates contents to be cached in each SBS via backhaul link
Original content server (full library for all contents)
SBS $j = 2$ + leased cache y_2^k
Backhaul links
SBS $j = 1$ + leased cache y_1^k
MBS s
Area $i \in \mathcal{I}$, demand $\lambda_{i,f}^k, \forall f, \forall k$,
cache leasing price h_j^k per unit bit, $\forall j, \forall k$
Decision: $(x^1(1), z^1(1))$
Decision: $(x^2(25), z^2(25))$
day
0
hour
T
$2T$
Decision: y^1
Decision: y^2
Decision: y^3
SBS $j = J$ + leased cache y_J^k
Average budget constraint for leasing cache space
$\min_{K\to\infty} \frac{1}{K}\sum_{k=0}^{K-1}\sum_{j=1}^{J} y_j^k h_j^k \le B_{avg}$
→ modeled as virtual queue of which update is
$Q(k+1) = \left[Q(k) + \sum_j y_j^k h_j^k - B_{avg}\right]^+$

Fig. 1: Overview of cache rental, file caching and region association in wireless elastic CDNs.

A. Feasible Cache Plan

We describe the decisions involved in deploying a caching plan within a day. We introduce the investment variables y_j^k to denote the amount of SBS storage that is leased for caching operations in day k. We may express the day-average budget constraint as follows:

$$\lim_{K\to\infty} \frac{1}{K}\sum_{k=0}^{K-1}\sum_{j\in\mathcal{J}} y_j^k h_j^k \le B_{avg}. \tag{1}$$

To express the delay benefit within a day, we introduce two more sets of decision variables: (i) file placement variable $z_{j,f}^k(t) \in \{0,1\}$ takes value 1 iff file f is cached at SBS j in slot t, and (ii) demand association variable $x_{ij,f}^k(t) \in [0,1]$ denotes the fraction of location i traffic demand for file f that is served by small cell j, again in slot t. We can now express the daily delay benefit obtained by cache investments in location j as

$$\Delta_{ij}^k(\boldsymbol{x}^k, \boldsymbol{z}^k; \boldsymbol{\lambda}^k) = (d_{is}^k - d_{ij}^k)\sum_{f\in\mathcal{F}}\sum_{t=kT}^{(k+1)T-1} x_{ij,f}^k(t) z_{j,f}^k(t) \lambda_{i,f}^k(t),$$

where observe that the benefit depends on the pure delay advantage that the location offers $(d_{is}^k - d_{ij}^k)$, on the fraction of traffic served at this location $x_{ij,f}^k(t)$, on whether the file is actually cached here $z_{j,f}^k(t)$, and finally on the volume of demand $\lambda_{i,f}^k(t)$. The total delay benefit in day k is:

$$g_k(\boldsymbol{x}^k, \boldsymbol{z}^k; \boldsymbol{\lambda}^k) = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}_i} \Delta_{ij}^k(\boldsymbol{x}^k, \boldsymbol{z}^k; \boldsymbol{\lambda}^k). \tag{2}$$

There is a number of constraints that must be satisfied within each day. Specifically, the entire demand of location i is served from some of the SBSs, hence it holds:

$$\sum_{j\in\mathcal{J}_i} x_{ij,f}^k(t) = 1, \quad \forall i, k, t. \tag{3}$$

Also, the total file placement should not exceed the leased cache:

$$\sum_{f\in\mathcal{F}} z_{j,f}^k(t) \le y_j^k / b, \quad \forall j, k, t, \tag{4}$$

where b is the size of each file; we assume that b is the same for all files for simplicity, but we can model a heterogeneous file size scenario by dividing different size of files into same size chunks.

Definition 1 (Feasible cache plan). *A feasible cache plan for day k is a selection of variables $(y_j^k, z_{j,f}^k(t), x_{ij,f}^k(t))$ such that $y_j^k \ge 0$, $z_{j,f}^k(t) \in \{0,1\}$, $x_{ij,f}^k(t) \in [0,1]$ and further* (3) *and* (4) *are satisfied.*

B. Problem Formulation

To tackle the average budget constraint (1) we introduce a virtual queue whose backlog is updated by

$$Q(k+1) = \left[Q(k) + \sum_{j\in\mathcal{J}} y_j^k h_j^k - B_{avg}\right]^+. \tag{5}$$

Prior work [22] shows that if the stability condition

$$\lim_{K\to\infty} \frac{1}{K}\sum_{k=0}^{K} Q(k) < \infty \tag{6}$$

is satisfied, then so is constraint (1). Intuitively, the backlog $Q(k)$ counts the excess budget spent in the previous days, which is valuable information for keeping track with the average expenditure. Hence, we formally define the state of the system at the beginning of day k as the tuple $(Q(k), \boldsymbol{\lambda}^k)$.

Definition 2 (Elastic CDN policy). *An admissible elastic CDN policy π is a (possibly randomized) mapping from the state $(Q(k), \boldsymbol{\lambda}^k)$ to a feasible cache plan. We denote with Π all the admissible policies.*

Then, our problem **(P)** is to find an elastic CDN policy that maximizes the average delay benefit, and can be formally

TABLE I: Summary of the notations

Notation	Definition
$i \in \mathcal{I}$	area index
$j \in \mathcal{J}$	small cell index
$f \in \mathcal{F}$	file index
k	day index
T	hours of a day, i.e., 24
t	index for passed hours from the beginning of the first day
$\lambda_{i,f}^k(t)$	demand profile for i, f, k and t
h_j^k	price to lease cache storage per unit bit for j and k
d_{ij}^k	average delay for serving area i by SBS j during k
d_{is}^k	average delay for serving area i by remote server during k
$x_{ij,f}^k(t)$	association probability for i, j, f, t and k
y_j^k	leased cache space at SBS j during k
$z_{j,f}^k(t)$	file caching indicator for f, j, k and t
B_{avg}	average budget constraint

stated as follows:

$$\begin{aligned} \textit{Val}(\mathbf{P}) = \quad & \sup_{\pi\in\Pi} \quad \lim_{K\to\infty} \frac{1}{K}\sum_{k=0}^{K-1} g_k, \\ & \textit{s.t.} \quad \lim_{K\to\infty} \frac{1}{K}\sum_{k=0}^{K-1}\sum_{j\in\mathcal{J}} y_j^k h_j^k \le B_{avg}. \end{aligned} \tag{7}$$

To facilitate reading, we summarize notations in Table I.

IV. Elastic Cache Scaling Algorithms

A. Upper Bound on Delay Benefit

We characterize the maximum performance by providing an upper bound on the achievable average delay benefit. Consider the problem of using a feasible cache plan to maximize the daily delay benefit given a budget $\hat{\boldsymbol{y}}$:

$$\begin{aligned} F(\hat{\boldsymbol{y}};\boldsymbol{\lambda}^k) \quad & \triangleq \max_{\boldsymbol{x}_k,\boldsymbol{z}_k} \quad g_k(\boldsymbol{x}^k,\boldsymbol{z}^k;\boldsymbol{\lambda}^k), \\ & \textit{s.t.} \quad \sum_{f\in\mathcal{F}} z_{j,f}^k(t) \le \frac{\hat{y}_j^k}{b}, \forall j,t, \\ & \qquad \sum_{j\in\mathcal{J}_i} x_{ij,f}^k(t) = 1, \forall i,f,t, \end{aligned} \tag{8}$$

Lemma 1 (Upper Bound)**.** *Let $p_{\boldsymbol{\lambda}}$ be the probability with which traffic profile $\boldsymbol{\lambda}$ occurs, and $\phi(\boldsymbol{y},\boldsymbol{\lambda})$ the probability that a stationary policy will use budget $\mathbf{y}$ when the observed traffic profile is $\boldsymbol{\lambda}$. An upper bound on the maximum value of $\boldsymbol{P}$ is:*

$$\begin{aligned} \textit{Val}(\boldsymbol{P}) \le \textit{Val}(\boldsymbol{UBound}) \doteq & \max_{\phi(\boldsymbol{y},\boldsymbol{\lambda})} \sum_{\boldsymbol{\lambda}} p_{\boldsymbol{\lambda}}\phi(\boldsymbol{y},\boldsymbol{\lambda})F(\boldsymbol{y};\boldsymbol{\lambda}), \\ & \textit{s.t.} \sum_{\boldsymbol{y},\boldsymbol{\lambda}} \phi(\boldsymbol{y},\boldsymbol{\lambda}) = 1, \\ & \quad 0 \le \phi(\boldsymbol{y},\boldsymbol{\lambda}) \le 1, \forall(\boldsymbol{y},\boldsymbol{\lambda}), \\ & \quad \sum_{\boldsymbol{\lambda}} p_{\boldsymbol{\lambda}}\phi(\boldsymbol{y},\boldsymbol{\lambda}) = B_{avg}. \end{aligned}$$

To prove the lemma, it suffices to observe the sequence of days where traffic profile $\boldsymbol{\lambda}$ occurs and show that any stationary policy will in the limit choose some ϕ from our feasible set above.[3] We note that the above bound is attainable by an oracle policy that knows the statistics $E[\boldsymbol{\lambda}^k]$. Next we provide a dynamic algorithm agnostic to $E[\boldsymbol{\lambda}^k]$ and we prove its optimality by comparing its performance to the oracle.

B. Day-by-day Algorithm

In this section, we focus on the problem of choosing the optimal investment (y_j^k) on a day-by-day basis. Focusing on day k, the decision maker is aware of the traffic profile realization $(\lambda_i^k(t))$, which in practice is typically obtained using machine learning methods, cf. [21]. Additionally, information about prices h_j^k, delays d_{is}^k, d_{ij}^k, and file size b is available. But most importantly, we assume that for any choice of investment (y_j^k), the best daily delay benefit $F(\boldsymbol{y};\boldsymbol{\lambda}^k)$ is also known. The goal is to produce a decision about the investment (y_j^k) at each cache j. To this aim we employ a *Lyapunov drift-minus-benefit* framework as follows.

To treat the budget constraint, we use the standard quadratic Lyapunov function to define the Lyapunov drift function as follows:

$$L(k) \triangleq \frac{1}{2}Q(k)^2, \tag{9}$$

$$\Delta(L(k)) \triangleq \mathbb{E}\{L(k+1) - L(k)|Q(k)\}. \tag{10}$$

Note that the Lyapunov drift depends on our day k decision (y_j^k) implicitly via the update of $Q(k+1)$, and provides information about the expected improvement of constraint satisfaction when taking the specific decision. Since we are also interested in maximizing the time average of g_k using feasible cache plans, we next introduce the Lyapunov drift-minus-benefit function (*DMB*):

$$\textit{DMB}(\boldsymbol{y};\boldsymbol{\lambda}^k) = \Delta(L(k)) - V\mathbb{E}\{F(\boldsymbol{y};\boldsymbol{\lambda}^k)|Q(k)\}, \tag{11}$$

where V is a constant parameter to balance the tradeoff between two conflicting objectives, (i) improving the budget constraint satisfaction, and (ii) greedily maximizing the daily benefit. We will see that V can be tuned to also tradeoff convergence with accuracy of the whole approach.

Applying the queue update equation (5) and lemma 4.3 from [24], we obtain under any possible decision $(y_j^k), \forall j \in \mathcal{J}$:

$$\begin{aligned} \textit{DMB}(\boldsymbol{y};\boldsymbol{\lambda}^k) \le & P - V\mathbb{E}\{F(\boldsymbol{y};\boldsymbol{\lambda}^k)|Q(k)\} \\ & - \mathbb{E}\Big\{\Big(B_{avg} - \sum_{j\in\mathcal{J}} y_j^k h_j^k\Big)Q(k)|Q(k)\Big\}, \end{aligned} \tag{12}$$

where $P = 1/2(B_{avg}^2 + |\mathcal{J}|y_{max}^2 h_{max}^2)$ is a positive constant. Prior work [25] shows that we can uncover optimal decisions by minimizing the RHS of (12).

Define the day-by-day DMB (*DBDD*) policy as the one that chooses $\boldsymbol{y}^k = \boldsymbol{y}^*$ as follows:

$$\boldsymbol{y}^* \in \arg\max_{\boldsymbol{y}} VF(\boldsymbol{y};\boldsymbol{\lambda}^k) - \sum_{j\in\mathcal{J}} Q(k)y_j^k h_j^k, \tag{13}$$

[3]Our result is given here for stationary policies whose decisions converge in the limit. It is possible to include non-stationary policies by replacing limits with $\liminf$, however such policies do not provide a better performance hence we leave them out of our considerations. Also, we have assumed a finite set of possible budget decisions and traffic demand profiles, which is an important technical assumption; to alleviate it we can use approximations used in [23].

and additional a feasible cache plan $\boldsymbol{x}^*, \boldsymbol{z}^*$ in $F(\boldsymbol{y}^*; \boldsymbol{\lambda}^k)$. We have the following results:

Theorem 1 (Optimality tradeoffs)**.** *The DBDD stabilizes $Q(k)$, and hence satisfies* (1)*, it achieves an average delay benefit: Val(**DBDD**) $\geq$ Val(**UBound**) $-$ $O(1/V)$, and the average queue length satisfies: $Q^{\textbf{DBDD}} \leq O(V)$.*

Proof. Due to the limited space, the proof is presented in our technical report [26]. □

Some remarks are in order:

- Our scheme always satisfies the budget constraint, as long as $\boldsymbol{\lambda}^k$ has finite second moment.
- It achieves a near-optimal average delay benefit without *a priori* knowledge of stationary popularity statistics, but rather by looking at the daily learned popularity and keeping a budget counter.
- If $\boldsymbol{\lambda}^k$ is stationary, we may tolerate a large value for $Q(k)$, and hence we can pick a large V. However, keeping the queue length small has the benefit of making the algorithm robust to time-varying traffic profile statistics.

The DBDD relies on a method to compute optimal cache plans that maximize average daily delay benefit. Below we focus on this aspect.

V. Intra-day Problem and Algorithms

In this section we turn our attention into solving the intra-day problem of finding the average delay benefit $F(\boldsymbol{y}; \boldsymbol{\lambda}^k)$ given budget investment $\boldsymbol{y}$.

A. Non-overlapping SBS Coverage

We study first the case where the small cells are non-overlapping. This immediately simplifies the routing decisions $x_{ij,f}^k(t)$, such that $x_{ij,f}^k(t) = 1$, $\forall t, f, k$ if location i is connected to SBS j and 0 otherwise. In essence this removes one of the constraints from the problem. We will see that this makes our problem relatively easy to solve.

Plugging (8) into (13) and omitting (3), we see that an optimal cache plan can be found by solving for fixed k:

$$\max_{\substack{y_j^k \geq 0 \\ z_{j,f}^k(t) \in \{0,1\}}} V \sum_{t=kT}^{(k+1)T-1} \sum_{j,f} D_{j,f}^k(t) z_{j,f}^k(t) - Q(k) \sum_{j \in \mathcal{J}} y_j^k h_j^k, \tag{14}$$

$$\textit{s.t.} \sum_{f \in \mathcal{F}} z_{j,f}^k(t) \leq \frac{y_j}{b}, \ \forall t \in \{kT, \ldots, (k+1)T-1\}, j.$$

where caching file f at location j in slot t brings the following delay improvement:

$$D_{j,f}^k(t) \triangleq \sum_i (d_{is}^k - d_{ij}^k) x_{ij,f}^k(t) \lambda_{i,f}^k(t),$$

which is computable using known parameters $\boldsymbol{d}, \boldsymbol{x}, \boldsymbol{\lambda}$. Although (14) is an Mixed Integer Linear Program (MILP), due to its simple form it can be solved by inspection. At each pair of location-slot (j,t) we order files in decreasing values of $D_{j,f}^k(t)$. We remark that if we make the investment y_j^k the best delay benefit will be harvested by caching the y_j^k/b files that rank higher in this list. This provides directly the solutions $\boldsymbol{z}$ as a function of $\boldsymbol{y}$, it remains now to determine the latter.

With a slight abuse of notation, let us call σ the permutation of file indices that implies $D_{j,\sigma(1)}^k(t) \geq \cdots \geq D_{j,\sigma(|\mathcal{F}|)}^k(t)$ (the abuse is because we do not explictly denote the dependence of σ on j,t to reduce clutter), then we can decompose the investment decisions per location, and find y_j^k that maximizes:

$$y_j^{k^*} \in \arg\max_{y_j^k \geq 0} \sum_{t=kT}^{(k+1)T-1} \sum_{f=1}^{\lfloor y_j^k/b \rfloor} D_{j,\sigma(f)}^k(t) - \frac{Q(k)}{V} h_j^k y_j^k.$$

Above, $y_j^{k^*}$ can be efficiently computed by listing partial sums $\sum_t \sum_{f=1}^{\lfloor y_j^k/b \rfloor} D_{j,\sigma(f)}^k(t)$ for $y_j^k/b = 1, 2, \ldots$ until the difference of one partial sum from the previous becomes smaller than $\frac{Q(k)}{V} h_j^k$.

Mathematically speaking, the above might include cases where the solution is to avoid investment alltogether ($y_j^k = 0$), or buy storage for all files ($y_j^k = |\mathcal{F}|$), however in practice these cases are extremely rare, because of the skewness of popularity: we will always benefit from storing popular files and we will seldom benefit from storing unpopular ones. Another remark is that this algorithm can be generalized in a straightforward manner to the case where the cache leasing is made on hour basis (i.e. when $\boldsymbol{y}$ change every slot). Below we give the algorithmic steps to find $\boldsymbol{y}$ and $\boldsymbol{z}$ in detail.

Joint cache rental and file caching algorithm for given area-SBS association (JCC) algorithm

Result: $y_j^k, z_{j,f}^k(t), \forall j, k, f, t$
In day k, read values $Q(k), x_{ij,f}^k(t), \lambda_{i,f}^k(t), d_{ij}^k$, d_{is}^k, h_j^k, $\forall i, j, f, t$

```
1:  For all SBSs $j$,
2:     For $t = kT : (k+1)T - 1$,
3:        For all files $f$,
4:           Calculate $D_{j,f}^k(t) = \sum_i (d_{is}^k - d_{ij}^k) x_{ij,f}^k(t) \lambda_{i,f}^k(t)$
5:        End For
6:     End For
7:     Sort $D_{j,f}^k(t)$ such that for permutation $\sigma()$ we have
       $D_{j,\sigma(1)}^k(t) \geq \cdots \geq D_{j,\sigma(|\mathcal{F}|)}^k(t)$
8:     Set partial sums $S(e) = \sum_t \sum_{f=1}^{e} D_{j,\sigma(f)}^k(t)$,
9:     Find $e^*$ which maximizes $V S(e) - Q(k) h_j^k b e$
10:    Choose cache lease: $y_j^k = e^* b$
11:    For $t = kT : (k+1)T - 1$,
12:       Choose file placement:
             $z_{j,\sigma(f)}^k(t) = \begin{cases} 1 & \text{if } f \leq \lfloor y_j^k/b \rfloor \\ 0 & \text{otherwise} \end{cases}$
15:    End For
16: End For
```

B. General Case with Overlapping SBS Coverage

Next, we consider the general case, where the areas that SBS cover may overlap. In this case, area-SBS association variables $x_{ij,f}^k(t)$ must be jointly decided with cache rental and file placement. The daily problem becomes:

$$\max_{\substack{y_j^k \geq 0 \\ z_{j,f}^k(t) \in \{0,1\}}} \quad V g_k(\boldsymbol{x}^k, \boldsymbol{z}^k; \boldsymbol{\lambda}^k) - Q(k) \sum_{j \in \mathcal{J}} y_j^k h_j^k, \tag{15}$$

$$\textit{s.t.} \sum_{f \in \mathcal{F}} z_{j,f}^k(t) \leq \frac{y_j}{b}, \ \forall t \in \{kT, \ldots, (k+1)T-1\}, j.$$

$$\sum_{j \in \mathcal{J}_i} x_{ij,f}^k(t) = 1, \forall i, f, t.$$

We note that (15) is a mixed integer *non-linear* program (MINLP) due to the product of variables $x_{ij,f}^k(t), z_{j,f}^k(t)$ that appears in the objective inside g_k. To proceed, we linearize the objective by removing from it the variables $z_{j,f}^k(t)$ and instead introducing an extra constraint $x_{ij,f}^k(t) \leq z_{j,f}^k(t)$. Note that if $z_{j,f}^k(t) = 0$ then the constraint implies that $x_{ij,f}^k(t) = 0$ as well, eliminating any delay benefit at the objective. Also if $z_{j,f}^k(t) = 1$, variable $x_{ij,f}^k(t)$ is not affected by the new constraint. To deal with the case where files are not cached, We also add a *dummy* allocation variable $x_{is,f}^k(t)$ to ensure that the constraint (3) is satisfied when file f is not cached anywhere.

Having obtained a Mixed Integer Linear Program (MILP), in the remaining of the paper we relax the integral placement variables $z_{j,f}^k(t)$ to take values in $[0,1]$, and we focus on the continuous relaxation of our problem. This is useful because:

- Combining with randomized rounding [27], it is possible to obtain approximation guarantees; this is left as future work.
- The continuous relaxation also gives an explicit upper bound on the maximum average delay benefit that can be achieved. In practice, such bounds are extremely useful to decide when to stop an iterative search for solutions.
- Last but not least, as explained in [7] it is possible to use MDS codes to achieve an effective "fractional file placement". In essence, each cache stores a number of linear combinations of file chunks which correspond to fractions of a file, and then each user can combine different such coded chunks to produce the original file. In this context, our problem becomes a Linear Program.

Henceforth, we consider the following **(LP)**:

$$\textbf{(LP):} \quad \max_{\boldsymbol{x}^k, \boldsymbol{y}^k, \boldsymbol{z}^k} \quad \Big\{ V \sum_j \sum_i (d_{is}^k - d_{ij}^k) \sum_f \sum_t x_{ij,f}^k(t) \lambda_{i,f}^k(t) - Q(k) \sum_j \sum_f h_j^k y_j^k \Big\}, \tag{16}$$

$$\textit{s.t.} \quad x_{ij,f}^k(t) \leq z_{j,f}^k(t), \ \forall i, j, f, t, \tag{17}$$
$$(3), (4),$$

We relax constraint (17) and introduce corresponding Lagrangian multipliers:

$$\mu_{ij,f}^k(t) \geq 0, \ \forall i, j, f, t. \tag{18}$$

Joint cache rental, file caching and area-SBS association (JCCA) algorithm

```
Result: $x_{ij,f}^k(t), y_j^k, z_{j,f}^k(t), \forall j, k, f, t$
At the beginning of day $k$, update $Q(k)$
Initialization: $\tau = 1$, $\mu_{ij,f}^k(t)[1] = 0$, $\forall i, j, f, t$, $UB = \infty$, $LB = -\infty$, $\epsilon = 0.1$ and $\tau_{max} = 1000$
1: While ($|\frac{UB-LB}{LB}| \geq \epsilon$ and $\tau \leq \tau_{max}$)
2:     Solve (P1) to find solutions $x_{ij,f}^k(t)[\tau], \forall i, j, t$
3:     Solve (P2) to find solutions $z_{j,f}^k(t)[\tau], \forall j, f, t$
5:     Update $LB$ based on the found solutions
6:     Update $UB = L(\boldsymbol{\lambda}, \boldsymbol{x}, \boldsymbol{z})$, $\theta[\tau] = \gamma \frac{UB-LB}{||g[\tau]||^2}$
7:     Update the dual variables $\mu_{ij,f}^k(t)[\tau+1]$
       using (22)
8:     Update $\tau = \tau + 1$
9: End While
```

Then, we formulate the partial Lagrangian function

$$L(\boldsymbol{\mu}^k, \boldsymbol{x}^k, \boldsymbol{y}^k, \boldsymbol{z}^k) = V \sum_{i,j,f,t} (d_{is}^k - d_{ij}^k) \lambda_{i,f}^k(t) x_{ij,f}^k(t) - \sum_{i,j,f,t} \mu_{ij,f}^k(t) (x_{ij,f}^k(t) - z_{j,f}^k(t)) - Q(k) \sum_j \sum_f h_j^k y_j^k, \tag{19}$$

and the dual problem

$$\min_{\boldsymbol{\mu}^k \geq 0} \max_{\boldsymbol{x}^k, \boldsymbol{y}^k, \boldsymbol{z}^k} \quad L(\boldsymbol{\mu}^k, \boldsymbol{x}^k, \boldsymbol{y}^k, \boldsymbol{z}^k), \tag{20}$$

$$\textit{s.t.} \quad (3), (4). \tag{21}$$

This problem can be solved in an iterative manner, using the dual decomposition technique [28]. Technically, the dual variables $\boldsymbol{\mu}^k$ are updated at each iteration τ, and the primal problem is solved according to multipliers $\boldsymbol{\mu}^k[\tau]$. Then, the primal solutions $\boldsymbol{x}^k, \boldsymbol{z}^k$ are used to produce $\boldsymbol{\mu}^k[\tau+1]$. This algorithm is summarized in JCCA algorithm.

The dual updates. In detail, since our Lagrangian $L(\boldsymbol{\mu}^k, \boldsymbol{x}^k, \boldsymbol{y}^k, \boldsymbol{z}^k)$ is not differentiable everywhere with respect to $\boldsymbol{\mu}^k$, we will employ a subgradient method as in [29]. Every iteration τ, dual variables $\mu_{ij,f}^k(t)$ for all i, j, f and t are updated as follows:

$$\mu_{ij,f}^k(t)[\tau+1] = \left(\mu_{ij,f}^k(t)[\tau] + \theta[\tau] \nabla \mu_{ij,f}^k(t)[\tau]\right)^+, \tag{22}$$

where $\left[\cdot\right]^+ = \max(0, \cdot)$, $\theta[\tau]$ denotes the step size at iteration τ, and $\nabla \mu_{ij,f}^k(t)[\tau]$ is the (i, j, f, t) coordinate of a subgradient vector at $\boldsymbol{\mu}^k[\tau]$. A subgradient vector can be found as usually by the actual value of the relaxed constrained–we omit proof for brevity:

$$\nabla(\mu_{ij,f}^k(t)[\tau]) = x_{ij,f}^k(t)[\tau] - z_{j,f}^k(t)[\tau], \forall i, j, f, t. \tag{23}$$

The primal problem. Every iteration τ, the primal problem is solved using dual variables $\boldsymbol{\mu}^k[\tau]$. Thanks to the chosen relaxation, the primal problem can be decomposed into two sub-problems to determine $\boldsymbol{x}^k$ and $\boldsymbol{z}^k$[4]:

$$\textbf{(P1):} \quad \max_{\boldsymbol{x}^k} \sum_{i,j,f,t} [V(d_{is}^k - d_{ij}^k) \lambda_{i,f}^k(t) - \mu_{ij,f}^k(t)] x_{ij,f}^k(t), \quad \textit{s.t.} \ (3),$$

[4]Note that $\boldsymbol{y}^k$ can be directly obtained from $\boldsymbol{z}^k$.

$$\textbf{(P2):} \max_{\boldsymbol{y}^k,\boldsymbol{z}^k} \sum_{i,j,f,t} \mu^k_{ij,f}(t) z^k_{j,f}(t) - Q(k) \sum_{j,f} h^k_j y^k_j, \ \ s.t.\ (4).$$

For **(P1)**, observe that the objective function is a linear function of $x^k_{ij,f}(t)$ so for each tuple (k, i, f, t) it suffices to pick the SBS j with the highest $V(d^k_{is} - d^k_{ij})\lambda^k_{i,f}(t) - \mu^k_{ij,f}(t)$ among all SBSs for all i and t.

We can solve **(P2)** similarly to (14) using the JCC algorithm except for replacing $D^k_{j,f}(t)$ with $U^k_{j,f}(t) = \sum_i \mu^k_{ij,f}(t)$. With our decomposition we have achieved that in **(P2)** the decisions are not coupled with area-SBS association $\boldsymbol{x}^k$, and the problem is separable to each SBS. Due to the limited space, please refer to our technical report [26] for detailed algorithm to solve **(P2)**.

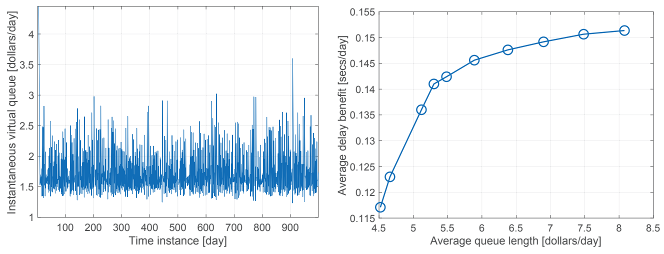
Our *JCCA* algorithm belongs to the class of dual subgradient methods, which are using a non-summable but square summable step size $\theta[\tau]$ can be shown to converge asymptotically to the optimal Lagrangian multipliers [29], i.e. we may show that $\boldsymbol{\mu}^{(\tau)} \rightarrow \boldsymbol{\mu}^*$. Since our relaxed constraint is not smooth and as a result the partial Lagrangian is not differentiable everywhere, it follows that the Lagrangian multipliers that solve the dual are not unique, and hence the found dual variables do not immediately imply a feasible (and hence optimal) primal solution. However, if we look at the cumulative average of primal iterates, we can obtain such an optimal primal solution [30]. Therefore, to obtain the final solutions we use the averages: $\overline{x}^k_{ij,f}(t)[W] \leftarrow \frac{1}{W}\sum_{\tau=1}^{W} x^k_{ij,f}(t)[\tau]$, $\overline{y}^k_j(t)[W] \leftarrow \frac{1}{W}\sum_{\tau=1}^{W} y^k_j(t)[\tau]$ and $\overline{z}^k_{j,f}(t)[W] \leftarrow \frac{1}{W}\sum_{\tau=1}^{W} z^k_{j,f}(t)[\tau]$. *JCCA* algorithm finally produces as the optimal primal solution $(\overline{x}^k_{ij,f}(t)[W], \overline{y}^k_j(t)[W], \overline{z}^k_{j,f}(t)[W])$ for some large W.

VI. Performance Evaluation

In this section, we execute simulations to demonstrate the performance of the proposed elastic cache scaling algorithms. **Simulation setup.** We consider a plane divided in 100 areas where the size of each area is 20m×20m, and 4 SBSs and 1 MBS are randomly deployed. Transmission power of a SBS is 30dBm whereas that of a MBS is 43dBm. The path loss is set to be 128.1 + 37.6$\log_{10}(d)$ where d is distance of the BS from the center of each area, the system bandwidth is set to be 10MHz. Average wireless delay from the MBS or SBSs is calculated as the size of a file (50Mbytes) divided by the average data rate for each area calculated by Shannon capacity formula. Moreover, since the file retrieved from MBS requires backhaul transmission, we assume that additional backhaul delay, which is an average value from a dataset in [31], is given. Demand shape for all hours in each day follows average traffic demand from real trace [32], and file popularity over days follows a normal distribution where mean values follow Zipf distribution[5] and variance is the mean value divided by $\sqrt{12}$. Average cache rental budget is set to be a budget to lease 10% of total file sizes, and price to lease unit cache is set to be 3.609×10^{-4} dollars per bit per day [5].

We compare five algorithms including the proposed JCCA and JCC algorithms. In the temporal cache budget sharing (TCS) algorithm, there is a dedicated average cache rental budget for each SBS, i.e., B_{avg}/J, hence each SBS manages their own virtual queue independently. In the spatial cache budget sharing (SCS) algorithm, cache rental budget is dedicated to each SBS by considering average content popularity for coverage of each SBS, and then existing content caching and user association algorithm [33] is used. In the fixed cache budget (FCB) algorithm, which is similar approach in [33], all SBSs always have the fixed cache budget, i.e., B_{avg}/J.

[5]We generate the popularity ranking randomly in each area.



(a) Virtual queue stability. (b) Benefit-queue tradeoff.

Fig. 2: Operation of the JCCA algorithm.

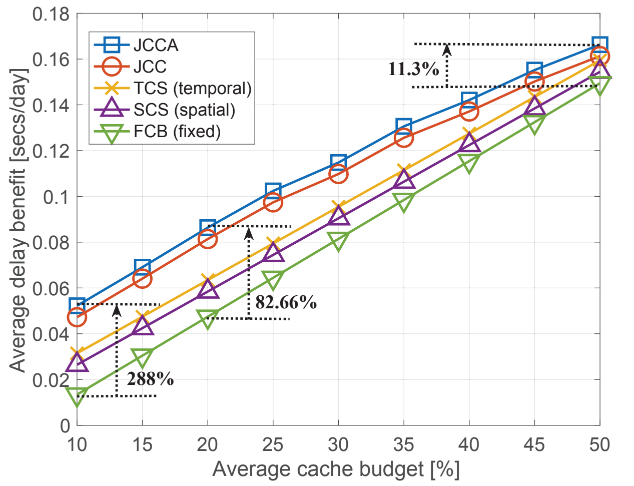


Fig. 3: Average cache budget and average delay benefit tradeoff.

Simulation results. We present our results by summarizing the key observations as follows.

Operation of the JCCA algorithm. Fig. 2(a) shows statistics of the virtual queue for all days of the proposed JCCA algorithm. First, the stability of the queue demonstrates that JCCA satisfies the budget constraint. Second, the variation of the queue reveals that JCCA exploits opportunism of environmental variations such as data rates and content popularity, e.g., using more cache budget when wireless channel states are good, and vice versa. Moreover, Fig. 2(b) depicts a tradeoff between average delay benefit and average virtual queue for different parameters V. Note that keeping the average virtual queue length small makes the robustness to changes in the statistics of the traffic profile. As the average virtual queue increases, the average delay benefit becomes higher since the degree of freedom to exploit the time-averaged cache rental budget gets higher. This tradeoff curve directly verifies the result of Theorem 1.

Cache budget and delay benefit tradeoff. Fig. 3 depicts tradeoff curves between average delay benefit and average cache budget for all algorithms. First, the average delay benefit of the JCCA algorithm is 82.66% higher than fixed cache budget algorithm (FCB) when the average cache budget is 20% of entire file library. Second, this result implies that the impact of the spatial and temporal cache budget sharing, applied in the JCCA and JCC algorithms is critical in terms of average delay benefit. Moreover, the difference of delay benefit between the dynamic JCCA (or JCC) algorithm and static cache storage-based algorithms, i.e., TCS, SCS and FCB becomes higher as the average cache budget becomes tighter. It is due to the fact that the effect of exploiting traffic profile among different areas and time slots would be more important in tight cache rental budget, similar to the philosophy of standard water-filling algorithms.[6] This is desirable result in a scenario that small market CPs lease the edge caches in cellular networks.

VII. Conclusion

As a way to fully exploit average cache rental budget for small-market content providers, we propose a joint cache rental, file caching and region association, namely JCCA algorithm in wireless elastic CDNs. The proposed policy makes an effort to maximize average delay benefit of a service provider while ensuring long-term cache rental budget under varying file popularity and wireless channel states over time and space. Simulation results reveal that the proposed dynamic JCCA algorithm would be more important when the average cache budget is very limited, which is one of common scenarios for small-market content providers.

Acknowledgement

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 713567. The authors also acknowledge the support from Science Foundation Ireland (SFI) under Grant Number 17/CDA/4760. Moreover, the ideas and opinions expressed in this paper are of the authors, and do not represent the official position of Huawei Technologies.

References

[1] "The elastic CDN solution (akamai-juniper)." [Online]. Available: https://www.juniper.net/assets/kr/kr/local/pdf/solutionbriefs/3510532-en.pdf

[2] Akamai White Paper, "The case for a virtualized CDN(vCDN) for delivering operator OTT video." [Online]. Available: https://www.akamai.com/cn/zh/multimedia/documents/white-paper/the-case-for-a-virtualized-cdn-vcdn-for-delivering-operator-ott-video.pdf

[3] "Akamai collaborates with orange on NFV initiative to dynamically scale CDN capacity for large events." [Online]. Available: https://www.akamai.com/us/en/about/news/press/2016-press/akamai-collaborates-with-orange-on-nfv-initiative.jsp

[4] Huawei, "Huawei uCDN solution." [Online]. Available: http://carrier.huawei.com/en/solutions/cloud-powered-digital-services/ucdn

[5] "Amazon elastic CDN service - ElastiCache." [Online]. Available: https://aws.amazon.com/elasticache/

[6] AT&T, "AT&T business, content delivery network." [Online]. Available: https://www.business.att.com/solutions/Family/cloud/content-delivery-network/

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

[8] J. Kwak, Y. Kim, L. Le, and S. Chong, "Hybrid content caching in 5G wireless networks: Cloud versus edge caching," *to appear, IEEE Trans. on Wireless Commun.*, pp. 1–17, Feb. 2018.

[9] J. Kwak, L. Le, and X. Wang, "Two time-scale content caching and user association in 5G heterogeneous networks," in *Proc. of IEEE GLOBECOM*, Dec. 2017, pp. 1–6.

[10] T. Bektas, O. Oguz, and I. Ouveysi, "Designing cost-effective content distribution networks," *Computers & Operations Research*, vol. 34, no. 8, pp. 2436–2449, Aug. 2007.

[11] K. Ho, S. Georgoulas, M. Amin, and G. Pavlou, "Managing traffic demand uncertainty in replica server placement with robust optimization," *NETWORKING 2006. Networking Technologies, Services, and Protocols*, pp. 727–739, 2006.

[12] W. Li, E. Chan, Y. Wang, D. Chen, and S. Lu, "Cache placement optimization in hierarchical networks: Analysis and performance evaluation," *NETWORKING 2008. Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet*, pp. 385–396, 2008.

[13] N. Laoutaris, V. Zissimopoulos, and I. Stavrakakis, "On the optimization of storage capacity allocation for content distribution," *Computer Networks*, vol. 47, no. 3, pp. 409–428, 2005.

[14] Z. Zhao, M. Peng, Z. Ding, W. Wang, and H. Poor, "Cluster content caching: An energy-efficient approach to improve quality of service in cloud radio access networks," *IEEE JSAC*, vol. 34, no. 5, pp. 1207–1221, May 2016.

[15] G. F. W. Jiang and S. Qin, "Optimal cooperative content caching and delivery policy for heterogeneous cellular networks," *IEEE Trans. on Mobile Comput.*, vol. 16, no. 5, pp. 1382–1393, May 2017.

[16] M. Chen, W. Saad, C. Yin, and M. Debbah, "Echo state networks for proactive caching in cloud-based radio access networks with mobile users," *IEEE Trans. on Wireless Commun.*, vol. 16, no. 6, pp. 3250–3535, Jun. 2017.

[17] F. Presti, C. Petrioli, and C. Vicari, "Distributed dynamic replica placement and request redirection in content delivery networks," in *Proc. of IEEE MOACOTS*, 2007, pp. 366–373.

[18] J. Llorca, A. Tulino, K. Guan, J. Esteban, M. Varvello, N. Choi, and D. Kilper, "Dynamic in-network caching for energy efficient content delivery," in *Proc. of IEEE INFOCOM*, 2013, pp. 245–249.

[19] G. Dan and N. Carlsson, "Dynamic content allocation for cloud-assisted service of periodic workloads," in *Proc. of IEEE INFOCOM*, 2014, pp. 853–861.

[20] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5G wireless networks," *IEEE Commun. Magazine*, vol. 52, no. 8, pp. 82–89, Feb. 2014.

[21] M. Neely, "Energy optimal control for time varying wireless networks," *IEEE Trans. on Inform. Theory*, vol. 52, no. 7, pp. 2915–2934, Jul. 2006.

[22] S. Paris, A. Destounis, L. Maggi, G. S. Paschos, and J. Leguay, "Controlling flow reconfigurations in SDN," in *Proc. of IEEE INFOCOM*, 2016, pp. 1–9.

[23] L. Georgiadis, M. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Foundation and Trends in Networking*, vol. 1, no. 1, pp. 1–149, 2006.

[24] M. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, pp. 1–211, 2010.

[25] J. Kwak, G. Paschos, and G. Iosifidis, "Dynamic cache rental and content caching in elastic wireless CDNs," *Technical Report*, Jan. 2018. [Online]. Available: https://www.dropbox.com/s/mpkj5rqlhiznua2/Kwak18elasticCDN.pdf?dl=0

[26] P. Raghavan and C. D. Tompson, "Randomized rounding: a technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, no. 4, pp. 365–374, 1987.

[27] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[28] S. Boyd and A. Mutapcic, "Subgradient methods," *Lecture notes of EE364b, Stanford University, Winter Quarter*, 2006.

[29] A. Nedić and A. Ozdaglar, "Approximate primal solutions and rate analysis for dual subgradient methods," *SIAM Journal on Optimization*, vol. 19, no. 4, pp. 1757–1780, 2009.

[30] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of YouTube network traffic at a campus network - Measurements, models, and implications," *Elsevier Computer Networks*, vol. 53, no. 4, pp. 501–514, Mar. 2009.

[31] E. Oh, B. Krishnamachari, X. Liu, and Z. Niu, "Towards dynamic energy-efficient operation of cellular network infrastructure," *IEEE Commun. Magazine*, vol. 49, no. 6, pp. 56–61, Jun. 2011.

[32] Y. Wang, X. Tao, X. Zhang, and G. Mao, "Joint caching placement and user association for minimizing user download delay," *IEEE Access*, vol. 4, pp. 8625–8633, 2016.

[6]When the overall power available is less, the effect of exploiting frequency selectivity across subcarriers would be greater.