2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)

Joint Online Edge Caching and Load Balancing for Mobile Data Offloading in 5G Networks

Yiming Zeng*, Yaodong Huang*, Zhenhua Liu†, Yuanyuan Yang*
*Department of Electrical and Computer Engineering
†Department of Applied Mathematics and Statistics
Stony Brook University, Stony Brook, NY 11794, USA
{yiming.zeng, yaodong.huang, zhenhua.liu, yuanyuan.yang}@stonybook.edu

***Abstract*—This paper considers how to cache popular contents and load balancing in 5G networks to minimize the total operating cost. Specifically, popular contents requested by mobile users (MUs) are cached in small base stations (SBSs) to serve them with better quality and lower cost because the SBSs are often much closer to MUs than the base station (BS). Due to limited caching capacity and bandwidth of SBSs, the caching policy and load balancing algorithm need to be carefully designed jointly and dynamically over time. In this paper, we formulate the joint content placement and load balancing by an online optimization problem. This problem is challenging because of the integer constraint in content placement and the lack of future information. We tackle the challenges in two progressive steps. First, we propose a primal-dual algorithm to solve the problem efficiently and prove it always achieves the optimal cost assuming all system information is available. Then we integrate promising online optimization algorithms with the proposed primal-dual algorithm so that only limited short-term predictions are needed. Theoretical performance bounds are also derived. We conduct extensive numerical simulations to evaluate the performance of proposed algorithms. Results highlight that the proposed online algorithms can reduce the system cost significantly (by as much as 27%) compared to the existing solutions and perform similarly to the offline optimal solution.**

I. Introduction

The explosive growth of smart edge devices such as smart phones and tablets has greatly enriched the mobile user (MU) experience by expanding available services such as social media applications and live video streaming [1]. This improvement comes at the cost of an exponential growth of data generated in communication networks. In particular, the global mobile data traffic is predicted to increase 7x times from 2016 to 2021 [2] and threatens to drain the capacity of cellular networks. As a result, wireless communication networks are required to increase their capacity at a similar pace, as well as to meet the stringent requirements of latency-sensitive applications of MUs [3].

To increase the capacity of the cellular networks, the 5G wireless network is designed with larger bandwidth, larger scale of antennas, higher frequency reuse with network densification, etc. [4]. Besides the engineering efforts, an increasing number of small base stations (SBSs), e.g., microcell, picocell and femtocell base stations [5], are being deployed for better service quality with lower costs. These SBSs are connected with the core base station (BS) with backhual links. Instead of always communicating with the BS, MUs are able to utilize nearby SBSs. The transmission cost between a user and a neighbouring SBS is much lower than that from the BS because of the lower energy consumption resulted from the shorter distance and spectrum reuse. However, the effectiveness of this approach heavily relies on the high speed backhaul links between the BS and every single SBS. The capacity of these links must exceed the aggregated data requests rate of all the users served, which is not practical due to the high structure costs.

Edge caching is proposed in 5G for better quality of service. SBSs in 5G networks are equipped with mobile computing servers, which provide both memory for caching and computing devices for intelligent decision making [6]. Due to the ever decreasing cost of memory devices, it is realistic to equip each SBS a limited but significant amount of memory space that enables edge caching. The temporal variability of network traffic provides the opportunity to perform caching updates during the periods with low traffic, which reduces peak traffic demands and the transmissions delay [7]. The computing devices enable the SBS to carefully balance load among competing user requests.

Many previous studies have been done on edge caching in the cellular network (including 5G network) [8]–[11]. Note that the content popularity or the MUs' request patterns are often modeled as Zipf distributions. The caching policy is modeled as integer programming in [8], [9]. The online caching problems are considered in [12]–[18]. Among them, [13]–[15] implement data analysis methodologies driven by real data for cache replacement algorithms. The time-correlated adjustment cost such as the system replacement cost is not considered in the most studies mentioned above. However, it has been widely considered in many domains, e.g., edge caching in Cloud Radio Access Networks (C-RAN) [12] and load balancing [19], [20].

In this paper, we aim to answer the following question:

How should each SBS cache contents (caching policy) and each MU be served from the SBS or the BS for each request (load balancing) in order to minimize the system cost?

The key challenges are:

- The caching policy and load balancing are interdependent and need to be jointly optimized. The caching policy depends on the load balancing decisions of MUs to decide which contents to be (re)placed. MUs need to know whether the requested contents are cached by each SBS

2575-8411/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDCS.2019.00096

to distribute their requests across the SBSs.

- The edge caching involves integer programming, which is NP-hard in many specific cases. Therefore, we need a computationally efficient solution instead of the brute-force method.
- The decisions are over a possibly long time horizon, during which the popularity of contents, user demand, and other factors may change. This makes it an online optimization problem, and it is challenging to solve online optimization with integer constraints.

By addressing these challenges, we make the following contributions.

- We formulate an online optimization problem for the joint caching policy and load balancing for the mobile data offloading in 5G networks.
- We propose an algorithm to solve the offline problem by separating the original problem into two sub-problems based on dual decomposition. The first sub-problem is a standard convex optimization problem which can be solved directly. We relax the integer variables in the other sub-problem to continuous ones and prove this relaxation is exact, i.e., the optimal integer solution is guaranteed.
- To solve the problem in an online manner with limited predictions, we incorporate three promising online algorithms, namely, the Receding Horizon Control (RHC) [19], Averaging Fixed Horizon Control (AFHC) [19] and the Committed Horizon Control (CHC) [21]. We prove the theoretical bound. Details of these algorithms are presented in Section IV.
- Extensive simulations are conducted to validate the performance of our online algorithms. Results highlight that the proposed algorithms reduce the system cost significantly (by at most 27%) compared to the existing scheme.

The rest of paper is organized as follows. In Section II, we describe the system model and formulate the joint caching and load balancing problem. Section III provides a solution based on the primal-dual method to solve the problem. The online algorithms are designed in Section IV. Section V provides our numerical results. Some related work is introduced in Section VI, and we conclude our work in Section VII.

II. System Model and Problem Formulation

We consider a discrete-time model whose timeslot matches the timescale at which caching policies and load balancing decisions can be updated. There is a (possible long) interval of interest $t \in \{1, 2, 3, ..., T\}$. In reality, T can be a year or a time slot every 10 minutes. Important notations are summarized in Table I.

A. Modeling the 5G network:

Stations: As depicted in Fig.1, we consider a down-link edge caching wireless 5G network with one base station (BS) and N small base stations (SBSs), indexed by $\mathcal{N} = \{1, 2, 3, ..., N\}$. Each SBS covers the area with the transmission range of a few tens of meters. We assume that SBSs do not interfere with the BS. For the presentation simplicity, we assume the regions covered by different SBSs are disjoint. This assumption accords with the real 5G cellular network topology structure as it reduces the implementations cost and enables spectrum reuse by different SBSs. Our models and algorithms can be readily extended to SBSs with overlaps in coverage.

TABLE I
Table of Notations Used In Problem Formulation

Notation	Definition
$\mathcal{N}$	Set of SBS $\mathcal{N} = \{1, 2, \ldots, N\}$
$\mathcal{K}$	Set of files $\mathcal{K} = \{1, 2, \ldots, K\}$
$\mathcal{M}_n$	Set of classes of MUs $\mathcal{M}_n = \{m_n : n \in \mathcal{N}\}$
$\mathcal{T}$	Set of timeslots $\mathcal{T} = \{1, 2, \ldots, T\}$
Λ^t	MUs requests matrix
$\lambda^t_{m_n,k}$	Demand of m_n for content k at time slot t
X^t	Set of cahching variables $X^t = (x^t_{n,k})_{n\in\mathcal{N},k\in\{1,2,3,\ldots,K\}}$
Y^t	Set of load balancing variables of SBS $(y^t_{m_n,k})_{m_n\in\mathcal{M}_n,k\in\mathcal{K}}$
C^n	Cache size of SBS n
B^n	Bandwidth capacity of SBS n
ω_{m_n}	Weighted transmission parameter to BS of the classes MUs m_n
$\hat{\omega}_{m_n}$	Weighted transmission parameter to SBS n of the classes MUs m_n
$\mathbf{X}^t$	Vector of X^t
$\mathbf{Y}^t$	Vector of Y^t
β_n	Cache replacement parameter of SBS n
$z^t_{m_n,k}$	Load balancing variable of BS

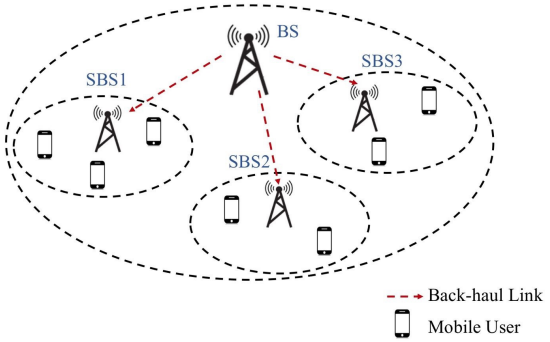

Fig. 1. An example of our proposed system model.

The BS offers a set $\mathcal{K} = \{1, 2, 3, ..., K\}$ of content items and these items are of the same size of o. Note that this assumption is justified in the real system that the contents are spilt into chunks of the same size and it has been used in many previous works (e.g. [12], [22]). For presentation simplicity, we assume $o = 1$ and normalize everything else accordingly.

Mobile users: Denote $\mathcal{M}_n = \{m_n : n \in \mathcal{N}\}$ as the set of classes of mobile users (MUs) in SBSs, where m_n represents the (multiple) MUs served by the SBS n. Each MU in m_n can either request from the BS or the SBS n.

The mean arrival rate at time t is denoted by $\lambda^t_{m_n,k}$, the demand of each MU class $m_n \in \mathcal{M}_n$ for content $k \in \mathcal{K}$ at time slot $t \in \mathcal{T}$. Denote by Λ^t the matrix of all $\lambda^t_{m_n,k}$. We set $\Lambda^t = 0$ for $t \leq 0$ and $t \geq T$. These request can be served by either the BS or the corresponding SBS, which is preferred when the requested item is in the cache of the SBS and there is

enough bandwidth for the transmission. Otherwise, the request is served by the BS instead.

Decision variables and constraints: We focus on the caching (re)placement policy and load balancing for each individual SBS in order to minimize the cost for serving requests from all MUs. Specifically, we seek the values of the following parameters:

- *Edge caching:* $X^t = (x^t_{n,k})_{n\in\mathcal{N},k\in\mathcal{K}}$, where $x^t_{n,k} \in \{0,1\}$ is the (integer) variable represents whether content k is cached in SBS n at time t ($x^t_{n,k} = 1$) or not ($x^t_{n,k} = 0$). Denote caching decision of all SBSs by X^t.
- *Load balancing between the BS and SBSs:* The fraction of requests served by the SBS is denoted by Y^t=$(y^t_{m_n,k})_{m_n\in\mathcal{M}_n,k\in\mathcal{K}}$, where $y^t_{m_n,k} \in [0,1]$ represents the fraction of requests from each MU class m_n for content k that is served by the SBS n at timeslot t. Similarly, the fraction of requests served by the BS is denoted by $Z^t = (z^t_{m_n,k})_{m_n\in\mathcal{M}_n,k\in\mathcal{K}}$, where $z^t_{m_n,k} \in [0,1]$ represents the fraction of requests from each MU class m_n for content k that is served by the BS n at timeslot t.

The caching policy of each SBS is restricted by its caching capacity C_n. Formally,

$$\sum_{k\in\mathcal{K}} x^t_{n,k} \le C_n, \forall n \in \mathcal{N}, t \in \mathcal{T}. \tag{1}$$

Similarly, the total load allocated to MUs $y^t_{m_n}$ cannot exceed the bandwidth capacity of each SBS, denoted as B_n. Formally, we have the following constraint:

$$\sum_{k\in\mathcal{K}} \sum_{m_n\in\mathcal{M}_n} \lambda^t_{m_n,k} y^t_{m_n,k} \le B_n, \quad \forall n \in \mathcal{N}, t \in \mathcal{T}. \tag{2}$$

The load balancing decision (percentage of the requested content served by the BS and the SBS) and the caching decision of SBS are tightly coupled. The request from an MU for an item k cannot be satisfied by SBS n if the item has not yet been cached by this SBS. This inter-dependency is captured by the following set of constraints:

$$y^t_{m_n,k} \le x^t_{n,k}, \quad \forall n \in \mathcal{N}, m_n \in \mathcal{M}_n, k \in \mathcal{K}, t \in \mathcal{T}, \tag{3}$$

which implies that if $x^t_{n,k} = 0$, then $y^t_{m_n,k} = 0$.

Actually, requests from the MUs must be satisfied by either BS or SBSs, i.e.,

$$z^t_{m_n,k} + y^t_{m_n,k} = 1, \quad \forall n \in \mathcal{N}, \forall m_n \in \mathcal{M}_n, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}. \tag{4}$$

Therefore, in the remainder of the paper, we ignore $z^t_{m_n,k}$ because it is fully decided by $y^t_{m_n,k} (z_{m_n,k} = 1 - y^t_{m_n})$.

B. The objective function

We aim to choose the $x^t_{n,k}$ and $y^k_{m_n,k}$ to minimize the cost for serving the requests from MUs, which is decomposed into the following three components:

- The operating cost incurred by serving the MUs directly by the BS: The cost is mostly due to resource consumption of the network such as spectrum, backhual link bandwidth, energy, etc.
- The operating cost incurred by serving the MUs by the SBSs: Similar to the BS, serving the requests by the SBSs consumes network resources. However, SBSs are close to the MUs in edge, therefore the cost such as delay and transmission energy is far less than that of the BS.
- The caching replacement cost incurred by updating the items cached in the SBSs: An SBS needs to fetch the new items from the BS and replace the old ones from the caching devices.

Now we model and discuss each component in details.

The operating cost for serving MUs from the BS: This is the sum of the operating cost of MUs requests that cannot be satisfied by the local SBSs across all MUs. For each SBS n, the operating cost depends on the number of received requests by the BS and the location of MUs. For instance, the MUs who are located at the boundary the BS cell incur higher operating cost, e.g., due to higher transmission power required and larger delay. We use a transmission weighted parameter $\omega_{m_n} \ge 0$ to describe the all these situations and captures the average impacts of the location of the MUs in the class m_n on the operating cost of the BS. In [23], the author models the energy consumption cost function as a linear function of the total transmission power of base stations. The assumption is at variance with objective reality because to deliver the packet to the MU under bad channel conditions, it takes more time which results in higher energy cost. In [8], the authors model the monetary cost for the energy consumption as a strictly convex function of the load and the transmission efficiency which can be modified to allow for the linear energy consumption function. In this paper, we assume that our cost function $f_t(\cdot)$ has the similar properties as we mentioned above. Formally, $f_t(\cdot)$ is assumed to be non-decreasing and jointly convex in all $y^t_{m_n,k}$'s.

The cost function $f_t(\cdot)$ can be any function that satisfies the properties discussed above. For example, the following is a representative function of the BS during timeslot t:

$$f_t(Y^t) = \sum_{n\in\mathcal{N}} \left(\sum_{m_n\in\mathcal{M}_n} \omega_{m_n} \sum_{k\in\mathcal{K}} \left(1 - y^t_{m_n,k}\right) \lambda^t_{m_n,k} \right)^2. \tag{5}$$

Note that this cost function is jointly convex in all $y^t_{m_n,k}$'s.

The operating cost for serving MUs by the SBSs: Similar to the previous description, it is the sum of the operating cost of MUs requests which can be served by the local SBSs directly across all classes of MUs. The operating cost function, denoted as $g_t(\cdot)$, depends on the volume of the received requests and the location of the MUs. The weighted parameter $\hat{\omega}_{m_n} \ge 0$ is used to describe how the locations of MUs effect the operating cost. Clearly, the distance from BS to MUs is far larger than the distance from SBSs to the corresponding MUs, the transmission power needed for BS is greater than SBSs. To describe this situation, the weighted parameter ω_{m_n} is greatly larger than the weighted parameter $\hat{\omega}_{m_n}$. Similiar with $f_t(\cdot)$, $g_t(\cdot)$ is assumed to be non-decreasing and jointly convex in all

$y^t_{m_n,k}$'s, the total operating cost of the SBSs during timeslot t is

$$g_t(Y^t) = \sum_{n\in\mathcal{N}}\Big(\sum_{m_n\in\mathcal{M}_n}\hat{\omega}_{m_n}\sum_{k\in\mathcal{K}} y^t_{m_n,k}\lambda^t_{m_n,k}\Big)^2. \quad (6)$$

Cache replacement cost: we consider the cost of updating cache contents between consecutive timeslots. Note that this component is often overlooked in previous work. This cost includes but not limits to the energy and delay incurred by the content update. Formally, the cache replacement cost for the SBS n from timeslot $t-1$ to t is

$$d(x^t_n, x^{t-1}_n) = \beta_n \sum_k \left(x^t_{n,k} - x^{t-1}_{n,k}\right)^+, \quad (7)$$

where β_n includes costs from different sources, e.g., the energy cost for cache replacement, the delay cost incurred during the update, network cost for downloading items from BS.

Therefore, the total cache placement cost is

$$h(X^t, X^{t-1}) = \sum_{n\in\mathcal{N}}\beta_n\sum_k\left(x^t_{n,k} - x^{t-1}_{n,k}\right)^+. \quad (8)$$

C. The optimization problem

Now we present the optimization problem that aims to minimize the objective function consisted of the three components mentioned previously by choosing the caching policy $x^t_{n,k}$ and the load balancing policy $y^t_{m_n,k}$ for each SBS and MU during the time horizon. Formally, we have the following formulation:

$$\min_{\mathbf{X}^t,\mathbf{Y}^t}\sum_{t\in\mathcal{T}}\left(f(Y^t)+g(Y^t)+h(X^t,X^{t-1})\right), \quad (9)$$

$$s.t.\ (1),(2),(3),$$

$$x^t_{n,k}\in\{0,1\},\ \forall n\in\mathcal{N}, k\in\mathcal{K}, t\in\mathcal{T}, \quad (10)$$

$$0\le y^t_{m_n,k}\le 1,\ \forall n\in\mathcal{N}, m_n\in\mathcal{M}_n, k\in\mathcal{K}, t\in\mathcal{T}. \quad (11)$$

This optimization problem is jointly convex in X^t and Y^t. However, there are two main challenges of this joint optimization problem from the following aspects:

(1) Caching policy $x^t_{n,k}$ needs to be an integer. This makes the problem a mixed-integer programming problem, which is NP-hard [24]. It is difficult to solve this problem even in the offline case, where all the information is known beforehand.

(2) Some information needed is not available when making the decision, e.g., future request arrival rates. However, due to the caching update costs, decisions of different timeslots are coupled. This makes the problem an online optimization. It is challenging to handle the integer constraint and the lack of future information simultaneously.

III. Offline Algorithm Design

We start from the offline algorithm design, where all necessary information is provided. This problem is a mixed 0-1 integer optimization, so it is not efficient to solve the problem directly. In this section, we propose a novel solution based on the primal-dual decomposition method to solve this offline problem and prove that it is guaranteed to find the optimal solution. This algorithm can be further used for the online optimization.

Note that $x^t_{n,k}$ and $y^t_{m_n,k}$ are coupled in the constraint (3). We start by relaxing the constraint (3) and introduce the set of dual Lagrange multipliers:

$$\mu^t = (\mu^t_{n,m_n,k} \ge 0 : \forall n\in\mathcal{N}, m_n\in\mathcal{M}_n, k\in\mathcal{K}, t\in\mathcal{T}). \quad (12)$$

As we show, this relaxation simplifies the problem since it can decouple the caching policy of SBSs and load balancing decision of the MUs. Similar methods are used in [8], [9].

The new Lagrange function is defined as follows,

$$\begin{aligned} L(\mathbf{X}^t,\mathbf{Y}^t,\mu^t) = &\sum_{t\in\mathcal{T}}\left(f(Y^t)+g(Y^t)+h(X^t,X^{t-1})\right)\\ &+\sum_{n\in\mathcal{N}}\sum_{m_n\in\mathcal{M}_n}\sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}}\mu^t_{n,m_n,k}(y^t_{m_n,k}-x^t_{n,k}). \end{aligned} \quad (13)$$

After introducing the Lagrange multiplier μ^t, the problem can be rewritten as,

$$\max_{\mu^{\mathbf{t}}}\min_{\mathbf{X}^t,\mathbf{Y}^t} L(\mathbf{X}^t,\mathbf{Y}^t,\mu^{\mathbf{t}}), \quad (14)$$

$$s.t.(1),(2),(10),(11),(12).$$

A. The primal-dual decomposition algorithm

To address this dual problem efficiently, we approach a primal-dual decomposition algorithm as shown in Algorithm 1. It is an iteration method, in each iteration l, the primal variable $(x^t_{n,k}, y^t_{m_n,k})$ is updated with the dual variable μ^t, then $x^t_{n,k}$ and $y^t_{m_n,k}$ will be inputted to the dual objective function.

Algorithm 1 Primal-dual Algorithm

Input: T, β_n, Λ^t, ω_{m_n}, $\hat{\omega}_{m_n}$ accuracy level $\epsilon = 0.0001$, maximum number of iterations L
Output: $\mathbf{X}^t$, $\mathbf{Y}^t$

```
Set μ = 0, the lower bound LB = −∞, the upper bound UB = +∞, and l = 1.
while (UB−LB)/UB > ε and l ≤ L do
    Solve sub-problems P1 for x^t_{n,k} and P2 for y^t_{m_n,k} (in parallel)
    Set h as the optimal value of the primal problem
    if h > LB then
        LB = h
    end if
    Update UB as the optimal value of (9)
    Update dual variables using (15)(16)(17)
    l=l+1
end while
```

The Dual Problem. The caching variable $x^t_{n,k}$ is discrete and the constraint sets of $x^t_{n,k}$ are discrete accordingly, the dual function can not be differentiable. Hence, the sub-gradient

method [25] [26] is employed. The following settings are chosen because of the easy implementations in practice, while other sub-gradient decent methods can also be adopted for our algorithm.

In each iteration of $l = 1, 2, 3, \ldots,$ for the dual problem, Lagrange multipliers are updated according to [26]

$$\mu_{n,m_n,k}^{t,(l+1)} = [\mu_{n,m_n,k}^{t,(l)} + \delta^{(l)} g_{n,m_n,k}^{t,(l)})]^+, \tag{15}$$

where $[\cdot]^+$ denotes the projections on the feasible set of μ^t. $\delta^{(l)}$ is the step size for the l update,

$$\delta^{(l)} = \frac{1}{1 + \alpha \cdot l}, \tag{16}$$

where α is the parameter to control the step size and $g_{n,m_n,k}^{t,(l)}$ is the current sub-gradient of iteration l. The sub-gradient of the dual variable is equal to the value of the respective inequality constraint of the primal problem [26].

$$g_{n,m_n,k}^{t,(l)} = y_{m_n,k}^{t,(l)} - x_{n,k}^{t,(l)}. \tag{17}$$

After obtaining the updated variables, the relaxed primal problem is solved to find the new primal variables.

The Primal Problem. Observe that the constraint sets of $x_{n,k}^t$ and $y_{m_n,k}^t$ are disjoint, then the primal problem can be decomposed as two separate classes of problems denoted as P_1 and P_2 respectively, each problem is solved in l iteration after the dual variables are updated, which as follows,

$$\begin{aligned} P_1 : \min_{\mathbf{X}^t} & \sum_{t\in\mathcal{T}} (h(X^t, X^{t-1}) - \\ & \sum_{n\in\mathcal{N}} \sum_{m_n\in\mathcal{M}_n} \sum_{k\in\mathcal{K}} \mu_{n,m_n,k}^t \cdot x_{n,k}^t), \\ & s.t. (1), (10), (12). \end{aligned} \tag{18}$$

$$\begin{aligned} P_2 : \min_{\mathbf{Y}^t} & \sum_{t\in\mathcal{T}} (y_t(Y^t) + g_t(Y^t) + \\ & \sum_{n\in\mathcal{N}} \sum_{m_n\in\mathcal{M}_n} \sum_{k\in\mathcal{K}} \mu_{m,k}^t y_{m_n,k}^t), \\ & s.t. (2), (11), (12). \end{aligned} \tag{19}$$

Caching Problem. Note that P_1 only involves the caching variables $x_{n,k}^t$, hence we name P_1 as caching problem. P_1 can also be composed into N independent sub-problems. For each SBS $n \in \mathcal{N}$, the sub-problem is denoted as P_1^n. Since $x_{n,k}^t$ is discrete, P_1^n is also an integer programming. Instead of solving this integer programming directly, we relaxed the constraint (10) (i.e., $x_{n,k}^t \in [0, 1]$), then the problem can be solved by the standard convex optimization techniques [25].

Load Balancing Problem. Similar to P_1, P_2 only involves the load balancing variables $y_{m_n,k}^t$, so we name it as load balancing problem. Note that the objective function is strictly convex, and the constraint sets are convex and continuous. Therefore, the standard convex optimization techniques can be applied [25].

B. Performance analysis

In Caching Problem P_1, caching variables are relaxed from {0,1} to [0,1], the solutions derived after the relaxation may not be feasible to the original problem if solution of the caching variables are decimal. To address this problem, in Theorem 1, we prove that the optimal solution to the relaxed integer problem is also the solution to the original integer problem. Formally,

Theorem 1. *The optimal solution of the relaxed integer problem is the optimal solution of the original integer problem and the optimal solution is integral.*

To prove Theorem 1, we need two lemmas first.

Lemma 1. *The optimal solution result of linear problem* $\min\{cx : Ax \leq b, x \geq 0\}$ *is the same with the integer linear problem* $\min\{cx : Ax \leq b, x \geq 0\}$ *cointegraing vectors if the A is the totally unimodular matrix.*

The proof of Lemma 1 can be found in [27].

Lemma 2. *(Hoffman and Kruskal) An integral matrix A is totally unimodular if and only if the polyhedron* $\{x : Ax \leq b, x \geq 0\}$ *is integral for each integral vector b.*

The proof of Lemma 2 can be found in [28].

Then, the proof of Theorem 1 is as follows,

Proof: To implement the properties of the linear programming, we write the P_1^n into the linear form by introducing a new set of variables,

$$p_n^t = \{p_{n,k}^t \geq 0 : \forall k \in \mathcal{K}\}. \tag{20}$$

Then the new problem can be re-formulated as :

$$\sum_{t\in\mathcal{T}} \left(\sum_{n\in\mathcal{N}} \sum_{k\in\mathcal{K}} \beta_n p_{n,k}^t - \sum_{n\in\mathcal{N}} \sum_{k\in\mathcal{K}} \sum_{m_n\in\mathcal{M}_n} \mu_{n,m_n,k}^t x_{n,k}^t \right), \tag{21}$$

$$s.t. (1), (10), (12), (20),$$

$$p_{n,k}^t \geq x_{n,k}^t - x_{n,k}^{t-1}, \forall n \in \mathcal{N}, k \in \mathcal{K}, t \in \mathcal{T}. \tag{22}$$

The new problem is equivalent to P_1^n and the new problem is a linear programming problem, hence we can employ Lemma 1 and Lemma 2 to prove Theorem 1.

From Lemma 1, if we prove that the constraints matrices are totally unimodular, we can prove that the optimal solution of relaxed linear problem is the optimal solution of integer problem. For constraint (1), it is clear that constraint matrix of $x_{n,k}^t$ is totally unimodular. For constraint (22), we rewrite (22) in the following form,

$$p_{n,k}^t - x_{n,k}^t + x_{n,k}^{t-1} \geq 0, \forall n \in \mathcal{N}, k \in \mathcal{K}, t \in \mathcal{T}, \tag{23}$$

Then with the constraint matrix D, constraint (23) can be transformed as follows,

$$D \cdot (p_{n,k}^t, x_{n,k}^t, x_{n,k}^{t-1})^T \geq 0, \tag{24}$$

where $(\cdot)^T$ means the transpose of a matrix. The matrix D can be written as

$$\boldsymbol{D} = \left\{1, -1, 1\right\}. \tag{25}$$

It is easy to check that D is a totally unimodular matrix. D can also be readily extended to the T dimensions. By constructing the totally unimodular matrix for the constraints and from Lemma 1, P_1^n can be solved with relaxed $x_{n,k}^t$ from $\{0,1\}$ to $[0,1]$. Next we prove that the optimal solution of relaxed integer problem is integral.

From Lemma 2, the optimal solution of the relaxed integer problem is on the vertex of the polyhedron by the constraints which are integers. ■

From Theorem 1, P_1^n can be solved by standard linear programming methods, simplex method is applied in this paper.

IV. Online Algorithms Design

Many online control algorithms have been studied in the literature to tackle the Online Convex Optimization (OCO) problems which consider the impact of the time-correlated adjustment cost [12], [19], [20]. Receding Horizon Control (RHC), also known as the Model Predictive Control (MPC) [29], [30] is a classic online control algorithm which has a long history handles both the prediction effect and the time coupling effect. In our previous work, we design the online algorithms Averaging Fixed Horizon Control (AFHC) [19] and Committed Horizon Control (CHC) [21].

We implement online algorithms RHC, AFHC and CHC to solve our problem, these online algorithms are designed for continuously convex problem. However, in this paper, the problem is an optimization with integer variables $x_{n,k}^t$. The theoretical bounds proposed for these online algorithms can not be guaranteed for integer programming, and solution employed by RHC, AFHC and CHC directly are not feasible for our integer problem. So RHC, AFHC and CHC can not be employed directly. To tackle this challenge, we propose the fixed version of RHC, AFHC and CHC for the integer programming and theoretical performance bounds are also derived.

In this section, firstly, we briefly introduce RHC and CHC (AFHC is the special case of CHC in the integer programming). Then we present the integer version and the proof of theoretical bounds for each of them.

In the offline problem, all the information of the system is available, e.g., all the MUs requests information Λ^t. In the online problem, information about the system in the future will be inaccurate or even unknown because there are often significant prediction errors, and the prediction quality would be worse if predicted further into the future. However, in many applications, it is possible to estimate the information in the near future, such as requests for videos [15], workloads for data centers [20] and information about solar and wind energy [19].

All these online algorithms use a prediction horizon/window of size w, RHC, AFHC and CHC make decisions in different ways. In each time slot, RHC determines the actions in horizon w to minimize the total cost. RHC assigns the first action of the horizon to the next one predicted time slot. Similar with RHC, AFHC first derives all w actions and then averages them. CHC is the generalization of RHC and AFHC. Instead of committing the fixed actions to the prediction, CHC allows for arbitrary levels of commitment.

A. Receding Horizon Control (RHC)

1) Introduction of RHC: At each time-step τ, RHC solves the cost optimization problem over the window $(\tau, \tau + w)$ when given the starting state $x_{\tau-1}$ and $y_{\tau-1}$ and the length the prediction window (horizon) w.

Formally, define $\lambda_{\cdot|\tau}$ as the vector $(\lambda_{\tau+1|\tau}, \cdots, \lambda_{\tau+w|\tau})$, the prediction of $\lambda_{\cdot|\tau}$ in a w time steps prediction window at time τ. Let $X^\tau(x_{n,k}^t, \lambda_{\tau+w|\tau})$ and $Y^\tau(y_{m_n,k}^t, \lambda_{\tau+w|\tau})$ as the vector in $\mathcal{R}^w$ indexed by $t \in T^\tau = \{\tau, \cdots, \tau+\omega\}$, which are the solutions to

$$\min_{\mathbf{X}^\tau, \mathbf{Y}^\tau} \sum_{t \in \mathcal{T}^t} \left(f_t(Y^t) + g_t(Y^t) + h(X^t, X^{t-1})\right), \tag{26}$$

$$\text{s.t.} \quad x_{n,k}^t \in \{0,1\}, \ \forall n \in \mathcal{N}, k \in \mathcal{K}, t \in \mathcal{T}^\tau \tag{27}$$

$$0 \le y_{m_n,k}^t \le 1, , \ \forall n \in \mathcal{N}, m \in \mathcal{M}, k \in \mathcal{K}, t \in \mathcal{T}^\tau, \tag{28}$$

$$\sum_{k \in \mathcal{K}} x_{n,k}^t \le C_n, \forall n \in \mathcal{N}, t \in \mathcal{T}^\tau, \tag{29}$$

$$\sum_{k \in \mathcal{K}} \sum_{m_n \in \mathcal{M}_n} \lambda_{m_n,k}^t y_{m_n,k}^t \le B_n, \quad \forall n \in \mathcal{N}, t \in \mathcal{T}^\tau, \tag{30}$$

$$y_{m_n,k}^t \le x_{n,k}^t, \ \forall n \in \mathcal{N}, m_n \in \mathcal{M}_n, k \in \mathcal{K}, \tau \in \mathcal{T}^\tau. \tag{31}$$

This problem can be solved in the similar method to get $X^t(x_{n,k}^t, \lambda_{\tau+w|\tau})$ and $Y^t(y_{m_n,k}^t, \lambda_{\tau+w|\tau})$ as we discussed in Section III. By introducing the Lagrange multipliers for constraint (31), the Algorithm 1 can be employed to solve this problem.

Algorithm 2. *Receding Horizon Control*

For all $t \le 0$, set $x_{RHC,n,k}^t = 0$ and $y_{RHC,m_n,k}^t = 0$. At each time slot $\tau \ge 1$, set the caching states of SBS n at time slot τ to

$$x_{RHC,n,k}^\tau = X_\tau^\tau(x_{RHC,n,k}^t, \lambda_{\tau+w|\tau}). \tag{32}$$

Similarly, set the vector of routing variable as

$$y_{RHC,m_n,k}^\tau = Y_\tau^\tau(y_{RHC,m_n,k}^t, \lambda_{\tau+w|\tau}). \tag{33}$$

The competitive ratios of RHC is $O(1 + \frac{1}{w})$ [19].

2) Integer version of RHC and theoretical bound: RHC studied in [19] is the convex problem with continuous decision variables. In this paper, the problem is the mixed integer programming, the competitive ratio of RHC can not be applied directly. However, we prove in the Theorem 2 that the RHC bound of the mixed inter problem is the same with the continuous convex problem.

Theorem 2. *The competitive ratio of the RHC with mixed integer problem is the same as the continuous convex problem.*

Proof: From Theorem 1, we know that the optimal solution of the caching problem can be solved with relaxed integer variables. In the online problem, after relaxing the integer variables $x_{n,k}^t$, the objective function (26) is strictly convex and the constraint sets are convex, so the competitive ratio of the integer problem is still $O(1+\frac{1}{w})$. ■

B. Committed Horizon Control (CHC)

1) Introduction of CHC: CHC is the generalization of the AFHC and RHC. CHC introduces a new parameter, commitment level $r \in [0, w]$, which allows to average fixed r levels decisions. Formally, let

$$\Psi_v = \{i : i \equiv v \mod r\} \cap [-r+1, T], v = 0, \cdots, r-1.$$

Actions denoted as $x_{FHC,n,k}^{t,(v)}$ and $y_{FHC,m_n,k}^{t,(v)}$ determined in the fixed commitment level r are defined as follows, by using (26) to set

$$x_{FHC,n,k}^{t,(v)} = X_\tau^\tau(x_{FHC,n,k}^{t,(v)}, \lambda_{\cdot|\tau}). \tag{34}$$

Similarly, set the vector of routing variable as

$$y_{FHC,m_n,k}^{t,(v)} = Y_\tau^\tau(y_{FHC,m_n,k}^{t,(v)}, \lambda_{\cdot|\tau}), \tag{35}$$

for all $t \leq 0$, set $x_{FHC,n,k}^t = 0$ and $y_{FHC,m_n,k}^t = 0$. At $\tau \in \Psi_v$, $t \in \{\tau, \cdots, \tau+w\}$, CHC takes the average of r with commitment level r, and window size is w.

Algorithm 3. *Committed Horizon Control.*

At time slot $t \in \Psi^r$, for all v, CHC averages the actions $\{x_{n,k}^\tau, x_{n,k}^{\tau+1}, \cdots, x_{n,k}^{\tau+v}\}$ and $\{y_{m_n,k}^\tau, \cdots, y_{m_n,k}^{\tau+v}\}$ determined by equations (34) and (35), then sets

$$x_{CHC,n,k}^t = \frac{1}{r}\sum_{v=0}^{r-1} x_{CHC,n,k}^{t,(v)}, \tag{36}$$

$$y_{CHC,m_n,k}^t = \frac{1}{r}\sum_{v=0}^{r-1} y_{CHC,m_n,k}^{t,(v)}. \tag{37}$$

2) Integer version of CHC and the theoretical bound: In [21], similar with the RHC, the objective function of CHC is assumed to be continuously convex and the results from equations (34) and (35) are continuous. Hence, after averaging the actions in each commitment window, the final results are also continuous and feasible to the original functions.

In this paper, problem (26) is an optimization with integer variables $x_{n,k}^t$. From Theorem 1, we know that the optimal solution of $x_{n,k}^t$ is integral. From (36), after averaging them, $x_{CHC,n,k}^t$ may be fractional unless every $x_{CHC,n,k}^{t,(v)}$ is 1. For instance, if the solutions of $x_{FHC,n,k}^{t,(v)}$ are {1,0,1,0,1} for the commitment level of $r = 5$, then $x_{CHC,n,k}^{t,(v)}$ is 0.6, which is not feasible and does not have the realistic meaning in our system. The average value of caching variables is denoted as $\widetilde{x}_{CHC,n}^\tau = \{\widetilde{x}_{CHC,n,1}^t, \cdots, \widetilde{x}_{CHC,n,K}^t\}$. To tackle this problem, we propose a rounding policy for CHC and prove the performance bound for the rounding policy.

CHC Rounding Policy. The caching policy $x_{n,k}^t$ and load balancing policy $y_{m_n,k}^t$ are coupled together. Hence, the rounding policy contains two steps. We determine $x_{n,k}^t$ first, and then $y_{m_n,k}^t$ is determined according to the $x_{n,k}^t$.

(i) $x_{n,k}^t$. We set the boundary value as ρ ($\rho \in (0,1)$). For all $k \in \mathcal{K}$, if $\widetilde{x}_{CHC,n,k}^t$ is greater or equal to ρ, then $x_{CHC,n,k}^t = 1$, otherwise $x_{CHC,n,k}^t = 0$.

(ii) $y_{m_n,k}^t$. If $x_{CHC,n,k}^t = 0$, then $y_{CHC,m_n,k}^t = 0$, else $y_{CHC,m_n,k}^t$ is calculated from (37).

The optimal rounding boundary value is derived in the Theorem 3.

To make it easier to derive the theoretical bound of CHC rounding policy, we rewrite another linear form of cache replacement cost $d(x_{n,k}^t, x_{n,k}^{t-1})$ as the following, which is equivalent to $d(x_{n,k}^t, x_{n,k}^{t-1})$,

$$\phi(x,t) = \begin{cases} \beta_n x_{n,k}^t, & if x_{n,k}^{t-1} = 0 \\ 0, & if x_{n,k}^{t-1} = 1 \end{cases} \tag{38}$$

then the switching cost can be rewritten as:

$$h(X^t, X^{t-1}) = \sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}} \phi(x,t). \tag{39}$$

To simplify the proof process, we denote the total operating cost from objective function (26) as $C(X^t, Y^t)$. $C(X^t, Y^t)^\dagger$, $h(X^t, X^{t-1})^\dagger$, $f_t(Y^t)^\dagger$ and $g_t(Y^t)^\dagger$ are the results after rounding, $C(X^t, Y^t)^*$, $h(X^t, X^{t-1})^*$, $f_t(Y^t)^*$ and $g_t(Y^t)^*$ are the results without rounding. The theoretical bound for the CHC rounding policy is derived as follows.

Theorem 3. *The CHC rounding policy is an approximation algorithm to the original CHC without rounding and it achieves an approximation ratio of* 2.62*, e.g.,* $C(X^t, Y^t)^\dagger \leq 2.62C(X^t, Y^t)^*$.

Proof: Theorem 3 is proved for three cost functions $h(X^t, X^{t-1})$, $f_t(Y^t)$ and $g_t(Y^t)$ respectively.

(1) The bound of $h(X^t, X^{T-1})$.

$$\begin{aligned} h(X^t, X^{t-1})^* &= \sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}} \phi(\widetilde{x}_{CHC,n}^t) && (40)\\ &\geq \sum_{\{[n,k]|\widetilde{x}_{CHC,n,k}^t \geq \rho\}} \phi(\widetilde{x}_{CHC,n}^t) && (41)\\ &\geq \rho \cdot \sum_{\{[n,k]|\widetilde{x}_{CHC,n,k}^t = 1\}} \phi(\widetilde{x}_{CHC,n}^t) && (42)\\ &= \rho \cdot \sum_{\{[n,k]|x_{CHC,n,k}^t\}} \phi(x_{CHC,n}^t) && (43)\\ &= \rho \cdot h(X^t, X^{t-1})^\dagger. && (44) \end{aligned}$$

We explain the deductions step by step here. Firstly, inequality (41) is satisfied because $\{[n,k]|\widetilde{x}_{CHC,n,k}^t \geq \rho\} \subseteq \{[n,k]|\widetilde{x}_{CHC,n,k}^t\}$. Inequality (42) exists because $\{[n,k]|\widetilde{x}_{CHC,n,k}^t \geq \rho\} = \{[n,k]|\widetilde{x}_{CHC,n,k}^t = 1\}$ and $\rho \geq 0$. In (43), equality exists because $\widetilde{x}_{CHC,n,k}^t = x_{CHC,n,k}^t = 1$. So $h(X^t, X^{T-1})^\dagger \leq \frac{1}{\rho} \cdot h(X^t, X^{T-1})^*$.

(2) The bound of $f_t(Y^t)$. After determining the $x^t_{CHC,n,k}$, the bound for $f_t(Y^t)$ can be derived accordingly. From the rounding policy, when $x^t_{CHC,n,k} = 1$, $y^t_{CHC,m_n,k}$ follows the equation (37), so there is no gap and $f_t(Y^t)^\dagger = f_t(Y^t)^*$. When $x^t_{CHC,n,k} = 1$, then $y^t_{CHC,m_n,k} = 0$, the bound of $f_t(Y^t)$ is derived by the following.

$$\begin{aligned} f_t(Y^t)^* = \quad & f_t(Y^t = \{y^t_{CHC,m_n,k} : t \in \mathcal{T}^\tau\}) && (45)\\ \geq \quad & f_t(Y^t = \{y^t_{CHC,m_n,k} = \rho : t \in \mathcal{T}^\tau\}) && (46)\\ = \quad & \frac{1}{(1-\rho)^2} f_t(Y^t = \{y^t_{CHC,m_n,k} = 0 : t \in \mathcal{T}^\tau\}) && (47)\\ = \quad & \frac{1}{(1-\rho)^2} f_t(Y^t)^\dagger && (48) \end{aligned}$$

We explain the deductions step by step. Inequality (46) exists because $y^t_{CHC,m_n,k} \leq \hat{x}^t_{CHC,n,k} = \rho$ and $f_t(Y^t)$ decreases when $y^t_{CHC,m_n,k}$ is larger. Then we put $\frac{1}{(1-\rho)^2}$ ahead which equals to $y^t_{CHC,m_n,k} = 0$, then we get (47) (48). So $f_t(Y^t)^* \geq \frac{1}{(1-\rho)^2} f_t(Y^t)^\dagger$.

(3) The bound of $g_t(Y^t)$. Similar proof can be derived as $f_t(Y^t)$. We get $g_t(Y^t)^* \geq \frac{1}{\rho^2} g_t(Y^t)^\dagger$.

So the approximation ratio for $C(X^t, Y^t)$ is

$$\max\{\frac{1}{\rho}, \frac{1}{\rho^2}, \frac{1}{(1-\rho)^2}\}$$

$\rho \in (0, 1)$, so when $\frac{1}{\rho} = \frac{1}{(1-\rho)^2}$ ($\rho = \frac{3-\sqrt{5}}{2} \approx 2.62$), the approximation ratio is minimal. Therefore, the approximation ratio is 2.62. ∎

Note that AFHC is an extreme case of CHC when the commitment level r equals to the window size w, AFHC is introduced repeatedly. As one specially case of CHC, the rounding policy can also be applied to AFHC, so does the theoretical bound.

V. Numerical Evaluation

In this section, we conduct numerical simulations to further evaluate the performance of our proposed algorithms. To fully understand the performance of the algorithms, we aim to answer the following three questions:

1) How much can our online and offline algorithms reduce the costs compared to existing solutions?
2) What are the impacts of time-correlated cache replacement costs on the caching and load balancing in the 5G edge caching system?
3) Which online algorithm should we choose?

To the best of our knowledge, this is the first paper for the implementation of CHC in the a specific problem .

A. Methodology and performance criteria

In this paper, we compare the performance of the following schemes,

- Offline optimal solution: The primal-dual algorithm is implemented over the entire time horizon with all the information provided. This is the optimal result of the problem and serves as an unrealistic lower bound.
- Least Recently Frequently Used (LRFU): LRFU is an algorithm combined with two typical caching algorithms, namely, the Least Frequently Used (LFU) and the Least Recently Used (LRU). In LRU, the cache replaces one item which has been requested least recently at each time, but it can only fit the situation with one user. In LFU, it replaces one item which has the most requests at each time slot, however, it can only replace one item at each time. To combine LRU and LFU together, LRFU, at each timeslot, SBSs cache the contents ranking by the MUs' requests number from high to low with the limitation of the cache size.
- Online algorithms: Integer versions of RHC, AFHC and CHC.

B. Simulation setup

We consider a single BS serving a circular area. The total number of files $K = 30$. In 5G network, each SBS is independent from each other and each classes of MUs only served by one SBS or the BS. Hence, the number of SBS is set as 1. When consider multiple SBSs, the final results are the sum of each SBS. The total time duration is 100 timeslots. Note that, each SBS n is endowed with the cache size of 5, bandwidth capacity of 30, e.g., the SBS is capable to transmit at most 30 files at one time slot.

The number of MUs is 30. All MUs are normally distributed in the coverage of the SBS. The transmission efficiency parameter ω_{m_n} is randomly chosen from $[0, 1]$ which means its distance to the BS normalized by the radius of the BS. The transmission efficiency parameter $\hat{\omega}_{m_n}$ is set as 0. The distance from SBS to MU is far less than the distance for BS to MU because SBSs are placed in edge to serve MUs. For example, if the distance from BS to one class of MUs is 100 times than the distance from the SBS to MUs, then $\hat{\omega}_{m_n} = 0.01\omega_{m_n}$, so the operating cost of SBSs can be ignored compared with the operating cost of BS in the simulation.

We use the Zipf-Mandelbrot model [31] to formulate the MUs' requests pattern which is presented as follows,

$$p(i) = \frac{K}{(i+q)^\alpha}, \tag{49}$$

in which the shape parameter $\alpha = 0.8$ and the shift parameter $q = 30$. The requests density of each class of MUs is randomly picked from [0,100].

In the offline optimal solution and LRFU, the MUs' requests information is accurate and known for all timeslots. In online algorithms, the short term of MUs' requests information of future is predicted, which is not accurate. Hence, we add the perturbation parameter $\eta \in [0, 1]$ which means the $p(i)$ would be randomly chosen from $[(1-\eta)p(i), (1+\eta)p(i)]$, unless otherwise specified, η is set as 0.1.

For the cache replacement cost, we set $\beta = 100$ by default, which corresponds to the operating cost to replace one content in SBS at each timeslot. Fig. 2 tests different β. For the prediction window, we set $w = 10$, by default. It corresponds to 10 time slots prediction of the MUs' requests. We vary w in Fig. 3 to examine the impact of total operating cost.

C. Experimental results

With the setup mentioned above, we perform several simulations to evaluate the performance of proposed algorithms and the impact of the time-correlated cache replacement cost.

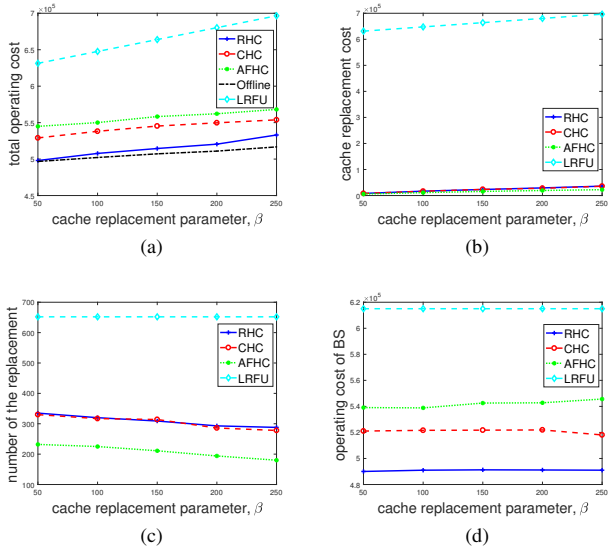

Fig. 2. The impact of the cache replacement cost β. (a) The total operating cost. (b) The cache replacement cost. (c) The number of cache replacement times. (d) The operating cost of BS.

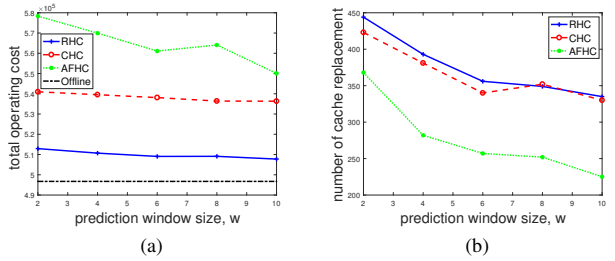

Fig. 3. The impact of the predict window w. (a) The total operating cost. (b) The number of cache replacement times.

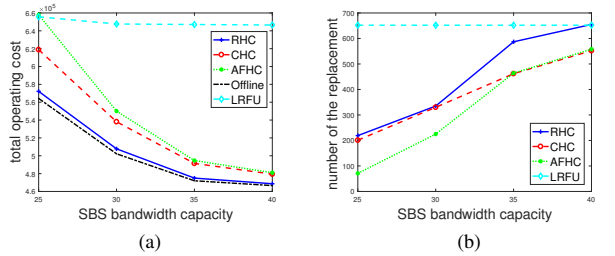

Fig. 4. The impact of the SBS bandwidth. (a) The total operating cost. (b) The number of cache replacement times.

1) *The performance of the online algorithms*. In Fig. 2a, we choose the point with $\beta = 50$. RHC, CHC and AFHC can reduce the total operating cost significantly by 27%, 20% and 17% respectively when compared with LRFU. The cost ratio of RHC, CHC, AFHC and LRFU to offline is 1.02, 1.08, 1,11 and 1.3. The total operating cost of RHC is very close to the offline. The performance of CHC and AFHC is not as good as RHC because of the rounding policy. As AFHC is the special case of CHC, CHC would not be worse than AFHC, and the simulation results also prove it.

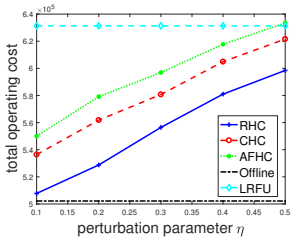

Fig. 5. The impact of the perturbation parameter η

2) *The impact of the cache replacement cost* β. In Fig. 2, we shows the how total operating cost, the cache replacement cost, the number of cache replacement times and the operating cost of BS influenced by the replacement cost parameter β. Fig. 2a exhibits that when the cache replacement parameter β increases, the total offline operating cost of proposed system, LRFU and online algorithms also increase. However, the performance of online algorithms is much better than LRFU, especially for RHC, the performance of RHC is very close to the optimum which is the offline. The total operating cost growth rate of LRFU is larger than online algorithms and the offline. Fig. 2b and Fig. 2c illustrate it. The number of cache replacement times of online algorithms reduces with increase of the β because when β is larger, the cache replacement cost will have larger impact of the total operating cost. Hence, the number of cache replacement times are adjusted to be smaller to minimize the total operating cost. The number of cache replacement times of LRFU is the same because when requests of MUs follow the same request pattern, the caching policy remains the same, the operating cost of BS also remains the same, and only the cache replacement cost increases linearly with β. In Fig. 2d, the operating cost of BS is steady with the increase of β because online algorithms minimize the total operating cost by reduce the number of cache replacement times to make the impact caused by β less.
3) *The impact of prediction window* w *for online algorithms.* Fig. 3 compares the total operating cost and the number of cache replacement times of RHC, AFHC and CHC as the prediction window size w varies. In Fig. 3a, as the prediction window becomes larger, all the online algorithms move closer to the optimal offline, and in Fig. 3b, the number of cache replacement times decrease. RHC has the least cost. We can conclude that when

the system has more prediction information about MUs' requests, the online algorithms perform better.

4) *The impact of SBS bandwidth capacity*. Fig. 4 depicts how the total operating cost and the number of cache replacement times vary when the SBS's bandwidth capacity changes. When SBS's bandwidth capacity becomes larger which means SBS can send more items to satisfy the requests of MUs as each time slot. The total operating cost of offline, online algorithms and LRFU reduces. The cost of LRFU reduces slowly. In Fig. 4b, the number of cache replacement times of LRFU remains the same, but for online algorithms, the number of cache replacement times increases fast. The reason for that is when SBS's cache size is fixed, the caching policy of LRFU does not change. However, for online algorithms, the number cache replacement times will increase a lot to satisfy the MUs' requests until SBS's bandwidth capacity is large enough to serve all the MUs' requests.
5) *The impact of the prediction noise*. Fig. 5 exhibits the total operating cost incurred by the predicted inputs with different perturbation parameter η. The larger the η, the more inaccurate for the predicted MUs' requests. The total operating cost of online algorithms is higher with the increase of the η. The total operating cost for LRFU dose not change because it implements the data of requests without noise. When $\eta = 0.5$, AFHC has the same performance with LRFU.

VI. Related Work

In recent years, many studies about edge caching have been conducted. The first type of caching systems relies on rule-based cache replacement algorithms as FIFO, Least Recently Used (LRU), Least Frequently Used (LFU), or their variants [32]. These algorithms follow simplified rules and are easy to be implemented in reality, but the fixed rules can hardly adapt to the dynamic content access patterns.

Many previous works focus on edge caching in the 5G network or the cellular network. Poularakis *et al.* define a network consists of one BS and several SBSs, in [8], the operator of BS leases the available cache and bandwidth resources to serve the requests of MUs, the problem is formulated as a two-stage Stackelberg game to minimize the total serving cost, in [9], they minimize the latency of the layered videos, formulate the optimization problem as a multiple-choice knapsack problem and propose an approximation algorithm within a 2 factor from the optimum. Li *et al.* [11] consider a edge optimization problem for the adaptive video-on-demand system to maximize the quality of experience. Du *et al.* [33] design the incentivized traffic offloading and resource allocation contracts to motivate SBS maximum their utility. In [10], the authors consider a heterogeneous cellular network and they design a distributed caching problem to minimize download latency via belief propagation, but they simply solve a caching problem and do no consider the transmission bandwidth constraints which is not practical. Nevertheless, the storage cost and the system replacement cost are not considered in the studies mentioned above.

With the advancement of data analysis, forecast-based cache replacement algorithms have been recently suggested in [13]–[15] . These well-trained models with engineering feature can achieve a high hit ratio. However, they require large amount of historical data for training the results which are heavily depend on the specific data set and is hardly adaptive. It is also impractical for the operator of BS to adapt the cache replacement policies because it may cost a long time for training data, and results in the heavy computing load and low adaptation for various system models.

Many works have been conducted to solve the online caching problem in a period of time. Gu et al. [16] model the cache replacement problem as the Markov decision process and propose a Q-learing based caching policy, and minimize the data transmission cost among the SBSs. In [17], the authors propose a problem to minimize caching switching cost in storage memory. Menache et al. [18] propose the problem how multiply users allocate one caching memory, they develop an online caching algorithm for arbitrary cost. The authors in [34] consider the performance of coded caching in online system. These works neglect either the storage cost or time-correlated adjustment costs.

The time-correlated adjustment cost has got widespread attention in many domains. In [12], the authors consider edge caching in Cloud Radio Access Networks (C-RAN), the problem is formulated as an integer programming. In [19], [20], the authors propose the load balancing problem to minimize the total data center cost in which the switch cost of servers is considered.

VII. Concluding Remarks

This paper studies the problem of jointly optimize edge caching and load balancing for mobile data offloading in 5G networks. Compared to existing work, the new challenge is to tackle the integer constraint in an online manner with limited future information. We first propose a primal-dual decomposition algorithm to solve the problem offline for the integer constraint with guaranteed optimality, and then incorporate various online algorithms for decision making with limited information with performance guarantee. Extensive numerical evaluations highlight the efficiency of the proposed algorithms. In the future, we plan to develop distributed algorithms and handle potential strategic behaviors of individual SBSs.

Acknowledgment

This work is supported in part by US National Science Foundation under grant numbers 1513719 and 1730291.

References

[1] D. M. Scott, *The new rules of marketing and PR: How to use social media, online video, mobile applications, blogs, news releases, and viral marketing to reach buyers directly*. John Wiley & Sons, 2015.
[2] V. N. I. Cisco, "Global mobile data traffic forecast update, 2015–2020 white paper," *Document ID*, vol. 958959758, 2016.

[3] S. K. Barker and P. Shenoy, "Empirical evaluation of latency-sensitive application performance in the cloud," in *Proceedings of the first annual ACM SIGMM conference on Multimedia systems*. ACM, 2010, pp. 35–46.

[4] A. Gupta and R. K. Jha, "A survey of 5g network: Architecture and emerging technologies," *IEEE access*, vol. 3, pp. 1206–1232, 2015.

[5] A. Ghosh, N. Mangalvedhe, R. Ratasuk, B. Mondal, M. Cudak, E. Visotsky, T. A. Thomas, J. G. Andrews, P. Xia, H. S. Jo *et al.*, "Heterogeneous cellular networks: From theory to practice," *IEEE communications magazine*, vol. 50, no. 6, 2012.

[6] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5g wireless networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1617–1655, 2016.

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

[8] K. Poularakis, G. Iosifidis, and L. Tassiulas, "A framework for mobile data offloading to leased cache-endowed small cell networks," in *Mobile Ad Hoc and Sensor Systems (MASS), 2014 IEEE 11th International Conference on*. IEEE, 2014, pp. 327–335.

[9] K. Poularakis, G. Iosifidis, A. Argyriou, I. Koutsopoulos, and L. Tassiulas, "Caching and operator cooperation policies for layered video content delivery," in *INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE*. IEEE, 2016, pp. 1–9.

[10] J. Li, Y. Chen, Z. Lin, W. Chen, B. Vucetic, and L. Hanzo, "Distributed caching for data dissemination in the downlink of heterogeneous networks," *IEEE Transactions on Communications*, vol. 63, no. 10, pp. 3553–3568, 2015.

[11] C. Li, L. Toni, J. Zou, H. Xiong, and P. Frossard, "Qoe-driven mobile edge caching placement for adaptive video streaming," *IEEE Transactions in Multimedia*, vol. 20, no. ARTICLE, pp. 965–984, 2018.

[12] L. Pu, L. Jiao, X. Chen, L. Wang, Q. Xie, and J. Xu, "Online resource allocation, content placement and request routing for cost-efficient edge caching in cloud radio access networks," *IEEE Journal on Selected Areas in Communications*, 2018.

[13] H. Pang, J. Liu, X. Fan, and L. Sun, "Toward smart and cooperative edge caching for 5g networks: A deep learning based approach."

[14] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "When edge meets learning: Adaptive control for resource-constrained distributed machine learning," *arXiv preprint arXiv:1804.05271*, 2018.

[15] G. Ma, Z. Wang, M. Zhang, J. Ye, M. Chen, and W. Zhu, "Understanding performance of edge content caching for mobile video streaming," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 5, pp. 1076–1089, 2017.

[16] J. Gu, W. Wang, A. Huang, H. Shan, and Z. Zhang, "Distributed cache replacement for caching-enable base stations in cellular networks," in *Communications (ICC), 2014 IEEE International Conference on*. IEEE, 2014, pp. 2648–2653.

[17] S. M. Azimi, O. Simeone, A. Sengupta, and R. Tandon, "Online edge caching in fog-aided wireless networks," in *Information Theory (ISIT), 2017 IEEE International Symposium on*. IEEE, 2017, pp. 1217–1221.

[18] I. Menache and M. Singh, "Online caching with convex costs," in *Proceedings of the 27th ACM symposium on Parallelism in Algorithms and Architectures*. ACM, 2015, pp. 46–54.

[19] M. Lin, Z. Liu, A. Wierman, and L. L. Andrew, "Online algorithms for geographical load balancing," in *Green Computing Conference (IGCC), 2012 International*. IEEE, 2012, pp. 1–10.

[20] Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang, M. Marwah, and C. Hyser, "Renewable and cooling aware workload management for sustainable data centers," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 1. ACM, 2012, pp. 175–186.

[21] N. Chen, J. Comden, Z. Liu, A. Gandhi, and A. Wierman, "Using predictions in online optimization: Looking forward with an eye on the past," *ACM SIGMETRICS Performance Evaluation Review*, vol. 44, no. 1, pp. 193–206, 2016.

[22] Y. Huang, X. Song, F. Ye, Y. Yang, and X. Li, "Fair caching algorithms for peer data sharing in pervasive edge computing environments," in *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*. IEEE, 2017, pp. 605–614.

[23] O. Arnold, F. Richter, G. Fettweis, and O. Blume, "Power consumption modeling of different base station types in heterogeneous cellular networks," *Future network and mobile summit*, vol. 2010, pp. 1–8, 2010.

[24] C. A. Floudas, *Nonlinear and mixed-integer optimization: fundamentals and applications*. Oxford University Press, 1995.

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

[26] D. P. Bertsekas and A. Scientific, *Convex optimization algorithms*. Athena Scientific Belmont, 2015.

[27] G. L. Nemhauser and L. A. Wolsey, "Integer programming and combinatorial optimization," *Wiley, Chichester. GL Nemhauser, MWP Savelsbergh, GS Sigismondi (1992). Constraint Classification for Mixed Integer Programming Formulations. COAL Bulletin*, vol. 20, pp. 8–12, 1988.

[28] A. J. Hoffman and J. B. Kruskal, "Integral boundary points of convex polyhedra," in *50 Years of Integer Programming 1958-2008*. Springer, 2010, pp. 49–76.

[29] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, 2017.

[30] W. Kwon and A. Pearson, "A modified quadratic cost problem and feedback stabilization of a linear system," *IEEE Transactions on Automatic Control*, vol. 22, no. 5, pp. 838–842, 1977.

[31] M. Hefeeda and O. Saleh, "Traffic modeling and proportional partial caching for peer-to-peer systems," *IEEE/ACM Transactions on networking*, vol. 16, no. 6, pp. 1447–1460, 2008.

[32] S. Podlipnig and L. Böszörmenyi, "A survey of web cache replacement strategies," *ACM Computing Surveys (CSUR)*, vol. 35, no. 4, pp. 374–398, 2003.

[33] J. Du, E. Gelenbe, C. Jiang, H. Zhang, and Y. Ren, "Contract design for traffic offloading and resource allocation in heterogeneous ultra-dense networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2457–2467, 2017.

[34] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, "Online coded caching," *IEEE/ACM Transactions on Networking (TON)*, vol. 24, no. 2, pp. 836–845, 2016.