Learning to Cache: Federated Caching in a Cellular Network With Correlated Demands

Citation for published version:
Krishnendu, S, Bharath, BN, Garg, N, Bhatia, V & Ratnarajah, T 2021, 'Learning to Cache: Federated Caching in a Cellular Network With Correlated Demands', *IEEE Transactions on Communications*. https://doi.org/10.1109/TCOMM.2021.3132048

Digital Object Identifier (DOI):
10.1109/TCOMM.2021.3132048

Published In:
IEEE Transactions on Communications

Learning to Cache: Federated Caching in a Cellular Network With Correlated Demands

S. Krishnendu, B. N. Bharath, Navneet Garg, Vimal Bhatia
and Tharmalingam Ratnarajah*‡

***Abstract*—In this paper, the problem of distributed content caching in a small-cell Base Stations (sBSs) wireless network that maximizes the cache hit performance is considered. Most of the existing works consider static demands, however, here, data at each sBS is considered to be correlated across time and sBSs. Federated learning (FL) based caching strategy is proposed which is assumed to be a weighted combination of past caching strategies of neighbouring base stations. A high probability generalization guarantees on the performance of the proposed federated caching strategy is derived. The theoretical guarantee provides following insights on obtaining the caching strategy: (i) run regret minimization at each sBS to obtain a sequence of caching strategies across time, and (ii) maximize an estimate of the bound to obtain a set of weights for the caching strategy which depends on the discrepancy. Also, theoretical guarantee on the performance of the least recently frequently used (LRFU) caching strategy is derived. Further, FL based heuristic caching algorithm is also proposed. Finally, it is shown through simulations using Movie Lens dataset that the proposed algorithm significantly outperforms the recent online learning algorithms.**

***Index Terms*—Distributed content caching, online learning, non-stationary demands, regret minimization.**

I. Introduction

There is a pressing need for revamping of the next generation wireless infrastructure network due to an unprecedented increase in the data demand in the recent years [1]. There has been several proposals for new wireless network designs for handling this surge in the data demand. A few examples designs include Fog network [2] with edge computing, deployment of small cells to offload wireless data from a macro Base Station (BS), integrating existing WiFi access points to share the load, distributed cache replacement strategy based on Q-learning, to name a few [3], [4], [5]. It is well known that small-cell infrastructure with edge computing facility alone cannot support the data demand since the data clogging in the backhaul acts as a bottleneck. A new paradigm to handle this data clogging is through caching in the cellular networks. Caching can reduce the peak traffic by prefetching popular contents into memories at the small-cell Base Stations (sBSs) or the end users [6], [7], [8], [9]. Past works in caching include the classical work from the point-of-view of information theory by Niesen et al. [10] (also, see [11]), combinatorial optimization approach [12], energy efficient caching of files in a Device-to-Device (D2D) network (see [12] - [14]), and proactive caching strategy, as in [15]. In [16], the authors employ Unmanned aerial vehicle (UAV) as aerial BSs and jointly optimize UAV trajectory and time scheduling to guarantee the secure transmission in UAV-relaying systems with local caching. Similarly, in [17], an efficient vehicular task offloading via heat-aware mobile edge computing cooperation is proposed.

One of the key problems to be addressed in caching is that of estimating/predicting the popularity profile or demands for the files. Majority of the existing work assume static demands, and hence algorithms are designed to get a good estimate of the popularity profile (see [18]-[22]). On the other hand, estimating the popularity profile based on the data assumes a naive estimate, i.e., a simple averaging, which may not perform well in highly non-stationary environments. However, the demands in reality are non-stationary, and perhaps correlated across time; this makes the algorithms designed for static demands/popularity profiles to underperform. One solution to solve this issue is to consider online learning algorithm to proactively cache the contents [15]. The authors in [23] showed that a good hit rate under non-stationary demands can be achieved through a Time to Live (TTL) based algorithm. Some past work assumed that there is a stationary caching policy such as Least Recently Used (LRU) [24], climb [25], [26], and k-LRU [27] and have characterized the learning errors as a function of time. The learning error depends on the stationary distribution, which in turn depends on the mixing time [28]. Many of these works result in a regret of $\Omega(T)$. In [29], the authors propose a collaborative caching optimization problem in a stationary environment to minimize the accumulated transmission delay over a finite time horizon in cache-enabled wireless networks in a multi-agent multi-armed bandit (MAMB) perspective, which results in a regret of $\mathcal{O}(\log T)$. In [30], function approximation based reinforcement learning approaches are proposed for a massive multi-input multi-output (mMIMO) based network. In [31] integrated access and backhaul (IAB) heterogeneous network for cache-enabled in-band full-duplex in the millimeter wave band is developed. Moreover, in order to use spatio-temporal information, the

*S. Krishnendu and Vimal Bhatia are with Indian Institute of Technology Indore, India, e-mail: {phd1701102001,vbhatia}@iiti.ac.in. B. N. Bharath is with Indian Institute of Technology Dharwad, India, e-mail: bharathbn@iitdh.ac.in. Navneet Garg and Tharmalingam Ratnarajah are with Institute for Digital Communications, The University of Edinburgh, Edinburgh, U.K. e-mail: {ngarg,T.Ratnarajah}@ed.ac.uk.

†This work was partially supported by the MeitY Visvesvaraya PhD Scheme, UK-India Education and Research Initiative Thematic Partnerships under grants DST-UKIERI-2016-17-0060, DST/INT/UK/P-129/2016, and SPARC/2018- 2019/P148/SL.

‡The work of T. Ratnarajah was supported by the U.K. Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/P009549/1.

authors in [32] developed a Bayesian dynamical model to predict the popularity and minimized the cost for transferring data among the sBSs in the network. On the other hand, in [33], active learning approach is used to learn the content popularities to design an accurate content request prediction model. In [34], the authors propose joint caching and the dynamic multicast scheduling to increase the robustness of wireless transmission.

The approach taken so far is either online learning in the adversarial setting leading to regret minimization or by designing caching strategies by estimating the popularity profile (see [35]). The disadvantage in the adversarial setting is that the statistical pattern in the data is completely ignored. An improvement on this is to account for statistical pattern and combine the strategies in a systematic way, this is termed as online-to-batch conversion in the literature [36]. There are several heuristics such as LRU, Least Frequently Used (LFU) and Least Recently Frequently Used (LRFU) (and its variants) which tend to work well in a non-stationary environment. However, these lack theoretical guarantees when the demand statistics are non-stationary with correlation across requests. The LRFU algorithm combines both LRU and LFU. In this algorithm, each file has combined recency and frequency count which is updated during each reference [37].

One of the most promising distributed learning algorithms is the emerging federated learning (FL) framework (see [38], [39], [40], [41]). In FL framework (a special case of distributed optimization), each node uses its own data to compute, say, a strategy, and sends the result to its neighbours or a central node. Thus, in FL, wireless devices can cooperatively execute a learning task by only uploading local learning models to the central BS or its neighbours instead of sharing the entirety of their training data. A framework similar to FL is adapted in this paper to solve the caching problem, where an objective called cache miss (hit) is minimized (maximized) by combining caching strategies obtained using local data from each node. In general, this objective needs to be optimized with respect to a general caching strategy [40], [42], [43], [44]. This leads to a complex online functional optimization problem, which is mathematically intractable and may lead to more complex algorithms. Therefore, before attempting to solve a general problem, it is natural to make suitable assumptions on the structure of the caching strategy that leads to a tractable problem, and provides insights on the general setting as well. The motivation for using a linear combination of caching strategies comes from online learning literature with independent and identically distributed (iid) data [45], [46]. It is shown that the technique called *online-to-batch conversion*, where the current strategy is the average of the past strategies (assumes convexity of the strategy set) results in a regret of the order of $1/T$, where T denotes the time slot. In contrast to $1/T$ regret, online learning algorithms often adopt an adversarial model, and obtain a convergence rate of $O(1/\sqrt{T})$ [47], [48]. Although, the references above were in the general context, it can be easily extended to the caching problem. A practical extension of the above to non-iid correlated requests is to use weighted average of the past strategies, and optimize the weights. An approach similar to this has been taken in the context of prediction problems in [49]. Towards filling shortcoming in the existing work, in this paper, a systematic approach driven by theory to designing caching strategy when the demands/requests are highly *non-stationary* is addressed. Further, the mathematical framework developed in this paper are used to provide guarantees for LRFU, and its variants under non-stationary and correlated demands.

In this paper, the problem of FL based caching across multiple sBSs with correlated demands across time as well as sBSs is considered. Since the demands can be correlated, a conditional average of the cache hit is considered as a metric to design caching strategies. Here, conditioning is with respect to the "local" data available at the sBS. Following are the main contributions of this paper:

- In this paper, assuming structured cache placement, high probability bounds on the conditional average cache hits are derived using Martingale difference equation [50]. In particular, it is assumed that the caching strategy at a given time is a linear combination of past caching decisions across time and across other sBSs in the given region. Insights provided by the bound including regret and discrepancy across temporal and spatial cache hits are used to design the iterative federated based caching algorithm, which optimizes the weights of the linear sum. As a corollary of the bound, a guarantee on the performance of the proposed algorithm using equal caching-weights is also obtained.
- Using the mathematical tools developed in the paper, a similar theoretical guarantee on the performance of the LRFU caching strategy under non-stationary demands is derived. Further, in the iid setting, it is shown that the LRFU performs close to the proposed caching strategy, as expected.
- A FL based heuristic caching algorithm motivated by a well-known algorithm called FedProx [51] is developed, where a proximal term is added to the local cache hit maximization problem, which enables to achieve the local policies close to the averaged caching policies across neighbors. In the numerical results, the proposed algorithms (both federated caching and federated caching heuristic algorithm) have been compared with other online learning algorithm such as follow-the-perturbed-leader (FTPL), follow-the-leader and average LFU [52], [53]. The results show that the proposed algorithms significantly outperform the existing algorithms as well as the equal weight algorithms.

II. System Model

A cellular network consisting of M sBSs denoted by the set $\mathbb{B}$, and users denoted by the set $\mathbb{U}$, as shown in Fig. 1 is considered. Each sBS is assumed to have a limited computation facility and a cache memory of size C bits to store popular contents. This computation capability facilitates distributed caching decisions to be taken at individual sBS without leveraging heavily on the central computing facility such as cloud service, thus saving tremendously on communication and computation costs. Further, it is assumed

that the sBSs can communicate with each other through a limited capacity link. For example, the neighboring sBSs can share limited information such as caching decisions, popular demands and its trends amongst each other. Note that this edge computing paradigm with communication links between sBSs encompasses the proposed Fog network architecture [2]. We assume a time slotted system, where in each slot a user requests contents from the content library $\mathcal{F}$ having N contents, i.e., $|\mathcal{F}| = N$. The demand for the content $f \in \mathcal{F}$ by the user u in the slot t is denoted by $d_{f,u}(t)$. The requests across time slots and sBSs can be correlated with an arbitrary distribution. Since in a practical content library, the files are of different sizes, hence the same is assumed in this work (see the next subsection).

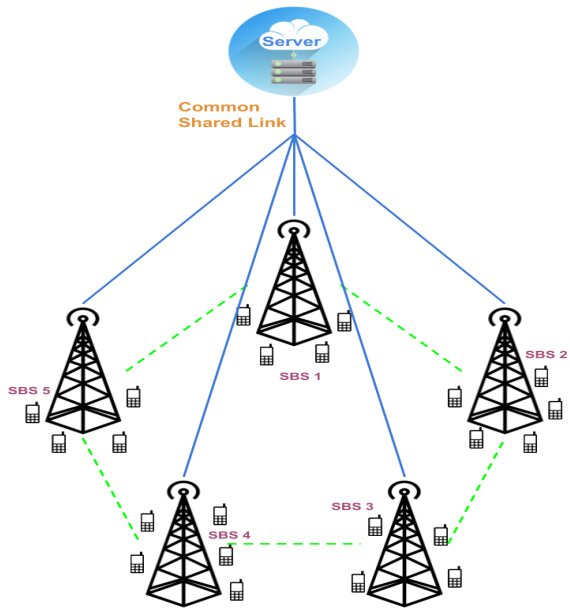

Fig. 1: System model showing multiple sBSs connected to users with limited cache memory.

In the standard cellular network setting without caching, the requested file is served by the sBS to which the user is associated by fetching the content from the server through backhaul and front-haul links of the network. Note that in the current implementation, each user is associated with a single sBS based on the signal-to-interference-plus-noise ratio (SINR) criterion. Keeping minimal changes to the current design, it is assumed that the scheduler associates a user to a sBS based on the SINR criterion. Let the set of users associated to the sBS b in the time slot t be denoted by $\mathbb{U}_b(t)$. The total demand for the file f at the sBS b in the time slot t is given by $D_{f,b}(t) = \sum_{u \in \mathbb{U}_b(t)} d_{f,u}(t)$. Let the data available at the sBS b at time T be denoted by $Z_{b,1}^T \subseteq \mathcal{Z}_{b,1}^T$, which includes demands of sBS b until time slot T, and the data shared by the neighboring sBSs. Here, $\mathcal{Z}_{b,1}^t$ denotes the set of all possible demands and caching strategy of the neighboring sBSs at the end of time slot t. The exact data that the neighboring sBSs provide will be explained in the later part of this paper. Further, $Z_{G,1}^T := \bigcup_{b \in \mathbb{B}} Z_{b,1}^T$ denotes the global data till time T. The following subsections describe the caching strategy employed, and the corresponding metric used to find the optimal strategy.

A. Caching Policy

At each SBS b, the cache placement is assumed to happen at the end of every time slot. In this paper, a FL based caching policy is considered, i.e., at the end of time slot $t-1$ for each file f, the caching policy for the next time slot is given by $\pi_{b,f,t} : \mathcal{Z}_{b,1}^{t-1} \to \mathcal{C}_{b,f}$, where $\mathcal{C}_{b,f}$ is the fraction of the file f stored at bth sBS. Thus, the overall caching policy is defined as $\boldsymbol{\pi}_{b,t} := \times_{f=1}^N \pi_{b,f,t} : \mathcal{Z}_{b,1}^{t-1} \to \times_{f=1}^N \mathcal{C}_{b,f}$. The choice of $\mathcal{C}_{b,f}$ depends on the type of caching employed. Here, online FL based caching is employed, as explained below:

- `Online FL based caching`: In a typical online caching scheme, an original file f of size K_f bits is mapped into S_f sub-packets of size l bits each in such a way that if a user recovers any L_f out of S_f sub-packets, it can recover the whole file. This gives the flexibility to store L_f or less number of packets at each sBS, and the remaining packets can be fetched from the server. For the sake of simplicity in notation, L_f is used to represent the number of packets instead of the size of the file in bits. Although storing any fraction is not possible, choosing $\mathcal{C}_{b,f} = [0,1]$ is a good approximation when the number of sub-packets, i.e, L_f is large. Note that the caching strategy $\boldsymbol{\pi}_{b,t}$ is a vector of dimension N. Since the cache size is limited to C bits, it imposes the constraint that $\sum_f \pi_{b,f,t} L_f l \leq C$. Here, $L_f l$ is the total number of bits that needs to be recovered under the caching scheme, and $\pi_{b,f,t}$ is the fraction of the packets stored.

The following subsection presents the problem of FL based caching addressed in this paper.

B. Problem Statement

In caching scheme, the "amount" of requests that are present in the caches of sBSs to which the users are connected is a good measure of performance; this is termed as *hit rate*. In view of this, the hit rate at the sBS b is given by

$$\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) := \sum_{f=1}^{N} \sum_{u \in \mathbb{U}_b(t)} d_{f,u}(t) \pi_{b,f} \mathcal{L}_f. \tag{1}$$

The above corresponds to the *instantaneous* hit rate at the sBS b in the time slot t when caching strategy $\pi_{b,f} \in [0,1]$ is employed with $\mathcal{L}_f := L_f l$. Note that the factor l does not impact the structure of the solution, and hence omitted from the definition of the hit rate. Since the hit rate is random, a widely used measure of performance is the average cache hit, i.e., $\sum_b \mathbb{E}\{\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t})\}$ [1], where the average is with respect to the global demands.[2] However, at time t, the sBS b will have access to its "local" data $Z_{b,1}^{t-1}$, and hence, conditional mean is the appropriate metric, i.e., $\sum_b \mathbb{E}\{\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\}$, where

[1] $\mathbb{E}[\cdot]$ represents the statistical expectation operator.

[2] Note that the demands across sBSs as well as time slots are correlated. Hence, the expectation should be with respect to all the total randomness.

the expectation is conditioned on the local demands, i.e., $Z_{b,1}^{T}$. Thus, the following problem needs to be solved

$$\max_{\boldsymbol{\pi}_{b,t}} \sum_{b} \mathbb{E}\{\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\}$$
$$\text{subject to} \sum_{f} \pi_{b,f,t}\mathcal{L}_f \leq C. \tag{2}$$

Let the set of all caching strategies be denoted by $\mathcal{C} := \{\pi_{b,f} : \pi_{b,f} \geq 0, \sum_f \pi_{b,f}\mathcal{L}_f \leq C\}$. The above is similar to the formulation considered in the prediction problems [49]. Unfortunately, in the real world scenario, the conditional expectation is difficult to compute, and hence the above problem cannot be solved. One possible approach could be to estimate the conditional expectation, and use it as a proxy in the above problem. Since the user demands arrive in real-time, this estimate could be updated online. However, in this paper, instead of updating the estimates online, the solution for caching problem will be obtained online using the available "local" data. In the following section, solution to the above problem for online FL based caching scenarios is presented.

III. Online Federated Learning based Caching

Towards addressing the problem, a few structural assumptions are made on the caching strategy employed. In a typical online learning with adversarial framework, a natural metric to consider is the "regret". In the present setting, the demands are random in nature and this corresponds to a stochastic setting rather than an adversarial setting, i.e., the nature reacts in a random fashion rather than an adversarial fashion. A well known strategy to handle this is through online-to-batch conversion [54], which is as follows: (i) at time slot t, solve the regret minimization problem to get a sequence of caching strategies, and (ii) use the average of these caching strategies at time t. This has the advantage of providing $\mathcal{O}(\frac{1}{T})$ regret when the problem is stochastic. The model considered in this paper has added complexity that the demands of any sBS b across time slots can be correlated. Further, it can also be correlated with the demands of other sBSs. In this scenario, a natural extension of online-to-batch conversion is to take the average of regret minimizing caching strategies across time as well as the sBSs [36], [49]. Towards this, consider the following weighted average of a sequence of caching strategies $\boldsymbol{\pi}_{b,t}$ from time slot $t = T-\tau+1$ to T given by

$$\bar{\boldsymbol{\pi}}_{b,T+1} := \sum_{t=T-\tau+1}^{T} \alpha_{b,t}\boldsymbol{\pi}_{b,t}, \tag{3}$$

where $\alpha_{b,t}$'s are the non-negative weights that satisfy $\sum_{t=T-\tau+1}^{T} \alpha_{b,t} = 1$. The symbol $\boldsymbol{\alpha}_{b,T} := (\alpha_{b,T-\tau+1}, \ldots, \alpha_{b,t})$ is used to denote vector of weights corresponding to the sBS b from time slot $T-\tau+1$ to T. The caching strategy has been taken as a weighted linear combination of all the neighboring SBSs caching strategies. It is important to note that the average of caching strategy across time is also a valid caching strategy, i.e., the set of all caching strategies $\mathcal{C}$ is a convex set. Since the demands are correlated across sBSs, a natural way to construct the caching strategy for the time slot $T+1$ is as follows

$$\boldsymbol{\pi}_{b,T+1}^{(av)} := w_b^{T+1}\bar{\boldsymbol{\pi}}_{b,T+1} + \sum_{b' \in \mathcal{N}_b} w_{j_b(b')}^{T+1}\bar{\boldsymbol{\pi}}_{b',T+1}, \tag{4}$$

where the map $j_b : \mathcal{N}_b \to \{1, 2, \ldots, |\mathcal{N}_b|\}$, $|\mathcal{N}_b|$ denotes the set of neighboring sBSs to which it is connected, and the weights are chosen to be non-negative with the constraint given by $\sum_{b' \in \mathcal{N}_b} w_{j_b(b')}^{T+1} + w_b^{T+1} = 1 \; \forall$ sBS b. Now, the problem is to choose weights in such a way that the average cache hit is maximized. One can expect that in order to solve this problem, any sBS $b \in \mathbb{B}$ at the end of time slot T should have access to neighboring sBSs' data. In this paper, a formal approach to answer the above is detailed. Obviously the choice of the weights $w_{j_b(b')}^{T+1}$ as well as $\alpha_{b,t}$ depend on how relevant (i) is its past caching decisions to the current demands, and (ii) caching decisions of neighboring sBSs are to the sBS b. These are captured through the following notions of mismatch and regret.

Definition (Mismatch): The mismatch between a sBS b and its neighbor with weights $w_{j_b(b')}^{T+1}$, $b' \in \mathcal{N}_b$ is given by

$$\mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) := \sum_{b' \in \mathcal{N}_b} w_{j_b(b')}^{T+1}\Delta_{b,T+1}(\boldsymbol{\alpha}_{b,T+1}, \boldsymbol{\alpha}_{b',T+1}), \tag{5}$$

where the weight vector $\boldsymbol{w}_{\neq b,T} := (w_{j_b(b')}^{T+1} : b' \in \mathcal{N}_b)$, and

$$\begin{aligned}\Delta_{b,T+1}(\boldsymbol{\alpha}_{b,T+1}, \boldsymbol{\alpha}_{b',T+1}) \;&:=\; \mathbb{E}\{\mathcal{R}_{b,T+1}(\bar{\boldsymbol{\pi}}_{b',T+1}) \mid Z_{b,1}^{T}\} \\ &-\; \mathbb{E}\{\mathcal{R}_{b,T+1}(\bar{\boldsymbol{\pi}}_{b,T+1}) \mid Z_{b,1}^{T}\}.\end{aligned}$$

The above captures mismatch or discrepancy across sBSs, which will help us in determining the relevance of the neighboring sBSs' decisions. If the mismatch is small for a sBS b essentially means that the neighboring sBSs strategy performs well on the sBS b. Similarly, to determine the relevant caching strategies across time to the current time slot, and to measure the performance, the two key metrics are discrepancy across time and the regret, which are defined as follows.

Definition (Discrepancy): Given the local information at the sBS b with caching strategies $\boldsymbol{\pi}_{b,t}$ for $b \in \mathbb{B}$, $t = T-\tau+1, \ldots, T$, the discrepancy at the end of time slot T is defined by

$$\mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T}) := \sup_{\boldsymbol{\pi}_{b,t}: t=T-\tau+1,\ldots,T} \left| \sum_{t=T-\tau+1}^{T} \alpha_{b,t}\Delta\bar{\mathcal{R}}_{T,t}(\boldsymbol{\pi}_{b,t}) \right|. \tag{6}$$

where $\Delta\bar{\mathcal{R}}_{T,t}(\boldsymbol{\pi}_{b,t}) := \mathbb{E}\{\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{T}\} - \mathbb{E}\{\mathcal{R}_{b,t+1}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t}\}$.

Definition (Regret): The regret at the sBS b at time T with respect to a sequence of strategy $\boldsymbol{\pi}_{b,t}$ is defined as

$$\begin{aligned}\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t}) \;&:=\; \sup_{\boldsymbol{\pi}_{b,t}^{*}} \sum_{t=T-\tau+1}^{T} \mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}^{*}) \\ &-\; \sum_{t=T-\tau+1}^{T} \mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}).\end{aligned} \tag{7}$$

The following theorem gives guarantees for the proposed caching strategy, and also provides insights on how to choose the weights, and the sequence of caching policies across time. The main result of the paper is stated below, and the corresponding proof is presented in Sec. VI.

Theorem 3.1: Given weights and a sequence of caching strategies as in (4) that is adapted to $Z_{b,1}^T$, with a probability of at least $1-\delta$, $\delta > 0$, the following two bounds hold:

$$\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{\boldsymbol{b,T+1}}^{(av)}) \mid Z_{b,1}^T\right] \geq \sum_{t=T-\tau+1}^{T} \alpha_{b,t}\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) - \mathcal{E}_{b,T}^{(1)}, \quad (8)$$

where

$$\mathcal{E}_{b,T}^{(1)} := H_{\max}\|\boldsymbol{\alpha}_{b,T}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}} + \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) + \mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T}),$$

$H_{\max}$ is the maximum cache hit, and

$$\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{\boldsymbol{b,T+1}}^{(av)}) \mid Z_{b,1}^T\right] \geq \sup_{\boldsymbol{\pi}_{b,t}} \sum_{t=T-\tau}^{T} \alpha_{b,t}\bar{\mathcal{R}}_{t,T}(\boldsymbol{\pi}_{b,t}) - \mathcal{E}_{r,T}^{(2)}. \quad (9)$$

for any $\gamma > 0$. In the above, $\bar{\mathcal{R}}_{t,T}(\boldsymbol{\pi}_{b,t}) := \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^T\right]$, and

$$\begin{aligned}\mathcal{E}_{b,T}^{(2)} &:= 2H_{\max}\|\boldsymbol{\alpha}_{b,T}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}} + \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) \\ &+ \frac{2\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau} + H_{\max}\sum_{t=T-\tau+1}^{T}\left|\alpha_{b,t} - \frac{1}{\tau}\right| \\ &+ 2\mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b.T}) + \gamma. \end{aligned} \quad (10)$$

An important special case of the above result is when uniform caching strategy is used, i.e., $\alpha_{b,t} = 1/\tau \ \forall\ t$, which is presented as a corollary.

Corollary 3.2: Given equal weights, i.e., $\alpha_{b,t} = 1/\tau \ \forall\, t$, and a sequence of caching strategies as in (4) that is adapted to $Z_{b,1}^T$, with a probability of at least $1-\delta$, $\delta > 0$, the following two bounds hold:

$$\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{\boldsymbol{b,T+1}}^{(av)}) \mid Z_{b,1}^T\right] \geq \frac{1}{\tau}\sum_{t=T-\tau+1}^{T} \mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) - \mathcal{E}_{b,T}^{(1)}, \quad (11)$$

where

$$\mathcal{E}_{b,T}^{(1)} := \frac{H_{\max}}{\tau}\sqrt{2\log\frac{1}{\delta}} + \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) + \mathbb{D}_{b,T}(\boldsymbol{u}_\tau),$$

$H_{\max}$ is the maximum cache hit, and

$$\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{\boldsymbol{b,T+1}}^{(av)}) \mid Z_{b,1}^T\right] \geq \sup_{\boldsymbol{\pi}_{b,t}} \frac{1}{\tau}\sum_{t=T-\tau}^{T} \bar{\mathcal{R}}_{t,T}(\boldsymbol{\pi}_{b,t}) - \mathcal{E}_{r,T}^{(2)}. \quad (12)$$

for any $\gamma > 0$. In the above, $\boldsymbol{u}_\tau := (\frac{1}{\tau}, \frac{1}{\tau}, \ldots, \frac{1}{\tau}) \in \mathbb{R}^{1\times\tau}$, $\bar{\mathcal{R}}_{t,T} := \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^T\right]$, and $\mathcal{E}_{r,T}^{(2)} := \frac{2H_{\max}}{\tau}\sqrt{2\log\frac{1}{\delta}} + \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) + \frac{2\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau} + 2\mathbb{D}_{b,T}(\boldsymbol{u}_\tau) + \gamma$.

A few observations are in order with reference to Theorem 3.1. The term $H_{\max}\sum_{t=T-\tau}^{T}\left|\alpha_{b,t} - \frac{1}{\tau}\right|$ in the second bound suggests that all the weights should be close to $1/\tau$, i.e., uniform weights. On the other hand, both the bounds also suggest that the discrepancies should be made low by choosing the weights appropriately. This requires non-uniform weights in general, and since the two tasks are conflicting, a nice balance needs to be maintained by properly choosing the weights. Further, it is clear from the second bound that the caching policy should be chosen in such a way that the regret is minimized. The following subsection presents a systematic approach to find an online distributed caching algorithm.

A. Algorithm for Federated Learning based Caching

In this subsection, the insights provided by the theory are used to propose an algorithm for FL based online caching. The main result states that upon using the caching strategy given in (4), the resulting cache hit is lower bounded by the expression in (9) with high probability. Now, at time slot $T+1$, the goal is to choose the individual strategy $\boldsymbol{\pi}_{b,t}$ to construct $\boldsymbol{\pi}_{b,T+1}^{(av)}$ as in (4) such that the right hand side of (9) consisting of regret and discrepancy terms to be maximized.[3] In particular, this can be done by using the following two steps: (i) choose the sequence $\boldsymbol{\pi}_{b,t}$ in such a way that minimizes the regret term, and (ii) minimize the mismatch terms $\mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T})$ and $\mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T})$ to get the optimal weights, which can be used to combine the caching sequence as in (4). The first step would be to find the regret minimizing caching strategy by solving the following optimization problem

$$\min_{\boldsymbol{\pi}_{b,t}:\mathbf{1}^T\boldsymbol{\pi}_{b,t}\leq C}\left[\sup_{\boldsymbol{\pi}_{b,t}^*}\sum_{t=T-\tau+1}^{T}\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}^*) - \sum_{t=T-\tau+1}^{T}\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t})\right] \quad (13)$$

to get a sequence of caching policies denoted by $\boldsymbol{\pi}_{b,t}^R \ \forall\, t$. Note that the above problem can be solved optimally at the end of time slot T as each sBS has access to the demands until time slot T. In the next step, we maximize the right hand side of (9) excluding the regret term. Unfortunately, the discrepancy term is unknown, and hence is estimated using the demands. Moreover, the discrepancy term involves an optimization. One way to deal with this is to use the regret minimizing caching strategy, and solve the following optimization problem to obtain the weights

$$\sup_{\alpha_{b,t},\boldsymbol{w}_{\neq b,T}} \sum_{t=T-\tau+1}^{T}\alpha_{b,t}\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}^R) - a\widehat{\mathbb{D}}_{b,T}(\boldsymbol{\alpha}_b) - b\widehat{\mathtt{M}}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) + \lambda\sum_{t=T-\tau+1}^{T}\left|\alpha_{b,t} - \frac{1}{\tau}\right| \quad (14)$$

[3]The regret and discrepancy have negative signs on the right hand side.

for some $\lambda > 0$, and $\widehat{\mathbb{D}}_{b,T}(\boldsymbol{\alpha}_b)$ is an estimate of the discrepancy given by

$$\widehat{\mathbb{D}}_{b,T}(\boldsymbol{\alpha}_b) := \sup_{\boldsymbol{\pi}_{b,t}:\boldsymbol{\pi}_{b,t}\mathbf{1}^T x \leq C} \left| \sum_{t=T-\tau+1}^{T} \alpha_{b,t} \sum_f \psi^{(f)}_{\tau_1,\tau_2}(t,T)\pi_{b,f}(t) \right|, \quad (15)$$

where $\psi^{(f)}_{\tau_1,\tau_2}(t,T) := \mathcal{L}_f\left(\frac{1}{\tau_1}\sum_{l=T-\tau_1+1}^{T}\phi_{b,f}(l) - \frac{1}{\tau_2}\sum_{l=t-\tau_2+1}^{t-1}\phi_{b,f}(l)\right)$, and the sum demand $\Phi_{b,f}(t) := \sum_{u\in\mathbb{U}_b(t)} d_{f,u}(t)$. The constants a and b are fine tuned to get better results. An estimate of the discrepancy across sBSs is given by

$$\widehat{\mathtt{M}}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) := \frac{1}{\tau}\sum_{b'\in\mathcal{N}_b} w^{T+1}_{j_b(b')}\left[\sum_{s,l=T-\tau+1}^{T}\alpha_{b,s}\mathcal{R}_{b,l}(\pi^R_{b,s}) - \sum_{s,l=T-\tau+1}^{T}\alpha_{b',s}\mathcal{R}_{b',l}(\pi^R_{b',s})\right]. \quad (16)$$

Note that the conditional expectations are replaced by the time average of the cache hit as a proxy to get the above estimate of the discrepancy. In the time slot T, the average cache hit from the time slot $T-\tau+1$ to T is used as a proxy for the conditional mean in the expression for $\mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T})$. Although the objective in (14) seems to be simple, it is a non-convex function of $\alpha_{b,t}$ and $\boldsymbol{w}_{\neq b,T}$, making the problem difficult to solve for global optima. However, a simple gradient descent algorithm can be used to achieve a local optima. Using the gradient descent approach leads to **Algorithm** 1, which is explained next. Note that the estimate of discrepancy above involves solving an optimization problem with respect to the caching strategy $\boldsymbol{\pi}_{b,t}$. However, this optimization problem depends on $\alpha_{b,t}$, which is unknown. A natural approach to this is to assume some initial $\alpha_{b,t}$, and solving the above optimization problem using gradient descent step, and project to satisfy the cache constraint. This is done in steps 1 and 2 of the **Subroutine**. Using this, in the step $k+1$, an update $\pi^t_{b,f}(k+1)$ is obtained. This is used in the expression for an estimate of the discrepancy in (15), and used in (14) to subsequently solve for weights $\alpha_{b,t}$ and $w^{T+1}_{j_b(b')}$. This is done by taking a gradient descent step with respect to $\alpha_{b,t}$ in the problem in (14) followed by projection to satisfy the constraint $\sum_{t=T-\tau+1}^{T}\alpha_{b,t} = 1$. These two steps correspond to steps 3 and 4 of the **Subroutine**. Similar gradient steps are taken for the weights $w^{T+1}_{j_b(b')}$. These steps correspond to steps 5 and 6 of the **Subroutine**. The details are provided in the algorithm below, and explained later in this section.

Algorithm 1 Algorithm for Federated Learning based Caching

1: **for** $T = 1, 2, \ldots,$ and sBS $b \in \mathbb{B}$ **do**
2: Run regret minimization as in (13) to get a sequence
3: $\boldsymbol{\pi}^R_{b,t}$, $t = 1, \ldots, T$
4: Call **Subroutine** (T, τ, $\boldsymbol{\pi}^R_{b,t}$, $\boldsymbol{\pi}^R_{b',t}$ **for all** $b' \in \mathcal{N}_b$) .
5: **to get** $\boldsymbol{\pi}_{b,T+1}$
6: **end for**

The stopping criterion of the algorithm in the **Subroutine** is determined by checking if the difference in weights is smaller than a threshold, which is chosen based on extensive simulations. The learning rate η_k, β_k, and γ_k are chosen such that it decays as $1/\sqrt{k}$ with the iteration k.

Subroutine (T, τ, $\boldsymbol{\pi}^R_{b,t}$, $\boldsymbol{\pi}^R_{b',t}$ **for all** $b' \in \mathcal{N}_b$):

- **for** each sBS b, **for** $k = 0, 1, 2, \ldots$ **do**
  1) **If** ($k = 0$), then `initialize` $\pi^{(0)}_{b,f} = \frac{C}{\sum_f \mathcal{L}_f}$, $\forall\, f$ and $\alpha^{(0)}_{b,t} = 1/\tau, \forall t \geq T-\tau+1$, and zero otherwise. Let $\Gamma_{t,T} := \sum_f \pi^t_{b,f}(k)\Psi^f_{\tau_1,\tau_2}(t,T)$. For $k \neq 0$, `update`

$$\pi^t_{b,f}(k+1) = \pi^t_{b,f}(k) + 2\eta_k g, \quad (17)$$

  where $g := \alpha_{b,t}(k)\Psi^f_{\tau_1,\tau_2}(t,T)$ if $\sum_{t=T-\tau+1}^{T}\Gamma_{t,T} > 0$, else choose $g := -\alpha_{b,t}(k)\Psi^f_{\tau_1,\tau_2}(t,T)$.
  2) `Project:` $\pi_{b,f}(k+1) \leftarrow \max\{\pi_{b,f}(k+1), 0\}$ and $\pi^{(k+1)}_{b,f} \leftarrow \frac{C\pi_{b,f}(k+1)}{\sum_f \pi_{b,f}(k+1)\mathcal{L}_f}$.
  3) `Update the` α`-weights:`

$$\alpha_{b,t}(k+1) = \alpha_{b,t}(k) + \beta_k\Big[\mathcal{R}_{b,t}(\pi^R_{b,t}) - \Theta - \nabla_\alpha \widehat{\mathtt{M}}_{b,T+1}(\boldsymbol{w}_{\neq b,T})\Big], \quad (18)$$

  where β_k is the step size, $\Gamma_{t,T}$ is as defined in step 1 above, $\Theta := 2\max\{\Gamma_{t,T}, -\Gamma_{t,T}\} - \lambda\nabla_\alpha\|\alpha_{b,t}(k) - u\|_1$,

$$\nabla_\alpha \widehat{\mathtt{M}}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) := \frac{2}{\tau}\sum_{b'\in\mathcal{N}_b} w^{T+1}_{j_b(b')}(k)\sum_{l=T-\tau+1}^{T}\mathcal{R}_{b,l}(\pi^R_{b,t}),$$

  and $\nabla_\alpha\|\alpha_{b,t}(k) - u\|_1 := \mathbf{1}\{\alpha_{b,t} < \frac{1}{\tau}\} - \mathbf{1}\{\alpha_{b,t} \geq \frac{1}{\tau}\}$.
  4) `Project:` $\alpha_{b,t}(k+1) \leftarrow \max\{\alpha_{b,t}(k+1), 0\}$, and $\alpha_{b,t}(k+1) \leftarrow \frac{\alpha_{b,t}(k+1)}{\sum_{t=T-\tau+1}^{T}\alpha_{b,t}(k+1)}$.
  5) `Update the` $\boldsymbol{w}$`-weights` for sBS b using data from neighboring sBSs as follows:

$$w^{T+1}_{j_b(b')}(k+1) = w^{T+1}_{j_b(b')}(k) - \frac{2\gamma_k}{\tau}\left[\sum_{s,l=T-\tau+1}^{T}\alpha_{b,s}(k) \times \mathcal{R}_{b,l}(\pi^R_{b,s}) - \sum_{s,l=T-\tau+1}^{T}\alpha_{b',s}(k)\mathcal{R}_{b,l}(\pi^R_{b',s})\right] \quad (19)$$

  6) `Project:` $w^{T+1}_{j_b(b')}(k+1) \leftarrow \max\{w^{T+1}_{j_b(b')}(k+1), 0\}$, and
     - `Normalize:` If $\sum_{b'\in\mathcal{N}_b} w^{T+1}_{j_b(b')}(k+1) < 1$, then $w^{T+1}_b = 1 - \sum_{b'\in\mathcal{N}_b} w^{T+1}_{j_b(b')}(k+1)$, else $w^{T+1}_b = 0$ and for all $b' \in \mathcal{N}_b$, $w^{T+1}_{j(b')}(k+1) = \frac{w^{T+1}_{j_b(b')}(k+1)}{\sum_{b'\in\mathcal{N}_b} w^{T+1}_{j_b(b')}(k+1)}$.

7) **if (not converged):** Broadcast the weights obtained in the current iteration to all neighboring sBSs, and go back to step 1 **else;** return

$$\bar{\boldsymbol{\pi}}_{b,T+1} = w_b^{T+1}(k+1) \sum_{t=T-\tau+1}^{T} \alpha_{b,t}(k+1)\boldsymbol{\pi}_{b,t}^R + \sum_{b' \in \mathcal{N}_b} w_{j_b(b')}^{T+1}(k+1) \sum_{t=T-\tau+1}^{T} \alpha_{b',t}(k+1)\boldsymbol{\pi}_{b',t}^R, \quad (20)$$

- **end for**

Since the above algorithm is a modification of gradient descent algorithm,[4] the convergence can be proved in a similar manner to that of classical gradient descent. The proof is omitted due to lack of space. In the following subsection, the proposed FL based heuristics algorithm for caching mechanism design that takes into account neighboring SBSs requests is detailed.

B. Federated Learning Based Heuristics Caching Mechanism

In the single SBS scenario, a natural approach to find a caching strategy is to solve the following optimization problem:

$$\min_{\pi: \sum_f \pi_f l_f \leq L} \hat{F}_k(\pi), \quad (21)$$

where $\hat{F}_k(\pi) := \sum_{f \in \mathcal{F}} (1-\pi_f) l_f \hat{d}_{f,k}^{(t)}$ is an estimate of the average cache miss, and $\hat{d}_{f,k}^{(t)} := \frac{1}{\tau}\sum_{s=t-\tau}^{t-1} d_{f,k}^{(s)}$. However, if the amount of data available is less, the estimate will be poor, and hence results in a poor caching strategy. One way to overcome this is to use the information available from the neighboring sBSs. This can be done by penalizing the caching strategies that are far from some average of the caching strategies of the neighboring sBSs, i.e., $\lambda \|\boldsymbol{\pi} - \bar{\boldsymbol{\pi}}_{\mathcal{N}_k}^{(t)}\|^2$, where $\bar{\boldsymbol{\pi}}_{\mathcal{N}_k}^{(t)}$ is the average of neighboring SBSs caching strategies. This requires information about past caching strategies from the neighboring sBSs, which is assumed to be available. The parameter $\lambda > 0$ controls the amount of deviation that can be tolerated. More details of the heuristic algorithm are provided in **Algorithm** 2. The following subsection presents an analysis of the LRFU scheme. To the best of authors knowledge, this analysis is the first of its kind in the literature.

LRFU Caching Policy: Analysis and Guarantees: In this scheme, an average of the past demands of each file is listed in the decreasing order, and the first k files are stored, where k is chosen in such a way that the cache size constraint is satisfied. In particular, in time slot t, at sBSS b, the following optimization problem is solved:

$$\max_{\boldsymbol{\pi}: \sum_f \pi_f \mathcal{L}_f \leq C} \sum_f \pi_f \hat{d}_{b,f,t} \mathcal{L}_f, \quad (24)$$

where $\hat{d}_{b,f,t} := \frac{1}{\tau}\sum_{s=t-\tau-1}^{t-1} d_{b,f,s} \; \forall f$. In the case of constant file sizes, i.e., $\mathcal{L}_f := L \; \forall f$, the solution to the above amounts to listing the files in the decreasing order of $\hat{d}_{b,f,t}$, and storing the top k files, where k is chosen to satisfy the cache constraint. However, when the files sizes are different, instead of the "average" demands $\hat{d}_{b,f,t}$, one should consider $\mathcal{L}_f \hat{d}_{b,f,t}$ in the above argument. By imposing the constraint $\pi_f \in \{0,1\} \; \forall f$ leads to the classical LRFU solution and the corresponding caching strategy is denoted by $\boldsymbol{\pi}_{b,t}^{\text{LRFU}}$. Before stating the main theorem, the following notions of discrepancy (similar to discrepancy described earlier) will be used to state the main result.

[4]The algorithm deviates from the classical gradient descent in the step 2 of the subroutine as the problem involves two optimization problems.

Algorithm 2 Algorithm for Federated Learning based Heuristics Caching

procedure FEDERATED LEARNING BASED HEURISTICS FOR CACHING
  for $\forall$ sBS $k = 1, \ldots, N$ and $\forall f = 1, \ldots, F$ **do**
    $\hat{d}_{f,k}^{(0)} \leftarrow$ initial demand
    $\pi_{\mathcal{N}_k}^{(0)} \leftarrow$ initial caching vector s.t. $\sum_f \pi_f \mathcal{L}_f \leq C$
  end for
  for $t = 1, 2 \ldots,$ **do**
    sBS k sents $\hat{\pi}_{k,t-1}^*$ to its neighboring sBSs.
    At each sBS k, estimate demand vectors
    and average caching vectors as follows:

$$\hat{d}_{f,k}^{(t)} := \frac{1}{\tau}\sum_{s=t-\tau}^{t-1} d_{f,k}^{(s)}, \text{ and } \bar{\boldsymbol{\pi}}_{\mathcal{N}_k}^{(t)} := \frac{1}{|\mathcal{N}_k|}\sum_{j \in \mathcal{N}_k} \hat{\pi}_{k,t-1}^*. \quad (22)$$

    Solve the following optimization problem to get
    $\hat{\pi}_{k,t}^*$:

$$\hat{\pi}_{k,t}^* := \arg\min_{\boldsymbol{\pi}} \hat{F}_{k,t}(\pi) + \lambda \|\boldsymbol{\pi} - \bar{\boldsymbol{\pi}}_{\mathcal{N}_k}^{(t)}\|^2, \quad (23)$$

    where $\hat{F}_{k,t}(\pi) := \sum_{f \in \mathcal{F}} (1-\pi_{f,k}) \mathcal{L}_f \hat{d}_{f,k}^{(t)}$, and $\lambda > 0$.
    Cache files at sBS k according to $\hat{\boldsymbol{\pi}}_{k,t}^*$, and
    distribute across its neighboring sBSs.
  end for
end procedure

Definition (Discrepancy across time and information): Given local and global information at the sBS b with caching strategies $\boldsymbol{\pi}_{b,t}$ for $b \in \mathbb{B}$, $t = T-\tau+1, \ldots, T$, the corresponding discrepancy between local and global information at the end of time slot T is defined by

$$\mathbb{D}_{GL,T}(\tau) := \sup_{\boldsymbol{\pi}_{b,t}: t=T-\tau+1,\ldots,T} \left| \frac{1}{\tau} \sum_{t=T-\tau+1}^{T} \left(\Delta \bar{\mathcal{R}}_{T,t}\right) \right|, \quad (25)$$

where $\Delta \bar{\mathcal{R}}_{T,t} := \mathbb{E}\{\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^T\} - \mathbb{E}\{\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{G,1}^T\}$.

The above measures the discrepancy between the local and the global data, i.e., the demands at sBS b and all other sBSs. In the iid demands scenario, it is clear that the discrepancy is zero, as expected. In other words, having access to global information is useful to improve the accuracy of the future demand estimate through averaging, and hence the average cache hit as well. The following theorem provides guarantees on the performance of the LRFU scheme in comparison with (2), which assumes perfect knowledge of statistics of the demands. Note that the analysis included in the proof of the

following result does not depend on whether $\pi_f \in \{0,1\}$ or $\pi_f \in [0,1]$. Therefore, this constraint is not explicitly stated.

Theorem 3.3: For the LRFU caching strategy $\boldsymbol{\pi}_{b,t}^{\text{LRFU}}$, with a probability of at least $1-\delta$, $\delta > 0$, the following bound hold:

$$\begin{aligned}\sum_f \boldsymbol{\pi}_{b,t}^{\text{LRFU}} \hat{d}_{b,f,t}\mathcal{L}_f &\leq \sup_{\boldsymbol{\pi_b}} \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z_{b,1}^{t-1}\right] \\ &+ \mathbb{D}_{GL,t}(\boldsymbol{\alpha}_b) + \mathbb{D}_{b,t}(\boldsymbol{u}_\tau) \\ &+ H_{\max}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2\log\frac{1}{\delta}}{\tau}} \quad (26)\end{aligned}$$

where $\mathbb{D}_{b,t}(\boldsymbol{u}_\tau)$ is as defined in (6) with $\boldsymbol{u}_\tau := (\frac{1}{\tau}, \frac{1}{\tau}, \ldots, \frac{1}{\tau})$ is a $1 \times \tau$ vector, and $\mathbb{D}_{GL,t}(\boldsymbol{\alpha}_b)$ is as defined in (25).

Proof: See Appendix VII.

It is clear from the above theorem that in the iid demands scenario, the right hand side will be $\sup_{\boldsymbol{\pi_b}} \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z_{b,1}^{t-1}\right] + H_{\max}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2\log\frac{1}{\delta}}{\tau}}$. It is clear that as $\tau \to \infty$, i.e., using more local data to compute the demand estimate, the metric used in the case of LRFU approaches that of the optimal cache hit in (2). The above result is independent of the demand process, as opposed to the existing work on LRFU, which typically assume iid demands. The following section presents simulation results to validate some of the insights provided by our theory to design online caching algorithm, and compare it with some of the well known algorithms.

1) Complexity Analysis:

- Algorithm I: The function in (2) $\mathcal{O}(d_{\max}^2\tau N)$ multiplications and $\mathcal{O}(d_{\max}\tau N)$ additions, where $d_{\max}$ is the maximum degree of the topology of the network presented earlier, N is the number of files and τ is the time slot. The proposed algorithm involves solving a optimization problem, whose complexity is analyzed under the assumption that the gradient descent method is used to find the optimal point. The gradient descent method involves the following two steps; (a) computing the updated vector, which involves using the current point and move in the direction of the gradient, and (b) projecting the result onto the constraint region. Thus, the total complexity is the number of times step (a) above is required to solve the problem with an accuracy of ζ, i.e., the difference between the solution obtained and the optimal value of the objective function, times the complexity of step (a). Thus, from [55], the number of times step (a) is executed is of the order of $\mathcal{O}(1/\zeta)$. Therefore, the total complexity is $\mathcal{O}(\frac{d_{\max}^2\tau N}{\zeta})$ multiplications and $\mathcal{O}(\frac{d_{\max}\tau N}{\zeta})$ additions.
- Algorithm II : The function in (21) requires $\mathcal{O}(d_{\max}\tau N)$ multiplications and $\mathcal{O}(d_{\max}\tau N)$ additions, where $d_{\max}$ is the maximum degree of the topology of the network presented earlier. Similar to the complexity analysis of Algorithm I, the total complexity of Algorithm II is $\mathcal{O}(\frac{d_{\max}\tau N}{\zeta})$ multiplications and $\mathcal{O}(\frac{d_{\max}\tau N}{\zeta})$ additions. The computational complexity if the LRFU algorithm is $\mathcal{O}(\log_2\tau)$ where τ is the cache size. Comparing this with the LRFU algorithm, it can be observed that both the proposed algorithms performs significantly better than the LRFU method at the expense of a slight increase in the complexity depending on the accuracy attained.

IV. Simulation Results

The simulation setup consists of five sBSs with multiple users connected to each of the sBS as shown in Fig. 1. Without loss of generality, it is assumed that the users can move, and over time connect to different sBSs. The demands from the users are generated using the Movie Lens data set.[5] The total number of files is 800, i.e., the users can possibly request from only these catalog of Movie Lens data. The size of each file is assumed to be chosen uniformly random from 10 to 100 units. The demands at each sBS are obtained by randomly dividing Movie Lens data into 5 disjoint chunks, which are spread across 200 time slots (here one time slot equals one day). Further, the demands are normalized in each slot to get the popularity profile. This is used in place of demands while defining the (weighted cache hit and discrepancy) metric to compute the optimal weights in **Algorithm 1**. The average cache hit with un-normalized demands is used as a performance measure. The optimization is done with respect to the weights across time as well as sBSs. In this section, for simplicity, the weights across time will be referred to as α, and the weights allocated across sBSs as $\boldsymbol{w}$. To understand the importance of past demands and the neighboring sBSs demand, it is important to compare the proposed scheme under various conditions. In particular, the proposed FL based caching algorithm is compared with (i) the FL based heuristic algorithm proposed in Sec. III-B, (ii) the algorithm that uses uniform $\boldsymbol{w}$ and optimal α, (iii) LRFU, (iv) algorithm with uniform α and optimal $\boldsymbol{w}$, and (v) follow-the-leader, (vi) FTPL, and (vii) average LFU. The following parameters were used: $\tau = 10$, $\tau_1 = \tau_2 = 5$, $\eta_k = 1/\sqrt{k}$, $\beta_k = 0.01/\sqrt{k}$, and $\gamma_k = 0.4/\sqrt{k}$, where k is the iteration index in the algorithm. The topology of the sBSs for Fig. 2 shows a plot of cache hit versus cache size for different caching algorithms. The topology of the sBS is described by $1 \leftrightarrow 2 \leftrightarrow 3$, $3 \leftrightarrow 4 \leftrightarrow 5$, and $5 \leftrightarrow 1$, where $a \leftrightarrow b$ indicates that sBSs a and b can communicate with each other. Fig. 2 shows the sum cache hit rate of all the sBSs summarizing the trends in all the sBSs. It is clear from the figure that the proposed algorithm (both proposed FL based caching algorithm and proposed FL based heuristic caching algorithm) performs better than the LRFU, follow-the-leader, FTPL, average LFU uniform α and optimal $\boldsymbol{w}$, as well as uniform $\boldsymbol{w}$ with optimal values of α. The difference here is around 10^4 demonstrating the benefit of using the proposed scheme(s).

The cache hit performance of the proposed caching algorithm (FL based) under various inter-sBS topologies are shown in Fig. 3. The following three topologies were considered: (i) centralized topology ($a \leftrightarrow b\ \forall a, b \in [1,5]$), (ii) circular ($1 \leftrightarrow 2 \leftrightarrow 3 \leftrightarrow 4 \leftrightarrow 5 \leftrightarrow 1$) and (iii) linear ($1 \leftrightarrow 2 \leftrightarrow 3 \leftrightarrow 4 \leftrightarrow 5$). It can seen from the Fig. 3 that for the centralized topology, the average cache hit is maximum since each sBS has access to all of its neighbouring sBSs data. The sBS needs to exchange

[5] http://grouplens.org/datasets/movielens/

real-time caching information with its neighbouring sBSs, and hence in real settings will lead to a delay while exchanging the information which in turn affects the performance of the online caching strategies. In Fig. 4, the average cache hit is plotted as the delay is varied for circular topology. As seen from Fig. 4 as the delay increases while exchanging the information among the sBSs, the average cache hit eventually reduces. In Fig. 5, average cache hit for different iterations (no. of times each cycle is repeated) is plotted and we can observe that as the number of iterations increases the average cache hit also increase.

V. Remarks and Future Directions

The paper proposed an algorithm for caching in a FL based network setting using theoretical guarantees provided in Theorem 3.1. Taking the best of both the statistical and adversarial setting we design a caching strategy for a distributed network when the demands are highly non-stationary. The proposed FL based caching algorithm, uses a discrepancy measure with the regret minimization. The LRFU algorithm uses a windowed average of demands, and caches the files with the highest average demands. Despite the simplicity of the algorithm, there are no theoretical guarantees when the demands are non-stationary and hence, theoretical guarantees on the performance of the LRFU caching strategy is provided in this work. It is also shown that the proposed FL based caching algorithms outperform LRFU, average LFU, follow-the-leader and FTPL. Further, FL based heuristic caching algorithm is also proposed and it is observed that it performs better than the FL based caching algorithm and hence motivating the future work on providing guarantees for the heuristic algorithm. Finally, it is interesting to explore average weighted demands in place of average demands in the LRFU performs better than the vanilla LRFU and the proposed FL based caching algorithm. In this case, how should one choose the weights? Answers to these questions will be a part of our future work.

VI. Proof of Theorem 3.1

Assume that each SBS b employs the caching strategy in (4) based on the local data $Z_{b,1}^T$. Then, the corresponding conditional average of the hit rate is given by

$$\begin{aligned}
&\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{\boldsymbol{b,T+1}}^{(av)}) \mid Z_{b,1}^T\right] \overset{(a)}{=} \\
&w_b^{T+1} \sum_{t=T-\tau}^{T} \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^T\right] + \\
&\sum_{b' \in \mathcal{N}_b} w_{b'}^{T+1} \sum_{t=T-\tau}^{T} \alpha_{b',t} \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b',t}) \mid Z_{b,1}^T\right] \\
&\overset{(b)}{=} \sum_{t=T-\tau}^{T} \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^T\right] - \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}), \quad (27)
\end{aligned}$$

where (a) follows simply by substituting for $\boldsymbol{\pi}_{\boldsymbol{b,T+1}}^{*}$ from (4). The equality (b) follows by (i) adding and subtracting the term $\sum_{b' \in \mathcal{N}_b} w_{b'}^{T+1} \sum_{t=T-\tau}^{T} \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^T\right]$, and using the definition of $\mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T})$, and (ii) using the fact that $w_b^{T+1} + \sum_{b' \in \mathcal{N}_b} w_{b'}^{T+1} = 1 \; \forall \; b \in \mathbb{B}$. Now, by adding and subtracting $\sum_{t=T-\tau}^{T} \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\right]$, and using the definition of $\mathbb{D}_{b,T}(\boldsymbol{\alpha_{b,T}})$ in (6), the above equation can be lower bounded as

$$\begin{aligned}
\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{\boldsymbol{b,T+1}}^{(av)}) \mid Z_{b,1}^T\right] \;\geq\; & \sum_{t=T-\tau}^{T} \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\right] \\
- \; & \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) \\
- \; & \mathbb{D}_{b,T}(\boldsymbol{\alpha_{b,T}}). \quad (28)
\end{aligned}$$

Similarly, an upper bound can also be obtained as follows

$$\begin{aligned}
\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{\boldsymbol{b,T+1}}^{(av)}) \mid Z_{b,1}^T\right] \;\leq\; & \sum_{t=T-\tau}^{T} \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\right] \\
+ \; & \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) \\
+ \; & \mathbb{D}_{b,T}(\boldsymbol{\alpha_{b,T}}) \quad (29)
\end{aligned}$$

where the above upper bound follows by adding the discrepancies instead of subtraction. Note that the term

$$A_t := \alpha_{b,t} \mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) - \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^t\right]$$

is a Martingale difference, i.e., $\mathbb{E}\left\{A_t \mid Z_{b,1}^t\right\} = 0$. Thus, the following event occurs with a probability of at least $1-\delta$, which follows from the Azuma's inequality

$$\sum_{t=T-\tau}^{T} A_t \leq H_{\max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2}{\tau} \log \frac{1}{\delta}}. \quad (30)$$

The above implies that

$$\begin{aligned}
\sum_{t=T-\tau}^{T} \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\right] \geq \; & \sum_{t=T-\tau}^{T} \alpha_{b,t} \mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \\
- \; & H_{\max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2}{\tau} \log \frac{1}{\delta}}, \quad (31)
\end{aligned}$$

where $H_{\max}$ is the maximum possible hit rate. Since $-A_t$ is also a Martingale difference, using Azuma's inequality, the following holds good with a probability of at least $1-\delta$

$$\begin{aligned}
\sum_{t=T-\tau}^{T} \alpha_{b,t} \mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \;\geq\; & \sum_{t=T-\tau}^{T} \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\right] \\
- \; & H_{\max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2}{\tau} \log \frac{1}{\delta}} \quad (32)
\end{aligned}$$

Using (31) in (28), the following holds good with a probability of at least $1-\delta$

$$\begin{aligned}
\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{\boldsymbol{b,T+1}}^{(av)}) \mid Z_{b,1}^T\right] \;\geq\; & \sum_{t=T-\tau}^{T} \alpha_{b,t} \mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \\
- \; & H_{\max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2}{\tau} \log \frac{1}{\delta}} \\
- \; & \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) \\
- \; & \mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T}). \quad (33)
\end{aligned}$$

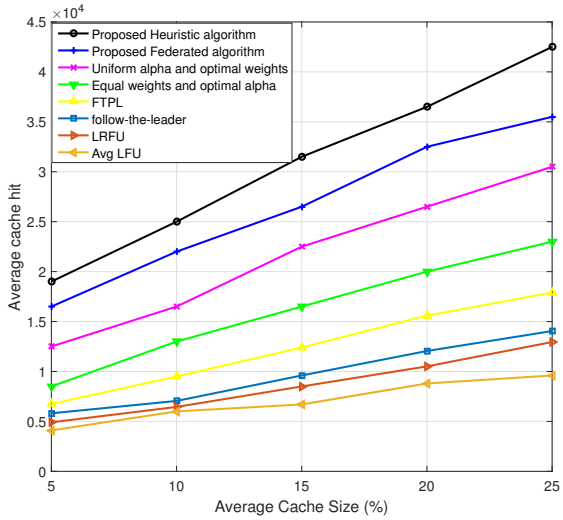

Fig. 2: Average sum cache hit versus cache size.

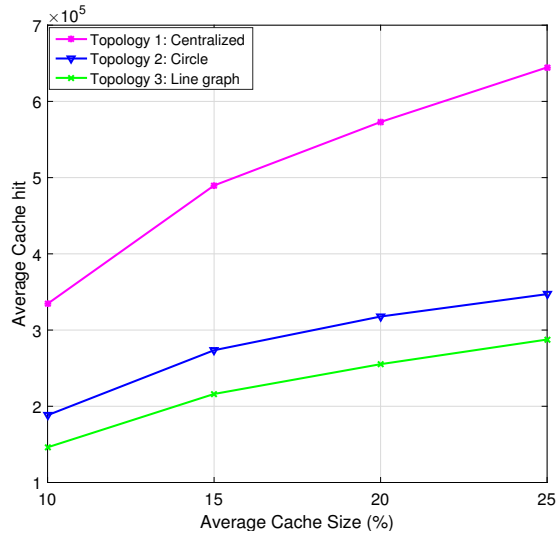

Fig. 3: Average sum cache hit versus topology.

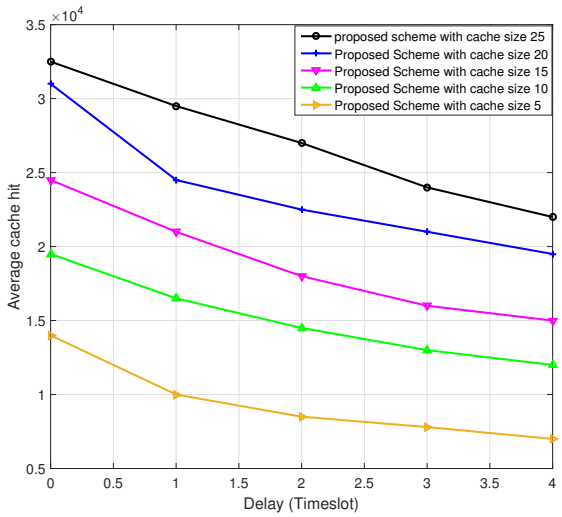

Fig. 4: Average sum cache hit versus delay.

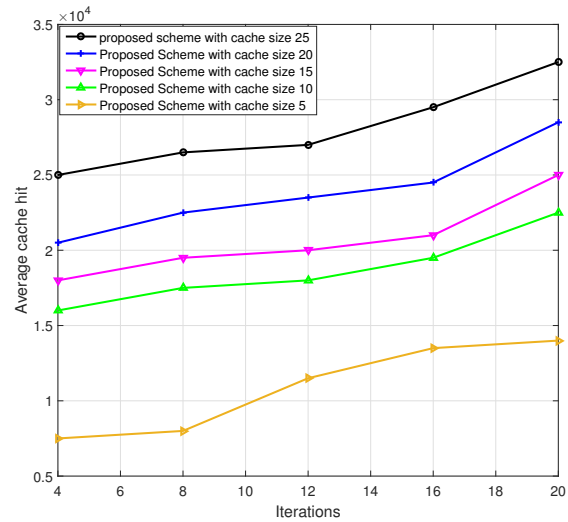

Fig. 5: Average sum cache hit versus iterations.

This proves the first result in the theorem. Similar to the above equation, using (32) in (29), the following holds good with a probability of at least $1-\delta$

$$\begin{aligned}\sum_{t=T-\tau}^{T}\alpha_{b,t}\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) &\geq \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}})\mid Z^{T}_{b,1}\right]\\ &- H_{\max}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}}\\ &- \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T})\\ &- \mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T}) \qquad (34)\end{aligned}$$

Let $\boldsymbol{C}^*_{b,t}$, $t=T-\tau,\ldots,T$, $b\in\mathbb{B}$ be some sequence of caching strategy. Now, consider the following term

$$\begin{aligned}&-\sum_{t=T-\tau}^{T}\alpha_{b,t}\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t})+\sum_{t=T-\tau}^{T}\alpha_{b,t}\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,t})\\ &\leq \sum_{t=T-\tau}^{T}\left(\alpha_{b,t}-\frac{1}{\tau}\right)\left(\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,t})-\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t})\right)\\ &+ \frac{1}{\tau}\sum_{t=T-\tau}^{T}\left(\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,t})-\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t})\right)\\ &\leq H_{\max}\sum_{t=T-\tau}^{T}\left|\alpha_{b,t}-\frac{1}{\tau}\right|+\frac{\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau}, \qquad (35)\end{aligned}$$

where the regret is as defined in (7). If the caching strategy used is $\boldsymbol{C}^*_{b,t}$, then, the above implies that

$$\begin{aligned}\sum_{t=T-\tau}^{T}\alpha_{b,t}\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) &\geq \sum_{t=T-\tau}^{T}\alpha_{b,t}\mathcal{R}_{b,T+1}(C^*_{b,t})\\ &- H_{max}\sum_{t=T-\tau}^{T}\left|\alpha_{b,t}-\frac{1}{\tau}\right|\\ &- \frac{\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau}. \qquad (36)\end{aligned}$$

From (33), we have

$$\mathbb{E}[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}})|Z^{T}_{b,1}] \geq \sum_{t=T-\tau}^{T}\alpha_{b,t}\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,t})$$

$$
\begin{aligned}
&- H_{max} \sum_{t=T-\tau}^{T} \left| \alpha_{b,T} - \frac{1}{\tau} \right| \\
&- \frac{\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau} \\
&\quad -H_{\max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2}{\tau} \log \frac{1}{\delta}} \\
&- \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) \\
&- \mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T,\tau}). \qquad (37)
\end{aligned}
$$

Now, using (34) with $\boldsymbol{C}^*_{b,T+1} := \sum_{t=1}^{T} \alpha_{b,t} \boldsymbol{C}^{(av)}_{b,t}$ in place of $\boldsymbol{\pi}^{(av)}_{b,T+1}$, we get

$$
\begin{aligned}
\mathbb{E}[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}}) | Z^T_{b,1}] &\geq \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,T+1}) | Z^T_{b,1}\right] \\
&- H_{max} \sum_{t=T-\tau}^{T} \left| \alpha_{b,t} - \frac{1}{\tau} \right| \\
&- \frac{2\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau} \\
&\quad -2H_{\max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2}{\tau} \log \frac{1}{\delta}} \\
&- \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) \\
&- 2\mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T,\tau}). \qquad (38)
\end{aligned}
$$

It is possible to choose $\boldsymbol{C}^*_{b,t}$ in such as way that

$$
\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,T+1}) \mid Z^T_{b,1}\right] \geq \sup_{\boldsymbol{h}_{b,t}} \sum_{t=T-\tau}^{T} \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{h}_{b,t}) \mid Z^T_{b,1}\right] - \gamma
$$

for some $\gamma > 0$. Using this in the above equation, and substituting the resulting equation in (33) gives

$$
\begin{aligned}
\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}}) \mid Z^T_{b,1}\right] &\geq \sup_{\boldsymbol{\pi}_{b,t}} \sum_{t=T-\tau}^{T} \alpha_{b,t} \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) \mid Z^T_{b,1}\right] \\
&- 2H_{\max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2}{\tau} \log \frac{1}{\delta}} \\
&- \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) \\
&- \frac{2\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau} \\
&- H_{\max} \sum_{t=T-\tau}^{T} \left| \alpha_{b,t} - \frac{1}{\tau} \right| \\
&- 2\mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T,\tau}) - \gamma. \qquad (39)
\end{aligned}
$$

This completes the proof of the theorem. ∎

VII. Proof of Theorem 3.3

Note that the sequence $A_{b,s} := \frac{1}{\tau}\left[\mathcal{R}_{b,s}(\boldsymbol{\pi}_s) - \mathbb{E}\left\{\mathcal{R}_{b,s}(\boldsymbol{\pi}_s) | Z^{s-1}_{b,1}\right\}\right]$ for $t-\tau-1 \leq s \leq t-1$ is a Martingale difference. The sequence is also bounded, i.e., $|A_{b,s}| \leq \frac{H_{\max}\|\boldsymbol{\alpha_{b,T}}\|_2}{\tau}$. Hence, by Azuma's inequality, it can be seen that with a probability of at least $1-\delta$, for any caching strategy $\boldsymbol{\pi}_b$, the following holds

$$
\begin{aligned}
\sum_f \pi_{b,t} \hat{d}_{b,f,t} \mathcal{L}_f &\leq \frac{1}{\tau} \sum_{s=t-\tau-1}^{t-1} \mathbb{E}\left[\mathcal{R}_{b,s}(\boldsymbol{\pi_b}) \mid Z^{s-1}_{b,1}\right] \\
&+ H_{\max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2 \log \frac{1}{\delta}}{\tau}}, \qquad (40)
\end{aligned}
$$

where the estimate $\hat{d}_{b,f,t} := \frac{1}{\tau} \sum_{s=t-\tau-1}^{t-1} d_{b,f,s}$ for all f. Now, the following bound can be obtained by adding and subtracting $\mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z^{t-1}_{b,1}\right]$, and taking the supremum of the modulus over caching strategies to get the following bound in terms of discrepancy

$$
\begin{aligned}
\sum_f \boldsymbol{\pi}_{b,t} \hat{d}_{b,f,t} \mathcal{L}_f &\leq \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z^{t-1}_{b,1}\right] \\
&+ \mathbb{D}_{b,t}(\boldsymbol{u}_\tau) + H_{\max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2 \log \frac{1}{\delta}}{\tau}} \qquad (41)
\end{aligned}
$$

Similarly, the following bound can be obtained by adding and subtracting $\mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z^{t-1}_{G,1}\right]$, and taking supremum of the modulus over all caching strategies (as done previously) to get

$$
\begin{aligned}
\sum_f \boldsymbol{\pi}_{b,t} \hat{d}_{b,f,t} \mathcal{L}_f &\leq \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z^{t-1}_{G,1}\right] \\
&+ \mathbb{D}_{b,t}(\boldsymbol{u}_\tau) + \mathbb{D}_{GL,t}(\boldsymbol{\alpha}_b) \\
&+ H_{\max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2 \log \frac{1}{\delta}}{\tau}}. \qquad (42)
\end{aligned}
$$

The desired result in the theorem can by obtained by taking supremum over all caching strategies $\boldsymbol{\pi}_b$, and identifying that the supremum in the right hand side results in the LRFU caching strategy. This completes the proof. ∎

References

[1] A. Furuskar, J. Charles, M. Frodigh, S. Jeux, M. Sayed Hassan, A. Saadani, A. Stidwell, J. Soder, and B. Timus, "Refined statistical analysis of evolution approaches for wireless networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 5, pp. 2700 – 2710, May 2015.

[2] K. Intharawijitr, K. Iida, and H. Koga, "Analysis of fog model considering computing and communication latency in 5G cellular networks," in *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, 2016, pp. 1–4.

[3] M. Bennis, M. Simsek, A. Czylwik, W. Saad, S. Valentin, and M. Debbah, "When cellular meets WiFi in wireless small cell networks," *IEEE Communications Magazine*, vol. 51, no. 6, pp. 44–50, Jun. 2013.

[4] S.-F. Chou, T.-C. Chiu, Y.-J. Yu, and A.-C. Pang, "Mobile small cell deployment for next generation cellular networks," in *Proc. IEEE Global Communications Conference*, Dec. 2014, pp. 4852–4857.

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

[6] U. Niesen, D. Shah, and G. W. Wornell, "Caching in wireless networks," *IEEE Transactions on Information Theory*, vol. 58, no. 10, pp. 6524–6540, Oct. 2012.

[7] Y. Wu, S. Yao, Y. Yang, Z. Hu, and C. X. Wang, "Semigradient-based cooperative caching algorithm for mobile social networks," in *Proc. IEEE Global Communications Conference*, Dec. 2016, pp. 1–6.

[8] S. Krishnendu, B. N. Bharath, and V. Bhatia, "Joint edge content cache placement and recommendation: Bayesian approach," in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, 2021, pp. 1–5.

[9] S. Krishnendu, B. N. Bharath, and V. Bhatia, "Cache enabled cellular network: Algorithm for cache placement and guarantees," *IEEE Wireless Communications Letters*, vol. 8, no. 6, pp. 1550–1554, 2019.

[10] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *CoRR*, vol. abs/1209.5807, 2012. [Online]. Available: http://arxiv.org/abs/1209.5807

[11] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution networks," in *2010 Proceedings IEEE International Conference on Computer Communications*, 2010, pp. 1–9.

[12] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless D2D networks," *IEEE Transactions on Information Theory*, vol. 62, no. 2, pp. 849–869, Jan. 2016.

[13] L. Zhang, M. Xiao, G. Wu, and S. Li, "Efficient scheduling and power allocation for D2D-assisted wireless caching networks," *IEEE Transactions on Communications*, vol. 64, no. 6, pp. 2438–2452, Jun. 2016.

[14] B. Chen and C. Yang, "Caching policy for cache-enabled D2D communications by learning user preference," *IEEE Transactions on Communications*, vol. 66, no. 12, pp. 6586–6601, 2018.

[15] M. Gregori, J. Gómez-Vilardebó, J. Matamoros, and D. Gündüz, "Wireless content caching for small cell and D2D networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1222– 1234, May 2016.

[16] F. Cheng, G. Gui, N. Zhao, Y. Chen, J. Tang, and H. Sari, "UAV-relaying-assisted secure transmission with caching," *IEEE Transactions on Communications*, vol. 67, no. 5, pp. 3140–3153, 2019.

[17] Z. Xiao, X. Dai, H. Jiang, D. Wang, H. Chen, L. Yang, and F. Zeng, "Vehicular task offloading via heat-aware mec cooperation using game-theoretic method," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2038–2052, 2020.

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

[19] A. Tatar, M. D. de Amorim, S. Fdida, and P. Antoniadis, "A survey on predicting the popularity of web content," *Journal of Internet Services and Applications*, vol. 5, no. 1, pp. 1–20, Aug. 2014.

[20] B. N. Bharath, K. G. Nagananda, and H. V. Poor, "A learning-based approach to caching in heterogenous small cell networks," *IEEE Transactions on Communications*, vol. 64, no. 4, pp. 1674–1686, Apr. 2016.

[21] J. Song, M. Sheng, T. Q. Quek, C. Xu, and X. Wang, "Learning-based content caching and sharing for wireless networks," *IEEE Transactions on Communications*, vol. 65, no. 10, pp. 4309–4324, 2017.

[22] B. Chen and C. Yang, "Caching policy for cache-enabled D2D communications by learning user preference," in *Proc. IEEE Vechicular Technology Conerence Spring*, 2016.

[23] S. Basu, A. Sundarrajan, J. Ghaderi, S. Shakkottai, and R. Sitaraman, "Adaptive TTL-based caching for content delivery," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1063–1077, 2018.

[24] N. Gast and B. Van Houdt, "Asymptotically exact TTL-approximations of the cache replacement algorithms lru(m) and h-lru," in *2016 28th International Teletraffic Congress (ITC 28)*, vol. 01, 2016, pp. 157–165.

[25] D. Starobinski and D. Tse, "Probabilistic methods for web caching," *Performance Evaluation*, 2001.

[26] E. G. Coffman and P. J. Denning, *Operating Systems Theory*. Prentice-Hall Englewood Cliffs, NJ, 1973.

[27] V. Martina, M. Garetto, and E. Leonardi, "A unified approach to the performance analysis of caching systems," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, 2014, pp. 2040–2048.

[28] J. Li, S. Shakkottai, J. C. Lui, and V. Subramanian, "Accurate learning or fast mixing? dynamic adaptability of caching algorithms," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 6, pp. 1314–1330, 2018.

[29] X. Xu, M. Tao, and C. Shen, "Collaborative multi-agent multi-armed bandit learning for small-cell caching," *IEEE Transactions on Wireless Communications*, vol. 19, no. 4, pp. 2570–2585, 2020.

[30] N. Garg, M. Sellathurai, V. Bhatia, and T. Ratnarajah, "Function approximation based reinforcement learning for edge caching in massive MIMO networks," *IEEE Transactions on Communications*, pp. 1–1, 2020.

[31] T. Zhang, S. Biswas, and T. Ratnarajah, "An analysis on wireless edge caching in in-band full-duplex FR2-IAB networks," *IEEE Access*, vol. 8, pp. 164 987–165 002, 2020.

[32] S. Mehrizi, S. Chatterjee, S. Chatzinotas, and B. Ottersten, "Online spatiotemporal popularity learning via variational bayes for cooperative caching," *IEEE Transactions on Communications*, vol. 68, no. 11, pp. 7068–7082, 2020.

[33] S. Bommaraveni, T. X. Vu, S. Chatzinotas, and B. Ottersten, "Active content popularity learning and caching optimization with hit ratio guarantees," *IEEE Access*, vol. 8, pp. 151 350–151 359, 2020.

[34] Z. Zhang, H. Chen, M. Hua, C. Li, Y. Huang, and L. Yang, "Double coded caching in ultra dense networks: Caching and multicast scheduling via deep reinforcement learning," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1071–1086, 2020.

[35] N. Garg, M. Sellathurai, V. Bhatia, B. N. Bharath, and T. Ratnarajah, "Online content popularity prediction and learning in wireless edge caching," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1087–1100, 2020.

[36] A. Agarwal and J. C. Duchi, "The generalization ability of online algorithms for dependent data," *IEEE Transactions on Information Theory*, vol. 59, no. 1, pp. 573–587, 2012.

[37] D. Lee, J. Choi, J.-H. Kim, S. H. Noh, S. L. Min, Y. Cho, and C. S. Kim, "LRFU: a spectrum of policies that subsumes the least recently used and least frequently used policies," *IEEE Transactions on Computers*, vol. 50, no. 12, pp. 1352–1361, 2001.

[38] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

[39] X. Wang, C. Wang, X. Li, V. C. M. Leung, and T. Taleb, "Federated deep reinforcement learning for internet of things with decentralized cooperative edge caching," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9441–9455, 2020.

[40] X. Wang, R. Li, C. Wang, X. Li, T. Taleb, and V. C. M. Leung, "Attention-weighted federated deep reinforcement learning for device-to-device assisted heterogeneous collaborative edge caching," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 154–169, 2021.

[41] Z. M. Fadlullah and N. Kato, "HCP: Heterogeneous computing platform for federated learning based collaborative content caching towards 6g networks," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2020.

[42] J. Makhoul, "Linear prediction: A tutorial review," *Proceedings of the IEEE*, vol. 63, no. 4, pp. 561–580, 1975.

[43] A. C. Singer and M. Feder, "Universal linear prediction by model order weighting," *IEEE Transactions on Signal Processing*, vol. 47, no. 10, pp. 2685–2699, 1999.

[44] S. S. Kozat, A. C. Singer, and G. C. Zeitler, "Universal piecewise linear prediction via context trees," *IEEE Transactions on Signal Processing*, vol. 55, no. 7, pp. 3730–3745, 2007.

[45] A. Agarwal, P. L. Bartlett, P. Ravikumar, and M. J. Wainwright, "Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization," *IEEE Transactions on Information Theory*, vol. 58, no. 5, pp. 3235–3249, 2012.

[46] P. Tarrès and Y. Yao, "Online learning as stochastic approximation of regularization paths: Optimality and almost-sure convergence," *IEEE Transactions on Information Theory*, vol. 60, no. 9, pp. 5716–5735, 2014.

[47] N. Cesa-Bianchi, A. Conconi, and C. Gentile, "On the generalization ability of on-line learning algorithms," *IEEE Transactions on Information Theory*, vol. 50, no. 9, pp. 2050–2057, 2004.

[48] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. 9, pp. 2121–2159, Feb. 2011.

[49] V. Kuznetsov and M. Mohri, "Time series prediction and online learning," in *Proceedings of The 29th Conference on Learning Theory*, 2016, pp. 1190–1213.

[50] R. S. Liptser and A. N. Shiryayev, *Theory of martingales. Transl. from the Russian by K. Dzjaparidze.* Dordrecht etc.: Kluwer Academic Publishers, 1989.

[51] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *IEEE Transactions on Computers*, 2018. [Online]. Available: http://arxiv.org/abs/1812.06127

[52] A. Cohen and T. Hazan, "Following the perturbed leader for online structured learning," in *International Conference on Machine Learning*, 2015, pp. 1034–1423.

[53] J. Abernethy, C. Lee, A. Sinha, and A. Tewari, "Online linear optimization via smoothing," in *Proceedings of the 27th Conference on Learning Theory,*, 2014, pp. 807–823.

[54] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. MIT press, 2018.

[55] S. Bubeck, "Convex Optimization: Algorithms and Complexity," *ArXiv e-prints*, May 2014.

PLACE PHOTO HERE

r eceived her B.Tech. degree in Electronics & Communications Engineering from the National Institute of Technology Calicut, India, in 2016. In 2017, she started working toward the Ph.D. degree at the Department of Electrical Engineering, Indian Institute of Technology Indore, India. From $2016-17$, she worked as an Engineer with Tata Elxsi, Trivandrum, India. She is a recipient of Visvesvaraya PhD Fellowship from MeitY, Government of India. Her current research interests include wireless communications, edge caching, optimization and machine learning.

PLACE PHOTO HERE

Bharath B. N. received the B.E. degree in Electrical and Electronics Engineering from the B.M.S. College of Engineering, Bengaluru, in 2005, and the Ph.D. degree from the Electrical Communication Engineering (ECE) Department, Indian Institute of Science (IISc), Bengaluru in 2013. He worked at Qualcomm Inc. from 2013 to 2014. He is currently an Assistant Professor with the Department of Electrical Engineering at Indian Institute of Technology Dharwad. His research interests include signal processing and machine learning for communication/network problems, information theory, optimization theory, and distributed stochastic optimization.

Navneet Garg received the B.Tech. degree in Electronics and Communication Engineering from College of Science & Engineering, Jhansi, India, in 2010, and the M.Tech. degree in Digital Communications from ABV-Indian Institute of Information Technology and Management, Gwalior, in 2012. He has completed the Ph.D. degree in June 2018 from the Department of Electrical Engineering at the Indian Institute of Technology Kanpur, India. From July 2018-Jan 2019, he visited The University of Edinburgh, UK. From February $2019-2020$, he is employed as a research associate in Heriot-Watt University, Edinburgh, UK. Since February 2020, he is working as a Research Associate in The University of Edinburgh, UK. His main research interests include wireless communications, signal processing, optimization, and machine learning.

Vimal Bhatia (SM'12) is currently working as a Professor with the Indian Institute of Technology (IIT) Indore, India, and is an adjunct faculty at IIT Delhi and IIIT Delhi, India and UHK, Czech Republic. He received Ph.D. degree from Institute for Digital Communications with The University of Edinburgh, Edinburgh, U.K., in 2005. During Ph.D. he also received the IEE fellowship for collaborative research at the Department of Systems and Computer Engineering, Carleton University, Canada, and is Young Faculty Research Fellow from MeitY, Government of India. He is also a recepient of Prof SVC Aiya Memorial Award (2019). He has worked with various IT companies for over 11 years both in India and the UK. He is a PI/co-PI/coordinator for external projects with funding of over USD 17 million from MeitY, DST, UKIERI, MoE, AKA, IUSSTF and KPMG. He has more than 300 peer reviewed publications, and has filed 13 patents (with two granted). He has supervised 15 defended/submitted PhD thesis. His research interests are in the broader areas of communications, non-Gaussian non-parametric signal processing, machine/deep learning with applications to communications and photonics. He is a reviewer for IEEE, Elsevier, Wiley, Springer, and IET. He is currently Senior Member of IEEE, Fellow IETE and certified SCRUM Master. He was also the General Co-Chair for IEEE ANTS 2018, and General Vice-Chair for IEEE ANTS 2017. He has served as founder head of Center for Innovation and Entrepreneurship, Associate Dean R&D and Dean, Academic Affairs. He has delivered many talks, tutorials and conducted faculty development programs for the World Bank's NPIU TEQIP-III programs. He is currently Associate Editor for IETE Technical Review, Frontiers in Communications and Networks, Frontiers in Signal Processing, and IEEE Wireless Communications Letters.

Tharmalingam Ratnarajah Tharmalingam Ratnarajah (Senior Member, IEEE) is currently with the Institute for Digital Communications, The University of Edinburgh, Edinburgh, UK, as a Professor in Digital Communications and Signal Processing. He was a Head of the Institute for Digital Communications during $2016-2018$. His research interests include signal processing and information theoretic aspects of 6G wireless networks, full-duplex radio, mmWave communications, random matrices theory, interference alignment, statistical and array signal processing and quantum information theory. He has published over 400 publications in these areas and holds four U.S. patents. He has supervised 16 PhD students and 21 post-doctoral research fellows and raised $ 11+ million USD of research funding. He was the coordinator of the EU projects ADEL (3.7M€) in the area of licensed shared access for 5G wireless networks, HARP (4.6M€) in the area of highly distributed MIMO, as well as EU Future and Emerging Technologies projects HIATUS (3.6M€) in the area of interference alignment and CROWN (3.4M€) in the area of cognitive radio networks. Dr Ratnarajah was an associate editor IEEE Transactions on Signal Processing, $2015-2017$ and Technical co-chair, The 17th IEEE International workshop on Signal Processing advances in Wireless Communications, Edinburgh, UK, $3-6$, July 2016. Dr Ratnarajah is a Fellow of Higher Education Academy (FHEA).