ALGORITHMS FOR MOBILE EDGE CACHING IN FUTURE WIRELESS NETWORKS

Ph.D. Thesis

by

KRISHNENDU S

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY INDORE

JUNE, 2022

ALGORITHMS FOR MOBILE EDGE CACHING IN FUTURE WIRELESS NETWORKS

A THESIS

Submitted in partial fulfillment of the

requirements for the award of the degree

of

DOCTOR OF PHILOSOPHY

by

KRISHNENDU S

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JUNE, 2022

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled "**ALGORITHMS FOR MOBILE EDGE CACHING IN FUTURE WIRELESS NETWORKS**" in the partial fulfillment of the requirements for the award of the degree of DOCTOR OF PHILOSOPHY and submitted in the DEPARTMENT OF ELECTRICAL ENGINEERING, **Indian Institute of Technology Indore**, is an authentic record of my own work carried out during the time period from May 2017 to June 2022 under the supervision of Dr. Vimal Bhatia, Professor, Indian Institute of Technology Indore, India.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other institute.

Krishnendu 03/06/2022

Signature of the student with date

(KRISHNENDU S)

This is to certify that the above statement made by the candidate is correct to the best of my/our knowledge.

03/06/22

Signature of Thesis Supervisor with date

(Prof. VIMAL BHATIA)

Krishnendu S has successfully given her Ph.D. Oral Examination held on 31/10/2022.

31/10/22

Signature of Thesis Supervisor with date

(Prof. VIMAL BHATIA)

ACKNOWLEDGEMENTS

I express my heartfelt and profound gratitude to my Ph.D. supervisor Prof. Vimal Bhatia for his invaluable guidance, encouragement, vision, direction throughout my thesis work and especially for his calm nature. It is my great honour and pleasure to work under the esteemed guidance of kind and responsible supervisor like Prof. Vimal Bhatia. I am thankful for all the freedom he gave me to explore various research topics and was always available for discussion despite his busy schedule. I would also like to thank my PSPC committee members Prof. Shaibal Mukherjee and Prof. Bhupesh Kumar Lad for their fruitful discussions and suggestions towards my research.

I would like to thank the Visvesvaraya PhD Scheme of the Ministry of Electronics & Information Technology, Government of India, being implemented by Digital India Corporation for providing financial assistance. I would also take this opportunity to thank Department of Electrical Engineering and the UKIERI project for funding my travels for conferences, workshops, and my research publication charges which provided me an excellent platform to share and discuss my ideas with the fellow researchers and experts who have helped to shape my Ph.D. thesis.

I am very thankful to Dr. Bharath B. N. at Indian Institute of Technology Dharwad, whose expertise and insightful feedback pushed me to sharpen my thinking and widen my research from various perspectives. I am very thankful for his assistance throughout my research period.

I am truly thankful to all the faculty members and the staff of IIT Indore for their invaluable cooperation throughout my thesis work. I would also like to thank all the members in the Signal and Software group (SaSg) for their encouragement and support. Specially, I would like to extend my appreciation to my seniors Dr. Abhijeet Bishnu, Dr. Praveen Kumar Singya and Dr. Arijit Datta, who helped me whenever required. I would also like to thank my cohort members and juniors: Puneet Singh Thakur, Kamal Kishore Garg, Anupma Sharma, Vaishali Sharma, Shubham Bisen, Abhinav Singh Parihar, Justin Jose, Deepak Kumar, Vidya Bhaskar Shukla, and other colleagues who helped me in several ways during my thesis work.

Life in campus and in general would be incomplete without mentioning few names to whom I will always be grateful for their constant companionship during all the ups and downs: Dr. Shanuja, Meghna, Tanushree, Hussain, Tinto, Kavin, Bhavana, Dr. Sandeep, Pooja, Nirmal, Ansil, Ashin, Robi and Jacob. Thanks for all the support and for all the happy times we had.

Moreover, whatever little I have achieved in my life, the credit goes to my family members, especially my parents and my brother, whom I take great pride in dedicating this thesis. They have always been a source of inspiration for me and have kept trust and faith in whatever I did. This journey was not possible without their support at each and every step of my life. Finally, I cannot express my gratitude towards my family, especially my mother, in words for being my constant anchor. I would have never made this far without her believing in me.

Krishnendu S

Dedicated to my parents
Amma & Achan

ABSTRACT

This thesis explores one of the key enablers of 5G wireless networks leveraging small cell network deployment, namely mobile edge caching. Endowed with predictive capabilities and harnessing recent developments in storage, context-awareness and social networks, peak traffic demands can be substantially reduced by caching at the edge of the network. The surge in Internet usage through smart phones, social media platforms and online video streaming has heralded an explosive growth in the amount of data being created. Due to the fast development of communication based applications, it is expected that there will be 5.3 billion total Internet users (66 percent of global population) by 2023, up from 3.9 billion (51 percent of global population) in 2018. This new phenomenon has urged mobile operators to redesign their current networks and seek more advanced and sophisticated techniques to increase coverage, boost network capacity, and cost-effectively bring contents closer to users. A promising approach to meet these unprecedented traffic demands is via the deployment of small cell networks (SCNs). SCNs represent a novel networking paradigm based on the idea of deploying short-range, low-power, and low-cost small base stations (SBSs) underlaying the macro cellular network. In addition to the vast and dynamic mobile data generated, the limited spectrum especially in the wireless link, due to the extensive use of smart phones and other devices, ultimately leads to congestion in the backhaul links. The implementation of dense and SBSs does reduce the latency and provides immense throughput in the 5G and beyond network, but the bottleneck issue still remains the same. Thus, the existing small cell networking paradigm falls short of solving peak traffic demands whose large-scale deployment hinges on expensive site acquisition, installation and backhaul costs. These shortcomings are set to become increasingly acute, due to the surging number of connected devices and the advent of ultra-dense networks, which will continue to strain current cellular network infrastructures. These key observations mandate a novel networking paradigm which goes beyond current heterogeneous small cell deployments leveraging the latest developments in storage, context-awareness, and social networking. This novel paradigm helps in storing the data locally at the edge of the network and is referred to as mobile edge caching. Mobile edge caching which exploits the vast data, compensates for the shortage of local computing capacity and high transmission costs of individual cloud computing. Thus, the demand of the hour is a shift from the large scale cloud data to wide range edge devices. Both edge caching and computation helps in storing and implementation of learning algorithms at the edge, hence making the edge intelligent and further reducing the burden on the backhaul. The standard simplified algorithms such as least frequently used (LFU), least recently used (LRU), least recently/frequently used (LRFU) and other variants, can be inefficient when it comes to dynamic environments, since they do not take into account the correlation and non-stationarity of the demand requests. Therefore, a gradual shift towards learning and optimizing the edge devices for content prediction is observed. This in turn yields significant gains in terms of network resources, minimizing operational and capital expenditures.

To strike a balance between increasing mobile traffic and user experience, mobile edge caching and computing can be seen as invariable solution by bringing storage and computation close to the edge. Further, it is observed that, users who have similar interests and backgrounds tend to rank the content in a similar way. However, in a realistic network, the content popularity tends to be dynamic hence

motivating the dynamic cache. Thus borrowing the tools from probability theory, machine learning, we have designed algorithms which take advantage of the memory distributed across the network. The proposed network architecture involves caching popular data sets closer to users, which will yield higher user's satisfaction; attain high backhaul offloading gains and more revenue for the operators. In the coming years, mobile edge caching is seen as a potential alternative to reduce congestion in the backhaul. Caching contents at the edge of the network not only brings the data closer to the user and makes the content access easier but also gives an opportunity for network service provider to fulfill user demands with limited resources. Recently, there is much work focusing on mobile caching strategies due to the advantages of fast response.

Contents

List of Figures

List of Abbreviations/Acronyms

ADAM adaptive moment estimation.

BS base station.

CDN content delivery network.

CF collaborative filtering.

C-RAN cloud radio access network.

D2D device-to-device.

DL deep learning.

DMT diversity-multiplexing gain tradeoff.

DNN deep neural network.

FL federated learning.

FTPL follow-the-perturbed-leader.

i.i.d. independent and identically distributed.

IA interference alignment.

IAB integrated access and backhaul.

LFU least frequently used.

LRFU least recently/frequently used.

LRU least recently used.

MAB multiarm bandit.

MABP multi-armed bandit problem.

MAMB multi-agent multi-armed bandit.

MBS macro base station.

MDP markov decision process.

MEC mobile edge computing.

MIMO multiple-input and multiple-output.

MLE maximum likelihood estimation.

PDF probability density function.

PTM probability transition matrix.

QoE quality-of-experience.

QoS quality-of-service.

Relu Rectified linear unit.

RMAB restless multi-armed bandit.

SBSs small base stations.

SCN small cell networks.

SINR signal-to-interference-plus-noise ratio.

SNR signal-to-noise ratio.

TDMA time-division multiple access.

TTL time to live.

UAV unmanned aerial vehicle.

List of Symbols

- Basic arithmetic and calculus notations with their definitions.

Probability & Statistics

Let X be a random variable (RV).

Notation	Definition
$\mathbb{E}[\cdot]$	statistical expectation operator
$\mathcal{P}[\cdot]$	statistical probability operator
$f_X(\cdot)$	probability density function (PDF) of a RV X
$F_X(\cdot)$	cumulative distribution function (CDF) of a RV X
$\mathcal{CN}(\mu, \sigma^2)$	complex normal distribution with mean μ and variance σ^2
$\sup A$	supremum of the set A

Vectors & Matrices

Let $\mathbf{a} \in \mathbb{P}^{1\times n}$ and $\mathbf{A} \in \mathbb{P}^{m\times n}$ be an $1 \times n$ complex vector and an $m \times n$ complex matrix, respectively.

Notation	Definition/interpretation
$\mathbf{a}_i$	i^{th} element of $\mathbf{a}$
$\mathbf{A}_{i,j}$	i^{th} element of j^{th} column of $\mathbf{A}$
$\|\| \cdot \|\|_F^2$	Squared Frobenius norm of $(\cdot)$
$\mathbf{I}_N$	Identity matrix of rank N
$(\cdot)^*$	conjugate of $(\cdot)$

Miscellaneous

Notation	Definition
$\lvert \cdot \rvert$	absolute value
$\lvert\lvert \cdot \rvert\rvert_{op}$	operator norm
$\lvert\lvert \cdot \rvert\rvert_F$	Frobenius norm
$\approx$	approximate value
vec	represents vector
$\binom{n}{k}$	binomial coefficient of n choose k
$Dirch(\alpha_1, \alpha_2, \ldots, \alpha_K)$	is the Dirichlet distribution with parameters $\alpha_1, \alpha_2, \ldots, \alpha_K$
$\arg\max_i b_i$	index i corresponding to the largest b_i
$\arg\min_i b_i$	index i corresponding to the smallest b_i
Superscript $(\cdot)^T$	represents transposition

Chapter 1

Introduction

1.1 Overview

1.2 Background & Motivation

The recent proliferation of smartphones has substantially enriched the mobile user experience, leading to a vast array of new wireless services, including multimedia streaming, web-browsing applications and socially-interconnected networks. This phenomenon has been further fueled by mobile video streaming, which currently accounts for almost 50% of mobile data traffic, with a projection of 500-fold increase over the next 5 years [2]. At the same time, social networking is already the second largest traffic volume contributor with a 60% average share [2]. This new phenomena has urged the wireless network operators to redesign the current networks and opt for more advanced techniques which helps in increasing the network capacity and cost-effectively bring the contents closer to the users. While small cell densification is clearly the way to go, a number of technical challenges remain unsolved. Indeed, while small cell densification was shown to boost capacity, simply adding small cells may turn out to be energy-inefficient [3, 4]. In addition, backhaul optimization and the optimal location of small cells represent one of the main limiting factors before a full rollout of small cells takes place. The importance of the backhaul is further underscored with the unabated proliferation of smartphones with the vast array of new wireless services. As a result, novel approaches to backhaul-aware small cell networking have been recently proposed in the literature [5] such as how to optimally

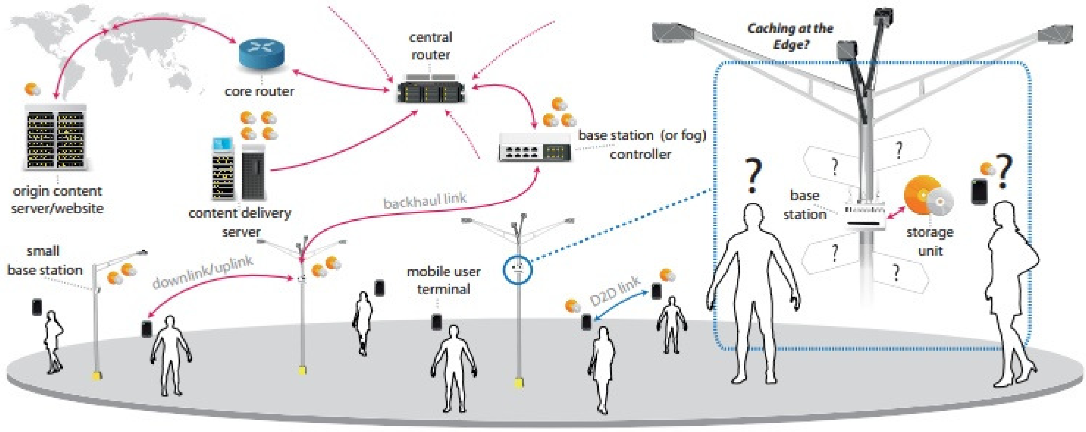


Figure 1.1: Illustration of mobile edge computation in future wireless networks. Contents available in the origin server are cached at the base stations and user devices, for offloading the backhaul and the wireless links. [1]

decouple control and data planes to make cells more adaptive to traffic dynamics and network state while having a global view of the network, backhaul offloading via smart edge caching [6, 7], cloud radio access network (C-RAN), software defined networking (SDN) [8], resource/network virtualization, ultra-dense networks, massive multiple-input and multiple-output (MIMO), etc. Among these approaches, in this thesis, we focus on mobile edge caching as a way of dealing with backhaul offloading, which is especially crucial in dense deployments (see [9], [10], or [11]). Fig. 1.1 illustrates the mobile edge caching in a heterogeneous wireless network. As shown in Fig. 1.1, in the network of the future, memory units can be installed in gateway routers between the wireless network and the Internet, in SBSs of different sizes (small or regular size cells), and in end-user devices (mobile phones, laptops, routers, etc.).

In the following section, we first give an overview of history of mobile edge caching and summarize the advancements in mobile edge caching. Afterwards, we discuss the outline and contributions of the thesis accordingly.

1.3 Caching: A Brief History and Related Works

The idea of caching goes back to the sixties in the context of algorithm design in operating systems. In the past decades, there has been extensive studies on web caching algorithms, aiming to improve the world wide web scalability and offloading the network, by caching contents in the proxy servers. Similarly, in wireless network for improving the content delivery, mobile edge caching has been considered as one of the important feature of the 5G networks. In recent years, numerous mobile edge caching algorithms for content delivery network (CDN) have emerged [12], allowing content providers to reduce access delays to the requested contents.

Similar to what we present in this thesis, the growing literature is mostly based on wireless caching at the edge of network. An exhaustive list of recent literature is given in [9, 12–15]. In the following, we summarize some of these works based on their similarities and directions.

1.3.1 Proactive Caching and Content Popularity Estimation

Proactive caching in small cell networks (SCN)s with perfect knowledge of the content popularity is given in [16, 17]. In [17], exploiting context-awareness, social networks, device-to-device (D2D) communications, the proactive caching approaches for SCNs are studied both at the SBSs and user terminals, showing that several gains are possible under the given numerical setup. Therein, instead of perfect knowledge of the content popularity, an estimation is done via machine learning tools (the collaborative filtering (CF) in particular), by exploiting correlations of human behaviour on their preferences. Thus, having such an estimation, the caching decision is applied more efficiently, yielding better performance in terms of the users' satisfaction and offloading of the network. On the other hand, a well-known problem in the CF literature is the cold-start problem which can occur in the case of estimation with very few amount of information. Therefore, to boost the content popularity estimation, one approach harnessing the machine learning literature is transfer learning, based on the idea of smartly transferring information from a target domain to a source domain (see [18] for a survey). Further investigations are needed to combine this approach with the proactive caching in SCNs. Additionally, in the context of proactive caching, the centrality measures for the content placement are exploited in [19]. Therein, a simple content dissemination process is introduced and the preliminary performance results of this centrality-based content placement methods are given via numerical simulations. Alternative to these proactive approaches, a game theoretical formulation of the proactive caching problem as a many-to-many matching game is introduced in [20]. A matching algorithm that reaches a pairwise stable outcome is provided for the caching problem, showing that the number of satisfied requests can be reach up to three times the satisfaction of a random caching policy.

1.3.2 Coded Caching Gains

Information-theoretic formulation of the caching problem is studied by [21]. Therein, local and global caching gains, which depend on the available memory of each user and cumulative memory of all users respectively, are derived based on a coded caching scheme. The proposed scheme consists of placement and delivery phases (i) is given for a centralized setup where the content placement is handled by a central server, (ii) is essentially offline as there is no content placement during the delivery phase, (iii) is shown to outperform conventional uncoded schemes under uniform content popularities, and (iv) works in a single shared link instead of more general networks. These results are then extended to non-uniform content popularities in [22, 23], non-uniform cache access in [24], heterogeneous cache sizes in [25], online caching systems in [26], hierarchical caching networks in [27] and multi-server case in [28]. Moreover, the improved bounds are given in [29], delay-sensitive content case is studied in [30] and the information-theoretic security aspects are shown in [31]. With similar line to these works, a decentralized approach for D2D networks with random coded caching is studied in [32] in terms of scaling laws where a protocol channel model similar to [33] is taken into account. In the same vein, the performance of decentralized random caching placement with a coded delivery scheme is given in [34], where the expected rate is characterized for random demands with Zipf popularity distribution. In the context of distributed storage systems and coding, the performance of simple caching, replication and regenerating codes is studied in a D2D scenario in [35], in which a simple decision rule for choosing simple caching and replication is derived for minimizing the expected total cost in terms of energy consumption. On the other hand, the study of the physical layer functionality of wireless distributed storage systems is given in [36] from point of space-time storage codes. Based on that work, a wireless storage system that communicates over a fading channel is studied in and a novel protocol for the transmission is proposed based on algebraic space-time codes, in order to improve the system reliability while keeping the decoding at a feasible level. It is shown that the proposed protocol performs better than the simple time-division multiple access (TDMA) protocol and falls behind the optimal diversity-multiplexing gain tradeoff (DMT). Alternatively, a triangular network coding approach for cache content placement is presented in [37], in which the uncoded content placement and the triangular network coding strategies are compared in a numerical setup. Additionally, a coded caching scheme over wireless fading channel is presented in [38], whereas [39] casts the caching problem into a multi-terminal source coding problem with side information.

1.3.3 Joint Designs

In terms of joint designs, a two time-scale joint optimization of power and cache control is given in [40] for cache-enabled opportunistic cooperative MIMO. First, for the short time scales, the closed-form expressions for the power control are derived from an approximated Bellman equation. Then, for the long time scales, the caching problem is translated into a convex stochastic optimization problem and a stochastic subgradient algorithm is provided for its solution. The proposed solution is shown to be asymptotically optimal for high signal-to-noise ratio (SNR) whereas its comparison with baseline approaches are done via simulations. Another mixed time-scale solution for cooperative MIMO is given in [41]. Therein, in order to minimize the transmit power under the quality-of-service (QoS) constraint, the MIMO precoding is optimized in the short time scale and cache control is done in the long time scale. Additional to these approaches, the joint optimization of cache control and playback buffer management for video streaming is given in [42]. The joint caching and beamforming for backhaul limited caching networks is studied in [43], and

finally the joint caching and interference alignment (IA) in MIMO interference channel under limited backhaul capacity is presented in [44].

1.3.4 Mobility

Mobility aspects of coded content delivery is analyzed in [45] based on a discrete-time Markov chain model. In order to minimize the probability of using the main base station in this model, a distributed approximation algorithm based on large deviation inequalities is introduced and numerical experiments on a real world dataset are conducted for the proposed algorithm. Another caching scheme that exploits users' mobility is given in [46], in which the influence of the system parameters on the delay gains are investigated via the system level simulations. The work in [47] also consider the impact of mobility in cache-enabled networks.

1.3.5 Energy Consumption

Energy consumption aspects of caching both in terms of area power consumption and energy efficiency are investigated in [48]. Therein, the cache-enabled base stations (BSs) are distributed according to a homogeneous Poisson point process and the optimization is done using a detailed power model. On the other hand, energy harvesting aspects of proactive caching is highlighted in [49], and an effective push mechanism for energy harvesting powered small-cell BSs is proposed in [50]. Also, a joint caching and BS activation for green cellular networks is proposed in [51].

1.3.6 Deployment Aspects

Concerning the deployment aspects of cache-enabled SBSs with limited backhaul, a study is given in [52]. In that study, the cache-enabled SBSs are stochastically distributed for the analysis rather than the traditional grid models. The expressions for the outage probability and average content delivery rate are derived as a function of the SNR, SBSs intensity, target content bitrate, cache size and shape of content popularity distribution. The results in [53] shows that storing the most popular contents is beneficial only in some particular deployment scenarios. On the other hand, for cache enabled D2D communications, another stochastic framework is shown in [54], by relying on two performance metrics that quantify the local and global fraction of served content requests. Yet another study for the stochastically distributed cache-enabled nodes is given in [55]. Given the fact that the cost is defined as a function of distance, the expected cost of obtaining the complete content under coded as well as uncoded content allocation strategies is investigated. As an extension to [55], the expected deployment cost of caches vs. the expected content retrieval from the caches is analyzed in [56].

1.3.7 Learning Algorithms

One of the key problems to be addressed in mobile edge 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 [57], [58], [59]-[62]). 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 [63]. The authors in [64] 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 LRU [65], climb [66, 67], and k-LRU [68] 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 [69]. Many of these works result in a regret of $\omega(T)$. In [70], 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 O(log T). In [71], function approximation based reinforcement learning approaches are proposed for a massive MIMO based network. In [72] integrated access and backhaul 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 authors in [73] 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 [74], active learning approach is used to learn the content popularities to design an accurate content request prediction model. In [75], 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 [76]). 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 [77].

1.4 Thesis Outline and Contributions

This thesis consists of Chapters 1 to 6, whose brief description is as follows:

Chapter 1. Introduction : In this chapter, a brief introduction about, wireless edge caching, motivation, and main contributions of the thesis are provided.

Chapter 2. Caching at Base Station : In this chapter, the problem of maximizing the average rate of cache hit in a heterogeneous wireless network for a given probability distribution of content popularity and a given network topology, under the constraint of cache size at each SBS is proposed [78].

Chapter 3. Federated Learning: In this chapter, Federated learning based caching strategy is proposed which is assumed to be a weighted combination of past caching strategies of neighbouring BSs [11]. Therefore, a structure on the caching strategy is assumed, and a high probability guarantee/bound on the conditional average cache hit is derived.

Chapter 4. Bayesian Learning for Joint Optimization : In this chapter, joint optimization of both caching and recommendation is formulated and the influence of the recommendation on the popularity is modelled through a probability transition matrix. Two estimation procedures namely `Point estimation` and `Bayesian estimation` methods are presented [10]. Further, theoretical guarantees are also provided on the performance of the algorithm.

Chapter 5. Recommendation based Caching using Whittle Index : In this chapter, caching and recommendation in a wireless network is jointly looked into. The demands are modelled as a two-state markov chains. The two objectives results in the classic exploration v/s exploitation tradeoff seen in sequential decision making problems and Whittle's indexability is established

Chapter 6. Conclusion and Future Work: In this chapter, the conclusions made from proposed algorithms in this thesis are summarized, and suggestions for future research are provided.

The contribution of this thesis include;

1. The problem of edge caching in a cellular network with a given topology and storage size in each SBS is looked into. The problem of maximizing the cache hit with respect to the cache placement strategy (which takes values in $\{0, 1\}$) subject to storage size constraint in each SBS turns out to be NP hard, and hence, an approximation is proposed to the problem. Theoretical guarantees are proven on the performance of the proposed method. A complexity analysis is also presented of the proposed algorithm.

2. Assuming structured cache placement, high probability bounds on the conditional average cache hits are derived using Martingale difference equation [79]. 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.

3. Two estimation procedures, `Point` and `Bayesian estimation` are provided. A probabilistic model using Bayesian inference based on Dirichlet distribution is proposed. Specifically, the influence of recommendation on the popularity profile is modelled using a conditional probability distribution. A high probability guarantee on the estimated caching and recommendation strategies is provided. Irrespective of the estimation method, it is shown that with a probability of $1 - \delta$ the proposed caching and recommendation strategy is ϵ close

to the optimal solution.

4. The joint caching and recommendation is modeled as an markov decision process (MDP) and its Whittle's indexability is established. Based on the Whittle's indexability condition, the whittle index policy for the caching problem is provided. The Whittle's index policy is known to have near-optimal performance and have shown low-complexity [80].

Chapter 2

Caching at Base Station

2.1 Overview

In the design of distributed content network, content popularity distribution is an important factor guiding SBSs to cache content preferred by users [15, 81]. The content popularity profile does not always follow a uniform distribution; in practice, some contents are accessed much more frequently than others, thereby rendering the problem of optimal edge caching an uphill and uncertain task. The cache algorithms such as LRU and LFU are being used in CDN, to improve the overall network capacity. Both these algorithms primarily work on the principle governing the last access time of the files. In these algorithms, each block or file is assigned a counter, which keeps track of the number of times the content is accessed. Whenever the cache memory is full, the block with the least count number is discarded to make room for the new content. For example, in case of LRU, the file which is least used in the past, is assumed to have least probability of use in the future, and hence evicted. On the other hand, LFU replaces the file that has been used least recently when a new file arrives. Nevertheless, these algorithms are not suitable for mobile edge caching in a wireless scenario with multiple SBSs, since a mobile user can fetch the requested content from multiple SBSs to which it is connected [82].

While there has been a lot of study on edge caching, this chapter proposes a ϵ close-to-optimal cache placement solution that performs better than the existing solutions. In particular, for a given popularity profile and network topology, a cache placement mechanism at the SBS is proposed such that the average rate of cache hit is maximized. Despite the knowledge of the popularity profile, the problem of optimal cache content placement turns out to be combinatorial, and more often NP-hard [83]. Thus, the hard combinatorial problem lies at the heart of most of the cache placement problems. This chapter proposes an efficient method of solving the problem of edge caching in a cellular network, where the caching is done only at the SBSs. The topology is assumed to be constant, and each user requests one of the J popular items with a certain rate; the rate is assumed to be known at each SBS. This chapter considers the problem of edge caching in a cellular network with a given topology and storage size in each SBS. The problem of maximizing the cache hit with respect to the cache placement strategy (which takes values in $\{0, 1\}$) subject to storage size constraint in each SBS turns out to be NP hard, and hence, an approximation is proposed to the problem. In particular, the following method is used to find an approximate solution: (i) the $0/1$ variable is relaxed to take any value in $[0, 1]$; (ii) a perturbation term is added to the objective function, and a condition is derived under which the resulting function is convex. Thus, the resulting problem is convex, and hence an optimal solution is found through optimization techniques, and finally (iii) the resulting non-integer solution is converted into an integer solution through randomized rounding algorithm. Theoretical guarantees are proven on the performance of the proposed method. Significant improvement in the cache hit performance of the proposed method over the popular greedy, LRU and LFU algorithms is demonstrated through simulation. A complexity analysis is also presented of the proposed algorithm in comparison with the greedy algorithm and exhaustive algorithm. Further, a deep learning (DL) model is incorporated to predict the popular contents. K-means clustering based DL framework is used to predict popularity at the central BS, which in turn collects the information from all the SBSs to learn the data. Simulation results are provided to show that the proposed DL model significantly outperforms the recent prediction models in terms of average cache hit rate and mean squared error.

2.2 System Model

The system model shown in Fig. 2.1 consists of a wireless cellular network with M SBSs and N users denoted by the sets $\mathcal{B}$ and $\mathcal{U}$, respectively. The variable $l_{im} = 1$ is used to denote the presence of a link between a user i and the SBS m, and $l_{im} = 0$, otherwise. This induces a bipartite graph, where the neighboring SBS of a user i is denoted by $\mathcal{N}_U(i) := \{m \in \mathcal{B} : l_{im} = 1\}$. Further, without loss of generality, each data file is assumed to be of equal size. The total number of files is assumed to be J.

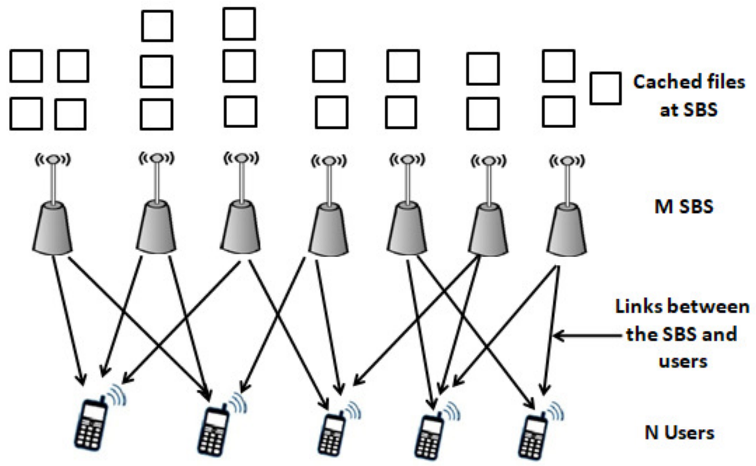


Figure 2.1: System model for the cache based wireless heterogeneous network

The m-th SBS is assumed to have a storage capacity of C_m files, $m = 1, 2, \ldots, M$; this forms a distributed caching network. The term $X_{jm} \in \{0, 1\}$ is used to denote the presence of the file j, $j = 1, 2, \ldots, J$ in the SBS m:

$$X_{jm} = \begin{cases} 1, & \text{if } j\text{th file is present in } m\text{th SBS}, \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

Assuming that SBS m can store at most C_m files leads to the following constraint

$$\sum_{j=1}^{J} X_{jm} \leq C_m, \;\; m = 1, 2, \ldots, M. \tag{2.2}$$

Each user i is assumed to make requests for a file j at a rate of $\lambda_{ij} > 0$. Here, λ_{ij} denotes the arrival rate of the files. For concreteness, it can be assumed that the request follows Poisson distribution with rate λ_{ij}. However, this is not required for the rest of the analysis. Thus, the average rate of cache hit is given by

$$F(X) = \sum_{i,j} \lambda_{ij} \left(1 - \prod_{m \in \mathcal{N}_U(i)} (1 - X_{jm}) \right), \tag{2.3}$$

where for brevity $X := \{X_{jm} : 1 \le j \le J, 1 \le m \le M\}$. Without loss of generality, the knowledge of λ_{ij} at the central SBS is assumed, then the objective is to solve the following optimization problem:

$$\begin{aligned} \max\nolimits_{X \in \{0,1\}^{MJ}} \quad & F(X) \\ \text{s.t.} \quad & \sum_{j=1}^{J} X_{jm} \le C_m, \;\; m = 1, 2, \ldots M \end{aligned} \tag{2.4}$$

In practice, the rate needs to be estimated from the requests, however for simplicity we assume it is known a priori. The above problem is shown to be NP hard, as it can not be solved in polynomial time and hence an approximate method of solving this is explained next. As a first step, a term is added to $F(X)$ such that the objective function remains unchanged. In other words, the above problem is equivalent to the following optimization problem

$$\begin{aligned} \max\nolimits_{X \in \{0,1\}^{MJ}} \quad & F(X) + \sum_{j,m} \epsilon_{jm} X_{jm}(1 - X_{jm}), \\ \text{s.t.} \quad & \sum_{j=1}^{J} X_{jm} \le C_m, \;\; m = 1, 2, \ldots M \end{aligned} \tag{2.5}$$

where $\epsilon_{jm} \ge 0, \forall\, 1 \le j \le J, 1 \le m \le M$. As a second step, the variable $X_{jm} \in \{0, 1\}$ is replaced by a continuous variable $\tilde{X}_{jm} \in [0, 1]$ to obtain an approximated version of the above problem:

$$\begin{aligned} \max\nolimits_{\tilde{X} \in [0,1]^{MJ}} \quad & F(\tilde{X}) + \sum_{j,m} \epsilon_{jm} \tilde{X}_{jm}(1 - \tilde{X}_{jm}) \\ \text{s.t.} \quad & \sum_{j=1}^{J} \tilde{X}_{jm} \le C_m, \;\; m = 1, 2, \ldots M \end{aligned} \tag{2.6}$$

where $\tilde{X} := \{\tilde{X_{jm}} : 1 \le j \le J, 1 \le m \le M\}$. Let the optimal solution to (2.5) and

(2.6) be X^* and $\tilde{X}^*$, respectively. Note that for any positive value $\epsilon_{jm} \geq 0$, we have

$$F(\tilde{X}^*) + \sum_{j,m} \epsilon_{jm} \tilde{X}^*_{jm}(1 - \tilde{X}^*_{jm}) \geq F(X^*). \tag{2.7}$$

Let $F_\Delta(\tilde{X}) := F(\tilde{X}) + \sum_{j,m} \epsilon_{jm} \tilde{X}_{jm}(1 - \tilde{X}_{jm})$. To simplify the presentation, vectorized $\tilde{X}$ can be written as $\tilde{X} := \{\tilde{Y}_1, \tilde{Y}_2, \ldots, \tilde{Y}_{JM}\}$, where $\tilde{Y}_s = \tilde{X}_{jm}$ such that $s = M(j-1) + m$, $1 \leq m \leq M, 1 \leq j \leq J$. Similarly, $\epsilon := \{\delta_1, \delta_2, \ldots, \delta_{JM}\}$, where $\delta_s = \epsilon_{jm}$ and $s = M(j-1) + m$. With the above notation $F(\tilde{X})$ is same as $F(Y)$, and hence $F_\Delta(\tilde{X})$ is same as $F_\Delta(\tilde{Y})$. Note that the relaxed problem in general is not convex, and hence δ_k's are chosen (equivalently $\epsilon'_{jm}s$) in such a way that the problem becomes a convex problem. The following Lemma provides a necessary condition for the problem to be a convex optimization.

Lemma I: The function $F_\Delta(\tilde{Y})$ (or equivalently $F_\Delta(Y)$) above is concave provided

$$\delta_k > \frac{1}{2} \sum_{j \neq m} |\mathcal{H}(\tilde{Y})_{jm}|, \ k = 1, 2, \ldots, JM. \tag{2.8}$$

where $\mathcal{H}(\tilde{Y})_{jm}$ is the jm-th term of the Hessian of -$F(\tilde{Y})$.[1]

Proof: To prove that the function $F_\Delta(\tilde{Y})$ is concave, it is sufficient to prove that -$F_\Delta(\tilde{Y})$ is convex. Let $\mathcal{H}_\epsilon(Y)$ denote the Hessian of the objective function -$F(Y)$. The proof is complete by showing that it is positive semidefinite or equivalently, all its eigenvalues are positive. From the objective function -$F_\Delta(\tilde{Y})$, it is seen that the diagonal entries of the Hessian matrix constitute $\epsilon := \{2\delta_1, 2\delta_2, \ldots, 2\delta_{JM}\}$. The Gershgorin disc theorem [84] states that for every square matrix A, its eigenvalues lie within a Gershgorin disc, *i.e.*,

$$|\gamma - A_{jj}| < \sum_{j \neq m} |A_{jm}|. \tag{2.9}$$

Thus, from the Gershgorin disc theorem, there exists an index k such that all the eigenvalues of the Hessian matrix $\mathcal{H}_\epsilon(Y)$ satisfy

[1] $\mathcal{H}(\tilde{Y})$ denotes the Hessian of the objective function.

$$\gamma \in \left[2\delta_k - \sum_{j \neq m} |\mathcal{H}_\epsilon(Y)_{jm}|, 2\delta_k + \sum_{j \neq m} |\mathcal{H}_\epsilon(Y)_{jm}|\right]. \tag{2.10}$$

The proof is complete by noting that all the eigenvalues of $\mathcal{H}_\epsilon(Y)_{jm}$ are positive if (2.8) is satisfied, and hence the function -$F_\Delta(\tilde{Y})$ is convex, consequently $F_\Delta(\tilde{Y})$ is concave.

From (2.7), it follows that $F(\tilde{X}^*) + \sum_{jm} \epsilon_{jm}\tilde{X}_{jm}(1 - \tilde{X}_{jm}) \geq F(X^*)$, where X^* is the solution of the original problem (2.5), and $\tilde{Y}^*$ (and, equivalently $\tilde{X}^*$) is the solution to the relaxed problem (2.6). Note that the resulting solution is fractional, and hence needs to be converted to a 0/1 solution. This is done using randomized rounding, as explained next.

Randomized Rounding: As shown in Raghavan and Thompson [85], this technique is probabilistic, i.e., with high probability, the algorithm will provide a solution in which the objective function takes on a value close to the optimum of the rational relaxation. The procedure for randomized rounding is as follows. The fractional optimal solution $\tilde{X}^*_{jm} \in [0, 1]$ obtained above is independently and randomly rounded to 1 with probability $\tilde{X}^*_{jm}$, and zero with probability $(1 - \tilde{X}^*_{jm})$. Let the resulting 0/1 solution be denoted by $\tilde{Z}^*_{jm}$. Note that the resulting objective function $F(\tilde{Z}^*)$ is random. The details of the proposed algorithm is shown in Algorithm 1.

Algorithm 1 Cache Placement Algorithm

1: **for** $\forall$ j, m **do**
2: Initialize $X_{jm} \leftarrow 0$;
3: **end for**
4: Solve (2.6) with ϵ_{jm} as obtained from (2.8).
5: **for** $\forall$ j, m **do**
6: Randomized rounding of $\tilde{X}^*_{jm}$ to get $\tilde{Z}^*_{jm} \in \{0, 1\}$ as described above.
7: **end for**

In the following, a theoretical guarantee of the proposed algorithm is presented. In particular, it is shown that on an average, the proposed cache hit is close to the cache hit of the optimal scheme.

Theorem 1: The proposed randomized algorithm satisfies the following inequality

$$F(X^*) - \mathbb{E}[F(\tilde{Z}^*)] \leq \frac{JM \max_{j,m} \delta_{(M(j-1)+m)}}{4}, \tag{2.11}$$

where the expectation $\mathbb{E}[\cdot]$ is with respect to the randomness involved in the rounding method.

Proof: Since $F(X^*) \leq F(\tilde{X}^*) + \sum_{j,m} \epsilon_{jm} \tilde{X}^*_{jm}(1 - \tilde{X}^*_{jm})$, and $\mathbb{E}[F(\tilde{Z}^*)] = F(\tilde{X}^*)$, we have $F(X^*) - \mathbb{E}[F(\tilde{Z}^*)] \leq \sum_{j,m} \epsilon_{jm} \tilde{X}^*_{jm}(1 - \tilde{X}^*_{jm})$. However, each term in the summation $X^*_{jm}(1 - \tilde{X}^*_{jm}) \leq 1/4$, which results in (B.14) after noting that $\max_{j,m} \epsilon_{jm} = \max_{j,m} \delta_{M(j-1)+m}$.

In the following it is shown that the proposed algorithm guarantee is better than the greedy algorithm, i.e. the proposed randomized algorithm has greater than (1/2)-approximation if it follows the following inequality:

$$\frac{\max_{j,m} \delta_{(M(j-1)+m)}}{F(X^*)} \leq \frac{2}{JM} \tag{2.12}$$

where $F(X^*)$ is the optimal solution. By rearranging the terms in Theorem 1, we get:

$$1 - \frac{JM \max_{j,m} \delta_{(M(j-1)+m)}}{4F(X^*)} \leq \frac{F(\tilde{X}^*)}{F(X^*)} \tag{2.13}$$

The greedy algorithm has the following approximation [86]:

$$(1/2) \leq \frac{F(X^*_{greedy})}{F(X^*)} \tag{2.14}$$

Thus from (2.13) and (2.14) it is seen that the proposed algorithm has a greater than (1/2)-approximation or a better bound when compared to the greedy algorithm, if it satisfies the following constraint:

$$\begin{gathered} (1/2) \leq 1 - \frac{JM \max_{j,m} \delta_{(M(j-1)+m)}}{4F(X^*)} \\ \frac{\max_{j,m} \delta_{(M(j-1)+m)}}{F(X^*)} \leq \frac{2}{JM} \end{gathered} \tag{2.15}$$

One such configuration where (2.15) is satisfied is when the number of files, J = 150, number of BSs, M = 30, value of $F(X^*) = 0.71$ and the maximum value of delta is 3 x 10^{-4}. Thus the proposed algorithm gives a better approximation than the (1/2)-approximation of the greedy algorithm.

2.3 Theoretical Guarantees

2.3.1 Complexity Analysis

The function $F(X)$ in (2.16) results in a computational complexity of $\mathcal{O}(d_{\mathtt{max}}JN)$ multiplications and $\mathcal{O}(JN)$ additions, where $d_{\mathtt{max}}$ is the maximum degree of the topology of the network presented earlier. Thus, the computational cost for greedy algorithm is $\mathcal{O}(d_{\mathtt{max}}JN)$ multiplications and $\mathcal{O}(JN)$ additions. And for exhaustive search, the computational cost is $\mathcal{O}(2^{d_{\mathtt{max}}JN})$ multiplications and $\mathcal{O}(2^{JN})$ additions. The proposed algorithm involves solving a convex 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 (see [87]), and (b) projecting the result onto the constraint region. The first step involves computing the gradient vector, which requires $\mathcal{O}(d_{\mathtt{max}}JN)$ number of multiplications, and $\mathcal{O}(JN)$ number of additions. The projection step involves finding the closest point in the constraint set from the query point; this, however, has a closed form expression, and hence requires $\mathcal{O}(1)$ additions and multiplications. 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). Note that the objective function in (2.16) has bounded Hessian, and hence Lipschitz continuous. Further, the constraint set in (2.5) is also bounded. Thus, applying Theorem 3.2 of [87], the number of times step (a) is executed is of the order of $\mathcal{O}(1/\zeta^2)$. Therefore, the total complexity is $\mathcal{O}(\frac{d_{\mathtt{max}}JN}{\zeta^2})$ multiplications and $\mathcal{O}(\frac{JN}{\zeta^2})$ additions. Comparing this with the greedy algorithm, it can be observed that the proposed algorithm performs significantly better than the greedy method at the expense of a slight increase in the complexity depending on the accuracy attained. Though exhaustive search algorithm is simple and gives the correct answer to this optimization problem, it goes over all the possible solutions and thus results in exponential complexity. Moreover, for an exhaustive search algorithm, when the possible solution set increases, it becomes impractical to go over all the points in the set and find an optimum point. Therefore, with respect to computational complexity the proposed algorithm performs better than the exhaustive search algorithm.

2.4 DNN based popularity prediction

Due to limited coverage range, each SBS can interact with only limited number of users, thus the dataset available in each SBS is sparse and hence it becomes difficult to predict the user popularity accurately. Since, the accuracy of prediction of popularity content directly affects the caching decision, proactive caching is one of the efficient methods to increase the cache hit rate by predicting the user's demand accurately. Further, in a heterogeneous network, the caching decision among multiple nodes should be made collaboratively and hence it suffers from dimensionality. Thus, in this section, we address the above challenges by applying a DL framework in heterogeneous network with a central BS, connected with multiple SBSs. We propose to equip the edge devices mainly BS with DL capabilities, such that the central BS is used to predict the demand requests of the users. A clustering based DL framework is used at the BS, which collects the information from all the SBSs. The clustering helps in analyzing the correlation in the past request patterns of all files. Then the clustering results are sent to the DL framework which further predicts the content. In this section, a heterogeneous network has been considered with M SBS with U multiple users, and is denoted by sets $\mathcal{B}$ and $\mathcal{U}$, respectively. The M SBSs are connected to a central BS, where all the computing takes place. The set of files is denoted by $\mathcal{N}$ and the user requests the files from the catalogue: $\{f_1, f_2, \ldots, f_N\}$. Each file is of variable size c_f. Each mth SBS is assumed to have varying cache size C_m. The variable $l_{im} = 1$ is used to denote the presence of a link between a user i and the SBS m, and $l_{im} = 0$, otherwise, hence induces a bipartite graph. The neighboring SBS of a user i is denoted by $\mathcal{N}_U(i) := \{m \in \mathcal{B} : l_{im} = 1\}$. Let the decision variable be denoted as $x_{fm} \in \{0, 1\}$, where $x_{fm} = 1$ if the file f is cached at SBS m, and $x_{fm} = 0$ otherwise. It is assumed that each user i makes requests for file f at a rate of λ_{if}. The average rate of cache hit $(F(\mathcal{X}))$ is given by the following equation:

$$F(\mathcal{X}) = \sum_{i,f} \lambda_{if} \left(1 - \prod_{m \in \mathcal{N}_U(i)} (1 - x_{fm}) \right), \tag{2.16}$$

where the set $\mathcal{X} := \{x_{fm} : 1 \leq f \leq N, 1 \leq m \leq M\}$ and $1 \leq i \leq U$. Maximizing the above function is an NP hard problem. Hence, we relax the constraint $x_{fm} \in \{0, 1\}$ to $x_{fm} \in [0, 1]$, and make the objective function concave by adding the term as follows:

$$\begin{aligned} &\max_{\mathcal{X} \in [0,1]^{MF}} \quad F(x) + \sum_{f,m} \epsilon_{fm} x_{fm} (1 - x_{fm}), \\ &\text{subject to} \quad \sum_{f=1}^{F} x_{fm} c_f \leq C_m, \forall m, \end{aligned} \tag{2.17}$$

where $\epsilon_{fm} \geq 0$. Since deep neural networks (DNNs) are known to have high accuracy, and are helpful in analyzing the temporal dynamics of input data, DNN is chosen for the estimation of the popularity profile. The next section explains how K-means clustering is used to analyze the correlation among the files.

2.5 K-means clustering

In this section, the time-varying content requests are clustered using the K-means clustering, which exploits the historical correlation among the files. Then a DL based framework is used to predict the content in each time slot for each cluster. Let $\bar{\boldsymbol{d}}(t) = [\bar{d}_1(t), \ldots, \bar{d}_N(t)]$ denote the predicted number of requests for file $f \in \mathcal{N}$ in time slot t. Let $\boldsymbol{p}_{t,f}$ denote a μ-dimensional feature vector of file f in time slot t, which is defined as follows:

$$\boldsymbol{p}_{t,f} = [d_f(t - \mu), d_f(t - \mu - 1), \ldots, d_f(t - 1)] \tag{2.18}$$

Thus the vector $\boldsymbol{p}_{f.t}$ contains the historical requests of file f during the previous time slot μ. With the help of clustering, the prediction algorithm can use the request information from other files also and hence increases the accuracy. Further, each feature vector is partitioned into C cluster. A K-means clustering algorithm is adopted for classification [88]. The similarity between each feature vector is found by the Euclidean distance. As proposed in [88], C points from the N feature vectors are selected and is denoted as $\{\boldsymbol{p}_1^c, \boldsymbol{p}_2^c, \ldots, \boldsymbol{p}_C^c\}$, be the initial cluster centers. At the beginning of each time slot $t \geq \mu + 1$, the cluster membership is determined according to the minimum Euclidean distance as follows:

$$l_{\boldsymbol{p}_{t,f}} = \underset{i \in 1,2,\ldots,C}{\operatorname{argmin}} ||\boldsymbol{p}_{t,f} - \boldsymbol{p}_i^c||_2^2 \tag{2.19}$$

At the end of each time slot t, the center of the cluster is updated by averaging the feature vector within the cluster as follows:

$$\boldsymbol{p}_i^c = \frac{\boldsymbol{p}_i^c U_i(t-1) + \sum_{I_{(\boldsymbol{p}_{t,f})=i, f \in \mathcal{N}}} \boldsymbol{p}_{t,f}}{U_i(t-1) + \sum_{I_{(\boldsymbol{p}_{t,f})=i, f \in \mathcal{N}}} 1}, \tag{2.20}$$

where $U_i(t)$ is the accumulated number of feature vector in the cluster i at the end of time slot t.

2.6 DNN based Popularity Prediction

A DNN consists of stacked neural networks and performs computations through many layers. These neural networks consists of layers with multiple neurons. Each neuron takes the input and generates an output, which is a nonlinear weighted sum of the output generated in the previous layer. The nonlinear function can be defined as:

$$f(z) = \begin{cases} 1, & \text{for } z < 0, \\ z, & \text{for } z \geq 0, \end{cases} \tag{2.21}$$

which corresponds to rectified linear unit function (Relu function) with z representing the input vector.

It is assumed that K layers are present in the DL model and the kth layer has N_k neurons. The output of each layer is denoted by the vector $\mathbf{o}$, and is written as follows:

$$\mathbf{o}^{(k+1)} = f^{(k)}(\mathbf{o}^{(k)}) = f(\mathbf{W}^{(k)}\mathbf{o}^{(k)} + \mathbf{b}^{(k)}), \tag{2.22}$$

where $\mathbf{W}^{(k)}$ is the $N_k \times N_k$ weight matrix and $\mathbf{b}^{(k)}$ is the $N_k \times 1$ bias vector. The output at the final layer, which is also a $N \times 1$ vector is generated as follows:

$$\bar{\boldsymbol{d}} = f(\mathbf{z}_{in}, \psi) = f^{(K)}(f^{(K-1)}(\dots f^{(1)}(\mathbf{z}_{in}))), \tag{2.23}$$

where $\psi = \{\mathbf{W}^{(1)}, \dots, \mathbf{W}^{(k)}, \mathbf{b}^{(1)}, \dots, \mathbf{b}^{(k)}\}$; $\bar{\mathbf{d}}$ and $\mathbf{z}_{in}$ are estimate of the popularity vector ($N \times 1$) and input vector ($N \times 1$) respectively. To measure the loss, mean squared error (MSE) function is written as follows:

$$L(\bar{\boldsymbol{d}}, \boldsymbol{d}) = ||\bar{\boldsymbol{d}} - \boldsymbol{d}||_2^2, \tag{2.24}$$

where $\boldsymbol{d}$ is the true content request vector ($N \times 1$), and $\bar{\boldsymbol{d}}$ is the estimated content request vector ($N \times 1$).

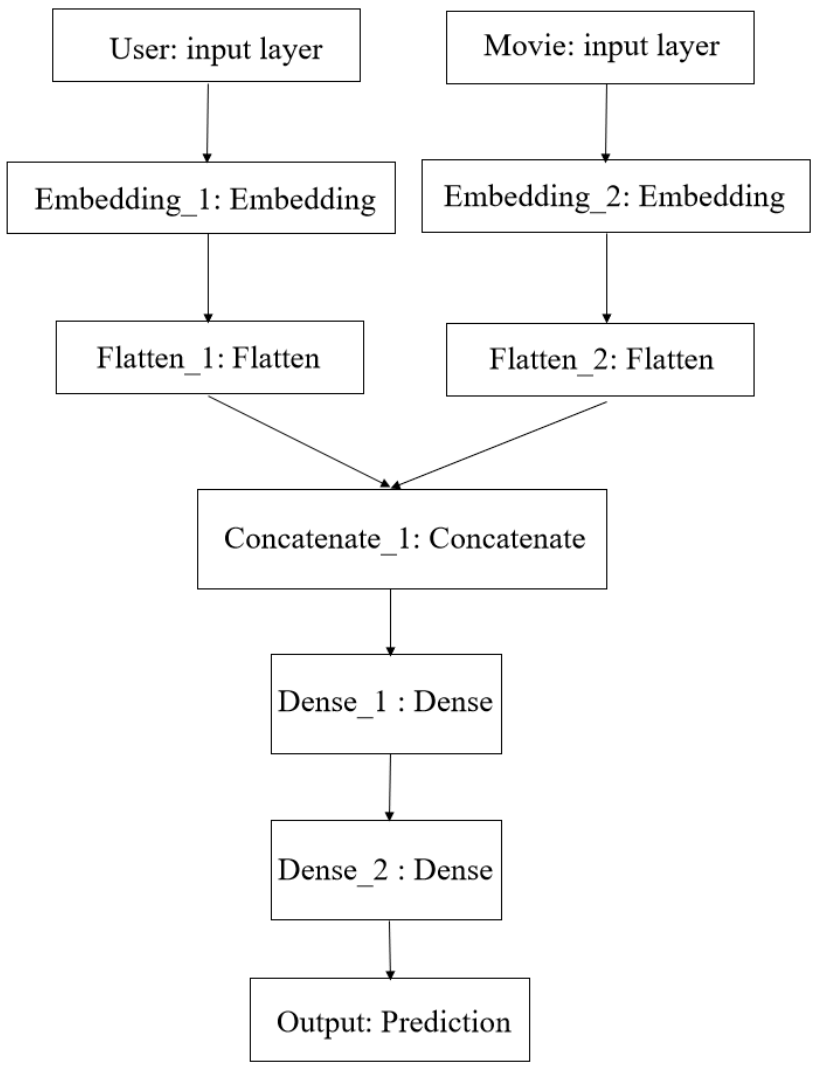


Figure 2.2: DNN based edge caching model.

To update the weight and bias, adaptive moment estimation (ADAM) is used as the optimizer, due to its superior performance [89]. The MovieLens data is used, consisting of ratings of various movies accumulated over decades. The central BS collects the data from the SBSs such as ratings for each movie-id, genre, name of the requested content, timestamp of the request, and type of the movie. The central BS collects all these information for a time duration T. The two input features of the data namely the movie name and the ratings are fed into the DL model. The model is trained such that it predicts how a user u would rate a file based on the available dataset of ratings, as shown in Fig. 2.2. The user input layer and the movie input layer are first processed parallelly and then later combined as explained next. Keras is used to define the three models: (a) user embedder that learns to represent each user's preference as a vector, (b) movie embedder that learns to represent each movie as a vector, and (c) rating model that concatenates user and movie embedding networks and applies a dense neural network to predict the rating. The flatten function reshapes the input vector to create a single long feature vector. The dense layer along with ReLU activation function is added after the concatenation. The output layer also consists of the dense layer, which is deeply connected to the preceding layer and the neural network layer in the dense layer introduces the non-linearity into the networks of neural networks so that the networks can learn the relationship between the input and output values. Keras functional API is used, in which the users have shared API model such that the inputs and embedding layers are shared. Therefore, any changes made to the user model are available in the rating model. The training iterations are then repeated over many epochs for loss minimalization.

2.7 Simulation Results

The simulation setup consists of SBSs and users distributed in a geographical area, and time slotted communication channel. The following two scenarios, namely fixed link and signal-to-interference-plus-noise ratio (`SINR`) based schemes, are considered in the simulation:

1. `Fixed Link Scheme:` The links between SBSs and users are uniformly and in-

dependently distributed in $\{0, 1\}$ with probability $1/2$, where 0 and 1 represent absence and presence of a link, respectively. Note that the absence/presence of a link dictates whether a user can download the file or not from the corresponding SBS.

2. `SINR Based Scheme:` Here, 5 SBSs and 35 users are distributed uniformly in a geographical area of radius 500m. The link between a SBS and a user exists if the corresponding `SINR` is greater than a threshold. The `SINR` is defined by

$$\texttt{SINR} := \frac{Pd_{im}^{-2}|h_{im}|^2}{\sum_{b:b\neq m} Pd_{ib}^{-2}|h_{ib}|^2 + \sigma^2}, \tag{2.25}$$

where P indicates the power used by SBSs, h_{im} is the Rayleigh fading channel between user i and SBS m, and d_{im} is the distance between user u and the SBS m. Hence, the links form a dynamic bipartite graph. Further, the threshold indicates the minimum rate at which a file will be transferred to the user from the SBS, and hence the reciprocal of the rate indicates the delay. In the simulation, $\tau := \frac{1}{\log(1+\texttt{SINR})}$ is used as a measure of the delay between a SBS and a user. However, when the requested file is absent, a backhaul fetching delay of $\alpha \times \tau$ is added to the down link delay of τ, i.e., the overall delay when the file is absent is $(\alpha + 1)\tau$. In the simulation, $\alpha = 5$.

In both the scenarios, the popularity of the files is assumed to be Zipf distributed with the corresponding parameter $\theta = 1.2$. Without loss of generality, each file requested by the user is assumed to be of the same size. The rate at which files are request is assumed to be the same, and the requests follow Poisson distribution with rate of 20×10^{-3} files per second for all SBSs and users, and the requested file follows the Zipf distribution. In other words, the request for a file j from any user i follows a Poisson distribution with rate $\lambda_{ij} := 2 \times 10^{-3} p_j$, where $p_j = \frac{1/j^\theta}{\sum_j 1/j^\theta}$.

`Fixed Link Scheme:` Figs. 2.3 and 2.4 show plots of the average cache hit performance of (a) the proposed caching algorithm, (b) greedy algorithm [90], (c) LRU and LFU algorithms [91], and (d) exhaustive search based caching algorithm versus cache size for various parameters of the system. In Fig. 2.3, the total number of files, the number of users and SBSs are 10, 15, and 3, respectively. While for Fig. 2.4, the total number of files, the number of users and SBSs are 50, 35, and

5, respectively. The results are obtained by averaging over different realizations of links. It is clear from Figs. 2.3 and 2.4, that the proposed caching algorithm performs much better compared to LRU/LFU, and greedy algorithms. Further, from Fig. 2.3, it is important to note that the proposed caching algorithm is reasonably close to the exhaustive search. Specifically, to achieve the same performance as that of the proposed scheme with 35 users and a cache size of 20 (see Fig. 2.4), the greedy algorithm requires approximately 40 percentage more memory to achieve the same average cache hit. Note that by scaling the number of users, and SBSs, the performance gap in terms of memory increases significantly. Further, the average cache hit performance decreases with the increase in the number of users, as observed from Figs. 2.3 and 2.4.

`SINR Based Scheme:` Fig. 2.5 shows the average delay versus the cache size for (a) proposed caching scheme, (b) greedy, (c) LRU and (d) LFU algorithms. In Fig. 2.5, the total number of files, the number of users, SBSs and threshold value for `SINR` are 50, 35, 5 and 12dB, respectively. The average delay is calculated as explained earlier in this section. It is important to note that this model captures more realistic scenarios such as the effect of distance, the power used etc., on the overall caching performance of the cellular network. Thus it is clear from the figure that the proposed algorithm outperforms the greedy, LRU and LFU algorithms in terms of the average cache hit and average delay.

The total number of SBS and users are assumed to be as 5 and 20 respectively for the proposed DL based caching framework. File sizes are assumed to take value between $[2, 10]$ units. To know the nature of user access patterns, the exploratory data analysis on the MovieLens dataset is performed. For training and testing the performance of prediction models, the dataset covering 100,000 ratings from 1000 users on 1700 movies (N) are used. The dataset is split into two parts, wherein the 80% is used for training and the rest 20% is used for testing.

The DNN consists of an input layer, five hidden layers and an output layer. The five hidden layers have 1000, 1000, 3000, 3000 and 1000 neurons. Further, we assume 1000 neurons in the input layer and 1700 neurons in the output layer. We consider 1000 epochs for DNN training.

Fig. 2.6 compares the DL model with the GPM, PPM, AR prediction model,

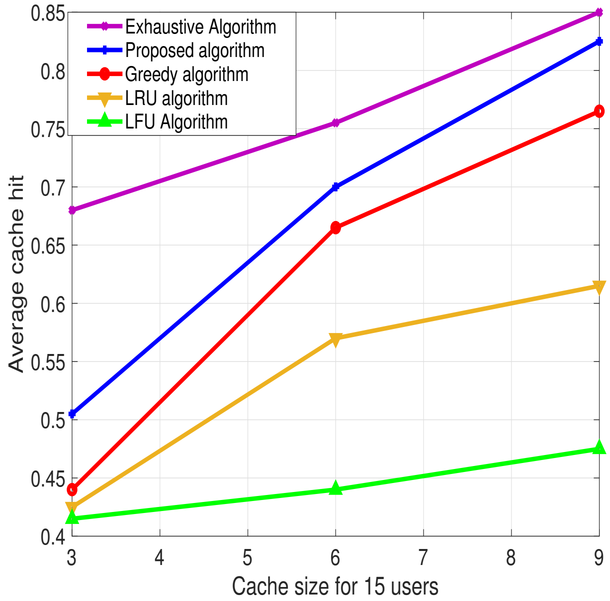


Figure 2.3: Average delay versus cache size with 15 users for the fixed link scenario.

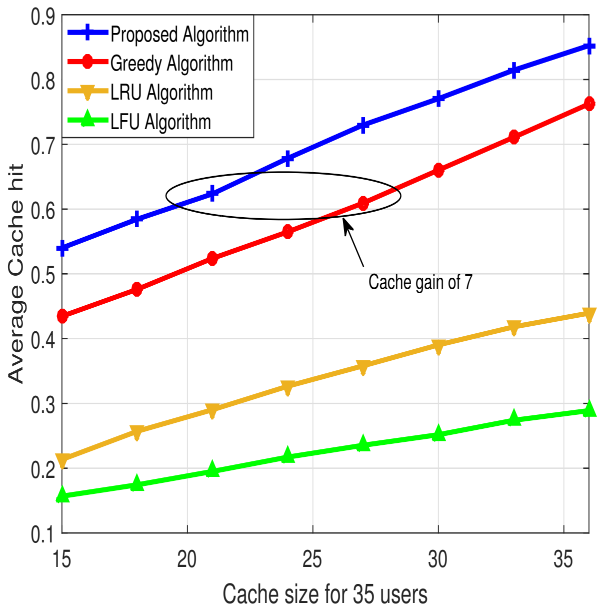


Figure 2.4: Average delay versus cache size with 35 users for the fixed link scenario.

weighted FTL, weighted FoReL, and mean guessing prediction model. It is observed from Fig. 2.6 that DL has the high prediction accuracy, since it has the least MSE value i.e. the value of MSE for the proposed DL algorithm is 0.025, while the MSE value for GPM, PPM, mean guessing, FTL, FoReL and AR are 0.06, 0.07, 0.09. 0.16, 0.36 and 0.45 respectively. Hence, DL based model outperforms the GPM,

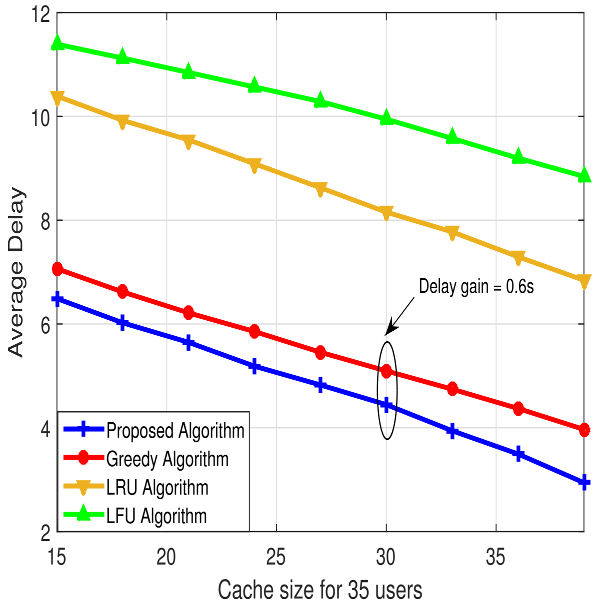


Figure 2.5: Average delay versus cache size with 35 users for the SINR based scenario.

PPM, FTL, AR, mean guessing and FoReL for MSE.

Fig. 2.7 compares the MSE metric with respect to content library size ($\log_2 N$). The content library size is varied from 2^2 to 2^{10}. The DL model is compared with GPM, PPM, FTL, AR, mean guessing and FoReL. From Fig. 2.7, we can observe that as the library size is varied, the MSE for the DL model is lower than the other prediction models. For a library size of 6 the MSE value of the proposed DL algorithm is 0.022, while for GPM, PPM, mean guessing, FTL, FoReL and AR, the MSE values are 0.079, 0.085, 0.095. 0.17, 0.5 and 0.8 respectively.

Fig. 2.8 shows the relationship between the hidden layers in DL model versus MSE. The content library size is fixed at 500 files (N). When the number of hidden layer is one, the model is referred to as a simple neural network (NN). The value of MSE for the proposed algorithm is 0.0335 when the number of hidden layers is 8 and for the simple neural network, the MSE value is 0.0426. Hence, as we increase the number of layers in the DNN model, the MSE reduces further and results in better performance.

Fig. 2.9 shows the average cache hit versus different cache size (C) and $C_m = C$, $\forall\ m$. The content library size is fixed at 500 files (N). In this plot, the comparison is done with linear prediction model (AR), and we can observe that DL model outperforms the linear prediction model in terms of average cache hit.

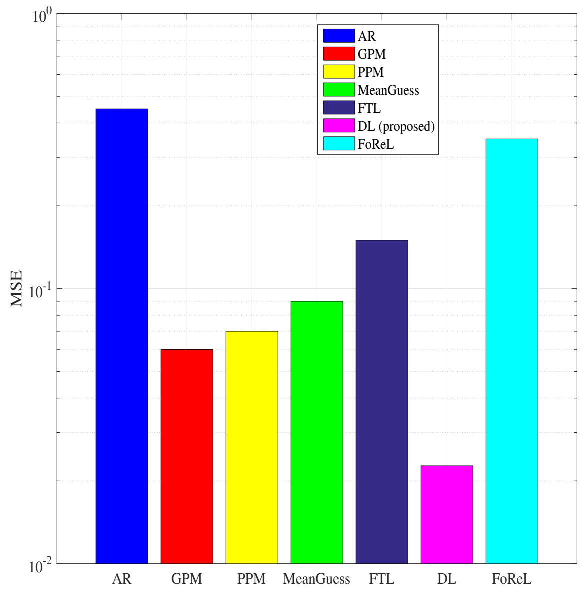


Figure 2.6: Average MSE for various prediction models.

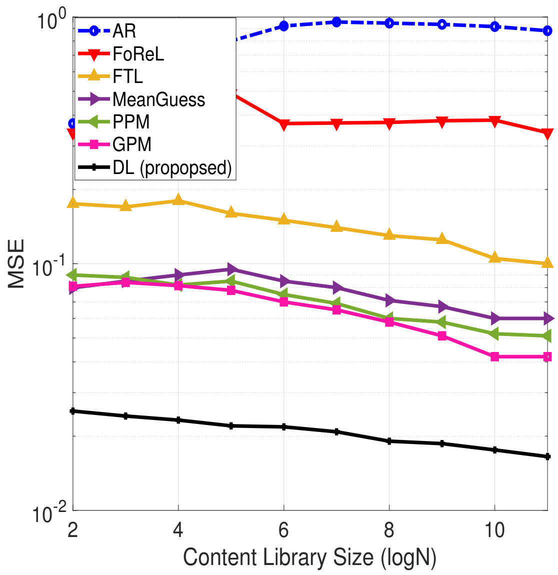


Figure 2.7: MSE versus content library size.

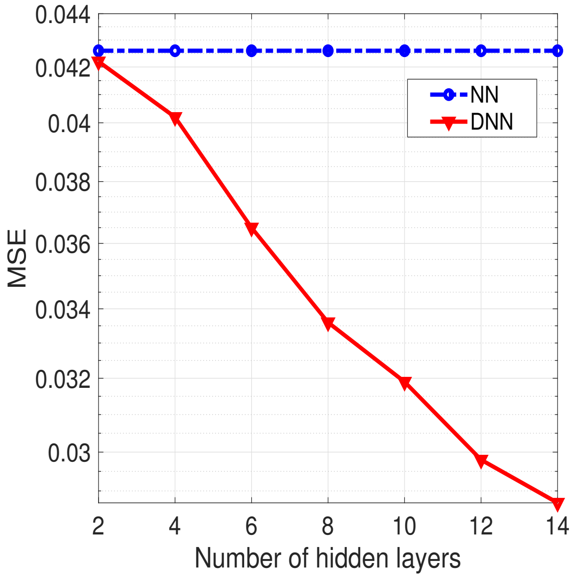


Figure 2.8: MSE versus number of hidden layers.

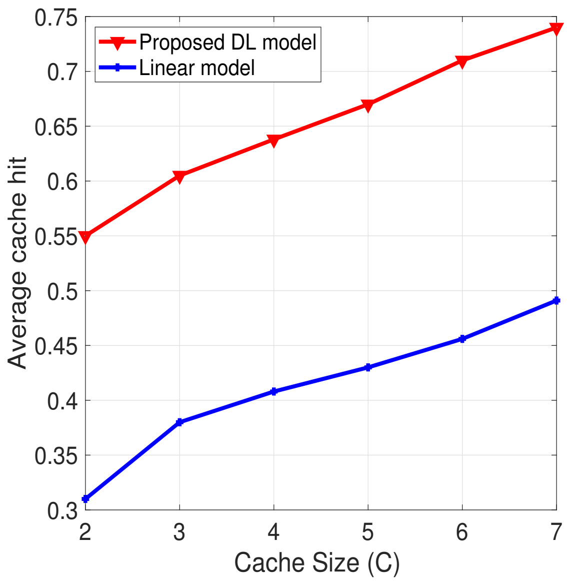


Figure 2.9: Average cache hit versus content library size.

2.8 Summary

In this chapter, an approximate cache placement algorithm in a cellular network is presented, where SBSs store contents from a large database. Using average cache hit as a metric, it is observed that solving for optimal cache placement that maximizes the cache hit is NP hard. Thus, the problem is relaxed to a convex optimization problem, and a solution is presented with performance guarantees. The approximation is performed such that the perturbation term that makes the problem convex, is small. Simulation results demonstrated that the proposed algorithm significantly outperforms the existing greedy based caching strategies, LRU and LFU algorithms in terms of the average cache hit and average delay. The proposed algorithm is shown to have a similar computational complexity compared to the conventional greedy algorithm with better overall performance. Further, a DL based prediction model is also proposed to predict the content popularity profile. K-means clustering framework at the central BS is used which collects the information from the SBSs and the training is performed on the MovieLens dataset in the offline mode. Simulation results support that the proposed DL model outperforms the existing models for MSE and average cache hit. However, the system model is far from a realistic practical scenario. Hence to bridge this gap, a distributed learning based caching model is analyzed in the next chapter.

Chapter 3

Federated Learning

3.1 Overview

In the previous chapter, mobile edge caching in a cellular network with a given topology and storage size in each SBS is looked into. The problem of maximizing the cache hit with respect to the cache placement strategy (which takes values in $\{0, 1\}$) subject to storage size constraint in each SBS turns out to be NP hard, and hence, an approximation is proposed to the problem. However, this model does not take into factor non-stationarity of popularity profile and correlation across time.

In this chapter, assuming structured cache placement, high probability bounds on the conditional average cache hits are derived using Martingale difference equation [79]. 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.

In the literature, 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 [92, 93] with edge computing, deployment of small cells to offload wireless data from a macro BS, integrating existing WiFi access points to share the load, distributed cache replacement strategy based on Q-learning, to name a few

[94], [95], [96]. 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 SBSs or the end users [97–99]. Past works in caching include the classical work from the point-of-view of information theory by Niesen et al. [100] (also, see [101]), combinatorial optimization approach [102], energy efficient caching of files in a D2D network (see [102] - [104]), and proactive caching strategy, as in [63]. In [105], 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 [106], 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 [59]-[62]). 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 [63]. The authors in [64] showed that a good hit rate under non-stationary demands can be achieved through a TTL based algorithm. Some past work assumed that there is a stationary caching policy such as LRU [65], climb [66, 67], and k-LRU [68] 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 [69]. Many of these works result in a regret of $\Omega(T)$. In [70], 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 MAMB perspective, which results in a regret of $\mathcal{O}(\log T)$. In [71], function approximation based reinforcement learning approaches are proposed for a massive MIMO based network. In [72] 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 authors in [73] 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 [74], active learning approach is used to learn the content popularities to design an accurate content request prediction model. In [75], 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 [76]). 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 [77]. There are several heuristics such as LRU, LFU and 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 [107].

One of the most promising distributed learning algorithms is the emerging federated learning (FL) framework (see [108–111]). 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 chapter 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 [110, 112–114]. 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 [115, 116]. 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 $O(1/\sqrt{T})$ [117, 118]. 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 [119]. Towards filling shortcoming in the existing work, in this chapter, 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 chapter are used to provide guarantees for LRFU, and its variants under non-stationary and correlated demands.

In this chapter, 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 chapter:

- In this chapter, assuming structured cache placement, high probability bounds on the conditional average cache hits are derived using Martingale difference equation [79]. 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 chapter, 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 [120] 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 [121, 122]. The results show that the proposed algorithms significantly outperform the existing algorithms as well as the equal weight algorithms.

3.2 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. 3.1 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 [93]. 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).

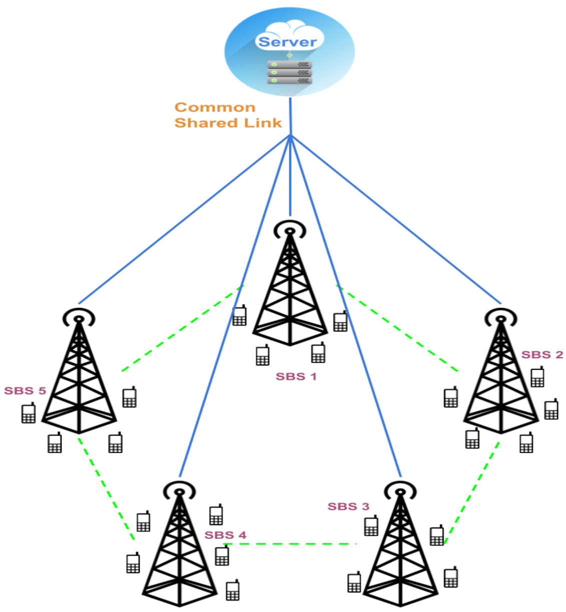


Figure 3.1: System model showing multiple SBSs connected to users with limited cache memory.

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 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 chapter. 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.

3.2.1 Caching Policy

At each SBS b, the cache placement is assumed to happen at the end of every time slot. In this chapter, 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 chapter.

3.2.2 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

$$\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) := \sum_{f=1}^{N} \sum_{u \in \mathbb{U}_b(t)} d_{f,u}(t) \pi_{b,f} \mathcal{L}_f. \tag{3.1}$$

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., $\sum_b \mathbb{E}\{\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t})\}$, where the average is with respect to the global demands.[1] 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 the expectation is conditioned on the local demands, i.e., $Z_{b,1}^{T}$. Thus, the following problem needs to be solved

$$\begin{aligned} &\max_{\boldsymbol{\pi}_{b,t}} \sum_b \mathbb{E}\{\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\} \\ &\text{subject to} \sum_f \pi_{b,f,t}\mathcal{L}_f \leq C. \end{aligned} \tag{3.2}$$

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 [119]. 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 chapter, 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.

3.3 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

[1] 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.

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 [123], 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 chapter 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 [77, 119]. 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

$$\bar{\boldsymbol{\pi}}_{b,T+1} := \sum_{t=T-\tau+1}^{T} \alpha_{b,t} \boldsymbol{\pi}_{b,t}, \tag{3.3}$$

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

$$\boldsymbol{\pi}_{b,T+1}^{(av)} := w_b^{T+1} \bar{\boldsymbol{\pi}}_{b,T+1} + \sum_{b' \in \mathcal{N}_b} w_{j_b(b')}^{T+1} \bar{\boldsymbol{\pi}}_{b',T+1}, \tag{3.4}$$

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 chapter, 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

$$\mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) := \sum_{b' \in \mathcal{N}_b} w_{j_b(b')}^{T+1} \Delta_{b,T+1}(\boldsymbol{\alpha}_{b,T+1}, \boldsymbol{\alpha}_{b',T+1}), \tag{3.5}$$

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

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

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

$$\mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T}) := \sup_{\boldsymbol{\pi}_{b,t} : t = T-\tau+1,\ldots,T} \left| \sum_{t=T-\tau+1}^{T} \alpha_{b,t} \Delta \bar{\mathcal{R}}_{T,t}(\boldsymbol{\pi}_{b,t}) \right|. \tag{3.6}$$

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

$$\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t}) \;\; := \;\; \sup_{\boldsymbol{\pi}_{b,t}^{*}} \sum_{t=T-\tau+1}^{T} \mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}^{*}) - \sum_{t=T-\tau+1}^{T} \mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}). \tag{3.7}$$

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 chapter is stated below, and the corresponding proof is presented in Appendix A.1.

Theorem 3.3.1. *Given weights and a sequence of caching strategies as in* (3.4) *that is adapted to* $Z_{b,1}^T$*, with a probability of at least* $1-\delta$, $\delta > 0$, *the following two bounds hold:*

$$\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}}) \mid Z_{b,1}^T\right] \;\geq\; \sum_{t=T-\tau+1}^{T} \alpha_{b,t}\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) - \mathcal{E}^{(1)}_{b,T}, \tag{3.8}$$

where $\mathcal{E}^{(1)}_{b,T} := H_{max}\|\boldsymbol{\alpha}_{b,T}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}} + M_{b,T+1}(\boldsymbol{w}_{\neq b,T}) + \mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T})$, H_{max} *is the maximum cache hit, and*

$$\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}}) \mid Z_{b,1}^T\right] \;\geq\; \sup_{\boldsymbol{\pi}_{b,t}} \sum_{t=T-\tau}^{T} \alpha_{b,t}\bar{\mathcal{R}}_{t,T}(\boldsymbol{\pi}_{b,t}) - \mathcal{E}^{(2)}_{r,T}. \tag{3.9}$$

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*

$$\begin{aligned}\mathcal{E}^{(2)}_{b,T} \;&:=\; 2H_{max}\|\boldsymbol{\alpha}_{b,T}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}} + M_{b,T+1}(\boldsymbol{w}_{\neq b,T}) + \frac{2Reg_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau} \\ &\quad + H_{max}\sum_{t=T-\tau+1}^{T}\left|\alpha_{b,t} - \frac{1}{\tau}\right| + 2\mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b.T}) + \gamma.\end{aligned} \tag{3.10}$$

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 3.3.2. *Given equal weights, i.e.,* $\alpha_{b,t} = 1/\tau \;\forall\; t$, *and a sequence of caching strategies as in* (3.4) *that is adapted to* $Z_{b,1}^T$*, with a probability of at least* $1-\delta$, $\delta > 0$, *the following two bounds hold:*

$$\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}}) \mid Z_{b,1}^T\right] \;\geq\; \frac{1}{\tau}\sum_{t=T-\tau+1}^{T} \mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) - \mathcal{E}^{(1)}_{b,T}, \tag{3.11}$$

where

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

H_{max} is the maximum cache hit, and

$$\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}}) \mid Z^T_{b,1}\right] \geq \sup_{\boldsymbol{\pi}_{b,t}} \frac{1}{\tau} \sum_{t=T-\tau}^{T} \bar{\mathcal{R}}_{t,T}(\boldsymbol{\pi}_{b,t}) - \mathcal{E}^{(2)}_{r,T}. \tag{3.12}$$

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

Proof. Given in Appendix A.1. □

A few observations are in order with reference to Theorem 3.3.1. The term $H_{\mathtt{max}} \sum_{t=T-\tau}^{T} \left|\alpha_{b,t} - \frac{1}{\tau}\right|$ 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.

3.3.1 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 (3.4), the resulting cache hit is lower bounded by the expression in (3.9) with high probability. Now, at time slot $T+1$, the goal is to choose the individual strategy $\boldsymbol{\pi}_{b,t}$ to construct $\boldsymbol{\pi}^{(av)}_{b,T+1}$ as in (3.4) such that the right hand side of (3.9) consisting of regret and discrepancy terms to be maximized.[2] 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 (3.4). The first step would be to find the regret

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

minimizing caching strategy by solving the following optimization problem

$$\min_{\boldsymbol{\pi}_{b,t}:\mathbf{1}^T\boldsymbol{\pi}_{b,t}\leq C}\left[\sup_{\boldsymbol{\pi}^*_{b,t}}\sum_{t=T-\tau+1}^{T}\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^*_{b,t})-\sum_{t=T-\tau+1}^{T}\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t})\right] \tag{3.13}$$

to get a sequence of caching policies denoted by $\boldsymbol{\pi}^R_{b,t}$ $\forall$ 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 (3.9) 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

$$\begin{aligned}\alpha_{b,t},\overset{\sup}{\boldsymbol{w}_{\neq b,T}}\quad &\sum_{t=T-\tau+1}^{T}\alpha_{b,t}\mathcal{R}_{b,t}(\boldsymbol{\pi}^R_{b,t})-a\widehat{\mathbb{D}}_{b,T}(\boldsymbol{\alpha}_b)-\\ &b\widehat{\mathtt{M}}_{b,T+1}(\boldsymbol{w}_{\neq b,T})+\lambda\sum_{t=T-\tau+1}^{T}\left|\alpha_{b,t}-\frac{1}{\tau}\right|\end{aligned} \tag{3.14}$$

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

$$\widehat{\mathbb{D}}_{b,T}(\boldsymbol{\alpha}_b):=\sup_{\boldsymbol{\pi}_{b,t}:\boldsymbol{\pi}_{b,t}\mathbf{1}^Tx\leq C}\left|\sum_{t=T-\tau+1}^{T}\alpha_{b,t}\sum_{f}\psi^{(f)}_{\tau_1,\tau_2}(t,T)\pi_{b,f}(t)\right|, \tag{3.15}$$

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

$$\begin{aligned}\widehat{\mathtt{M}}_{b,T+1}(\boldsymbol{w}_{\neq b,T}):=\frac{1}{\tau}\sum_{b'\in\mathcal{N}_b}w^{T+1}_{j_b(b')}\Bigg[&\sum_{s,l=T-\tau+1}^{T}\alpha_{b,s}\mathcal{R}_{b,l}(\pi^R_{b,s})-\\ &\sum_{s,l=T-\tau+1}^{T}\alpha_{b',s}\mathcal{R}_{b',l}(\pi^R_{b',s})\Bigg].\end{aligned} \tag{3.16}$$

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 (3.14) 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 (3.15), and used in (3.14) 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 (3.14) 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 2 Algorithm for Federated Learning based Caching

```
for T = 1, 2, ..., and SBS  b ∈ 𝔹 do
    Run regret minimization as in (3.13) to get a sequence
    π^R_{b,t}, t = 1, ..., T
    Call Subroutine (T, τ, π^R_{b,t}, π^R_{b',t} for all b' ∈ N_b) .
    to get π_{b,T+1}
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`

$$\pi^t_{b,f}(k+1) = \pi^t_{b,f}(k) + 2\eta_k g, \tag{3.17}$$

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:`

$$\begin{aligned} \alpha_{b,t}(k+1) &= \alpha_{b,t}(k) + \beta_k \Big[\mathcal{R}_{b,t}(\pi^R_{b,t}) - \Theta \\ &- \nabla_\alpha \widehat{\mathtt{M}}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) \Big], \end{aligned} \tag{3.18}$$

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:

$$\begin{aligned} w^{T+1}_{j_b(b')}(k+1) &= w^{T+1}_{j_b(b')}(k) - \frac{2\gamma_k}{\tau} \Bigg[\sum_{s,l=T-\tau+1}^{T} \alpha_{b,s}(k) \\ &\times \mathcal{R}_{b,l}(\pi^R_{b,s}) - \sum_{s,l=T-\tau+1}^{T} \alpha_{b',s}(k) \mathcal{R}_{b,l}(\pi^R_{b',s}) \Bigg] \end{aligned} \tag{3.19}$$

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 it-

eration to all neighboring SBSs, and go back to step 1 **else;** return

$$\bar{\boldsymbol{\pi}}_{b,T+1} = w_b^{T+1}(k+1)\sum_{t=T-\tau+1}^{T}\alpha_{b,t}(k+1)\boldsymbol{\pi}_{b,t}^R + \sum_{b'\in\mathcal{N}_b} w_{j_b(b')}^{T+1}(k+1)\sum_{t=T-\tau+1}^{T}\alpha_{b',t}(k+1)\boldsymbol{\pi}_{b',t}^R \tag{3.20}$$

- **end for**

Since the above algorithm is a modification of gradient descent algorithm,[3] the convergence can be proved in a similar manner to that of classical gradient descent. In the following subsection, the proposed FL based heuristics algorithm for caching mechanism design that takes into account neighboring SBSs requests is detailed.

3.3.2 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:

$$\min_{\pi:\sum_f \pi_f l_f \le L} \hat{F}_k(\pi), \tag{3.21}$$

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

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

presents an analysis of the LRFU scheme. To the best of authors knowledge, this analysis is the first of its kind in the literature.

Algorithm 3 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:

$$\hat{d}_{f,k}^{(t)} := \frac{1}{\tau} \sum_{s=t-\tau}^{t-1} d_{f,k}^{(s)}, \text{ and } \bar{\boldsymbol{\pi}}_{\mathcal{N}_k}^{(t)} := \frac{1}{|\mathcal{N}_k|} \sum_{j \in \mathcal{N}_k} \hat{\pi}_{k,t-1}^{*}. \tag{3.22}$$

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

$$\hat{\pi}_{k,t}^{*} := \arg\min_{\boldsymbol{\pi}} \hat{F}_{k,t}(\pi) + \lambda \|\boldsymbol{\pi} - \bar{\boldsymbol{\pi}}_{\mathcal{N}_k}^{(t)}\|^2, \tag{3.23}$$

  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

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 SBS b, the following optimization problem is solved:

$$\max_{\boldsymbol{\pi}: \sum_f \pi_f \mathcal{L}_f \leq C} \sum_f \pi_f \hat{d}_{b,f,t} \mathcal{L}_f, \tag{3.24}$$

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 $\boldsymbol{\pi}_{b,t}^{\texttt{LRFU}}$. Before stating the main theorem, the following notions of discrepancy (similar to discrepancy described earlier) will be used to state the main result.

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

$$\mathbb{D}_{GL,T}(\tau) := \sup_{\boldsymbol{\pi}_{b,t}: t=T-\tau+1,\ldots,T} \left| \frac{1}{\tau} \sum_{t=T-\tau+1}^{T} \left(\Delta \bar{\mathcal{R}}_{T,t} \right) \right|, \tag{3.25}$$

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 (3.2), 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 3.3.3. *For the LRFU caching strategy $\boldsymbol{\pi}_{b,t}^{LRFU}$, with a probability of at least $1-\delta$, $\delta > 0$, the following bound hold:*

$$\begin{aligned} \sum_{f} \boldsymbol{\pi}_{b,t}^{LRFU} \hat{d}_{b,f,t} \mathcal{L}_f \quad &\leq \quad \sup_{\boldsymbol{\pi_b}} \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z_{b,1}^{t-1} \right] + \mathbb{D}_{GL,t}(\boldsymbol{\alpha}_b) + \mathbb{D}_{b,t}(\boldsymbol{u}_\tau) \\ &+ \quad H_{max} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2\log\frac{1}{\delta}}{\tau}} \end{aligned} \tag{3.26}$$

where $\mathbb{D}_{b,t}(\boldsymbol{u}_\tau)$ is as defined in (3.6) *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* (3.25).

Proof. Given in Appendix A.2. □

It is clear from the above theorem that in the iid demands scenario, the right hand side will be $\sup_{\boldsymbol{\pi_b}} \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z_{b,1}^{t-1}\right] + H_{\mathtt{max}} \|\boldsymbol{\alpha_{b,T}}\|_2 \sqrt{\frac{2\log\frac{1}{\delta}}{\tau}}$. 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 (3.2). 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.

Complexity Analysis

- Algorithm I: The function in (3.2) $\mathcal{O}(d_{\mathtt{max}}^2 \tau N)$ multiplications and $\mathcal{O}(d_{\mathtt{max}} \tau N)$ additions, where $d_{\mathtt{max}}$ 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 [87], the number of times step (a) is executed is of the order of $\mathcal{O}(1/\zeta)$. Therefore, the total complexity is $\mathcal{O}(\frac{d_{\mathtt{max}}^2 \tau N}{\zeta})$ multiplications and $\mathcal{O}(\frac{d_{\mathtt{max}} \tau N}{\zeta})$ additions.

- Algorithm II : The function in (3.21) requires $\mathcal{O}(d_{\mathtt{max}} \tau N)$ multiplications and $\mathcal{O}(d_{\mathtt{max}} \tau N)$ additions, where $d_{\mathtt{max}}$ 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 $\mathcal{O}(\frac{d_{\mathtt{max}} \tau N}{\zeta})$ multiplications and $\mathcal{O}(\frac{d_{\mathtt{max}} \tau N}{\zeta})$ 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.

3.4 Simulation Results

The simulation setup consists of five SBSs with multiple users connected to each of the SBS as shown in Fig. 3.1. 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.[4] 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. 3.3.2, (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. 3.2 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. 3.2 shows the sum cache hit rate of all the SBSs summarizing the trends in all the SBSs. It is clear from the figure that

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

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).

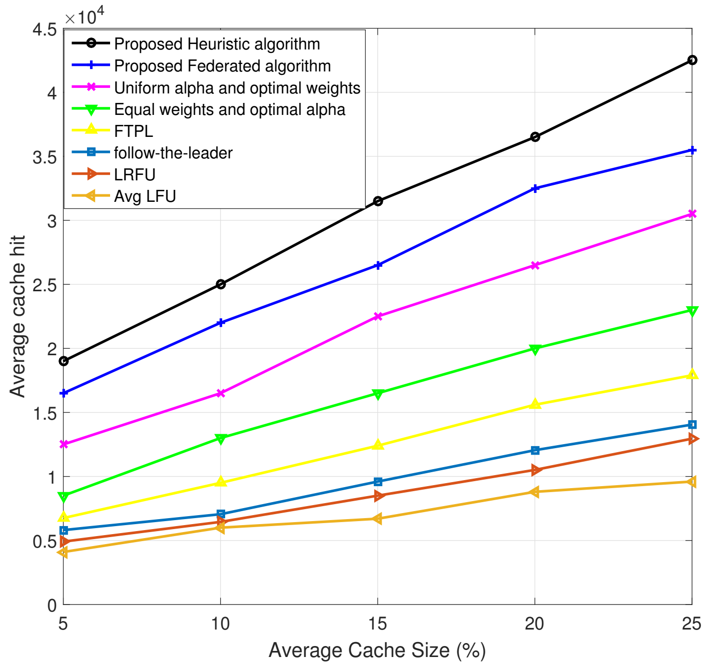


Figure 3.2: Average sum cache hit of all SBSs versus cache size.

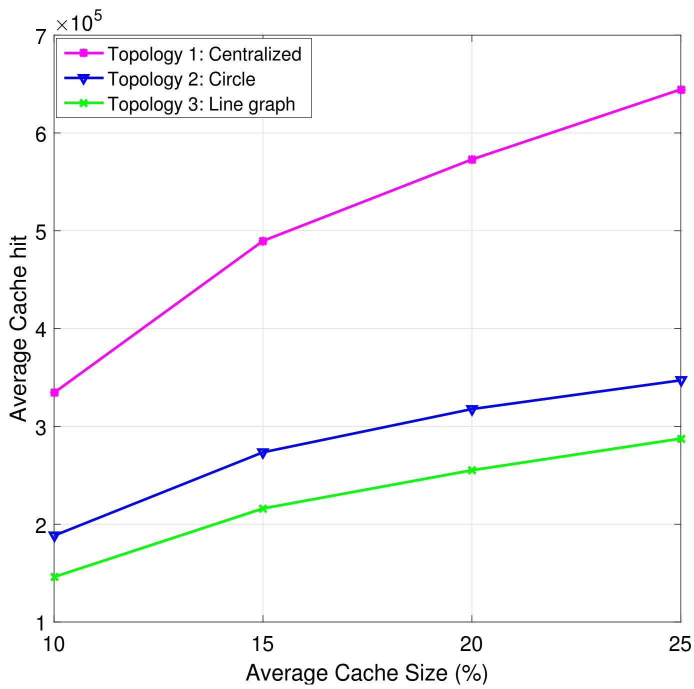


Figure 3.3: Average cache hit of all SBSs versus cache size for different topologies for Algorithm I.

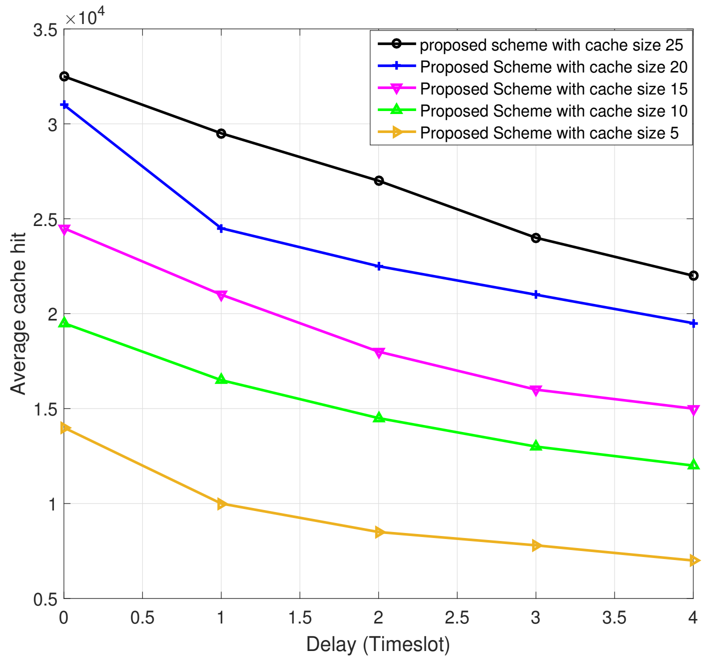


Figure 3.4: Average cache hit of all SBSs versus cache size for different delay for Algorithm I.

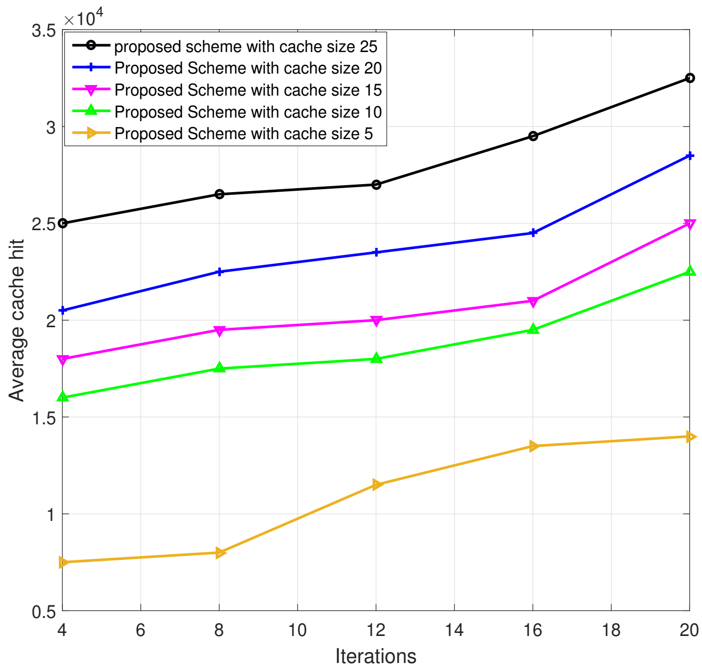


Figure 3.5: Average cache hit of all SBSs versus cache size for different iterations for Algorithm I.

The cache hit performance of the proposed caching algorithm (FL based) under various inter-SBS topologies are shown in Fig. 3.3. 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. 3.3 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 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. 3.4, the average cache hit is plotted as the delay is varied for circular topology. As seen from Fig. 3.4 as the delay increases while exchanging the information among the SBSs, the average cache hit eventually reduces. In Fig. 3.5, average cache hit for different iterations (number 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.

3.5 Summary

In this chapter, assuming structured cache placement, high probability bounds on the conditional average cache hits are derived using Martingale difference equation. 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. Simulation results demonstrated that the proposed federated algorithm significantly outperforms the existing FTPL, LRFU and average LFU algorithms in terms of the average cache hit. Further, simulation results support that the proposed federated model works well in the centralized topology scenario. However, this chapter confines to a specific distributed scenario, wherein the joint optimization problems are not addressed and this is addressed by considering a general scenario of mobile edge caching in the next chapter.

Chapter 4

Bayesian Learning for Joint Optimization

4.1 Overview

In the previous chapter, a FL based caching strategy was proposed which takes into account the correlation between the 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. However, this model is specific to distributed algorithms and does not consider the recommendation based systems. To fill this gap, in this chapter, a general setting has been used, wherein the joint optimization of caching and recommendation is looked into. Two estimation procedures, point and Bayesian estimation are provided. A probabilistic model using Bayesian inference based on Dirichlet distribution is proposed. Specifically, the influence of recommendation on the popularity profile is modelled using a conditional probability distribution. A high probability guarantee on the estimated caching and recommendation strategies is provided. Irrespective of the estimation method, it is shown that with a probability of $1 - \delta$ the proposed caching and recommendation strategy is ϵ close to the optimal solution.

In the literature, to further enhance the user's quality of experience, mobile edge computing (MEC) jointly with recommendation has been proposed which stores the popular contents close to the edge devices beforehand, and hence reduces the delay and alleviates the backhaul congestion [10]. While, on the other hand rapidly growing file sizes, reducing cache sizes (compared to the traditional content delivery networks), and unpredictable user demands make the task of caching algorithms even more difficult. For example, the total data generated by Google per day is in the order of PBs, while installing 1TB in every small cell in the heterogeneous network will only shift less than 1 % of the data for even one content provider. To overcome these issues, it has been observed that user demands are increasingly driven by recommendation based systems. Recommendation based on an individual's preference have become an integral part of e-commerce, entertainment and other applications. The success of recommender systems in Netflix and Youtube shows that 80% of hours streamed at Netflix and 30% of the overall videos viewed owes to recommender systems [124, 125]. With recommendation the user's request can be nudged towards locally cached contents, and hence resulting in lower access cost and latency.

Usually, the recommender systems and the caches at the wireless network are owned by different entities. The recommender engines are managed through applications that interact with users, while the caching systems are controlled by the wireless network operators. This further motivates the need for coordination of the different mechanisms towards optimization of user and network-centric performance measures. Firstly, caching reduces backhaul traffic and brings the content closer to the user. Secondly, with recommendation, quality-of-experience (QoE) of users enhances hence benefiting the caching mechanism.

The idea of content caching is borrowed from the domains of wired network, where contents were replicated at CDNs that are closer to the end users. A similar approach was adopted in wireless network due to the increasing network traffic [17]. However, the content placement in the wireless network largely depends on the user behaviour and file popularity. The recent success of integration of artificial intelligence in the wireless communications has further led to better understanding of user behaviors and the characteristics of the network [126]. Especially the edge networks can now predict the content popularity profile hence increasing the average cache hit. The high accuracy in prediction by the neural networks has resulted in many of the content popularity prediction models, such as, collaborative filtering with recurrent neural networks [127], the stack auto encoder [128], deep neural networks [129] and others. However, the local content popularity profile need not match the global prediction by the central server. Many of the recent works have proposed the edge caching strategies by learning the user preferences and content popularity [130–132]. Context awareness helps in classifying the environment, hence enabling the intelligent decisions at the edge to select the appropriate contents, for instance, Chen *et al.* [133] presented the edge cooperative strategy based on neural collaborative filtering. Jiang *et al.* [134] used the offline user preferences data and statistical traffic patterns and proposed an online content popularity tracking algorithm. However, the offline data will not be available always. The multiarm bandit (MAB) problem has also been extensively investigated, [135, 136] which addresses the tradeoff between the exploration and exploitation to achieve the maximum reward. These works assume that users have identical preferences and no correlation amongst the data, which may not be the case in a practical situation.

For example, consider a SBS to which mobile users are connected. The idea is to recommend files which adequately matches the users' preference rather than the file which ranks as the top most relevant content, thus nudging the content access patterns of users and achieving higher caching efficiency and better users' QoE. However, while jointly considering caching and recommendation, computational complexity of optimizing both has to be taken into account. Recommendation and caching can be individually approximated, however the joint optimization is NP hard without an optimal decomposition [13]. Hence, in this chapter, two estimation procedures namely point estimation and Bayesian estimation have been proposed for optimization [137–139]. Further, considering a real-time system, the estimation has been applied to a more practical heterogeneous network. A typical heterogeneous network consists of a macro base station (MBS) and various low power nodes [140]. Similarly, in this work we have considered a heterogeneous system for the estimation model.

In order to determine which content needs to be cached in SBS, it is important to predict the user's content probability request. One way of estimating the user requests, is to take the average of the instantaneous requests, which is also equivalent to maximum likelihood estimation (MLE) [141]. This performs well as long as the size of the request sample is large. Since the number of content requests is comparatively small, it is difficult to use the conventional *frequentist* approach where the request is observed over an interval of time and inference is drawn based on the data. A typical SBS receives 0.1 requests/content/day and hence MLE provides an inaccurate estimation of content popularity in local caches. Hence, with this motivation, a probabilistic model for content requests has also been proposed. Bayesian approach is used to obtain a prior belief and is also used to determine the parameters. In Bayesian estimation, probability is expressed as a degree of belief in an event. Bayesian estimation helps in providing a framework to overcome the overfitting problem. To predict the future requests, a multivariate distribution i.e. Dirichlet distribution is used to model the prior probabilities. In Bayesian statistics, Dirichlet distribution is a popular conjugate prior for the multinomial distribution. Dirichlet distribution is used to appropriately model the prior knowledge about the requests and hence improve the accuracy of popularity estimation. One use of Dirichlet prior is that it avoids over-fitting the estimate of a multinomial distribution from a small amount of data. For recommendation systems, where only few user ratings are available for a particular file, the MLE of the rating distribution can be very inaccurate and hence cannot estimate the true popularity. While on the other hand, when only few user ratings are present, Dirichlet prior prevents the estimate from being overfitted. The popularity distribution starts with a Dirichlet prior and is updated whenever new data is received. With this knowledge, the caching and recommendation action is updated accordingly. Simulation results further verify that the proposed method performs better than the existing benchmark algorithms in terms of delay, average cache hit and throughput.

The main contributions of the chapter are summarized as follows:

- For the first time two estimation procedures, `Point` and `Bayesian estimation` are provided. A probabilistic model using Bayesian inference based on Dirichlet distribution is proposed. Specifically, the influence of recommendation on the popularity profile is modelled using a conditional probability distribution. A high probability guarantee on the estimated caching and recommendation strategies is provided. Irrespective of the estimation method, it is shown that with a probability of $1-\delta$ the proposed caching and recommendation strategy is ϵ close to the optimal solution.

- A high probability bound on the regret for `Bayesian estimation` method is provided. To compare and contrast the obtained regret bound, we also derive a regret bound on the genie aided scenario using the `Point estimation` method. First, we consider the `Point estimation` case and provide lower bound on the waiting time that is required to achieve an error ϵ optimal solution with high probability. Assuming a genie aided scenario, we prove a regret bound of $\mathcal{O}(T^{2/3}\sqrt{\log T})$. Using the Martingale difference technique, we prove a high probability bound on the regret achieved by the `Bayesian estimation` method. In particular, we show that $\mathcal{O}(\sqrt{T})$ regret is achievable, which is better than the genie aided scenario.

- The proposed `Point estimation` and `Bayesian estimation` are further extended to a heterogeneous network consisting of M SBSs with a central MBS. The MBS computes an estimate of the probability transition matrix (PTM) and gives an update to each of the SBSs. For computing guarantees, first M is taken as two for the sake of simplicity and a lower bound on the waiting time is provided. The same is generalized to a heterogeneous network consisting of M SBSs, the estimation of the probability matrix has been derived and useful insights are drawn.

- Numerical results are presented and it is shown that the proposed algorithm outperforms the existing least recently/frequently used (LRFU), least frequently used (LFU) and least recently used (LRU) in terms of average cache hit.

To the best of the author's knowledge, this is the first time in the literature that `point estimation` and `Bayesian estimation` has been used to obtain a lower bound on the training time in the wireless edge caching. In Section 4.2, we present the system model followed by the problem statement. The two estimation methods: `point estimation` and `Bayesian estimation` are also developed in Section 4.2. In Section 4.3.1, theoretical guarantees have been provided, in which a lower bound on the training time is derived which estimates the caching and recommendation strategies which are ϵ close to the optimal strategies. Further, in Section 4.4, the system model has been generalized to a heterogeneous network consisting of M SBSs and to make the analysis simpler, first the model is developed using M as two and then is generalized for a M SBS model. Simulation results are provided in Section V.

4.2 System Model and Problem Statement

The considered system model is shown in Fig. 4.1 which consists of a wireless network with single SBS serving multiple users. The heterogeneous scenario, which is an extension of the problem, is presented later in the chapter. The SBS is assumed to have a storage capacity of F contents of equal sizes from a catalog of contents $\mathcal{C} := \{1, 2, \ldots, F\}$. The requests are assumed to be independent-and-identically-distributed (iid) across time. Since, recommending a file influences the users requests', thus, by jointly optimizing caching and recommendation improves the overall performance significantly. Using a conditional probability distribution, the influence of recommendation on the popularity of a file is modelled, i.e., p_{ij} represents the probability that a user requests a file i given content j was recommended. This leads to a PTM denoted by $\mathbf{P} \in \mathbb{R}^{F \times F}$. The matrix P is assumed to be fixed across time slots and the time is assumed to be slotted. For the sake of simplicity, it is assumed that at least one file every slot is requested by each user N in the network. Initially, the recommendation and caching strategies are sampled from a uniform distribution. The set of caching and recommendation strategies are denoted by

$$\mathcal{C}_{c,r} := \{(\boldsymbol{u}, \boldsymbol{v}) \in [0, 1]^{2 \times F} : \boldsymbol{u}^T \mathbf{1} \le c, \boldsymbol{v}^T \mathbf{1} \le r\}, \tag{4.1}$$

where r and c are recommendation and cache constraints, respectively. The variables $\boldsymbol{v}$ and $\boldsymbol{u}$ are the recommendation and caching probabilities. In the sequel, the strategy is defined by the pair $(\boldsymbol{u}, \boldsymbol{v})$. For a given strategy $(\boldsymbol{u}, \boldsymbol{v}) \in \mathcal{C}_{c,r}$, the average cache hit is given by $\boldsymbol{u}^T \mathbf{P} \boldsymbol{v}$, in the t-th time slot. If the matrix $\mathbf{P}$ is known apriori, the optimal strategy is found by solving $\max_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}^T \mathbf{P} \boldsymbol{v}$. However, the matrix $\mathbf{P}$ is not known, and therefore it is estimated from the demands. The variable $d_i^{(t)}$ denotes the demand and is defined as the total number of requests in the time slot t for the file i. Following two estimation procedure will be addressed in this chapter

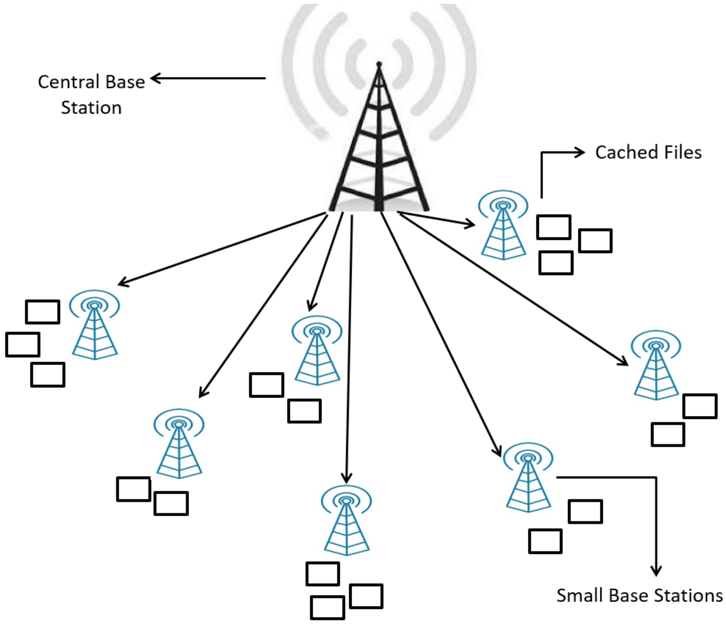


Figure 4.1: Distributed caching in a cellular network.

using the above demand:

- `Point estimation:` In this method, the demands until t time slots is used to compute an estimate of the matrix $\mathbf{P}$. During the first t time slots, recommendation and caching are done with probabilities q and p, respectively. Let $v_j^t = 1$ if file j was recommended in slot $t-1$, and zero otherwise. The recommendation and caching constraints in (4.1) are satisfied by, $q := r/F$ and $p := C/F$. We can see that as the value of t increases, the estimate becomes better, and hence results in better performance. The estimate of the ij-th entry of the $\mathbf{P}$ matrix is given as follows

$$\hat{p}_{ij}^{(t)} := \frac{\sum_{s=0}^{t-1} d_i^{(s)} v_j^{s-1}}{N \sum_{s=0}^{t-1} v_j^{s-1}}. \tag{4.2}$$

The corresponding estimate of the matrix $\mathbf{P}$ be denoted by $\hat{\mathbf{P}}^{(t)}$. Since $\mathbb{E}\{\hat{p}_{ij}^{(t+1)} | \sum_{s=0}^{t-1} v_j^s > 0\} = p_{ij}$, the point estimator is an unbiased estimator. The recommendation and caching probabilities are selected by solving the following optimization problem, after t time slots:

$$(\hat{\boldsymbol{u}}_{o,t}^*, \hat{\boldsymbol{v}}_{o,t}^*) = \arg \max_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}^T \hat{\mathbf{P}}^{(t)} \boldsymbol{v}. \tag{4.3}$$

- `Bayesian estimation:` In this method, for a given time slot, a prior on the rows of the matrix $\mathbf{P}$ is obtained. Dirichlet distribution is chosen as a prior which is based on the past demands. The Dirichlet pdf is a multivariate generalization of the Beta distribution, and is given by

$$f(x_1, \ldots, x_K, \alpha_1, \ldots, \alpha_K) = \frac{\Gamma(\sum_{j=1}^{K} \alpha_j)}{\prod_{j=1}^{K} \Gamma(\alpha_j)} \prod_{j=1}^{K} x_j^{\alpha_j - 1}, \tag{4.4}$$

$\alpha_j \geq 0 \; \forall \; j$. The Dirichlet distribution is used as a conjugate pair in Bayesian analysis and the shape of the distribution is determined by the parameter α. If $\alpha_j = 1 \; \forall \; j$, then it leads to an uniform distribution. The higher the value of α_j, the greater the probability of x_j. The notation $(x_1, x_2, \ldots, x_K) \sim \texttt{Dirch}(\alpha_1, \alpha_2, \ldots, \alpha_K)$ indicates that $(x_1, x_2, \ldots, x_K)$ is sampled from a Dirichlet distribution in (4.10). An estimate in the beginning of the time slot t of the i-th row of the matrix $\hat{\mathbf{P}}$ is given by

$$(\hat{\mathbf{P}}^{(t)})_i \sim \texttt{Dirch}\left(\sum_{i=1}^{t-1} d_1^{(i)} v_j^{i-1}, \sum_{i=1}^{t-1} d_2^{(i)} v_j^{i-1}, \sum_{i=1}^{t-1} d_F^{(i)} v_j^{i-1} \right), \tag{4.5}$$

where v_j^{i-1} is as defined earlier with v_j^0 sampled from $\{0, 1\}$ with probability $q := r/F$. Using the above estimate, the recommendation and caching strategy at time slot t is obtained by solving the following problem:

$$(\hat{\boldsymbol{u}}_{b,t}^*, \hat{\boldsymbol{v}}_{b,t}^*) = \arg \max_{(\boldsymbol{u}, \boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}^T \hat{\mathbf{P}}^{(t)} \boldsymbol{v}. \tag{4.6}$$

A procedure to find a strategy is given in **Algorithm** 1.

In the sequel, the strategy is defined by the pair $(\boldsymbol{u}, \boldsymbol{v})$. For a given strategy $(\boldsymbol{u}, \boldsymbol{v}) \in \mathcal{C}_{c,r}$, the average cache hit at the SBS k is given by $\boldsymbol{u}^T \mathbf{P}_k \boldsymbol{v}$. If the matrix $\mathbf{P}_k$ is known apriori at the SBS k, the optimal strategy can be found by solving $\max_{(\boldsymbol{u}, \boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}^T \mathbf{P}_k \boldsymbol{v}$. However, the matrix $\mathbf{P}_k$ is unknown, and therefore it needs to be estimated from the demands. Let the variable $d_{k,i}^{(t)}$ denotes the demand at the SBS k, and is defined as the total number of requests in the time slot t for the file i. Since the demands arrive in a sequential manner, the PTMs need to be estimated and updated in an online fashion. The performance of such algorithms is measured in terms of regret. As apposed to adversarial setting of online learning, here we have assumed that there is an underlying distribution from which the requests are generated, namely the PTM. Accordingly, the following provides the definition of the regret, which depends on the PTM:

Definition 1. *(Regret) The regret at the SBS k after T time slots with respect to any sequence of strategy $(\boldsymbol{u}_{k,t}, \boldsymbol{v}_{k,t})$, $t = 1, 2, \ldots, T$ is defined as*

$$\mathtt{Reg}_{k,T} \;:=\; T\boldsymbol{u}_{k,*}^T \mathbf{P}_k \boldsymbol{v}_{k,*} - \sum_{t=1}^{T} \boldsymbol{u}_t^T \mathbf{P}_k \boldsymbol{v_t}, \tag{4.7}$$

where $(\boldsymbol{u}_{k,*}, \boldsymbol{v}_{k,*}) := \arg\max_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}^T P_k \boldsymbol{v}$ is the optimal strategy at the SBS k.

The goal of the paper is to come up with a strategy at each SBS that results in a minimum regret. Any caching and recommendation algorithm, either directly or indirectly estimates the PTM $\boldsymbol{P}_k$. Therefore, the above goal translates to finding better estimates of the PTMs. In this paper, we consider two approaches to finding the estimates of the PTMs in an online fashion, namely (i) `Point estimation` and (ii) `Bayesian estimation` methods. Further, when there are multiple SBSs, any given SBS can potentially improve its estimate of PTM by fusing the estimates of the other SBSs. The question of how to fuse the estimates that results in a good regret is another question to which we will shed some lights. In the following section, we provide caching and recommendation algorithms for single SBS scenario, and provide theoretical guarantees for them.

4.3 Joint Caching and Recommendation for Single SBS Scenario

In this section, we assume single SBS, and therefore, $M = 1$ as shown in Fig. 4.2. As mentioned above, using the demands obtained at the SBS, an estimate of the PTM matrix is computed using either `Point estimation` or `Bayesian estimation` method. Given an estimate $\hat{\boldsymbol{P}}_k^{(t)}$, the caching and recommendation strategies will be found by solving the following problem[1]

[1]For theoretical analysis, we assume that the problem can be solved exactly.

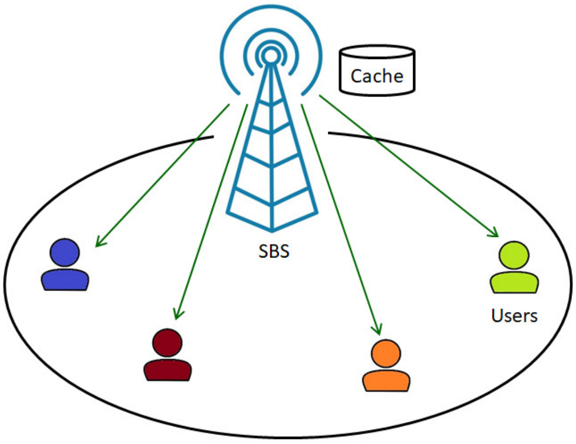


Figure 4.2: Distributed caching in a cellular network assuming single SBS.

$$(\hat{\boldsymbol{u}}^*_{o,t}, \hat{\boldsymbol{v}}^*_{o,t}) = \arg \max_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}^T \hat{\boldsymbol{P}}^{(t)}_k \boldsymbol{v}. \tag{4.8}$$

Now, we present the following two estimation procedures used in this paper.

- `Point estimation:` Given any SBS k, in this method, the demands until t time slots is used to compute an estimate of the matrix $\mathbf{P}_k$. During the first t time slots, recommendation and caching are done with probabilities q and p, respectively. Let $v^t_{jk} = 1$ if file j was recommended in slot $t-1$, and zero otherwise. The recommendation and caching constraints in (4.1) are satisfied by choosing $q := r/F$ and $p := c/F$. We can see that as the value of t increases, the estimate becomes better, and hence results in better performance. The estimate of the ij-th entry for SBS k of the $\mathbf{P}_k$ matrix is given by

$$\hat{p}^{(t)}_{ij,k} := \frac{\sum_{s=0}^{t-1} d^{(s)}_{ik} v^{s-1}_{jk}}{N \sum_{s=0}^{t-1} v^{s-1}_{jk}}. \tag{4.9}$$

  The above is a naive estimate of the probabilities by using a simple counting of events. The corresponding estimate of the matrix $\mathbf{P}_k$ be denoted by $\hat{\boldsymbol{P}}_k^{(t)}$. Since $\mathbb{E}\{\hat{p}^{(t+1)}_{ij,k} | \sum_{s=0}^{t-1} v^s_{jk} > 0\} = p_{ij,k}$, the point estimator is an unbiased estimator. After every time slot t, the recommendation and caching probabilities are selected by solving the optimization problem in (4.8) with $\hat{\boldsymbol{P}}^{(t)}_k$ obtained in (4.9). A procedure to find a strategy is given in **Algorithm** 1.

- `Bayesian estimation:` In this method, for a given time slot, the rows of the matrix $\boldsymbol{P}^{(t)}_k$ is sampled using a prior distribution, which is updated based on the past demands. This may tradeoff the exploration versus exploitation while solving for the optimal recommendation and caching strategies. Here, Dirichlet distribution is chosen as a prior. The Dirichlet pdf is a multivariate generalization of the Beta distribution, and is given by

$$f(x_1, x_M, \alpha_1, \alpha_M) = \frac{\Gamma(\sum_{j=1}^{M} \alpha_j)}{\prod_{j=1}^{M} \Gamma(\alpha_j)} \prod_{j=1}^{M} x_j^{\alpha_j - 1}, \tag{4.10}$$

$\alpha_j \geq 0 \; \forall \; j$. The Dirichlet distribution is used as a conjugate pair in bayesian analysis and the shape of the distribution is determined by the parameter α_j. If $\alpha_j = 1$ for all j, then it leads to a uniform distribution. The higher the value of α_j, the greater the probability of occurence of x_j. The notation $(x_1, x_2, \ldots, x_M) \sim \texttt{Dirch}(\alpha_1, \alpha_2, \ldots, \alpha_M)$ indicates that $(x_1, x_2, \ldots, x_M)$ is sampled from a Dirichlet distribution in (4.10). An estimate in the beginning of the time slot t of the i-th row of the matrix $\hat{\boldsymbol{P}}_k^{(t)}$ is given by

$$(\hat{\boldsymbol{P}}_k^{(t)})_i \sim \texttt{Dirch}\left(\sum_{s=1}^{t-1} d_{1k}^{(s)} v_{jk}^{s-1}, \sum_{i=1}^{t-1} d_{2k}^{(s)} v_{jk}^{s-1}, \sum_{s=1}^{t-1} d_{Fk}^{(s)} v_{jk}^{s-1} \right), \tag{4.11}$$

where v_{jk}^{s-1} is as defined earlier with v_{jk}^{0} sampled from $\{0,1\}$ with probability $q := r/F$. After every time slot t, the recommendation and caching probabilities are selected by solving the optimization problem in (4.8) with $\hat{\boldsymbol{P}}_k^{(t)}$ obtained in (4.11). A procedure to find a strategy is given in **Algorithm** 4.

Algorithm 4 Caching and recommendation algorithm (one SBS case)

```
procedure POINT ESTIMATION/BAYESIAN ESTIMATION
    û*_{b,0} ~(i.i.d.) {0,1} from p = c/F, & v̂*_{b,0} ~(i.i.d.) {0,1} from q = r/F.
    Recommend & cache according to v̂*_{b,0} & û*_{b,0}.
    for t = 0, 1, ..., T do
        Observe demands d_{ik}^{(t)} in slot t.
        Compute P̂_k^{(t)} from (4.9) for point estimation
        Compute P̂_k^{(t)} from (4.11) for Bayesian estimation
        Solve (4.8)
        Use (v̂*_{b,t}, û*_{b,t}) to recommend and cache.
    end for
end procedure
```

In the following subsection, we provide theoretical guarantees of the above algorithm.

4.3.1 Theoretical Guarantees

In this section, we provide a high probability bound on the regret for both `Point estimation` and `Bayesian estimation`. For the `Point estimation` case, we start by providing a lower bound on the waiting time which is ϵ close to the optimal caching strategies. The result will be of the following form: With a probability of at least $1-\delta$, the following holds provided $t \geq$ `constant`

$$\boldsymbol{u}_t^T \mathbf{P}_k \boldsymbol{v}_t \geq \sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}^T \mathbf{P}_k \boldsymbol{v} - \epsilon, \tag{4.12}$$

where $(\boldsymbol{u}_t, \boldsymbol{v}_t)$ is the caching strategy obtained by using any algorithm. The constant ϵ depends on various parameters, as explained next. Towards stating theoretical guarantees, the following definition is useful.

Definition 2. *(Covering number) A set $\mathcal{N}_\epsilon := \{(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2), \ldots, (\boldsymbol{x}_{\mathcal{N}_\epsilon}, \boldsymbol{y}_{\mathcal{N}_\epsilon})\}$ is said to be an ϵ-cover of $\mathcal{C}_{c,r}$ if for any $(\boldsymbol{u}, \boldsymbol{v}) \in \mathcal{C}_{c,r}$, there exists $(\boldsymbol{x}_j, \boldsymbol{y}_j) \in \mathcal{N}_\epsilon$ for some j such that $\|\boldsymbol{u} - \boldsymbol{x}_j\| \leq \frac{\epsilon}{8}$ and $\|\boldsymbol{v} - \boldsymbol{y}_j\| \leq \frac{\epsilon}{8}$.*

The following theorem provides a bound that is useful to provide the final result.

Theorem 4.3.1. *For a given estimate of the PTM denoted $\hat{\boldsymbol{P}}_k^{(t)}$ using `Point estimation` or `Bayesian estimation`, the following holds good*

$$\Pr\left\{ \sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}^T \mathbf{P}_k \boldsymbol{v} - \boldsymbol{u}_t \mathbf{P}_k \boldsymbol{v}_t \geq \epsilon \right\} \leq |\mathcal{N}_\epsilon| \Pr\left\{ \|\widehat{\Delta P}^{(t)}\|_F \geq \frac{\epsilon}{4\kappa r c} \right\}, \tag{4.13}$$

where $(\boldsymbol{u}_t, \boldsymbol{v}_t)$ is the output of the Algorithm 4 at time t, and $\widehat{\Delta P}^{(t)} := \mathbf{P}_k - \hat{\boldsymbol{P}}_k^{(t)}$. Further, $\kappa > 0$ is some constant.

Proof. Given in Appendix B.1. □

Using the above result, in the following, we provide our first main result on the performance of the `Point estimation` scheme.

Theorem 4.3.2. *Using (4.8) for caching and recommendation in slot t, for any $\epsilon > 0$, with a probability of at least $1-\delta$, $\delta > 0$, $(\boldsymbol{u}_{o,t}^*)^T \mathbf{P}_k \boldsymbol{v}_{o,t}^* \geq \sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}^T \mathbf{P}_k \boldsymbol{v} - \epsilon$ provided*

$$t \geq \frac{1}{q\left(1 - \exp\{-\frac{N\epsilon^2}{8\kappa^2 F^2 c^2 r^2}\}\right)} \log \frac{2\mathcal{N}_\epsilon F^2}{\delta}. \tag{4.14}$$

Proof. Given in Appendix B.2. □

As we know, the regret achieved by the `Point estimation` method is $\mathcal{O}(T)$ as it incurs non-zero constant average error for all the slots t satisfying (4.14). In this method, the estimation of PTM is done using the samples obtained from the first t slots, and the caching strategy is decided based on this estimate. However, an improvement over this is to continuously update the estimates, and the caching/recommendation strategies. Instead of analyzing the regret for this, we assume that at any time slot t, a genie provides an estimate of the PTM as in (4.9) to compute the caching/recommendation strategies, and provide the corresponding approximate regret bound. In particular, in Appendix B.3, we show that a regret of $\mathcal{O}(T^{2/3}\sqrt{\log T})$ can be achieved through the genie aided `point estimation` method. As opposed to `point estimation` method, here (genie aided) the caching/recommendation decisions can be made using an improved estimate of the PTM in every time slot leading to a better regret. We use this as a benchmark to compare the regret obtained from the `Bayesian estimation` method. In the next section, we extend the result to two SBS scenario, and use the insights to extend it further in the later part of the paper to any number of SBSs.

4.3.2 Bayesian Estimation: Single SBS Scenario

Note that unlike the analysis for `point estimation`, in this case, the strategies are correlated across time. This makes the analysis non-trivial. The approach we take is to convert a sequence of random variables (function of caching and recommendation across time) into a Martingale difference. This enables us to use the Azuma's inequality, which can be used to provide high probability result on the regret. In the following, we provide the result.

Theorem 4.3.3. *For the* `Bayesian estimation` *in Algorithm 4, for any* $\epsilon > 0$, *with a probability of at least* $1-\delta$, $\delta > 0$, *the following bound on the regret holds*

$$\mathtt{Reg}_T \;\leq\; 2rc\max_{ij} p_{ij}|\mathcal{N}_\epsilon|\sum_t \exp\left\{\frac{-8\psi_t^2}{cr\,|\mathcal{N}_\epsilon|^2\,\bar{\sigma}_t^2(t)}\right\} + 2\sum_t \psi_t + \sqrt{128r^2c^2T\log(1/\delta)}, \tag{4.15}$$

where $\alpha_{ij}^{(t)} = \sum_{q=1}^{t-1} d_i^{(q)} v_j^{(q-1)}$, $\bar{\sigma}_t^2 := \left[\sum_{j=1}^{F} \frac{1}{\left(\sum_i \alpha_{ij}^{(t)}+1\right)^2}\right]$, *and* ψ_t *is any non-negative number.*

Proof. Given in Appendix B.5. □

Remark: Note that the above result is an algorithm dependent bound as it depends on the recommendation strategy, which is determined by the algorithm. In order to provide more insights into the result, we will make certain assumption about the demands. In particular, if the demands $d_i^{(q)} > 0$ *almost surely* for all i and q, then, $\alpha_{ij}^{(t)}$ will be 0 when $v_j^{(q-1)} = 0$ for all $q \le t-1$ or $\mathcal{O}(t)$ in case of $\sum_{q=1}^{t} v_j^{(q-1)} = \mathcal{O}(\sqrt{t})$. Note that this depends on the algorithm output, which we presume that at least files of higher probability transition values with recommendation will be sampled multiple times in a time frame. This leads to $\bar{\sigma}_t^2 = \mathcal{O}(\frac{1}{t^2})$. Thus, assuming $\Psi_t = \mathcal{O}(\sqrt{t})$, the summation in the first term of the regret is $\mathcal{O}(1)$. Overall, this results in $\mathcal{O}(\sqrt{T})$ regret. Recall that an approximate regret of $\mathcal{O}(T^{2/3}\sqrt{\log T})$ is shown for the genie aided case while the `Bayesian estimation` method achieves a regret of the order $\sqrt{T}$. The genie aided regret is worse by an order of $T^{1/6}$, which is partly due to the fact that the genie estimation of PTM is less accurate than the `point estimation` method. In the next section, we extend our results to two SBS scenario.

4.4 Proposed Caching and Recommendation Strategies With Multiple SBSs

In this section, we present caching and recommendation algorithms when there are multiple SBSs. In particular, we provide insights on how to use the neighboring SBSs estimates to further improve the overall caching and recommendation performance of the network. First, we present the results for two SBS scenario, and similar analysis will be used to extend the results to multiple SBSs.

4.4.1 Two Small Base Station Scenario

In this subsection, we consider a two SBSs scenario, as shown in Fig. 4.3. As described in Section 4.2, $\mathbf{P}_1$ and $\mathbf{P}_2$ represents PTM for SBS-1 and SBS-2, respectively.

The central MBS sends the global update of the recommendation and caching decisions to each SBS. Assume that the request across SBSs are independent. Let each SBS use one of the estimation methods in Algorithm 4. Let $\hat{\mathbf{P}}_1^{(t)}$ and $\hat{\mathbf{P}}_2^{(t)}$ be the corresponding estimates (either `point` or `Bayesian estimate`) of $\mathbf{P}_1$ and $\mathbf{P}_2$, respectively. The two SBSs convey their respective PTM to the central MBS. The central MBS computes an estimate $\hat{\mathbf{Q}}_k^{(t)}$, $k = 1, 2$ for SBS 1 and SBS 2 as a linear combination of the two estimates as given below

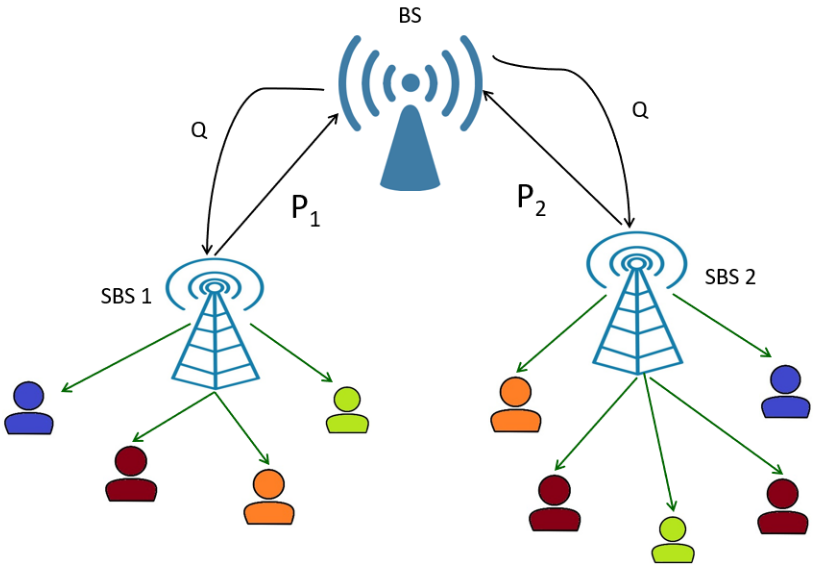


Figure 4.3: Heterogeneous network with distributed caching consisting of two SBS.

$$\hat{\mathbf{Q}}_k^{(t)} = \lambda_k \hat{\mathbf{P}}_1^{(t)} + (1 - \lambda_k) \hat{\mathbf{P}}_2^{(t)}, \tag{4.16}$$

where $\lambda_k \in [0, 1]$, $k = 1, 2$ strikes a balance between the two estimates. The above estimate is used to compute the respective caching and recommendation strategies for the two SBSs and will be communicated to the respective SBSs. The above results in a better estimate, for example, when $\mathbf{P}_1 = \mathbf{P}_2$ or when the two matrices are close to each other. The corresponding algorithm is shown below. First, we prove the following guarantee for the `Point estimation` method.

Theorem 4.4.1. *For Algorithm 5 with* `point estimation`*, for any SBS k and for any $\epsilon > 0$, with a probability of at least $1 - \delta$, $\delta > 0$, the regret $\texttt{Reg}_{k,T} < \epsilon$, i.e.,*

$\Pr\left\{(\boldsymbol{u}_{k,t}^*)^T \boldsymbol{P}_k \boldsymbol{v}_{k,t}^* \geq \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \boldsymbol{u}^T \hat{\boldsymbol{Q}}_k^{(t)} \boldsymbol{v} - \epsilon_k\right\} > 1-\delta$ *provided*

$$t \geq \max\left\{\tau\left(\frac{\epsilon_k}{\lambda_k}, \frac{\delta}{2}\right), \tau\left(\frac{\epsilon_k}{(1-\lambda_k)}, \frac{\delta}{2}\right)\right\}, \tag{4.17}$$

where

$$\tau(\epsilon,\delta) := \frac{1}{q\left(1-\exp\{-\frac{N\epsilon^2}{8\kappa^2 F^2 c^2 r^2}\}\right)} \log\frac{2|\mathcal{N}_\epsilon|F^2}{\delta}. \tag{4.18}$$

Further, $\epsilon_k := \epsilon/2 - (1-\lambda_k)\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T(\boldsymbol{P}_2 - \boldsymbol{P}_1)\boldsymbol{v}\right|$

Proof. Given in Appendix B.6. □

Algorithm 5 Caching and recommendation algorithm (two SBS case)

procedure POINT ESTIMATION/BAYESIAN ESTIMATION
    $\hat{\boldsymbol{u}}_{b,0}^* \overset{i.i.d.}{\sim} \{0,1\}$ from $p = c/F$, and $\hat{\boldsymbol{v}}_{b,0}^* \overset{i.i.d.}{\sim} \{0,1\}$ from $q = r/F$.
    Recommend & cache according to $\hat{\boldsymbol{v}}_{b,0}^*$ & $\hat{\boldsymbol{u}}_{b,0}^*$.
    for $t = 0,1,\ldots,T$ **do**
        Observe demands $d_{ijk}^{(t)}$ in slot t, $k = 1,2$.
        if `point estimation` **then**
            Compute $\hat{\boldsymbol{P}}_k^{(t)}$ from (4.9) for `point estimation`
        else
            Compute $\hat{\boldsymbol{P}}_k^{(t)}$ from (4.11) for `Bayesian estimation`
        end if
        Choose λ_i, and find $\hat{\boldsymbol{Q}}_i^{(t)}$ from (4.16), and solve $(\hat{\boldsymbol{u}}_{k,t}^*, \hat{\boldsymbol{v}}_{k,t}^*) = \arg\max_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \boldsymbol{u}^T \hat{\boldsymbol{Q}}_k^{(t)} \boldsymbol{v}$
        Use $(\hat{\boldsymbol{v}}_{k,t}^*, \hat{\boldsymbol{u}}_{k,t}^*)$ to recommend and cache.
    end for
end procedure

As in the single SBS case, to benchmark the performance of `Bayesian estimation` method, we consider a genie aided scenario, and in Appendix B.4, we show that it achieves an approximate regret of

$$\mathtt{Reg}_{k,T} \lessapprox \max\left\{\Theta\lambda_k, \Theta(1-\lambda_k)\right\} T^{2/3} + 2T(1-\lambda_k)\mathcal{V}_{12},$$

where $\mathcal{V}_{12} := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T(\mathbf{P}_2 - \mathbf{P}_1)\boldsymbol{v}\right|$ and $\Theta = \sqrt[3]{\frac{8\kappa^2F^2c^2r^2(\log 4F^2T^2+F)}{qN}}$. The above clearly shows the trade-off between the two terms. The first term scales as $T^{2/3}$ while the second term scales with T linearly. This can be balanced by using $\lambda_k = 1 - \frac{1}{\sqrt{T}}$, which results in $\mathcal{O}(\sqrt{T})$ scaling of regret. Note that the choice

$\lambda_k = 1 - \frac{1}{\sqrt{T}}$ reveals that as time progresses, i.e., as the BS k collects more samples, the weights allocated to the neighboring BS should go down to zero, as expected. Furthermore, by appropriately choosing λ_k as above, the regret obtained is of the order $T^{2/3}$. Next, we present the guarantees for Algorithm 5.

Theorem 4.4.2. *For Algorithm 5 with* `Bayesian estimation`*, for*

$$\epsilon > 2 \max_{k=1,2}(1-\lambda_k)T \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T(\boldsymbol{P}_1 - \boldsymbol{P}_2)\boldsymbol{v}\right|, \tag{4.19}$$

with probability of at least $1-\delta$, $\delta > 0$, for any BS $k \in \{1,2\}$, the regret can be bounded as

$$\texttt{Reg}_{k,T} \leq \max\left\{R_k\left(\frac{2\epsilon_k}{\lambda_k}, \frac{\delta}{2}\right), R_k\left(\frac{2\epsilon_k}{(1-\lambda_k)}, \frac{\delta}{2}\right)\right\}. \tag{4.20}$$

In the above,

$$R_k(\epsilon,\delta) := 2rc \max_{ijk} p_{ijk}|\mathcal{N}_\epsilon| \sum_t \exp\left\{\frac{-8\psi_t^2}{cr\left|\mathcal{N}_\epsilon\right|^2 \bar{\sigma}_k^2(t)}\right\} + 2\sum_t \psi_t + \sqrt{128r^2c^2T\log(1/\delta)},$$

$\alpha_{ijk}^{(t)} = \sum_{q=1}^{t-1} d_{ik}^{(q)} v_{jk}^{(q-1)}$, $\epsilon_k := \epsilon/2 - (1-\lambda_k)T\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}}\left|\boldsymbol{u}^T(\boldsymbol{P}_1-\boldsymbol{P}_2)\boldsymbol{v}\right|$, *and* $\bar{\sigma}_k^2(t) := \left[\sum_{j=1}^F \frac{1}{\left(\sum_i \alpha_{ijk}^{(t)}+1\right)^2}\right]$.

Proof. Given in Appendix B.7. □

Remark: The result shows the tradeoff exhibited by λ_k. In particular, larger λ_k makes the first regret inside the max term in (4.21) larger, and smaller λ_k ensures that the second term inside the max above dominates. It is essential to balance the two depending on the number of samples received. We further elaborate this in the simulation results. Further, the above result is an algorithm dependent bound as the bound depends on the recommendation strategy, which is determined by the algorithm. Following the single SBS analysis, we make similar assumptions in two SBS scenario. In particular, if the demands $d_{ik}^{(q)} > 0$ *almost surely* for all i and q, then, $\alpha_{ijk}^{(t)}$ will be 0 when $v_{jk}^{(q-1)} = 0$ for all $q \leq t-1$ or $\mathcal{O}(t)$ in case of $\sum_{q=1}^t v_{jk}^{(q-1)} = \mathcal{O}(\sqrt{t})$. Note that this depends on the algorithm output, which we presume that at least files of higher probability transition values with recommendation will be sampled multiple times in a time frame. This leads to $\bar{\sigma}_k^2(t) = \mathcal{O}(\frac{1}{t^2})$. Thus, assuming $\Psi_t = \mathcal{O}(\sqrt{t})$, the summation in the first term of the regret is $\mathcal{O}(1)$. Overall, this results in $\mathcal{O}(\sqrt{T})$

regret. Furthermore, the regret obtained by the genie aided method is of the order $T^{2/3}$, which is higher than the one achieved by the `Bayesian estimation` method. This along with the experimental results establishes the superiority of the proposed `Bayesian estimation` method.

4.4.2 Multiple Small Base Station Scenario

In this section, we consider a heterogeneous network with M SBSs connected to a central MBS. The requests at each SBS are assumed to be i.i.d. with PTM $\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_M$ as described in Section 4.2. Similar to the two SBS model, each SBS computes an estimate of the PTM as follows

$$\hat{\mathbf{Q}}_k^{(t)} = \lambda_1^{(k)} \hat{\mathbf{P}}_1^{(t)} + \lambda_2^{(k)} \hat{\mathbf{P}}_2^{(t)} + \ldots + \lambda_M^{(k)} \hat{\mathbf{P}}_M^{(t)}, \tag{4.21}$$

where $\lambda_1^{(k)}, \lambda_2^{(k)}, \ldots, \lambda_M^{(k)},\ k = 1, 2, \ldots, M$ are coefficients to be determined later. Further, it satisfies $\sum_{j=1}^{M} \lambda_j^{(k)}$. The following theorem is a generalization of two BS model which provides a guarantee on the minimum time required to achieve a certail level of accuracy with high probability.

Theorem 4.4.3. *Using* (4.21) *for any*

$$\epsilon > M^2 \max_k \{(1 - \lambda_1^{(k)})\mathcal{D}_1 - \lambda_2^{(k)}\mathcal{D}_2 - \ldots - \lambda_M^{(k)}\mathcal{D}_M\},$$

for `point estimation`*, with a probability of at least* $1-\delta$*,* $\delta > 0$*, for any BS* k*, the regret (*$\texttt{Reg}_{k,T}$*) is less than* ϵ $\left\{ i.e.\ (\boldsymbol{u}_{o,t}^*)^T \boldsymbol{P}_1 \boldsymbol{v}_{o,t}^* \geq \sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}^T \boldsymbol{P}_1 \boldsymbol{v} - \epsilon \right\}$ *provided*

$$t \geq \max\left\{ \tau\left(\frac{\epsilon_1}{\lambda_1^{(k)}}, \frac{\delta}{M}\right), \tau\left(\frac{\epsilon_2}{\lambda_2^{(k)}}, \frac{\delta}{M}\right), \ldots, \tau\left(\frac{\epsilon_M}{\lambda_M^{(k)}}, \frac{\delta}{M}\right) \right\}, \tag{4.22}$$

$$\tau(\epsilon, \delta) := \frac{1}{q\left(1 - \exp\{-\frac{N\epsilon^2}{8\kappa^2 F^2 c^2 r^2}\}\right)} \log \frac{2\mathcal{N}_\epsilon F^2}{\delta}, \tag{4.23}$$

where, $\epsilon_k := \epsilon/M^2 - (1 - \lambda_1^{(k)})\mathcal{D}_1 + \lambda_2^{(k)}\mathcal{D}_2 +, \ldots, + \lambda_M^{(k)}\mathcal{D}_M$*, and* $\mathcal{D}_k := \sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \left|\boldsymbol{u}^T P_k \boldsymbol{v}\right| \ \forall\ k = 1, 2, \ldots, M.$

Proof. Given in Appendix B.8. $\square$

The regret for the genie aided case after appropriate choice for λ_k turns out to be of the order of $\sqrt{T}$. The genie aided regret analysis is relegated to Appendix B.4.1. Next we present the regret bound for the `Bayesian estimation` method.

Theorem 4.4.4. *Using* (4.21) *for any*

$$\epsilon > M^2 \max_k \{(1-\lambda_1^{(k)})\mathcal{I}_1 - \lambda_2^{(k)}\mathcal{I}_2 -, \dots, -\lambda_M^{(k)}\mathcal{I}_M\},$$

with probability of at least $1-\delta$, $\delta > 0$, *for any BS* k, $k \in \{1, 2, \dots, M\}$ $\textbf{\textit{Reg}}_{k,T}$, *the regret of the* ***`Bayesian estimation`*** *method satisfies the following bound*

$$\textbf{\textit{Reg}}_{k,T} \leq \max\left\{R_k\left(\frac{\epsilon_1}{\lambda_1^{(k)}}, \frac{\delta}{M}\right), R_k\left(\frac{\epsilon_2}{\lambda_2^{(k)}}, \frac{\delta}{M}\right), \dots, R_k\left(\frac{\epsilon_M}{\lambda_M^{(k)}}, \frac{\delta}{M}\right)\right\}. \quad (4.24)$$

In the above

$$R_k(\epsilon, \delta) = 2rc \max_{ijk} p_{ijk} |\mathcal{N}_\epsilon| \sum_t \exp\left\{\frac{-8\psi_t^2}{cr\,|\mathcal{N}_\epsilon|^2\,\bar{\sigma}_k^2(t)}\right\} + 2\sum_t \psi_t + \sqrt{128 r^2 c^2 T \log(1/\delta)},$$

$\alpha_{ijk}^{(t)} = \sum_{q=1}^{t-1} d_{ik}^{(q)} v_{jk}^{(q-1)}$, $\epsilon_k := \epsilon/M^2 - (1-\lambda_1^{(k)})\mathcal{I}_1 + \lambda_2^{(k)}\mathcal{I}_2 +, \dots, +\lambda_M^{(k)}\mathcal{I}_M$,
$\mathcal{I}_k := \sum_t \sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \left|\boldsymbol{u}^T P_k \boldsymbol{v}\right| \; \forall\; k = 1, 2, \dots, M$ *and* $\bar{\sigma}_k^2(t) := \left[\sum_{j=1}^F \frac{1}{\left(\sum_i \alpha_{ijk}^{(t)} + 1\right)^2}\right]$.

Proof. Given in Appendix B.9. □

Remark: Note that the above result is an algorithm dependent bound as it depends on the recommendation strategy, which is determined by the algorithm. Following the single SBS analysis, we make similar assumptions in multiple SBS scenario. In particular, if the demands $d_{ik}^{(q)} > 0$ *almost surely* for all i and q, then, $\alpha_{ijk}^{(t)}$ will be 0 when $v_{jk}^{(q-1)} = 0$ for all $q \leq t-1$ or $\mathcal{O}(t)$ in case of $\sum_{q=1}^t v_{jk}^{(q-1)} = \mathcal{O}(\sqrt{t})$. Note that this depends on the algorithm output, which we presume that at least files of higher probability transition values with recommendation will be sampled multiple times in a time frame. This leads to $\bar{\sigma}_k^2(t) = \mathcal{O}(\frac{1}{t^2})$. Thus, assuming $\Psi_t = \mathcal{O}(\sqrt{t})$, the summation in the first term of the regret is $\mathcal{O}(1)$. Overall, this results in $\mathcal{O}(\sqrt{T})$ regret. As in the case of two SBS, the regret obtained is better than the genie aided scenario. In the next section, we present experimental results that corroborates some of our theoretical observations.

4.5 Simulation Results

In this section, simulation results are presented to highlight performance of the proposed caching and recommendation model. The simulation setup consists of one SBS model, two-SBS model and a heterogeneous model with multiple users. We assume a time-slotted system in the simulation setup. For the heterogeneous model, the simulation consists of two scenarios as follows:

- Fixed Link Scenario: In this, the links between SBS and users are uniformly and independently distributed in $\{0, 1\}$ with probability 1/2.

- $\mathtt{SINR}$ Based Scenario: In this, the SBS and users are assumed to be distributed uniformly in a geographical area of radius 500m. It is assumed that a SBS and user can communicate only if the corresponding $\mathtt{SINR}$ is greater than a threshold. This $\mathtt{SINR}$ takes into account the fading channel, the path loss, power used, and the distance between the user and the SBS. The minimum rate at which a file can be transferred from the SBS to a user is given by the threshold, and hence the reciprocal of the rate indicates the delay. In the simulation, we have used $\tau := \frac{1}{\log(1+\mathtt{SINR})}$ as a measure of the delay between a user and a SBS. However, when the requested file is absent, a backhaul fetching delay of $\alpha \times \tau$ is counted in addition to the down link delay of τ, i.e., the overall delay when the file is absent is $(\alpha + 1)\tau$, with $\alpha = 10$. Also, if the threshold is R, then at least R bits can be sent in a time duration of at most $1/\log(1+\mathtt{SINR})$ seconds, and hence the throughput is roughly $R\log(1+\mathtt{SINR})$ bits/second.

Fig. 4.4 shows the plot for a heterogeneous system the metric used for comparison is throughput. In Fig. 4.4, the number of SBSs, users, the total number of files and threshold value for $\mathtt{SINR}$ are 5, 30, 100,and 12dB, respectively. The throughput for the proposed algorithm with recommendation is 225 bits/s for a cache size of 24, while LRFU, LRU and LFU algorithm has a throughput of 100 bits/s, 85 bits/s and 70 bits/s respectively for the same cache size. Thus from Fig. 4.4 we can see that the proposed algorithm has higher throughput as compared to the existing algorithms.

Fig. 4.5 corresponds to the $\mathtt{SINR}$ scenario for a two SBS model. Fig. 4.5 shows the average delay versus cache size plot for (a) cache placement algorithm with recommendation (b) cache placement algorithm without recommendation, (c) LRFU algorithm, (d) LRU algorithm and (e) LFU algorithm. In Fig. 4.5, SBSs, the number of users, the total number of files and threshold value for SINR are 2, 25, 100 and 12dB, respectively. From Fig. 4.5 we can observe that the delay of both the proposed algorithms is less as compared to the other benchmark algorithms, since pre-fetching files according to the estimated methods results in lower fetching costs from the backhaul and hence less delay.

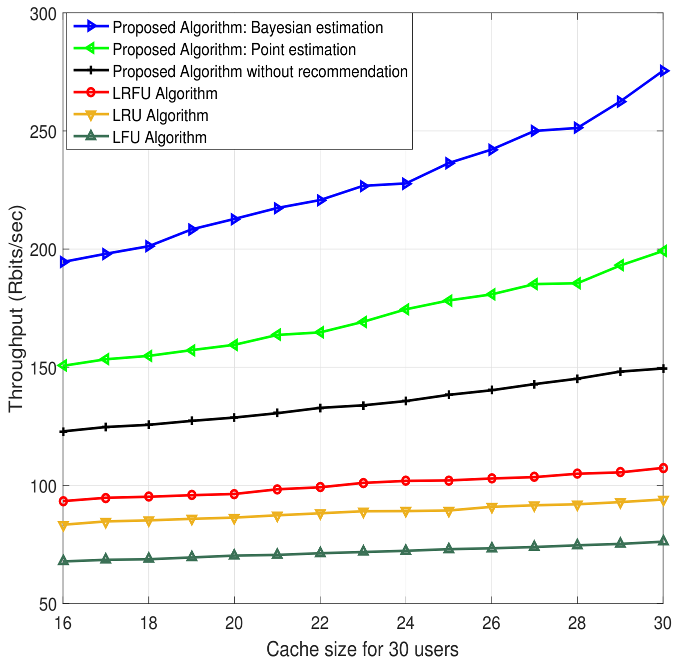


Figure 4.4: Average throughput v/s cache size for 15 SBS model.

Fig. 4.6 shows the plot for two BS model. The value of λ_1 is varied between 0.1 and 1. From the Fig. 4.6, we can observe that as the value of λ_1 approaches 0.5, the average cache hit increases, this is because for $\lambda_1 = 0.5$ and $P_1 = P_2$, the Q popularity profile matrix of BS will have maximum similarity to the individual SBS popularity profile matrix and hence the cache hit will be maximum for $\lambda_1 = 0.5$ and it will gradually decrease as we further increase the value of λ_1. Fig. 4.7 shows the plot for average cache hit versus λ for 2 SBS when $\mathbf{P}_1 \neq \mathbf{P}_2$. From the Fig. 4.7, we can observe that for larger T, the optimal lambda value is close to 1. Also, for smaller value of T, depending on the value of Θ, the optimal value of λ is less than 1. Thus, the simulation results prove that the recommendation helps in increasing the

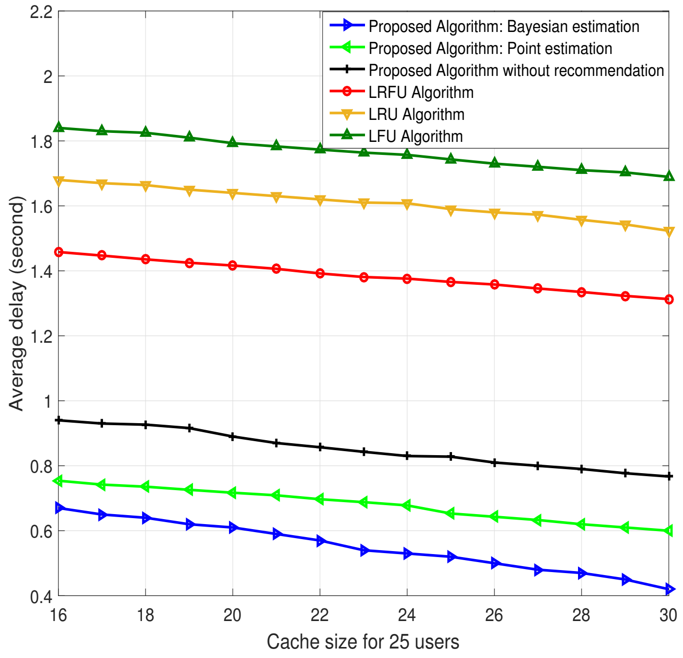


Figure 4.5: Average delay v/s cache size for two-SBS model.

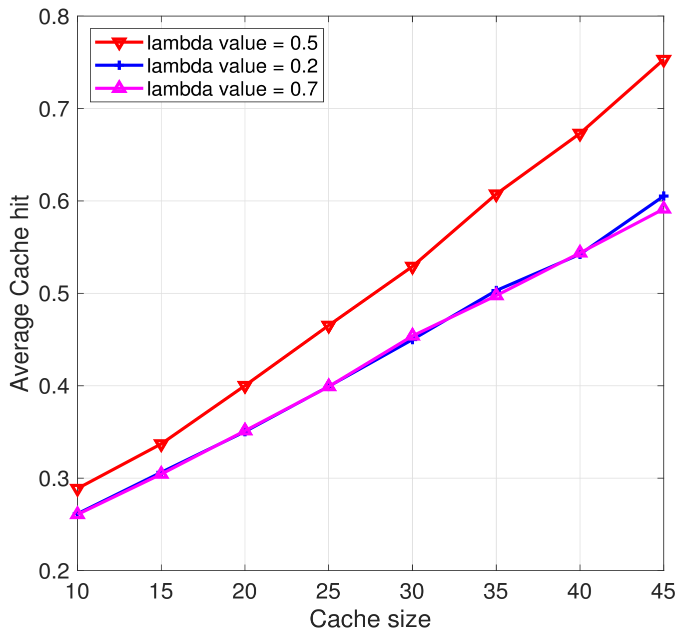


Figure 4.6: Cache hit v/s cache size for 2 SBS and $\mathbf{P}_1 = \mathbf{P}_2$.

average cache hit when compared to the algorithm without recommendation and it also performs better than the existing popular LRFU, LRU and LFU algorithms.

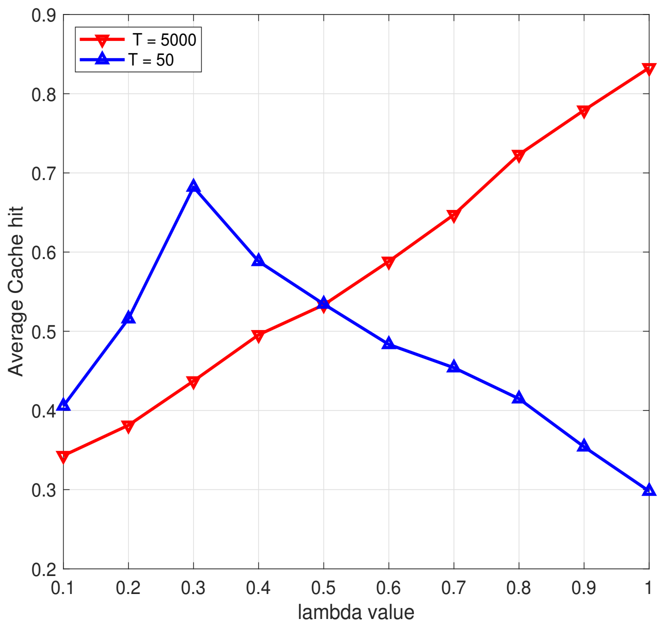


Figure 4.7: Cache hit v/s λ for 2 SBS and $\mathbf{P}_1 \neq \mathbf{P}_2$.

4.6 Summary

The deployment of MEC in the current wireless heterogeneous networks is an important application for the smooth transition to the distributed cloud based platform. In this chapter, a model that captures the caching decisions along with recommendation has been introduced. Implications of recommendation on user requests has been studied and it has been observed that the recommendation does influence the demands from the users. Bayesian and point estimation methods are used to determine the user request pattern. An algorithm is then proposed to jointly optimize caching and recommendation. A multi-tier heterogeneous model consisting of MBS and SBSs is also presented and an upper bound on the estimation accuracy of popularity profile is provided. Finally, simulation results and theoretical proofs support the superior performance of the proposed method over the existing algorithms. In this chapter, we addressed the joint optimization problem of caching and recommendation, however, we have not considered the situations wherein the outcomes are partly under the control of a decision maker and sequential. Hence, in the next chapter, we look into discrete time stochastic process (MDP) and jointly optimize caching and recommendation.

Chapter 5

Recommendation based Caching using Whittle Index

5.1 Overview

In this chapter, the joint caching and recommendation is modeled as an MDP process and. The joint optimization problem is a partially observable MDP with the classic exploitation versus exploration tradeoff that is difficult to quantify. We, therefore, study the problem in the framework of restless multiarmed bandit processes, and perform a Whittle's indexability analysis. The multi-armed bandit problem (MABP) is a decision making problem that sequentially activates one out of I parallel Markov processes and leaves the remaining $I-1$ passive, the passive processes will be frozen (with no state transition) until they become active. Gittins [142] proved the optimality of a simple index policy for the MABP, subsequently referred to as the Gittins index policy, which always activates the process with the highest state-dependent index. Whittle extended the conventional MABP to a restless case and proposed an index policy, referred to as the Whittle index policy, and conjectured it to be asymptotically optimal as the number of parallel processes tends to infinity. The Whittle index policy, always prioritizes the processes with the highest statedependent indices, referred to as the Whittle indices, which are calculated by solving sub-problems with remarkably reduced state spaces.

The edge systems should be capable of determining the environment and then take optimal or near optimal caching actions, without any explicit programming.

The task of Q-learning is usually described as MDP, however, state space, transition probabilities and reward functions are not required. Inspired by the success of the MDP models, it can be deployed at user edge networks to understand the user's behaviour and network patterns.

In this chapter, caching and recommendation in a wireless network is jointly looked into. The demands are modelled as a two-state markov chains. At the end of each slot, the caching decision is associated with two objectives: (a) the reward achieved when a cached file is served to a user, (b) exploration of the recommendation of a file for more informed decisions and associated rewards in the future. These two objectives results in the classic exploration v/s exploitation tradeoff seen in sequential decision making problems. Specifically the contributions are as follows:

- In this chapter, the joint caching and recommendation is modeled as an MDP and its Whittle's indexability is established.
- Based on the Whittle's indexability condition, the whittle index policy for the caching problem is provided. The Whittle's index policy is known to have near-optimal performance and have shown low-complexity [80].

5.2 System Model

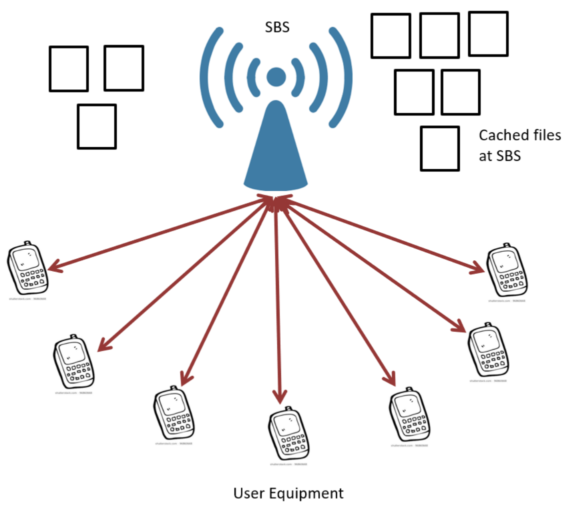


Figure 5.1: Caching based wireless heterogeneous network

Fig. 5.1 depicts the wireless caching network consisting of one BS, $\mathcal{U}$ mobile users and $\mathcal{F}$ files. The assumption is made that the SBS is capable of storing C files of equal sizes. Fig. 5.1 illustrates how the SBS is connected with the user, and this connection is scheduled by the scheduler, irrespective of the caching algorithm used. It is assumed that the time is slotted. The demand variable $d_{f,t}$ denotes the total number of files requested from all the connected users to the BS in time slot t. The demand for each file is modelled as a two-state Markov chain i.e. the state remains static within each time slot and evolves across time slots according to Markov chain statistics. The state space A_i for each file i is given as $S_i = \{l_i, h_i\}$. State h_i corresponds to when the complete file is cached. When the demand in state l_i, there exists a fraction δ_i, $0 \leq \delta_i < 1$, such that demands below the fraction δ_i do not get cached, and is cached above the fraction δ_i. Thus for a file i, the two-state Markov process is given by a 2 x 2 PTM.

$$P_i = \begin{bmatrix} p_i & 1-p_i \\ r_i & 1-r_i \end{bmatrix},$$

where

$$p_i := \text{prob}(A_i[t] = h_i | A_i[t-1] = h_i),$$
$$r_i := \text{prob}(A_i[t] = h_i | A_i[t-1] = l_i),$$

where $A_i[t]$ denote the state of the cached file in the BS in time slot t.

5.2.1 Caching Model

At the beginning of the time slot, the BS does not have the exact knowledge of the state for each files. Thus it maintains a belief value π_i for each file i which is the probability that A_i is in state h_i. Specifically, in each slot the BS jointly makes the following decisions: (a) the BS decides on the optimal files to be cached and calculates the expected reward, (b) based on the reward achieved the BS picks a file for recommendation, while simultaneously taking into account the effect of this decision on the long-term reward. At the end of the time slot, the BS receives the information on the Markov state and this is used as an feedback to update its belief.

Let u be the cached file and v be the recommended file. The caching problem can be modelled as a partially observable MDP. Hence, $\phi : \{u, v\}$ denote the caching and recommendation pair and Φ denote the set of such pairs. In each slot, the aim is to find the optimal pair which maximizes the reward. Let π denote the belief value for each file i. Then the optimal pair $\phi^*_{i,\pi_i} = \{u^*_{i,\pi_i}, v^*_{i,\pi_i}\}$, for file i, as a function of the belief π_i is given as

$$\phi^*_{i,\pi_i} = \underset{\phi\in\Phi}{\operatorname{argmax}}\, \mathbb{E}_{A_i}[\gamma_i(A_i, \phi)], \tag{5.1}$$

where $\gamma(A_i, \phi)$ is the average cost of caching for file i when the underlying Markov process is in state A_i and the caching and recommendation pair ϕ is deployed. The distribution of the state A_i is characterized by the belief value π_i as follows:

$$A_i = \begin{cases} h_i, & \text{w. p. } \pi_i \ , \\ l_i, & \text{w. p. } 1 - \pi_i. \end{cases} \tag{5.2}$$

The expected immediate reward when file i is cached is given by

$$R_i(\pi_i) = \mathbb{E}_{A_i}[\gamma_i(A_i, \phi^*_{i,\pi_i})] \tag{5.3}$$

Here ϕ^*_{i,π_i} denotes the optimal caching strategy that maximizes the expected immediate reward.

Let $\bar{\boldsymbol{\pi}}[t] = (\pi_1[t], \ldots, \pi_N[t])$ denote the belief vector at the beginning of the time slot. A stationary caching policy, $\boldsymbol{\Psi}$, is a stationary mapping $\boldsymbol{\Psi} : \hat{\boldsymbol{\pi}} \to I$ between the belief vector and the index of the cached file. For this stationary policy $\boldsymbol{\Psi}$, the infinite horizon discounted reward under initial belief $\bar{\boldsymbol{\pi}}$ is given by

$$V(\boldsymbol{\Psi}, \bar{\boldsymbol{\pi}}) = \sum_{t=0}^{\infty} \beta^t \mathbb{E}_{\bar{\boldsymbol{\pi}}} R_{I[t]=\boldsymbol{\Psi}(\bar{\boldsymbol{\pi}}[t])}(\boldsymbol{\pi}_{I[t]}[t]) \tag{5.4}$$

where $\bar{\boldsymbol{\pi}}$ is the belief vector in time slot t, $\pi_i[t]$ denotes the belief value of file i in slot t, $\bar{\boldsymbol{\pi}}[0] = \bar{\boldsymbol{\pi}}$, $I[t]$ denotes the index of the file cached in time slot t. The discount factor is given by $\beta \in [0, 1]$. The optimal expected discount reward is given by the Bellman equation as follows:

$$V(\bar{\boldsymbol{\pi}}) = \max_{I}\{R_I(\pi_I) + \beta \mathbb{E}_{\bar{\boldsymbol{\pi}}[t+1]}[V(\bar{\boldsymbol{\pi}}[t+1])]\} \tag{5.5}$$

The belief evolution $\boldsymbol{\pi}[t] \rightarrow \boldsymbol{\pi}[t+1]$ proceeds as follows:

$$\pi_i[t+1] = \begin{cases} p_i & \text{if I[t] = i and } A_i[t] = h_i, \\ r_i & \text{if I[t] = i and } A_i[t] = l_i, \\ Q_i(\pi_i[t]) & \text{if I[t]} \neq \text{i} \end{cases} \tag{5.6}$$

where $Q_i(\pi_i) = \pi_i p_i + (1-\pi_i) r_i$ is the belief evolution when the file i is not cached in time slot t.

The expected average cost of caching for each arm i is given by

$$C_i^{avg}(\pi) \triangleq \lim_{T\rightarrow\infty} \sup \frac{1}{T}\mathbb{E}\Bigg[\sum_{t=1}^{T} f_i(A_i^{\pi}(t))\Bigg] \tag{5.7}$$

where $A_i^{\pi}(t)$ is the fraction of the ith file cached under caching policy π. The aim is to find the caching policy π such that it minimizes the average costs for caching. Let Π denote the set of all the caching policies, then the optimization problem is as follows :

$$C^* = min_{\pi\in\Pi} \sum_{i=1}^{N} C_i^{avg}(\pi) \tag{5.8}$$

where C^* is the minimum average cost and π^* is the optimal caching policy.

5.3 Optimal Caching Cost

In this section, bounds on the expected reward has been derived.

Lemma 1:The expected immediate reward $R(\pi)$ has the following properties::

(a) *$R(\pi)$ is convex and increasing in π for $\pi \in [0,1]$*

(b) *$R(\pi)$ is bounded as follows:*

$$\max\{\delta, \pi\} \leq R(\pi) \leq (1-\delta)\pi + \delta \tag{5.9}$$

Proof: Let Φ^* be the set of optimal caching and recommendation strategy pairs for all $\pi \in [0,1]$, i.e. $\Phi^* = \{\phi^*, \pi \in [0,1]\}$. The expected reward in (5.3) can be rewritten as

$$\begin{aligned} R(\pi) &= \max_{\phi\in\Phi^*} \mathbb{E}_A[\gamma(A,\phi)] \\ &= \max_{\phi\in\Phi^*}[\pi\gamma(h,\phi) + (1-\pi)\gamma(l,\phi)], \end{aligned}$$

where $\gamma(S,\phi)$ denotes the average cost of caching when the demand state is $S \in \{l,h\}$. If we look at the average cost, when ϕ is fixed, we can see that $\pi\gamma(h,\phi) + (1-\pi)\gamma(l,\phi)$ is linear in π. Hence, $R(\pi)$ can be taken as a point-wise maximum over a family of linear functions, and thus is convex. $R(\pi)$ is convex thus proving the convexity statement in (a).
Now to determine the bounds to $R(\pi)$, from equation (5.3) we have,

$$R(\pi) = \max_{\phi\in\Phi} \mathbb{E}_A[\gamma(A,\phi)] \geq \max_{u} \mathbb{E}_A[\gamma(A,\{u,*\})]$$

where $\gamma(A,\{u,*\})$ is the average cost when only caching is considered without taking into account the recommendation. Note that without the recommendation, caching is only a function of the belief value π. Thus the average cost under the caching strategy can be expressed in terms of the indicator functions as follows:

$$\begin{aligned} & \max_{u} \mathbb{E}_A[\gamma(A,\{u,*\})] \\ = \;& \max_{u}[P(A=l)u \cdot \mathbb{I}(u \leq \delta) + P(A=h)u \cdot \mathbb{I}(u \leq 1)] \\ = \;& \max_{u} u[P(A=l) \cdot \mathbb{I}(u \leq \delta) + P(A=h) \cdot \mathbb{I}(u \leq 1)] \\ = \;& \max\{\delta, \pi\}. \end{aligned}$$

Thus the lower bound in (b) is proved. The upper bound corresponds to when the full demand state information is available.

5.4 RMAB and Whittle's Indexability

In this section we show that joint caching and recommendation problem can be formulated as restless multi-armed bandit (RMAB). For our problem, the RMAB can be defined as a family of sequential dynamic resource allocation problem in the presence of several independent evolving demands. For each time slot, the state of each arm stochastically evolve in time based on the current state and the action taken. At the end of each time slot, reward dependent on the states and the action taken is obtained. Thus, RMABs can be denoted as a tradeoff between decisions giving high immediate rewards versus those that sacrifice immediate rewards for better future rewards. Whittle index policies, if existent, are known to have near optimal performance in an average constraint RMAB problem. However, Whittle's index policy exists only if the RMAB problem satisfies the following condition known as Whittle's indexability. Let the activation charge to pull an arm be denoted by ω, which is strictly positive. Let $D(\omega)$ be the set of states for the RMAB problem such that it is optimal to to stay passive.

Indexability property: The RMAB problem is Whittle indexable if and only if, as ω increases from 0 to ∞, the set $D(\omega)$ monotonically decreases from entire set to empty set $\{\phi\}$.

For the joint caching and recommendation problem, in each time slot, based on the current belief value π, the BS caches the file f (action $a = 1$) or stays idle ($a = 0$). In an idle state, it obtains a subsidy of ω. The belief value is updated based on the reward obtained and feedback received. Henceforth, the optimal scheduling policy maximizes the infinite horizon discounted reward, which is given by the following Bellman equation:

$$V_\omega(\pi) = \max\{[R(\pi) + \beta(\pi V_\omega(p) + (1-\pi)V_\omega(r))], [\omega + \beta V_\omega(Q(\pi))]\} \tag{5.10}$$

The first term inside the max operator refers to the infinite horizon reward when a file is cached and the second term corresponds to the idle decision in the current slot.

5.4.1 Thresholdability of the ω-subsidy policy

The convexity property of the infinite horizon discounted reward is proved in the following proposition:

Proposition 5.4.1. *: The infinite horizon discounted reward, $V_\omega(\pi)$ is convex in $\pi \in [0,1]$.*

Proof. Given in Appendix C.1. □

Proposition 5.4.2. *: The optimal ω subsidy policy is thresholdable in the belief space π. There exists a threshold $\pi^*(\omega)$ such that the optimal action a is 1 if the current belief $\pi > \pi^*$ and the optimal action a is 0, otherwise. The value of the threshold $\pi^*(\omega)$ depends on the subsidy ω as given below:*
(i) if $\omega \geq 1$, $\pi^*(\omega) = 1$,
(ii) if $\omega \leq \delta, \pi^*(\omega) = \kappa$, for some $\kappa < 0$,
(iii) if $\delta \leq \omega \leq 1, \pi^*(\omega) \in (0, 1)$.

Proof. Given in Appendix C.2. □

5.4.2 Whittle Indexability

The caching problem is Whittle indexable if the threshold $\pi^*(\omega)$ monotonically increases with the subsidy ω. To prove the indexability, the closed form expression of the discounted reward $V_\omega(p)$ and $V_\omega(r)$ is important and is proved in the following lemma.

Lemma 2: The closed form expression of $V_\omega(p)$ and $V_\omega(r)$ is given as follows.
Case 1: $p > r$ (positive correlation)

$$V_\omega(p) = \begin{cases} \sum_{k=0}^{\infty} \beta^k R\left(\frac{r+(p-r)^{k+1}(1-p)}{1+r-p}\right) & \text{if } \pi^*(\omega) < \pi^0, \\ \frac{\beta(1-p)\omega+(1-\beta)R(p)}{(1-\beta)(1-\beta p)} & \text{if } \pi^0 \leq \pi^*(\omega) < p \ , \\ \frac{\omega}{(1-\beta)} & \text{if } \pi^*(\omega) \geq p \end{cases}$$

$$V_\omega(r) = \begin{cases} \sum_{k=0}^{\infty} \beta^k R\left(\frac{r-(p-r)^{k+1}r}{1+r-p}\right) & \text{if } \pi^*(\omega) < r, \\ \upsilon & \text{if } r \leq \pi^*(\omega) < \pi^0 \ , \\ \frac{\omega}{(1-\beta)} & \text{if } \pi^*(\omega) \geq \pi^0 \end{cases}$$

Case 2: $p \leq r$ (negative correlation)

$$V_\omega(p) = \begin{cases} \sum_{k=0}^{\infty} \beta^k R\left(\frac{r+(p-r)^{k+1}(1-p)}{1+r-p}\right) & \text{if } \pi^*(\omega) < p, \\ \frac{\omega+\beta R(Q(p))+\beta^2(1-Q(p))V_\omega(r)}{(1-\beta)^2 Q(p)} & \text{if } p \leq \pi^*(\omega) < Q(p) \ , \\ \frac{\omega}{(1-\beta)} & \text{if } \pi^*(\omega) \geq Q(p) \end{cases}$$

$$V_\omega(r) = \begin{cases} \frac{R(r)+\beta r V_\omega(p)}{1-\beta(1-r)} & \text{if } \pi^*(\omega) < p, \\ \frac{(1-\beta(1-r))[\omega+\beta R(Q(p))]+\beta^2(1-Q(p))R(r)}{(1-\beta(1-r))(1-\beta^2 Q(p))-\beta^3 r(1-Q(p))} & \text{if } Q(p) \leq \pi^*(\omega) < r \ , \\ \frac{\omega}{(1-\beta)} & \text{if } \pi^*(\omega) \geq r \end{cases}$$

Proof. Given in Appendix C.2. □

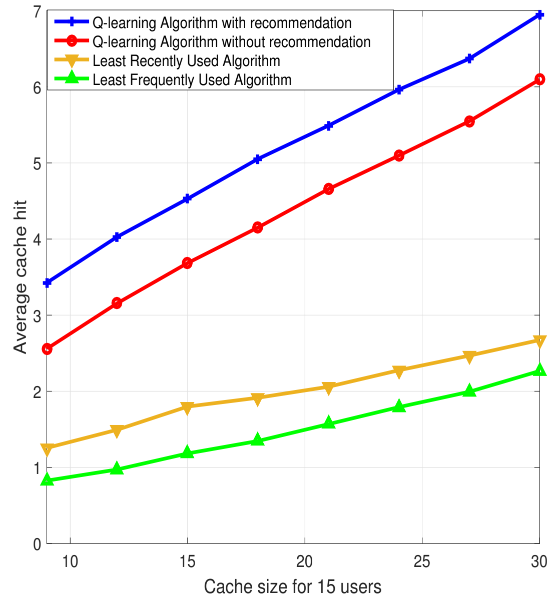
Proposition 5.4.3. *The threshold value is strictly increasing with ω. Therefore, the problem is Whittle indexable.*

Proof. The proof of indexability follows the lines of []. Details are given in Appendix C.3. □

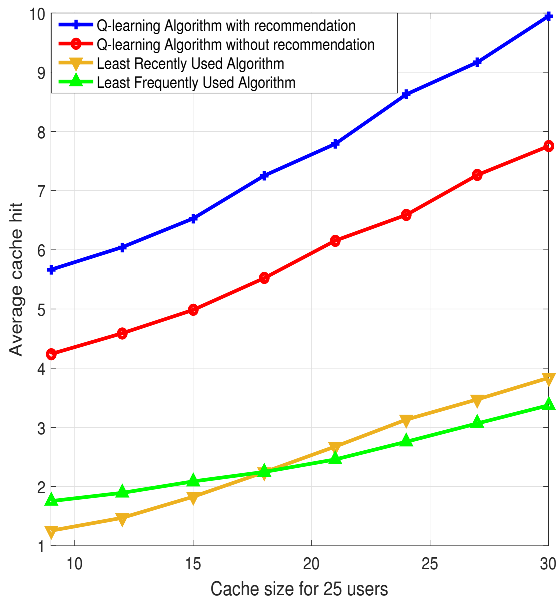
5.5 Simulation Results

In the simulation setup, the number of files taken is 500. The number of SBS is assumed to be 30. For simplicity, in the simulation, the assumption is made that each user requests file of similar size. Average cache hit is obtained over an ensemble of 100 simulation runs.

Fig. 5.2(a) and Fig. 5.2(b) compares the performance of the proposed algorithm (MDP algorithm with recommendation) for the metric average cache hit with (a) MDP algorithm without recommendation, (b) LRU and (c) LFU algorithms. The number of users is 15 and 25 respectively for Fig. 5.2(a) and Fig. 5.2(b). It is observed from Fig. 5.2(a) that with recommendation the average reward obtained is higher compared to all other algorithms. When a file is recommended, the user requests changes and this is captured by the recommendation vector. In Fig. 5.2(a), we see that for the algorithm without recommendation to obtain the average cache hit equal to that of the algorithm with recommendation, it has to increase its cache size by 6 units. Similarly in Fig. 5.2(b), we observe that an increase of 9 cache size is required for the algorithm without recommendation to perform equal to the algorithm with recommendation. Further, in Fig. 5.2(b) we see that as the number of users is increased from 15 to 25, the difference between both the algorithm increases. Thus, the simulation results prove that the recommendation algorithm using whittle index helps in increasing the average cache hit in comparison with the algorithm



(a)



(b)

Figure 5.2: Average cache hit versus cache size for 500 files.

without recommendation and it also gives better result than the existing LFU and LRU algorithms.

5.6 Summary

In this chapter, a model which captures the caching decisions along with recommendation has been introduced. Implications of recommendation on user requests has been studied and it has been seen that recommendation does influence the demands of the users. It has been shown that with recommendation the average reward increases and thus, implying the fact the recommendation does affect the user request. A MDP framework for the cache placement problem has been constructed and a Whittle index based algorithm to update the caches has been developed. Based on the Whittle's indexability condition, the whittle index policy for the caching problem is provided. Thus, in this chapter joint optimization of caching and recommendation has been carried out using Whittle index algorithm. Further, the simulation results show that the proposed algorithm with recommendation performs better when compared with the algorithm which has no recommendation, as well as LRU and LFU algorithms.

Chapter 6

Conclusions and Future Works

6.1 Conclusions

In this thesis, we have focused on proactive caching paradigm by leveraging small cell network deployments and caching capabilities at the edge of network, namely at small cells. Initially, we have characterized the gains of caching for different topologies, content popularity distributions and caching policies. The modeling has been carried out by using recent tools from probability theory, and our expressions for average delivery rate and delay have been validated via numerical simulations. In the second part of the thesis, we have approached to the problem from a distributed point of view, and proposed several joint optimization algorithms for content popularity estimation and algorithmic aspects. The tools from machine learning and enabling big data allowed us to show the benefits of mobile edge caching in practical scenarios, where we have drawn several conclusions based on storage size, content rating density, behaviour of content popularity and caching policy.

Initially, the problem of mobile edge caching in a cellular network with a given topology and storage size in each SBS is looked into. The problem of maximizing the cache hit with respect to the cache placement strategy (which takes values in $\{0, 1\}$) subject to storage size constraint in each SBS turns out to be NP hard, and hence, an approximation is proposed to the problem. Theoretical guarantees are proven on the performance of the proposed method. A complexity analysis is also presented of the proposed algorithm.

In the second chapter, assuming structured cache placement, high probability bounds on the conditional average cache hits are derived using Martingale difference equation [79]. 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.

In the third chapter, two estimation procedures, point and Bayesian estimation are provided. A probabilistic model using Bayesian inference based on Dirichlet distribution is proposed. Specifically, the influence of recommendation on the popularity profile is modelled using a conditional probability distribution. A high probability guarantee on the estimated caching and recommendation strategies is provided. Irrespective of the estimation method, it is shown that with a probability of $1-\delta$ the proposed caching and recommendation strategy is ϵ close to the optimal solution.

Finally, the joint caching and recommendation is modeled as an MDP and its Whittle's indexability is established. Based on the Whittle's indexability condition, the whittle index policy for the caching problem is provided. The Whittle's index policy is known to have near-optimal performance and have shown low-complexity [80].

6.2 Future Work

Despite the fact that mobile edge caching is gainful especially in limited-backhaul scenarios, there exist still several challenges which needs to be investigated in the future. In particular, we have the following future directions.

1. *Integrating with physical layer parameters:* Integrating of the caching parameters such as storage, average cache hit rate, with the physical layer parameters such as SINR, bit rate, etc. would provide a holistic view to the system. Further, integration of caching parameters with the physical parameters can be decomposed and independently investigated.

2. *D2D communications:* D2D communication jointly with cache-enabled cellular networks will provide additional insights to the network designers. The D2D cache-enabled communication requires understanding of user pattern, similarity of content and geographical connectivity to the BSs.

3. *Green aspects:* The energy efficiency and area power consumption during caching is of high interest, since the energy required to prefetch the files from cache is usually considered low as compared to the energy required to prefetch the files from backhaul. Some of the recent works [14, 143], have started to focus on these areas, however more investigations are needed to gain insights about the network deployment.

4. *Average delivery rate:* The performance metric we have defined is based on fixed rate transmission and does not exploit the full potential of instantaneous signal to noise ratio to achieve higher rates in downlink. Even though this is done for tractability and we expect that new insights might be somewhat similar, additional effort for detailing of this metric is of high interest, in order to have a more realistic view to the system.

Appendix A

Proofs for Chapter 3

A.1 Derivation of (3.3.2)

Proof: Assume that each SBS b employs the caching strategy in (3.4) based on the local data $Z_{b,1}^{T}$. Then, the corresponding conditional average of the hit rate is given by

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

where (a) follows simply by substituting for $\boldsymbol{\pi}^{*}_{\boldsymbol{b,T+1}}$ from (3.4). 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 (3.6), the above equation can be lower bounded as

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

Similarly, an upper bound can also be obtained as follows

$$\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}}) \mid Z_{b,1}^{T}\right] \leq \sum_{t=T-\tau}^{T} \alpha_{b,t}\mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\right] + \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) + \mathbb{D}_{b,T}(\boldsymbol{\alpha_{b,T}}) \tag{A.2}$$

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

$$\sum_{t=T-\tau}^{T} A_t \leq H_{\mathtt{max}}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}}. \tag{A.3}$$

The above implies that

$$\sum_{t=T-\tau}^{T} \alpha_{b,t}\mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\right] \geq \sum_{t=T-\tau}^{T} \alpha_{b,t}\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) - H_{\mathtt{max}}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}}, \tag{A.4}$$

where $H_{\mathtt{max}}$ 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 $1-\delta$

$$\sum_{t=T-\tau}^{T} \alpha_{b,t}\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \geq \sum_{t=T-\tau}^{T} \alpha_{b,t}\mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) \mid Z_{b,1}^{t-1}\right] - H_{\mathtt{max}}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}} \tag{A.5}$$

Using (A.4) in (A.1), the following holds good with a probability of at least $1-\delta$

$$\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}}) \mid Z_{b,1}^{T}\right] \geq \sum_{t=T-\tau}^{T} \alpha_{b,t}\mathcal{R}_{b,t}(\boldsymbol{\pi}_{b,t}) - H_{\mathtt{max}}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}} - \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) - \mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T}). \tag{A.6}$$

This proves the first result in the theorem. Similar to the above equation, using (A.5) in (A.2), the following holds good with a probability of at least $1-\delta$

$$\begin{aligned} \sum_{t=T-\tau}^{T} \alpha_{b,t}\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) &\geq \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}}) \mid Z_{b,1}^{T}\right] - H_{\mathtt{max}}\|\boldsymbol{\alpha}_{\boldsymbol{b,T}}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}} \\ &- \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) - \mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T}) \end{aligned} \tag{A.7}$$

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

$$\begin{aligned} &- \sum_{t=T-\tau}^{T} \alpha_{b,t}\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) + \sum_{t=T-\tau}^{T} \alpha_{b,t}\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,t}) \\ \leq &\sum_{t=T-\tau}^{T}\left(\alpha_{b,t}-\frac{1}{\tau}\right)\big(\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,t}) - \mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t})\big) \\ + &\frac{1}{\tau}\sum_{t=T-\tau}^{T}\big(\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,t}) - \mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t})\big) \\ \leq & H_{\mathtt{max}}\sum_{t=T-\tau}^{T}\left|\alpha_{b,t}-\frac{1}{\tau}\right| + \frac{\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau}, \end{aligned} \tag{A.8}$$

where the regret is as defined in (4.7). If the caching strategy used is $\boldsymbol{C}^*_{b,t}$, then, the above implies that

$$\begin{aligned} \sum_{t=T-\tau}^{T} \alpha_{b,t}\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) &\geq \sum_{t=T-\tau}^{T} \alpha_{b,t}\mathcal{R}_{b,T+1}(C^*_{b,t}) - H_{max}\sum_{t=T-\tau}^{T}\left|\alpha_{b,t}-\frac{1}{\tau}\right| \\ &- \frac{\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau}. \end{aligned} \tag{A.9}$$

From (A.6), we have

$$\begin{aligned} \mathbb{E}[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}})|Z_{b,1}^{T}] &\geq \sum_{t=T-\tau}^{T} \alpha_{b,t}\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,t}) - H_{max}\sum_{t=T-\tau}^{T}\left|\alpha_{b,T}-\frac{1}{\tau}\right| \\ &-\frac{\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau} - H_{\mathtt{max}}\|\boldsymbol{\alpha}_{\boldsymbol{b,T}}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}} \\ &-\mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) - \mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T,\tau}). \end{aligned} \tag{A.10}$$

Now, using (A.7) 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

$$\begin{aligned}\mathbb{E}[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}})|Z^T_{b,1}] \;\; \geq \;\; & \mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{C}^*_{b,T+1})|Z^T_{b,1}\right] - H_{max}\sum_{t=T-\tau}^{T}\left|\alpha_{b,t}-\frac{1}{\tau}\right| \\ & -\frac{2\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau} - 2H_{\mathtt{max}}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}} \\ & -\mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) - 2\mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T,\tau}). \qquad \text{(A.11)}\end{aligned}$$

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 (A.6) gives

$$\begin{aligned}\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}^{(av)}_{\boldsymbol{b,T+1}}) \mid Z^T_{b,1}\right] \;\; \geq & \sup_{\boldsymbol{\pi}_{b,t}}\sum_{t=T-\tau}^{T}\alpha_{b,t}\mathbb{E}\left[\mathcal{R}_{b,T+1}(\boldsymbol{\pi}_{b,t}) \mid Z^T_{b,1}\right] \\ & -2H_{\mathtt{max}}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2}{\tau}\log\frac{1}{\delta}} - \mathtt{M}_{b,T+1}(\boldsymbol{w}_{\neq b,T}) \\ & -\frac{2\mathtt{Reg}_{b,T,\tau}(\boldsymbol{\pi}_{b,t})}{\tau} - H_{\mathtt{max}}\sum_{t=T-\tau}^{T}\left|\alpha_{b,t}-\frac{1}{\tau}\right| - \\ & 2\mathbb{D}_{b,T}(\boldsymbol{\alpha}_{b,T,\tau}) - \gamma. \qquad \text{(A.12)}\end{aligned}$$

This completes the proof of the theorem.

A.2 Derivation of (3.3.3)

Proof: 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., $|A_{b,s}| \leq \frac{H_{\mathtt{max}}\|\boldsymbol{\alpha_{b,T}}\|_2}{\tau}$. 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

$$\sum_f \pi_{b,t}\hat{d}_{b,f,t}\mathcal{L}_f \leq \frac{1}{\tau}\sum_{s=t-\tau-1}^{t-1}\mathbb{E}\left[\mathcal{R}_{b,s}(\boldsymbol{\pi_b}) \mid Z^{s-1}_{b,1}\right] + H_{\mathtt{max}}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2\log\frac{1}{\delta}}{\tau}}, \text{(A.13)}$$

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 $\mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z_{b,1}^{t-1}\right]$, and taking the supremum of the modulus over caching strategies to get the following bound in terms of discrepancy

$$\sum_f \boldsymbol{\pi}_{b,t}\hat{d}_{b,f,t}\mathcal{L}_f \leq \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z_{b,1}^{t-1}\right] + \mathbb{D}_{b,t}(\boldsymbol{u}_\tau) + H_{\mathtt{max}}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2\log\frac{1}{\delta}}{\tau}} \tag{A.14}$$

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

$$\begin{aligned}\sum_f \boldsymbol{\pi}_{b,t}\hat{d}_{b,f,t}\mathcal{L}_f \quad &\leq \quad \mathbb{E}\left[\mathcal{R}_{b,t}(\boldsymbol{\pi_b}) \mid Z_{G,1}^{t-1}\right] + \mathbb{D}_{b,t}(\boldsymbol{u}_\tau) + \mathbb{D}_{GL,t}(\boldsymbol{\alpha}_b) \\ &+ \quad H_{\mathtt{max}}\|\boldsymbol{\alpha_{b,T}}\|_2\sqrt{\frac{2\log\frac{1}{\delta}}{\tau}} \end{aligned}\tag{A.15}$$

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.

Appendix B

Proofs for Chapter 4

B.1 Derivation of (4.3.1)

Proof: From [144], it follows that

$$\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \boldsymbol{u}^*\boldsymbol{P}_k\boldsymbol{v}^* - \boldsymbol{u}^T\boldsymbol{P}_k\boldsymbol{v} \leq 2 \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left| \boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v} \right|. \tag{B.1}$$

Let $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ be solutions to $\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left| \boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v} \right|$. Since $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ belong to $\mathcal{C}_{c,r}$, for some $i = 1, 2, \ldots, \mathcal{N}_\epsilon$, there exist $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ in $\mathcal{A}_\epsilon$ such that $\|\boldsymbol{x}^* - \boldsymbol{x}_i\|_2 \leq \epsilon/8$, and $\|\boldsymbol{y}^* - \boldsymbol{y}_i\|_2 \leq \epsilon/8$. Further, by adding and subtracting $\boldsymbol{x}_i \widehat{\Delta P}^{(t)} \boldsymbol{y}_i$ and $\boldsymbol{x}_i \widehat{\Delta P}^{(t)} \boldsymbol{y}_i$, we get

$$\left| (\boldsymbol{u}^*)^T \widehat{\Delta P}^{(t)} \boldsymbol{v}^* \right| \leq \left| \boldsymbol{x}_i \widehat{\Delta P}^{(t)} \boldsymbol{y}_i \right| + \frac{\epsilon \|\widehat{\Delta P}^{(t)}\|_{op}}{4}. \tag{B.2}$$

From (B.1) and (B.2)

$$\Pr\left\{ \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \boldsymbol{u}^T\boldsymbol{P}_k\boldsymbol{v} - \boldsymbol{u}^*\boldsymbol{P}_k\boldsymbol{v}^* \geq \epsilon \right\} \leq \Pr\left\{ \bigcup_{i=1}^{\mathcal{N}_\epsilon} \mathcal{B}_i \geq g_\epsilon \right\} \leq \sum_{i=1}^{\mathcal{N}_\epsilon} \Pr\left\{ \mathcal{B}_i \geq \frac{\epsilon}{4} \right\},$$

where $g_\epsilon := \frac{\epsilon}{2} - \epsilon\|\widehat{\Delta P}^{(t)}\|_{op}/4 \leq \frac{\epsilon}{4}$, using $\|\widehat{\Delta P}^{(t)}\|_{op} \leq 1$, and $\mathcal{B}_i := \left| \boldsymbol{x}_i \widehat{\Delta P}^{(t)} \boldsymbol{y}_i \right|$. Using the fact that $\left| \boldsymbol{x}_i \widehat{\Delta P}^{(t)} \boldsymbol{y}_i \right| \leq \kappa \max_j \|\boldsymbol{x}_j\|_1 \max_i \|\boldsymbol{y}_i\|_1 \|\widehat{\Delta P}^{(t)}\|_F \leq \kappa rc \|\widehat{\Delta P}^{(t)}\|_F$, the above can be further bounded to get $\mathcal{N}_\epsilon \Pr\left\{ \|\widehat{\Delta P}^{(t)}\|_F \geq \frac{\epsilon}{4\kappa rc} \right\}$.

This completes the proof.

B.2 Derivation of (4.3.2)

Proof: Consider the following

$$\begin{aligned}\Pr\left\{\|\widehat{\Delta P}^{(t)}\|_F^2 \geq \gamma\right\} &\leq \Pr\left\{\max_{k,l}(p_{kl} - \hat{p}_{kl}^{(t)})^2 \geq \frac{\gamma}{F^2}\right\} \\ &\leq F^2 \Pr\left\{(p_{kl} - \hat{p}_{kl}^{(t)})^2 \geq \frac{\gamma}{F^2}\right\}, \qquad \text{(B.3)}\end{aligned}$$

where $\gamma := \frac{\epsilon}{4\kappa rc}$, and the second inequality above follows from the union bound. Conditioning on $V_{ls} := \sum_{s=1}^{t} v_l^{(s-1)} = m$, there are Nm i.i.d. samples available to estimate p_{kl}. Using Hoeffdings inequality

$$\mathbb{E}\Pr\left\{(p_{kl} - \hat{p}_{kl}^{(t)})^2 \geq \frac{\gamma}{F^2} \mid V_{ls} = m\right\} \leq 2\mathbb{E}\exp\left\{-\frac{2Nm\gamma}{F^2}\right\}.$$

Since V_{ls} is a binomial random variable with parameter q, the above average with respect to V_{ls} becomes

$$2\left(1 - q\left(1 - \exp\{-2N\gamma/F^2\}\right)\right)^t. \qquad \text{(B.4)}$$

The following bound on the left hand side of (4.13) can be obtained using the above in (B.3), and substituting it in (4.13)

$$2\mathcal{N}_\epsilon F^2\left(1 - q\left(1 - \exp\{-2N\gamma/F^2\}\right)\right)^t. \qquad \text{(B.5)}$$

An upper bound on the above can be obtained by using $1 - x \leq e^{-x}$. Using the resulting bound, $\Pr\left\{\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \boldsymbol{u}^T\boldsymbol{P}_k\boldsymbol{v} - \boldsymbol{u}^*\boldsymbol{P}_k\boldsymbol{v}^* \geq \epsilon\right\} < \delta$ provided t satisfies the bound in the theorem.

B.3 Genie Aided Regret Analysis: Heuristics for Two SBSs Case

Consider the instantaneous regret given by $\texttt{Reg}_k(t) := (\boldsymbol{u}_{k,t}^*)^T \mathbf{P}_k \boldsymbol{v}_{k,t}^* \geq \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \boldsymbol{u}_{k,t}^T \mathbf{P}_k \boldsymbol{v}_{k,t}$ at time t. Using the union bound, we can write

$$\Pr\left\{\frac{1}{T}\sum_{t=1}^{T}\texttt{Reg}_k(t) \geq \frac{1}{T}\sum_{t=1}^{T}\epsilon_t\right\} \leq \sum_{t=1}^{T}\Pr\left\{\texttt{Reg}_k(t) \geq \epsilon_t\right\}, \tag{B.6}$$

where $\epsilon_t > 0$. From Theorem 4.3.2, it follows that for any $\epsilon_t > 0$, we have $\Pr\{\texttt{Reg}_k(t) \leq \epsilon_t\} \geq 1-\delta$ provided (4.14). By choosing $\delta = \frac{1}{T^2}$, the approximation $e^{-x} \approx 1-x$ for small x, and $|\mathcal{N}_\epsilon| \lessapprox 1/\epsilon^F$ in Theorem 4.3.2, we get $\Pr\{\texttt{Reg}_k(t) \leq \epsilon_t\} \geq 1 - \frac{1}{T^2}$ provided

$$t \gtrapprox \frac{8\kappa^2F^2c^2r^2}{qN\epsilon_t^3}\log\frac{2F^2T^2}{\epsilon_t^F}, \tag{B.7}$$

where $\gtrapprox$ is used to denote "approximately greater than or equal to". Assuming $\epsilon_t < 1$ and using $\log x \approx x$ for small x, we have $\log\frac{2F^2T^2}{\epsilon_t^F} = \log 2F^2T^2 + F\log\frac{1}{\epsilon_t} \leq \frac{(\log 2F^2T^2+F)}{\epsilon_t}$.[1] Now, we can use (B.7) to write ϵ_t in terms of t to get

$$\epsilon_t \gtrapprox \sqrt[3]{\frac{8\kappa^2F^2c^2r^2(\log 2F^2T^2+F)}{qNt}}. \tag{B.8}$$

In other words, with a probability of at least $1-\frac{1}{T^2}$, $\texttt{Reg}_k(t) \leq \sqrt[3]{\frac{8\kappa^2F^2c^2r^2(\log 2F^2T^2+F)}{qNt}}$. Using this result in (B.6), we get the following result. With a probability of at least $1-\frac{1}{T}$,

$$\texttt{Reg}_{k,T} \lessapprox \sqrt[3]{\frac{8\kappa^2F^2c^2r^2(\log 2F^2T^2+F)}{qN}}\sum_{t=1}^{T}\frac{1}{t^{1/3}} = \mathcal{O}(T^{2/3}\sqrt{\log T}). \tag{B.9}$$

Thus, the above shows that the regret achieved grows sub-linearly with time, and hence (genie aided) achieves a zero asymptotic average regret.

[1] The case of $\epsilon_t > 1$ can be handled in a similar fashion, and hence ignored.

B.4 Regret Analysis for Two SBS: Heuristics

The analysis here is very similar to the analysis of single BS case. We repeat some of the analysis for the sake of clarity and completeness. Let the instantaneous regret at the BS k at time t is given by $\mathtt{Reg}_k(t) := (\boldsymbol{u}_{k,t}^*)^T \mathbf{P}_k \boldsymbol{v}_{k,t}^* \geq \sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \boldsymbol{u}_{k,t}^T \mathbf{P}_k \boldsymbol{v}_{k,t}$ at time t. Using the union bound, we can write

$$\Pr\left\{\frac{1}{T}\sum_{t=1}^{T} \mathtt{Reg}_k(t) \geq \frac{1}{T}\sum_{t=1}^{T} \epsilon_{k,t}\right\} \leq \sum_{t=1}^{T} \Pr\left\{\mathtt{Reg}_k(t) \geq \epsilon_{k,T}\right\}, \tag{B.10}$$

where $\epsilon_{k,t} > 0$ and $\epsilon_{k,T} = \frac{1}{T}\sum_{t=1}^{T} \epsilon_{k,t}$. From Theorem 4.3.2, it follows that for any $\epsilon_{k,t} > 0$, we have $\Pr\left\{\mathtt{Reg}_k(t) \leq \epsilon_{k,t}\right\} \geq 1 - \delta$ provided (B.6) is satisfied. By choosing $\delta = \frac{1}{T^2}$, assuming $\epsilon_{k,t} < 1$, using the approximations $e^{-x} \approx 1 - x$ for small x, and $|\mathcal{N}_\epsilon| \lessapprox 1/\epsilon^F$, we have $\log \frac{2F^2T^2}{\epsilon_{k,t}^F} \leq \frac{(\log 2F^2T^2 + F)}{\epsilon_{k,t}}$. Using this in Theorem 4.3.2, we get $\Pr\left\{\mathtt{Reg}_k(t) \leq \epsilon_{k,t}\right\} \geq 1 - \frac{1}{T^2}$ provided

$$\tau\left(\frac{\epsilon_{k,t}}{\lambda_k}, \frac{\delta}{2}\right) \gtrapprox \frac{8\kappa^2 F^2 c^2 r^2 \lambda_k^3 (\log 4F^2T^2 + F)}{qN\epsilon_{k,t}^3}, \tag{B.11}$$

where $\gtrapprox$ is used to denote "approximately greater than or equal to". Using the above in (B.6), we get

$$t \gtrapprox \max\left\{\frac{8\kappa^2 F^2 c^2 r^2 \lambda_k^3 (\log 4F^2T^2 + F)}{qN\epsilon_{k,t}^3}, \frac{8\kappa^2 F^2 c^2 r^2 (1-\lambda_k)^3 (\log 4F^2T^2 + F)}{qN\epsilon_{k,t}^3}\right\}.$$

By rearranging and summing over t, the error can be written as follows

$$\epsilon_{k,t} \lessapprox \frac{1}{\sqrt[3]{t}} \max\left\{\Theta\lambda_k, \Theta(1-\lambda_k)\right\},$$

where $\Theta = \sqrt[3]{\frac{8\kappa^2 F^2 c^2 r^2 (\log 4F^2T^2 + F)}{qN}}$. Using this in the place of $\epsilon_{k,t}$ in the above theorem, and summing over t, we get with a probability of at least $1 - \frac{1}{T}$, the following holds for BS k

$$\mathtt{Reg}_{k,T} \;\lessapprox\; \max\left\{\Theta\lambda_k, \Theta(1-\lambda_k)\right\} \frac{1}{\sqrt[3]{T}} \sum_{t=1}^{T} 1 + 2T(1-\lambda_k)\mathcal{V}_{12},$$

where $\mathcal{V}_{12} := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T(\mathbf{P}_2 - \mathbf{P}_1)\boldsymbol{v}\right|$ and $\Theta = \sqrt[3]{\frac{8\kappa^2F^2c^2r^2(\log 4F^2T^2+F)}{qN}}$. This completes the approximate analysis.

B.4.1 Regret Analysis for Multiple SBS: Heuristics

The analysis here is again very similar to the analysis of single BS case. From Theorem 4.3.2, it follows that for any $\epsilon_{k,t} > 0$, we have $\Pr\{\mathtt{Reg}_k(t) \leq \epsilon_{k,t}\} \geq 1-\delta$, where $\epsilon_{k,t} > 0$ and $\epsilon_k = \frac{1}{T}\sum_{t=1}^{T}\epsilon_{k,t}$. By choosing $\delta = \frac{1}{T^2}$, assuming $\epsilon_{k,t} < 1$, using the approximation $e^{-x} \approx 1 - x$ for small x, and $|\mathcal{N}_\epsilon| \lessapprox 1/\epsilon^F$, we have $\log\frac{2F^2T^2}{\epsilon_{k,t}^F} \leq \frac{(\log 2F^2T^2+F)}{\epsilon_{k,t}}$. Using this in Theorem 4.3.2, we get $\Pr\{\mathtt{Reg}_k(t) \leq \epsilon_{k,t}\} \geq 1 - \frac{1}{T^2}$ provided

$$\tau\left(\frac{\epsilon_1}{\lambda_1^{(k)}}, \frac{\delta}{M}\right) \gtrapprox \frac{8\kappa^2F^2c^2r^2\lambda_k^3(\log 4F^2T^2+F)}{qN\epsilon_{k,t}^3}, \tag{B.12}$$

where $\gtrapprox$ is used to denote "approximately greater than or equal to". Using the above in (B.3), we get

$$t \gtrapprox \frac{\Theta^3}{\epsilon_{k,t}^3}\max\left\{(\lambda_1^{(k)})^3, (\lambda_2^{(k)})^3, \ldots, (\lambda_M^{(k)})^3\right\},$$

where, $\Theta = \sqrt[3]{\frac{8\kappa^2F^2c^2r^2(\log F^2T^2+F)}{qN}}$. From the above, it is clear that the waiting time t scales as the square of F, c and r, and is inversely proportional to the error $\epsilon_{k,t}^3$. By rearranging and summing over t, the error can be written as $\epsilon_{k,t} \lessapprox \frac{1}{\sqrt[3]{t}}\max\left\{\Theta\lambda_1^{(k)}, \ldots, \Theta\lambda_M^{(k)}\right\}$. Using this in the place of $\epsilon_{k,t}$ in the above theorem, and summing over t, we get with a probability of at least $1 - \frac{1}{T}$ the following result on the regret for BS k holds

$$\mathtt{Reg}_{k,T} \lessapprox \frac{\Theta M^2}{\sqrt[3]{T}}\max\left\{\lambda_1^{(k)}, \ldots, \lambda_M^{(k)}\right\}\sum_{t=1}^{T} 1 + M^2T\big\{(1-\lambda_1^{(k)})\mathcal{D}_1 - \ldots, -\lambda_M^{(k)}\mathcal{D}_M\big\}$$

Remark: Note that the value of regret depends on the values of $\lambda_1^{(k)}, \ldots, \lambda_M^{(k)}$ and the term $\mathcal{D}_k$. The first term scales as $T^{2/3}$ while the second term scales with T linearly. This can be balanced by using $\lambda_1 = 1 - \frac{1}{\sqrt{T}}$ and $\lambda_k = \frac{1}{(M-1)\sqrt{T}}$, which results in $\mathcal{O}(\sqrt{T})$ scaling of regret. Similar to the single SBS case, the choice $\lambda_k = \frac{1}{(M-1)\sqrt{T}}$ reveals that as time progresses, i.e., as the BS k collects more samples, the weights allocated to the neighboring BS should go down to zero, as expected. Otherwise, one can optimize the above regret with respect to λ_k's, and find the optimal choice.

sec

B.5 Derivation of (4.3.4)

Proof: Similar to the proof of Theorem 4.3.1, from [144], it follows that at time t, the performance gap of the proposed algorithm with respect to the optimal is given by

$$\boldsymbol{u}^* \boldsymbol{P}_k \boldsymbol{v}^* - \boldsymbol{u}_t^T \boldsymbol{P}_k \boldsymbol{v}_t \;\leq\; 2 \sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \left| \boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v} \right| .$$

Summing the above over all t, we get

$$T\boldsymbol{u}^* \boldsymbol{P}_k \boldsymbol{v}^* - \sum_t \boldsymbol{u}_t^T \boldsymbol{P}_k \boldsymbol{v}_t \leq 2 \sum_t \sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \left| \boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v} \right| .$$

For a given ϵ, the above implies that

$$\begin{aligned} \Pr\left\{ T\boldsymbol{u}^* \boldsymbol{P}_k \boldsymbol{v}^* - \sum_t \boldsymbol{u}_t^T \boldsymbol{P}_k \boldsymbol{v}_t \geq \frac{\epsilon}{2} \right\} &\leq \Pr\left\{ \sum_t \sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \left| \boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v} \right| \geq \frac{\epsilon}{2} \right\} \\ &= \Pr\left\{ \sum_t Y_t \geq \frac{\epsilon}{2} - \sum_t \mathbb{E}\left[\sup_{(\boldsymbol{u},\boldsymbol{v}) \in \mathcal{C}_{c,r}} \left| \boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v} \right| \right] \right\}, \end{aligned} \tag{B.13}$$

where $Y_t = \sup_{(u,v)\in\mathcal{N}_\epsilon} \left| \boldsymbol{u} \widehat{\Delta P}^{(t)} \boldsymbol{v} \right| - \mathbb{E}\left[\sup_{(u,v)\in\mathcal{N}_\epsilon} \left| \boldsymbol{u} \widehat{\Delta P}^{(t)} \boldsymbol{v} \right| \right]$ is a martingale difference, i.e., $\mathbb{E}\{Y_t\} = 0$, and $\mathcal{N}_\epsilon$ is the covering set of $\mathcal{C}_{c,r}$ as in Definition I. By Azuma's inequality, we have

$$\Pr\left\{ \sum_t Y_t > \frac{\epsilon}{2} \right\} \leq \exp\left\{ \frac{-\epsilon^2}{2(4rc)^2} \right\}. \tag{B.14}$$

The above follows due to the fact that $|Y_t| \leq 4rc$, which is explained below:

$$\begin{aligned} Y_t &\leq \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right| + \mathbb{E}\left[\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right|\right] \\ &\leq \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T P \boldsymbol{v}\right| + \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T \widehat{P}^{(t)} \boldsymbol{v}\right| + \mathbb{E}\left[\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T P \boldsymbol{v}\right|\right] \\ &\quad + \mathbb{E}\left[\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T \widehat{P}^{(t)} \boldsymbol{v}\right|\right] \\ &\leq 4rc, \end{aligned} \tag{B.15}$$

which follows from $|Y_t| \leq 4rc$ and $\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T P \boldsymbol{v}\right| \leq rc$. Thus it follows from (B.14), $\Pr\left\{\sum_t Y_t > \frac{\epsilon}{2}\right\} \leq \delta$ if $\epsilon \geq 32r^2c^2T\log(1/\delta)$. Using this definition of $|Y_t|$, it follows that with a probability of at most δ, we have

$$\sum_t \sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}\widehat{\Delta P}^{(t)} \boldsymbol{v}\right| \geq 2\sum_t \mathbb{E}\left[\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right|\right] + \sqrt{128r^2c^2T\log(1/\delta)}.$$

Choosing $\epsilon = 32r^2c^2T\log(1/\delta)$ in (B.13), the following bound for regret is satisfied with a probability of at least $1-\delta$:

$$\boldsymbol{u}^* \boldsymbol{P}_k \boldsymbol{v}^* - \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{u}_t^T \boldsymbol{P}_k \boldsymbol{v}_t < \frac{2}{T}\sum_t \mathbb{E}\left[\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right|\right] + \sqrt{\frac{128r^2c^2\log(1/\delta)}{T}}.$$

Now, it remains to bound the first term on the right hand side above. For a given $\psi_t > 0$ (to be chosen later), using total expectation rule, we get

$$\begin{aligned} \sum_t \mathbb{E}\left[\sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right|\right] &\leq \sum_t \mathbb{E}\left[\sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right|, \sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right| > \psi_t\right] \\ &\quad \times \Pr\left\{\sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right| > \psi_t\right\} + \sum_t \psi_t \\ &\leq \sum_t rc \max_{kl} p_{kl} \Pr\left\{\sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right| > \psi_t\right\} + \sum_t \psi_t, \end{aligned}$$

where the first inequality above follows by using the bound $\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right| \leq$

ψ_t. Since $\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T\widehat{\Delta P}^{(t)}\boldsymbol{v}\right| > \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T\widehat{\Delta P}^{(t)}\boldsymbol{v}\right|$, we get

$$\begin{aligned}\sum_t \mathbb{E}\Bigg[\sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T\widehat{\Delta P}^{(t)}\boldsymbol{v}\right|\Bigg] \;&\leq\; rc|\mathcal{N}_\epsilon| \max_{kl} p_{kl} \sum_t \Pr\left\{\sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T\widehat{\Delta P}^{(t)}\boldsymbol{v}\right| > \psi_t\right\} \\ &\quad + \sum_t \psi_t. \end{aligned} \tag{B.16}$$

Now, consider

$$\begin{aligned}\sum_t \Pr\left\{\sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T\widehat{\Delta P}^{(t)}\boldsymbol{v}\right| > \psi_t\right\} \;&\overset{(a)}{\leq}\; \sum_t \Pr\left\{\sum_{i,j=1}^{|\mathcal{N}_\epsilon|} \boldsymbol{u}_i^T\widehat{\Delta P}^{(t)}\boldsymbol{v}_j > \psi_t\right\} \\ &\leq\; \sum_t \Pr\left\{\sum_{i,j=1}^{|\mathcal{N}_\epsilon|}\sum_{l=1}^{|\mathcal{N}_\epsilon|} u_{ij}\widehat{\Delta P}_j^{(t)T} v_l > \psi_t\right\} \\ &\leq\; \sum_t \Pr\left\{e^{s\sum_{i,j=1}^{|\mathcal{N}_\epsilon|} u_{ij}^T X_j} > e^{s\psi_t}\right\} \\ &\overset{(b)}{\leq}\; \sum_t e^{-s\psi_t}\mathbb{E}[e^{s\sum_{i,j=1}^{|\mathcal{N}_\epsilon|} u_{ij}^T X_j}] \end{aligned} \tag{B.17}$$

where $X_j := \sum_{l=1}^{|\mathcal{N}_\epsilon|} v_l\widehat{\Delta P}_{lj}^{(t)}$. In the above, (a) follows from the covering argument, and (b) follows from the Chernoff bound. From [145], using the optimal proxy variance, we get the following bound

$$\sum_t \Pr\left\{\sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T\widehat{\Delta P}^{(t)}\boldsymbol{v}\right| > \psi_t\right\} \leq \sum_t \exp\left\{-s\psi_t + s^2 \sum_{i,j,l=1}^{|\mathcal{N}_\epsilon|} \frac{\|u_{il}\|^2\|v_j\|^2\sigma_j^2}{2}\right\} \tag{B.18}$$

where an upper bound on σ_l (see [145]) is given by $\sigma_j \leq \frac{1}{4(\sum_i \alpha_{ij}^{(t)}+1)}$ and $\alpha_{ij}^{(t)} = \sum_{q=1}^{t-1} d_i^{(q)} v_j^{(q-1)}$. Optimizing the exponent in (B.18), the optimal $s^* = \frac{\psi_t}{\sum_{i,j,l=1}^{|\mathcal{N}_\epsilon|}\|u_{il}\|^2\|v\|_F^2\sigma_j^2}$. Further, $\|u_{il}\|^2 \leq \|u_{il}\| \leq c$ and $\|v\|_F^2 \leq r|\mathcal{N}_\epsilon|^2$. Using these bounds, and the bound on σ_l above, (B.18) can be written as follows

$$\sum_t \Pr\left\{\sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T\widehat{\Delta P}^{(t)}\boldsymbol{v}\right| > \psi_t\right\} \leq \sum_t \exp\left\{-\frac{8\psi_t^2}{cr\left|\mathcal{N}_\epsilon\right|^2\bar{\sigma}^2(t)}\right\}, \tag{B.19}$$

where $\bar{\sigma}^2(t) := \left[\sum_j \frac{1}{\left(\sum_i \alpha_{ij}^{(t)}+1\right)^2}\right]$. Substituting the above in (B.16) results in

$$\sum_t \mathbb{E}\left[\sup_{(u,v)\in\mathcal{N}_\epsilon} \left|\boldsymbol{u}^T\widehat{\Delta P}^{(t)}\boldsymbol{v}\right|\right] \leq rc\max_{ij} p_{ij}|N_\epsilon| \sum_t \exp\left\{\frac{-8\psi_t^2}{cr\left|\mathcal{N}_\epsilon\right|^2 \bar{\sigma}^2(t)}\right\} + \sum_t \psi_t.$$

Thus, using the above, the regret can be written as

$$\mathtt{Reg}_T \leq 2rc\max_{ij} p_{ij}|\mathcal{N}_\epsilon| \sum_t \exp\left\{\frac{-8\psi_t^2}{cr\left|\mathcal{N}_\epsilon\right|^2 \bar{\sigma}^2(t)}\right\} + 2\sum_t \psi_t + \sqrt{128r^2c^2T\log(1/\delta)}.$$

This completes the proof.

B.6 Derivation of (4.4.1)

Proof: The analysis is done only for the first SBS as the analysis for the second SBS is similar. As in (B.1), since $\Pr\left\{(\boldsymbol{u}_{k,t}^*)^T\mathbf{P}_k\boldsymbol{v}_{k,t}^* \geq \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \boldsymbol{u}^T\hat{\boldsymbol{Q}}_k^{(t)}\boldsymbol{v} - \epsilon\right\} \leq \Pr\left\{2\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T\widehat{\Delta P}^{(t)}\boldsymbol{v}\right| > \epsilon\right\}$, it is sufficient to consider the following

$$\begin{aligned}
\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T\widehat{\Delta P}^{(t)}\boldsymbol{v}\right| &= \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T(\mathbf{P}_1 - \hat{\boldsymbol{Q}}_1^{(t)})\boldsymbol{v}\right| \\
&= \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\lambda_1\boldsymbol{u}^T\widehat{\Delta P}_1^{(t)}\boldsymbol{v} + (1-\lambda_1)\boldsymbol{u}^T\widehat{\Delta P}_2^{(t)}\boldsymbol{v} + (1-\lambda_1)\boldsymbol{u}^T(\mathbf{P}_1 - \mathbf{P}_2)\boldsymbol{v}\right| \\
&\leq \ \lambda_1\mathcal{V}_1 + (1-\lambda_1)\mathcal{V}_2 + (1-\lambda_1)\mathcal{V}_{12},
\end{aligned}$$

where $\mathcal{V}_1 := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T\widehat{\Delta P_1}^{(t)}\boldsymbol{v}\right|$, $\mathcal{V}_2 := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T\widehat{\Delta P_2}^{(t)}\boldsymbol{v}\right|$, and $\mathcal{V}_{12} := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T(\mathbf{P}_2 - \mathbf{P}_1)\boldsymbol{v}\right|$. Here, $\widehat{\Delta P}^{(t)} := \mathbf{P}_1 - \hat{Q}_1^{(t)}$, $\widehat{\Delta P_k}^{(t)} := \mathbf{P}_k - \hat{\mathbf{P}}_k^{(t)}$, $k = 1, 2$. Using the union bound, we get the following

$$\Pr\{\lambda_1\mathcal{V}_1 + (1-\lambda_1)\mathcal{V}_2 > \epsilon_1\} \leq \Pr\left\{\mathcal{V}_1 > \frac{\epsilon_1}{\lambda_1}\right\} + \Pr\left\{\mathcal{V}_2 > \frac{\epsilon_1}{(1-\lambda_1)}\right\},$$

where $\epsilon_1 := \epsilon/2 - (1-\lambda_1)\mathcal{V}_{12}$.Using results from Theorem 4.3.2 to each of the above term with ϵ replaced by ϵ_1/λ_1 and $\frac{\epsilon_1}{(1-\lambda_1)}$ with δ replaced by $\delta/2$, we get the following bound on the regret: the regret $\mathtt{Reg}_{k,T} < \epsilon$, i.e., $\Pr\Big\{(\boldsymbol{u}_{k,t}^*)^T\mathbf{P}_k\boldsymbol{v}_{k,t}^* \geq$

$\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \boldsymbol{u}^T\hat{\boldsymbol{Q}}_k^{(t)}\boldsymbol{v} - \epsilon_k \Big\} > 1-\delta$ provided

$$t \geq \max\left\{\tau\left(\frac{\epsilon_k}{\lambda_k}, \frac{\delta}{2}\right), \tau\left(\frac{\epsilon_k}{(1-\lambda_k)}, \frac{\delta}{2}\right)\right\},$$

where

$$\tau(\epsilon,\delta) := \frac{1}{q\left(1-\exp\{-\frac{N\epsilon^2}{8\kappa^2F^2c^2r^2}\}\right)}\log\frac{2|\mathcal{N}_\epsilon|F^2}{\delta}.$$

Further, $\epsilon_k := \epsilon/2 - (1-\lambda_k)\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}}\left|\boldsymbol{u}^T(\mathbf{P}_2-\mathbf{P}_1)\boldsymbol{v}\right|$. This completes the proof.

B.7 Derivation of (4.4.2)

Similar to the proof provided of Theorem 4.1, the analysis is done only for the first SBS using Bayesian estimate.

$$T\boldsymbol{u}^*\mathbf{P}_1\boldsymbol{v}^* - \sum_t \boldsymbol{u}_t^T\hat{\boldsymbol{Q}}_k^{(t)}\boldsymbol{v}_t \leq \lambda_1\mathcal{U}_1 + (1-\lambda_1)\mathcal{U}_2 + (1-\lambda_1)\mathcal{U}_{12},$$

where $\mathcal{U}_1 := \sum_t \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}}\left|\boldsymbol{u}^T\widehat{\Delta P_1}^{(t)}\boldsymbol{v}\right|$, $\mathcal{U}_2 := \sum_t \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}}\left|\boldsymbol{u}^T\widehat{\Delta P_2}^{(t)}\boldsymbol{v}\right|$, and $\mathcal{U}_{12} := \sum_t \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}}\left|\boldsymbol{u}^T(\mathbf{P}_1-\mathbf{P}_2)\boldsymbol{v}\right|$. Here, $\widehat{\Delta P_k}^{(t)} := \mathbf{P}_k - \hat{\mathbf{P}}_k^{(t)}$, $k=1,2$. Consider the following

$$\begin{aligned}\Pr\{T\boldsymbol{u}^*\mathbf{P}_1\boldsymbol{v}^* - \sum_t \boldsymbol{u}_t^T\hat{\boldsymbol{Q}}_1^{(t)}\boldsymbol{v}_t \geq \epsilon\} &\leq \Pr\{\lambda_1\mathcal{U}_1 + (1-\lambda_1)\mathcal{U}_2 + (1-\lambda_1)\mathcal{U}_{12} \geq \epsilon\} \\ &\leq \Pr\left\{\mathcal{U}_1 > \frac{\epsilon_1}{\lambda_1}\right\} + \Pr\left\{\mathcal{U}_2 > \frac{\epsilon_1}{(1-\lambda_1)}\right\},\end{aligned}$$

where $\epsilon_1 := \epsilon/2 - (1-\lambda_1)\mathcal{U}_{12}$. Using results from Theorem 4.3.3 to each of the above term with ϵ replaced by $2\epsilon_1/\lambda_1$ and $\frac{2\epsilon_1}{(1-\lambda_1)}$ and δ replaced by $\delta/2$, we get the following bound on regret:

$$\mathtt{Reg}_{k,T} \leq \max\left\{R_k\left(\frac{2\epsilon_k}{\lambda_k}, \frac{\delta}{2}\right), R_k\left(\frac{2\epsilon_k}{(1-\lambda_k)}, \frac{\delta}{2}\right)\right\},$$

where

$$R_k(\epsilon,\delta) := 2rc\max_{ijk} p_{ijk}|\mathcal{N}_\epsilon|\sum_t \exp\left\{\frac{-8\psi_t^2}{cr\left|\mathcal{N}_\epsilon\right|^2\bar{\sigma}_k^2(t)}\right\} + 2\sum_t \psi_t + \sqrt{128r^2c^2T\log(1/\delta)},$$

and $\bar{\sigma}_k^2(t) := \left[\sum_{j=1}^{F} \frac{1}{\left(\sum_i \alpha_{ijk}^{(t)}+1\right)^2}\right]$, $\alpha_{ijk}^{(t)} = \sum_{q=1}^{t-1} d_{ik}^{(q)} v_{jk}^{(q-1)}$, and $\epsilon_k := \epsilon/2 - (1-\lambda_k)\sum_t \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T(\mathbf{P}_1 - \mathbf{P}_2)\boldsymbol{v}\right|$. This completes the proof.

B.8 Derivation of (4.4.3)

Proof: The proof for multiple SBSs is a generalization of two SBSs and the analysis is done only for the first SBS, the rest of the SBSs are similar.

$$\begin{aligned}
\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T \widehat{\Delta P}^{(t)} \boldsymbol{v}\right| &= \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T(\mathbf{P}_1 - \lambda_1^{(k)}\hat{\mathbf{P}}_1^{(t)}, \ldots - \lambda_M^{(k)}\hat{\mathbf{P}}_M^{(t)})\boldsymbol{v}\right| \\
&= \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} |\lambda_1^{(k)}\boldsymbol{u}^T(\mathbf{P}_1 - \hat{\mathbf{P}}_1^{(t)})\boldsymbol{v} + \lambda_2^{(k)}\boldsymbol{u}^T(\mathbf{P}_2 - \hat{\mathbf{P}}_2^{(t)})\boldsymbol{v}, \ldots \\
&\quad + \lambda_M^{(k)}\boldsymbol{u}^T(\mathbf{P}_M - \hat{\mathbf{P}}_M^{(t)})\boldsymbol{v} + (1-\lambda_1^{(k)})\{\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T\mathbf{P}_1\boldsymbol{v}\right|\}, \ldots, \\
&\quad - \lambda_M^{(k)}\{\sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T P_M \boldsymbol{v}\right|\} \\
&\leq \lambda_1^{(k)}\mathcal{V}_1 + \lambda_2^{(k)}\mathcal{V}_2 +, \ldots, + \lambda_M^{(k)}\mathcal{V}_M + (1-\lambda_1^{(k)})\mathcal{D}_1 - \lambda_2^{(k)}\mathcal{D}_M -, \ldots, \\
&\quad - \lambda_M^{(k)}\mathcal{D}_M,
\end{aligned}$$

where $\mathcal{V}_l := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T \widehat{\Delta P_l}^{(t)} \boldsymbol{v}\right|$, $l = 1, 2, \ldots, M$, $\mathcal{D}_1 := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T\mathbf{P}_1\boldsymbol{v}\right|$, $\mathcal{D}_2 := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T\mathbf{P}_2\boldsymbol{v}\right|$, $\ldots$, $\mathcal{D}_M := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T P_M \boldsymbol{v}\right|$. Here, $\widehat{\Delta P_k}^{(t)} := \mathbf{P}_1 - \hat{\mathbf{Q}}_1^{(t)}$, $\widehat{\Delta P_k}^{(t)} := \mathbf{P}_k - \hat{\mathbf{P}}_k^{(t)}$, $k = 1, 2, \ldots, M$. Thus we can write the following:

$$\Pr\{\lambda_1^{(k)}\mathcal{V}_1 + \ldots, + \lambda_M^{(k)}\mathcal{V}_M > \epsilon'\} \leq \sum_{l=1}^{M} \Pr\left\{\mathcal{V}_l > \frac{\epsilon_l}{\lambda_l^{(k)}}\right\},$$

where $\epsilon' := \epsilon/M - (1-\lambda_1^{(k)})\mathcal{D}_1 + \lambda_2^{(k)}\mathcal{D}_2 +, \ldots, + \lambda_M^{(k)}\mathcal{D}_M$, and $\epsilon_k := \epsilon/M^2 - (1 - \lambda_1^{(k)})\mathcal{D}_1 + \lambda_2^{(k)}\mathcal{D}_2 +, \ldots, + \lambda_M^{(k)}\mathcal{D}_M, \forall\ k = 1, 2, \ldots, M$. Using results from Theorem 4.3.2 to each of the above term with ϵ replaced by ϵ' and ϵ_k and δ replaced by δ/M we get the following bound on regret, i.e., $\texttt{Reg}_{k,T}$ is less than ϵ $\left\{\text{i.e. } (\boldsymbol{u}_{o,t}^*)^T\mathbf{P}_1\boldsymbol{v}_{o,t}^* \geq \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \boldsymbol{u}^T\mathbf{P}_1\boldsymbol{v} - \epsilon\right\}$ provided

$$t \geq \max\left\{\tau\left(\frac{\epsilon_1}{\lambda_1^{(k)}}, \frac{\delta}{M}\right), \tau\left(\frac{\epsilon_2}{\lambda_2^{(k)}}, \frac{\delta}{M}\right), \ldots, \tau\left(\frac{\epsilon_M}{\lambda_M^{(k)}}, \frac{\delta}{M}\right)\right\},$$

$$\tau(\epsilon, \delta) := \frac{1}{q\left(1 - \exp\{-\frac{N\epsilon^2}{8\kappa^2 F^2 c^2 r^2}\}\right)} \log \frac{2\mathcal{N}_\epsilon F^2}{\delta},$$

where, $\epsilon_k := \epsilon/M^2 - (1 - \lambda_1^{(k)})\mathcal{D}_1 + \lambda_2^{(k)}\mathcal{D}_2+, \ldots, +\lambda_M^{(k)}\mathcal{D}_M$, and $\mathcal{D}_k := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T P_k \boldsymbol{v}\right| \; \forall\; k = 1, 2, \ldots, M$. This completes the proof.

B.9 Derivation of (4.4.4)

Proof: The proof for multiple SBSs is a generalization of two SBSs and the analysis is done only for the first SBS for Bayesian estimate, the rest of the SBSs are similar. First consider the following

$$T\boldsymbol{u}^*\mathbf{P}_1\boldsymbol{v}^* - \sum_t \boldsymbol{u}_t^T\mathbf{P}_1\boldsymbol{v}_t \leq \sum_{j=1}^{M} \lambda_j^{(k)}\mathcal{W}_j + (1 - \lambda_1^{(k)})\mathcal{I}_1 \; - \sum_{j=2}^{M} \lambda_j^{(k)}\mathcal{I}_j,$$

where $\mathcal{W}_j := \sum_t \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T \widehat{\Delta P_j}^{(t)} \boldsymbol{v}\right|, \mathcal{I}_j := \sum_t \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T\mathbf{P}_j\boldsymbol{v}\right|, j = 1, 2, \ldots, M$. Here, $\widehat{\Delta P_k}^{(t)} := \mathbf{P}_k - \hat{\mathbf{P}}_k^{(t)}$, $k = 1, 2, \ldots, M$. Thus we can write the following:

$$\Pr\{\lambda_1^{(k)}\mathcal{W}_1, .., +\lambda_M^{(k)}\mathcal{W}_M \geq \epsilon^{'}\} \leq \sum_{l=1}^{M} \Pr\left\{\mathcal{W}_l > \frac{\epsilon_l}{\lambda_l^{(k)}}\right\},$$

where $\epsilon^{'} := \epsilon/M - (1 - \lambda_1^{(k)})\mathcal{I}_1 + \sum_{l=2}^{M} \lambda_l^{(k)}\mathcal{I}_k$, and $\epsilon_k := \epsilon/M^2 - (1 - \lambda_1^{(k)})\mathcal{I}_1 + \sum_{l=2}^{M} \lambda_l^{(k)}\mathcal{I}_k$, for all $k = \{1, 2, \ldots, M\}$. Using results from Theorem 4.3.3 to each of the above term with ϵ replaced by $\epsilon^{'}$ and ϵ_k and δ replaced by δ/M, we get the following bound on the regret for BS k:

$$\texttt{Reg}_{k,T} \leq \max\left\{R_k\left(\frac{\epsilon_1}{\lambda_1^{(k)}}, \frac{\delta}{M}\right), R_k\left(\frac{\epsilon_2}{\lambda_2^{(k)}}, \frac{\delta}{M}\right), \ldots, R_k\left(\frac{\epsilon_M}{\lambda_M^{(k)}}, \frac{\delta}{M}\right)\right\},$$

where,

$$R_k(\epsilon, \delta) := 2rc \max_{ijk} p_{ijk}|\mathcal{N}_\epsilon| \sum_t \exp\left\{\frac{-8\psi_t^2}{cr\left|\mathcal{N}_\epsilon\right|^2 \bar{\sigma}_k^2(t)}\right\} + 2\sum_t \psi_t + \sqrt{128r^2c^2T\log(1/\delta)},$$

and $\alpha_{ijk}^{(t)} = \sum_{q=1}^{t-1} d_{ik}^{(q)} v_{jk}^{(q-1)}$, $\bar{\sigma}_k^2(t) := \left[\sum_{j=1}^{F} \frac{1}{\left(\sum_i \alpha_{ijk}^{(t)}+1\right)^2}\right]$, $\epsilon_k := \epsilon/M^2 - (1-\lambda_1^{(k)})\mathcal{I}_1 + \lambda_2^{(k)}\mathcal{I}_2+, \ldots, +\lambda_M^{(k)}\mathcal{I}_M, \mathcal{I}_i := \sup_{(\boldsymbol{u},\boldsymbol{v})\in\mathcal{C}_{c,r}} \left|\boldsymbol{u}^T P_k \boldsymbol{v}\right| \; \forall\; k = 1, 2, \ldots, M$. This completes the proof.

Appendix C

Proofs for Chapter 5

C.1 Derivation of (5.4.1)

Proof: Let us first consider the discounted reward for finite horizon ω-subsidy problem. Let $\nu^1(\pi) = R(\pi)$ and $\nu^0(\pi) = \omega$ represent the immediate reward corresponding to active and idle states respectively. The N-stage finite horizon reward function is given as below:

$$\widetilde{V}_N(\pi[0]) = \max_{a[t],t=0,\dots,N-1} \mathbb{E}\big[\sum_{t=0}^{N} \beta^t \nu^{a[t]}(\pi[t])|\pi[0]\big]$$

Let $\widehat{V}_{\omega,t}(\pi)$ be the reward at time t with belief value $\pi[t] = \pi$. Therefore, $\widetilde{V}_N(\pi[0]) = \widehat{V}_{\omega,0}(\pi[0])$ and the final stage value function $\widehat{V}_{\omega,N-1}(\pi[N-1])$ is given by

$$\begin{aligned}\widehat{V}_{\omega,N-1}(\pi[N-1]) &= \max_{a[N-1]}\{\nu^{a[N-1]}(\pi[N-1])\}\\ &= \max\{\omega, R(\pi[N-1])\}\end{aligned}$$

Thus, $\widehat{V}_{\omega,N-1}(\pi)$ is convex with π, since it is the maximum of a constant and a convex function. Thus, the Bellman equation for any time $0 \le t \le N-1$ can be written as

$$\widehat{V}_{\omega,t}(\pi[t]) = \max\{\widehat{V}^0_{\omega,t}(\pi[t]), \widehat{V}^1_{\omega,t}(\pi[t])\}.$$

where

$$\widehat{V}^0_{\omega,t}(\pi[t]) = \omega + \beta \widehat{V}^0_{\omega,t+1}(Q(\pi)),$$
$$\widehat{V}^1_{\omega,t}(\pi[t]) = R(\pi) + \beta(\pi \widehat{V}_{\omega,t+1}(p) + (1 - \pi \widehat{V}_{\omega,t+1}(r)))$$

Since, $\widetilde{V}_N(\pi) = V_{\omega,0}(\pi)$, $\widetilde{V}_N(\pi)$ is a convex function of π. The infinite horizon reward is the limit of the finite horizon reward for the discounted problem with bounded reward per slot, we have, $V_\omega(\pi) = \lim_{N \leftarrow \infty} V_{\omega,N}(\pi)$. Hence $V_\omega(\pi)$ is a convex function in π. This completes the proof.

C.2 Derivation of (5.4.2)

Proof: From the Bellman equation 5.10, let $V^1_\omega(\pi)$ be the reward corresponding to the cache decision and $V^0_\omega(\pi)$ be the reward corresponding to the idle decision, i.e.,

$$\begin{aligned} V^1_\omega(\pi) &= R(\pi) + \beta(\pi V_\omega(p) + (1-\pi)V_\omega(r)), \\ V^0_\omega(\pi) &= \omega + \beta V_\omega(Q(\pi)) \end{aligned}$$

Case (i): If $\omega > 1$ and since $R(\pi) < 1$, thus it will be always optimal to remain idle. Thus, the threshold can be set to 1.

Case (ii); If $\omega \leq \delta$, then we have,

$$\begin{aligned} V^0_\omega(\pi) &= \omega + \beta V_\omega(Q(\pi)) \\ &= \omega + \beta V_\omega(\pi p + (1-\pi)r) \\ &\leq R(\pi) + \beta(\pi V_\omega(p) + (1-\pi)V_\omega(r)) \\ &= V^1_\omega(\pi), \end{aligned}$$

where the inequality is due to $\delta \leq R(\pi)$ and Jensen's inequality which comes from the convexity of $V_\omega(\pi)$. Hence, the optimal decision is to stay active. Thus, we can set the threshold to κ, for some $\kappa < 0$.

Case (iii): If $\delta \leq \omega \leq 1$, then we have,

$$\begin{aligned} V_\omega^0(\pi) &= \omega + \beta V_\omega(r) > \delta + \beta V_\omega(r) = V_\omega^1(0) \\ V_\omega^0(\pi) &= \omega + \beta V_\omega(p) > 1 + \beta V_\omega(r) = V_\omega^1(1) \end{aligned}$$

Thus, there exists a threshold $\pi^*(\omega)$ such that $a = 1$, whenever $\pi > \pi^*(\omega)$. This completes the proof.

C.3 Derivation of (5.4.3)

Proof: First, the structural properties are established when the user stays idle. It is assumed that the user has initial belief value as $\pi[0]$ and stays idle at all times. Then the belief value at t^{th} slot is given by $\pi[t] = Q_t(\pi_i[0])$, where Q^t is the t^{th} iteration of the function Q, and is given by:

$$Q^t(\pi) = \frac{r - (p-r)^t(r - (1+r-p)\pi)}{1+r-p}$$

Assuming, $\pi[0]$ as the steady state distribution of the two-state channel, i.e.

$$\pi^0 = \frac{r}{1+r-p}$$

It is obvious that $\pi^0 = \lim_{t\to\inf} Q^t(\pi)$. The structural properties of $Q^t(\pi)$ are as follows:

(i) For positively correlated channel (i.e., $p > r$), $\pi[t]$ converges to steady state π^0 monotonically. For negatively correlated channel (i.e., $p \leq r$), $\pi[t]$ converges to steady state π^0 with oscillation and a monotonically converging envelope.

(ii) $\min\{p, r\} \leq Q^t(\pi_i[0]) \leq \max\{p, r\}$ $\forall\ t = 1, 2, \ldots$ and $\pi_i[0] \in [0, 1]$.

Proof (i) Since we have $0 < p - r \leq 1$ for positively correlated channel and $-1 \leq p - r \leq 0$ for negatively correlated channel, it is clear from the expression of (C.3) that $\pi[t]$ converges to steady state π^0 monotonically and approaches steady state π^0 with oscillation and a monotonically converging envelope.

(ii) We need to prove $\min\{p, r\} \leq Q[\pi] \leq \max\{p, r\}$, for all π.

$$Q[\pi] = \frac{r - (p-r)(r-(1+r-p)\pi)}{1+r-p}$$

For positively correlated channel, since $p - r > 0$

$$\begin{aligned} Q[\pi] &\geq \frac{r-(p-r)r}{1+r-p} = r. \\ Q[\pi] &\leq \frac{r-(p-r)(r-(1+r-p))}{1+r-p} = p. \end{aligned}$$

For negatively correlated channel, since $p - r < 0$

$$\begin{aligned} Q[\pi] &\leq \frac{r-(p-r)r}{1+r-p} = r. \\ Q[\pi] &\geq \frac{r-(p-r)(r-(1+r-p))}{1+r-p} = p. \end{aligned}$$

Thus proving the structural properties of $Q^t(\pi_i[0])$. Recall that π^* is defined as the threshold in Proposition 2. Let $L(\pi, \pi^*)$ be the time needed for the belief value of a user to exceed π^* from below, starting from initial value π, i.e.

$$L(\pi, \pi^*) = \min_t \{Q^t(\pi) > \pi^*\}$$

$L(\pi, \pi^*)$ can be calculated as follows:

- Positive correlation $(p > r)$

$$L(\pi, \pi^*) = \begin{cases} 0, & \text{if } \pi > \pi^* \ , \\ \log_{p-r} \frac{r-(1+r-p)\pi^*}{r-(1+r-p)\pi} + 1, & \text{if } \pi \leq \pi^* < \pi, \\ \inf, & \text{if } \pi \leq \pi^* \text{ and } \pi^* \geq \pi_0 \end{cases}$$

- Negative correlation $(p \leq r)$

$$L(\pi, \pi^*) = \begin{cases} 0, & \text{if } \pi > \pi^* \ , \\ 1, & \text{if } \pi \leq \pi^* < \pi \text{ and } Q(\pi) > \pi, \\ \inf, & \text{if } \pi \leq \pi^* \text{ and } Q(\pi) \leq \pi^*. \end{cases}$$

Next, the value functions $V_\omega(p)$ and $V_\omega(r)$ based on the value of $\pi^*(\omega)$ is derived as follows: (1) Positive correlation $(p > r)$

- When $\pi^*(\omega) \geq p$, the belief value p is in the idle set. If $\pi[0] = p$, the system stays idle. Hence the reward function is expressed as

$$V_\omega(p) = \omega + \beta\omega + \beta^2\omega + \ldots = \frac{\omega}{1-\beta}.$$

- When $\pi^*(\omega) < p$, the belief value p is then in the active set. Hence from the Bellman equation, we have

$$V_\omega(p) = R(p) + \beta(pV_\omega(p) + (1-p)V_\omega(r))$$

  where, $R(.)$ is the immediate reward function.

$$V_\omega(p) = \frac{R(p) + \beta(1-p)V_\omega(r)}{1-\beta p}$$

- When $\pi^*(\omega) < r$, the value r is in active set, thus the belief value stays in the active set. Therefore,

$$V_\omega(r) = \sum_{t=0}^{\inf} \beta^t R(Q^t(r)) = \sum_{t=0}^{\inf} \beta R(\frac{r-(p-r)^{t+1}r}{1+r-p}).$$

- When $\pi^*(\omega) \geq \pi^0$, since $\pi^0 \geq r$, the belief value r is in idle set. Hence,

$$V_\omega(r) = \omega + \beta\omega + \beta^2\omega + \ldots = \frac{\omega}{1-\beta}.$$

- When $r < \pi^*(\omega) < \pi^0$, the belief value r is in the idle set and the channel is positively correlated. Therefore,

$$V_\omega(r) = \frac{1-\beta^{L(r,\pi^*(\omega))}}{1-\beta}\omega + \beta^{L(r,\pi^*(\omega))}V_\omega^1(Q^{L(r,\pi^*(\omega))}(r)).$$

  where $V_\omega^1(Q^{L(r,\pi^*(\omega))}(r)) = R(Q^{L(r,\pi^*(\omega))}(r)) + \beta(Q^{L(r,\pi^*(\omega))}(r)V_\omega(p) + (1 - Q^{L(r,\pi^*(\omega))}(r))V_\omega(r))$

This completes the proof.

References

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

[2] Cisco, "Cisco annual internet report (2018-2023)," 2020.

[3] Q. Ye, B. Rong, Y. Chen, M. Al-Shalash, C. Caramanis, and J. G. Andrews, "User association for load balancing in heterogeneous cellular networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 6, pp. 2706–2716, 2013.

[4] K. Poularakis *et al.*, "Exploiting Caching and Multicast for 5G Wireless Networks," *IEEE Transactions on Wireless Communications*, vol. 15, pp. 2995 – 3007, 2016.

[5] Q. Ye, M. Al-Shalash, C. Caramanis, and J. G. Andrews, "On/off macrocells and load balancing in heterogeneous cellular networks," in *2013 IEEE Global Communications Conference (GLOBECOM)*, 2013, pp. 3814–3819.

[6] F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta, and P. Popovski, "Five disruptive technology directions for 5G," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 74–80, 2014.

[7] E. Bastug, M. Bennis, and M. Debbah, "Cache-enabled small cell networks: Modeling and tradeoffs," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 82–89, 2014.

[8] S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao, "Are we ready for SDN? implementation

challenges for software-defined networks," *IEEE Communications Magazine*, vol. 51, no. 7, pp. 36–43, 2013.

[9] 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.

[10] 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.

[11] S. Krishnendu, B. N. Bharath, N. Garg, V. Bhatia, and T. Ratnarajah, "Learning to cache: Federated caching in a cellular network with correlated demands," *IEEE Transactions on Communications*, pp. 1–1, 2021.

[12] 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.

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

[14] J. Song, M. Sheng, T. Q. S. 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.

[15] B. N. Bharath, K. G. Nagananda, D. Gündüz, and H. V. Poor, "Caching with time-varying popularity profiles: A learning-theoretic perspective," *IEEE Transactions on Communications*, vol. 66, no. 9, pp. 3837 – 3847, May 2018.

[16] J. G. Andrews, H. Claussen, M. Dohler, S. Rangan, and M. C. Reed, "Femtocells: Past, present, and future," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 3, pp. 497–508, 2012.

[17]

[18] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[19] E. Baştuğ, M. Bennis, and M. Debbah, "A transfer learning approach for cache-enabled wireless networks," in *2015 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2015, pp. 161–166.

[20] K. Hamidouche, W. Saad, and M. Debbah, "Many-to-many matching games for proactive social-caching in wireless small cell networks," in *2014 12th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2014, pp. 569–574.

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

[22] U. Niesen and M. A. Maddah-Ali, "Coded caching with nonuniform demands," *IEEE Transactions on Information Theory*, vol. 63, no. 2, pp. 1146–1158, 2017.

[23] N. Karamchandani, U. Niesen, M. A. Maddah-Ali, and S. Diggavi, "Hierarchical coded caching," in *2014 IEEE International Symposium on Information Theory*, 2014, pp. 2142–2146.

[24] J. Hachem, N. Karamchandani, and S. Diggavi, "Content caching and delivery over heterogeneous wireless networks," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, 2015, pp. 756–764.

[25] S. Wang, W. Li, X. Tian, and H. Liu, "Fundamental limits of heterogenous cache," *ArXiv*, vol. abs/1504.01123, 2015.

[26] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, "Online coded caching," in *Proc. IEEE International Conference on Communications*, Jun. 2014, pp. 1878–1883.

[27] N. Karamchandani, U. Niesen, M. A. Maddah-Ali, and S. N. Diggavi, "Hierarchical coded caching," *IEEE Transactions on Information Theory*, vol. 62, no. 6, pp. 3212–3229, 2016.

[28] S. P. Shariatpanahi, S. A. Motahari, and B. H. Khalaj, “Multi-server coded caching,” *IEEE Transactions on Information Theory*, vol. 62, no. 12, pp. 7253–7271, 2016.

[29] Z. Chen, “Fundamental limits of caching: Improved bounds for small buffer users,” *CoRR*, vol. abs/1407.1935, 2014.

[30] U. Niesen and M. A. Maddah-Ali, “Coded caching for delay-sensitive content,” in *2015 IEEE International Conference on Communications (ICC)*, 2015, pp. 5559–5564.

[31] A. Sengupta, R. Tandon, and T. C. Clancy, “Fundamental limits of caching with secure delivery,” *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 2, pp. 355–370, 2015.

[32] N. Golrezaei, A. G. Dimakis, and A. F. Molisch, “Wireless device-to-device communication with distributed caching,” in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2012, pp. 2781–2785.

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

[34] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, “Order-optimal rate of caching and coded multicasting with random demands,” *IEEE Transactions on Information Theory*, vol. 63, no. 6, pp. 3923–3949, 2017.

[35] J. Pääkkönen, C. Hollanti, and O. Tirkkonen, “Device-to-device data storage for mobile cellular systems,” in *2013 IEEE Globecom Workshops (GC Wkshps)*, 2013, pp. 671–676.

[36] C. Hollanti, D. Karpuk, A. Barreal, and H.-f. F. Lu, “Space-time storage codes for wireless distributed storage systems,” in *2014 4th International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace and Electronic Systems (VITAE)*, 2014, pp. 1–5.

[37] P. Ostovari, A. Khreishah, and J. Wu, “Cache content placement using triangular network coding,” in *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, 2013, pp. 1375–1380.

[38] W. Huang, S. Wang, L. Ding, F. Yang, and W. Zhang, "The performance analysis of coded cache in wireless fading channel," *CoRR*, vol. abs/1504.01452, 2015.

[39] C.-Y. Wang, S. H. Lim, and M. Gastpar, "Information-theoretic caching: Sequential coding for computing," *IEEE Transactions on Information Theory*, vol. 62, no. 11, pp. 6393–6406, 2016.

[40] A. Liu and V. K. N. Lau, "Cache-enabled opportunistic cooperative MIMO for video streaming in wireless systems," *IEEE Transactions on Signal Processing*, vol. 62, no. 2, pp. 390–402, 2014.

[41] A. Liu and V. Lau, "Cache-induced opportunistic MIMO cooperation: A new paradigm for future wireless content access networks," in *2014 IEEE International Symposium on Information Theory*, 2014, pp. 46–50.

[42] A. Liu and V. K. N. Lau, "Exploiting base station caching in MIMO cellular networks: Opportunistic cooperation for video streaming," *IEEE Transactions on Signal Processing*, vol. 63, no. 1, pp. 57–69, 2015.

[43] X. Peng, J.-C. Shen, J. Zhang, and K. B. Letaief, "Joint data assignment and beamforming for backhaul limited caching networks," in *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*, 2014, pp. 1370–1374.

[44] M. Deghel, E. Baştuğ, M. Assaad, and M. Debbah, "On the benefits of edge caching for MIMO interference alignment," in *2015 IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2015, pp. 655–659.

[45] K. Poularakis and L. Tassiulas, "Exploiting user mobility for wireless content delivery," in *2013 IEEE International Symposium on Information Theory*, 2013, pp. 1017–1021.

[46] V. A. Siris, X. Vasilakos, and G. C. Polyzos, "Efficient proactive caching for supporting seamless mobility," in *Proceeding of IEEE International Sympo-*

sium on a World of Wireless, Mobile and Multimedia Networks 2014, 2014, pp. 1–6.

[47] G. Alfano, M. Garetto, and E. Leonardi, "Content-centric wireless networks with limited buffers: When mobility hurts," *IEEE/ACM Transactions on Networking*, vol. 24, no. 1, pp. 299–311, 2016.

[48] B. Perabathini, E. Baştuğ, M. Kountouris, M. Debbah, and A. Conte, "Caching at the edge: A green perspective for 5G networks," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 2830–2835.

[49] S. Zhou, J. Gong, Z. Zhou, W. Chen, and Z. Niu, "Green delivery: proactive content caching and push with energy-harvesting-based small cells," *IEEE Communications Magazine*, vol. 53, no. 4, pp. 142–149, 2015.

[50] J. Gong, S. Zhou, Z. Zhou, and Z. Niu, "Proactive push with energy harvesting based small cells in heterogeneous networks," in *2015 IEEE International Conference on Communications (ICC)*, 2015, pp. 25–30.

[51] K. Poularakis, G. Iosifidis, and L. Tassiulas, "Joint caching and base station activation for green heterogeneous cellular networks," in *2015 IEEE International Conference on Communications (ICC)*, 2015, pp. 3364–3369.

[52] E. Baştuğ, M. Bennis, M. Kountouris, and M. Debbah, "Cache-enabled small cell networks: modeling and tradeoffs," *EURASIP J. Wireless Commun. Net.*, vol. 2015:41, Feb. 2015.

[53] J. Elias and B. Błaszczyszyn, "Optimal geographic caching in cellular networks with linear content coding," in *2017 15th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2017, pp. 1–6.

[54] A. Altieri, P. Piantanida, L. R. Vega, and C. G. Galarza, "On fundamental trade-offs of device-to-device communications in large wireless networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 9, pp. 4958–4971, 2015.

[55] E. Altman, K. Avrachenkov, and J. Goseling, "Coding for caches in the plane," *CoRR*, 2013.

[56] M. Mitici, J. Goseling, M. de Graaf, and R. J. Boucherie, "Deployment versus data retrieval costs for caches in the plane," *IEEE Wireless Communications Letters*, vol. 3, no. 4, pp. 385–388, 2014.

[57] P. Blasco and D. Gündüz, "Learning-based optimization of cache content in a small cell base station," in *2014 IEEE International Conference on Communications (ICC)*, 2014, pp. 1897–1903.

[58] A. Sadeghi, G. Wang, and G. B. Giannakis, "Deep reinforcement learning for adaptive caching in hierarchical content delivery networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 1024–1033, 2019.

[59] 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.

[60] 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.

[61] 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.

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

[63] 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.

[64] 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.

[65] 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.

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

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

[68] 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.

[69] 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.

[70] 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.

[71] 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.

[72] 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.

[73] 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.

[74] 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.

[75] 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.

[76] 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.

[77] 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.

[78] 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, Dec 2019.

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

[80] P. Ansell, K. Glazebrook, J. Niño-Mora, and M. O'Keeffe, "Whittle's index policy for a multi-class queueing system with convex holding costs," *Mathematical Methods of Operations Research*, vol. 57, no. 1, p. 21–39, 2003.

[81] M.-C. Lee, A. F. Molisch, N. Sastry, and A. Raman, "Individual preference probability modeling for video content in wireless caching networks," *IEEE Global Communications Conference*, pp. 1–7, 2017.

[82] G. Paschos, E. Baştuğ, I. Land, G. Caire, and M. Debbah, "Wireless Caching: Technical Misconceptions and Business Barriers," *IEEE Commun. Mag.*, vol. 54, pp. 16–22, Aug 2016.

[83] G. Ausiello, M. Protasi, A. Marchetti-Spaccamela, G. Gambosi, P. Crescenzi, and V. Kann, *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*, 1st ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1999.

[84] R. A. Horn and C. R. Johnson, *Matrix analysis.* Cambridge University Press, 1990.

[85] P. Raghavan and C. D. Tompson, "Randomized rounding: A technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, no. 4, pp. 365–374, Dec 1987. [Online]. Available: https://doi.org/10.1007/BF02579324

[86] B. Lehmann, J. Lehmann, and N. Nisan, "Combinatorial auctions with decreasing marginal utilities," *ACM Conference on El. Commerce*, pp. 18–28, 2001.

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

[88] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms.* USA: Society for Industrial and Applied Mathematics, 2007, p. 1027–1035.

[89] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

[90] K. Shanmugam, N. Golrezaei, A. Dimakis, A. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Trans. Inf. Theory*, vol. 51, no. 4, pp. 142–149, Apr 2013.

[91] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," *Proc. IEEE 18th Annu. Joint Conf. Comput. Commun. Soc.*, vol. 1, pp. 126–134, 1999.

[92] 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.

[93] 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.

[94] 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.

[95] 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.

[96] 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.

[97] 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.

[98] 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.

[99] 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.

[100] 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

[101] 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.

[102] 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.

[103] 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.

[104] 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.

[105] 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.

[106] 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.

[107] 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.

[108] 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.

[109] 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.

[110] 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.

[111] 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.

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

[113] 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.

[114] 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.

[115] 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.

[116] 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.

[117] 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.

[118] 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.

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

[120] 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

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

[122] 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.

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

[124] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Transactions on Management Information Systems*, vol. 6, no. 4, pp. 1–19, 2016.

[125] Z. Zhou, S. Khemmarat, and G. L., "The impact of youtube recommendation system on video views," in *Proceedings of the 10th ACM Conference on Internet Measurement Conference*, 2010, pp. 404–410.

[126] H. Zhu, Y. Cao, X. Wei, W. Wang, T. Jiang, and S. Jin, "Caching transient data for internet of things: A deep reinforcement learning approach," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2074–2083, 2019.

[127] R. Devooght and H. Bersini, "Collaborative filtering with recurrent neural networks," *CoRR*, vol. abs/1608.07400, 2016. [Online]. Available: http://arxiv.org/abs/1608.07400

[128] W. Liu, J. Zhang, Z. Liang, L. Peng, and J. Cai, "Content popularity prediction and caching for ICN: A deep learning approach with SDN," *IEEE Access*, vol. 6, pp. 5075–5089, 2018.

[129] Z. Chang, L. Lei, Z. Zhou, S. Mao, and T. Ristaniemi, "Learn to cache: Machine learning for network edge caching in the big data era," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 28–35, 2018.

[130] S. Müller, O. Atan, M. van der Schaar, and A. Klein, "Context aware proactive content caching with service differentiation in wireless networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 2, pp. 1024–1036, 2017.

[131] K. Thar, N. H. Tran, T. Z. Oo, and C. S. Hong, "DeepMEC: Mobile edge caching using deep learning," *IEEE Access*, vol. 6, pp. 78 260–78 275, 2018.

[132] S. Li, J. Xu, M. van der Schaar, and W. Li, "Trend-aware video caching through online learning," *IEEE Transactions on Multimedia*, vol. 18, no. 12, pp. 2503–2516, 2016.

[133] Y. Chen, Y. Liu, J. Zhao, and Q. Zhu, "Mobile edge cache strategy based on neural collaborative filtering," *IEEE Access*, vol. 8, pp. 18 475–18 482, 2020.

[134] Y. Jiang, M. Ma, M. Bennis, F. Zheng, and X. You, "User preference learning-based edge caching for fog radio access network," *IEEE Transactions on Communications*, vol. 67, no. 2, pp. 1268–1283, 2019.

[135] S. A. Bitaghsir, A. Dadlani, M. Borhani, and A. Khonsari, "Multi-armed bandit learning for cache content placement in vehicular social networks," *IEEE Communications Letters*, vol. 23, no. 12, pp. 2321–2324, 2019.

[136] 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.

[137] C. Robert, "The bayesian choice: from decision-theoretic foundations to computational implementation," *Springer Science and Business Media*, 2007.

[138] J. Zhu, N. Chen, and E. P. Xing, "Bayesian inference with posterior regularization and applications to infinite latent SVMs," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1799–1847, 2014.

[139] T. Shi and J. Zhu, "Online bayesian passive-aggressive learnings," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 1084–1122, 2014.

[140] A. Ghosh, N. Mangalvedhe, R. Ratasuk, B. Mondal, M. Cudak, E. Visotsky, T. A. Thomas, J. G. Andrews, P. Xia, H. S. Jo, H. S. Dhillon, and T. D.

Novlan, “Heterogeneous cellular networks: From theory to practice,” *IEEE Communications Magazine*, vol. 50, no. 6, pp. 54–64, 2012.

[141] S. Van de Geer, *Empirical Processes in M-Estimation.* Cambridge: Cambridge Univ. Press, U.K., 2000.

[142] J. C. Gittins, “Bandit processes and dynamic allocation indices,” *Journal of the Royal Statistical Society. Series B (Methodological),*, pp. 148– 177, 1979. [Online]. Available: https://doi.org/10.1007/BF02579324

[143] C. Meng, G. Wang, X. Dai, S. Chen, and W. Ni, “An energy-efficient transmission strategy for cache-enabled wireless networks with non-negligible circuit power,” *IEEE Access*, vol. 7, pp. 74 811–74 821, 2019.

[144] O. Bousquet, S. Boucheron, and G. Lugosi, “Introduction to statistical learning theory,” in *Summer School on Machine Learning.* Springer, 2003, pp. 169–207.

[145] O. Marchal and J. Arbel, “On the sub-gaussianity of the beta and dirichlet distributions,” *Electronic Communications in Probability*, vol. 22, Jan 2017.

List of Publications

Journal Papers:

1. 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, Dec. 2019.

2. S. Krishnendu, B. N. Bharath, N. Garg, V. Bhatia and T. Ratnarajah, "Learning to Cache: Federated Caching in a Cellular Network With Correlated Demands," *IEEE Transactions on Communications,* 2021.

3. S. Krishnendu, B. N. Bharath, V. Bhatia, J. Nebhen, M. Dobrovolny and T. Ratnarajah, "Wireless Edge Caching and Content Popularity Prediction Using Machine Learning," *IEEE Consumer Electronics Magazine,* 2022.

4. S. Krishnendu, B. N. Bharath and V. Bhatia "Online Learning of Caching and Recommendation Policies in a Multi-BS Cellular Networks," *2022. [Online]. Available: https://arxiv.org/abs/2210.07747,* under review.

Conference Papers:

1. S. Krishnendu, B. N. Bharath and V. Bhatia, "Joint Edge Content Cache Placement and Recommendation: Bayesian Approach," *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring),* 2021, pp. 1-5.

2. Sourab Mishra, Sparsh Bansal, S. Krishnendu, V. Bhatia, "KNN based Novel Recommendation System for Mobile Edge Caching," *IEEE INDICON Conference,* Dec. 2021.

3. S. Krishnendu, B. N. Bharath and V. Bhatia, "Recommendation based Edge Caching for Cellular Network Using Whittle Index," *under review.*