Dynamic Content Updates in Heterogeneous Wireless Networks

Mehdi Salehi Heydar Abad [1], Emre Ozfatura[2], Ozgur Ercetin[1], and Deniz Gündüz[2]

[1]Faculty of Engineering and Natural Sciences, Sabanci University
[2]Department of Electrical and Electronic Engineering Imperial College London
[1]{*mehdis,oercetin*}@*sabanciuniv.edu*
[2]{*m.ozfatura,d.gunduz*}@*imperial.ac.uk*

***Abstract*—Content storage at the network edge is a promising solution to mitigate the excessive traffic load due to on-demand streaming applications as well as to reduce the streaming delay. To this end, cache-enabled cellular architectures can be utilized to increase the provided quality-of-service (QoS) and to reduce the network cost. However, there are certain issues to be considered in the design of the content storage strategy such that the contents should be refreshed in order to responds user‘s expectations. Using a frequent cache refreshment strategy the ratio of satisfied users can be increased at an increasing network cost. In this paper, we introduce a cache refreshment strategy via leveraging learning techniques so that users' tolerance to the age of content is learned and the content is refreshed accordingly.**

***Index Terms*—Content caching, content refreshment, Markov decision process (MDP), quality of service, multi-armed bandit (MAB)**

I. Introduction

While proactive content caching has received significant interest in the recent years, most of the existing strategies in the literature (both with uncoded [1, 2, 3] and coded placement [4, 5]) have been designed under the assumption that content popularities are known in advance. Although it is possible to observe the global popularity of contents in on-demand video streaming services, such as YouTube [6], small-cell base stations (SBSs) usually serve a small geographical area, where the local content popularity might not be aligned with the global popularity [7]. This mismatch between the local and global content popularities requires the design of predictive caching policies that aim to learn the local content popularity from the user requests. Predictive caching policies can be classified into two main groups, namely *predictive caching with unknown popularities* [8, 9, 10] and *predictive caching with time-varying popularities* [11, 12, 13]. In [8, 9], the authors focus on a single SBS and model the predictive caching problem as a multi-arm bandit problem, in which the received user requests are utilized to predict the file popularities, and the optimal caching strategy is obtained by taking into account the cost of file replacements. In [10], this approach has been extended to a cooperative caching framework, where the SBSs fetch the requested content from the neighboring SBSs if the corresponding file is cached there. Another strategy for predictive caching follows a user-centric approach, possibly implemented at a higher layer, and records user requests for different contents as a matrix. Future user requests are predicted using matrix completion techniques by exploiting the correlations among the requests for different files, similarly to recommendation systems [14]. In [11], a cache replacement strategy has been introduced for time-varying popularity scenario to maximize the local service rate with a minimum replacement cost, while a more theoretical approach is taken in [13], which studies the cache update policy in the case of time-varying content popularities. In parallel to the aforementioned works, another relevant research direction is *contextual predictive caching*, where various features of the requests, such as genre of the video file, age of the user, or the time of the requests, are utilized to predict future requests and shape the caching strategy accordingly [15, 16]. Although the predictive caching framework is highly effective to increase the efficiency of edge caching, there are certain limitations. Most of the aforementioned predictive caching strategies are designed to predict only the content popularity. However, in many applications, e.g., news, weather, etc., freshness of the content is another important factor for the user satisfaction. Content caching and refreshment problem has been previously studied in [17, 18]. In this paper, we consider a heterogeneous cellular network with cache enabled SBSs and provide a cost aware content update policy. The proposed content update strategy consists of two parts; in the first part, we show that the structure of the optimal periodic content update policy that minimizes the network cost for given users' tolerance to the age of the content is of threshold type, and in the second part, we utilize the multi-armed bandit approach to learn users' tolerance to the age of the content. To the best of our knowledge, ours is the first work that utilizes the learning framework to analyze the user behavior based on the content age. Accordingly, we design an online cache refreshment policy to minimize the overall network cost.

II. System model and problem formulation

We consider a cellular network with a macro base station (MBS) and a SBS serving the users in a cell. It is assumed that both the MBS and the SBS are equipped with cache memories, storing a library of N distinct dynamic contents, denoted by

This work was in part supported by EC H2020-MSCA-RISE-2015 programme under grant number 690893.

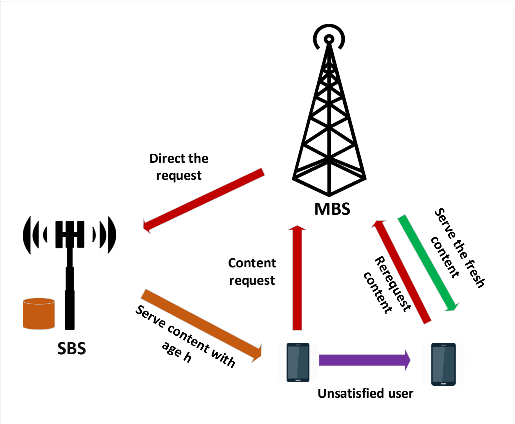

Figure 1: User requests are first off-loaded to the SBS; however, users may become unsatisfied with the freshness of the contents they receive from the SBS, and consequently, such requests are re-directed to the MBS.

$S_1,\dots,S_N$. We note that each dynamic content has a different popularity, i.e., the probability that a user requests content S_n is p_n, for $n=1\dots,N$.

A. Content freshness

Dynamic contents, such as news videos, traffic and weather updates, may change frequently over time. While we assume that the MBS always has the fresh content updates, thanks to its relatively higher-bandwidth connection to the content server in the core network, the SBS needs to regularly refresh the dynamic contents in its cache to keep them up-to-date. For the SBS, downloading all the contents from the MBS through its limited backhaul link is costly in terms of energy, time and spectrum.

We consider a discrete time system model with equal-length time slots. At the beginning of each time slot the SBS decides on which contents to be updated. We assume that a content is updated at the beginning of the next time slot. The corresponding decision vector is denoted by $\mathbf{d}(t)=(d_1,\dots,d_N)$, i.e., when the application S_n is updated at the end of time slot t, $d_n(t)=1$, and $d_n(t)=0$ otherwise.

We denote by $h_n(t)$ the *age* of the content S_n in the SBS cache at time slot t. We assume a maximum age T_{max} at which a content becomes obsolete. In other words, the age of a content increases until it becomes obsolete. Accordingly, age of content S_n, $n=1,\dots,N$, evolves over time in the following way:

$$h_n(t+1)=\max\{(1-d_n(t))(h_n(t)+1),T_{max}\}. \tag{1}$$

We denote the length-N vector of ages associated with all the dynamic contents in the library by $\mathbf{h}(t)$.

B. User behavior

Let $\lambda(t)$ denote the number of users that request a content at time slot t. Whenever a user requests a content, the request is first off-loaded to the SBS to be served. Users have different tolerance levels to the age of the contents they receive. Hence, we consider that, with probability $P^{(n)}_{redirect}(h)$, a user is not satisfied with the age h of content S_n, and thus, it places another request for content S_n. In that case, the new user request is served directly by the MBS with a fresh content. Let $\lambda_n(t)$ be the number of users that request content n, $n=1,\dots,N$, in time slot t, which is governed by the popularity profile p_n. The number of users that request content n is split into two disjoint sets, where the first set of users are redirected to the MBS, while the second set consists of the users satisfied with the service provided by the SBS. We denote these numbers by $\lambda_{rn}(t)$, $\lambda_{an}(t)$, respectively. Note that $\lambda_{rn}(t)$ and $\lambda_{an}(t)$ are governed by the random process $P^{(n)}_{redirect}(h_n(t))$.

Let $\boldsymbol{\lambda}_r(t)=(\lambda_{r1}(t),\dots,\lambda_{rN}(t))$ be the vector associated with the number of redirected users for each content. We have the following:

$$\sum_{n=1}^{N}\lambda_n(t)=\lambda(t), \tag{2}$$

$$\text{where,}\qquad \lambda_{nr}(t)+\lambda_{na}(t)=\lambda_n(t). \tag{3}$$

The corresponding expected values of these parameters are given as:

$$\mathbb{E}[\lambda_n|\lambda]=\lambda p_n, \tag{4}$$

$$\mathbb{E}[\lambda_{nr}|\lambda,h_n]=\lambda p_n P^{(n)}_r(h_n). \tag{5}$$

C. Decision model and the problem formulation

Let $C(\boldsymbol{\lambda}_r(t),\mathbf{d}(t))$ be the cost associated with serving the users redirected to the MBS at time t, and the backhaul cost associated with updating the contents, if there is any. In this work, we assume that this cost is linear in λ_{rn}[1]. Hence,

$$C(\boldsymbol{\lambda}_r(t),\mathbf{d}(t))=\sum_{n=1}^{N}C_n(\lambda_{rn}(t),d_n(t)). \tag{6}$$

Similarly, we define the back-haul cost function $C_{BH}(\mathbf{d}(t))$ which is also a linear function.

$$C_{BH}(\mathbf{d}(t))=\sum_{n=1}^{N}C^{(n)}_{BH}(d_n(t)), \tag{7}$$

where $C^{(n)}_{BH}(d_n(t))=d_n(t)\mathscr{E}_n$, with $\mathscr{E}_n$ being the average back-haul cost of updating application n. If the SBS decides to update the content n (or multiple contents), the age of the content is updated at the end of the time slot. Hence,

$$C_n(\lambda_{rn}(t),d_n(t))=\beta_n+\alpha_n\lambda_{rn}(t)+d_n(t)\mathscr{E}_n \tag{8}$$

Note that updating a content has an immediate cost which is larger than not-updating. However, the incurred extra cost in updating the content enables more users to be served at a local SBS.

We aim at minimizing the average total cost as follows:

$$\min_{\mathbf{d}(t)\in\{0,1\}^N}\ \lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left[C(\boldsymbol{\lambda}_r(t),\mathbf{d}(t))\right] \tag{9}$$

[1] For example, in OFDMA, a user re-directed to the MBS is assigned a subcarrier, and the power allocated to that subcarrier adds linearly to the energy cost.

III. MDP formulation

Define the state of the system to be $\mathbf{h}$. We denote by $V(\mathbf{h})$ the differential value function at state $\mathbf{h}$. The differential Bellman equations can be written as:

$$V(\mathbf{h}) + \mu^* = \min_{\mathbf{d}} V_{\mathbf{d}}(\mathbf{h}), \tag{10}$$

where μ^* is the optimal average cost and $V_{\mathbf{d}}(\mathbf{h})$ is the differential action-value function defined by:

$$V_{\mathbf{d}}(\mathbf{h}) = \bar{C}(\mathbf{h},\mathbf{d}) + \mathbb{P}(\acute{\mathbf{h}}|\mathbf{h},\mathbf{d})V(\acute{\mathbf{h}}), \tag{11}$$

where $\mathbb{P}(\acute{\mathbf{h}}|\mathbf{h},\mathbf{d})$ is the transition probability from state $\mathbf{h}$ into $\acute{\mathbf{h}}$ when action $\mathbf{d}$ is taken which is governed by (1), and

$$\bar{C}(\mathbf{h},\mathbf{d}) = \sum_{\mathbf{\Lambda}} \mathbb{P}(\boldsymbol{\lambda_r} = \mathbf{\Lambda}) C(\boldsymbol{\lambda}_r, \mathbf{d}) \tag{12}$$

The MDP associated with the average cost minimization problem can be solved by well known value iteration algorithm. However, the cardinality of state space (i.e., $(T_{max})^N$) and action space (i.e., 2^N) grow exponentially with the number of contents. Hence, the curse of dimensionality is the bottleneck for an efficient solution. To bypass this bottleneck, we note that the cost function in (6) is linear and the transition probabilities of each content does not affect the other. Hence, we can separate the value function in (10) into N independent value functions each representing a distinct application. For each application n, we have

$$V^{(n)}(h_n) + \mu_n^* = \min_{d_n} V_d^{(n)}(h_n) \tag{13}$$

$$V_d^{(n)}(h_n) = \bar{C}_n(h_n, d_n) + d_n V^{(n)}(0) + (1-d_n)V^{(n)}(h_n+1) \tag{14}$$

We have developed a framework that has enabled distributed policies with respect to the individual contents. We will show that for each content, there exists a threshold policy on the age of the content for which it is optimal to update the content. The following lemma establishes the key property used to prove the structure of the optimal policy.

Lemma 1. *The differential value function $V^{(n)}(h_n)$ for all $n = 1, \dots, N$ is non-decreasing with respect to the age of the content, h_n.*

Proof. We use the value iteration algorithm to prove the lemma. We start by an arbitrary $V_0^{(n)}(h)$ differential value function and obtain the k-step differential value function $V_k^{(n)}(h)$ as follows:

$$\begin{aligned} &V_{k+1}^{(n)}(h) \\ &= \min_d(-\mu_n^* + \bar{C}_n(h_n, d_n) + d_n V_k^{(n)}(0) + (1-d_n)V_k^{(n)}(h+1)) \end{aligned} \tag{15}$$

Note that $\lim_{k\to\infty} V_k^{(n)}(h) = V^{(n)}(h)$. The proof is by induction. For $k=1$, $V_1^{(n)}(h) = \min_{d_n}(-\mu_n^* + \bar{C}_n(h_n, d_n) + d_n V_0^{(n)}(0) + (1-d_n)V_0^{(n)}(h_n+1))$ which is the minimum of two non-decreasing functions, and thus, itself is a non-decreasing function in h_n. Assume that the lemma holds for k. Then according to (15), $V_{k+1}^{(n)}(h_n)$ is also a non-decreasing function with respect to h_n. By letting $k \to \infty$, we conclude the proof by showing that $V^{(n)}(h_n)$ is also a non-decreasing function in h_n. □

The lemma is intuitively clear considering the non-decreasing property of the cost functions and Bellman equations in (14).

Theorem 1. *For each content the optimal policy minimizing the average cost is a threshold policy.*

Proof. The monotonicity of the differential value functions prove the optimality of the threshold policy [19, Chapter 7]. Intuitively, due to the non-decreasing property of the $V^{(n)}(h_n)$, at some age, it would be optimal to update the content. Since the differential value function is non-decreasing, a larger, or smaller age would not be able to yield a smaller average cost. □

IV. Learning Content Popularity and Age Tolerance

In the previous section, we showed that the problem is separable and thus, the optimization can be performed for each content separately. Second, we proved that the policy minimizing the cost is a threshold policy. Hence, the SBS by monitoring the age of the contents individually, needs to optimize according to a single threshold for each content. Under the threshold policy, the age of a content increases linearly until it reaches the threshold wherein the content will be updated and the age will refresh to a value of zero. Thus the minimum cost associated with content n is the solution of:

$$\mu_n^* = \min_{H_n} \frac{1}{H_n+1}\left(\sum_{h=0}^{H_n} \bar{C}_n(h,0) + \mathscr{E}_n\right), \tag{16}$$

where

$$\bar{C}_n(h,0) = \beta_n + \alpha_n \lambda p_n P_{redirect}^{(n)}(h). \tag{17}$$

Considering the linearity of the cost functions, the average cost optimization becomes:

$$\min_{H_n}\left\{\beta_n + \frac{\mathscr{E}_n + \alpha_n p_n \sum_{h=0}^{H_n} P_{redirect}^{(n)}(h)}{H_n+1}\right\}. \tag{18}$$

The equivalent optimization problem depends on the redirection probabilities $P_{redirect}^{(n)}(h)$, that are unknown. Hence, in the following we resort to reinforcement learning methods to infer the redirection probabilities.

We consider a sequential learning framework in which the SBS at each iteration of the learning algorithm faces choosing a threshold $0 \le H_n \le T_{max}$. After choosing the threshold, the SBS will observe a random cost associated with its decision;

$$\hat{C}_n(H_n) = \frac{\mathscr{E}_n + \sum_{t=1}^{H_n} C_n(\lambda_{rn}(t), 0)}{H_n+1} \tag{19}$$

The learning algorithm should provide the SBS a method to adjust its strategy by observing the outcomes of its decision. This resembles the well-known multi-armed bandit (MAB) problem. In MAB, each action (i.e., thresholds) has an expected return value which is called *value* of that action. We denote the true value of action H_n by $q_n(H_n) = \frac{1}{H_n+1}\left(\sum_{h_n=0}^{H} \bar{C}_n(h,0) + \mathscr{E}_n\right)$. If the agent (i.e., SBS) knows $q(.)$ values, then it can simply choose the action with the minimum expected cost. Thus, we need an algorithm that can learn $q_n(.)$ values. A well-studied algorithm for learning those values is the ε-greedy algorithm [20] which starts by an arbitrary estimate, $Q_n(.)$ about the value of the actions and interacts with the environment to update its initial estimates, eventually converging to the true estimates. Two critical phases associated with ε-greedy algorithm is the exploitation and exploration stages. The agent utilizing the estimates greedily chooses an action, and thus, it exploits what it knows already. Meanwhile, if it chooses an action completely random regardless of the estimates we say that it explores. Exploitation is necessary to act upon the experience while exploration helps to improve the estimate values and it facilitates convergence to the true action values. The ε-greedy algorithm is presented in Algorithm 1.

Algorithm 1 ε-greedy

```
for $i = 1, 2, \dots$ do
    $H_n \leftarrow \begin{cases} \arg\max_{H_n} Q_n(H_n) & \text{with probability } 1-\varepsilon, \\ \text{random action} & \text{with probability } \varepsilon. \end{cases}$
    Apply $H_n$ and observe $\hat{C}_n(H_n)$
    $Q_n(H_n) \leftarrow (1-\zeta)Q_n(H_n) + \zeta\hat{C}_n(H_n)$
```

V. Numerical Results

In this section, we aim at evaluating the performance of the ε-greedy algorithm in finding the optimal thresholds that minimize the total cost of the system. Due to the separability, we consider only one content and we note that the learning processes for all the contents are the same. The popularity of the contents are modeled by a Zipf distribution with an exponent of 1.1. We assume that on average a given user becomes dissatisfied with a content of age h with probability of $e^{-0.4h}$. Users arrive at the system according to a Poisson distribution with rate 100 users per time slot. The cost of re-directed users to MBS is $C_n(\lambda_{rn}(t)) = 10 \cdot \lambda_{rn}(t)$ and the backhaul cost is assumed to be 500. In Figure 2, we illustrate the performance of the ε-greedy algorithm for $\varepsilon = 0,\ 0.05,\ 0.1$ by adopting average regret as the metric. The regret of a learning algorithm is defined to be the difference between the cost achieved by the learner and the optimal cost. Here, we obtain the optimal cost by assuming that $P_{redirect}(h)$ is known, and by numerically solving (16). Note that the estimates of the action values, $Q(H)$, is initialized to be 0 for all $H = 0, \dots, T_{max}$. Note also that, the greedy algorithm seems to achieve a better performance even if it always exploits. This is not too surprising considering that the action-values are initialized opportunistically (i.e., the initial costs are believed to be zero by the agent). At the beginning, i.e., $t = 0$, the greedy algorithm believes that every action returns a value of zero. However, by trying each action it gets disappointed in that action and tries the rest. In other words the estimates are biased. Opportunistic initialization is a simple method to incentivize exploration. However, it can only happen once and at the beginning of time. This method quickly fails in non-stationary environments.

To show this, we also study the performance of the ε-greedy algorithm in a time varying environment. We assume that at $t = 300$ the backhaul cost decreases to a value of 400. The results are depicted in Figure 3. We can see that the greedy algorithm cannot adapt to the non-stationary environment and it gets stuck in a sub-optimal threshold. Meanwhile, for 0.05 and 0.1-greedy algorithm, it is able to adapt to the environment thanks to their exploration strategy. A large value of ε results in more exploration, and thus, we can see that 0.1-greedy algorithm has a faster decay in terms of the average regret compared to the 0.05-greedy algorithm. However, note that ε-greedy algorithm is expected to be at least ε away from the optimal cost. Thus, there is a trade-off between the rate of convergence and the value of convergence.

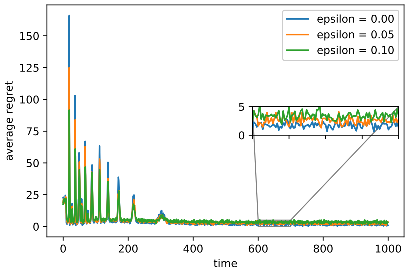

Figure 2: Average regret of the ε-greedy algorithm for $\varepsilon = 0, 0.05, 0.1$ in a stationary environment.

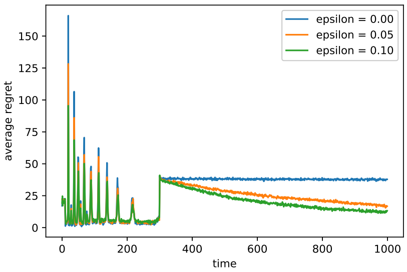

Figure 3: Average regret of the ε-greedy algorithm for $\varepsilon = 0, 0.05, 0.1$ in a non-stationary environment.

VI. Conclusion

In this work, we developed a framework for a cost minimization problem in a dynamic content caching setting. Specifically, we aimed at striking a balance between the number of unsatisfied users whom are redirected to MBS and the cost of accessing the backhaul link by SBS in updating the dynamic contents. We formulated the cost minimization problem as an MDP and showed that the problem is separable with respect to the contests. Subsequently, we proved that a threshold policy in the age of the contents is optimal. In finding the optimal thresholds, we resorted to learning algorithms since users' preferences are not known and vary with each content. To that extend, we represented the problem in MAB framework and through numerical results, we showed that it is possible to make the expected regret of the learning algorithm arbitrarily close to zero. The learning algorithm even shows adaptability to non-stationary settings. As a future work, we aim to investigate the system with non-linear cost functions. Under the non-linear cost functions, the problem is no longer separable and new solution methods needs to be investigated. As an extension of this work we will also analyze the heterogeneous cellular network architecture with energy harvesting SBSs.

References

[1] K. Poularakis, G. Iosifidis, and L. Tassiulas, "Approximation Algorithms for Mobile Data Caching in Small Cell Networks," *IEEE Transactions on Communications*, vol. 62, no. 10, pp. 3665–3677, Oct 2014.

[2] W. Jiang, G. Feng, and S. Qin, "Optimal Cooperative Content Caching and Delivery Policy for Heterogeneous Cellular Networks," *IEEE Transactions on Mobile Computing*, vol. 16, no. 5, pp. 1382–1393, May 2017.

[3] M. Dehghan, B. Jiang, A. Seetharam, T. He, T. Salonidis, J. Kurose, D. Towsley, and R. Sitaraman, "On the complexity of optimal request routing and content caching in heterogeneous cache networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1635–1648, June 2017.

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

[5] E. Ozfatura and D. Gündüz, "Mobility and popularity-aware coded small-cell caching," *IEEE Communications Letters*, vol. 22, no. 2, pp. 288–291, Feb 2018.

[6] X. Cheng, C. Dale, and J. Liu, "Statistics and social network of youtube videos," in *2008 16th Interntional Workshop on Quality of Service*, June 2008, pp. 229–238.

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

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

[9] P. Blasco and D. Gündüz, "Multi-armed bandit optimization of cache content in wireless infostation networks," in *2014 IEEE International Symposium on Information Theory*, June 2014, pp. 51–55.

[10] 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, Oct 2017.

[11] A. Sadeghi, F. Sheikholeslami, and G. B. Giannakis, "Optimal and scalable caching for 5g using reinforcement learning of space-time popularities," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 180–190, Feb 2018.

[12] N. Zhang, K. Zheng, and M. Tao, "Using grouped linear prediction and accelerated reinforcement learning for online content caching," in *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, May 2018, pp. 1–6.

[13] 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, Sept 2018.

[14] Y. Wang, Y. Chen, H. Dai, Y. Huang, and L. Yang, "A learning-based approach for proactive caching in wireless communication networks," in *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*, Oct 2017, pp. 1–6.

[15] S. Müller, O. Atan, M. van der Schaar, and A. Klein, "Context-aware proactive content caching with service differentiation in wireless networks," vol. 16, no. 2, Feb 2017, pp. 1024–1036.

[16] P. Yang, N. Zhang, S. Zhang, L. Yu, J. Zhang, and X. S. Shen, "Content popularity prediction towards location-aware mobile edge caching," *IEEE Transactions on Multimedia*, pp. 1–1, 2018.

[17] C. Kam, S. Kompella, G. D. Nguyen, J. E. Wieselthier, and A. Ephremides, "Information freshness and popularity in mobile caching," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 136–140.

[18] R. D. Yates, P. Ciblat, A. Yener, and M. Wigger, "Age-optimal constrained cache updating," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 141–145.

[19] B. D.P., *Dynamic programming and optimal control*. Athena Scientific, 1995, vol. Volume 1.

[20] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998. [Online]. Available: http://www.cs.ualberta.ca/%7Esutton/book/ebook/the-book.html