Reinforcement Learning for Dynamic Dimensioning of Cloud Caches: A Restless Bandit Approach

Guojun Xiong, *Graduate Student Member, IEEE*, Shufan Wang, Gang Yan, and Jian Li, *Member, IEEE*

***Abstract*— We study the dynamic cache dimensioning problem, where the objective is to decide how much storage to place in the cache to minimize the total costs with respect to the storage and content delivery latency. We formulate this problem as a Markov decision process, which turns out to be a restless multi-armed bandit problem and is provably hard to solve. For given dimensioning decisions, it is possible to develop solutions based on the celebrated Whittle index policy. However, Whittle index policy has not been studied for dynamic cache dimensioning, mainly because cache dimensioning needs to be repeatedly solved and jointly optimized with content caching. To overcome this difficulty, we propose a low-complexity fluid Whittle index policy, which jointly determines dimensioning and content caching. We show that this policy is asymptotically optimal. We further develop a lightweight reinforcement learning augmented algorithm dubbed fW-UCB when the content request and delivery rates are unavailable. fW-UCB is shown to achieve a sub-linear regret as it fully exploits the structure of the near-optimal fluid Whittle index policy and hence can be easily implemented. Extensive simulations using real traces support our theoretical results.**

***Index Terms*— Cloud cache dimensioning, index policy, restless bandits, reinforcement learning.**

I. Introduction

CONTENT delivery networks (CDNs) carry more than 50% of the Internet traffic today, and this number is predicted to increase over the coming years [2]. However, the entry cost of traditional CDNs is high, especially for small content providers (CPs) since the lease is on a long-term basis with fixed prices. This motivated the popular "cloud CDNs", which provide managed platforms with a pay-as-you-go model for CPs. For example, Amazon AWS [3] provides both *cache dimensioning* and caching implementation software services to CPs. A CP can lease storage from AWS and implement caching policy using open software such as Amazon CloudFront [4], which allows dynamic storage scaling. Caching dimensioning in cloud CDNs provides large economic yields to CPs since CPs can shut down storage in clouds when the traffic is low.

At the same time, this new model brings significant challenges. First, the CPs not only have to decide how much storage to lease to meet user content requests, but also what content to store in cache. This challenge is further exacerbated in cloud CDNs, where the storage is leased on-the-fly using cloud resources, and dimensioning needs to be solved regularly, often on a per-day or per-hour basis, to accommodate the time-varying nature of content requests. Finally, leasing more storage improves caching performance, e.g., reduced user content service latency, but also incurs additional expenditure. With a given operational expenditure budget, all cache dimensioning decisions are strongly coupled over time. Indeed, finding the optimal tradeoff between maximizing caching performance and minimizing cache dimensioning costs naturally leads to a new set of algorithmic challenges as the caching performance also depends on the variability of content popularity.

In this paper, we are interested in jointly optimizing cache dimensioning and content caching when a CP regularly leases storage from cloud CDNs. We note that there is a natural timescale separation between cache dimensioning and content caching, where the former is a much slower operation than the latter. Using this observation, we formulate the problem of dynamic dimensioning of cloud caches as a two-timescale Markov decision process (MDP) [5] in Section III, where the goal is to propose provably optimal algorithms to minimize the total expected costs due to cache dimensioning and content delivery latency. This MDP turns out to be a restless multi-armed bandit (RMAB) problem [6]. Though in theory it can be solved by relative value iteration [5], this approach suffers from the curse of dimensionality. Therefore, it is highly desirable to design provably optimal and low-complexity solutions. A celebrated heuristic is the Whittle index policy [6], which relaxes the hard constraint, in which the number of cached contents is *exactly* the leased storage, to a time-averaged constraint, in which the number of cached contents is the leased storage *on average*. However, to the best of our knowledge, there is no such Whittle index policy for dynamic dimensioning of cloud caches. Part of the difficulty is

Manuscript received 24 July 2022; revised 21 November 2022; accepted 30 December 2022; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor I.-H. Hou. This work was supported in part by the National Science Foundation (NSF) under Grant CRII-CNS-NeTS-2104880 and Grant RINGS-2148309; in part by the Office of the Under Secretary of Defense, Research and Engineering (OUSD R&E), National Institute of Standards and Technology (NIST), and industry partners as specified in the Resilient and Intelligent NextG Systems (RINGS) Program; and in part by the U.S. Department of Energy under Grant DE-EE0009341. This work was partially presented in the IEEE Conference on Computer Communications (IEEE INFOCOM), Virtual Conference, May 2022 [DOI: 10.1109/INFOCOM48880.2022.9796809]. *(Corresponding author: Jian Li.)*

The authors are with the Department of Electrical and Computer Engineering, Binghamton University, State University of New York, Binghamton, NY 13902 USA (e-mail: gxiong1@binghamton.edu; swang214@binghamton.edu; gyan2@binghamton.edu; lij@binghamton.edu).

Digital Object Identifier 10.1109/TNET.2023.3235480

that Whittle index policy is not feasible when the lease storage is unknown, which needs to be repeatedly solved and jointly optimized with content caching.

In this paper, we design a new Whittle-like index policy for dynamic dimensioning of cloud caches. Similar to Whittle [6], we first relax the MDP, and then overcome the above difficulties of directly designing Whittle policy by studying the corresponding fluid dynamics, the optimal solution to which is described by a linear programming (LP) problem. The optimal values of the LP (i.e., the optimal dimensioning decisions) provide a lower bound on the cost of the original MDP [7], [8], and are then used to design Whittle index policy for the original MDP. We show that our original MDP is indexable, and derive explicitly the Whittle indices of each content. Based on this result, we propose an associated policy by caching contents with the largest Whittle indices whose number is constrained by the optimal dimensioning decisions. Finally, we prove that our proposed index policy is asymptotically optimal. Our contribution in Section IV is non-trivial since establishing the Whittle indexablity of RMAB problems is typically intractable [9] and the Whittle indices of many practical problems remain unknown except for a few special cases. Exacerbating this problem is the fact that the cache dimensioning and content caching decisions are coupled in our problem.

Due to the time-varying nature of cloud CDNs, the system parameters such as content request and delivery rates are typically unknown in practice. In Section V, we further propose a reinforcement learning (RL) augmented algorithm to address this issue. However, simply applying off-the-shelf methods such as UCRL2 [10] or Thompson Sampling [11] is tricky due to the curse of dimensionality, since these generic RL approaches ignore the rich underlying structure of our problem and hence are inefficient of learning in our settings. To this end, we propose *fW-UCB* that not only leverages the approach of *optimism-in-the-face-of-uncertainty* [10], [12] to balance exploration and exploitation, but more importantly, it learns to leverage the near-optimal index policy for making decisions. We show that *fW-UCB* achieves an optimal sub-linear regret with a low-complexity, and hence can be easily implemented in real systems. To the best of our knowledge, our work is the first in the literature to design an index based RL algorithm for dynamic dimensioning of cloud caches.

We support our analytical results with extensive evaluations in Section VI using both synthetic trace based and real trace based simulations, and a Redis-based implementation [13] running on our experimental testbed via Amazon ElastiCache service [14]. Numerical results demonstrate the superior performance of our proposed policies over state of the arts that are employed in today's major CDNs. Furthermore, our evaluation results are in close agreement with theoretical models and confirm a desired tradeoff between cache dimensioning and content delivery latency costs, which can be achieved by tuning a system-wide parameter.

An earlier version of this work appeared in [1]. The differences between our earlier work and this paper are: (i) we further capture the dynamic nature of cloud CDN with dynamic content population, i.e., the content catalog is varying over time frames in our system model; (ii) we provide the details of the optimistic planning and the extended LP in the design of *fW-UCB*; (iii) full proofs of all theoretical results are presented in the appendix; and (iv) we provide a prototype testbed implementation, i.e., a Redis-based implementation via Amazon ElastiCache Service to cache dimensioning to further evaluate our proposed policies using real traces.

II. Related Work

In this section, we mainly overview two main areas that are closely related to our work: cache dimensioning and restless multi-armed bandits, and further provide a brief discussion of our design methodology in the context of prior work.

Cache Dimensioning. Though offline cache dimensioning with known content requests has been studied for designing CDN systems prior to their deployment [15], [16], [17], [18], it is not appropriate for cloud CDNs with unknown and time-varying content requests. Online cache dimensioning has received little attention to date. A TTL-based approximation approach was proposed for elastic cloud provisioning to minimize the storage and miss costs [19]. We instead formulate the problem as a MDP and study the storage and content delivery latency costs, which are critical for many emerging delay-sensitive applications. Cache dimensioning in wireless CDNs studied in [20] requires strong assumptions on request process, while our proposed research provides provable performance for any request process. Similar problem was considered in memory management systems [21] for the assignment of memory to various applications with a computationally intensive policy. None of these works provided an index based policy for cache dimensioning.

Restless Multi-Armed Bandits. RMAB is a general model for sequential decision making problems ranging from wireless communication [22], [23], [24], congestion control [25], sensor management [26], [27], cloud computing [28], healthcare [29], [30], [31], queueing systems [32], etc. However, the RMAB problem is PSPACE hard [33]. To this end, Whittle [6] proposed a heuristic "Whittle index policy" for the infinite-horizon RMAB. Since then, many studies focused on finding index policies for RMABs, e.g., [8], [34], [35], [36], and [37]. These works assume that the system parameters are known. Since the true parameters are typically unavailable, it is important to examine RMAB from a learning perspective, e.g., [23], [38], [39], [40], [41], [42], [43], and [44]. However, existing RL solutions (e.g., colored-UCRL2 [41] and Thompson sampling-based methods [42], [43]) suffer from an exponentially computational complexity and the regret bound grows exponentially with the size of state space, yielding the algorithms to be too slow to be useful. Recently, [45], [46], [47], [48], [49], [50], [51], [52], and [53] developed RL-based algorithms to explore the problem structure through index policies. However, [45], [46], [47], [49], and [50] lacked finite-time performance analysis and multi time-scale stochastic approximation algorithms usually suffer from slow convergence. The explore-then-commit mechanism [48] depends on the performance of an offline "black-boxed" oracle approximator in the exploitation phase, while our *fW-UCB* leverages an

explicit Whittle index policy. References [52] and [53] focused on the finite-time horizon setting, making their approaches not directly applicable to our problem.

Our Design Philosophy. This paper contributes to both areas. First, we pose the cache dimensioning problem as a RMAB to minimize the overall costs, including not only dimensioning costs, bust also content delivery latency costs. This enables us to characterize tradeoffs between those two interrelated costs. Second, we depart from existing assumptions on content request processes and consider this RMAB from an online perspective. A key differentiator between our approach and existing ones stems from two perspectives: (i) we focus on designing index policies for dynamic cache dimensioning, which operate on a much smaller dimensional subspace by exploiting the inherent structure of our problem; and (ii) our index-policy approach naturally lends itself to a lightweight RL based framework that can fully exploit the structure of our index policy so as to reduce the high computational complexity and achieve an optimal sub-linear regret.

III. Model and Problem Formulation

In this section, we present the system model and formulate the dynamic cache dimensioning problem.

A. System Model

Consider a system where a CP dynamically leases storage from cloud CDNs to provide services to users. We note that there is a natural timescale separation between cache dimensioning and content caching, where the former is a much slower operation than the latter. To this end, we consider a two-timescale dynamic system. Specifically, the cache dimensioning is performed in a slower (discrete) timescale indexed by $k \in \mathcal{K} = \{1, \cdots, K\}$ and $K < \infty$, while the content caching is performed in a faster (continuous) timescale indexed by t. We call each k as one *frame* with a fixed duration, e.g., 1 hour or 1 day in some real-world systems. Due to the time-varying nature of content requests, "new" contents may be generated while "old" contents may no longer be requested. As a result, the content catalog is often changing over time. To this end, we denote the dynamic content catalog over each frame as $\mathcal{N}_k, \forall k$. Without loss of generality, we denote the *full content catalog* over all frames as $\mathcal{N} = \{1, \cdots, N\}$ with each integer representing a different content of equal size, and hence $\mathcal{N}_k \subseteq \mathcal{N}, \forall k$.

Requests. Requests for content $n \in \mathcal{N}_k$ follow a Poisson process[1] with arrival rate $\lambda_{n,k}$ in frame k. The time taken to deliver content n to users in frame k is a random variable that is exponentially distributed with mean $1/\mu_{n,k}$, which is independently across different contents. Note that the content arrival rates and delivery rates often vary across different frames due to the time-varying nature of CDN systems which necessitates the need of dynamic cache dimensioning for CPs.

[1]Poisson arrivals are widely used in the literature, e.g., [54], [55], [56], [57], [58], and [59]. However, our model holds for general stationary process [60] and our RL solutions in Section V hold for any request process. See Section V for details.

Cache Dimensioning. At the beginning of each frame, the CP determines how much storage to lease from the cloud CDN. We denote the leased storage in frame k as a random variable B_k, which is measured in the number of content size units. Note that B_k must be non-negative as it is impossible to sell storage, and it is meaningless to lease more than $|\mathcal{N}_k|$ content size units since that would be enough to fit the entire content catalog. Hence we have $B_k \in [0, |\mathcal{N}_k|], \forall k$.

Content Caching. We use a binary variable A_n to indicate caching decisions on content n, where $A_n = 1$ means content n is cached and $A_n = 0$, otherwise. Since the CP leases B_k storage in frame k, the set of feasible content caching decisions in frame k is $\mathcal{A}_k = \{A_n \in \{0,1\}^{|\mathcal{N}_k|} : \sum_{n=1}^{|\mathcal{N}_k|} A_n \leq B_k\}$.

B. System Dynamics and Problem Formulation

Now we formulate the dynamic cache dimensioning problem for the above model as a MDP.

States. We denote the number of outstanding requests for content n at time t in frame k as $S_n(k,t) \leq S_{\max}$, where $S_{\max}$ is the maximum number of requests expected in each frame for any content, and can be arbitrarily large but bounded. Then the states of the system at time t in frame k can be denoted $\mathbf{S}(k,t) = (S_1(k,t), \cdots, S_{|\mathcal{N}_k|}(k,t))$. For ease of readability, we denote $\mathcal{S}$ as the finite state space in our system.

Actions. The action $A_n(k,t)$ for content n at time t in frame k is defined as whether to cache content n or not. Hence $A_n(k,t) \in \mathcal{A}_k$, $\forall t$. Since the cache dimensioning decision B_k is fixed in frame k and for abuse of notation, denote $\mathbf{A}(k,t) := (A_1(k,t), \cdots, A_{|\mathcal{N}_k|}(k,t), B_k)$, and $\mathbf{A} = \{(\mathbf{A}(1,t), \cdots, \mathbf{A}(K,t)), \forall t\}$. Decisions are made only at those time instants when either a new request arrives, or a content delivery occurs. As a result, $\mathbf{A}(k,t)$ stays unchanged in between these time instants.

A cache dimensioning policy π maps the state of the system $\mathbf{S}(k,t)$ to the action $\mathbf{A}(k,t)$, i.e., $\mathbf{A}(k,t) = \pi(\mathbf{S}(k,t))$.

Controlled Transition Kernel. At each time instant in frame k, the state of content n can change from S_n to either $S_n + 1$ or $(S_n - 1)_+$ with other $|\mathcal{N}_k| - 1$ contents' states unchanged. The transition rates satisfy

$$\mathbf{S} \to \begin{cases} \mathbf{S} + \mathbf{e}_n, & \text{with transition rate } b_{n,k}(S_n, A_n), \\ \mathbf{S} - \mathbf{e}_n, & \text{with transition rate } d_{n,k}(S_n, A_n), \end{cases} \tag{1}$$

where $\mathbf{e}_n$ is a $|\mathcal{N}_k|$-dimensional vector with the n-th entry being 1 and all other elements being zero, and $b_{n,k}(\cdot,\cdot)$ and $d_{n,k}(\cdot,\cdot)$ are given as $b_{n,k}(S_n, A_n) = \lambda_{n,k}, d_{n,k}(S_n, A_n) = \mu_{n,k}(S_n)A_n$, with $\mu_{n,k}(0) = 0$. We allow a state-dependent content delivery rate, which enables us to model realistic settings [61], [62], [63]. In this paper, we consider the classic $M/M/k$ queue, i.e., $d_{n,k}(S_n, A_n) = \mu_{n,k} S_n A_n$.

Dynamic Cache Dimensioning Problem. A CP may have different requirements on content delivery latency and dimensioning costs. A CP for applications that are more delay-sensitive may place a greater penalty for its content delivery latency, while a CP with a smaller dimensioning expenditure may be more sensitive to the dimensioning costs. To posit a tradeoff between content delivery latency and dimensioning

cost, we incorporate unit costs for content delivery latency, c_d and for cache dimensioning, c_b.

Our goal is to derive a policy π to minimize the total expected costs incurred by cache dimensioning and content delivery latency as defined below:

$$C_\pi(\mathbf{A}) = \kappa C_{\text{latency}} + (1-\kappa)C_{\text{dim}}, \tag{2}$$

where C_{latency} is the total content delivery latency cost, C_{dim} is the total cache dimensioning cost and $\kappa \in [0,1]$ is a system-wide weighting factor that determines how the two costs are weighted against each other. By Little's Law, the total expected content delivery latency cost under policy π is given by

$$C_{\text{latency}} = \mathbb{E}_\pi\left(c_d \sum_{k=1}^{K} \limsup_{T_k\to\infty} \sum_{n=1}^{|\mathcal{N}_k|} \frac{1}{T_k} \int_{t=1}^{T_k} S_n(k,t)dt \right), \tag{3}$$

where the subscript denotes the fact that expectation is taken with respect to the measure induced by the policy π. We focus on Markovian policies which base their decisions on the current state and time. Similarly, we have $C_{\text{dim}} = \sum_{k=1}^{K} c_b B_k$. Then we can write the overall cost under policy π as

$$C_\pi(\mathbf{A}) = \mathbb{E}_\pi\left(\kappa c_d \sum_{k=1}^{K} \limsup_{T_k\to\infty} \sum_{n=1}^{|\mathcal{N}_k|} \frac{1}{T_k} \int_{t=1}^{T_k} S_n(k,t)dt + (1-\kappa)c_b \sum_{k=1}^{K} B_k \right). \tag{4}$$

Therefore, our dynamic cache dimensioning problem to minimize the total costs subject to the dimensioning constraint in each frame can be formulated as the following MDP:

$$\min_{\pi\in\Pi} \; C_\pi(\mathbf{A}), \quad \text{s.t.} \sum_{n=1}^{|\mathcal{N}_k|} A_n(k,t) \le B_k, \; \forall k,t. \tag{5}$$

We refer to (5) as the "original problem".

Remark 1: We cannot directly apply the conventional relative value iteration method [5] to solve (5) since the cache dimensioning and content caching decisions are strongly coupled and need to be jointly optimized in (5). Even when the cache dimensioning decisions, i.e., B_k, $\forall k$ are given, it is well known that solving the above MDP using relative value iteration suffers from the curse of dimensionality and lacks of insights. The dimension is dominated by the number of contents N, the state space $\mathcal{S}$, and the number of frame K. As a result, many efforts have been focusing on designing low-complexity solutions. One celebrated heuristic is the Whittle index policy [6]. However, Whittle index policy is infeasible [6], [7], [8] if the resource constraint is not given or unknown, which in our case is the cache dimensioning B_k in each frame k. This is because in the RMAB literature, whether to activate an arm (resp. whether to store a content in our model) is not only determined by its Whittle index value, but also depends on constrained resource (resp. B_k in our model). To overcome this challenge, we next introduce a new notion of fluid Whittle index policy.

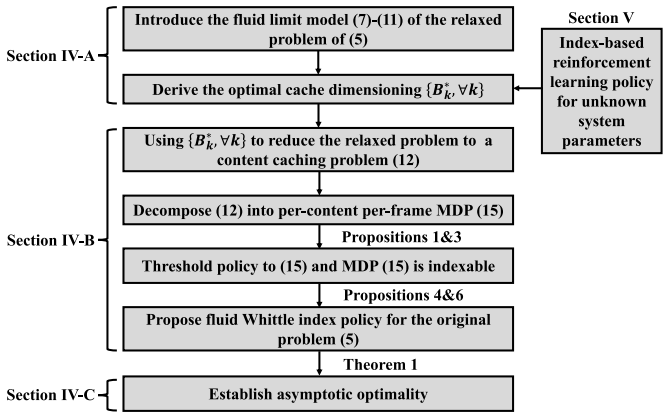


Fig. 1. The framework of the proposed solutions.

IV. Fluid Whittle Index Policy

In this section, we propose asymptotically optimal index policies when cache dimensioning and content caching decisions are coupled. Specifically, we introduce the notation of *fluid Whittle index policy*, which generalizes the classic Whittle index policy to the dynamic cache dimensioning setting.

Our proposed policy is based on a relaxed formulation of (5) and consists of two interdependent steps, as illustrated in Figure 1. First, we show that the optimal cache dimensioning decisions in each frame can be solved using a fluid version of the relaxed problem. Second, given the optimal cache dimensioning decision, we show that our problem is Whittle indexable and explicitly derive the Whittle indices. These two steps enable us to design the *fluid Whittle index policy*, which we show is asymptotically optimal. Note that our relaxed formulation and hence the fluid problem is very general and can be applied to other MDPs that are provably indexable. Hence our methodologies not only apply to the dynamic cache dimensioning problem in this paper but also to other large MDP problems with coupled decision-makings on resource allocation and scheduling.

A. Optimal Fluid Control

In this subsection, we first introduce the relaxed formulation. We then solve the fluid version of this relaxed formulation, i.e., we only take into account the average behavior of the system. The optimal fluid solution provides a lower bound on the optimal cache dimensioning in the original model [7].

Following Whittle's approach [6], we study the relaxed problem in which the cache dimensioning constraint at each time in a frame in (5) is satisfied on average, i.e.,

$$\min_{\pi\in\Pi} \; C_\pi(\mathbf{A}), \quad \text{s.t.} \; \limsup_{T_k\to\infty} \frac{1}{T_k} \mathbb{E}_\pi \int_{t=1}^{T_k} \sum_{n=1}^{|\mathcal{N}_k|} A_n(k,t)dt \le B_k, \; \forall k. \tag{6}$$

We then define the fluid limit model of the relaxed problem under a stationary Markovian policy π, which is independent of the initial state of the MDP [7], [64], [65]. Let $x^a_{n,k,s}$ be the fraction of content n in state s under action a in frame k and let $x_{n,k,s} = x^0_{n,k,s} + x^1_{n,k,s}$ be the fraction of content n

in state s. Denote the transition rate from state s to state s' with action a for content n in frame k as $q_{n,k}(s'|s,a)$. Similar to [8], we focus on finding an optimal equilibrium point of the fluid dynamics that minimize the total costs. Since there exists a stationary Markovian policy π_k for each frame $k \in \mathcal{K}$ that is independent of the initial state, the relaxed problem (6) can be decomposed into each frame k. To this end, we decompose the relaxed problem (6) to obtain the following "per-frame" LP problem for each frame k:

$$\min_{\{x_{n,k,s}\},B_k} \quad \kappa c_d \sum_{n=1}^{|\mathcal{N}_k|} \sum_{a\in\mathcal{A}_k} \sum_{s\in\mathcal{S}} s x^a_{n,k,s} + (1-\kappa) c_b B_k \quad (7)$$

$$\text{s.t.} \sum_{n=1}^{|\mathcal{N}_k|} \sum_{s\in\mathcal{S}} x^1_{n,k,s} \le B_k, \quad (8)$$

$$\sum_{a\in\mathcal{A}_k} \sum_{s'\in\mathcal{S}} x^a_{n,k,s} q_{n,k}(s'|s,a) = \sum_{a\in\mathcal{A}_k} \sum_{s'\in\mathcal{S}} x^a_{n,k,s'} q_{n,k}(s|s',a), \ \forall n \in \mathcal{N}_k, \quad (9)$$

$$\sum_{a\in\mathcal{A}_k} \sum_{s\in\mathcal{S}} x^a_{n,k,s} = 1, \ x^a_{n,k,s} \ge 0, \ \forall n,s,a, \quad (10)$$

$$0 \le B_k \le |\mathcal{N}_k|, \quad (11)$$

where (8) is equivalent with the constraint in (6), (9) represents the equilibrium condition of the fluid system, i.e., the fluid flows in state s equals to the fluid flows out state s, (10) holds due to the definition of $x^a_{n,k,s}$, and (11) follows from the cache dimensioning definition.

Denote the optimal cache dimensioning solutions to the LP (7)-(11) as $B^\star_k$. It is well known that this LP is equivalent to the relaxed problem (6) [7] and there exists a stationary policy for this LP [8]. Furthermore, the fluid analysis achieves a lower bound of the original problem [6], [8], which is leveraged for the asymptotic optimality analysis of index policy.

B. Fluid Whittle Index Policy

Provided the optimal caching dimensioning $B^\star_k$ by solving the fluid control in (7)-(11), the CP only needs to make content caching decisions in each frame k:

$$\min_{\pi_k\in\Pi} \ \mathbb{E}_{\pi_k} \limsup_{T_k\to\infty} \frac{1}{T_k} \sum_{n=1}^{|\mathcal{N}_k|} \int_{t=1}^{T_k} S_n(k,t)dt$$

$$\text{s.t.} \ \limsup_{T_k\to\infty} \frac{1}{T_k} \mathbb{E}_{\pi_k} \int_{t=1}^{T_k} \sum_{n=1}^{|\mathcal{N}_k|} A_n(k,t)dt \le B^\star_k, \quad (12)$$

where we drop the constant κc_d for simplicity but we consider it when compute the total costs. Next we consider the following Lagrangian associated with (12),

$$L_{\pi_k}(W_k) = \limsup_{T_k\to\infty} \frac{1}{T_k} \mathbb{E}_{\pi_k} \int_{t=1}^{T_k} \left[\sum_{n=1}^{|\mathcal{N}_k|} S_n(k,t) - W_k \Big(B^\star_k - \sum_{n=1}^{|\mathcal{N}_k|} A_n(k,t) \Big) \right] dt, \quad (13)$$

where W_k is the Lagrangian multiplier for the k-th frame. The dual function is then defined as

$$D(W_k) := \min_{\pi_k} L_{\pi_k}(W_k). \quad (14)$$

A key observation made by Whittle is that (14) can be decomposed into $|\mathcal{N}_k|$ subproblems, one for each content n. As a result, we obtain the following "per-content per-frame" MDP:

$$\min_{\pi_k} \limsup_{T_k\to\infty} \frac{1}{T_k} \mathbb{E}_{\pi_k} \int_{t=1}^{T_k} \Big(S_n(k,t) - W_k(1 - A_n(k,t)) \Big) dt. \quad (15)$$

Definition 1: (Passive Set) Consider the per-content per-frame MDP in (15), let $\mathcal{M}_n(W_k)$ be the set of states s for which the optimal action for content n in frame k is passive, i.e., not to cache content n in state s in frame k.

Definition 2: (Indexability) The per-content per-frame MDP in (15) for content n in frame k is indexable if the passive set $\mathcal{M}_n(W_k)$ increases with W_k, i.e., if $W_k > W'_k$, then $\mathcal{M}_n(W_k) \supseteq \mathcal{M}_n(W'_k)$.

Definition 3: (Whittle Index) If the per-content per-frame MDP for content n in frame k is indexable, then the Whittle index in state s is denoted as $W_{n,k}(s)$, and is given as follows:

$$W_{n,k}(s) := \inf_{W_k \ge 0} \{ s \in \mathcal{M}_n(W_k) \}, \quad (16)$$

which is the smallest value of the W_k such that the optimal policy for content n is indifferent towards $a = 0$ and $a = 1$ under state s in frame k.

Threshold Policies. We show that there exists an optimal policy of (15) that is of threshold-type, and then the MDP (15) is indexable. Based on these results, we explicitly derive the Whittle indices and the fluid Whittle index policy for the original problem (5).

Proposition 1: For a fixed W_k, there exists a threshold type policy depending on W_k that optimally solves the per-content per-frame MDP in (15).

Proof: Proof is provided in Appendix A-A. □

We then compute the stationary distribution under a threshold policy as a function of the threshold.

Proposition 2: The stationary distribution of the threshold policy $\pi_{n,k} = R$ satisfies $\phi^R_n(R') = 0, \forall 0 \le R' < R$, and

$$\phi^R_n(R) = 1 \Big/ \left(1 + \sum_{j=1}^{\infty} \left(\frac{\lambda_{n,k}}{\mu_{n,k}} \right)^j \frac{1}{\Pi^j_{k=1}(R+k)} \right),$$

$$\phi^R_n(R+l) = \left(\frac{\lambda_{n,k}}{\mu_{n,k}} \right)^l \frac{1}{\Pi^l_{k=1}(R+k)} \phi^R_n(R), l = 1,2,\cdots. \quad (17)$$

For the notation abuse, we use a superscript R to denote the stationary distribution under a particular threshold policy R.

Proof: Proof is provided in Appendix A-B. □

Proposition 3: The MDP (15) is indexable.

Proof: Proof is provided in Appendix A-C. □

We are now ready to derive the Whittle indices for (15).

Proposition 4: For an indexable (15) *with* $\sum_{s=0}^{R}\phi_n^R(s)$ *strictly increasing with* R*, the Whittle index is given by*

$$W_k(R)=\frac{\mathbb{E}_{R+1}[S_n]-\mathbb{E}_R[S_n]}{\sum_{S_n=0}^{R}\phi_n^R(S_n)-\sum_{S_n=0}^{R-1}\phi_n^{R-1}(S_n)}. \quad (18)$$

Proof: Proof is provided in Appendix A-D. □

Since the cost function and the stationary probabilities are known, the Whittle indices (18) can be computed. Such an approach was also used in [62] and [63] for queuing systems.

Fluid Whittle Index Policy. We now describe how the solutions to the LP (7)-(11) and relaxed problem (12) are used to obtain a policy for the original problem (5). It is clear that the optimal solutions to (7)-(11) and (12) are not always feasible for (5), since in the latter at most B_k content can be cached. If B_k is known, then we can follow the Whittle policy. However, in our problem, $B_k, \forall k$ are unknown and need to be solved from the LP (7)-(11). Hence, we refer to this as the *fluid Whittle index policy* as defined below.

Definition 4: (Fluid Whittle Index Policy) At time t *in frame* k*, the Fluid Whittle index policy prioritizes the content in the decreasing order of their Whittle indices* $W_{n,k}(S_n(k,t))$ *and caches the top* $B_k^\star$ *contents that have the largest index values.*

C. Asymptotic Optimality

Our *fluid Whittle Index Policy* achieves asymptotic optimality when the number of contents $|\mathcal{N}_k|$ and the cache dimensioning $B_k^\star$ go to infinity while holding $B_k^\star/|\mathcal{N}_k|$ constant in any frame $k\in\mathcal{K}$. Both $B_k^\star$ and $|\mathcal{N}_k|$ are typically large in practice. Hence we only present the main results in a particular frame k for ease of exposition. This asymptotic regime is the same as Whittle [6] and many others [7], [8], [62], [66]. For abuse of notation, we let the number of contents be $\rho|\mathcal{N}_k|$ and the value of cache dimensioning be $\rho B_k^\star$ in the limit with $\rho\to\infty$. In other words, it represents the scenarios where there are $|\mathcal{N}_k|$ different classes of contents and each class contains ρ contents. Let B_k^{opt} be the optimal cache dimensioning for the original problem (5) obtained by a genie-aided policy π_k^{opt}. Denote the corresponding cost as $C(\pi_k^{opt},\rho B_k^{opt},\rho|\mathcal{N}_k|)$. Similarly, let $C(\pi_k,\rho B_k,\rho|\mathcal{N}_k|)$ be the expected cost under a stationary policy π_k. Following the fact that the LP (7)-(11) is invariant with the scaling parameter ρ, the per-frame optimal cost of the fluid model (7)-(11) satisfies $\rho C_{fluid}(B_k^\star,|\mathcal{N}_k|)\le C(\pi_k^{opt},\rho B_k^{opt},\rho|\mathcal{N}_k|)$.

Theorem 1: Denote the fluid Whittle Index Policy under $B_k^\star$ *as* $\pi_k^\star$*. Then,* $\pi_k^\star$ *achieves the asymptotic optimality as follows*

$$\lim_{\rho\to\infty}\frac{1}{\rho}\Big(C(\pi_k^\star,\rho B_k^\star,\rho|\mathcal{N}_k|)-C(\pi_k^{opt},\rho B_k^{opt},\rho|\mathcal{N}_k|)\Big)=0. \quad (19)$$

Proof: Proof is provided in Appendix A-E. □

V. Reinforcement Learning Solutions

The knowledge of content request and delivery rates are needed for the computation of the *fluid Whittle index policy*. However, these parameters are often unknown and time-varying in cloud CDNs over different frames. Hence, we now adopt a learning perspective on top of the *fluid Whittle index policy*. We denote the resulting learning rule as *fW-UCB* and show that *fW-UCB* achieves an optimal sub-linear regret. Due to the decomposition nature of our problem across different frames as discussed in Section IV, we describe our proposed *fW-UCB* and its finite-time performance in a particular frame for ease of readability, which applies to any frame $k\in\mathcal{K}$.

Algorithm 1 *fW-UCB* Policy

Require: Horizon T_k in frame k and learning counts $f(T_k)$.
1: **for** $n=1,2,\ldots,|\mathcal{N}_k|$ **do**
2: Observe content n until there are $f(T_k)$ visits of pairs $(s,1)$, $\forall s\in\mathcal{S}$.
3: **end for**
4: Construct $\mathcal{P}_{n,k}(s,1)$ according to (25);
5: Construct the plausible set of MDPs $\mathcal{M}_{n,k},\forall n$;
6: Solve the extended LP in (29) to determine $\tilde{P}_{n,k}^\star(s'|s,1)$ for all s', s according to (30) and $B_k^\star$;
7: Compute Whittle indices (18) based on $\tilde{P}_{n,k}(s'|s,a)$;
8: Establish the corresponding fluid Whittle index policy $\pi_k^\star$;
9: Execute $\pi_k^\star$ for the rest of the time in frame k.

A. Algorithm Description

We adapt the upper confidence bound (UCB) [12] to our setting and design the *fW-UCB* policy as summarized in Algorithm 1, which has the "explore-then-commit" structure. More precisely, *fW-UCB* have two phases: a planning phase and a policy execution phase. The planning phase focuses on defining a set of plausible MDPs [10] based on the number of visits to state-action pairs (s,a) and transitions tuples (s,a,s') as accurate as possible (Lines 1-5). We can define the corresponding *fluid Whittle index policy* by solving an optimistic planning problem, which is expressed as an LP (Lines 6-8). Our key contribution here is to choose the right value of $f(T_k)$ to balance the accuracy and complexity, which contributes to the sub-linear regret and low-complexity of *fW-UCB*. At the policy execution phase (Line 9), the derived *fluid Whittle index policy* is executed for the rest in frame k. Our key contribution here is to fully exploit our *fluid Whittle index policy*, instead of directly contending with the high-dimensional state-action space as in generic RL algorithms. As a result, *fW-UCB* achieves a sub-linear regret and performs close to the offline optimum since our proposed *fluid Whittle index policy* is asymptotically optimal.

1) ***Optimistic Planning****:* In frame k, the CP first observes each content $n\in\mathcal{N}_k$ with a state-action pair (s,a) for $f(T_k)$ times (the value of $f(T_k)$ will be specified later). Since the transition is deterministic with $a=0$, we only observe the transition under $a=1$. We denote the number of times that a transition tuple $(s,1,s')$ was observed within $f(T_k)$ as

$$T_{n,k}(s,1,s')=\sum_{h=1}^{f(T_k)}\mathbb{1}(s_n(h+1)=s'|s_n(h)=s,a_n(h)=1), \times\forall(s,1,s')\in\mathcal{S}\times\mathcal{A}\times\mathcal{S}, \quad (20)$$

where $s_n(h)$ represents the state for content n at time h, $a_n(h)$ is the corresponding action and $\mathcal{A}=\{0,1\}$. Without loss of generality, we normalize the true transition rates to

be 1 at any moment in time, i.e., $\sum_{s'\in\mathcal{S}} P_{n,k}(s'|s,1) = 1, \forall n,k,s$. Following (1), the true transition rate $P_{n,k}(s'|s,1)$, $\forall (s,a,s') \in \mathcal{S}\times\mathcal{A}\times\mathcal{S}$ satisfies

$$P_{n,k}(s'|s,1) = \begin{cases} \lambda_{n,k}, & \text{if } s' = s+1, \\ s\mu_{n,k}, & \text{if } s' = s-1, \\ 1-\lambda_{n,k}-s\mu_{n,k}, & \text{if } s' = s. \end{cases} \tag{21}$$

In order to estimate the true transition rates, the CP first counts the corresponding empirical probability as

$$I_{n,k}(s'|s,1) = \frac{T_{n,k}(s,1,s')}{f(T_k)}. \tag{22}$$

Since the observations $T_{n,k}(s,1,s')$ are made only when state transition occurs, from (21) and (22), we have

$$\frac{\bar{\lambda}^s_{n,k}}{\bar{\mu}^s_{n,k}} = \frac{sI_{n,k}(s+1|s,1)}{I_{n,k}(s-1|s,1)} = \frac{sT_{n,k}(s,1,s+1)}{T_{n,k}(s,1,s-1)}, \ \forall s\in\mathcal{S}. \tag{23}$$

To this end, the CP estimates the true transition rates by the corresponding empirical average as

$$\bar{P}_{n,k}(s'|s,1) = \begin{cases} \frac{sT_{n,k}(s,1,s+1)}{sT_{n,k}(s,1,s+1)+T_{n,k}(s,1,s-1)}, & \text{if } s' = s+1, \\ \frac{sT_{n,k}(s,1,s-1)}{sT_{n,k}(s,1,s+1)+T_{n,k}(s,1,s-1)}, & \text{if } s' = s-1. \\ \frac{-(s-1)T_{n,k}(s,1,s-1)}{sT_{n,k}(s,1,s+1)+T_{n,k}(s,1,s-1)}, & \text{if } s' = s. \end{cases} \tag{24}$$

The CP further defines the confidence interval such that the true transition rates lie in them with high probability. Formally, for any content n in state s at frame k, we define

$$\mathcal{P}_{n,k}(s,1) := \{\tilde{P}_{n,k}(s'|s,1) : |\tilde{P}_{n,k}(s'|s,1) - \bar{P}_{n,k}(s'|s,1)| \leq \delta^s_{n,k}\}, \tag{25}$$

where the size of the confidence interval $\delta^s_{n,k}$ is built using the Hoeffding inequality [67], i.e., $\forall \eta\in(0,1)$,

$$\delta^s_{n,k} = \sqrt{\frac{s^2}{2f(T_k)}\log\frac{|\mathcal{S}||\mathcal{N}_k|f(T_k)}{\eta}}. \tag{26}$$

The set of plausible MDPs associated with the confidence intervals is then given as

$$\mathcal{M}_k = \{P_{n,k} | P_{n,k}(\cdot|s,1)\in\mathcal{P}_{n,k}(s,1), \forall n\in\mathcal{N}_k, s\in\mathcal{S}\}. \tag{27}$$

Then *fW-UCB* computes a policy $\pi^\star_k$ by performing optimistic planning. In other words, given the set of plausible MDPs, it selects an optimistic MDP and an optimistic policy with respect to our MDP formulation, which is expressed as the following LP:

$$\begin{aligned} \min_{B_k,\{\tilde{P}_{n,k}\}} \ & \kappa c_d \sum_{n=1}^{|\mathcal{N}_k|}\sum_{a\in\mathcal{A}_k}\sum_{s\in\mathcal{S}} s x^a_{n,k,s} + (1-\kappa)c_b B_k \\ \text{s.t.} \ & \sum_{n=1}^{|\mathcal{N}_k|}\sum_{s\in\mathcal{S}} x^1_{n,k,s} \leq B_k, \\ & \sum_{a\in\mathcal{A}_k}\sum_{s'\in\mathcal{S}} x^a_{n,k,s}\tilde{P}_{n,k}(s'|s,a) \\ & = \sum_{a\in\mathcal{A}_k}\sum_{s'\in\mathcal{S}} x^a_{n,k,s'}\tilde{P}_{n,k}(s|s',a), \\ & \sum_{a\in\mathcal{A}_k}\sum_{s\in\mathcal{S}} x^a_{n,k,s} = 1, \ x^a_{n,k,s} \geq 0, \ \forall n,s,a, \\ & 0 \leq B_k \leq |\mathcal{N}_k|. \end{aligned} \tag{28}$$

*2) **The Extended LP**:* The LP (28) is similar to the LP in (7)-(11) except that the true transition rates are unknown. This makes it hard to solve. To this end, we rewrite it as an extended LP problem by defining $z_{n,k}(s,a,s') := x^a_{n,k,s}\tilde{P}_{n,k}(s'|s,a)$ to express the confidence intervals of the transition rates. The extended LP over $z = \{z_{n,k}(s,a,s')\}$ is expressed as

$$\begin{aligned} \min_{B_k,\{z_{n,k}\}} \ & \kappa c_d \sum_{n=1}^{|\mathcal{N}_k|}\sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}_k}\sum_{s'\in\mathcal{S}} s z_{n,k}(s,a,s') \\ & + (1-\kappa)c_b B_k \\ \text{s.t.} \ & \sum_{n=1}^{|\mathcal{N}_k|}\sum_{s\in\mathcal{S}}\sum_{s'\in\mathcal{S}} z_{n,k}(s,1,s') \leq B_k, \\ & \sum_{a\in\mathcal{A}_k}\sum_{s'\in\mathcal{S}} z_{n,k}(s,a,s') = \sum_{a\in\mathcal{A}_k}\sum_{s'\in\mathcal{S}} z_{n,k}(s',a,s), \\ & \frac{z_{n,k}(s,1,s')}{\sum_y z_{n,k}(s,1,y)} - (\bar{P}_{n,k}(s'|s,1) + \delta^s_{n,k}) \leq 0, \\ & -\frac{z_{n,k}(s,1,s')}{\sum_y z_{n,k}(s,1,y)} + (\bar{P}_{n,k}(s'|s,1) - \delta^s_{n,k}) \leq 0, \\ & \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}_k}\sum_{s'\in\mathcal{S}} z_{n,k}(s,a,s') = 1, \ \forall n,s,a, \\ & 0 \leq B_k \leq |\mathcal{N}_k|. \end{aligned} \tag{29}$$

Since the extended LP (29) is for the per-frame MDP, the computational complexity is $O(N|\mathcal{S}|^2)$, i.e., linear in the number of contents N and quadratic in the dimension of state space $|\mathcal{S}|$. This approach was also used for adversarial and constrained MDPs [68], [69], [70]. Once we compute the optimal $z^\star_{n,k}$ and $B^\star_k$, the transition rates are recovered by

$$\tilde{P}^\star_{n,k}(s'|s,1) = \begin{cases} \frac{sz^\star_{n,k}(s,1,s+1)}{sz^\star_{n,k}(s,1,s+1)+z^\star_{n,k}(s,1,s-1)}, & \text{if } s' = s+1, \\ \frac{sz^\star_{n,k}(s,1,s-1)}{sz^\star_{n,k}(s,1,s+1)+z^\star_{n,k}(s,1,s-1)}, & \text{if } s' = s-1. \\ \frac{-(s-1)z^\star_{n,k}(s,1,s-1)}{sz^\star_{n,k}(s,1,s+1)+z^\star_{n,k}(s,1,s-1)}, & \text{if } s' = s. \end{cases} \tag{30}$$

Finally, using these transition rates, we can compute the Whittle indices in (18) for all states of all contents, and then define the *fluid Whittle index policy* as in Definition 4, and execute this policy for the rest of time in frame k.

Though the whole system (the content caching process) operates in continuous time, decisions are made only at those time instants when either a new request arrives, or a content delivery occurs (see Section III-B). Those time instants can be treated as in the discrete time domain. Hence, in this section, samples are made on discrete time instants, e.g., in (20).

B. The Learning Regret

We evaluate the efficiency of *fW-UCB* policy using *regret*, which is defined as the expected gap between the cost obtained

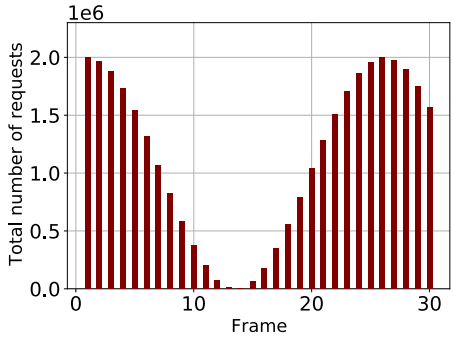


Fig. 2. Number of requests per frame in synthetic trace.

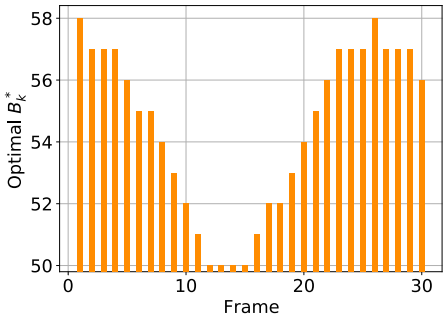


Fig. 3. Cache dimensioning per frame in synthetic trace.

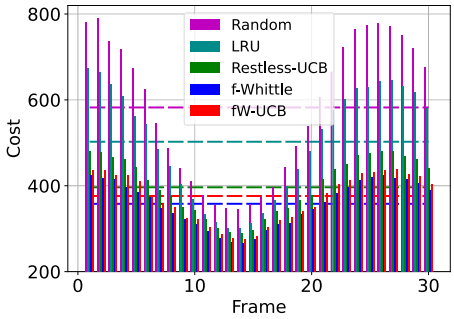


Fig. 4. Comparison of costs in synthetic trace.

by the offline optimum, i.e., the genie-aided policy with full knowledge of all transition rates, and that of *fW-UCB*. Denote the cumulative cost under policy π_k as $C(\pi_k, T_k) := \kappa c_d \sum_{t=1}^{T_k} \sum_{n=1}^{|\mathcal{N}_k|} S_n(t)$, which is a random variable. Then the expected average cost under policy π_k satisfies $\gamma_k := \lim_{T_k \to \infty} \frac{1}{T_k} \mathbb{E}_{\pi_k}[C(\pi_k, T_k)]$, and the optimal average reward is $\gamma_k^{opt} := \inf_{\pi_k} \gamma_k$. Then the regret of π_k is defined as $\Delta(T_k) := \mathbb{E}_{\pi_k}[C(\pi_k, T_k)] - T_k \gamma_k^{opt}$.

Theorem 2: The regret of fW-UCB policy satisfies

$$\Delta(T_k) = \mathcal{O}\left((1 + B_k^\star \sqrt{\log T_k}) S_{max} |\mathcal{N}_k| \sqrt{T_k}\right). \quad (31)$$

Proof Outline: Since there are two phases in *fW-UCB*, we decompose the regret in two parts as $\Delta(T_k) = \Delta(L_1) + \Delta(L_2)$, where $\Delta(L_1)$ is the regret for the planning phase with any random policy and $\Delta(L_2)$ is the regret for the policy execution phase, and $L_2 = T_k - L_1$. The regret of the planning phase is upper bounded by $\mathcal{O}(S_{\max} |\mathcal{N}_k| \sqrt{T_k})$. The regret of the policy execution phase is caused by either "failure event" (confidence ball fails) or "good event" (true MDP is within confidence ball). We show that they are bounded by $\mathcal{O}(2|\mathcal{N}_k|\eta\sqrt{T_k})$ and $\mathcal{O}(S_{\max} B_k^\star |\mathcal{N}_k| \sqrt{T_k \log T_k})$, respectively. Combining them completes the proof. Please see Appendix B for the detailed proof.

Remark 2: Although fW-UCB is a non-episodic algorithm in each frame k, it still achieves the $\mathcal{O}(\sqrt{T_k})$ regret no worse than the episodic UCRL2. Specifically, the proposed fW-UCB spends no time for searching a better MDP instance. Instead, it constructs a upper confidence ball for all plausible MDPs. Then, it determines the optimal policy by calculating the extended LP (29) for only once, which significantly reduces the exponential implementation complexity compared with UCRL2 [10], colored-UCRL2 [41] to a linear scale.

VI. Numerical Results

In this section, we numerically evaluate the performance of our proposed *fluid Whittle index policy* (f-Whittle) and *fW-UCB* using both synthetic trace based and real trace based simulations, and a Redis-based implementation [13] running on our experimental testbed via Amazon ElastiCache service [14]. The LP is solved using Gurobi. We compare to the widely used Least Recently Used (LRU) and Random policies, as well as Restless-UCB [48], a state-of-the-art RL solution for RMAB with low computational complexity. Since LRU, Random and Restless-UCB are designed for a given cache size and do not account for cache dimensioning, we evaluate them using the same cache dimensioning decisions over each frame that are determined by our proposed policies.

A. Evaluation Using Synthetic Trace

We simulate a system with $N = 100$ contents over $K = 30$ frames. The arrival rate for requests in each frame are generated according to a Zipf distribution. The content delivery rate is set to be 0.1. The Zipf parameter is fixed to be 0.6 but the total number of requests varies across frames, as shown in Figure 8. We use $\kappa = 0.5$, $c_d = 1$ and $c_b = 10$. The leased storage in each frame is shown in Figure 9. The per-frame cost is presented in Figure 4, with the dashed lines representing the average cost over all frames. It is evident from Figure 9 that dynamic cache dimensioning can adaptively tune the leased storage to meet the time-varying trends of content requests, as it is common in cloud CDNs. This results in a much smaller cost than conventional policies (LRU and Random) and RL-based solution, Restless-UCB as shown in Figure 4. These results reflect the intelligence of our proposed policies: when the number of requests is low, the CP can lease less cloud storage with a smaller total cost. We further note that our learning policy *fW-UCB* can almost achieve the same performance as the f-Whittle, which is consistent with our theoretical results.

We further validate the asymptotic optimality of our proposed *fluid Whittle index policy* (f-Whittle) (see Theorem 1) and the sub-linear regret of *fW-UCB* (see Theorem 2). Due to the decomposition nature of our problem, we use the requests from a frame randomly selected in Figure 8. Similar trends hold for all other frames and hence are omitted. We consider a similar setting as above. The average cost obtained by our

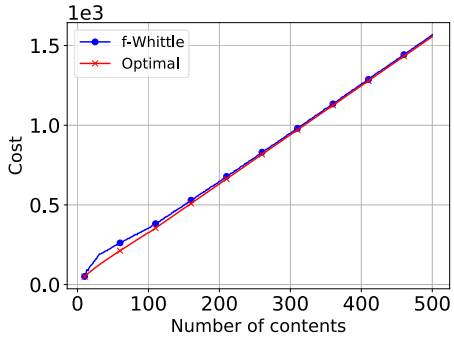


Fig. 5. Asymptotic optimality.

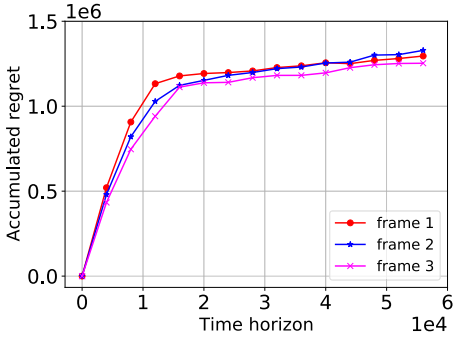


Fig. 6. Accumulated regret.

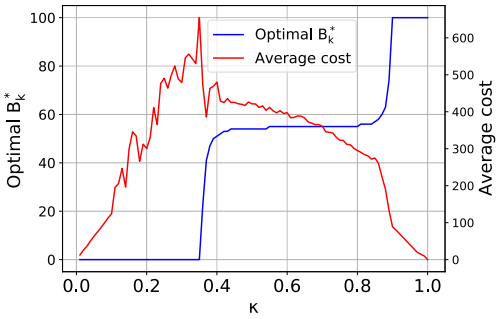


Fig. 7. Impact of tradeoff parameter κ.

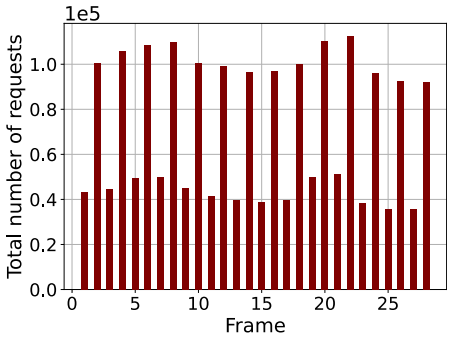


Fig. 8. Number of requests per frame in synthetic trace.

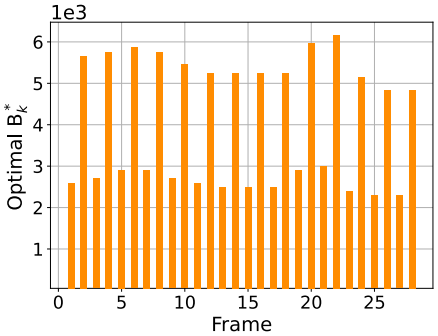


Fig. 9. Cache dimensioning per frame in synthetic trace.

f-Whittle and the theoretical upper bound achieved by solving the LP in (7)-(11) is shown in Figure 5. It is clear that f-Whittle achieves asymptotic optimality when the number of contents increases. The learning regrets of *fW-UCB* in three randomly selected frames are shown in Figure 6, where we use the Monte Carlo simulation with $10,000$ independent trails. Finally, we investigate the impact of the system-wide parameter κ. The blue curve in Figure 7 presents the optimal cache dimensioning vs. κ in one frame and the red curve is the corresponding average cost. A key takeaway from Figure 7 is that the variation of cache dimensioning and content delivery latency costs as κ goes from 0 to 1. When κ is small, our problem (5) weighs more on minimizing cache dimensioning costs, hence a smaller leased storage is preferred. For example, when $\kappa < 0.35$, no leasing is the optimal decision. On the other hand, when κ is large, our problem tends to minimize the content delivery latency costs, and hence a larger leased storage is beneficial. This provides a tunable knob that can be used by the network operator to balance the dimensioning and latency costs for applications with different dimensioning-latency tradeoff requirement.

B. Evaluation Using Real Trace

We further evaluate our proposed policies for dynamic cache dimensioning using two real CDN traces: (i) *Iqiyi* [71], which contains mobile video behaviors; and (i) *YouTube* [72], which contains trace data about user requests for specific YouTube content collected from a campus network. For the Iqiyi (resp. YouTube) trace, there are more than 6.7 (resp. 0.6) million requests for more than 1.4 million (resp. 0.3) unique contents over a period of 335 (resp. 336) hours. To this end, we consider the cache dimensioning occurring every 1, 3, 6, 12, and 24 hours. For ease of exposition, we only present results for 12 hours and similar trends hold for other cases, and hence are omitted here. The number of requests and the cache dimensioning in each frame in Iqiyi and YouTube traces are shown in Figures 8 and 9, and Figures 11 and 12, respectively. Again, it is clear that our dynamic cache dimensioning is able to tune the leased storage to follow the variations of content requests in real systems. As a result, our proposed policies significantly outperform existing policies with a smaller cost as shown in Figure 10 for Iqiyi trace and in Figure 13 for YouTube trace, respectively. Finally, we note that *fW-UCB* can quickly learn the system dynamics and perform close to f-Whittle, which matches well with our theoretical results.

C. Prototype Evaluation

Our implementation testbed consists of a cache server and an origin server. The cache server receives requests and checks

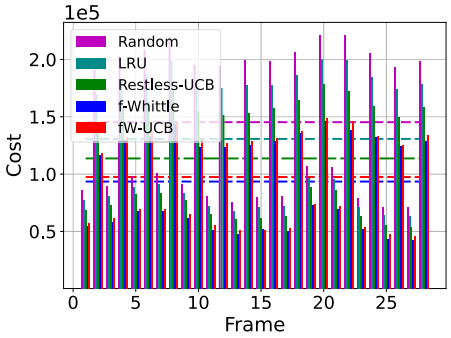


Fig. 10. Comparison of total costs in Iqiyi trace.

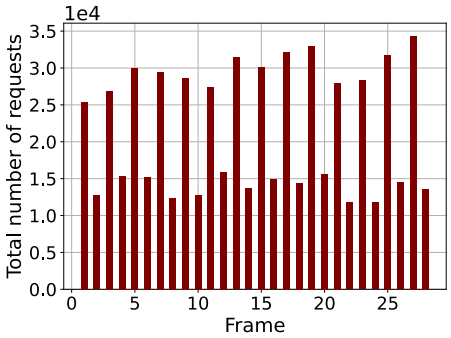


Fig. 11. Number of requests per frame in YouTube trace.

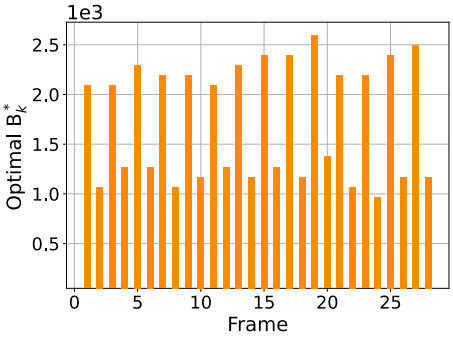


Fig. 12. Cache dimensioning per frame in YouTube trace.

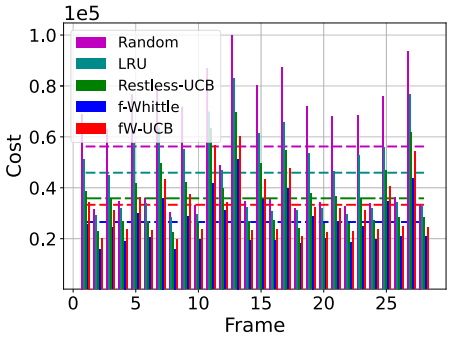


Fig. 13. Comparison of total costs in YouTube trace.

if the content is in the cache. If not, the cache server retrieves the content from the origin server, serves the user and stores the content in the cache. In our evaluation, our proposed algorithms on the cache server is running the Redis [13], which is one of the most popular open source, in-memory data store.

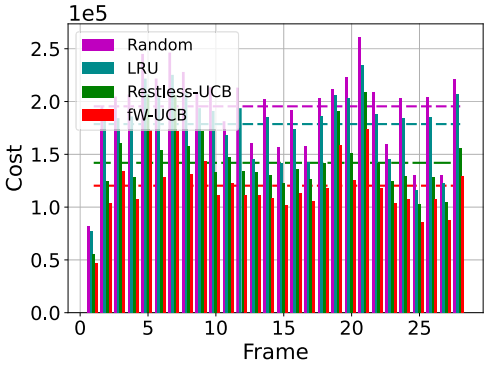


Fig. 14. Prototype results using Iqiyi trace.

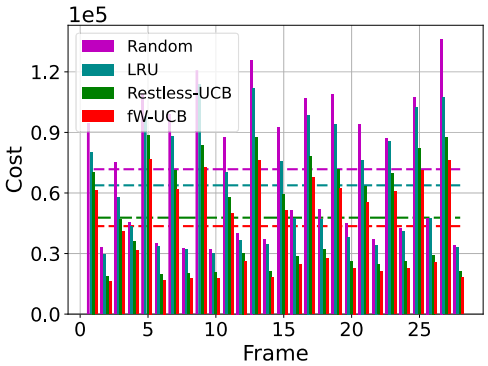


Fig. 15. Prototype results using YouTube trace.

We leverage the same setting as in the trace-base simulations using Iqiyi and YouTube traces. In addition, we implement *fW-UCB* in the prototype design since it is an online algorithm while f-Whittle is offine. The prototype results are presented in Figures 14 and 15, respectively. Again, we observe that *fW-UCB* consistently outperforms existingR policies.

VII. Conclusion

We studied the problem of dynamic cache dimensioning in the cloud to minimize the total average costs for both storage and content delivery latency. Though it can be posed as an MDP, it is hard to solve due to the curse of dimensionality. To this end, we proposed a fluid Whittle index policy which is provably asymptotically optimal. Since the system parameters are often unknown and time-varying in the cloud, we proposed an RL algorithm entitled *fW-UCB* that fully exploit the structure of our index policy and hence is lightweight. We showed that it has an optimal sub-linear regret. Extensive simulations using real traces demonstrated the significant gains of our proposed policies over conventional ones.

Appendix A
Proofs on Fluid Whittle Index Policy

A. Proof of Proposition 1

Proof: Since the set of feasible policies Π is non-empty, there exists a stationary policy π^* that optimally solves (15). Let $R^* = \max\{S \in \{0, 1, \cdots\} : A_n^{\pi^*}(S) = 0\}$, where we use the superscript π^* to denote actions under policy π^* and let

$A_n^{\pi^*}(S)$ be the action at state S for content n under policy π^*. By definition, we have $A_n^{\pi^*}(S) = 1, \forall S > R^*$.

Given the transition rates in (1), we have $d_n(S_n, 0) = d_n(S_n, 1) = 0, \forall S_n < R^*$. Zero departure rate indicates that all states below R^* are transient [63], implying the stationary probability for content n in state S_n under policy π^* being 0, i.e., $\phi_n^{\pi^*}(S_n) = 0, \forall S_n < R^*$. Hence, the average cost under the optimal policy π^* reduces to

$$\mathbb{E}_{\pi^*}[S] - W_k \mathbb{E}[\mathbb{1}_{\{A_n^{\pi^*}(S)=0\}}] = \sum_{S=0}^{R^*-1} S\phi_n^{\pi^*}(S) + R^*\phi_n^{\pi^*}(R^*) + \sum_{S=R^*+1}^{\infty} S\phi_n^{\pi^*}(S) - W_k \sum_{S=0}^{R^*} \phi_n^{\pi^*}(S)\mathbb{1}_{\{A_n^{\pi^*}(S)=0\}} = \sum_{S=R^*}^{\infty} S\phi_n^{\pi^*}(S) - W_k\phi_n^{\pi^*}(R^*),$$

i.e., the threshold-type policy optimally solves (15). Similar techniques have also been used in [62] and [63], we present the proof here for completeness. □

B. Proof of Proposition 2

Proof: For ease of exposition, we only consider the state of content n in frame k. Denote its queue length as S. From our definition, the transition rate satisfies $q_{n,k}(S+1|S,0) = q_{n,k}(S+1|S,1) = \lambda_{n,k}$, and $q_{n,k}(S-1|S,0) = 0$ for $S \le R$ and $q_{n,k}(S-1|S,1) = \mu_{n,k}S$ for $S > R$. It is clear that the dummy states in which $R' < R$ is transient because the queue length keeps increasing. Therefore, the stationary probabilities for dummy states are all zero, i.e, $\phi_n^R(R') = 0$. Based on standard birth-and-death process, the stationary probabilities of content n can be expressed as

$$\phi_n^R(R+l) = \left(\frac{\lambda_{n,k}}{\mu_{n,k}}\right)^l \frac{1}{\Pi_{j=1}^{l}(R+j)}\phi_n^R(R).$$

Since $\phi_n^R(R) + \phi_n^R(R+1) + \phi_n^R(R+2) + \cdots = 1$, we obtain (17). □

C. Proof of Proposition 3

Proof: Since the optimal policy for (15) is a threshold policy, for a given W_k, the optimal average cost under threshold R is $J(W_k) := \min_R\{J^R(W_k)\}$, where

$$J^R(W_k) := \sum_{S=0}^{\infty} L\phi_n^R(S) - W_k \sum_{S=0}^{R} \phi_n^R(S). \tag{32}$$

It is easy to show that $J^R(W_k)$ is a concave non-increasing function since it is a upper envelope of linear non-increasing functions in W_k, i.e., $J^R(W_k) < J^R(W_k')$ if $W_k > W_k'$. This means we can choose a larger threshold R when W_k increases to further increase the total cost according to (32), i.e, $R(W_k) \subseteq R(W_k')$ when $W_k < W_k'$. Next we show that $\sum_{S=0}^{R}\phi_i^R(S)$ is strictly increasing in R. From Proposition 2, we have

$$\sum_{S=0}^{R} \phi_n^R(S) = \phi_n^R(R) = \frac{1}{1 + \sum_{S=1}^{\infty}\left(\frac{\lambda_{n,k}}{\mu_{n,k}}\right)^S \prod_{m=0}^{S-1}\frac{1}{R+1+m}},$$

which is strictly increasing in R. □

D. Proof of Proposition 4

Proof: This follows from the definition of Whittle index that the performance of a policy with threshold R equals to the performance of a policy with threshold $R+1$ [62], [63], i.e.,

$$\mathbb{E}_R[S] - W\mathbb{E}_R[\mathbb{1}_{\{A_n(S)=0\}}] = \mathbb{E}_{R+1}[S] - W\mathbb{E}_{R+1}[\mathbb{1}_{\{A_n(S)=0\}}], \tag{33}$$

where a subscript denotes the fact that the associated quantities involve a threshold policy with the value of threshold equal to this value. Since the evolution of the n-th content is described by the transition kernel (a birth-and-death process) in (1), we have $\mathbb{E}_R[\mathbb{1}_{\{A_n(S)=0\}}] = \sum_{S=0}^{R}\phi_n^R(S)$. □

E. Proof of Theorem 1

Proof: To prove (19), it suffices to show that

$$\lim_{\rho\to\infty}\frac{1}{\rho}C(\pi_k^\star, \rho B_k^\star, \rho|\mathcal{N}_k|) \le \lim_{\rho\to\infty}\frac{1}{\rho}C(\pi_k^{opt}, \rho B_t^{opt}, \rho|\mathcal{N}_k|).$$

Let $D_{n,k}(s)$ be the average number of class-n contents under state s in frame k under the stationary policy $\pi_k^\star$. Following [66], the difference between the state distribution under the Whittle index policy $\pi_k^\star$ and the steady-state distribution under the optimal policy for the relaxed problem in (6) diminishes to zero, when $\rho|\mathcal{N}_k| \to \infty$ and $B_k^\star/|\mathcal{N}_k|$ is a constant, i.e., $\lim_{\rho\to\infty} D_{n,k}(s)/\rho = x_{n,k,s}, \forall n, k$. Therefore, we have

$$\begin{aligned}
&\lim_{\rho\to\infty}\frac{C(\pi_k^\star, \rho B_k^\star, \rho|\mathcal{N}_k|)}{\rho} \\
&= \lim_{\rho\to\infty}\frac{1}{\rho}\mathbb{E}_{\pi_k^\star}\left[\kappa c_d \limsup_{T_k\to\infty}\sum_{n=1}^{\rho|\mathcal{N}_k|}\frac{1}{T_k}\int_{t=1}^{T_k} S_n(k,t)dt\right] + (1-\kappa)c_b B_k^\star \\
&\overset{(a)}{=} \lim_{\rho\to\infty}\left[\kappa c_d \sum_{n=1}^{|\mathcal{N}_k|}\sum_{s\in\mathcal{S}}\frac{sD_{n,k}(s)}{\rho}\right] + (1-\kappa)c_b B_k^\star \\
&\overset{(b)}{=} \kappa c_d \sum_{n=1}^{|\mathcal{N}_k|}\sum_{s\in\mathcal{S}} s x_{n,k,s}^\star + (1-\kappa)c_b B_k^\star = C_{fluid}(B_k^\star, |\mathcal{N}_k|) \\
&\overset{(c)}{\le} \lim_{\rho\to\infty}\frac{C(\pi_n^{opt}, \rho B_k^{opt}, \rho|\mathcal{N}_k|)}{\rho},
\end{aligned}$$

where (a) follows from the definition of $D_{n,k}(s)$, (b) holds because of $\lim_{\rho\to\infty} D_{n,k}(s)/\rho = x_{n,k,s}$, and (c) is due to definition. □

APPENDIX B
PROOF OF THEOREM 2

Since there are two phases in *fW-UCB*, we decompose the regret in two parts as $\Delta(T_k) = \Delta(L_1) + \Delta(L_2)$, where $\Delta(L_1)$ is the regret for the planning phase with any random policy and $\Delta(L_2)$ is the regret for the policy execution phase, and $L_2 = T_k - L_1$.

A. The Regret of the Planning Phase

In the planning phase, each state-action pair $(s, 1)$ for each content is randomly sampled for $f(T_k)$ times. The performance gap between the genie-aided policy and the

random policy for each time is bounded because the cost is bounded, and hence we have

Lemma 1: Since the reward is bounded, the regret in the planning phase can be bounded by $\Delta(L_1) = \mathcal{O}\left(S_{max}|\mathcal{N}_k|f(T_k)\right)$.

Proof: The result follows the fact that there are $|\mathcal{N}_k|$ contents with a total $S_{\max}$ state-action pairs and to guarantee each state-action pair being sampled for $f(T_k)$ times. □

B. The Regret of the Policy Execution Phase

We next analyze the regret of the policy execution phase, i.e., $\Delta(L_2)$, which is defined as $\Delta(L_2) := L_2\gamma_k^{opt} - \mathbb{E}[C(\pi_k^\star, L_2)]$, which characterizes the accumulated cost gap when the true MDP employs the optimal policy π^{opt} and the learned policy $\pi_k^\star$, respectively. For the entire parameter space, two possible events can occur at the policy execution phase, which separates the regret into two disjoint parts. Next we bound these two parts separately.

The first event is called *the failure event*, which occurs when the true MDP $M_k := \{P_{n,k}, \forall n\}$ lies outside the plausible MDPs set $\mathcal{M}_k$ (see definition in (27)), and the second is *the good event* when true MDP M_k lies inside the plausible MDPs set $\mathcal{M}_k$. Therefore, the regret of the policy execution phase can be decomposed into two parts as follows

$$\Delta(L_2) = \Delta(L_2)\mathbb{1}(M_k \notin \mathcal{M}_k) + \Delta(L_2)\mathbb{1}(M_k \in \mathcal{M}_k).$$

1) Regret Conditioned on the Failure Event: Define the failure event as $\mathcal{E}_p := \{\exists s, n, |P_{n,k}(s'|s,1) - \bar{P}_{n,k}(s'|s,1)| > \delta_{n,k}^s\}$, which means that the true parameters are outside the confidence interval constructed in (25). The associated complementary event is denoted as $\mathcal{E}_p^c$. Therefore, we have the following relations: $\{M_k \notin \mathcal{M}_k\} := \mathcal{E}_p$, and $\{M_k \in \mathcal{M}_k\} := \mathcal{E}_p^c$. We now characterize the probability that the failure event occurs.

Lemma 2: With $\delta_{n,k}^s = \sqrt{\frac{s^2}{2f(T_k)}\log\left(S_{max}|\mathcal{N}_k|f(T_k)/\eta\right)}$, *we have*

$$\mathbb{P}(M_k \notin \mathcal{M}_k) \leq \tfrac{2\eta}{f(T_k)}.$$

Proof: By Chernoff-Hoeffding inequality [67], we have

$$\mathbb{P}\big(|P_{n,k}(s'|s,1) - \bar{P}_{n,k}(s'|s,1)| > \delta_{n,k}^s\big) \leq \frac{2\eta}{S_{\max}|\mathcal{N}_k|f(T_k)}.$$

By leveraging union bound over all states, actions, number of arms, we have $\mathbb{P}(M_k \notin \mathcal{M}_k) \leq \sum_{n=1}^{|\mathcal{N}_k|}\sum_s \mathbb{P}\big(|P_{n,k}(s'|s,1) - \bar{P}_{n,k}(s'|s,1)| > \delta_{n,k}^s\big) \leq 2\eta/f(T_k)$. □

Lemma 3: The regret conditioned on the failure event is

$$\Delta(L_2)\mathbb{1}(M_k \notin \mathcal{M}_k) = \mathcal{O}\Big(2|\mathcal{N}_k|L_2\eta/f(T_k)\Big).$$

Proof: From Lemma 2, we have

$$\begin{aligned}\Delta(L_2)\mathbb{1}(M_k \notin \mathcal{M}_k) &= \mathcal{O}(|\mathcal{N}_k|L_2\mathbb{1}(M_k \notin \mathcal{M}_k)) \\ &= \mathcal{O}(2|\mathcal{N}_k|L_2\eta/f(T_k)),\end{aligned}$$

where the first equality is due to that $\Delta(L_2)$ is bounded by $\mathcal{O}(|\mathcal{N}_k|L_2)$. □

2) Regret Conditioned on the Good Event: From Lemma 5, we have that the true MDP is inside the plausible MDPs set, i.e., $M_k \in \mathcal{M}_k$, with probability at least $1 - 2\eta/f(T_k)$. Now we consider the regret conditioned on the good event $M_k \in \mathcal{M}_k$. Define γ_k^{opt} as the optimal average cost achieved by the optimal policy π_k^{opt} and $\gamma_k^\star$ as optimistic average reward achieved by the learned policy $\pi_k^\star$ for the ture MDP M_k. Then

$$\Delta(L_2)\mathbb{1}(M_k \in \mathcal{M}_k) = L_2\gamma_k^\star - L_2\gamma_k^{opt}.$$

Before showing the regret conditioned on the good event, we first present a key lemma.

Lemma 4: (Optimism) Conditioned on the good event, there exists a transition $\tilde{P}_{n,k} \in \mathcal{P}_{n,k}, \forall n$ *such that*

$$\sum_{n=1}^{|\mathcal{N}_k|}\sum_{s\in\mathcal{S}}\phi_{n,k}^{opt}(s)s \geq \sum_{n=1}^{|\mathcal{N}_k|}\sum_{s\in\mathcal{S}}\tilde{\phi}_{n,k}(s)s,$$

where $\tilde{\phi}_{n,k}$ *is the stationary distribution derived from* $\{\tilde{P}_{n,k}, \forall n\}$.

Proof: The true $P_{n,k}$ is contained in $\mathcal{P}_{n,k}, \forall n$ for a good event. The result directly comes from the fact that confidence interval expands the feasibility region of original problem. □

Remark 3: Lemma 4 indicates that inside the plausible MDPs set $\mathcal{M}_k$, *there exists an MDP* $\tilde{M}_k$ *with parameters* $\{\tilde{P}_{n,k}, \forall n\}$ *achieving no more accumulated cost compared to the cost achieved by optimal policy for the true MDP* M_k.

For ease of expression, we denote the state for all contents as a stacked vector $\mathbf{s} \in \mathcal{S}^{|\mathcal{N}_k|\times 1} := [s_1, s_2, \ldots, s_{|\mathcal{N}_k|}]$, the corresponding actions under policy π as $\pi(\mathbf{s})$, and the unknown MDPs as $\Theta_k := [P_{1,k}, P_{2,k}, \ldots, P_{|\mathcal{N}_k|,k}]$. The transition kernel of the stacked system is then $P_\Theta(\cdot|\mathbf{s}, \pi(\mathbf{s})), \forall \mathbf{s} \in \mathcal{S}^{|\mathcal{N}_k|\times 1}$.

Let $\pi_{\Theta_k}^\star$ denote the proposed *Fluid Whittle* index policy corresponding to the transition model Θ_k and P_{Θ_k} be the controlled transition matrix under policy $\pi_{\Theta_k}^\star$. Denote $\gamma_{\Theta_k}^\star$ as the average reward of policy $\pi_{\Theta_k}^\star$, which satisfies the average reward Bellman equation [5],

$$\begin{aligned}\gamma_{\Theta_k}^\star + F_{\Theta_k}(\mathbf{s}) = R(\mathbf{s}, \pi_{\Theta_k}^\star(\mathbf{s})) \\ + [P_{\Theta_k}F_{\Theta_k}](\mathbf{s}), \forall \mathbf{s} \in \mathcal{S}^{|\mathcal{N}_k|\times 1},\end{aligned} \quad (34)$$

where $F_{\Theta_k}(\mathbf{s})$ is the relative value function.

Define γ_k^{opt} as the optimal average reward achieved by the optimal policy π_k^{opt} and $\gamma_k^\star$ as average reward achieved by the learned policy $\pi_k^\star$ for the true MDP M_k. Define $\tilde{\gamma}_k^\star$ as the optimistic average reward achieved by the learned policy $\tilde{\pi}_k^\star$ for the optimistic MDP $\tilde{M}_k$. Then we have the following lemma to upper bound the regret conditioned on good event, i.e., $M_k \in \mathcal{M}_k$.

Lemma 5: The regret for conditioned on the good event can be expressed as

$$\begin{aligned}&\Delta\ (L_2)\mathbb{1}(M_k \in \mathcal{M}_k) \\ &\leq \left[\sum_{t=1}^{L_2} F_{\tilde{\Theta}_k}(\mathbf{s}_{t+1}) - [P_{\tilde{\Theta}_k}F_{\tilde{\Theta}_k}](\mathbf{s}_t)\right] + c_1,\end{aligned}$$

where $\tilde{\Theta}_k$ *is the parameter of the optimistic MDP* $\tilde{M}_k$ *and* c_1 *is a constant.*

Proof: The proof goes as follows:

$$\begin{aligned}
\Delta(L_2)\mathbb{1}(M_k \in \mathcal{M}_k) &= \sum_{t=1}^{L_2} R(\mathbf{s}(t), \pi_k^\star(\mathbf{s}(t))) - L_2\gamma_k^{opt} \\
&= \sum_{t=1}^{L_2} R(\mathbf{s}(t), \pi_k^\star(\mathbf{s}(t))) - L_2\gamma_k^{opt} \\
&\quad - L_2\tilde{\gamma}_k^\star + L_2\tilde{\gamma}_k^\star \\
&\overset{(a)}{\leq} \sum_{t=1}^{L_2} R(\mathbf{s}(t), \pi_k^\star(\mathbf{s}(t))) - L_2\tilde{\gamma}_k^\star \\
&\overset{(b)}{=} -\sum_{t=1}^{L_2} R(\mathbf{s}(t), \tilde{\pi}_k^\star(\mathbf{s}(t))) \\
&\quad - \sum_{t=1}^{L_2} [P_{\tilde{\Theta}_k} F_{\tilde{\Theta}_k}](\mathbf{s}_t) + F_{\tilde{\Theta}_k}(\mathbf{s}_t) \\
&\quad + \sum_{t=1}^{L_2} R(\mathbf{s}(t), \pi_k^\star(\mathbf{s}(t))) \\
&\overset{(c)}{=} \sum_{t=1}^{L_2} -[P_{\tilde{\Theta}_k} F_{\tilde{\Theta}_k}](\mathbf{s}_t) + F_{\tilde{\Theta}_k}(\mathbf{s}_t) \\
&= \sum_{t=1}^{L_2} -[P_{\tilde{\Theta}_k} F_{\tilde{\Theta}_k}](\mathbf{s}_t) + F_{\tilde{\Theta}_k}(\mathbf{s}_{t+1}) \\
&\quad - F_{\tilde{\Theta}_k}(\mathbf{s}_{t+1}) + F_{\tilde{\Theta}_k}(\mathbf{s}_t) \\
&= \sum_{t=1}^{L_2} -[P_{\tilde{\Theta}_k} F_{\tilde{\Theta}_k}](\mathbf{s}_t) + F_{\tilde{\Theta}_k}(\mathbf{s}_{t+1}) \\
&\quad - F_{\tilde{\Theta}_k}(\mathbf{s}_{L_2+1}) + F_{\tilde{\Theta}_k}(\mathbf{s}_1) \\
&\overset{(d)}{\leq} \sum_{t=1}^{L_2} F_{\tilde{\Theta}_k}(\mathbf{s}_{t+1}) - [P_{\tilde{\Theta}_k} F_{\tilde{\Theta}_k}](\mathbf{s}_t) + c_1,
\end{aligned}$$

where (a) holds due to the fact the optimistic average reward $\tilde{\gamma}_k^\star$ for the optimistic MDPs $\tilde{\Theta}_k$ is no larger that the optimal average reward γ_k^{opt} for the true MDP according to Lemma 4; (b) directly follows from the Bellman equation in (34); (c) is due to the fact that $\sum_{t=1}^{L_2} R(\mathbf{s}(t), \tilde{\pi}_k^\star(\mathbf{s}(t))) = \sum_{t=1}^{L_2} R(\mathbf{s}(t), \pi_k^\star(\mathbf{s}(t)))$, and c_1 in (d) is a constant term. □

To bound the regret we present two key lemmas as follows.

Lemma 6 Azuma-Hoeffding inequality [73]: Let $X_1, X_2,$ be a martingale difference sequence with $|X_i| \leq c$ for all i. Then for all $\epsilon > 0$

$$\mathbb{P}\left(\sum_{i=1}^{n} X_i > \epsilon\right) \leq \exp\left(-\frac{\epsilon^2}{2nc^2}\right).$$

Lemma 7 [74]: For any Θ, we have that $0 \leq \gamma_\Theta \leq S_{max} B_k^\star$ and span$(F_\Theta) \leq c_2\gamma_\Theta$, with c_2 being a constant value related with the ergodicity coefficient.

We are now ready to characterize the regret conditioned on the good event.

Lemma 8: Conditioned on the good event, the regret is given by

$$\Delta(L_2)\mathbb{1}(M_k \in \mathcal{M}_k) = \mathcal{O}\left(S_{max} B_k^\star |\mathcal{N}_k| \sqrt{T_k \log T_k}\right),$$

with a probability larger than $1 - \frac{1}{T_k^2}$.

Proof: Define $X_t = \mathbb{E}\left[[P_{\tilde{\Theta}_k} F_{\tilde{\Theta}_k}](\mathbf{s}_{t+1}) - F_{\tilde{\Theta}_k}(\mathbf{s}_t)\right]$. We have

$$\Delta(L_2)\mathbb{1}(M_k \in \mathcal{M}_k) \leq \textstyle\sum_{t=1}^{L_2} X_t + c_1.$$

Due to the fact that

$$\begin{aligned}
&\mathbb{E}\left[[P_{\tilde{\Theta}_k} F_{\tilde{\Theta}_k}](\mathbf{s}_{t+1}) - F_{\tilde{\Theta}_k}(\mathbf{s}_t)\right] \\
&\quad \leq c_2 S_{\max} B_k^\star \sum_n \mathbb{E}[|P_n(s'|s,a) - \tilde{P}_n(s'|s,a)|_1] \\
&\quad \leq c_2 S_{\max} B_k^\star |\mathcal{N}_k|,
\end{aligned}$$

we have

$$|X_t| \leq c_2 S_{\max} B_k^\star |\mathcal{N}_k|.$$

Since $\mathbb{E}[X_t|\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_t, \mathbf{a}_t] = 0$, X_t is a sequence of martingale difference due to the Bellman equation in (34). Applying Azuma-Hoeffding inequality yields,

$$\mathbb{P}\left(\sum_t X_t \geq c_2 S_{\max} B_k^\star |\mathcal{N}_k| \sqrt{4T_k \log T_k}\right) \leq \left(\frac{1}{T_k}\right)^2.$$

Hence, conditioned on good event we have

$$\Delta(L_2)\mathbb{1}(M_k \in \mathcal{M}_k) = \mathcal{O}\left(S_{\max} B_k^\star |\mathcal{N}_k| \sqrt{T_k \log T_k}\right).$$

□

3) Total Regret: Combining Lemma 1, 3 and 8 and let $f(T_k) = \sqrt{T_k}$, the total regret is given by

$$\begin{aligned}
\Delta(T_k) &= \Delta(L_1) + \Delta(L_2) \\
&= \mathcal{O}((1 + B_k^\star \sqrt{\log T_k}) S_{\max} |\mathcal{N}_k| \sqrt{T_k}).
\end{aligned}$$

REFERENCES

[1] G. Xiong, S. Wang, G. Yan, and J. Li, "Reinforcement learning for dynamic dimensioning of cloud caches: A restless bandit approach," in *Proc. IEEE INFOCOM*, May 2022, pp. 2108–2117.

[2] G. Forecast et al., "Cisco visual networking index: Global mobile data traffic forecast update, 2017–2022," Cisco, San Jose, CA, USA, 2019.

[3] *Amazon AWS*. Accessed: May 2021. [Online]. Available: https://aws.amazon.com/

[4] *Amazon Cloudfront*. Accessed: May 2021. [Online]. Available: https://aws.amazon.com/

[5] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Hoboken, NJ, USA: Wiley, 1994.

[6] P. Whittle, "Restless bandits: Activity allocation in a changing world," *J. Appl. Probab.*, vol. 25, no. A, pp. 287–298, 1988.

[7] J. Gittins, K. Glazebrook, and R. Weber, *Multi-Armed Bandit Allocation Indices*. Hoboken, NJ, USA: Wiley, 2011.

[8] I. M. Verloop, "Asymptotically optimal priority policies for indexable and nonindexable restless bandits," *Ann. Appl. Probab.*, vol. 26, no. 4, pp. 1947–1995, Aug. 2016.

[9] J. Niño-Mora, "Dynamic priority allocation via restless bandit marginal productivity indices," *Top*, vol. 15, no. 2, pp. 161–198, 2007.

[10] T. Jaksch, R. Ortner, and P. Auer, "Near-optimal regret bounds for reinforcement learning," *J. Mach. Learn. Res.*, vol. 11, no. 51, pp. 1563–1600, 2010.

[11] A. Gopalan and S. Mannor, "Thompson sampling for learning parameterized Markov decision processes," in *Proc. 28th Conf. Learn. Theory*, 2015, pp. 861–898.

[12] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2, pp. 235–256, 2002.

[13] *Redis*. Accessed: May 2021. [Online]. Available: https://redis.io/

[14] *Amazon Web Service ElastiCache*. Accessed: May 2021. [Online]. Available: https://aws.amazon.com/elasticache/

[15] J.-C. Bolot and P. Hoschka, "Performance engineering of the world wide web: Application to dimensioning and cache design," *Comput. Netw. ISDN Syst.*, vol. 28, nos. 7–11, pp. 1397–1405, May 1996.

[16] J. Sahoo et al., "A survey on replica server placement algorithms for content delivery networks," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 1002–1026, 2nd Quart., 2017.

[17] G. S. Paschos, G. Iosifidis, M. Tao, D. Towsley, and G. Caire, "The role of caching in future communication systems and networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1111–1125, Jun. 2018.

[18] G. Iosifidis, J. Kwak, and G. Paschos, "19 Economic ecosystems in elastic wireless edge caching," in *Wireless Edge Caching: Modeling, Analysis, and Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2021, p. 387.

[19] D. Carra, G. Neglia, and P. Michiardi, "Elastic provisioning of cloud caches: A cost-aware TTL approach," *IEEE/ACM Trans. Netw.*, vol. 28, no. 3, pp. 1283–1296, Jun. 2020.

[20] J. Kwak, G. Paschos, and G. Iosifidis, "Dynamic cache rental and content caching in elastic wireless CDNs," in *Proc. 16th Int. Symp. Model. Optim. Mobile, Ad Hoc, Wireless Netw. (WiOpt)*, 2018, pp. 1–8.

[21] A. Cidon, A. Eisenman, M. Alizadeh, and S. Katti, "Dynacache: Dynamic cloud caching," in *Proc. USENIX HotCloud*, 2015, pp. 1–27.

[22] K. Liu and Q. Zhao, "Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access," *IEEE Trans. Inf. Theory*, vol. 56, no. 11, pp. 5547–5567, Nov. 2010.

[23] W. Dai, Y. Gai, B. Krishnamachari, and Q. Zhao, "The non-Bayesian restless multi-armed bandit: A case of near-logarithmic regret," in *Proc. IEEE Int. Conf. Acoustics, Speech Signal Process. (ICASSP)*, May 2011, pp. 2940–2943.

[24] S. P. Sheng, M. Liu, and R. Saigal, "Data-driven channel modeling using spectrum measurement," *IEEE Trans. Mobile Comput.*, vol. 14, no. 9, pp. 1794–1805, Sep. 2015.

[25] K. Avrachenkov, U. Ayesta, J. Doncel, and P. Jacko, "Congestion control of TCP flows in internet routers by means of index policy," *Comput. Netw.*, vol. 57, no. 17, pp. 3463–3478, Dec. 2013.

[26] A. Mahajan and D. Teneketzis, "Multi-armed bandit problems," in *Foundations and Applications of Sensor Management*. Berlin, Germany: Springer, 2008, pp. 121–151.

[27] S. H. A. Ahmad, M. Liu, T. Javidi, Q. Zhao, and B. Krishnamachari, "Optimality of myopic sensing in multichannel opportunistic access," *IEEE Trans. Inf. Theory*, vol. 55, no. 9, pp. 4040–4050, Sep. 2009.

[28] V. S. Borkar, K. Ravikumar, and K. Saboo, "An index policy for dynamic pricing in cloud computing under price commitments," *Applicationes Mathematicae*, vol. 44, pp. 215–245, Jan. 2017.

[29] E. Lee, M. S. Lavieri, and M. Volk, "Optimal screening for hepatocellular carcinoma: A restless bandit model," *Manuf. Service Operations Manage.*, vol. 21, no. 1, pp. 198–212, Jan. 2019.

[30] A. Mate, A. Perrault, and M. Tambe, "Risk-aware interventions in public health: Planning with restless multi-armed bandits," in *Proc. AAMAS*, 2021, pp. 880–888.

[31] J. A. Killian, A. Perrault, and M. Tambe, "Beyond 'to act or not to act': Fast Lagrangian approaches to general multi-action restless bandits," in *Proc. AAMAS*, 2021, pp. 710–718.

[32] T. W. Archibald, D. P. Black, and K. D. Glazebrook, "Indexability and index heuristics for a simple class of inventory routing problems," *Operations Res.*, vol. 57, no. 2, pp. 314–326, Apr. 2009.

[33] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of optimal queueing network control," in *Proc. IEEE Conf. Struct. Complex. Theory*, Jun. 1994, pp. 318–322.

[34] J. Niño-Mora, "Restless bandits, partial conservation laws and indexability," *Adv. Appl. Probab.*, vol. 33, no. 1, pp. 76–98, Mar. 2001.

[35] D. Bertsimas and J. Niño-Mora, "Restless bandits, linear programming relaxations, and a primal-dual index heuristic," *Operations Res.*, vol. 48, no. 1, pp. 80–90, Feb. 2000.

[36] W. Hu and P. Frazier, "An asymptotically optimal index policy for finite-horizon restless bandits," 2017, *arXiv:1707.00205*.

[37] G. Zayas-Cabán, S. Jasin, and G. Wang, "An asymptotically optimal heuristic for general nonstationary finite-horizon restless multi-armed, multi-action bandits," *Adv. Appl. Probab.*, vol. 51, no. 3, pp. 745–772, Sep. 2019.

[38] H. Liu, K. Liu, and Q. Zhao, "Logarithmic weak regret of non-Bayesian restless multi-armed bandit," in *Proc. IEEE ICASSP*, May 2011, pp. 1968–1971.

[39] H. Liu, K. Liu, and Q. Zhao, "Learning in a changing world: Restless multiarmed bandit with unknown dynamics," *IEEE Trans. Inf. Theory*, vol. 59, no. 3, pp. 1902–1916, Mar. 2013.

[40] C. Tekin and M. Liu, "Online learning of rested and restless bandits," *IEEE Trans. Inf. Theory*, vol. 58, no. 8, pp. 5588–5611, Aug. 2012.

[41] R. Ortner, D. Ryabko, P. Auer, and R. Munos, "Regret bounds for restless Markov bandits," in *Proc. Algorithmic Learn. Theory*, 2012, pp. 214–228.

[42] Y. H. Jung and A. Tewari, "Regret bounds for Thompson sampling in episodic restless bandit problems," in *Proc. NeurIPS*, 2019, pp. 1–10.

[43] Y. Hun Jung, M. Abeille, and A. Tewari, "Thompson sampling in non-episodic restless bandits," 2019, *arXiv:1910.05654*.

[44] C. Tekin and M. Liu, "Adaptive learning of uncontrolled restless bandits with logarithmic regret," in *Proc. 49th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, 2011, pp. 983–990.

[45] V. S. Borkar and K. Chadha, "A reinforcement learning algorithm for restless bandits," in *Proc. Indian Control Conf. (ICC)*, Jan. 2018, pp. 89–94.

[46] K. Avrachenkov and V. S. Borkar, "A learning algorithm for the whittle index policy for scheduling web crawlers," in *Proc. 57th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, 2019, pp. 1001–1006.

[47] J. Fu, Y. Nazarathy, S. Moka, and P. G. Taylor, "Towards Q-learning the whittle index for restless bandits," in *Proc. Austral. New Zealand Control Conf.*, 2019, pp. 249–254.

[48] S. Wang, L. Huang, and J. Lui, "Restless-UCB, an efficient and low-complexity algorithm for online restless bandits," in *Proc. NeurIPS*, 2020, pp. 1–12.

[49] J. A. Killian, A. Biswas, S. Shah, and M. Tambe, "Q-learning Lagrange policies for multi-action restless bandits," in *Proc. ACM SIGKDD*, 2021, pp. 871–881.

[50] A. Biswas, G. Aggarwal, P. Varakantham, and M. Tambe, "Learning index policies for restless bandits with application to maternal healthcare," in *Proc. AAMAS*, 2021, pp. 1–2.

[51] K. Nakhleh et al., "NeurWIN: Neural whittle index network for restless bandits via deep RL," in *Proc. NeurIPS*, 2021, pp. 1–12.

[52] G. Xiong, J. Li, and R. Singh, "Reinforcement learning augmented asymptotically optimal index policy for finite-horizon restless bandits," in *Proc. AAAI Conf. Artif. Intell.*, Jun. 2022, vol. 36, no. 8, pp. 8726–8734.

[53] G. Xiong, X. Qin, B. Li, R. Singh, and J. Li, "Index-aware reinforcement learning for adaptive video streaming at the wireless edge," in *Proc. ACM MobiHoc*, 2022, pp. 81–90.

[54] V. Martina, M. Garetto, and E. Leonardi, "A unified approach to the performance analysis of caching systems," in *Proc. IEEE INFOCOM*, Apr. 2014, pp. 2040–2048.

[55] S. Ioannidis and E. Yeh, "Adaptive caching networks with optimality guarantees," in *Proc. ACM SIGMETRICS*, 2016, pp. 113–124.

[56] J. Li et al., "DR-cache: Distributed resilient caching with latency guarantees," in *Proc. IEEE INFOCOM*, Apr. 2018, pp. 441–449.

[57] M. Mahdian, A. Moharrer, S. Ioannidis, and E. Yeh, "Kelly cache networks," in *Proc. IEEE INFOCOM*, Jan. 2019, pp. 1130–1143.

[58] N. K. Panigrahy, J. Li, D. Towsley, and C. V. Hollot, "Network cache design under stationary requests: Exact analysis and Poisson approximation," *Comput. Netw.*, vol. 180, Oct. 2020, Art. no. 107379.

[59] Y. Li and S. Ioannidis, "Universally stable cache networks," in *Proc. IEEE INFOCOM*, Jul. 2020, pp. 546–555.

[60] F. Baccelli and P. Brémaud, *Elements of Queueing Theory: Palm Martingale Calculus and Stochastic Recurrences*, vol. 26. Berlin, Germany: Springer, 2013.

[61] A. Wierman, L. L. Andrew, and A. Tang, "Power-aware speed scaling in processor sharing systems," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 2007–2015.

[62] M. Larra naga, U. Ayesta, and I. M. Verloop, "Index policies for a multi-class queue with convex holding cost and abandonments," in *Proc. ACM Sigmetrics*, 2014, pp. 1–13.

[63] M. Larrnaaga, U. Ayesta, and I. M. Verloop, "Dynamic control of birth-and-death restless bandits: Application to resource-allocation problems," *IEEE/ACM Trans. Netw.*, vol. 24, no. 6, pp. 3812–3825, Dec. 2016.

[64] Y.-P. Hsu, "Age of information: Whittle index for scheduling stochastic arrivals," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 2634–2638.

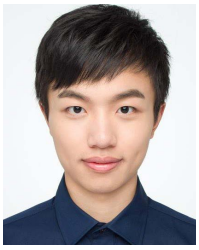
[65] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[66] R. R. Weber and G. Weiss, "On an index policy for restless bandits," *J. Appl. Probab.*, vol. 27, no. 3, pp. 637–648, Sep. 1990.

[67] W. Hoeffding, "Probability inequalities for sums of bounded random variables," in *The Collected Works of Wassily Hoeffding*. Berlin, Germany: Springer, 1994, pp. 409–426.

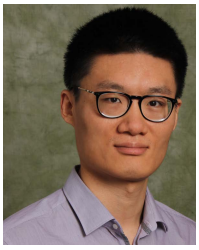
[68] C. Jin, T. Jin, H. Luo, S. Sra, and T. Yu, "Learning adversarial MDPs with bandit feedback and unknown transition," 2019, *arXiv:1912.01192*.
[69] A. Rosenberg and Y. Mansour, "Online convex optimization in adversarial Markov decision processes," in *Proc. ICML*, 2019, pp. 5478–5486.
[70] K. C. Kalagarla, R. Jain, and P. Nuzzo, "A sample-efficient algorithm for episodic finite-horizon MDP with constraints," in *Proc. AAAI*, 2021, pp. 8030–8037.
[71] G. Ma, Z. Wang, M. Zhang, J. Ye, M. Chen, and W. Zhu, "Understanding performance of edge content caching for mobile video streaming," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1076–1089, May 2017.
[72] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Watch global, cache local: YouTube network traffic at a campus network: Measurements and implications," in *Proc. SPIE, Multimedia Comput. Netw.*, San Jose, CA, USA, vol. 6818, Jan. 2008, Art. no. 681805.
[73] B. Bercu, B. Delyon, and E. Rio, *Concentration Inequalities for Sums and Martingales*. Berlin, Germany: Springer, 2015.
[74] N. Akbarzadeh and A. Mahajan, "On learning whittle index policy for restless bandits with scalable regret," 2022, *arXiv:2202.03463*.

Guojun Xiong (Graduate Student Member, IEEE) received the B.S. degree in information science and technology from the Sun Yat-sen University in 2015 and the M.S. degree in electrical engineering and computer science from the University of Kansas in 2020. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Binghamton University, State University of New York. His research interests include wireless communication, networking, optimization and control, and reinforcement learning.



Shufan Wang received the B.S. degree from Nanjing University in June 2017 and the M.S. degree from Binghamton University in September 2022, where he is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering. His research interests include online algorithms in large-scale networked systems and reinforcement learning.

Gang Yan received the B.S. degree from Nankai University in June 2016, where he received the M.S. degree in statistics in June 2019. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Binghamton University, State University of New York. His research interests include federated learning, security and defense in distributed systems, and caching and content delivery.



Jian Li (Member, IEEE) received the B.E. degree from Shanghai Jiao Tong University in June 2012 and the Ph.D. degree in computer engineering from Texas A&M University in December 2016. He was a Post-Doctoral Researcher at the College of Information and Computer Sciences, University of Massachusetts Amherst, from January 2017 to August 2019. He is currently an Assistant Professor of computer engineering with the Department of Electrical and Computer Engineering, Binghamton University, State University of New York. His current research interests include intersection of algorithms for reinforcement learning, federated/distributed learning, and stochastic optimization and control, with applications to next-generation networked systems.