# When Lyapunov Drift Based Queue Scheduling Meets Adversarial Bandit Learning

Jiatai Huang, Leana Golubchik, *Senior Member, IEEE*, and Longbo Huang, *Senior Member, IEEE*

***Abstract*— In this paper, we study scheduling of a queueing system with zero knowledge of instantaneous network conditions. We consider a one-hop single-server queueing system consisting of $K$ queues, each with time-varying and non-stationary arrival and service rates. Our scheduling approach builds on an innovative combination of adversarial bandit learning and Lyapunov drift minimization, without knowledge of the instantaneous network state (the arrival and service rates) of each queue. We then present two novel algorithms `SoftMW` (SoftMaxWeight) and `SSMW` (Sliding-window SoftMaxWeight), both capable of stabilizing systems that can be stabilized by some (possibly unknown) sequence of randomized policies whose time-variation satisfies a mild condition. We further generalize our results to the setting where arrivals and departures only have bounded moments instead of being deterministically bounded and propose `SoftMW+` and `SSMW+` that are capable of stabilizing the system. As a building block of our new algorithms, we also extend the classical `EXP3.S` algorithm for multi-armed bandits to handle unboundedly large feedback signals, which can be of independent interest.**

***Index Terms*— Scheduling, queueing, bandit learning, Lyapunov analysis.**

## I. Introduction

STOCHASTIC network scheduling is concerned with a fundamental problem of allocating resources to serving demand in dynamic environments, and it has found wide applicability in modeling real-world networked systems, including data communication [2], [3], cloud computing and server farms [4], [5], [6], [7], smart grid management [8], [9], [10], supply chain management [11], [12], and control of transportation networks [13], [14], [15]. One basic requirement of most existing scheduling solutions is having knowledge of the instantaneous network state – i.e., the amount of arrival traffic and the amount of service under any feasible control action, e.g., the power allocation among all links – before taking a new scheduling action. Given this information, there have been many successful network scheduling algorithms, with various aspects of theoretical performance guarantees, including queue stability [16], [17], [18], delays [19], [20], [21], and utilities [21], [22], [23].

However, in many real-world scenarios, such network-state knowledge may not always be available if its measurement or estimation is too difficult or costly to obtain. Even when such knowledge is available, it can be biased and imperfect. For instance, in an IoT system, due to sensors' temperature-drift or device malfunction, unexpected changes in traffic and channel patterns can occur at any time [24]. In an underwater communication system, it is extremely challenging to perform perfect channel state estimation [25]. Moreover, in applications where the communicating parties can move rapidly, e.g., self-driving vehicles [26], or in an arbitrary manner, e.g., wireless AR/VR devices [27], channel conditions can also change rapidly and thus difficult to estimate accurately. Therefore, scheduling policies relying on precise network-state knowledge may not be applicable to many real-world tasks; relying on such policies can result in significant performance degradation due to inaccurate information. Hence, network scheduling *without* instantaneous knowledge and accurate estimation of the network state is important both, in theory and in practice, i.e., it can significantly improve robustness and availability of large-scale networked systems while reducing operational and maintenance costs.

To this end, in this paper, we focus on a novel *scheduling without network-state knowledge* formulation. Specifically, we focus on a one-hop scheduling task, where a single-server serves $K$ queues, each corresponding to a job type. The server chooses a single queue to serve in each time slot. The network dynamics, i.e., arrival and service rates, evolve in an *oblivious adversarial* manner and are unknown before the scheduling decision. Moreover, the service outcome is only observed after the action with bandit feedback, i.e., only the served queue produces an observation. Our goal is to seek an efficient scheduling policy to stabilize the network. It turns out that in such systems which have time-varying network dynamics, many attractive properties of classical scheduling policies for stationary systems no longer hold. For example, different work-conserving policies may induce different busy time period distributions. Therefore, queue stability in this setting is a fundamental and a non-trivial problem, and is an important focus of our work.

To solve this problem, we introduce novel learning-augmented scheduling algorithms, inspired by the celebrated `MaxWeight` queue scheduling algorithm [28] and the success

Manuscript received 15 August 2023; revised 15 January 2024; accepted 27 February 2024; approved by IEEE/ACM Transactions on Networking Editor B. Ji. Date of publication 26 March 2024; date of current version 20 August 2024. The work of Jiatai Huang and Longbo Huang was supported by the Technology and Innovation Major Project of the Ministry of Science and Technology of China under Grant 2020AAA0108400 and Grant 2020AAA0108403. The work of Leana Golubchik was supported in part by NSF under Award CNS-1816887. *(Corresponding author: Longbo Huang.)*

Jiatai Huang and Longbo Huang are with the Interdisciplinary Information Sciences (IIIS), Tsinghua University, Beijing 100084, China (e-mail: hjt18@mails.tsinghua.edu.cn; longbohuang@tsinghua.edu.cn).

Leana Golubchik is with the Department of Computer Science, University of Southern California, Los Angeles, CA 90089 USA (e-mail: leana@usc.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TNET.2024.3374755, provided by the authors.

Digital Object Identifier 10.1109/TNET.2024.3374755

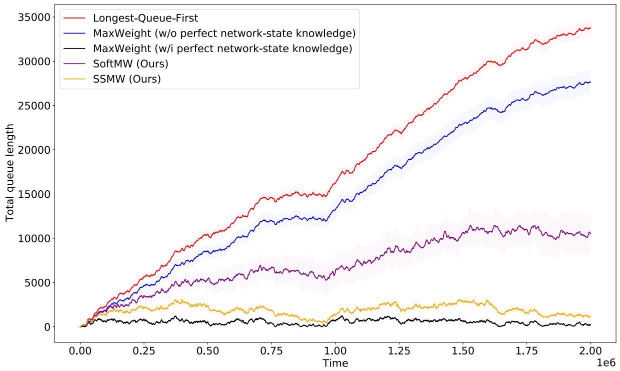


Fig. 1. Numerical evaluation of a non-stationary system (see Appendix A for details).

of the EXP3 family of algorithms on non-stationary Multi-Armed Bandits (MAB) problems [1]. The proposed algorithms are capable of stabilizing a non-stationary system, as long as the system can be stabilized by a randomized policy whose total variation of probabilities to serve each type of job is not too large. Perhaps surprisingly, our algorithms rely on neither knowing the network statistics before-hand, nor on complicated explicit real-time estimation of the system. As a result, compared to its network-state knowledge dependent counterparts, our algorithms are naturally more robust to jitter and unexpected traffic/service patterns in the system. Indeed, a numerical comparison of our algorithms with their accurate knowledge dependent counterparts shows that the presented algorithms do give superior performance for systems with service state noise, as depicted in Figure 1. In particular, our proposed algorithms are capable of stabilizing complex time-varying systems with unknown network states in contrast to popular policies, such as MaxWeight and Longest-Queue-First, which fail to control the queue lengths well. The detail of the numerical experiment is presented in Appendix A.

Our work differs from the existing learning-augmented network control literature, e.g., [29], [30], [31], [32], and [33], in the following aspects. References [29], [30], [31], and [32] study the scheduling or load-balancing tasks on stationary systems with rate statistics unknown before-hand, while in our setting the system can be time-varying and adversarial. Reference [33] also considers non-stationary systems, but they assume smoothly time-varying service rates and explicitly estimate the instantaneous service rates using exponential average and discounted UCB bonus. Compared to these works, our approach requires neither to explicitly optimize off-line problems nor to explicitly probe and estimate the instantaneous channel states, but rather uses adversarial bandit learning techniques to coherently explore and stabilize the system at the same time.

On the technical side, utilizing adversarial MAB algorithms in stochastic network scheduling and obtaining provable stability guarantees is non-trivial. Firstly, it requires transforming the scheduling problem into an equivalent adversarial bandit problem, where the key is to properly specify the corresponding queue-dependent rewards and the overall objective. Secondly, the analysis requires extending the adversarial bandit algorithms to handle the potentially unbounded reward due to queue sizes as well as establishing a connection between regret analysis and the queue stability result. There also exist recent works that utilize reinforcement learning (RL) for queue scheduling, e.g., [34] and [35]. However, results there typically rely on learning the unknown stationary distribution. When the environment is adversarial, information learned from past history does not form a good estimator for future dynamics. In this case, how to design RL algorithms with rigorous performance guarantees remains a challenging task.

Our contributions in this work can be summarized as follows:

- We propose two novel scheduling algorithms SoftMW (Algorithm 2) and SSMW (Sliding-window SoftMW, Algorithm 3) that are capable of scheduling one-hop queueing systems *without channel state knowledge*, while *stablizing the systems* under mild conditions on the time-variation of the reference randomized policy.
- In designing these two algorithms, we carefully combine techniques from online bandit learning and Lyapunov drift based scheduling approaches and analysis. Specifically, the bandit part is used to guarantee that our algorithms' "regret" against an unknown time-varying randomized policy over a finite time-horizon is small. This regret guarantee is coupled (in an innovative manner) with Lyapunov drift analysis to develop the stability result (see Section V-C and Appendix F).
- We extend the EXP3.S algorithm [1], originally designed for adversarial MAB problems with bounded rewards, such that time-varying learning rates and exploration rates are applicable to handling unboundedly large feedback (see Section V-A, Algorithm 1). This extended EXP3.S algorithm (termed EXP3.S+) is used as a building block in SoftMW and SSMW. However, it is also of independent interest beyond the scope of queueing problems.
- We further generalize our results to the setting where arrivals and departures have bounded moments instead of being deterministically bounded (see Appendix H) and present SoftMW+ (Algorithm 4) and SSMW+ (Algorithm 5) that are capable of stabilizing such a system.

Table I provides a comparison summary between our proposed algorithms and closely related efforts. Theoretical results on system stability and average queue length bounds in zero-knowledge network systems are still largely open. To our knowledge, our work is the *first* to utilize adversarial MAB algorithms with dynamic regret guarantees in queueing systems scheduling, and is capable of providing satisfactory average queue length bounds (and thus provable stability) under very mild assumptions. Most prior work is based on epsilon-greedy or Upper Confidence Bounds (UCB), where the assumption is needed that the system is either stationary or non-stationary but with arrival (departure) rates having adequate smoothness. Hence, our algorithms apply to *more general and complex settings*. We believe our approach can facilitate novel and interesting insights into MaxWeight-type as well as other queueing scheduling algorithm design problems. We also note that our proposed average queue length bounds results can lead to fruitful, non-trivial delay bounds when certain mild technical conditions hold. For instance, when arrival rates are universally bounded above some $\zeta > 0$, then `SoftMW` and `SoftMW+` guarantee $\mathcal{O}\left(\frac{C_W K M^2}{\epsilon\zeta}\right)$ queueing delays.

## II. Notation

Throughout this paper, for $n \geq 1$, we denote the set $\{1, 2, \ldots, n\}$ by $[n]$ and the $(n-1)$-dimensional probability simplex over $[n]$ by $\triangle^{[n]}$. We use bold English letters (e.g., $\mathbf{Q}_t$, $\mathbf{S}_t$) and Greek letters with arrows above (e.g., $\vec{\sigma}_t$, $\vec{\lambda}_t$) to denote vector-valued variables. We use $\mathbf{0}$ to denote the all-zero vector, and $\mathbf{1}$ to denote the all-one vector. We use $\mathbf{1}_i$ to denote the one-hot vector with 1 on the $i$-th coordinate, i.e., $(\mathbf{1}_i)_j = 1$ if $i = j$ and 0 otherwise. We use $\mathbb{1}[\text{statement}]$ to denote the indicator of a given statement; its value is taken as 1 if the statement holds and 0 otherwise. We use $\mathbf{x} \odot \mathbf{y}$ to denote the element-wise product of two vectors $\mathbf{x}$ and $\mathbf{y}$.

Let $f$ be a strictly convex function defined on some convex domain $A \subseteq \mathbb{R}^K$. For any $\mathbf{x}, \mathbf{y} \in A$, if $\nabla f(\mathbf{x})$ exists, we write the Bregman divergence between $y$ and $x$ induced by $f$ as

$$D_f(\mathbf{y}, \mathbf{x}) \triangleq f(\mathbf{y}) - f(\mathbf{x}) - \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle$$

We use $f^*(\mathbf{y}) \triangleq \sup_{\mathbf{x} \in \mathbb{R}^K} \{\langle \mathbf{y}, \mathbf{x} \rangle - f(\mathbf{x})\}$ to denote the convex conjugate of $f$.

We use $\widetilde{\mathcal{O}}$, $\widetilde{\Omega}$ or $\widetilde{\Theta}$ to suppress poly-logarithmic factors in $T$ (the length of the decision horizon) and $K$ (the number of queues). Unless stated otherwise, we use

$$\mathcal{F}_t = \sigma\left(a_1, \ldots, a_t, \mathbf{Q}_0, \ldots, \mathbf{Q}_t, \mathbf{A}_1, \ldots, \mathbf{A}_t, S_{1,a_1}, \ldots, S_{t,a_t}\right)$$

for any $t \geq 0$ to denote the filtration of $\sigma$-algebra when studying random quantities indexed by time, i.e., $\mathcal{F}_t$ is generated by all decisions and quantities visible to a scheduling policy at the end of $t$-th time slot.

## III. Problem Setting

We consider the problem of scheduling $K$ job types on a single work-conserving server with a slotted time system. Each arriving job first joins a queue associated with its type $i$, which we denote by $Q_i$. Denote by $A_{t,i}$ the amount of arriving jobs of type $i$ in the $t$-th time slot, and by $S_{t,i}$ the maximum amount of jobs of type $i$ the server can serve in the $t$-th time slot. At the beginning of each time slot $t$, the server chooses *exactly one* type of a job $a_t \in [K]$ to serve. Denote by $Q_{t,i}$ the queue length of type $i$ jobs at the end of time slot $t$. Then, each $Q_{t,i}$ evolves according to the following equation:

$$Q_{t,i} = \max\left\{Q_{t-1,i} + A_{t,i} - S_{t,i}\mathbb{1}[i = a_t], 0\right\}$$

where $\mathbf{Q}_0 = (Q_{0,1}, \ldots, Q_{0,K}) = \mathbf{0}$. At the beginning of each time slot $t$, the latest queue lengths $Q_{t-1,1}, \ldots, Q_{t-1,K}$ are available to the server for making new decisions. The maximum service amount of past actions $S_{0,a_0}, \ldots, S_{t-1,a_t}$ are also visible to the server.

We assume that there are two sequences of distributions $\{\mathcal{A}_1, \mathcal{A}_2, \ldots\}$ and $\{\mathcal{S}_1, \mathcal{S}_2, \ldots\}$, all fixed before the queue process starts, and their statistics are unknown to the scheduler before-hand. All distributions $\mathcal{A}_t$s and $\mathcal{S}_t$s are supported on $[0, M]^K$, where $M$ is a constant known before-hand. We further assume that each $\mathbf{A}_t$ is randomly sampled from $\mathcal{A}_t$, each $\mathbf{S}_t$ is sampled from $\mathcal{S}_t$, and all $\mathbf{A}_t$s and $\mathbf{S}_t$s are independent random vectors. We denote by $\vec{\lambda}_t$ the mean of $\mathcal{A}_t$, and by $\vec{\sigma}_t$ the mean of $\mathcal{S}_t$.

Our objective is to design a scheduling policy, such that the average expected queue lengths

$$\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i=1}^{K}\mathbb{E}[Q_{t,i}] \tag{1}$$

on any finite time-horizon of sufficiently large length $T$ is well-controlled. Building an upper-bound for the average queue length in Eq. (1) is one of the core problem in network optimization, it is closely related to other important network performance metrics (e.g., the delay bound can be implied via Little's law). Conventionally, we say a scheduling policy *stabilizes* the system, or the system is *stable* under some scheduling policy, if the average queue length in Eq. (1) is uniformly bounded by some finite number for all $T \geq 1$.

Classical scheduling tasks on stationary systems (e.g., [30], [31]) correspond to the case where $\mathcal{A}_t = \mathcal{A}_1$ ($\mathcal{S}_t = \mathcal{S}_1$), i.e., the distributions are time-invariant in our setting. In this paper, we consider general environments where the network state information is unknown [33], [37], [38], the arrival and service rates can also be time-varying [33]. This is an important setting in robust scheduling algorithm design, and building average queue length bounds in this case is still largely open. Intuitively, to schedule such systems well, one needs to explore and estimate the time-varying service distributions subject to queue stability, which is much more complicated than stationary systems.

## IV. A Sufficient Condition for Stabilizing the System

In our paper, we make the following assumption on the system, which is analogous to the capacity region definition in stationary network scheduling [28], and can be viewed as a generalized stability condition for scheduling in adversarial environments.

*Assumption 1 (Piecewise Stabilizability): There exist $C_W \geq 0$, $\epsilon > 0$, $\vec{\theta}_1, \vec{\theta}_2, \cdots \in \Delta^{[K]}$ and a partition of $\mathbb{N}_+$ into intervals $W_0, W_1, \cdots$, such that for any $T \geq 1$ we have*

$$\sum_{i:\min_{t\in W_i} t<T} (|W_i| - 1)^2 \leq C_W T \tag{2}$$

*and for any $i \geq 0$ and $j \in [K]$ we have*

$$\frac{1}{|W_i|}\sum_{t\in W_i}\theta_{t,j}\sigma_{t,j} \geq \epsilon + \frac{1}{|W_i|}\sum_{t\in W_i}\lambda_{t,j}. \tag{3}$$

Assumption 1 can be regarded as a generalization of the $(W, \epsilon)$-constrained dynamics in [39]. It essentially assumes that the time horizon can be divided into intervals, within which there exist stationary policies that can stablize the network (Eq. (3)). As a quick sanity check, for stationary instances where the arrival rate vector is in the interior of the capacity region, Assumption 1 is automatically satisfied with $C_W = 0$ (hence all $W_i$s are singleton sets) and all $\vec{\theta}_i$s are

TABLE I

OVERVIEW OF OUR ALGORITHMS AND CLOSELY RELATED WORK

| Algorithm | System Stabilizable Under These Assumptions | Average Queue Length |
|---|---|---|
| `MaxWeight` [36] | Homogeneous Jobs | $\mathcal{O}\left(\frac{KM^2}{\epsilon}\right)$ |
| | Assumption 1 + accurate service rate forecasts | $\mathcal{O}\left(\frac{C_W KM^2}{\epsilon}\right)$ |
| `MaxWeight` with Discounted UCB [33] | Assumption 1,<br>Service rates have smoothness matching the discounting factor | $\left(MK\epsilon^{-1}\right)^{\mathcal{O}(1/\delta)}$ |
| `SoftMW` (**Ours**, Algorithm 2) | Assumption 1,<br>Assumption 2 ($\mathcal{O}(T^{\frac{1}{2}-\delta})$ reference policy total variation) | $\mathcal{O}\left(\frac{C_W KM^2}{\epsilon}\right)$ |
| `SSMW` (**Ours**, Algorithm 3) | Assumption 1,<br>Assumption 3 ($\mathcal{O}(T^{1-\delta})$ reference policy time-homogeneous total variation) | $\left((1+C_V)MK\epsilon^{-1}\right)^{\mathcal{O}(1/\delta)}$ |
| `SoftMW+` (**Ours**, Algorithm 4) | Assumption 1, Assumption 2 ($\mathcal{O}(T^{\frac{1}{2}-\delta})$ reference policy total variation),<br>Arrivals and departures can be unbounded, but have bounded $\alpha$-th moment, $\alpha \cdot \delta > 7$ | $\mathcal{O}\left(\frac{C_W KM^2}{\epsilon}\right)$ |
| `SSMW+` (**Ours**, Algorithm 5) | Assumption 1,<br>Assumption 3 ($\mathcal{O}(T^{1-\delta})$ reference policy time-homogeneous total variation),<br>Arrivals and departures can be unbounded, but have bounded 2nd moment | $\left((1+C_V)MK\epsilon^{-1}\right)^{\mathcal{O}(1/\delta)}$ |

[33] uses similar assumption where the one-step service rate drift of each channel is universally upper-bounded by some polynomial of $(1-\gamma)^{-1}$. Here $\gamma$ is a hyper-parameter of their algorithm, namely the discounting factor in UCB.

equal to some fixed element $\vec{\theta} \in \Delta^{[K]}$, which is a randomized policy capable of stabilizing the system.

*Remark:* In fact, under the above assumption, by a quadratic Lyapunov drift argument (see Theorem 1), we can also show that a policy, in which at each time step $t$ we serve a type of job $a_t$ independently at random according to the distribution indicated by $\vec{\theta}_t$, can stabilize the system as well (require knowing $\vec{\theta}_t$ beforehand). We call $\{\vec{\theta}_t : t \ge 1\}$ *the reference mixed action sequence*, and refer to the above randomized policy induced by $\{\vec{\theta}_t : t \ge 1\}$ as *the reference randomized policy*.

With Assumption 1, in general, it is still a challenging problem to scheduling the system. For our main results in Section V, we need another technical assumption presented below.

*Assumption 2 (Reference Policy Stationarity): For the reference mixed action sequence $\{\vec{\theta}_t\}$ in Assumption 1, there exist some $\delta > 0$ and $C_V > 0$ such that*

$$\sum_{t=1}^{T-1} \|\vec{\theta}_{t+1} - \vec{\theta}_t\|_1 \le C_V T^{\frac{1}{2}-\delta}$$

*for any $T \ge 1$.*

Intuitively speaking, Assumption 2 says that the sequence $\{\vec{\theta}_t\}$ (and hence the environment) does not change in a very abrupt way. Similar smooth assumptions have also been made in existing results, e.g., [33].[1] In Section VI, we will also study when can we handle problems where the reference policy has significantly larger variation. Note that Assumption 2 is not restrictive. For instance, when the network is stationary, if it is stabilizable, one can show that there exists a fixed reference policy that stabilizes the network, i.e., $\vec{\theta}_t = \vec{\theta}$ for all $t$.

## V. QUEUE SCHEDULING WITH ONLY BANDIT FEEDBACK

In contrast to the setting with perfect network state knowledge, in our case, there is no such accurate channel condition for the scheduler. Specifically, the server only receives a bandit feedback for each time step's actual service, i.e., only $S_{t,a_t}$ is known after the service decision $a_t$ is made.

In this section, we present a novel algorithm, which is capable of stabilizing the system using only bandit feedback, $S_{t,a_t}$. Our core idea is to embed a suitable Multi-Armed Bandit algorithm into the `MaxWeight` scheduler [36], so that the term $\mathbb{E}[\sum_{t=1}^{T} Q_{t-1,a_t} S_{t,a_t}]$, which is the key ingredient of `MaxWeight`, is guaranteed to be not too far from $\mathbb{E}[\sum_{t=1}^{T} \langle \mathbf{Q}_{t-1} \odot \mathbf{S}_t, \vec{\theta}_t \rangle]$. Given access to $\vec{\sigma}_t$, `MaxWeight` achieves this by greedily choosing $a_t = \arg\max_i Q_{t-1,i}\sigma_{t,i}$ at each time step $t$. However, when $\vec{\sigma}_t$ is unknown and time-varying, it is hard to guarantee that each summand $Q_{t-1,a_t} S_{t,a_t}$ is large. Thus, we focus on optimizing the whole sum $\mathbb{E}[\sum_{t=1}^{T} Q_{t-1,a_t} S_{t,a_t}]$.

In the remainder of this section, we will first present `EXP3.S+`, an extended version of the `EXP3.S` [1] algorithm for adversarial MAB (Section V-A). `EXP3.S+` has adequate flexibility to serve as an important building block of our novel scheduling algorithm `SoftMW` (Section V-B). We also present its performance guarantee, as it is key for understanding our later analysis. Finally, in Section V-C, we outline analysis of `SoftMW` and describe several important novel techniques to relate adversarial MAB learning to Lyapunov drift analysis.

### A. `EXP3.S+`: An Extended Version of `EXP3.S`

We first present `EXP3.S+`, which extends the `EXP3.S` algorithm [1], designed originally for solving adversarial Multi-Armed Bandit (MAB) problems, to address the potentially unbounded queue lengths in queueing systems, which cannot be directly handled by existing bandit algorithms.

[1] Strictly speaking, [33] introduces a smoothness assumption on the arrival and service rate rather than the reference randomized policy.

```
Algorithm 1 EXP3.S+
Input: Number of actions $K$, time-horizon length $T$,
       initial mixed action $\mathbf{x}_1 \in \Delta^{[K]}$
Output: A sequence of actions $a_1, a_2, \dots, a_T \in [K]$
1 Intermediate Variables A sequence of learning rates
    $\eta_1, \eta_2, \dots, \eta_T \in \mathbb{R}_+$, a sequence of implicit
    exploration rates $\beta_1, \beta_2, \dots, \beta_T \in [0, 1/K]$,
    a sequence of explicit exploration rates
    $\gamma_1, \gamma_2, \dots, \gamma_T \in [0, 1/2]$, a sequence of explicit
    exploration normal vectors $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_T \in \Delta^{[K]}$
2 $\Psi(\mathbf{x}) \triangleq \sum_{i=1}^{K} (x_i \ln x_i - x_i)$
3 for $t = 1, 2, \dots, T$ do
4     Choose $\beta_t$, $\eta_t$, $\mathbf{e}_t$ and $\gamma_i$
5     Below denote by
        $\Delta^{[K],\beta_t} \triangleq \{\mathbf{x} \in \Delta^{[K]} : \mathbf{x}_i \geq \beta_t \quad \forall i \in [K]\}$
6     $\mathbf{p}_t \leftarrow (1 - \gamma_t)\mathbf{x}_t + \gamma_t \mathbf{e}_t$
7     Sample $a_t \sim \mathbf{p}_t$, take action $a_t$, observe $g_{t,a_t}$
8     $\widetilde{\mathbf{g}}_t \leftarrow \begin{cases} g_{t,a_t}/p_{t,a_t} & \text{the } a_t\text{-th coordinate} \\ 0 & \text{the other coordinates} \end{cases}$
9     $\mathbf{x}_{t+1} \leftarrow \arg\min_{\mathbf{x}' \in \Delta^{[K],\beta_t}} \langle -\eta_t \widetilde{\mathbf{g}}_t, \mathbf{x}' \rangle + D_\Psi(\mathbf{x}', \mathbf{x}_t)$
```

More formally, EXP3.S+ applies to the following scenario: there is an agent and an adversary simultaneously making decisions on a finite-length time-horizon $t = 1 \dots T$. At each time $t$, the agent chooses an $\mathbf{x}_t \in \Delta^{[K]}$ deterministically based on observed history, then samples $a_t \in [K]$ according to $\mathbf{x}_t$. Simultaneously (at time $t$), the adversary chooses $\mathbf{g}_t \in \mathbb{R}^K$ deterministically, based on observed history. Then, $g_{t,a_t}$ is revealed to the agent. The high-level objective for the agent is to maximize the cumulative feedback $\sum_{t=1}^{T} g_{t,a_t}$. The details of our EXP3.S+ are described in Algorithm 1.

Specifically, EXP3.S+ works by producing sequences of candidate mixed actions $\mathbf{x}_t$s according to mirror descent steps (Line 9). At each time step $t$, $\mathbf{x}_t$ will be further mixed with an exploration vector $\mathbf{e}_t$ to obtain $\mathbf{p}_t$ (Line 6); the final chosen action $a_t$ is then sampled according to $\mathbf{p}_t$ (Line 7). After receiving the reward feedback $g_{t,a_t}$, an importance-sampling estimate $\widetilde{\mathbf{g}}_t$ for the whole reward vector $\mathbf{g}_t$ is calculated (Line 8) and used as the gradient in the next mirror descent step (Line 9). The mirror descent step picks a new mixed action $\mathbf{x}_{t+1}$ that is not only close to the last mixed action $\mathbf{x}_t$, but also gains a large single step reward with respect to the reward estimator $\mathbf{g}_t$.

*Remark:* The amplitude of feedback value $g_{t,a_t}$ in Line 7 is crucial to the correctness of EXP3.S+. The original EXP3.S algorithm in [1] uses a constant learning rate $\eta$ and a constant exploration rate $\gamma$ across all $T$ time steps. However, the algorithm can only support problems with reward feedback values no more than $\eta^{-1}\gamma$, and does not apply to our setting. For our purpose, in the presented algorithms, we will feed $Q_{t-1,a_t} S_{t,a_t}$ into EXP3.S+ as the reward value, which is a quantity that can be arbitrarily large (since $Q_{t-1,a_t}$ can go unbounded as $t \to \infty$). In EXP3.S+, the learning rates and exploration rates can both be time-varying, and the exploration rates can even be action-dependent (it allows specifying any $\mathbf{e}_t \in \Delta^{[K]}$ rather than $\mathbf{1}/K$). In each mirror descent step, we choose to pick the new action on a subset $\Delta^{[K],\beta_t}$ of the whole simplex $\Delta^{[K]}$, which is different from vanilla EXP3. This novel design is crucial to guarantee a small regret against a change sequence of actions (i.e., dynamic regret) instead of against a fixed action.

The formal performance guarantee of EXP3.S+ for $\sum_{t=1}^{T} g_{t,a_t}$ is given in Theorem 1 below.

*Theorem 1 (EXP3.S+ Dynamic Regret Guarantee): During the execution of Algorithm 1, for any fixed sequence $\vec{\theta}_1, \dots, \vec{\theta}_T \in \Delta^{[K]}$, if w.p.1 the following events happen,*

(i) $\mathbf{x}_1 \in \Delta^{[K],\beta_1}$,

(ii) $\mathbf{g}_t \leq \eta_t^{-1} \gamma_t \mathbf{e}_t$ *for all* $1 \leq t \leq T$,

(iii) $\eta_1 \geq \eta_2 \geq \cdots \geq \eta_T$,

(iv) $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_T$,

(v) $\vec{\theta}_t \in \Delta^{[K],\beta_t}$ *for all* $1 \leq t \leq T$,

*then let*

$$V \triangleq \sum_{t=1}^{T-1} \|\vec{\theta}_{t+1} - \vec{\theta}_t\|_1,$$

*we will have*

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle \mathbf{g}_t, \vec{\theta}_t \rangle\right] - \mathbb{E}\left[\sum_{t=1}^{T} g_{t,a_t}\right] \leq (1 + V)\mathbb{E}\left[\eta_T^{-1} \ln \frac{1}{\beta_T}\right] + e\mathbb{E}\left[\sum_{t=1}^{T} \eta_t \|\mathbf{g}_t\|_2^2\right] + \mathbb{E}\left[\sum_{t=1}^{T} \gamma_t \langle \mathbf{g}_t, \mathbf{e}_t \rangle\right].$$

In Theorem 1, the value $V$ characterizes how frequently $\vec{\theta}_t$ (the reference policy) changes over time. Our results build on Assumptions 2 and 3, which only require $V$ to be $O(\sqrt{T})$ (the $C_V T^{1/2-\delta}$ term) in Assumption 2 and $O(T^{1-\delta})$ (the $C_V T^{1-\delta}$ term) in Assumption 3. These two conditions are not restrictive. For instance, in the case of stationary networks, $V = O(1)$ since there exist constant reference policies.

In Appendix B, we provide a formal proof for Theorem 1 using an analysis based on Online Mirror Descent [40], which is much more suitable for handling time-varying learning rates compared to the classical sum-of-exp potential function approach in [1]. We also discuss a practical implementation of the $\arg\max$ calculation (at Line 9) in Appendix C. We note that Algorithm 1 and its analysis can be of independent interest and applied to problems other than stochastic network scheduling.

### B. Soft Max-Weight Scheduling Using EXP3.S+

We now present our novel scheduling algorithm, SoftMW, in Algorithm 2. SoftMW is based on carefully designed feedback signals as well as parameters and learning rates in EXP3.S+. Its name refers to the computation in EXP3.S+ (Algorithm 1) that is heavily based on the softmax operation (see Appendix C).

At the beginning of SoftMW, an EXP3.S+ instance is created. Then, at each time step $t$, the parameters $\beta_t, \eta_t, \mathbf{e}_t, \gamma_t$ in EXP3.S+ are determined, depending on the current time index $t$ and current queueing backlog $\mathbf{Q}_{t-1}$. EXP3.S+ will then output an action $a_t$, SoftMW just choose to serve the

**Algorithm 2** `SoftMW` (**Soft MaxWeight**)

```
Input: One-step arrival/service upper-bound $M > 0$,
       Number of job types $K$, Problem instance
       smoothness parameter $\delta > 0$
Output: A sequence of job types to serve $a_1, a_2, \ldots \in [K]$
Initialize an EXP3.S+ instance with $K$ available actions and
  $\mathbf{x}_1 = \mathbf{1}/K$
for $t = 1, 2, \ldots$ do
    Pick the following parameters of EXP3.S+ for time slot
      $t$:
        $\beta_t \leftarrow t^{-3}/K$
        $\eta_t =$
          $\left(t^{-(\frac{1}{4}-\frac{\delta}{2})} M \sqrt{86M^2K^6t^{\frac{3}{2}} + \sum_{s=0}^{t-1}\|\mathbf{Q}_s\|_2^2}\right)^{-1}$
        $\mathbf{e}_t = \mathbf{Q}_{t-1}/\|\mathbf{Q}_{t-1}\|_1$
        $\gamma_t = M\eta_t\|\mathbf{Q}_{t-1}\|_1 =$
          $\|\mathbf{Q}_{t-1}\|_1 \left(t^{-(\frac{1}{4}-\frac{\delta}{2})} \sqrt{86M^2K^6t^{\frac{3}{2}} + \sum_{s=0}^{t-1}\|\mathbf{Q}_s\|_2^2}\right)^{-1}$
    Take a new action decision output $a_t$ from EXP3.S+,
      serve the $a_t$-th queue, regard $Q_{t-1,a_t}S_{t,a_t}$ as a new
      feedback $g_{t,a_t}$ and feed it into the current EXP3.S+
      instance
```

$a_t$-th queue. Finally, after observing the service amount $S_{t,a_t}$, `SoftMW` feeds $Q_{t-1,a_t}S_{t,a_t}$ as the MAB feedback at time $t$ into the `EXP3.S+` instance.

The intuitive reason why Algorithm 2 works is as follows. We use `EXP3.S+` in a carefully designed way to drive the scheduling process, so that the effect of Algorithm 2 is very closed to (or better than) the reference randomized policy given by Assumption 1, in the sense that under Algorithm 2, the queues' total quadratic Lyapunov drift is only slightly larger (or even smaller) than that under the reference randomized policy. Therefore, Algorithm 2 has similar (or even stronger) capability of stabilizing the system.

Algorithm 2's average queue length bound on any finite time-horizon is given in Theorem 2.

*Theorem 2: For problem instances satisfying Assumptions 1 and 2, `SoftMW` (Algorithm 2) guarantees*

$$\frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{Q}_t\|_1\right] \le \frac{2(K+1)M^2 + 4C_W(KM^2 + \epsilon KM)}{\epsilon} + o(1).$$

*In particular, the system is stable.*

*Remark:* As a quick sanity check, for stationary problem instances, Theorem 2 gives $\mathcal{O}(KM^2/\epsilon)$ average queue length bound, which coincides with the classical result we can achieve in stationary problems ([28] Sec. 3.1 ). In fact, one can show that for both (i) pretending to have accurate one-step forecasts for service rates and running vanilla `MaxWeight`, and (ii) running the reference randomized policy $\{\vec{\theta}_t\}$ specified in Assumption 1, the average queue length bounds via a standard quadratic Lyapunov analysis are $\mathcal{O}\left(\frac{(C_W K + K + 1)M^2}{\epsilon}\right)$. Therefore, informally, in terms of queue length bound, the overhead due to `SoftMW` on problem instances satisfying Assumption 2 is insignificant.

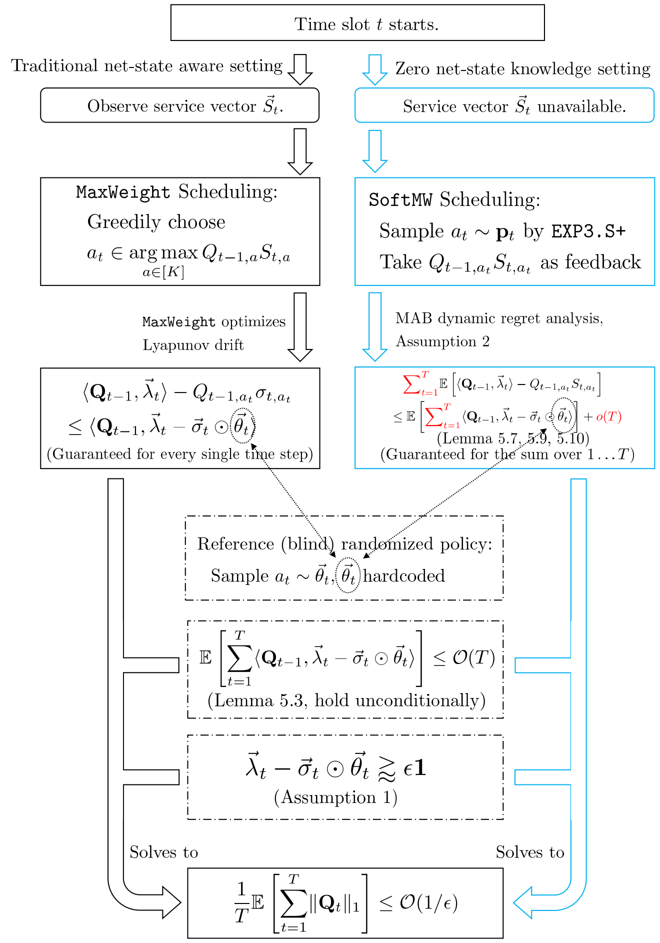

Time slot $t$ starts.
Traditional net-state aware setting
Zero net-state knowledge setting
Observe service vector $\vec{S}_t$.
Service vector $\vec{S}_t$ unavailable.
MaxWeight Scheduling:
Greedily choose
$a_t \in \arg\max_{a\in[K]} Q_{t-1,a}S_{t,a}$
SoftMW Scheduling:
Sample $a_t \sim \mathbf{p}_t$ by EXP3.S+
Take $Q_{t-1,a_t}S_{t,a_t}$ as feedback
MaxWeight optimizes
Lyapunov drift
MAB dynamic regret analysis,
Assumption 2
$\langle \mathbf{Q}_{t-1}, \vec{\lambda}_t\rangle - Q_{t-1,a_t}\sigma_{t,a_t}$
$\le \langle \mathbf{Q}_{t-1}, \vec{\lambda}_t - \vec{\sigma}_t \odot \vec{\theta}_t\rangle$
(Guaranteed for every single time step)
$\sum_{t=1}^{T}\mathbb{E}\left[\langle \mathbf{Q}_{t-1}, \vec{\lambda}_t\rangle - Q_{t-1,a_t}S_{t,a_t}\right]$
$\le \mathbb{E}\left[\sum_{t=1}^{T}\langle \mathbf{Q}_{t-1}, \vec{\lambda}_t - \vec{\sigma}_t \odot \vec{\theta}_t\rangle\right] + o(T)$
(Lemma 5.7, 5.9, 5.10)
(Guaranteed for the sum over $1\ldots T$)
Reference (blind) randomized policy:
Sample $a_t \sim \vec{\theta}_t$, $\vec{\theta}_t$ hardcoded
$\mathbb{E}\left[\sum_{t=1}^{T}\langle \mathbf{Q}_{t-1}, \vec{\lambda}_t - \vec{\sigma}_t \odot \vec{\theta}_t\rangle\right] \le \mathcal{O}(T)$
(Lemma 5.3, hold unconditionally)
$\vec{\lambda}_t - \vec{\sigma}_t \odot \vec{\theta}_t \gtrapprox \epsilon\mathbf{1}$
(Assumption 1)
Solves to
Solves to
$\frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{Q}_t\|_1\right] \le \mathcal{O}(1/\epsilon)$

Fig. 2. Illustration for the analysis outline of `SoftMW`.

### C. Queue Stability Analysis Outline

In this section, we provide a brief outline of how to formally establish the queue stability result (Theorem 2). We first review the general procedure from quadratic Lyapunov drift analysis, which is capable of building the average queue length bounds for both the reference randomized policy $\{\vec{\theta}_t\}$ and ordinary `MaxWeight` policy (the left path in Figure 2). Then, using the result on `EXP3.S+`'s total dynamic regret (Theorem 1), we show that the `EXP3.S+` scheduling used in `SoftMW` can lead to terminal Lyapunov function values close to the reference policy in Assumption 1, differing by a term proportional to $\sqrt{\sum\|\mathbf{Q}_t\|_2^2}$. Finally, we relate this $\sqrt{\sum\|\mathbf{Q}_t\|_2^2}$ term with the queue lengths ($\sum\|\mathbf{Q}_t\|_1$) we want to bound, and show that this difference term is indeed $o(T)$, so that we can obtain an average queue length bound of the same order as compared to the reference randomized policy and `MaxWeight`. This novel queue stability analysis combining a Lyapunov drift argument and adversarial MAB dynamic regret analysis is illustrated in the rightmost, highlighted (in blue) part of Figure 2.

*1) Recap of Lyapunov Drift Analysis:* In our analysis, we use standard results from quadratic Lyapunov drift analysis [28]. Conventionally, we define

$$L_t \triangleq \frac{1}{2}\|\mathbf{Q}_t\|_2^2 = \frac{1}{2}\sum_{i=1}^{K} Q_{t,i}^2,$$

as the quadratic Lyapunov function of the queue lengths. We first have the following standard lemma regarding the drift upper bound.

*Lemma 1 (General quadratic Lyapunov Drift Upper-bound [28]): Consider any scheduling policy for this queueing system and suppose that the policy randomly picks a job type $a_t$ according to a probability distribution $\mathbf{p}_t$ (which may depend on the system's history, i.e., $\mathbf{p}_t$ is an $\mathcal{F}_{t-1}$-measurable random vector supported on $\Delta^{[K]}$). Let $\mathbf{Q}_t$ denote the queue length vector under that policy. We have*

$$\begin{aligned}&\mathbb{E}\left[L_t - L_{t-1} \middle| \mathcal{F}_{t-1}\right]\\ &\le \frac{(k+1)M^2}{2} + \langle \mathbf{Q}_{t-1}, \vec{\lambda}_t - \vec{\sigma}_t \odot \mathbf{p}_t\rangle\\ &= \frac{(k+1)M^2}{2} + \langle \mathbf{Q}_{t-1}, \vec{\lambda}_t\rangle - \mathbb{E}\left[Q_{t-1,a_t}S_{t,a_t} \middle| \mathcal{F}_{t-1}\right]\end{aligned}$$

*for any $t \ge 1$. By summing the inequalities over $1 \le t \le T$, taking total expectation and then rearranging the terms, we get*

$$\mathbb{E}\left[\sum_{t=1}^{T} Q_{t-1,a_t}S_{t,a_t} - \langle \mathbf{Q}_{t-1}, \vec{\lambda}_t\rangle\right] \le \frac{(K+1)M^2 T}{2} \quad (4)$$

*for any time horizon length $T \ge 1$.*

Next, we have Lemma 2 regarding the drift value under the reference policies. As in the standard Lyapunov drift analysis [28], this bound will be useful for deriving queue length results for queue-based policies.

*Lemma 2 (Negative Lyapunov Drift under Reference Policy): Suppose Assumption 1 holds. Consider any scheduling policy for this queueing system, under which the queue length vectors are denoted by $\{\mathbf{Q}_t\}$. Let $\{\vec{\theta}_t : t \ge 1\}$ be the sequence of probabilities to serve each queue as defined in Assumption 1. Then, for any time horizon length $T \ge 1$, we can find a constant $\mathcal{T}_T$ that depends only on $T$, such that $T \le \mathcal{T}_T \le T + \sqrt{\frac{T}{C_W} + 1}$ and*

$$\begin{aligned}&\mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T} \langle \mathbf{Q}_{t-1}, \vec{\sigma}_t \odot \vec{\theta}_t - \vec{\lambda}_t\rangle\right]\\ &\ge \epsilon\mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T} \|\mathbf{Q}_{t-1}\|_1\right] - (KM^2 + \epsilon KM)C_W\mathcal{T}_T\\ &\ge \epsilon\mathbb{E}\left[\sum_{t=1}^{T} \|\mathbf{Q}_{t-1}\|_1\right] - (KM^2 + \epsilon KM)C_W\mathcal{T}_T.\end{aligned}$$

*Here $\odot$ is the element-wise product, i.e., $\vec{a} \odot \vec{b} = (a_1b_1, \ldots, a_Kb_K)$, and $C_W$ is the constant defined in Assumption 1.*

*Proof:* See Appendix D. □

Combining Theorem 1 and Theorem 2, we obtain the following important proposition for our analysis.

*Theorem 1 (Sufficiently-Large-Weight Implies Queue Stability):* Suppose Assumption 1 holds, also suppose a scheduling policy guarantees the following.

$$\mathbb{E}\left[\sum_{t=1}^{T} Q_{t-1,a_t}S_{t,a_t}\right] \ge \mathbb{E}\left[\sum_{t=1}^{T} \langle \mathbf{Q}_{t-1}, \vec{\sigma}_t \odot \vec{\theta}_t\rangle\right] - f(T)$$

for all $T \ge \max\{\frac{4}{C_W}, C_W\}$, where $f(T)$ is some non-negative, increasing function of $T$. Then, we have

$$\begin{aligned}&\frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T} \|\mathbf{Q}_t\|_1\right]\\ &\quad \le \frac{(K+1)M^2 + 2C_W(KM^2 + \epsilon KM)}{\epsilon} + \frac{f(2T)}{\epsilon T}.\end{aligned}$$

In particular, if $f(T)$ is $\mathcal{O}(T)$, then this policy stabilizes the system.

*Proof:* See Appendix D. □

*Remark:* Theorem 1 implies that for problem instances satisfying Assumption 1, serving the queue according to either $\{\vec{\theta}_t\}$ (the reference randomized policy) or the vanilla `MaxWeight` algorithm (assuming that service rate forecasts are available to the algorithm at the time of decision making), the average queue length will be no more than $\frac{(K+1)M^2 + 2C_W(KM^2 + \epsilon KM)}{\epsilon}$, as claimed earlier in Section IV. This is because in both cases, we have $\mathbb{E}\left[Q_{t-1,a_t}S_{t,a_t} \middle| \mathcal{F}_{t-1}\right] \ge \langle \mathbf{Q}_{t-1}, \vec{\sigma}_t \odot \vec{\theta}_t\rangle$. Hence, the condition in Theorem 1 holds with $f(T) = 0$ for these two policies.

In the remaining of the analysis, we will derive the corresponding $f(T)$ for `SoftMW+`, so that we can conclude the queue stability via an argument similar to Theorem 1.

*2) From `EXP3.S+` Regret Bound to Lyapunov Function Value Bound:* To build the queue stability result for `SoftMW` (Algorithm 2), our high-level idea is to develop the required condition in Theorem 1 such that $f(T)$ can also be properly controlled. Since `SoftMW` makes decisions based on `EXP3.S+`, intuitively, we should utilize the regret upper-bound result Theorem 1. In order to do that, we need to verify that the required conditions (i)-(iv)[2] in Theorem 1 hold.

In fact, in `SoftMW`, our choices of $\eta_t$s and $\beta_t$s are obviously decreasing, hence condition (iii) and (iv) hold. We choose $\mathbf{x}_1 = (1/K, \ldots, 1/K)$ thus condition (i) also holds; the choice of $\gamma_t$ and $\mathbf{e}_t$ also guarantees condition (ii). The real issue is whether $\gamma_\tau$'s exceed $\frac{1}{2}$. This is established in the following proposition.

*Theorem 2 (Feasibility of the Exploration Rates in `SoftMW`):* For all $t \ge 1$, we have $\gamma_t \le \frac{1}{2}$ in `SoftMW`.

Appendix E gives a detailed proof of Theorem 2. Having confirmed that the algorithm is feasible, we can now safely apply Theorem 1, resulting in the following property of `SoftMW`.

*Lemma 3 (`SoftMW` Large-Weight Guarantee): Suppose Assumptions 1 and 2 hold; then, running Algorithm 2 guarantees*

$$\begin{aligned}&\sum_{t=1}^{T} \mathbb{E}\left[\langle \mathbf{Q}_{t-1}, \mathbf{S}_t \odot \vec{\theta}_t\rangle - Q_{t-1,a_t}S_{t,a_t}\right]\\ &\le 4M^2 + \left[9M(1 + C_V)T^{\frac{1}{4}-\frac{\delta}{2}}(3\ln T + \ln K)\right]\end{aligned}$$

[2] The reference policy $\{\vec{\theta}_t\}$ itself may not satisfies condition (v), but we will project each $\vec{\theta}_t$ onto $\Delta^{[K],\beta_t}$ as $\vec{\theta}'_t$, and only use Theorem 1 to obtain a regret bound against the action sequence $\{\vec{\theta}'_t\}$.

$$\cdot\, \mathbb{E}\left\{\sqrt{86M^2K^6T^{\frac{3}{2}} + \sum_{t=1}^{T}\|\mathbf{Q}_{t-1}\|_2^2}\right\} \tag{5}$$

*for any time horizon length $T \geq 1$. Here $\{\vec{\theta}_t\}$ is the reference policy in Assumptions 1 and 2.*

*Proof:* See Appendix E. □

Theorem 3 gives an upper-bound for $\mathbb{E}\left[\sum Q_{t-1,a_t}S_{t,a_t} - \sum\langle\mathbf{Q}_{t-1}, \vec{\sigma}_t \odot \vec{\theta}_t\rangle\right]$, which is closely related to the condition required by Theorem 1. However, this upper-bound is not yet a quantity that depends solely on $T$; it still has a factor of $\sqrt{\mathbb{E}[\sum\|\mathbf{Q}_t\|_2^2]}$, depending on the actual queueing trajectory. Therefore, we are unable to apply Theorem 1 directly to claim queue stability. Rather, we need to work with the $\sqrt{\mathbb{E}[\sum\|\mathbf{Q}_t\|_2^2]}$ factor, to convert it to the cumulative queue length $\mathbb{E}[\sum\|\mathbf{Q}_t\|_1]$, just as we did in Theorem 2 to convert $\mathbb{E}\left[\sum\langle\mathbf{Q}_{t-1}, \vec{\sigma}_t \odot \vec{\theta}_t - \vec{\lambda}_t\rangle\right]$ to queue lengths.

*3) Relate Regrets in $\sqrt{\mathbb{E}[\sum\|\mathbf{Q}_t\|_2^2]}$ to Queue Lengths $\mathbb{E}[\sum\|\mathbf{Q}_t\|_1]$:* Plugging Eq. (5) into Eq. (4) in Theorem 1, after further applying Theorem 2 and rearranging terms, we get the following proposition, which offers an inequality connecting $\sqrt{\mathbb{E}[\sum\|\mathbf{Q}_t\|_2^2]}$ and $\mathbb{E}[\sum\|\mathbf{Q}_t\|_1]$.

*Theorem 3:* Given Assumptions 1 and 2, Algorithm 2 gives us

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T}\|\mathbf{Q}_{t-1}\|_1\right]\\
&\leq \frac{(K+1)M^2 + 2C_W(KM^2+\epsilon KM)}{\epsilon}\mathcal{T}_T + \frac{4M^2}{\epsilon}\\
&\quad + \frac{g(\mathcal{T}_T)}{\epsilon}\cdot\sqrt{86M^2K^6\mathcal{T}_T^{\frac{3}{2}} + \mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T}\|\mathbf{Q}_{t-1}\|_2^2\right]}\\
&\leq \frac{(K+1)M^2 + 2C_W(KM^2+\epsilon KM)}{\epsilon}\mathcal{T}_T + \frac{4M^2}{\epsilon}\\
&\quad + \frac{\sqrt{86}MK^3\mathcal{T}_T^{\frac{3}{4}}g(\mathcal{T}_T)}{\epsilon} + \frac{g(\mathcal{T}_T)}{\epsilon}\cdot\sqrt{\mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T}\|\mathbf{Q}_{t-1}\|_2^2\right]}
\end{aligned} \tag{6}$$

for any $T \geq \max\{\frac{4}{C_W}, C_W\}$, where $\mathcal{T}_T$ is some constant no more than $2T$, and

$$g(T) = 9M(1+C_V)T^{\frac{1}{4}-\frac{\delta}{2}}(3\ln T + \ln K) = \widetilde{\mathcal{O}}(T^{\frac{1}{4}-\frac{\delta}{2}}).$$

Recall that all arrivals and departures are assumed to be bounded by a constant $M$. Therefore, each dimension of the queue length vectors $\{\mathbf{Q}_t\}$ is a sequence of non-negative numbers, where the difference between any two adjacent terms is within ±$M$. We may then make use of the following lemma for such bounded-difference sequences.

*Lemma 4: Suppose $x_1 = 0$, $x_2, \ldots, x_n \geq 0$, $|x_{i+1} - x_i| \leq 1$ for all $1 \leq i < n$. Denote by $S = \sum_{i=1}^{n} x_i$; then we have*

$$\sum_{i=1}^{n} x_i^2 \leq 4S^{\frac{3}{2}}.$$

*Proof:* See Appendix E. □

For our purposes, Theorem 4 guarantees that

$$\begin{aligned}
\sum_{t=1}^{T}\|\mathbf{Q}_{t-1}\|_2^2 &\leq 4\sqrt{M}\sum_{i=1}^{K}\left(\sum_{t=1}^{T}Q_{t-1,i}\right)^{\frac{3}{2}}\\
&\leq 4\sqrt{M}\left(\sum_{t=1}^{T}\|\mathbf{Q}_{t-1}\|_1\right)^{\frac{3}{2}}.
\end{aligned} \tag{7}$$

Then, plugging Eq. (7) into Eq. (6), we obtain the following inequality that depends *entirely* on $\mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T}\|\mathbf{Q}_{t-1}\|_1\right]$:

$$\mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T}\|\mathbf{Q}_{t-1}\|_1\right] \leq h(\mathcal{T}_T) + g(\mathcal{T}_T)\left(\mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T}\|\mathbf{Q}_{t-1}\|_1\right]\right)^{\frac{3}{4}} \tag{8}$$

where

$$\begin{aligned}
g(T) &= 18M^{\frac{5}{4}}(1+C_V)T^{\frac{1}{4}-\frac{\delta}{2}}(3\ln T + \ln K)\\
&= \widetilde{\mathcal{O}}\left(\frac{T^{\frac{1}{4}-\frac{\delta}{2}}}{\epsilon}\right),\\
h(T) &= \frac{(K+1)M^2 + 2C_W(KM^2+\epsilon KM)}{\epsilon}T + \widetilde{\mathcal{O}}(T^{1-\frac{\delta}{2}}).
\end{aligned}$$

It remains to solve Eq. (8), in order to obtain an upper bound for $\mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T}\|\mathbf{Q}_{t-1}\|_1\right]$. To do so, we utilize the following lemma.

*Lemma 5: Let $y, f, g : \mathbb{R}_+ \to [1, \infty)$ be three non-decreasing functions. If*

$$y(x) \leq f(x) + y(x)^{\frac{1}{4}}g(x)$$

*for all $x \geq 0$, then we have*

$$y(x) \leq \left(f(x)^{\frac{1}{4}} + g(x)\right)^4.$$

*Proof:* See Appendix E. □

Finally, according to Theorem 5, the solution of Eq. (8) gives us:

$$\begin{aligned}
&\mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T}\|\mathbf{Q}_{t-1}\|_1\right]\\
&\leq \left(h(\mathcal{T}_T)^{\frac{1}{4}} + g(\mathcal{T}_T)\right)^4\\
&\leq \frac{(K+1)M^2 + 2C_W(KM^2+\epsilon KM)}{\epsilon}\mathcal{T}_T + o(\mathcal{T}_T).
\end{aligned}$$

Thus,

$$\begin{aligned}
&\frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{Q}_{t-1}\|_1\right] \leq \frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{\mathcal{T}_T}\|\mathbf{Q}_{t-1}\|_1\right]\\
&\leq \frac{(K+1)M^2 + 2C_W(KM^2+\epsilon KM)}{\epsilon}\frac{\mathcal{T}_T}{T} + o(\mathcal{T}_T/T)\\
&\leq \frac{2(K+1)M^2 + 4C_W(KM^2+\epsilon KM)}{\epsilon} + o(1)
\end{aligned}$$

as desired.

**Algorithm 3** `SSMW` (**S**liding-window **S**oft**MW**)

```
Input: One-step arrival/service moment upper-bound
       parameter $M > 0$, Number of job types $K$,
       Problem instance smoothness parameter $\delta > 0$
Output: A sequence of job types to serve
        $a_1, a_2, \ldots \in [K]$
1  while true do
2      $T_0 \leftarrow$ the latest time index $t$
         at which we have made a new decision $a_t$
       // for the first iteration, we should have
       $T_0 = 0$
3      $m \leftarrow \max\left\{\lceil\frac{\|\mathbf{Q}_{T_0}\|_\infty}{2M}\rceil, 1\right\}$
4      Run a fresh EXP3.S+ instance for $m$ time steps
         with the following configuration (below $\tau$ denotes
         the time index within the epoch of length $m$,
         1-based):
5          $\beta = m^{-2}/K$
6          $\mathbf{x}_1$ can be any element in
             $\Delta^{[K],\beta} \triangleq \{\mathbf{x} \in \Delta^{[K]} : \mathbf{x}_i \geq \beta \quad \forall i \in [K]\}$
7          $\eta_\tau = \left(6M^2Km^{1+\frac{\delta}{2}}\right)^{-1}$
8          $\mathbf{e}_\tau = \mathbf{Q}_{T_0+\tau-1}/\|\mathbf{Q}_{T_0+\tau-1}\|_1$
9          $\gamma_\tau = M\eta_\tau\|\mathbf{Q}_{T_0+\tau-1}\|_1 =$
             $\frac{1}{6}K^{-1}M^{-1}m^{-1-\frac{\delta}{2}}\|\mathbf{Q}_{T_0+\tau-1}\|_1$
10     Take a new action decision output from the current
         EXP3.S+ instance, serve this type of jobs (recall
         we are at the $(T_0+\tau)$-th time step of the whole
         time horizon), regard $Q_{T_0+\tau-1,a_{T_0+\tau}}S_{T_0+\tau,a_{T_0+\tau}}$
         as a new feedback $g_{\tau,a_\tau}$ and feed it into
         EXP3.S+
```

## VI. Taming Time-Homogeneous $\mathcal{O}(T^{1-\delta})$ Reference Policy Total Variation

In this section, we propose another novel algorithm capable of stabilizing our adversarial queueing system. Specifically, this algorithm is stable under a reference randomized policy with $O(T^{1-\delta})$ total variation, as long as that much total variation is "evenly" distributed throughout the infinite time horizon. This new condition is formalized as follows.

*Assumption 3 (Time-Homogeneous Reference Policy Stationarity): For the sequence $\{\vec{\theta}_t\}$ in Assumption 1, there exist some $\delta > 0$ and $C_V > 0$ such that*

$$\sum_{t=T_0+1}^{T_0+T-1} \|\theta_{t+1} - \theta_t\|_1 \leq C_V T^{1-\delta}$$

*for any $T_0 \geq 0$ and $T \geq 1$.*

*Remark:* Assumption 3 can be viewed as a shift-invariant version of Assumption 2, with the degree of $T$ relaxed from $\frac{1}{2} - \delta$ to $1 - \delta$. Roughly speaking, this assumption holds as long as there is only a finite number of time periods on which the reference policy variation accumulates at a linear rate. For example, if $\sum_{t=0}^{T}\|\theta_{t+1}-\theta_t\|_1 = \Theta(T^{1-\delta})$, then Assumption 3 is satisfied.

For problem instances where Assumptions 1 and 3 hold, we present a new algorithm to stabilize the system, namely **S**liding **S**oft**MW** (`SSMW`), which is detailed in Algorithm 3.

Compared to `SoftMW` (Algorithm 2), `SSMW` (Algorithm 3) does not use historical queue lengths at the beginning to tune the `EXP3.S+` learning rates. Instead, `SSMW` starts with new `EXP3.S+` instances of lengths proportional to the current queue lengths (Line 3). As a result, `SSMW` initiates many more `EXP3.S+` instances throughout its execution, though each `EXP3.S+` period is likely to be short. In this sense, `SSMW` is more similar to `MaxWeight`, since `MaxWeight` always uses the current queue length vector for making new decisions, and disregards how the system arrived at the current state. Theorem 3 gives the queue stability result for `SSMW`.

*Theorem 3: For problem instances satisfying Assumptions 1 and 3, `SSMW` (Algorithm 3) guarantees*

$$\frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{Q}_t\|_1\right] \leq \left[3KM^2m_0 + \frac{(K+1)M^2}{2} + (KM^2+\epsilon KM)C_W + 6M^2\right] \cdot \frac{10}{\epsilon}$$

*for any time horizon of length $T \geq \frac{4}{C_W} + C_W$. In particular, the system is stable. Here $m_0$ is defined as*

$$m_0 \triangleq \inf\left\{m : m \geq 2, f(m') \leq \frac{\epsilon}{2} \forall m' \geq m\right\} \leq \left((1+C_V)MK\ln K\epsilon^{-1}\right)^{\mathcal{O}(1/\delta)}$$

*where*

$$f(m) = 88(1+C_V)MKm^{-\frac{\delta}{2}}(2\ln m + \ln K).$$

The complete formal proof of Theorem 3 can be found in Appendix F. In brief, the derivation of Theorem 3 can be reduced to an important observation, presented next.

*For any fixed problem instance satisfying Assumptions 1 and 3, there exists an instance-dependent constant $m_0 \geq 1$, such that, at some time step $T_0$, we have (a) an `EXP3.S+` instance in `SSMW` that just ended, and (b) $\|\mathbf{Q}_{T_0}\|_\infty \geq 2Mm_0$ (i.e., the next `EXP3.S+` instance that lasts for $m = \lceil\frac{\|\mathbf{Q}_{T_0}\|_\infty}{2M}\rceil$ time steps, where $m \geq m_0$); then we have*

$$\mathbb{E}\left[\sum_{t=1}^{m}\langle\mathbf{Q}_{T_0+t-1}, \vec{\lambda}_{T_0+t}\rangle - Q_{T_0+t-1,a_{T_0+t}}S_{T_0+t,a_{T_0+t}}\right] \leq -\frac{\epsilon}{2}\mathbb{E}\left[\sum_{t=1}^{m}\|\mathbf{Q}_{T_0+t-1}\|_1\right],$$

*where the two expectations are both conditioned on the system state at the end of time $T_0$.*

This claim is formally stated and proved in Theorem 20 (Appendix F). In fact, the magic number $m_0$ in Theorem 3 is just a feasible choice for $m_0$ in the above claim. Assuming the claim is true and letting $Q'_{t,i} = \max\{Q_{t,i} - m_0, 0\}$, one can see that the "shifted queue" $\mathbf{Q}'_t$ enjoys an average queue length bound $\frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{Q}'_t\|_1\right] \leq O(1/\epsilon)$ by standard Lyapunov drift analysis of $\mathbf{Q}'_t$. Therefore, we can conclude that

$\frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{Q}_t\|_1\right] \le \frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{Q}'_t\|_1\right] + Km_0 \le Km_0 + O(1/\epsilon)$.

*Remark:* Compared to the $\mathcal{O}(\epsilon^{-1})$ queue length bound of SoftMW (Theorem 2), Theorem 3 only gives an $\epsilon^{\mathcal{O}(1/\delta)}$ queue length guarantee (here $m_0$ is significantly larger than $O(\epsilon^{-1})$ and hence the bottleneck). Nevertheless, simulation results in Appendix A show that the empirical performance of SSMW is comparable to, or even better than that of, SoftMW.

## VII. Related Work

Recent literature includes learning-based scheduling policies that require little prior-knowledge and can gather channel statistics at run-time.

Learning-based approaches to scheduling queueing systems without perfect channel state knowledge require substantial exploration, to probe for more information of all the channels inside the system instead of merely exploiting the statistics at hand (e.g., via a MaxWeight style planning). Typical ways to introduce adequate exploration include epsilon-greedy, which explicitly allocates a small probability to serve each channel unconditionally [31], [41], [42]; here the exploration is independent of the queue sizes and historical channel statistics and thus almost decoupled from exploitation. By contrast, optimistic exploration works by adding bonus terms to current channel statistics, so that exploration and exploitation are naturally coupled during scheduling [30], [31], [33], [43]. Upper confidence bound (UCB) [44] is a classical method for designing a bonus term.

Existing works on scheduling in non-stationary queueing systems include [33], which uses discounted UCB estimators for an up-to-date service rate of each link to replace the actual mean services rate in classical MaxWeight. The resulting policy can stabilize problem instances where the difference of each link's arrival (and service) rates between any two time steps in any time window of length $W$ is sufficiently small, and this window length $W$ needs to match with the discounting factor $\gamma$ used in discounted UCB estimators. Compared to [33], our smoothness assumption is on the reference randomized policies rather than the true service rates.

## VIII. Conclusion and Future Work

In this paper, we propose a novel approach to apply adversarial bandit learning techniques to schedule queueing systems with unknown, time-varying network states. The presented new algorithms SoftMW and SSMW are capable of stabilizing the system whenever the system can be stabilized by some (possibly unknown) sequence of randomized policies, and their time-variation satisfies some mild condition. We further generalize our results to the setting where arrivals and departures only have bounded moments and develop two stablizing algorithms SoftMW+ and SSMW+.

We believe our approach can be generalized to more complex stochastic networks (e.g., multi-hop networks), and to achieve other tasks such as utility optimization subject to queue stability. It is also an interesting future work to design distributed network scheduling algorithms using adversarial bandit learning techniques.

## References

[1] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The nonstochastic multiarmed bandit problem," *SIAM J. Comput.*, vol. 32, no. 1, pp. 48–77, 2002.

[2] X. Kong, N. Lu, and B. Li, "Optimal scheduling for unmanned aerial vehicle networks with flow-level dynamics," *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 1186–1197, Mar. 2021.

[3] S. Tsanikidis and J. Ghaderi, "On the power of randomization for scheduling real-time traffic in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 29, no. 4, pp. 1703–1716, Aug. 2021.

[4] S. T. Maguluri, R. Srikant, and L. Ying, "Stochastic models of load balancing and scheduling in cloud computing clusters," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 702–710.

[5] S. El Kafhali and K. Salah, "Stochastic modelling and analysis of cloud computing data center," in *Proc. 20th Conf. Innov. Clouds, Internet Netw. (ICIN)*, Mar. 2017, pp. 122–126.

[6] K. Psychas and J. Ghaderi, "A theory of auto-scaling for resource reservation in cloud services," *Stochastic Syst.*, vol. 12, no. 3, pp. 227–252, Sep. 2022.

[7] B. Berg, M. Harchol-Balter, B. Moseley, W. Wang, and J. Whitehouse, "Optimal resource allocation for elastic and inelastic jobs," in *Proc. 32nd ACM Symp. Parallelism Algorithms Architectures*, Jul. 2020, pp. 75–87.

[8] S. Hu, X. Chen, W. Ni, X. Wang, and E. Hossain, "Modeling and analysis of energy harvesting and smart grid-powered wireless communication networks: A contemporary survey," *IEEE Trans. Green Commun. Netw.*, vol. 4, no. 2, pp. 461–496, Jun. 2020.

[9] T. H. Kim, H. Shin, K. Kwag, and W. Kim, "A parallel multi-period optimal scheduling algorithm in microgrids with energy storage systems using decomposed inter-temporal constraints," *Energy*, vol. 202, Jul. 2020, Art. no. 117669.

[10] L. Lv et al., "Contract and Lyapunov optimization-based load scheduling and energy management for UAV charging stations," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 3, pp. 1381–1394, Sep. 2021.

[11] M. Rahdar, L. Wang, and G. Hu, "A tri-level optimization model for inventory control with uncertain demand and lead time," *Int. J. Prod. Econ.*, vol. 195, pp. 96–105, Jan. 2018.

[12] O. Ben-Ammar, B. Bettayeb, and A. Dolgui, "Optimization of multi-period supply planning under stochastic lead times and a dynamic demand," *Int. J. Prod. Econ.*, vol. 218, pp. 106–117, Dec. 2019.

[13] H. Wei et al., "PressLight: Learning Max pressure control to coordinate traffic signals in arterial network," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, vol. 1, 2019, pp. 1290–1298.

[14] A. Braverman, J. G. Dai, X. Liu, and L. Ying, "Empty-car routing in ridesharing systems," *Oper. Res.*, vol. 67, no. 5, pp. 1437–1452, Sep. 2019.

[15] A. Braverman, J. G. Dai, X. Liu, and L. Ying, "Fluid-model-based car routing for modern ridesharing systems," in *Proc. ACM SIGMETRICS/Int. Conf. Meas. Model. Comput. Syst.*, Jun. 2017, pp. 11–12.

[16] V. Tsibonis, L. Georgiadis, and L. Tassiulas, "Exploiting wireless channel state information for throughput maximization," in *Proc. 22nd Annu. Joint Conf. IEEE Comput. Commun. Societies (INFOCOM)*, vol. 1, Mar. 2003, pp. 301–310.

[17] B. Sadiq and G. de Veciana, "Throughput optimality of delay-driven MaxWeight scheduler for a wireless system with flow dynamics," in *Proc. 47th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2009, pp. 1097–1102.

[18] S. Liu, L. Ying, and R. Srikant, "Throughput-optimal opportunistic scheduling in the presence of flow-level dynamics," *IEEE/ACM Trans. Netw.*, vol. 19, no. 4, pp. 1057–1070, Aug. 2011.

[19] M. J. Neely, "Order optimal delay for opportunistic scheduling in multi-user wireless uplinks and downlinks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 5, pp. 1188–1199, Oct. 2008.

[20] M. J. Neely, "Delay analysis for max weight opportunistic scheduling in wireless systems," *IEEE Trans. Autom. Control*, vol. 54, no. 9, pp. 2137–2150, Sep. 2009.

[21] L. Huang, S. Moeller, M. J. Neely, and B. Krishnamachari, "LIFO-backpressure achieves near-optimal utility-delay tradeoff," *IEEE/ACM Trans. Netw.*, vol. 21, no. 3, pp. 831–844, Jun. 2013.

[22] L. Huang and M. J. Neely, "Utility optimal scheduling in processing networks," *Perform. Eval.*, vol. 68, no. 11, pp. 1002–1021, Nov. 2011.

[23] M. J. Neely, "Delay-based network utility maximization," *IEEE/ACM Trans. Netw.*, vol. 21, no. 1, pp. 41–54, Feb. 2013.

[24] A. Gaddam, T. Wilkin, M. Angelova, and J. Gaddam, "Detecting sensor faults, anomalies and outliers in the Internet of Things: A survey on the challenges and solutions," *Electronics*, vol. 9, no. 3, p. 511, 2020.

[25] M. R. Khan, B. Das, and B. B. Pati, "Channel estimation strategies for underwater acoustic (UWA) communication: An overview," *J. Franklin Inst.*, vol. 357, no. 11, pp. 7229–7265, Jul. 2020.

[26] M. Ashjaei, L. Lo Bello, M. Daneshtalab, G. Patti, S. Saponara, and S. Mubeen, "Time-sensitive networking in automotive embedded systems: State of the art and research opportunities," *J. Syst. Archit.*, vol. 117, Aug. 2021, Art. no. 102137.

[27] J. Chen, F. Qian, and B. Li, "Enhancing quality of experience for collaborative virtual reality with commodity mobile devices," in *Proc. IEEE 42nd Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2022, pp. 1018–1028.

[28] M. J. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems* (Synthesis Lectures on Learning, Networks, and Algorithms), vol. 3. Morgan & Claypool, 2010, pp. 1–211.

[29] S. Krishnasamy, A. Arapostathis, R. Johari, and S. Shakkottai, "On learning the $c\mu$ rule in single and parallel server networks," in *Proc. 56th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Oct. 2018, pp. 153–154.

[30] T. Choudhury, G. Joshi, W. Wang, and S. Shakkottai, "Job dispatching policies for queueing systems with unknown service rates," in *Proc. 22nd Int. Symp. Theory, Algorithmic Found., Protocol Design Mobile Netw. Mobile Comput.*, Jul. 2021, pp. 181–190.

[31] S. Krishnasamy, R. Sen, R. Johari, and S. Shakkottai, "Learning unknown service rates in queues: A multiarmed bandit approach," *Oper. Res.*, vol. 69, no. 1, pp. 315–330, Jan. 2021.

[32] W.-K. Hsu, J. Xu, X. Lin, and M. R. Bell, "Integrated online learning and adaptive control in queueing systems with uncertain payoffs," *Oper. Res.*, vol. 70, no. 2, pp. 1166–1181, Mar. 2022.

[33] Z. Yang, R. Srikant, and L. Ying, "Learning while scheduling in multi-server systems with unknown statistics: MaxWeight with discounted UCB," in *Proc. 26th Int. Conf. Artif. Intell. Statist.*, in Proceedings of Machine Learning Research, vol. 206, F. Ruiz, J. Dys, and J.-W. van de Meent, Eds., 2023, pp. 4275–4312. [Online]. Available: https://proceedings.mlr.press/v206/yang23d.html

[34] B. Liu, Q. Xie, and E. Modiano, "RL-QN: A reinforcement learning framework for optimal control of queueing systems," *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 7, no. 1, pp. 1–35, Aug. 2022.

[35] B. S. Pavse, M. Zurek, Y. Chen, Q. Xie, and J. P. Hanna, "Learning to stabilize online reinforcement learning in unbounded state spaces," 2023, *arXiv:2306.01896*.

[36] L. Tassiulas and A. Ephremides, "Dynamic server allocation to parallel queues with randomly varying connectivity," *IEEE Trans. Inf. Theory*, vol. 39, no. 2, Mar. 1993.

[37] X. Fu and E. Modiano, "Joint learning and control in stochastic queueing networks with unknown utilities," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 51, no. 1, pp. 77–78, Jun. 2023.

[38] X. Fu and E. Modiano, "Optimal routing to parallel servers with unknown utilities—Multi-armed bandit with queues," *IEEE/ACM Trans. Netw.*, vol. 31, no. 3, pp. 1997–2012, 2022.

[39] Q. Liang and E. Modiano, "Minimizing queue length regret under adversarial network models," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 46, no. 1, pp. 31–32, Jan. 2019.

[40] E. Hazan, "Introduction to online convex optimization," *Found. Trends Optim.*, vol. 2, nos. 3–4, pp. 157–325, 2016.

[41] M. J. Neely, S. T. Rager, and T. F. La Porta, "Max weight learning algorithms for scheduling in unknown environments," *IEEE Trans. Autom. Control*, vol. 57, no. 5, pp. 1179–1191, May 2012.

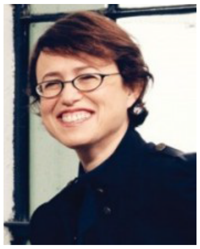
[42] S. Krishnasamy, P. T. Akhil, A. Arapostathis, R. Sundaresan, and S. Shakkottai, "Augmenting max-weight with explicit learning for wireless scheduling with switching costs," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2501–2514, Dec. 2018.

[43] T. Stahlbuhk, B. Shrader, and E. Modiano, "Learning algorithms for scheduling in wireless networks with unknown channel statistics," in *Proc. 18th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, Jun. 2018, pp. 31–40.

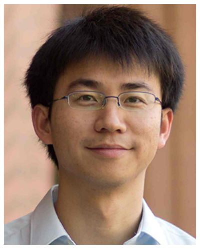
[44] P. Auer, "Using confidence bounds for exploitation-exploration trade-offs," *J. Mach. Learn. Res.*, vol. 3, pp. 397–422, Nov. 2002.

[45] R. T. Rockafellar, *Convex Analysis*. Princeton, NJ, USA: Princeton Univ. Press, 2015.

**Jiatai Huang** was born in Hangzhou, China, in May 1996. He received the Ph.D. degree in computer science and technology from the Institute for Interdisciplinary Information Sciences (IIIS), Tsinghua University, Beijing, China, in 2023, under the supervision of Prof. Longbo Huang. His research interests include multi-armed bandit algorithm design and analysis.



**Leana Golubchik** (Senior Member, IEEE) is currently the Stephen and Etta Varra Professor of electrical and computer engineering (with a joint appointment in computer science) with the University of Southern California (USC). She is also the Director of the Women in Science and Engineering (WiSE) Program. Prior to that, she was a Faculty Member with the University of Maryland and Columbia University. Her research interests include the design and evaluation of large-scale distributed systems, including hybrid clouds and data centers, and their applications in data analytics, machine learning, and privacy. She is a member of the IFIP WG 7.3 and a Fellow of AAAS. She is the Editor-in-Chief of *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*.



**Longbo Huang** (Senior Member, IEEE) has held visiting positions at the LIDS Laboratory, MIT, CUHK, Bell-Labs France, and Microsoft Research Asia (MSRA). He was a Visiting Scientist with the Simons Institute for the Theory of Computing in Fall 2016. He is currently a Professor with the Institute for Interdisciplinary Information Sciences (IIIS), Tsinghua University, Beijing, China. He is an ACM Distinguished Scientist, a CCF Distinguished Member, an IEEE ComSoc Distinguished Lecturer, and an ACM Distinguished Speaker. He received the Outstanding Teaching Award from Tsinghua University in 2014 and the Google Research Award and the Microsoft Research Asia Collaborative Research Award in 2014. He was selected into the MSRA StarTrack Program in 2015. He won the ACM SIGMETRICS Rising Star Research Award in 2018. He serves/served on the editorial board for IEEE Journal on Selected Areas in Communications, IEEE Transactions on Communications, *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, IEEE/ACM Transactions on Networking, *Performance Evaluation* (Elsevier), and IEEE Transactions on Pattern Analysis and Machine Intelligence.