Decentralized Scheduling with QoS Constraints: Achieving O(1) QoS Regret of Multi-Player Bandits

Qingsong Liu[1], **Zhixuan Fang**[1,2] *

[1] IIIS, Tsinghua University, Beijing, China
[2] Shanghai Qi Zhi Institute, Shanghai, China
liu-qs19@mails.tsinghua.edu.cn, zfang@mail.tsinghua.edu.cn

Abstract

We consider a decentralized multi-player multi-armed bandit (MP-MAB) problem where players cannot observe the actions and rewards of other players and no explicit communication or coordination between players is possible. Prior studies mostly focus on maximizing the sum of rewards of the players over time. However, the total reward maximization learning may lead to imbalanced reward among players, leading to poor Quality of Service (QoS) for some players. In contrast, our objective is to let each player n achieve a predetermined expected average reward over time, i.e., achieving a predetermined level of QoS. We develop a novel decentralized MP-MAB algorithm to accomplish this objective by leveraging the methodology of randomized matching. We prove that our decentralized algorithm can ensure that all players have an $O(1)$ QoS regret. We also reveal an analog between our MP-MAB model and the online wireless queuing systems, which builds a connection between QoS in MP-MAB learning and stability in queuing theory.

Introduction

Decentralized multi-player multi-armed bandits (MP-MAB) is a promising framework for solving various problems in computer science and operations research, which has seen emerging interest in recent years. In this framework, there are N players competing for pulling K arms over T rounds. Players cannot observe the actions and received rewards of other players, and no explicit communication or coordination between players is possible. One primary motivation for studying this framework is distributed resource allocation (Hanawal and Darak 2018), as coordinating a large number of users in a centralized manner is infeasible and conflicts will arise when multiple users compete for the same resource. The goal of MP-MAB algorithms is to provide a decentralized way to learn how to share these resources optimally in an online manner.

Towards this goal, a common objective is the sum of rewards of the players over time, which has received the vast majority of attention in the literature (Boursier and Perchet 2019; Wang et al. 2020a,b; Tibrewal et al. 2019b; Boursier et al. 2020; Shi et al. 2021) (See the latest survey (Boursier and Perchet 2022) for reference). However, in broader network scenarios, this objective might result in some users being allocated only the worst resource, i.e., the maximal sum of rewards assignment might starve some users (Fang et al. 2023; Liu et al. 2022). Thus, one severe drawback of this objective is that it has no individual performance guarantees. In many applications, each user seeks to have a desired Quality of Service (QoS), and the objective of the sum of rewards fails to achieve this goal. This motivates our work.

While several very recent works (Bistritz et al. 2020; Shi et al. 2021) focus on the fairness objectives (e.g., max-min fairness, proportional fairness) among the accumulated rewards of players, which exhibit some level of QoS guarantees for all players (a fair allocation may prevent some players from getting their best arm to significantly improve the allocation for less fortunate players), but obviously the policy with these fairness objectives does not promise to meet all the player-specific QoS requirements. In fact, these fairness metrics and the sum of rewards are all specific instances of a celebrated notion known as α-fairness (Mo and Walrand 2000): $\max \sum_n x_n^{1-\alpha}/1-\alpha$, where x_n is the accumulated reward of player n. This is because $\alpha \to 1$ yields proportional fairness, $\alpha = 0$ yields sum of rewards, and $\alpha = \infty$ yields the case of max-min fairness. Hence, these metrics can be optimized in a similar manner (Bistritz and Leshem 2018), as there exists an optimal static allocation/matching between the players and arms over the entire learning process, and the high-level idea of their algorithms is to identify this optimal matching via elimination. Some bandit works have studied corresponding objectives that can also potentially exhibit some level of fairness (Darak and Hanawal 2019; Bar-On and Mansour 2019). However, none of them successfully achieve the decentralized cooperation among players to guarantee stringent per-player QoS constraints.

In this work, we make progress in the aforementioned problems for decentralized MP-MAB. Instead of optimizing the sum of rewards or fairness metrics, our objective is to guarantee that the QoS, i.e., empirical average reward, of each player is at least a certain value. We consider the general but challenging setting where arm rewards can differ across players. Our model of decentralized MP-MAB involves new technical challenges that do not arise in the case of optimizing the sum of rewards or fairness metrics. As pre-

*Corresponding author: Zhixuan Fang.

viously mentioned, the optimal static allocation or matching between players and arms always exists for these objectives, and the next thing to do is identify one of these optimal matchings through elimination. However, this is not the case when aiming for individual QoS guarantees since there may not exist optimal matchings between players and arms that satisfy all players' QoS simultaneously. This complicates the decentralized learning process, as players cannot agree on a specific matching to play but should pull arms in a round-robin manner avoiding collisions, which is difficult to do when no communication is allowed. Moreover, our model allows for heterogeneous arm means and QoS constraints among players, and players do not know each other's arm means and QoS values. This makes it difficult for each player to determine whether the current round-robin strategy can satisfy others' QoS requirements, adding a new layer of difficulty to the decentralized cooperation among players to achieve their target QoS. Therefore, due to these new technical challenges, achieving QoS guarantees for every player in a decentralized manner requires a different and novel algorithm and analysis.

Contributions. Our main contributions in this paper are summarized as follows.

(a) We study a new objective for decentralized MP-MAB model that expands the application scope of MP-MAB by allowing all players to enjoy a required QoS. We build a connection between the QoS in decentralized learning and stability in queuing by showing that the MP-MAB game in our model behaves like an online queuing system in wireless networks, and any sublinear QoS regret achieving algorithm can also stabilize the online queuing system.

(b) We design a novel decentralized algorithm that ensures all players have an $O(1)$ QoS regret and further guarantees the strong stability of the corresponding queuing system, given the prior knowledge of system capacity Δ. Even without prior knowledge of Δ, our algorithm still achieves $O(\log T)$ QoS regret and maintains mean rate stability in the corresponding queuing system.

Other related works. Beyond multi-player bandits, our work is also related to the literature in learning-aided scheduling (Walton and Xu 2021; Chen, Dong, and Shi 2020; Krishnasamy et al. 2016, 2018, 2021; Choudhury et al. 2021; Stahlbuhk, Shrader, and Modiano 2021; Liu and Fang 2023; Liu and Xu 2024) and learning-aided queuing (Gaitonde and Tardos 2020, 2021; Sentenac, Boursier, and Perchet 2021; Gaitonde and Tardos 2023; Fu, Hu, and Lin 2022; Bistritz and Bambos 2022; Freund, Lykouris, and Weng 2022; Krishnasamy et al. 2016, 2021).

Our Model

We study a stochastic multi-player game played by a set of N players $\mathcal{N} = \{1, ..., N\}$ over a finite time horizon T. Each player cannot communicate with other players and faces with a common set of K arms denoted by $\mathcal{K} = \{1, ..., K\}$. We assume that $K \geq N$, since otherwise we can simply add dummy arms with 0 reward. The horizon T is unknown a priori to any player, and is considered to be much larger than $\max\{N, K\}$ as we assume that the game is played for a long time. At each round t, all players simultaneously pick one arm to play. We denote by $a_n(t)$ the arm that player n chooses at round t, and the action profile (vector of arms selected) at round t is $\boldsymbol{a}(t) \in [K]^N$. Players do not know which arms the other players chose, and need not even know the number of players N.

Rewards setting. We assume that, when multiple players choose the same arm, none of them can obtain a reward due to collision. We denote by $\eta_i(\boldsymbol{a})$ the no-collision indicator of arm i with respect to the action profile $\boldsymbol{a}$ such that $\eta_i(\boldsymbol{a}) = 0$ if $|\mathcal{N}_i(\boldsymbol{a})| > 1$, and $\eta_i(\boldsymbol{a}) = 1$ otherwise, where $\mathcal{N}_i(\boldsymbol{a}) = \{n \,|\, a_n = i\}$ is the set of players that chose arm i in action profile $\boldsymbol{a}$. Then for each player n, the instantaneous reward of hers at round t is $S_t^n = r_{n,a_n(t)}(t) \cdot \eta_{a_n(t)}(\boldsymbol{a}(t))$, where $r_{n,a_n(t)}(t)$ is a random reward that has a continuous distribution on $[0, 1]$. The reward sequence of arm i for player n, $\{r_{n,i}(t)\}_{t=1}^T$, is i.i.d. with an unknown expectation of $\mu_{n,i}$. We consider the heterogeneous setting where $\mu_{n,i}$ may not equal $\mu_{m,i}$ when $m \neq n$. An immediate example for the above collision reward model is wireless channel allocation, where the transmission of one user creates interference for other users on the same channel and causes all transmissions to fail. Moreover, the collision model applies to resource allocation scenarios where the resources are discrete items that cannot be shared.

Feedback setting. We assume that at each round t, each player n can observe her reward S_t^n together with the collision indicator $\eta_{a_n(t)}(\boldsymbol{a}(t))$. Note that this makes sense as it is natural in many applications (e.g., SNR wireless networks (Atapattu, Tellambura, and Jiang 2011)) that the probability for zero reward in a non-collision is zero. Thus, receiving zero reward means there exists a collision, which is equivalent to the setting where each player can observe their collision indicator in addition to their reward. Moreover, in the context of cellular networks, the user (transmitter) can receive an ACK/NACK signal after each transmission, which can be used to determine if a collision has occurred.

Objective. Here we introduce the objective for our model. Unlike most literature on decentralized MP-MAB that aims to maximize the total reward, our goal is to let every player n achieve at least a target QoS value γ_n. More precisely, we aim to design a decentralized policy such that each player can guarantee her QoS requirement in expectation, i.e.,

$$\mathbb{E}[S_t^n] \geq \gamma_n, \ \forall t \in [T], \ n \in [N], \tag{1}$$

where the expectation is over the randomness of rewards and policy. We emphasize that our model is fully decentralized, i.e., every player cannot communicate with others and use extra information made by others to make her decisions, and players do not know each other's QoS values.

The leading example for this objective is in wireless networks, where a self-organizing network needs to guarantee the demand throughput to all users. In this example, our decentralized policy should learn over time how to assign channels (arms) to users (players) such that the pre-defined QoS requirements are satisfied for all users. Another example is allocating each user the required computation resources for the task they are running. Regarding the objective (1), we adopt QoS regret as our performance metric that defined as follows:

$$R(T) = \sum_{t=1}^{T} \max_n \left(\gamma_n - \mathbb{E}[S_t^n]\right)^+, \tag{2}$$

where $(x)^+$ denotes $\max\{x, 0\}$. A meaningful policy should produce at least sublinear QoS regret performance, i.e., $R(T)/T \to 0$, and would be ideal if $R(T) = O(1)$, i.e., *bounded* QoS regret. Of course, we cannot hope the bounded QoS regret is possible unless there exists a centralized algorithm that can make such a guarantee. We thus first understand what conditions the QoS requirements $\boldsymbol{\gamma}$ should satisfy for the players to guarantee their QoS requirements under centralized coordination.

To formally state these conditions, consider the players make decisions via a centralized method. We denote $P_{n,i}$ as the fraction of rounds or probability in which the centralized controller schedules player n to pull arm i, and these fractions form the set $\Phi = \big\{P : \sum_{i=1}^{K} P_{n,i} = 1; \sum_{n=1}^{N} P_{n,i} \leq 1; P_{n,i} \geq 0\big\}$. As each pull from player n on arm i can yield an expected reward of $\mu_{n,i}$, the expected accumulated QoS of player n under this centralized controller is $T\sum_{i=1}^{K} \mu_{n,i} P_{n,i}$. Meanwhile, for each player n, the required QoS during the time horizon T is $T \cdot \gamma_n$. To guarantee the QoS constraints, we should ensure that the obtained accumulated reward of player n is larger than her required QoS, i.e., $T\gamma_n < T\sum_{i=1}^{K} \mu_{n,i} P_{n,i}$. This motivates defining the capacity of our MP-MAB game as follows,

$$\Delta = \max_{P \in \Phi} \min_{n} \big(\sum_{i=1}^{K} P_{n,i} \cdot \mu_{n,i} - \gamma_n \big) > 0. \tag{3}$$

If $\Delta < 0$, it means that no matter what the central controller's policy is, there exists at least one player whose QoS requirement γ_n is larger than her effective reward rate, and her QoS regret will grow over time. Indeed, even if $\Delta = 0$, i.e., there exists an n such that $\gamma_n = \sum_{i=1}^{K} P_{n,i}\mu_{n,i}$, the QoS regret still grows over time due to stochastic fluctuations. Hence, we require $\Delta > 0$ in our model. Note that the smaller Δ is, the more difficult it is to develop a decentralized online learning policy to satisfy all players' QoS constraints. Our work studies two regimes, depending on whether the prior knowledge of Δ is lacking.

The analog between our MP-MAB model and queuing model. Here we reveal that our MP-MAB game behaves like an online queuing system in wireless networks, and then investigate the connection between QoS in decentralized learning and stability in queuing theorem. To the begin, we formally introduce the following wireless discrete-time queuing system. The system consists of N sources competing for K channels to transmit packets to a common Base Station (BS). At each (discrete) time $t = 0, 1, \ldots$, the following occurs: **(1)** A new data packet will arrive at the source n's queue with a fixed, time-independent probability γ_n. We model the arrival event of time t as A_t^n, and we have $\mathbb{P}(A_t^n = 1) = \gamma_n,\ \forall t$. **(2)** Each source n chooses one channel $a_n(t) \in [K]$ to transmit the first data packet in her queue. If the queue is empty, she would send a null/hello packet on her chosen channel (The BS will examine all received packets and discard null/hello packets). **(3)** Each channel i is unreliable and experiences i.i.d. ON-OFF channel fading. At any time, the probability of channel i being ON between source n and BS is $\mu_{n,i}$ Here $\mu_{n,i}$ is heterogeneous w.r.t the source n due to differing channel quality for each source in the cognitive radio context. When more than one source transmits the packet (including null/hello packet) on the same channel, their transmission would fail due to interference. **(4)** If a data packet fails to transmit, the source would transmit it again in the next time until it succeeds. Each source not only receives feedback on whether her packet is transmitted successfully at her chosen channel, but also whether there exists other players choosing the same channel as she does via ACK/NACK signals, i.e., collision sensing.

We remark that $\boldsymbol{\mu}$ is unknown to all sources, and the source n knows its packet arrival rate γ_n (which can be quickly estimated), while other players do not. We denote $Q^n(t)$ as the number of untransmitted data packets of source n at the beginning of time t (before a new packet arrives). Formally, if S_t^n is the event indicator that source n clears a packet at time t and A_t^n is again the event indicator source n received a new packet at time t, then the dynamics of $Q^n(t)$ are as follows:

$$Q^n(t+1) = \max\{Q^n(t) + A_t^n - S_t^n, 0\},\ Q^n(0) = 0. \tag{4}$$

We note that S_t^n is reset to 0 if $Q_t^n + A_t^n = 0$ (i.e. source n had no data packets and didn't receive a new one at this round, so she only sends a null/hello packet on the chosen channel). Since coordinating a large number of sources in a centralized manner is infeasible, decentralized scheduling policies are desirable in practice. Hence, a common objective is to design a fully-decentralized algorithm to guarantee the stability of this queueing system in the following sense:

Definition 1 *The above system is **strongly stable** under some given dynamics if, for any $n \in [N]$, the random process $Q^n(t)$ satisfies $\mathbb{E}[Q^n(t)] \leq C$ for some absolute constant C that does not depend on t. That is, the queue length $Q^n(t)$ is bounded. We say the system is **(mean rate) stable** if for any $n \in [N]$, $\mathbb{E}[Q^n(t)]/t$ converges to zero almost surely, i.e., $\mathbb{E}[Q^n(t)] = o(t)$. That is, the growth of $Q^n(t)$ is sublinear.*

Intuitively, if we think of channel selecting as arm pulling, and successful transmission as receiving reward 1, any decentralized algorithm developed for our MP-MAB problem can also apply to the corresponding queuing system, as both QoS regret and queue length can be viewed as the accumulations of "unsatisfactory states". The following theorem reveals an analog between our MP-MAB model and the corresponding queuing model, bridging QoS in MP-MAB learning and stability in queuing theory. This underscores the significance of our studied MP-MAB model.

Theorem 1 *For our MP-MAB problem, any algorithm that achieves sublinear QoS regret can also stabilize the corresponding queuing system, wherein all sources follow this algorithm by replacing the arm pulling with channel selecting.*

We remark that an algorithm that achieves bounded QoS regret may not make the corresponding queuing system strongly stable. Moreover, the opposite direction of Theorem 1 does not hold, i.e., the algorithm that achieves strong stability for the queuing system may not guarantee a sublinear QoS regret for our MP-MAB model, as we can find some counter-examples to verify this. This helps strengthen the significance of our work.

Algorithms and Main Results

This section presents the proposed decentralized algorithms, accompanied with their performance bounds. We design our decentralized algorithms by using randomized matching/scheduling to allocate arms to players (a round-robin manner), which is a non-standard algorithm in MP-MAB literature. We first give a definition of the doubly stochastic matrix below, which can characterize any randomized scheduling policy between players and arms.

Definition 2 *A nonnegative square matrix $P \in [0,1]^{K\times K}$ is a **doubly stochastic matrix** if each row and column sums up to 1, i.e., $\sum_i P_{i,j} = \sum_j P_{i,j} = 1$. It is not difficult to verify that P lies in an $(K-1)^2$-dimensional affine subspace of K^2-dimensional Euclidean space. We denote by $\mathbb{B}_K$ the set of doubly stochastic matrices that belongs to $[0,1]^{K\times K}$, which is a convex polytope known as the **Birkhoff polytope**.*

Here $P_{i,j}$ can be viewed as the probability or fraction of rounds that player i selects arm j (we add fictitious or virtual players such that the number of players and arms are equal, where these fictitious players have no QoS requirements, so P is a square matrix). Since in the ideal case, each arm is only played by one player, each column of P sums up to 1. It is obvious that $\Delta = \max_{P\in\Phi} \min_n(\sum_{i=1}^K P_{n,i}\cdot\mu_{n,i} - \gamma_n) = \max_{P\in\mathbb{B}_K} \min_n(\sum_{i=1}^K P_{n,i}\cdot\mu_{n,i} - \gamma_n)$.

The doubly stochastic matrix P is **feasible** if $\sum_{i=1}^K P_{n,i}\cdot\mu_{n,i} > \gamma_n, \forall n$. Conversely, we call P **unfeasible**. Furthermore, if there exists an $\epsilon > 0$ such that $\sum_{i=1}^K P_{n,i}\cdot\mu_{n,i} \geq \gamma_n + \epsilon, \forall n$, we call P ϵ-**feasible**. To formalize our decentralized algorithm design, we still need a few definitions below.

Definition 3 *A **dominant mapping** is a function $\phi : \mathbb{R}^N \times \mathbb{R}^K \to \mathbb{B}_K$ which takes $(\boldsymbol{\gamma}, \boldsymbol{\mu})$ as input and returns a feasible doubly stochastic matrix P if it exists (and the identity matrix otherwise).*

Definition 4 *A **permutation matrix** $P \in [0,1]^{K\times K}$ is a square binary matrix that has exactly one entry of 1 in each row and each column and 0s elsewhere. Each such matrix represents an one-to-one matching between players and arms (where we pad with some virtual players since $K \leq N$). For convenience, we denote by $\mathfrak{S}_K$ the set of permutation matrices in $\mathbb{R}^{K\times K}$.*

Definition 5 *A **BvN (Birkhoff von Neumann) decomposition** is a function $\psi : \mathbb{B}_K \to \mathbb{P}(\mathbb{B}_K)$ that associates to any doubly stochastic matrix P a random variable $\psi(P)$ such that $\mathbb{E}[\psi(P)] = P$; stated otherwise, it expresses P as a convex combination of permutation matrices, i.e., there exist $\theta_1, ..., \theta_m \geq 0$, $\sum_{i=1}^m \theta_i = 1$, $m = (K-1)^2$ and permutation matrices $P_1, ..., P_m$ such that $\psi(P) = \sum_{i=1}^m \theta_i P_i$. Obviously, $(\theta_1, \theta_2, ..., \theta_m)$ forms a distribution over $\mathbb{B}_K$.*

Informally speaking, those definitions describe the policies players would follow in the decentralized case: a dominant mapping (Definition 3) gives adequate marginals that ensure zero QoS regret (since each player n obtains in expectation a reward of $\sum_{i=1}^K P_{n,i}\cdot\mu_{n,i}$ at each round, which is larger than γ_n by definition). And a BvN decomposition describes the associated coupling to avoid collisions while maintaining marginals. Explicitly, given a common $\phi(\boldsymbol{\gamma}, \boldsymbol{\mu})$, the joint decentralized strategy for each player is to draw a **shared random variable** $\omega \in \mathbb{R}$ and then choose arms according to the permutation $\psi(\phi(\boldsymbol{\gamma}, \boldsymbol{\mu}))(\omega)$. We can verify that such strategy can ensure that players select arms in a collision-free round-robin manner while satisfying all players' QoS requirements.

In our paper, the existence of a **shared randomness** between all players is assumed. We remark that this assumption is already satisfied by default in our model. Firstly, the players have the same time count. This is often referred to as synchronicity in MP-MAB. Secondly, all the players start at the same time $t = 1$, which is the static assumption in MP-MAB. Thus, players can simply use the time index as the common random seed. Studying the MP-MAB problem with neither synchronicity (Yang et al. 2021) nor a static assumption is left open for future work, which presents a significant challenge.

Design of a Dominant Mapping

Recall that the dominant mapping is a mapping that takes as input $(\boldsymbol{\gamma}, \boldsymbol{\mu})$ and returns, if possible, a doubly stochastic matrix P such that

$$\gamma_n < \textstyle\sum_{i=1}^K P_{n,i}\cdot\mu_{n,i} \text{ for all } n \in [N]. \tag{5}$$

Obviously, the number of mappings that satisfy this property is infinite. However, many of them are non-continuous and highly sensitive to the problem parameters (i.e., a small change in $(\boldsymbol{\gamma}, \boldsymbol{\mu})$ can lead to totally different outputs), which may impede the stability of the decentralized algorithm in the learning process. Thus, we need to design a regular mapping that satisfies (5) and is robust to the problem parameters $(\boldsymbol{\gamma}, \boldsymbol{\mu})$. Inspired by the log-barrier method in optimization, we achieve this by taking the minimizer of the following strongly convex program:

$$\phi(\boldsymbol{\gamma}, \boldsymbol{\mu}) = \arg\min_{P\in\mathbb{B}_K} \max_{n\in[N]} -\ln\Big(\sum_{i=1}^K P_{n,i}\cdot\mu_{n,i} - \gamma_n\Big) + \frac{1}{2K}\|P\|_2^2. \tag{6}$$

As required, ϕ always returns a doubly stochastic matrix P that satisfies $\gamma_n < \sum_{i=1}^K P_{n,i}\cdot\mu_{n,i}$ for all n if possible, since otherwise the objective is infinite (and in that case we let ϕ return the identity matrix). Moreover, the objective function is $1/K$-strongly convex, which guarantees some regularity of the dominant mapping, namely local-Lipschitzness, leading to Lemma 1 below.

Lemma 1 *For any $(\boldsymbol{\gamma}, \boldsymbol{\mu})$ with positive Δ, if $\|(\tilde{\boldsymbol{\gamma}} - \boldsymbol{\gamma}, \tilde{\boldsymbol{\mu}} - \boldsymbol{\mu})\|_\infty \leq c_1\Delta$, for any $c_1 < \frac{1}{2\sqrt{e}+2}$, we have*

$$\|\phi(\tilde{\boldsymbol{\gamma}}, \tilde{\boldsymbol{\mu}}) - \phi(\boldsymbol{\gamma}, \boldsymbol{\mu})\|_2 \leq c_2 K/\Delta \cdot \|(\tilde{\boldsymbol{\gamma}} - \boldsymbol{\gamma}, \tilde{\boldsymbol{\mu}} - \boldsymbol{\mu})\|_\infty, \tag{7}$$

where $c_2 = \frac{4}{(1-2c_1)/\sqrt{e}-2c_1}$. Moreover, denoting $\tilde{P} = \phi(\tilde{\boldsymbol{\gamma}}, \tilde{\boldsymbol{\mu}})$, it holds for any $n \in [N]$,

$$\gamma_n \leq \textstyle\sum_{i=1}^K \tilde{P}_{n,i}\cdot\mu_{n,i} - (\frac{1-2c_1}{\sqrt{e}} - 2c_1)\Delta. \tag{8}$$

The inequality (7) implies that if the estimation $(\tilde{\boldsymbol{\gamma}}, \tilde{\boldsymbol{\mu}})$ is more accurate compared with $(\boldsymbol{\gamma}, \boldsymbol{\mu})$, the returned doubly stochastic matrix $\tilde{P}$ is closer to the target doubly stochastic matrix P. Moreover, the inequality (8) implies that once the estimation error is below a threshold, the returned doubly stochastic matrix $\tilde{P}$ can strictly satisfy the QoS requirements of all players, with a margin of order Δ.

Remark 1 *An alternative dominant mapping, lacking the regularizing term in* (6)*, can also satisfy* (5)*. However, this choice of dominant mapping is problematic as it may not exhibit similar sensitivity properties as Lemma 1. Moreover, it makes the returned doubly stochastic matrix non-unique, leading to frequent collisions even when all players have the same estimation, i.e., harmful to the synchronization between players and stability of decentralized algorithm. Using a regularization term in* (6) *ensures the returned doubly stochastic matrix is unique that can avoid this problem.*

Algorithm for Known Δ Regime: AdeQoS

First, we consider the situation where Δ or its lower bound $\Delta' > 0$ (conveniently, we still use the symbol Δ to represent it) is known. Our decentralized QoS guaranteeing algorithm, **AdeQoS**, proceeds in epochs comprising three phases: exploration, (implicit) communication, and consensus. The exploration phase allows players to obtain enough samples of each arm to estimate their expected rewards. During the communication phase, players attempt to infer information about the arm statistics and others' QoS values through forced collisions. A similar structure of communication phase has been adopted by other heterogeneous MP-MAB algorithms (Mehrabian et al. 2020; Shi et al. 2021; Tibrewal et al. 2019a). After going through these two phases, players independently converge to an identical doubly stochastic matrix, obviating the need for a central entity. This allows for collision-free arm selection using BvN decomposition and shared randomness, making the decentralized problem resemble the centralized one. The remaining challenge is to verify if the current doubly stochastic matrix meets all players' QoS requirements without a central entity, which necessitates a consensus phase. The complete pseudocode of AdeQoS is provided in Algorithm 1.

Before running Algorithm 1, we first run an initialization procedure allowing each player to individually estimate the number of players N and assign herself of a unique index $n \in [N]$ in a decentralized manner. Then our exploration and communication protocols can start. Once this procedure completes, all players accurately learn the number of players N and each of them is assigned with a unique index between 1 and N. Several works (e.g., (Wang et al. 2020b)) present an example of such an initialization procedure and illustrate that its expected duration is about $O(K^3)$ rounds.

Exploration phase. In this phase, each player receives stochastic rewards from different arms and utilizes all the samples from previous and current exploration phases to estimate the expected reward of each arm. The k-th exploration phase proceeds as follows. Over time, players sequentially pull among all arms for a total of $c_0 K \log(k+1)$ rounds. Any arm is thus pulled $c_0 \log(k+1)$ times by each player. Using their internal rank/index, players start and remain in an orthogonal setting during the exploration phase, which is collision-free. Since for every player, each arm is pulled an equal number of times at any exploration phase, we denote by $T_e(k)$ the number of pulls for any arm during the first k exploration phases, where $T_e(k) = \sum_{j=1}^{k} c_0 \log(j+1) = \Omega(k \log k)$. For any player $n \in [N]$, we also denote by $\hat{\mu}_{n,i}^k$ the empirical reward expectation (sample mean) of arm i using all the exploration phases up to epoch k.

Algorithm 1: AdeQoS

Initialization: player index n, $u_{n,i} = s_{n,i} = 0,\ \forall i$.
For each epoch $k = 1, 2, \ldots$
  Exploration Phase:
  1. For the next $\lceil c_0 K \log(k+1) \rceil$ rounds:
     (a) Play arm $i = (n+t) \bmod K + 1$.
     (b) Receive $r_{n,i}(t)$ and then update $u_{n,i} = u_{n,i} + 1$ and $s_{n,i} = s_{n,i} + r_{n,i}(t)$.
  2. Estimate the expected reward of arm i as $\hat{\mu}_{n,i}^k = s_{n,i}/u_{n,i}$ for each $i = 1, ..., K$.
  Communication phase:
  1. If $n = 1$ (the player is leader) then run leader communication protocol
  2. Else $n \neq 1$ (the player is follower) run follower communication protocol
     // All players obtain the same $(\tilde{\boldsymbol{\gamma}}^k, \tilde{\boldsymbol{\mu}}^k)$ after communication phase
  Consensus phase: Run Algorithm 2
End For

(Implicit) Communication phase. In this phase, player 1 becomes the *leader*, and other players become *followers*. The leader executes additional computations, and communicates with the followers individually, while each follower communicates only with the leader. The communication phase starts at the same time for every player. During this phase, the default behavior of each player is to pull her *communication arm*. It is crucial that these communication arms are distinct: a simple way to do so is that players use their ranks as the communication arm. Suppose at a certain time the leader wants to send a sequence of b bits, $\{t_1, \ldots, t_b\}$, to the player i (player with rank i). Then during the next b rounds, for each $j = 1, 2, \ldots, b$, if $t_j = 1$, the leader pulls arm i; otherwise, she pulls her own communication arm 1, while all followers stick to their communication arms. Player i can thus reconstruct these b bits after these b rounds, by observing the collisions on arm i. The converse communication between follower i and the leader follows a similar approach. This trick (forced-collision) enables the transmission of (approximate) arm statistics and QoS values between any two players.

However in practice, players should quantize arm sample means and QoS values to avoid potentially infinite communication length upon communication, as these numbers are often decimal numbers in the range $[0, 1]$, while forced-collision is fundamentally a digital communication protocol. Specifically, for each arm i, player n truncates the sample mean $\hat{\mu}_{n,i}^k$ and sends only the $p_{n,i}^k$ most significant bits of this number to the leader, i.e., sends number $\tilde{\mu}_{n,i}^k$ that is the quantized version of $\hat{\mu}_{n,i}^k$, and the truncation/quantization error is at most $1/2^{p_{n,i}^k}$. Similarly, player n also truncates her QoS value γ_n and sends only the $p_{n,K+1}^k$ most significant bits of this number to the leader. In order to maintain syn-

chronization among players, communicated arm statistics and QoS values are all of length $m_k = \lceil -\log(\sqrt{k/T_e(k)}) \rceil$, i.e, $p^k_{n,i} = m_k,\ \forall n \in [N], i \in [K+1]$, where $\sqrt{k/T_e(k)}$ is exactly the adopted confidence bound for arm sample means at epoch k that will be introduced later. This choice of m_k can make sure the confidence interval of the truncated sample mean $\tilde{\mu}^k_{n,i}$ is at most $2\sqrt{k/T_e(k)}$, i.e., no more than twice the confidence interval of the sample mean $\hat{\mu}^k_{n,i}$ (The quantization error for γ_n is only at most $\sqrt{k/T_e(k)}$). The players' ranks are also useful for determining in which order communications should be performed, as the $N-1$ followers successively communicate their arm sample means and QoS values to the leader, followed by the leader successively communicating all arm sample means and QoS values (including her own and other players') to the $N-1$ followers. Thanks to this sequential communication and the identical communication length for each exchange, the players can communicate in turn without interfering with each other.

After a succession of two phases (exploration phase and communication phase), we have the following lemma to characterize the estimation and quantization errors.

Lemma 2 *Let $\{\hat{\mu}^k_{n,i}\}$ denote the empirical reward expectations using all the exploration phases up to epoch k, $\{\tilde{\mu}^k_{n,i}\}$ denote the the quantized version of $\{\hat{\mu}^k_{n,i}\}$ at epoch k, and $\{\tilde{\gamma}^k_n\}$ the quantized version of $\{\gamma_n\}$ at epoch k. Recall that $T_e(k) = c_0 \sum_{i=1}^k \log(i+1)$ then we have*

- $\mathbb{P}(|\tilde{\mu}^k_{n,i} - \mu_{n,i}| \geq 2\sqrt{k/T_e(k)}) \leq 2e^{-2k},\ \forall n, i.$
- $\mathbb{P}(|\tilde{\gamma}^k_n - \gamma_n| \geq 2\sqrt{k/T_e(k)}) = 0,\ \forall n.$

It is worth noting that the confidence interval for $\{\tilde{\mu}^k_{n,i}\}$, i.e., $2\sqrt{k/T_e(k)}$, diminishes as the epoch index k increases. Therefore, after a finite number of epochs, the precondition of Lemma 1 holds with high-probability. To facilitate our algorithm analysis, we call epoch k is **accepted** if $|\tilde{\mu}^k_{n,i} - \mu_{n,i}| \leq c_1\Delta$ and $|\tilde{\gamma}^k_n - \gamma_n| \leq c_1\Delta,\ \forall n, i$. Otherwise, epoch k is **rejected**. The following lemma provides an upper bound on the probability that epoch k is **accepted**.

Lemma 3 *Define the k-th epoch accepted event $G_k = \{$epoch k is accepted$\}$. Then there exists a constant k_0 such that for all $k \geq k_0 = 2e^{8/c_0 c_1^2 \Delta^2}$ we have $\mathbb{P}(G_k) \geq 1 - 2NKe^{-2k}$.*

Consensus phase. In our consensus phase, the decentralized players jointly play the doubly stochastic matrix $P^k = \phi(\tilde{\boldsymbol{\gamma}}, \tilde{\boldsymbol{\mu}})$, and judge whether the current P^k is feasible, i.e., satisfy all players' QoS requirements, then this knowledge is leveraged to expedite the consensus phase when P^k is unfeasible. To that end, our consensus phase employs an exploitation process with a random duration that terminates when one of the players no longer believes that the P^k being played can satisfy her QoS requirement. To enable this mechanism, each player collects all the reward samples when playing P^k during the k-th consensus phase to estimate her obtained QoS.

Algorithm 2: Consensus phase protocol

1: Construct doubly stochastic matrix $P^k \leftarrow \phi(\tilde{\boldsymbol{\gamma}}^k, \tilde{\boldsymbol{\mu}}^k)$.
2: Reset counter $R^k_n = 0$, $V^k_n = 0$. Set **d** = run (discontent indicator).
3: While (**d** = run) do
  1. For the next $\max\{k, N\}$ rounds:
    (a) Draw permutation matrix $\psi(P^k)(\omega) \sim P^k$.
    (b) Select the arm $a_n(t) = \psi(P^k)(\omega)(n)$ and receive $r_{n,a_n(t)}(t)$.
    (c) If $\eta_{a_n(t)}(\boldsymbol{a}(t)) = 1$ then Update $V^k_n = V^k_n + 1$ and $R^k_n = R^k_n + r_{n,a_n(t)}(t)$.
    (d) If $\eta_{a_n(t)}(\boldsymbol{a}(t)) = 0$ then update **d** = terminate
  2. Estimate current empirical QoS as $\tilde{O}^k_n = R^k_n / V^k_n$.
  3. Construct confidence interval C^k_n for $\tilde{O}^k_n$ as $1/(V^k_n)^{\frac{1}{4}}$.
  4. If $\tilde{O}^k_n \leq \gamma_n + c_3\Delta - C^k_n$ and **d** = run then update **d** = signal.
4: End While
5: If **d** = signal then for the next $\max\{k, N\}$ rounds:
  1. Draw permutation matrix $\psi(P^k)(\omega) \sim P^k$ and select the arm $\psi(P^k)(\omega)(t \bmod N + 1)$.

Note that the consensus phase of our algorithm has a random duration, making the length of each epoch also random. This poses a synchronization problem that one does not encounter when all epochs have fixed lengths. We address this by breaking the consensus phase into sub-phases of length $\max\{k, N\}$, and giving each player a discontent indicator *d* to track her status. Every player starts with *d* = *run*, indicating that she believes that each player n is receiving at least $\gamma_n + O(\Delta)$ level of QoS, where Δ is the capacity or its lower bound. If her estimate of her obtained QoS falls below $\gamma_n + O(\Delta)$, then she switches to *d* = *signal*. This means that in the next sub-phase, she will signal to every player to switch to *d* = *terminate*. A status of *d* = *terminate* occurs when a player experiences a collision during the consensus phase, which can only happen from a player with *d* = *signal* signaling her discontent. Note that *terminate* supersedes *signal*, in that if a player experiences a collision she will always set *d* = *terminate*, and if *d* = *terminate* she will never switch to *d* = *signal*. We can see that the players all terminate the consensus phase at the same time step, as no one terminates before the end of a sub-phase. Additionally, no player will terminate in a sub-phase where all players start with *d* = *run*. In the first sub-phase where a player sets *d* = *signal*, all players will terminate the consensus phase at the end of the subsequent sub-phase. For convenience, we call these sub-phases as *exploitation* sub-phase, except for the last sub-phase we called *signaling* sub-phase.

Remark 2 *Here we introduce the reason why we set the length of the sub-phase at k-th consensus phase to be $\max\{k, N\}$. On the one hand, the sub-phase length must be N or greater, otherwise, the players with* d = signal *cannot notify all other players within a sub-phase, leading to a synchronization problem. On the other hand, the sub-phase length must grow as the index of the epoch grows. This is because, when the sub-phase length of all consensus phases*

is less than a certain value c, for any epoch the probability that the consensus phase terminates is always greater than a constant $O(e^{-c})$ by Hoeffding's inequality, even if all players play an $O(\Delta)$-feasible P^k. An increasing sub-phase length can make sure the probability of consensus phase termination exponentially decrease with the increase of epoch index. Someone would argue that in this case, it also takes longer time (more rounds) in judging an unfeasible P^k. Intriguingly, the combination of Lemmas 2 and 3 shows that all players can play a $O(\Delta)$-feasible P^k with high-probability only after a limited number of epochs. This compensates for the longer time to judge an unfeasible P^k when the sub-phase length is increasing.

Performance and Analysis for AdeQoS

Here we provide the performance guarantee and corresponding analysis for AdeQoS. Theorem 2 below bounds the QoS regret of AdeQoS for our MP-MAB model.

Theorem 2 *For any $\Delta > 0$, if each player runs Algorithm 1 with $c_0 \geq 2$, then the QoS regret is **bounded**:*

$$R(T) \leq O(K^3 + N^3K^2 + N^4Ke^{16/\Delta^2}/\Delta^8).$$

Theorem 2 implies that the fraction of time during which the QoS unsatisfactory of each player is bounded. The exponential dependency on Δ of our regret bound arises from the fact that our algorithm can guarantee the underlying epoch is accepted with high-probability only after $k_0 = O(\exp(8/\Delta^2))$ epochs (Lemma 3). Despite yielding an exponential dependency on Δ, our numerical results suggest that the incurred QoS regret of our algorithm can be much smaller. We leave a tighter theoretical analysis for future work.

Proof sketch of Theorem 2. To prove Theorem 2, we start by showing that when epoch k is rejected, the probability that k-th consensus phase terminates within $\tau_0 = 16N/c_3^4\Delta^4$ sub-phases is at least $1 - O(1/k)$. Conversely, when epoch k is accepted, the probability that our algorithm gets stuck in the k-th consensus phase is no less then $1 - O(\exp(-\sqrt{k}))$. Equipped with Lemma 3 that epoch k is accepted with high-probability when k is sufficient large, we then deduce that with high probability, the k-th epoch never happens, as epochs with a large index only occur with exponentially vanishing probability. This means that at some point, the algorithm gets stuck in a consensus phase where a $O(\Delta)$-feasible P^k is played forever. Finally, we conduct a careful separation of QoS regret for each epoch, and establish upper bounds on the QoS regret incurred during accepted epochs and rejected epochs, respectively. Denote L_k as the number of exploitation sub-phases spent in the consensus phase of epoch k until one of the players sets *d=signal*. Our analysis is based on the observations that (a) when epoch k is rejected, then no more than $(L_k + 1)\max\{k, N\}$ QoS regret is incurred at k-th consensus phase, since the consensus phase terminates $\max\{k, N\}$ time-slots after a player has set *d=signal*; and (b) when epoch k is accepted, the k-th consensus phase incurs QoS regret only if it terminates, in which case we accumulate at most $\max\{k, N\}$ value of QoS regret from signaling time-slots. Accumulating the upper bounds on QoS regret of all epochs leads to the desired bound in Theorem 2.

Equipped with Theorem 1, Theorem 2 also implies that the AdeQoS algorithm developed for our MP-MAB model can stabilize the corresponding queuing system. Below, we show that AdeQoS can further make the corresponding queuing system strongly stable.

Theorem 3 *For any $\Delta > 0$, all sources in the wireless queuing system follow Algorithm 1 with $c_0 \geq 2$, then the queuing system is **strongly-stable**, i.e., $\forall n$ we have*

$$\mathbb{E}[Q^n(T)] \leq O\left(K^3 + \frac{NK}{\ln(1 + \Delta/\gamma_n)} + N^4K/\Delta^8 + N^3K^2\right).$$

The reason why our decentralized algorithm AdeQoS can strongly stabilize the corresponding queuing system is that it could guarantee asymptotically almost surely that the obtained QoS of each player strictly satisfies her requirement with a margin of order Δ, with the sacrifice that we require a prior knowledge of the (lower bound on) capacity Δ. In the later, we show that only mean rate stability can be achieved without knowing Δ. Our finite-time analysis of the queue length at epoch k is by connecting its dynamics with a random walk process given the probability pairs of going left and going right equal to $(\boldsymbol{\gamma}, P^k)$.

Adapt AdeQoS to Unknown Δ Regime

We now adapt AdeQoS policy to the situation where prior knowledge of Δ is lacking. In such case, we let players in AdeQoS only do exploitation at the k-th consensus phase, with a length of 2^k rounds. It means the consensus phase in AdeQoSreduces to exploitation phase without signaling procedure, resulting in players undergoing $O(\log T)$ epochs. We remark that the necessity to ascertain the feasibility of the current doubly stochastic matrix is obviated. This is due to the fact that the condition $\epsilon_n^k \geq \gamma_n$ will eventually be met after a finite number of epochs, as the estimations become more accurate over time. Consequently, no QoS regret will occur during the subsequent epochs' exploitation phases, which allows us to achieve $O(\log T)$ QoS regret and mean rate stability of the corresponding queuing model (by Theorem 1). We call this adaptation of AdeQoS as **A2deQoS**, and Theorem 4 below presents its performance bounds.

Theorem 4 *For any $\Delta > 0$, if each player runs A2deQoS, then the QoS regret is upper bounded by:*

$$R(T) \leq O(K^3 + N^2K\log T \cdot \log\log T + N^3K^2 + e^{8/\Delta^2}/\Delta^2).$$

*Furthermore, the queuing system would be **mean rate stable** when all sources in the system follow A2deQoS.*

Conclusion

We design decentralized algorithms for agents to achieve a predfined QoS in an MP-MAB game where they cooperate to learn how to allocate arms (thought of as resources). We introduce an analogy between QoS in decentralized learning and stability in queuing, and show that our developed algorithms can also stabilize the corresponding queuing system. Future research directions may involve refining theoretical performance bounds and achieving the stability of asynchronous queues.

Acknowledgements

This work is supported by Tsinghua University Dushi Program.

References

Atapattu, S.; Tellambura, C.; and Jiang, H. 2011. A mixture gamma distribution to model the SNR of wireless channels. *IEEE transactions on wireless communications*, 10(12): 4193–4203.

Bar-On, Y.; and Mansour, Y. 2019. Individual regret in cooperative nonstochastic multi-armed bandits. *Advances in Neural Information Processing Systems*, 32.

Bistritz, I.; Baharav, T.; Leshem, A.; and Bambos, N. 2020. My fair bandit: Distributed learning of max-min fairness with multi-player bandits. In *International Conference on Machine Learning*, 930–940. PMLR.

Bistritz, I.; and Bambos, N. 2022. Queue Up Your Regrets: Achieving the Dynamic Capacity Region of Multiplayer Bandits. In *Advances in Neural Information Processing Systems*.

Bistritz, I.; and Leshem, A. 2018. Distributed multi-player bandits-a game of thrones approach. In *Advances in Neural Information Processing Systems*, 7222–7232.

Boursier, E.; Kaufmann, E.; Mehrabian, A.; and Perchet, V. 2020. A Practical Algorithm for Multiplayer Bandits when Arm Means Vary Among Players. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*. Palermo, Sicily, Italy.

Boursier, E.; and Perchet, V. 2019. SIC-MMAB: synchronisation involves communication in multiplayer multi-armed bandits. In *Advances in Neural Information Processing Systems*, 12071–12080.

Boursier, E.; and Perchet, V. 2022. A survey on multi-player bandits. *arXiv preprint arXiv:2211.16275*.

Chen, J.; Dong, J.; and Shi, P. 2020. A survey on skill-based routing with applications to service operations management. *Queueing Systems*, 96: 53–82.

Choudhury, T.; Joshi, G.; Wang, W.; and Shakkottai, S. 2021. Job dispatching policies for queueing systems with unknown service rates. In *Proceedings of the Twenty-second International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 181–190.

Darak, S. J.; and Hanawal, M. K. 2019. Multi-player multi-armed bandits for stable allocation in heterogeneous ad-hoc networks. *IEEE Journal on Selected Areas in Communications*, 37(10): 2350–2363.

Fang, S.; Liu, Q.; Xu, L.; and Wu, W. 2023. Learning To Regularized Resource Allocation with Budget Constraints. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1–5. IEEE.

Freund, D.; Lykouris, T.; and Weng, W. 2022. Efficient decentralized multi-agent learning in asymmetric queuing systems. *arXiv preprint arXiv:2206.03324*.

Fu, H.; Hu, Q.; and Lin, J. 2022. Stability of Decentralized Queueing Networks Beyond Complete Bipartite Cases. In *International Conference on Web and Internet Economics*, 96–114. Springer.

Gaitonde, J.; and Tardos, É. 2020. Stability and learning in strategic queuing systems. In *Proceedings of the 21st ACM Conference on Economics and Computation*, 319–347.

Gaitonde, J.; and Tardos, E. 2021. Virtues of patience in strategic queuing systems. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, 520–540.

Gaitonde, J.; and Tardos, É. 2023. The Price of Anarchy of Strategic Queuing Systems. *Journal of the ACM*.

Hanawal, M. K.; and Darak, S. J. 2018. Multi-Player Bandits: A Trekking Approach. *arXiv preprint arXiv:1809.06040*.

Krishnasamy, S.; Akhil, P.; Arapostathis, A.; Sundaresan, R.; and Shakkottai, S. 2018. Augmenting max-weight with explicit learning for wireless scheduling with switching costs. *IEEE/ACM Transactions on Networking*, 26(6): 2501–2514.

Krishnasamy, S.; Sen, R.; Johari, R.; and Shakkottai, S. 2016. Regret of queueing bandits. *Advances in Neural Information Processing Systems*, 29.

Krishnasamy, S.; Sen, R.; Johari, R.; and Shakkottai, S. 2021. Learning unknown service rates in queues: A multiarmed bandit approach. *Operations Research*, 69(1): 315–330.

Liu, Q.; and Fang, Z. 2023. Learning to schedule tasks with deadline and throughput constraints. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*, 1–10. IEEE.

Liu, Q.; Xu, W.; Wang, S.; and Fang, Z. 2022. Combinatorial bandits with linear constraints: Beyond knapsacks and fairness. *Advances in Neural Information Processing Systems*, 35: 2997–3010.

Liu, Q.; and Xu, Z., Weihang andFang. 2024. Learning-based Scheduling for Information Gathering with QoS Constraints. In *IEEE INFOCOM 2024-IEEE Conference on Computer Communications*, 1–10. IEEE.

Mehrabian, A.; Boursier, E.; Kaufmann, E.; and Perchet, V. 2020. A practical algorithm for multiplayer bandits when arm means vary among players. In *International Conference on Artificial Intelligence and Statistics*, 1211–1221. PMLR.

Mo, J.; and Walrand, J. 2000. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on networking*, 8(5): 556–567.

Sentenac, F.; Boursier, E.; and Perchet, V. 2021. Decentralized Learning in Online Queuing Systems. *Advances in Neural Information Processing Systems*, 34: 18501–18512.

Shi, C.; Xiong, W.; Shen, C.; and Yang, J. 2021. Heterogeneous Multi-player Multi-armed Bandits: Closing the Gap and Generalization. *Advances in Neural Information Processing Systems*, 34: 22392–22404.

Stahlbuhk, T.; Shrader, B.; and Modiano, E. 2021. Learning algorithms for minimizing queue length regret. *IEEE Transactions on Information Theory*, 67(3): 1759–1781.

Tibrewal, H.; Patchala, S.; Hanawal, M. K.; and Darak, S. J. 2019a. Distributed learning and optimal assignment in multiplayer heterogeneous networks. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, 1693–1701. IEEE.

Tibrewal, H.; Patchala, S.; Hanawal, M. K.; and Darak, S. J. 2019b. Multiplayer Multi-armed Bandits for Optimal Assignment in Heterogeneous Networks. *arXiv preprint arXiv:1901.03868*.

Walton, N.; and Xu, K. 2021. Learning and information in stochastic networks and queues. In *Tutorials in Operations Research: Emerging Optimization Methods and Modeling Techniques with Applications*, 161–198. INFORMS.

Wang, P.-A.; Proutiere, A.; Ariu, K.; Jedra, Y.; and Russo, A. 2020a. Optimal Algorithms for Multiplayer Multi-Armed Bandits. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*. Palermo, Sicily, Italy.

Wang, P.-A.; Proutiere, A.; Ariu, K.; Jedra, Y.; and Russo, A. 2020b. Optimal algorithms for multiplayer multi-armed bandits. In *International Conference on Artificial Intelligence and Statistics*, 4120–4129. PMLR.

Yang, L.; Chen, Y.-Z. J.; Pasteris, S.; Hajiesmaili, M.; Lui, J.; and Towsley, D. 2021. Cooperative stochastic bandits with asynchronous agents and constrained feedback. *Advances in Neural Information Processing Systems*, 34: 8885–8897.