Multi-Agent Reinforcement Learning for a Random Access Game

Dongwoo Lee, *Student Member, IEEE*,
Yu Zhao, *Graduate Student Member, IEEE*,
Jun-Bae Seo, *Member, IEEE*, and Joohyun Lee, *Member, IEEE*

***Abstract*—This work investigates a random access (RA) game for a time-slotted RA system, where N players choose a set of slots of a frame and each frame consists of M multiple time slots. We obtain the pure strategy Nash equilibria (PNEs) of this RA game, where slots are fully utilized as in the centralized scheduling. As an algorithm to realize a PNE (Pure strategy Nash Equilibrium), we propose an Exponential-weight algorithm for Exploration and Exploitation (EXP3)-based multi-agent (MA) learning algorithm, which has the computational complexity of $O(NN_{\max}^2 T)$. EXP3 is a bandit algorithm designed to find an optimal strategy in a multi-armed bandit (MAB) problem that users do not know the expected payoff of each strategy. Our simulation results show that the proposed algorithm can achieve PNEs. Moreover, it can adapt to time-varying environments, where the number of players varies over time.**

***Index Terms*—Multi-armed bandit, nash equilibrium, non-cooperative game, random access.**

I. Introduction

As applications of the Internet-of-Things (IoT) devices become popular such as smart home, health care, and smart cities, the uplink random access (RA) system is important for a large number of IoT devices to transfer their data [1], [2]. In particular, since numerous IoT devices rarely and randomly generate small-sized packets, a coordinated resource allocation for each session incurs a large overhead of signaling. Thus, data transmission via RA seems reasonable in terms of transmitted data packets per signaling overhead [3]. However, collisions by simultaneous transmissions from multiple IoT devices and idleness of the channel without a single transmission are unavoidable in RA systems. This eventually results in throughput degradation. Accordingly, it is essential to design a retransmission algorithm for IoT devices to employ in order to maximize the throughput in the long run.

This work formulates a time-slotted uplink RA system as a RA game, where N IoT devices (or players) are interested in choosing a conflict-free slot in a frame of M slots. In the RA game, a frame is further divided into a number of subframes so that the players choose an action consisting of two parameters. One parameter is the size of a subframe (in slots) and the other is a slot in a subframe, to which the players transmit their packet. We derive the pure strategy Nash equilibria (PNEs) and mixed strategy Nash equilibrium (MNE) of the RA game, where each player among a total of N players chooses his own slot in a subframe of k slots. In this RA game, the players and the access point (AP) do not have the information on the number of players in the system a priori, and the number of players can even vary over time. However, we assume that the information on the minimum and the maximum number of contending players is available as a coarse estimation. Furthermore, we assume that the players do not cooperate nor communicate with each other and only the channel outcome at each slot is available.

As an algorithm for the players to deploy in order to realize PNEs while the exact population size N is not available, we propose a multi-agent learning algorithm based on the Exponential-weight algorithm for Exploration and Exploitation [4] (EXP3) for the multi-agent multi-armed bandit (MAB) problem, where an action of our RA game can be viewed as an arm for each player to pull. The simulation results show that the proposed algorithm achieves better throughput than ϵ-greedy [5] and upper-confidence bound (UCB) [6] in various scenarios, and can adapt to environmental changes.

In addressing some advantages of the game-theoretic approach, let us recall that in slotted RA systems, slotted ALOHA (S-ALOHA) and the splitting algorithms are frequently considered. In S-ALOHA system, a backoff algorithm for retransmitting a collided packet is based on random retransmission; that is, either a retransmission probability for each slot or a window-based backoff algorithm, e.g., exponential or uniform backoff window, can be used. If a retransmission probability, say p, in S-ALOHA systems is controlled such that an opportunity of slot use is evenly distributed over N users, i.e., $p = \frac{1}{N}$, its maximum throughput of 0.368 (packets/slot) can be achieved for large N. On the other hand, in splitting algorithms, a sequence of retransmissions is determined based on a coin-toss by users, which forms a tree. Over time, collisions from the last leaf to the root are resolved. So far, the first-come-first-serve (FCFS) splitting algorithm is known the best, which can achieve 0.487 (packets/slot), while there may be a possibility of deadlock [7]. In comparison, the RA game formulation enables us to design a learning algorithm for the users (or players); that is, the players learn a throughput-optimal action as a favorite over time. This eventually converges to PNEs in the RA game. We shall see that the system with the optimal action behaves like a genie-aided time-division multiple access (TDMA) system, where all users are scheduled without a collision by an omniscient genie.

The prior work can be summarized as follows:

1) Game Theoretical Approaches for Random Access: The random access system has been modeled as a game in several papers [8]–[10], which characterized NEs in several game models for random access. In [8], the mixed strategy is characterized based on the reward and cost in the S-ALOHA game. It is found that the maximum throughput of 0.368 is achieved at the MNEs. In [9], the MNEs were found in the S-ALOHA game, where players calculated their transmission probabilities and announced them to other players. In addition, MNEs is obtained in [10], when the mixed strategy is the transmission probability as a function of collision cost.

2) Multi-Agent Multi-Armed Bandits: In [11], [12], the decentralized multi-user MAB is studied in the system, where users do not cooperate or communicate with other users and do not know the collision information. Particularly, a logarithmic regret is achieved in [11] if every user knows the exact number of users inside the system. The near-optimal throughput in the cognitive radio network is obtained in [12], where

Manuscript received 26 May 2021; revised 8 October 2021 and 21 April 2022; accepted 15 May 2022. Date of publication 23 May 2022; date of current version 15 August 2022. This work was supported by the Samsung Research Funding & Incubation Center of Samsung Electronics under Grant SRFC-TB1803-05. The review of this article was coordinated by Dr. Ai-Chun Pang. *(Dongwoo Lee and Yu Zhao are co-first authors.) (Corresponding authors: Jun-Bae Seo; Joohyun Lee.)*

Dongwoo Lee, Yu Zhao, and Joohyun Lee are with the Department of Electrical and Electronic Engineering, Hanyang University, Ansan 15588, South Korea (e-mail: ehddn0302@hanyang.ac.kr; zhaoyu0112@hanyang.ac.kr; joohyunlee@hanyang.ac.kr).

Jun-Bae Seo is with the Department of Information and Communication Engineering, Gyeongsang National University, Tongyeong 53064, South Korea (e-mail: jbseo@gnu.ac.kr).

Digital Object Identifier 10.1109/TVT.2022.3176722

users sense the channel before they choose their channel to transmit. In this paper, users learn the best action by only observing the reward, and our method achieves near-optimal throughput in our simulation results.

3) User Estimation in the Cell: The estimating method of the users inside the cell has been studied in [13]–[15]; the collision detection history from the medium is collected for Kalman filter in [14]; unsupervised neural network in [15], where the medium access control address and received signal strength from the received packet are used as a data set for an unsupervised learning algorithm to estimate the number of users in [13].

4) Random Access Channel Procedure: There are several studies for RA procedure in the IoTs [16]–[19]. The collision resolution scheme is proposed for adjusting the backoff indicator dynamically [16]. In [17], a mathematical model is introduced for the narrowband IoT network load to estimate and predict load changes. Dynamic binary countdown-access barring (DBCA) algorithm for the dynamic load-adaptive algorithm is proposed in [18]. [19] proposed three RA schemes that provide ultra-reliable low latency communication access for IoT applications.

II. Random Access System as a Game

A. RA System Model

In the RA system, time is divided into slots. Each slot is a constant length, which corresponds to one packet transmission time. A total of N players can (re)transmit their packets in a slot to the AP without coordination. When only one player (re)transmits its packet in a slot, its transmission is said to be successful. Just after each slot boundary, the AP broadcasts the channel outcome such as success or failure without an error. Notice that the AP and the players do not know the exact population size N, but we assume that the players can have the minimum $N_{\min}$ and the maximum $N_{\max}$ of the population size.

Let us define a *frame* as a group of slots. It consists of M slots, which is determined by the least common multiple (LCM) of $N_{\min}, N_{\min}+1, \ldots, N_{\max}$. These numbers will be used as the minimum and the maximum lengths of a subframe inside a frame. For example, when $N_{\min}=3$ and $N_{\max}=5$, a frame consists of $M=60$ slots. As in the exponential backoff algorithm, where the minimum and maximum window sizes are heuristically defined, $N_{\min}$ and $N_{\max}$ can play a similar role in this system.

B. RA Game Model

The RA game is denoted by $\mathcal{G}=(\mathcal{N},(A_n)_{n\in\mathcal{N}},(r_n)_{n\in\mathcal{N}})$, where $\mathcal{N}=\{1,2,\ldots,N\}$ denotes the set of players (or users) and A_n is the set of actions for player n to take. The payoff function of player n is denoted by $r_n: A\to\mathbb{R}$, where $A=A_1\times A_2\times\cdots\times A_N$ is the set of action vectors. An action $a_n\in A_n$ for the RA game is defined as slot indices in a frame, to which a player transmits its packet. More precisely, a frame is divided into subframes, each of which takes k slots for $k\in\{N_{\min},N_{\min}+1,\ldots,N_{\max}\}$. The number of subframes in a frame is M/k. An action is characterized by a duplet (k,l), where k and l denote the length of a subframe and the slot index in a frame to transmit a packet, respectively. Player n transmits its packet to the l-th slot of every subframe. Notice that since a subframe consists of k slots for $k\in\{N_{\min},N_{\min}+1,\ldots,N_{\max}\}$, l can take a value between 1 and k. For example, when a player takes an action $(k,l)=(4,2)$ for a frame of $M=420$ slots, there are 105 subframes, and the player transmits its packet to the second slot in every subframe of four slots. Then, a total number of actions for player n to have are $|A_n|=\sum_{\beta=N_{\min}}^{N_{\max}}\beta$ for $n\in\mathcal{N}$.

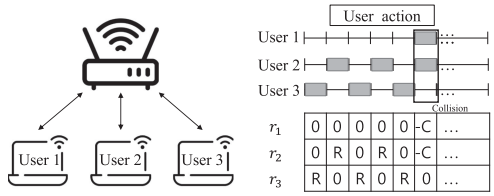

Fig. 1. Example of the time slotted random access problem when $N=3$.

TABLE I
The Payoff Matrix When $N=N_{\min}=N_{\max}=3$

Player 3: 1

Player 1 \ Player 2	1	2	3
1	$-C,-C,-C$	$-C,R,-C$	$-C,R,-C$
2	$R,-C,-C$	$-C,-C,R$	R,R,R
3	$R,-C,-C$	R,R,R	$-C,-C,R$

Player 3: 2

Player 1 \ Player 2	1	2	3
1	$-C,-C,R$	$R,-C,-C$	R,R,R
2	$-C,R,-C$	$-C,-C,-C$	$-C,R,-C$
3	R,R,R	$R,-C,-C$	$-C,-C,R$

Player 3: 3

Player 1 \ Player 2	1	2	3
1	$-C,-C,R$	R,R,R	$R,-C,-C$
2	R,R,R	$-C,-C,R$	$R,-C,-C$
3	$-C,R,-C$	$-C,R,-C$	$-C,-C,-C$

The mixed strategy of player n is denoted by $s_n=s_n(t)=(\pi_a, a\in A_n)$ such that $\sum_{a\in A_n}\pi_a=1$, which is a probability distribution over actions. In particular, a strategy s_n is said to be a pure strategy if $\pi_{a_n}=1$ for an action $a_n\in A_n$. The payoff function of player n in *each frame* is denoted by $r_n(s_n,s_{-n})$, where $s_{-n}=(s_1,\ldots,s_{n-1},s_{n+1},\ldots,s_N)$ denotes the strategies of all the players except player n. If player n is the only transmitting player in a slot, it makes a successful transmission and receives a positive payoff R for success in a slot. If more than one player transmit packets, then a collision occurs so that these players receive a negative payoff (or cost) $-C$,[1] e.g., wasted energy for a collision. Then, it shall get its payoff in a frame as

$$r_n=r_n(s_n,s_{-n})=iR-jC, \tag{1}$$

where i and j denote the number of successes and collisions in a frame, respectively.

In Fig. 1, we depict a system model of the time-slotted RA when $N=3$ on the left, and the corresponding payoff of the three players on the right. Table I shows a three-player game with $N_{\min}=N_{\max}=N=3$. The first table shows the payoff matrix when player 3 chooses the first slot in every three slots. The slot index for players 1 and 2 to choose is given in the column and row, respectively. The elements in each matrix denote the payoff that players 1, 2, and 3 obtain, respectively.

III. Game Analysis and Algorithms

A. Game Analysis

We examine the PNEs of this RA game in Theorem III.1 and MNE in Theorem III.2, respectively.

Theorem III.1: Consider the pure strategies of the players $(a_1^*,\ldots,a_N^*)$, where $a_n=(k,\sigma_n)$ for $\forall n\in\mathcal{N}$, $k=N$ and σ_n is the

[1]The priority between the players can be implicitly applied in our game model by setting the different values of R and C between the players in (1). In this paper, we assume that the values of R and C are equal for all players for simplicity.

n-th element of a permutation from the set $\{1, \ldots, N\}$. Accordingly, σ_n denotes a slot index every N slots for player n. This is a PNE in a frame so that there are $N!$ PNEs. At a PNE, each player takes a conflict-free slot of a subframe.

Proof: The PNEs in Theorem III.1 have $k = N$ but a different l for player $n \in \mathcal{N}$. Hence, every player transmits to his own l-th slot in every subframe and gets the payoff of R whenever it sends a packet successfully. This is shown in a three-player game in Table I, where every player gets the payoff of R.

If the other players follow a^*_{-n}, and player n does not follow a^*_n, player n always obtains a smaller payoff as he will experience a collision if $k \geq N$, or receives fewer successes when $k < N$. Therefore, a^*_n is the unique best response strategy to player n and the pure strategies in Theorem III.1 form a PNE. There are $N!$ possible choices of slot indices for all the players to transmit without a collision. In Table I, we can see six pure strategy NEs, i.e., 3!. Notice that the system throughput at these PNEs is 1 (packets/slot) as centralized scheduling like TDMA. □

Theorem III.2: Let us assume that either (i) $N_{\min} = N$ and $R \geq \frac{1-Q}{Q}C$, or (ii) $N_{\max} = N$ and $R \leq \frac{1-Q}{Q}C$, where $Q = (1 - \frac{1}{N})^{N-1}$. Consider the mixed strategies of players, $(s^*_1, \ldots, s^*_N)$, where each s^*_n is the mixed strategy such that $\pi_{a_n} = 1/N, \forall n \in \mathcal{N}$ for the pure strategies $a_1, \ldots, a_N$ in Theorem III.1, i.e., $k = N, l \in \{1, 2, \ldots, N\}$. This is a mixed strategy NE in each frame.

Proof: First, let us consider the condition (i). The proof for the condition (ii) is similar and omitted. The mixed strategy of each player in Theorem III.2 chooses the strategies used in Theorem III.1 with probability $\frac{1}{N}$ and does not play other strategies. The expected payoff for each player following the strategies of Theorem III.2 is

$$r_n(s^*_n, s^*_{-n}) = Q \times \frac{M}{N}R - (1 - Q) \times \frac{M}{N}C, \forall n \in \mathcal{N}, \quad (2)$$

where Q is the probability that all other players remain idle (i.e., do not transmit any packets). When each player chooses an action (N, l) for $l = \{1, 2, \ldots, N\}$ with probability $\frac{1}{N}$, the probability that the other players do not choose the action that player n chooses is $Q = (1 - \frac{1}{N})^{N-1}$. Consider any player n while the other players follow s^*_{-n}. If the player n chooses an action (k, l), the expected payoff of player n can be expressed by replacing N to k in (2). Since $N = N_{\min}$ and $R \geq \frac{1-Q}{Q}C$, the actions with $k = N_{\min}$ have the highest expected payoff as in (2). Thus, the player n's expected payoff is equal to or smaller than (2). The mixed strategy s^*_n is one of each player's best response strategies for the other players' strategy s^*_{-n}. Notice that the throughput of this mixed strategy NE is $(1 - \frac{1}{N})^{N-1}$, which is equivalent to the optimal throughput of S-ALOHA with access probability $\frac{1}{N}$. □

B. Proposed Algorithm

In order for the players to realize the PNEs of this RA game, we propose an EXP3-based multi-agent (EXP3-MA) learning algorithm. In general, EXP3-MA solves MAB problems by exploring the arms (here actions) in a way of getting the maximum payoff over time; that is, the EXP3-MA can realize PNEs after a sufficient learning period. Notice that each action available at the players corresponds to an arm of MAB. Beside EXP3-MA, two other algorithms such as ϵ-greedy and UCB algorithms can be exercised for MAB problems. However, the EXP3 algorithm is preferred since it achieves an order-optimal regret bound in single-agent adversarial bandit problems [20], compared to ϵ-greedy, and UCB algorithms. We present how these algorithms are applied to our RA game after expounding the EXP3-MA algorithm and compare the performances of three algorithms in Section IV.

EXP3-MA algorithm works as follows: In every frame t, each player $n \in \mathcal{N}$ calculates the probability of choosing action a denoted by $p_{n,a}(t)$, and samples an action $a_n(t)$ from the probability distribution $\mathbf{P}_n(t) = \{p_{n,1}(t), \ldots, p_{n,|A_n|}(t)\}$, shown in lines 4 and 5 in Algorithm 1. In line 4, $q_{n,a}(t)$ indicates the total estimated payoff of a up to frame t, which shall be updated at the end of each frame. It is notable that $p_{n,a}(t)$ is a function of $q_{n,a}(t)$, and $\sum_{a \in A_n} p_{n,a}(t) = 1$, while it is also an exponential weighting function. The function of $p_{n,a}(t)$ allows $\mathbf{P}_n(t)$ to quickly increase the probability of choosing an outstanding outcome while rapidly reducing the probability of poor ones, for all actions $a \in A_n$ tuned by parameter $\eta > 0$. For a large η, $p_{n,a}(t)$ concentrates on an action with the largest estimated payoff, which can be said that the action is exploited aggressively by player n. For a small η, $p_{n,a}$ becomes more uniform across actions so that player n can explore the actions evenly.

Algorithm 1: EXP3-MA.

1 Initialize: $q_{n,a}(1) = 0, \forall n \in \mathcal{N}$, $a \in A_n$
2 **for** $t = 1, 2, ..., T$ **do**
3 **for** $n = 1, 2, ..., N$ **do**
4 $p_{n,a}(t) = \frac{\exp(\eta q_{n,a}(t))}{\sum_{a' \in A_n} \exp(\eta q_{n,a'}(t))}, \forall a \in A_n$.
5 $a_n(t) = a$ with probability $p_{n,a}(t)$.
6 **end**
7 **for** $n = 1, 2, ..., N$ **do**
8 Receive payoff $r_n(a_n(t), a_{-n}(t))$.
9 Calculate $\hat{\mathbf{r}}_n(t)$, estimated payoffs of actions as in Eq. (3).
10 **for** $a \in A_n$ **do**
11 $q_{n,a}(t+1) = q_{n,a}(t) + \hat{r}_{n,a}(t)$.
12 **end**
13 **end**
14 **end**

After playing the RA game according to their actions, all the players get a payoff $r_n(a_n(t), a_{-n}(t))$ in line 8 of the Algorithm 1 at the end of the frame t. Then the players calculate an estimated payoff vector, denoted by $\hat{\mathbf{r}}_n(t) = \{\hat{r}_{n,1}(t), \ldots, \hat{r}_{n,|A_n|}(t)\}$ for all actions, in line 9 of the Algorithm 1 as follows:

$$\hat{r}_{n,a}(t) = \frac{r_n(a_n(t), a_{-n}(t))\mathbb{1}_{\{a_n(t)=a\}}}{p_{n,a}(t)}, \quad \forall a \in A_n, \quad (3)$$

where $\mathbb{1}_{\{a_n(t)=a\}}$ is the indication function that becomes 1 if $a_n(t) = a$; otherwise, zero.

Without dividing $r_n(a_n(t), a_{-n}(t))$ by $p_{n,a}(t)$ in (3), it can be expected that an action with a very low $p_{n,a}(t)$ is less likely chosen. To avoid this, the payoff r_n is divided by $p_{n,a}(t)$. This also guarantees that the conditional expectation of the estimated payoff associated with any fixed action $a_n(t) = a$ is equal to the actual payoff; that is,

$$\mathbb{E}_{t-1}[\hat{r}_{n,a}(t)|a_n(1), \ldots, a_n(t-1)] = \sum_{a' \in A_n} \hat{r}_{n,a}(t) \cdot p_{n,a'}(t)$$

$$= \hat{r}_{n,a}(t) \cdot p_{n,a}(t) = r_n(a_n(t), a_{-n}(t)). \quad (4)$$

While $a_n(t)$ is randomly chosen according to $p_{n,a}(t)$, $p_{n,a}(t)$ is influenced by the past actions made, $a_n(1), \ldots, a_n(t-1)$ [21]. Consequently, $\hat{r}_{n,a}(t)$ also gets influenced by $a_n(1), \ldots, a_n(t)$ as $p_{n,a}(t)$ is used to calculate $\hat{r}_{n,a}(t)$. Hence, the expectation of $\hat{r}_{n,a}(t)$ is taken with respect to $a_n(1), \ldots, a_n(t)$. Since we fixed the action $a_n(t) = a$, the only non-zero estimated payoff is $\hat{r}_{n,a}(t) \neq 0$, and the rest of the estimated payoffs for other actions are 0. Therefore, $\sum_{a' \in A_n} \hat{r}_{n,a}(t) \cdot p_{n,a'}(t)$ becomes $\hat{r}_{n,a}(t) \cdot p_{n,a}(t)$, and from the definition of $\hat{r}_{n,a}(t)$, $\hat{r}_{n,a}(t) \cdot p_{n,a}(t) = r_n(a_n(t), a_{-n}(t))$. Furthermore, (3) is an unbiased

estimate of the payoff, which means that expectation of $\hat{r}_{n,a}(t)$ is equal to $r_n(a_n(t), a_{-n}(t))$ as shown in (4). Hence, (3) is used to estimate the payoff of action a. In line 11 of the Algorithm 1, $\hat{r}_{n,a}(t)$ will be added to $q_{n,a}(t)$ at the end of frame t such that the player n's total estimated payoff until frame t is defined as $q_{n,a}(t) = \sum_{\tau=1}^{t-1} \hat{r}_{n,a}(\tau)$. Then, $q_{n,a^*}(t)$ of an optimal action a^* with the highest average payoff grows faster than other non-optimal actions, which in turn maximizes the probability to choose this optimal action, $p_{n,a^*}(t)$. In each frame, each player calculates the probability of each action and selects a transmission policy in lines 3∼6. We know that the total number of action for each player is $A = \sum_{i=N_{\min}}^{N_{\max}} i$, and therefore the maximum number of actions for each player is $\frac{(N_{\max}+N_{\min})(N_{\max}-N_{\min}+1)}{2} = O(N_{\max}(N_{\max}-N_{\min}))$. Hence, the computational complexity for lines 3∼6 is $O(NN_{\max}(N_{\max}-N_{\min})))$. Similarly, in lines 7∼13, the computational complexity is $O(NN_{\max}(N_{\max}-N_{\min})))$ as well. As a result, the computational complexity of Algorithm 1 is $O(NN_{\max}^2 T)$. In lines 1 and 4, there are two matrices $q_{n,a}$ and $p_{n,a}$ $\forall n \in \mathcal{N}$, $a \in A_n$. Thus, the maximum size of each matrix is $\frac{N(N_{\max}+N_{\min})(N_{\max}-N_{\min}+1)}{2} = O(N(N_{\max}(N_{\max}-N_{\min})))$. As a result, the memory complexity of Algorithm 1 is $O(N_{\max}^2)$ for each user. Note that each user only needs to store its own $q_{n,a}$ and $p_{n,a}$.

Finally, let us explain how the ϵ-greedy algorithm and the UCB algorithm work. These two algorithms can be implemented to the EXP3-MA as an exploration strategy replacing the EXP3. In the ϵ-greedy algorithm, each player chooses an action with probability $\epsilon(t)$, say exploring, and chooses the best action observed so far with probability $1-\epsilon(t)$, say exploiting, at frame t [5]. In our simulation, we use $\epsilon(t) = \frac{1}{t^\alpha}$, where α is an exponent parameter.

The UCB algorithm is on the other hand based on the optimistic principle [6]: It assigns a confidence bonus term $\sqrt{\frac{2\log(1/\delta)}{\kappa_{n,a}(t)}}$ to each action, i.e.,

$$a_n(t) = \arg\max_a \left\{ \mu_{n,a}(t) + \sqrt{\frac{2\log(1/\delta)}{\kappa_{n,a}(t)}} \right\}, \tag{5}$$

where $\kappa_{n,a}(t)$ denotes the number of times that action a has been selected by player n up to frame t and $\delta \in [0,1]$ is the error probability and $\mu_{n,a}(t)$ is player n's empirical average payoff of a up to frame t. If δ is small, then the algorithm exploits the highest $\mu_{n,a}$ more. Otherwise, it explores other actions more. UCB selects an action according to (5).

IV. Simulation Results

Throughout this section, we use $R = 1$ and $C = 2$ and set $C > R$ to incentivize players to avoid collision. Each simulation run length is 10^4 frames, and 1000 simulations are averaged.

The sum throughput (packets/slot) of each frame obtained by EXP3-MA algorithm is illustrated with different values of η, when $N = 3$, $N_{\min} = 3$, $N_{\max} = 6$ in Fig. 2(a) and $N = 8$, $N_{\min} = 6$, $N_{\max} = 8$ in Fig. 2(b). Note that as η gets larger, $p_{n,a}(t)$ concentrates more on an arm with a larger estimated payoff. Thus, in both Fig. 2(a) and (b), the result with $\eta = 0.1$ is better than those with the other smaller η's before frame index does not pass too much, e.g., 100 in Fig. 2(a). Since the algorithm with a smaller η explores more actions in the beginning, it can converge to a better policy later, which results in a higher long-term throughput.

In Fig. 3, EXP3-MA algorithm is compared to ϵ-greedy, and UCB algorithms. As we have seen, η is a tunable parameter of EXP3-MA algorithm, on which the sum throughput may depend. For ϵ-greedy and UCB algorithms, α and δ are the tunable parameter, respectively. By trying η, α, and δ of each algorithm over 10^3 frames in an interval

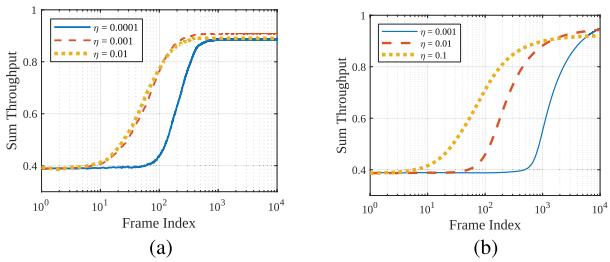

Fig. 2. The average sum throughput (packets/slot) of EXP3-MA algorithm with different values of η. (a) $N = 3$, $N_{\min} = 3$, $N_{\max} = 6$. (b) $N = 8$, $N_{\min} = 6$, $N_{\max} = 8$.

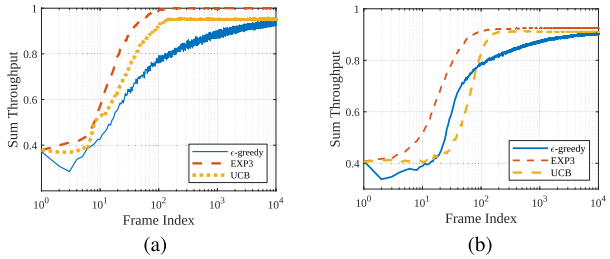

Fig. 3. Sum throughput of EXP3, ϵ-greedy, and UCB. (a) $N = 3$, $N_{\min} = 1$, $N_{\max} = 3$. (b) $N = 5$, $N_{\min} = 4$, $N_{\max} = 6$.

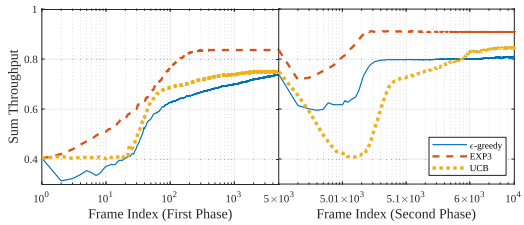

Fig. 4. Average throughputs when a new active player joins the system at the 5000-th frame.

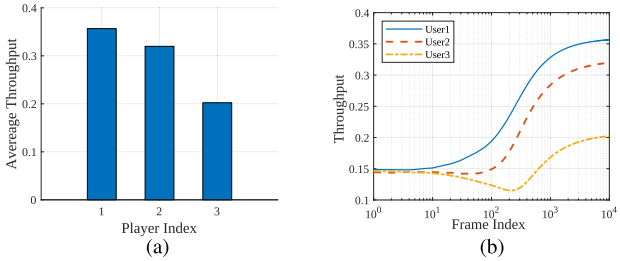

Fig. 5. Average throughputs of players with different priorities (different R/C) by EXP3-MA. (a) Average throughputs of players. (b) Throughputs of players over time.

$[10^{-4}, 5]$ for η, and $[0, 1)$ for other parameters, we find the best value of maximizing the sum throughput and use this value in Figs. 3, 4, and 5.

Fig. 3 compares the sum throughput of ϵ-greedy, EXP3, and UCB algorithms. It can be seen that the EXP3-MA algorithm outperforms other algorithms because it achieves the best trade-off between exploration and exploitation. The ϵ-greedy algorithm on the other hand shows the worst performance because when an action with an optimal payoff is found, the exploration of other non-optimal actions occurs with

TABLE II
THROUGHPUT COMPARISON BETWEEN THE MAB ALGORITHMS OVER 10^3 FRAMES

N	$N_{\min}$	$N_{\max}$	Average throughput EXP3	ϵ-greedy	UCB
4	2	4	**0.915**	0.8581	0.9075
4	2	7	**0.9053**	0.7388	0.7781
5	3	5	**0.9429**	0.8698	0.933
5	3	8	**0.8455**	0.7275	0.73
6	4	6	**0.8977**	0.8462	0.8585
6	4	9	**0.8736**	0.7219	0.8082
7	5	7	**0.9312**	0.8391	0.9147
7	5	10	**0.8685**	0.7313	0.7963
8	6	8	**0.9006**	0.8451	0.8458
8	6	11	**0.8907**	0.7329	0.8011

probability ϵ every frame regardless of their average payoffs in history. This results in slow learning. The UCB algorithm also performed no better than the EXP3-MA algorithm due to how it chooses actions. If the scenario is adversarial and the payoff varies over time as the other players change their actions, the UCB algorithm struggles to estimate the correct upper confidence bound and shows worse performance than its performance in the fixed payoff distribution [5]. As the payoff distribution of each player changes as the other players' actions vary, the system becomes non-stationary for each player. We summarize the average throughputs over 10^3 frames for various values of N, $N_{\min}$, $N_{\max}$ in Table II. It can be observed that the average throughput of the EXP3 algorithm is better than the other MAB algorithms for all tested cases. We check that when $N \leq 8$, $N - N_{\min} \leq 2$ and $N_{\max} - N \leq 3$, EXP3 is better than ϵ-greedy and UCB.

In practice, the population size changes over time. To see how the proposed EXP3-MA algorithm and other MAB algorithms can adapt to it, we vary N over time in Fig. 4, where N varies over the 5000-th frame but $N_{\min}$ and $N_{\max}$ are the same. The left half of the figure (the first 5000 frames) shows the simulation results before a new player joins the system, and the right half of the figure (another 5000 frames) shows the simulation results after a new player joins the system. To adapt to the environment, where a population of players can vary, we use the discounting method of discounted-UCB (D-UCB) in [22]. We calculate the empirical average $\mu_{\gamma,n,a}(t) = \frac{1}{\kappa_{\gamma,n,a}(t)} \sum_{\tau=1}^{t} \gamma^{t-\tau} r_{n,a}(t) \mathbb{1}_{a_n(t)=a}$ for both UCB and the ϵ-greedy algorithm, where $\kappa_{\gamma,n,a}(t) = \sum_{\tau=1}^{t} \gamma^{t-\tau} \mathbb{1}_{a_n(t)=a}$, and $\gamma \in [0, 1)$ is the discount factor. For EXP3 algorithm, we modify $q_{n,a}(t)$ to $q_{\gamma,n,a,}(t) = \sum_{\tau=1}^{t} \gamma^{t-\tau} \hat{r}_{n,a}(t) \mathbb{1}_{a_n(t)=a}$. We use $\gamma = 0.6$.

Initially, the simulation starts with $N = 5$, $N_{\min} = 4$, and $N_{\max} = 7$. At the 5000-th frame, a new player joins the system and N becomes 6. As the new player starts its MAB algorithm and transmits packets, the other players do not receive the same payoff distributions that they have received before the new player joins. Therefore it can be seen in the right figure that the throughputs of all algorithms drop right after a new player joins the system. However, the simulation results show that even if the number of players increases, the MAB algorithms can learn the environment and increase the throughputs gradually. In this case, the performance of EXP3 algorithm outperforms both UCB and ϵ-greedy algorithms. This shows that our proposed learning algorithm can adapt to a time-varying environment better than other MAB algorithms.

We also show that the RA game can apply different priorities to each player in an implicit way by modifying the parameters of each player's payoff function. A player who has higher priority than the other players should have a higher R/C than the other players to send packets aggressively. Fig. 5 presents the simulation result of implementing the priority among the players with $N = 3, N_{\min} = 2, N_{\max} = 5$. We set $R = 4, C = 0$ for player 1, $R = 2, C = 2$ for player 2, and $R = 1, C = 4$ for player 3. It can be seen that the player with higher priority achieved higher throughput than those with lower priority as shown in Fig. 5(a).

V. CONCLUSION

In this paper, we formulated slotted RA system as a RA game. Each frame of M slots was divided into k subframes, each of which has M/k slots. The actions for the players to choose were expressed as (k, l), where l is the slot index of a subframe. Without the players communicating or cooperating with other players, it was shown that this RA game has PNEs and MNE in terms of the size of a subframe and a slot index when the minimum and maximum of the population size were available. In order to realize the PNEs, we proposed the EXP3-based multi-agent learning algorithm for the players to exercise. Our simulation results showed that each time slot can be fully utilized at PNEs, and that EXP3-MA algorithm obtained throughput better than ϵ-greedy and UCB algorithms and can adapt to time-varying environments.

In contrast with the results, this work has some limitations due to analytical complexity and the assumptions we made. Although the parameters of the proposed algorithms are tuned for a fair comparison, it is not clearly shown how to set the parameters such as η and γ for given N, $N_{\min}$, $N_{\max}$. Furthermore, it is hard to give a guideline for selecting an optimal γ for general problems, which is still an open problem [23], [24]. Also, we will mathematically analyze the upper bound of the regret in the EXP3-MA algorithm, and the convergence rate to the PNE that achieves the maximum throughput. Moreover, the algorithm for estimating and calculating $N_{\min}$ and $N_{\max}$ values will be studied and implemented to the RA system model. Lastly, we will extend the RA game model into a more realistic RA scenario with multiple preambles.

REFERENCES

[1] A. Laya, L. Alonso, and J. Alonso-Zarate, "Is the random access channel of LTE and LTE-A suitable for M2M communications? A survey of alternatives," *IEEE Commun. Surv. Tut.*, vol. 16, no. 1, pp. 4–16, Jan.–Mar. 2014.

[2] C. Tang *et al.*, "Comparative investigation on CSMA/CA-based opportunistic random access for Internet of Things," *IEEE Internet Things J.*, vol. 1, no. 2, pp. 171–179, Apr. 2014.

[3] N. Srinidhi, S. D. Kumar, and K. Venugopal, "Network optimizations in the Internet of Things: A review," *Eng. Sci. Technol., Int. J.*, vol. 22, no. 1, pp. 1–21, 2019.

[4] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. Schapire, "The non-stochastic multiarmed bandit problem," *SIAM J. Comput.*, vol. 32, no. 1, pp. 48–77, 2003.

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

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

[7] W. T. Toor, J.-B. Seo, and H. Jin, "Practical splitting algorithm for multichannel slotted random access systems," *IEEE Trans. Mobile Comput.*, vol. 19, no. 12, pp. 2863–2873, Dec. 2020.

[8] J. Seo and H. Jin, "Revisiting two-user S-ALOHA games," *IEEE Commun. Lett.*, vol. 22, no. 6, pp. 1172–1175, Jun. 2018.

[9] Y. Jin and G. Kesidis, "Equilibria of a noncooperative game for heterogeneous users of an aloha network," *IEEE Commun. Lett.*, vol. 6, no. 7, pp. 282–284, Jul. 2002.

[10] J. Barcelo, H. Inaltekin, and B. Bellalta, "Obey or play: Asymptotic equivalence of slotted aloha with a game theoretic contention model," *IEEE Commun. Lett.*, vol. 15, no. 6, pp. 623–625, Jun. 2011.

[11] G. Lugosi and A. Mehrabian, "Multiplayer bandits without observing collision information," *Math. Operations Res.*, vol. 47, no. 2, pp. 1247–1265, 2022.

[12] R. Bonnefoi, L. Besson, C. Moy, E. Kaufmann, and J. Palicot, "Multi-armed bandit learning in IoT networks: Learning helps even in non-stationary settings," in *Proc. Int. Conf. Cogn. Radio Oriented Wireless Netw.*, 2017, pp. 173–185.

[13] E. Longo, A. E. Redondi, and M. Cesana, "Accurate occupancy estimation with WiFi and bluetooth/BLE packet capture," *Comput. Netw.*, vol. 163, 2019, Art. no. 106876.

[14] G. Bianchi and I. Tinnirello, "Kalman filter estimation of the number of competing terminals in an IEEE 802.11 network," in *Proc. 22nd Annu. Joint Conf. IEEE Comput. Commun. Societies*, 2003, pp. 844–852.

[15] R. Yin, Z. Zou, C. Wu, J. Yuan, X. Chen, and G. Yu, "Learning-based WiFi traffic load estimation in NR-U systems," *IEICE Trans. Fundamentals Electron., Commun. Comput. Sci.*, vol. 104, no. 2, pp. 542–549, 2021.

[16] H. D. Althumali, M. Othman, N. K. Noordin, and Z. M. Hanapi, "Dynamic backoff collision resolution for massive M2M random access in cellular IoT networks," *IEEE Access*, vol. 8, pp. 201345–201359, 2020.

[17] L. Wang, W. Wang, X. Hu, and T. Xie, "Optimization of large-scaled random access congestion control oriented to narrow band Internet of Things," *J. Phys.*, vol. 1570, no. 1, 2020, Art. no. 012089.

[18] M. Vilgelm, S. R. Liñares, and W. Kellerer, "Dynamic binary countdown for massive IoT random access in dense 5G networks," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6896–6908, Aug. 2019.

[19] T. N. Weerasinghe, I. A. Balapuwaduge, and F. Y. Li, "Priority-based initial access for URLLC traffic in massive IoT networks: Schemes and performance analysis," *Comput. Netw.*, vol. 178, 2020, Art. no. 107360.

[20] J.-Y. Audibert, S. Bubeck, and G. Lugosi, "Regret in online combinatorial optimization," *Math. Operations Res.*, vol. 39, no. 1, pp. 31–45, 2014.

[21] T. Lattimore and C. Szepesvári, *Bandit Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2020.

[22] A. Garivier and E. Moulines, "On upper-confidence bound policies for switching bandit problems," in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2011, pp. 174–188.

[23] N. Denis, "Issues concerning realizability of blackwell optimal policies in reinforcement learning," 2019, *arXiv:1905.08293*.

[24] V. Dewanto and M. Gallagher, "Examining average and discounted reward optimality criteria in reinforcement learning," 2021, *arXiv:2107.01348*.