IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 1, 2018, accepted April 1, 2018, date of publication April 6, 2018, date of current version April 25, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2823725

Topology-Transparent Scheduling Based on Reinforcement Learning in Self-Organized Wireless Networks

MU QIAO[1], HAITAO ZHAO[1], LI ZHOU[1], CHUNSHENG ZHU[2], AND SHENGCHUN HUANG[1]

[1]College of Electronic Science, National University of Defense Technology, Changsha 410073, China
[2]Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC V6T1Z4, Canada

Corresponding author: Haitao Zhao (haitaozhao@nudt.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61471376, Grant 61601482, and Grant 61501482.

ABSTRACT Topology-transparent scheduling policies do not require the maintenance of accurate network topology information and therefore are suitable for highly dynamic scenarios in self-organized wireless networks. However, in topology-transparent scheduling, it is a very challenging problem to make individual nodes efficiently select their transmission slots in a distributed manner. It is desirable for individual nodes, through time slot selection, to avoid collision on the one hand and utilize as many time slots as possible (i.e., minimize the number of redundant slots) on the other. In this paper, learning-based approaches are employed to solve the time slot scheduling problem. Specifically, the proposed method uses a temporal difference learning approach to address the collision issue and use a stochastic gradient descent approach to reduce the number of redundant slots. Unlike previous works, this learning approach is trained through self-play reinforcement learning without incurring communication overhead for the exchange of reservation information, thereby improving the network throughput. Extensive simulation results validate that our proposal can achieve better efficiency than the existing approaches.

INDEX TERMS Topology-transparent scheduling, reinforcement learning, collision avoidance, redundant slot utilization.

I. INTRODUCTION

The time-division multiple access (TDMA) scheme has been widely used in Internet of Things (IoT) applications to support large and dense networks [1]–[6]. It is a better solution for achieving fair and collision-free scheduling than competition-based schemes (e.g. CSMA/CA) because it allows interfering nodes to transmit in different time slots. To allocate different time slots to interfering nodes, TDMA scheduling schemes usually require the topological information of the network [7], [8]. However, in a dynamic, large and dense network, acquiring this information may incur massive computational complexity and communication overhead [9]–[11]. Consequently, topology-transparent scheduling schemes have recently attracted considerable research interest [12], [13].

In topology-transparent scheduling, each node selects a time slot in a distributed manner without requiring topological information [17]. The main objective is that interfering nodes (i.e., destination node and its neighbors select the transmission slot which has been selected by source node) should select different slots in which to transmit. This is a non-trivial problem due to the lack of topological information. Therefore, this problem has attracted extensive investigation over the past decade [14]–[20], and the related proposals can be broadly divided into two categories: reservation-based schemes [16], [18], [20] and polynomial-based schemes [14], [15], [19]. The basic idea of reservation-based schemes is to reserve different slots for interfering nodes and thus avoid collision. However, these schemes require reservation information to be exchanged among nodes, and the exchange of a large amount of reservation information may result in excessive communication overhead. The polynomial-based approach can achieve considerable improvements in efficiency and robustness in mobile environments [19] by taking advantage of the multi-packet reception capability. This capability allows the collision problem to be resolved even

when two nodes are transmitting in the same time slot. However, this approach introduces complicated polynomial-based algorithms and requires each node to have the capability of multi-packet reception. Another important problem faced by schemes in both categories is that to avoid potential collisions, the time slots are usually only sparsely occupied by nodes; i.e., redundant slots exist that are not efficiently utilized.

In this paper, we propose an intelligent topology-transparent scheduling policy that can learn and optimize rules for avoiding collisions and utilizing redundant slots. This work requires neither central control nor information exchange among nodes. Most importantly, we make use of artificial intelligence techniques, e.g. temporal difference learning and the experience replay approach, to address the slot selection issue for collision avoidance and redundant slot utilization.

The remainder of this paper is organized as follows. Section II describes the system model. In Section III, we explain the details of the proposed topology-transparent scheduling policy. Section IV reports the validation of the proposal via simulation. Section V concludes the paper.

II. SYSTEM MODEL AND PROBLEM DEFINITION

There are N nodes in the considered self-organized wireless network. The network can be represented by a directed graph $G(V, E)$, where V and E are the sets of nodes and edges, respectively [14]. Let S_u denote the neighbors of node u ($\forall u \in V$). $D_u = |S_u|$ is the degree of node u. The maximum node degree is defined as $D_{\max} = \max_{u \in V} D_u$. The nodes do not need to support multi-packet reception, and we assume that unsuccessful transmission is caused only by collisions. Suppose that node u is within the interference range of node v; then, for a transmission from node u to node v in slot i, a collision occurs when node v or another neighbor of node v simultaneously intends to transmit in slot i. Therefore, the interfering node set for transmission from u to v is represented by $S_v \cup \{v\} - \{u\}$.

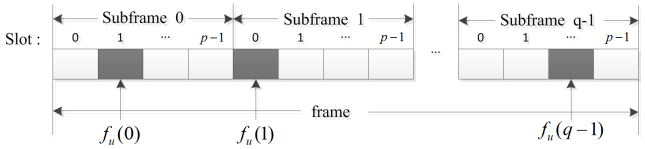


FIGURE 1. The topology-transparent frame structure. [12], [13].

The conventional topology-transparent scheduling policy is based on Galois field theory [21]. Consider a single-channel TDMA network, for which the frame has the structure shown in Fig. 1 [12], [13]. Specifically, each frame is divided into a number of subframes of equal duration; accordingly, each node has multiple transmission opportunities within a frame. Moreover, each subframe further consists of multiple time slots, and a node can select one of these time slots in which to transmit.

The method of selecting transmission slots is a key issue for topology-transparent scheduling. In [12] and [13], the following slot selection function was proposed:

$$f(x) = \sum_{i=0}^{k} a_i x^i (\mathrm{mod} p), \tag{1}$$

where k is the maximum number of collisions between two arbitrary nodes in one frame; $a_i \in 0, 1, \ldots, p-1$ is a coefficient parameter that randomizes the slot selection (different nodes will select different a_i values); x is the index of the current subframe; p is the number of slots in one subframe; and q is the number of subframes in one frame. In accordance with (1), all selected slots for an arbitrary node constitute the following set:

$$GF(x) = \{xq + f(x), x = 0, 1, \ldots, q-1\}. \tag{2}$$

This function indicates the position in the frame of the transmission slot selected for each node. The randomness of $f(x)$ will prevent different nodes from choosing the same slot.

This function indicates the position in the frame of the transmission slot selected for each node. The maximum degree of the function is equal to k. The set $GF(x)$ should include all possible functions. The number of functions covered by $GF(x)$ is p^{k+1}. To guarantee that each node has at least the minimum guaranteed throughput, each node should be given a unique slot selection function [13], and thus, the following constraint should be satisfied:

$$p^{k+1} \geq N. \tag{3}$$

For each node, slot $(f(x) \bmod p)$ in subframe x will be selected in accordance with the slot selection function $f(x)$. To guarantee that each node has at least one slot in which it can transmit data to any neighbor during each frame, the following constraint should be satisfied:

$$q \geq kD_{\max} + 1. \tag{4}$$

The basic idea of topology-transparent scheduling is to ensure that different nodes will select different time slots via the randomness of (1). However, two major problems arise. First, collisions still occur. More explicitly, for node u and node v, if $f_u(i) - f_v(i) = 0$ (i.e., node u transmits data in the $f_u(i)$-th slot in the i-th subframe), nodes u and v will encounter transmission collision. Second, improper setting of the parameter k may lead to a highly sparse or crowded scheduling policy, which means that in some cases, there may be many redundant slots in the time slot selection process, whereas in other cases, the selected time slots for different nodes will be concentrated within a short period.

Motivated by these challenges, we improve the slot structure to enable a learning process to address the collision issue and reduce the number of redundant slots. As shown in Fig. 2, in the improved slot structure, each slot is further divided into three intervals: Beacon, DATA and ACK.

The Beacon interval enables each node to collect the slot selection information from its one-hop neighbors during the

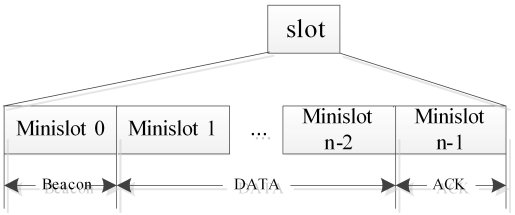


FIGURE 2. The improved slot structure.

redundant slot utilization stage. The information collected from each node includes its own slot index, and this information is obtained by receiving broadcasts from neighbors. Thus, each node can determine the number of redundant slots through a simple subtraction calculation. If the value of $|f(x)-f_{neighbor}(x)|$ is equal to '1', this means that there are no redundant slots. Otherwise, redundant slots exist in the time slot selection process. The DATA interval is used to transmit data packets, and the ACK interval is used to acknowledge the DATA transmission. Moreover, the nodes can observe the collision state based on the ACK message. If there is no collision in current slot, node will receive the ACK message. If a node does not receive ACK before time expires, it means collision state.

III. IMPROVED PARAMETER SELECTION

In this section, we propose a flexible and robust time slot selection function that can increase the randomness of slot selection and thus alleviate the collision problem.

In previous studies, the frame length has been minimized by adjusting the parameters p and q while setting k to a fixed value. However, the maximization of the minimum guaranteed node throughput, as introduced in [15] and [20], has not been considered. As mentioned above, if $q \leq p$, then any two nodes may make the same time slot selection at most k times. To maximize the minimum guaranteed node throughput and guarantee that each node has at least one unique slot selection function during each frame, the parameters p and q should be selected in accordance with constraints (3) and (4).

A. ALGORITHM DESCRIPTION

Consider the transmission $u \rightarrow v, \forall u \in V, \forall v \in S_u$, with a one-frame duration. We use the same method discussed in [13] and [20], in which the optimal throughput does not depend on the number of subframes q. Thus, we assume $p = q$. If any arbitrary pair of nodes have different slot selection functions, then each node can occupy p slots during one frame. Therefore, the minimum guaranteed node throughput can be expressed as

$$T_{\min} = \frac{p - kD_{\max}}{p^2}, \tag{5}$$

where $kD_{\max}$ is the maximum number of collisions between any arbitrary pair of nodes during one frame, $p - kD_{\max}$ is the minimum number of guaranteed successful transmissions for each node during one frame, and p^2 is the overall frame length. According to the discussed in [13], the equation (3) and (4) should be satisfied. To maximize $T_{\min}$ as the parameter selection criterion of p and k. The maximal minimum guaranteed node throughput (i.e. upper bound) can be expressed as

$$\begin{cases} \max\left(T_{\min}\left(k\right)\right) = \dfrac{2kD_{\max} - kD_{\max}}{\left(2kD_{\max}\right)^2}, & N^{1/(k+1)} \leq 2kD_{\max}, \\ \max\left(T_{\min}(k)\right) = \dfrac{N^{1/(k+1)} - kD_{\max}}{\left(N^{2/(k+1)}\right)}, & N^{1/(k+1)} > 2kD_{\max}. \end{cases} \tag{6}$$

Based on the optimal k and p, each node can obtain a unique time slot selection function for randomly choosing transmissions. There are p^{k+1} available functions covered by $GF(x)$.

Theorem 1: The optimal parameters k and p should satisfy the following:

$$\begin{cases} p = 2\left\lceil k_0 \right\rceil D_{\max}, & k = \left\lceil k_0 \right\rceil; \\ p = N^{1/(\lfloor k_0 \rfloor + 1)}, & k = \lfloor k_0 \rfloor; \\ p = N^{1/(k+1)}, & k < \lfloor k_0 \rfloor. \end{cases} \tag{7}$$

Proof: To find the maximum value of $T_{\min}$, we should find a p that satisfies $\frac{\partial T_{\min}}{\partial p} = 0$. This calculation yields

$$p^{-3}(2kD_{\max} - p) = 0. \tag{8}$$

Since $p \geq kD_{\max} \geq 0$, we find from (8) that $p = 2kD_{\max}$. $T_{\min}$ increases with increasing p when $kD_{\max} \leq p \leq 2kD_{\max}$ and decreases with increasing p when $p \geq 2kD_{\max}$. From (3) and (4), it should hold that $p \geq \max\left\{2kD_{\max}, N^{1/(k+1)}\right\}$.

We assume that $x = k_0$ is the unique positive root of

$$2xD_{\max} = N^{1/(x+1)}. \tag{9}$$

From (9), we can observe that $2kD_{\max}$ increases with increasing k, while $N^{1/(x+1)}$ decreases with increasing k. Since $k \geq \lceil k_0 \rceil$, $N^{1/(k+1)} \leq 2kD_{\max}$ should be satisfied. According to (6), $\max\left(T_{\min}\left(k\right)\right)$ decreases with increasing k. Therefore, when $k \geq \lceil k_0 \rceil$, this implies that $\max\left(T_{\min}\left(k\right)\right) \leq \max\left(T_{\min}\left(\lceil k_0 \rceil\right)\right)$.

In addition, since $k \leq \lfloor k_0 \rfloor$, $N^{1/(k+1)} \geq 2kD_{\max}$ should be satisfied. If $\lfloor k_0 \rfloor = 0$ and k is a non-negative integer, then $\max\left(T_{\min}\left(k\right)\right) = \max\left(T_{\min}\left(0\right)\right)$. Now, let us consider the scope of $\lfloor k_0 \rfloor \geq 1$; if $1 \leq k \leq k_0$, it can be derived from (6) as shown in (10), as shown at the bottom of this previous page.

$$\max\left(T_{\min}\left(k\right)\right) - \max\left(T_{\min}\left(k-1\right)\right) = \frac{\left(N^{\frac{3k+1}{k(k+1)}} \cdot \left(N^{\frac{1}{k(k+1)}} - 1 - kD_{\max}N^{\frac{1-k}{k(k+1)}} + (k-1)D_{\max}N^{\frac{-1}{k}}\right)\right)}{\left(N^{\frac{2}{k}}\right)\left(N^{\frac{2}{k+1}}\right)} \tag{10}$$

By taking the derivative with respect to k in (10), we obtain

$$\begin{aligned}&\frac{\partial\left(N^{\frac{1}{k(k+1)}}-1-kD_{\max}N^{\frac{1-k}{k(k+1)}}+(k-1)D_{\max}N^{\frac{-1}{k}}\right)}{\partial k}\\&=N^{\frac{1}{k(k+1)}}\cdot\ln N\cdot\left(\frac{-2k-1}{k^2(k+1)^2}\right)\\&\quad-kD_{\max}\cdot\ln N\cdot N^{\frac{1-k}{k(k+1)}}\cdot\left(\frac{k^2-2k-1}{k^2(k+1)^2}\right)\\&\quad-D_{\max}N^{\frac{1-k}{k(k+1)}}+D_{\max}N^{\frac{-1}{k}}\\&\quad+(k-1)D_{\max}\cdot\ln N\cdot N^{\frac{-1}{k}}\cdot\left(\frac{1}{k^2}\right).\end{aligned}\tag{11}$$

For all $k\le\lfloor k_0\rfloor$, it is found that $N^{1/(k+1)}\ge 2kD_{\max}$, and we have

$$\begin{aligned}&N^{\frac{1}{k(k+1)}}\cdot\ln N\cdot\left(\frac{2k+1}{k^2(k+1)^2}\right)\\&\quad+kD_{\max}\cdot\ln N\cdot N^{\frac{1-k}{k(k+1)}}\cdot\left(\frac{k^2-2k-1}{k^2(k+1)^2}\right)\\&>\ln N\cdot N^{\frac{-1}{k}}\cdot kD_{\max}\cdot\left(\frac{1}{k^2}\right).\end{aligned}\tag{12}$$

Because $N^{1-k/k(k+1)}>N^{-1/k}$ is satisfied, by jointly considering (10) and (11), we find that

$$\frac{\partial\left(N^{1/k(k+1)}-1-kD_{\max}N^{1-k/k(k+1)}+(k-1)D_{\max}N^{-1/k}\right)}{\partial k}<0.$$

As discussed above, when $1\le k\le k_0$, $\max(T_{\min}(k))-\max(T_{\min}(k-1))$ is decreasing. Therefore, if $N^{1/(k_0+1)}=2k_0D_{\max}$, $\max(T_{\min}(k))-\max(T_{\min}(k-1))\ge 0$ is satisfied. Thus, for all $1\le k\le\lfloor k_0\rfloor$, it holds that $\max(T_{\min}(k))\le\max(T_{\min}(\lfloor k_0\rfloor))$.

Since $N^{1/(k_0+1)}=2k_0D_{\max}$ and $\max(T_{\min}(k))-\max(T_{\min}(k-1))\ge 0$, considering (10), we have

$$\begin{aligned}&\max(T_{\min}(k_0))-\max(T_{\min}(k_0-1))\\&=\left(\frac{\left(N^{\frac{1}{k_0(k_0+1)}}-1-\sqrt{1/k_0}\right)\left(N^{\frac{1}{k_0(k_0+1)}}-1+\sqrt{1/k_0}\right)}{2N^{\frac{1}{k_0(k_0+1)}}}\right).\end{aligned}\tag{13}$$

Then, from (10) and (13), $N^{1/k_0(k_0+1)}-1-\sqrt{1/k_0}\ge 0$ implies that $\max(T_{\min}(k_0))-\max(T_{\min}(k_0-1))\ge 0$, and for all $k\le\lfloor k_0\rfloor$, the inequality expression $\max(T_{\min}(k))\le\max(T_{\min}(\lfloor k_0\rfloor))$ is satisfied.

To maximize the minimum guaranteed node throughput, we must select the optimal values of k and p under the given N and $D_{\max}$. Based on the above analysis, the optimal k and p can be determined as shown in Algorithm 1:

Thus, we find that the optimal parameters k and p are determined by N and $D_{\max}$ instead of by the specific network topology.

Algorithm 1 Algorithm for Determining the Optimal Parameters k and p

1: Obtain k_0 by using (9).
2: **if** $N^{1/k_0(k_0+1)}-1-\sqrt{1/k_0}\ge 0$ **then**
3: Determine k in the scope of $\lfloor k_0\rfloor\le k\le\lceil k_0\rceil$ from (6).
4: **else**
5: Find k in the scope of $k\le\lfloor k_0\rfloor$ and $k\le\lceil k_0\rceil$.
6: **end if**
7: Calculate the optimal number of slots p by using the optimal k.

B. THROUGHPUT ANALYSIS

In this subsection, we analyze the normalized average node throughput that is guaranteed by the proposed time slot selection algorithm.

The average probability that node u successfully chooses at least one open slot during one subframe can be formulated as

$$p_{u,average}=\frac{1}{D_{\max}}.\tag{14}$$

The probability that node u collides with at least one interfering node v or other neighbor of node v of $S_v\cup\{v\}-\{u\}$ in slot set $GF_u(x)$ can be calculated as

$$\begin{aligned}p_{u,collision}(l)=&\prod_{k=1}^{l}\frac{p^k-k}{p^{k+1}-k}\\&s.t.\ l\le\min(D_{\max},p^k-1),\end{aligned}\tag{15}$$

where l is the number of interfering nodes that collide with node u in the same slot.

The probability that node u does not collide with the other $D_{\max}-l$ interfering nodes in slot set $GF_u(x)$ can be calculated as

$$\begin{aligned}p_{u,collisionfree}(D_{\max}-l)=&\prod_{k=1}^{D_{\max}-l}\frac{p^{k+1}-p^k-k+1}{p^{k+1}-k-l}\\&s.t.\ l\le\min(D_{\max},p^k-1).\end{aligned}\tag{16}$$

By jointly considering (14) and (15), the probability that node u collides with at least one interfering node in slot set $GF_u(x)$ is found to be

$$\begin{aligned}P_u=&\sum_{l=1}^{D}\binom{D_{\max}}{l}\cdot\left(\prod_{k=1}^{l}\frac{p^k-k}{p^{k+1}-k}\right)\\&\cdot\left(\prod_{k=1}^{D_{\max}-l}\frac{p^{k+1}-p^k-k+1}{p^{k+1}-k-l}\right)\\&s.t.\ l\le\min(D_{\max},p^k-1),\\&D=\min(D_{\max},p^k-1).\end{aligned}\tag{17}$$

Based on (16), the normalized average node throughput is

$$\begin{aligned}T=&\frac{1}{N}\sum_{i=1}^{N}\frac{1}{p^2}\cdot\left(\frac{p(n-2)}{n}\cdot\left(1-P_u^i\right)\right.\\&\left.+\left(p\cdot P_u^i+I^i\right)\cdot p_{u,average}\cdot\left(\frac{n-2}{n}\right)\right),\end{aligned}\tag{18}$$

where I is the number of redundant slots and n is the number of minislots.

Considering the issues of collision and redundant slots in the network, we improve upon the conventional topology-transparent scheduling policy by introducing the proposed parameter selection scheme and collision probability calculation. The benefit is that the enhanced policy can be used to adapt to the current requirements of the communication environment. However, the inherent problem of the conventional scheduling policy is not completely resolved; highly sparse and conflicting situations still exist. In the following section, a reinforcement learning model is introduced to address this issue.

IV. REINFORCEMENT LEARNING FOR TOPOLOGY-TRANSPARENT SCHEDULING

A. FRAMEWORK OF THE REINFORCEMENT LEARNING MODEL

Reinforcement learning (RL) is usually adopted to solve problems in which a learning agent interacts with its environment to achieve goals related to the state of the environment [22]–[26]. Here, we propose an RL model to allow nodes to optimally select slots while utilizing more redundant slots. The RL model is composed of two stages. First, we use the temporal difference (TD) learning approach to select the collision-free slots in one subframe. In this stage, we build a tree-based structure to search for the maximum expected return. Second, we use the experience replay approach to further optimize the slot utilization in each subframe. In this stage, we consider the expected return obtained in the first stage as a fixed target. Then, we find a slot selection vector that minimizes the sum-squared error between the approximate and real values. Several definitions related to the model are given as follows.

1) STATE

We define S_t as the state observed by node i at time t. We use $S = \{S_1, \ldots, S_t\}$ to denote the countable non-empty set of the state space, $S_t \in S$. The state transition from S_t to S_{t+1} depends on the corresponding action, and accordingly, the next state S_{t+1} can be observed when the next action occurs.

2) REWARD

We use $r = \{r_1, \ldots, r_t\}$ to denote the countable non-empty set of the reward space. r_t is the reward associated with the $(t-1)$-*th* state transition. The feedback information is stored in minislot 0. If node i selects a slot with no collision at time t, then the reward value is '1'. Otherwise, the reward value is '-1'. To preserve the throughput performance, it is desirable for the time slots to be fully utilized. In the second stage, the negative state is inefficiency slot utilization caused by highly sparse scheduling policy rather than transmission failure. It also can be used to keep the slot scheduling policy working. Therefore, if node i senses that there are no redundant slots at time t, then the reward value is increased to '100'. Otherwise, the reward value remains equal to '1'.

3) ACTION

We define $A_{i,t}$ as the action taken by node i at time t. We use $A = \{A_{i,1}, \ldots, A_{i,t}\}$ to denote the countable non-empty set of the action space.

4) ACTION-VALUE FUNCTION

The action-value function $Q(S_t, A_t)$ is associated with the action A_t and the state S_t at time t. It is equivalent to the average node throughput as defined in Section II.

B. TEMPORAL DIFFERENCE LEARNING APPROACH FOR COLLISION AVOIDANCE

In the reinforcement learning framework introduced above, each node interacts with the subframe environment. In each discrete time step t, a node observes the current state S_t, takes an action A_t, and obtains a feedback reward r_t. During the slot selection phase, each node uses a unique time slot selection function $f(x)$ to obtain a slot in each subframe. If $f_u(x) - f_v(x) = 0$, this means that the slot selection functions of node u and its interfering neighboring node v have the same root. In this case, it is desirable for these interfering neighboring nodes to select different, collision-free slots in the same subframe. We use a temporal difference learning algorithm to evaluate a value function via minimax search [27]–[31]. Fig. 3 shows the search structure. Each child node has a certain expected return value. The state transition from S_t to S_{t+1} has only two possible states: collision or collision-free.

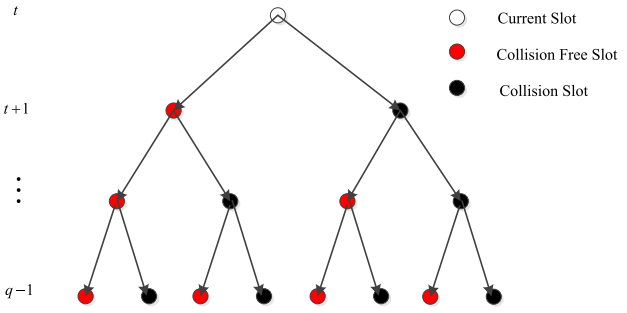


FIGURE 3. Expected slot selection state search structure.

If we assume that node i takes action A_t to select its slot in accordance with the slot selection function $f_i(S_t)$ for the current subframe state S_t, then node i receives a scalar reward $r_t = \sum_{k=0}^{p-1} \gamma^k r_{t+k}$, which is the total accumulated return from time step t with a discount factor of γ ($\gamma \in (0, 1]$). The goal of the node is to maximize the expected return from each state S_t. The optimal value function $Q^*(S, A) = \max_{\pi=greedy} Q(S, A)$ yields the maximum action-value function for state S and action A. The expected reward from each state is given by

$$J^*(S) = E\left[r_t | S_t = S, A\right], t \in [0, p-1]. \tag{19}$$

A naive means of achieving optimal slot selection behavior in the tree structure is to list all behaviors and then identify the one that results in the highest possible value for each initial state. However, because the state and action spaces could be very large and very high dimensional in the forward-looking search process, such a policy is not viable. To circumvent this problem, we compute an approximation function to estimate the value function. More explicitly, a slot selection vector w that minimizes the mean square error between the approximate value $\widehat{J}(S)$ and the real value $J^*(S)$ should be found.

Suppose that $S_1, \ldots, S_t, \ldots, S_{p-1}$ is a sequence of slot selection states. For a given slot selection vector w, we define the temporal difference error of the subframe state transition from S_t to S_{t+1} as

$$d_t = \widehat{J}(S_{t+1}, w) - \widehat{J}(S_t, w). \tag{20}$$

Note that d_t quantizes the difference between the reward predicted by $\widehat{J}(\cdot, w)$ in state S_{t+1} and the reward predicted by $\widehat{J}(\cdot, w)$ in state S_t. The real value function J^* has the property

$$E_{S_{t+1}|S_t}\left[J^*(S_{t+1}) - J^*(S_t)\right] = 0. \tag{21}$$

This means that if $\widehat{J}(\cdot, w)$ is a proper approximation of J^*, then (21) should be satisfied. In other words, node i select a conflict-free slot in states S_t and S_{t+1}.

For the approximation of a linear function, the establishment of a tabular approach can be a viable choice. We can use the feature value to indicate the current slot state. If node i selects a conflict-free slot in state S_t, then the feature value will be equal to '1'. Otherwise, the feature value will be equal to '-1'. The ideal tabular features can be expressed as

$$x(S) = \begin{pmatrix} 1(S = s_0) \\ \vdots \\ 1(S = s_{p-1}) \end{pmatrix}. \tag{22}$$

The effect of the slot selection vector w on each individual state can be expressed as

$$\widehat{J}(S_t, w) = \begin{pmatrix} 1(S = s_0) \\ \vdots \\ 1(S = s_{p-1}) \end{pmatrix} \begin{pmatrix} w_0 \\ \vdots \\ w_{p-1} \end{pmatrix}. \tag{23}$$

In the above discussion, we store the expected reward values in a feature value table. Similarly, we can apply supervised learning to build a set of real values as the 'training data' obtained from the time slot selection function. This training set can be written as

$$J^*(S) = \{< S_1, r_2 + \gamma\widehat{J}(S_2, w) >, < S_2, r_3 + \gamma\widehat{J}(S_3, w) >, \ldots, < S_{p-2}, r_{p-1} + \gamma\widehat{J}(S_{p-1}, w) >\}. \tag{24}$$

The slot selection vector w can be updated by using the forward-looking linear temporal difference learning algorithm:

$$\begin{aligned} \Delta w &= \alpha(r_{t+1} + \gamma\widehat{J}(S_{t+1}, w) - \widehat{J}(S_t, w))\nabla_w\widehat{J}(S_t, w) \\ &= \alpha(r_{t+1} + \gamma\widehat{J}(S_{t+1}, w) - \widehat{J}(S_t, w))x(S), \end{aligned} \tag{25}$$

Algorithm 2 Algorithm for Collision Avoidance Based on Temporal Difference (TD) Learning

```
// Initialization
Each node initializes its slot selection vector w.
Each node executes an initial action A_0 in accordance with the time slot selection function f(x)_{S_0}. Each node observes the initial state S_0 and reward r_0.
// Learning the collision avoidance rule
for i = 1 to N do
  for t = 0 to p − 1 do
    if Δw ≠ 0 then
      Node i determines its current action in accordance with the slot selection vector w.
    else
      Node i executes its current action to select a slot.
    end if
  end for
end for
Node i selects a slot based on the updated slot selection vector w.
```

where $r_{t+1} + \gamma\widehat{J}(S_{t+1}, w)$ is a biased sample of the real value set J^* and α is a step-size parameter that specifies the update speed of the slot selection vector w.

Equation (25) is used to update the vector w successively over time to obtain better predictions of the expected reward $\widehat{J}(\cdot, w)$. For a linear $\widehat{J}(\cdot, w)$, it has been proven that w can converge to a near-optimal solution if the slot selection actions A_t are independent of the slot selection vector w [27].

Each node selects its slot in accordance with the slot selection vector w, which eventually converges to a collision-avoiding solution. The detailed operations for obtaining w are summarized in Algorithm 2. In the initialization phase, each node initializes its slot selection vector w and then observes the resultant state S_0 and reward r_0, after which the initial action A_0 is executed in accordance with the time slot selection function $f(x)_{S_0}$. During the learning phase, node i successively updates its slot selection vector w. If the current slot selection action allows the expected reward value to be reached, it will be executed by node i. Otherwise, node i should determine its current slot selection action in accordance with the updated slot selection vector w.

C. STOCHASTIC GRADIENT DESCENT WITH EXPERIENCE REPLAY FOR UTILIZING REDUNDANT SLOTS

The conventional topology-transparent scheduling policy described in Section II helps to reduce transmission conflicts. However, it is a highly sparse scheduling policy with many redundant slots. To enable full utilization of these redundant slots, we use a stochastic gradient descent approach with experience replay to improve the efficiency and stability of the topology-transparent scheduling policy. Experience replay provides a mechanism for the distribution of reward prediction samples toward rewarding actions [32]–[36]. A higher reward in each subframe is

required to avoid sparse scheduling. Unlike in the above discussion, the state S_t here represents the number of redundant slots in subframe t, $t \in [0, p-1]$.

Provided that the approximate collision avoidance value $\widehat{J}(S)$ well approximates the real value $J^*(S)$, we obtain a fixed state value function pair $< S_t, J^*(S_t) >$ without utilizing redundant slots. The main idea of this approach is to store state transitions in a replay memory $M = \{< S_1, J^*(S_1) >, < S_2, J^*(S_2) >, \ldots, < S_{q-1}, J^*(S_{q-1}) >\}$ and then apply stochastic gradient descent updates to sample random transitions from this memory. In such a sampling of transitions from a fixed replay memory, rewarding target states will be oversampled and thus will be learned from more frequently. If we sample sequences directly from the dynamic state value function pairs, then the update results will become unstable. Let ω denote the redundant slot elimination vector. By resampling the previous fixed experience values and randomly utilizing the redundant slots over t subframes, an iterative state value function replay computation is performed that exploits newly discovered states based on reward prediction. This approach can be viewed as a simplified subframe-by-subframe prioritized replay process.

Let $J(S, A; \omega)$ be a state value function with the redundant slot elimination vector ω. The mean square error (MSE) between the real value $\widehat{J}_{rs}(S)$ and the fixed value $J^*_{rs}(S)$ can be optimized by iteratively minimizing a sequence of *i-th* loss functions, which can be calculated as

$$LS_i(\omega_i) = E_{S,A,r,S' \sim M}\Big[(r + \gamma \max_{A'} J^*_{rs}(S', A'; \omega_i^-) - \widehat{J}_{rs}(S, A; \omega_i))^2\Big]. \quad (26)$$

where S' is the target state which is a value of expectation. A' is the target slot selection action which is a value of expectation. ω_i^- is the redundant slot elimination vector which corresponding to the target state.

Applying stochastic gradient descent to update the redundant slot elimination vector ω yields

$$\Delta\omega = \alpha(J^*_{rs}(S_t) - \widehat{J}_{rs}(S_t, \omega))\nabla_\omega \widehat{J}_{rs}(S_t, \omega), \quad (27)$$

where $(J^*_{rs}(S_t) - \widehat{J}_{rs}(S_t, \omega))$ is the prediction error between the fixed and real values and $\nabla_\omega \widehat{J}_{rs}(S_t, \omega)$ is the gradient of the real state value function.

The target redundant slot elimination vector ω satisfies

$$LS(\omega) = \sum_{t=1}^{p-1} (J^*_{rs}(S_t) - \widehat{J}_{rs}(S_t, \omega))^2 = E_M\Big[(J^*_{rs}(S_t) - \widehat{J}_{rs}(S_t, \omega))^2\Big]. \quad (28)$$

The above analysis suggests that each node should find a redundant slot elimination vector ω that minimizes the sum-squared error between the real value $\widehat{J}_{rs}(S)$ and the fixed value $J^*_{rs}(S)$. In the initialization phase, each node first initializes its replay memory M and its redundant slot elimination vector ω. Next, each node observes the resultant state S_0 and reward r_0(the state and reward was observed when nodes take initial slot selection action A_0), based on which the time slot selection function $f(x)_{S_0}$ is computed to specify the initial action A_0, and the triggered state transition is recorded in the replay memory M. During the learning phase, node i first samples a state value function pair $< S_t, J^*(S_t) >$ from the replay memory M. Next, node i optimizes the MSE between the sampled fixed value and the current real value, after which the redundant slot elimination vector ω is updated by using a variant of stochastic gradient descent. When ω converges to the least-square solution $\omega^\pi = \arg\min_{\omega} LS(\omega)$, node i can select a slot based on the updated result of ω (i.e., π is the slot selection policy corresponding to ω). Throughout the entire process, node i should periodically update the replay memory M. The process described above is summarized in Algorithm 3.

Algorithm 3 Algorithm for Utilizing Redundant Slots Based on Experience Replay

```
// Initialization
Each node initializes its replay memory M and its redundant slot elimination vector ω.
Each node executes an initial action A_0 in accordance with the time slot selection function f(x)_{S_0} and observes the resultant state S_0 and reward r_0.
// Learning the rule for utilizing redundant slots
for i = 1 to N do
  while (ω^π = arg min_ω LS(ω)) do
    Node i stores the state transitions in the replay memory M.
    for t = 0 to p − 1 do
      Node i samples the current state value function pair < S_t, J*(S_t) > from M.
      Node i optimizes the MSE between the current real value and the fixed value.
      Node i updates the redundant slot elimination vector ω via (28).
    end for
      Node i periodically updates the replay memory M.
  end while
end for
Node i selects a slot based on the updated redundant slot elimination vector ω.
```

V. EVALUATION

We first compare the proposed hybrid topology-transparent scheduling policy with the algorithm proposed by Ju and Li [13] and the algorithm proposed by Liu *et al.* [19] in terms of the normalized average node throughput. We also compare the performance of the proposed polynomial-based method with that of the method based on the reinforcement learning model. Without loss of generality, we fix the number of nodes to $N = 200$. The maximum node degree is initialized to 1 and is then incremented by 5 until it reaches 96. A higher

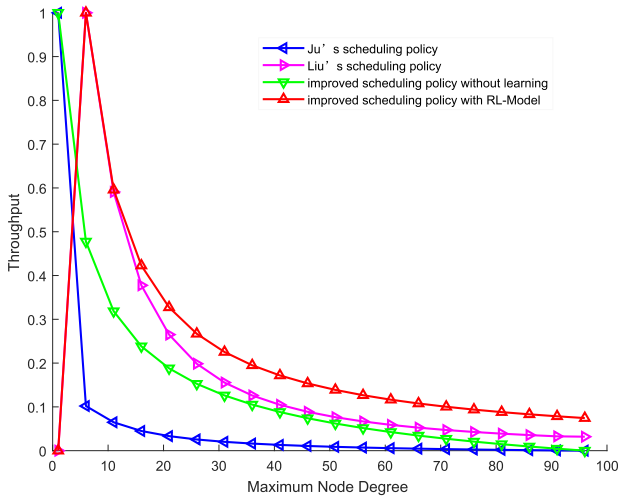


FIGURE 4. The achieved normalized average node throughput versus the maximum node degree.

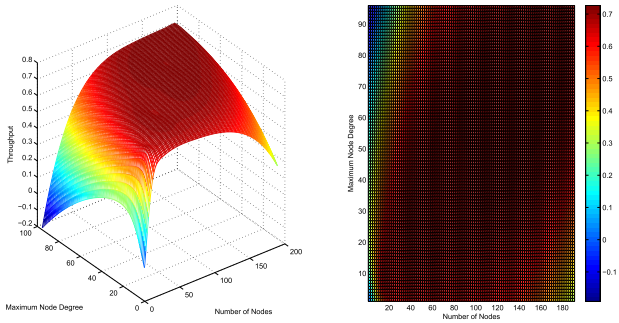


FIGURE 5. Variation of the system throughput with both the number of nodes and the maximum node degree under the RL model.

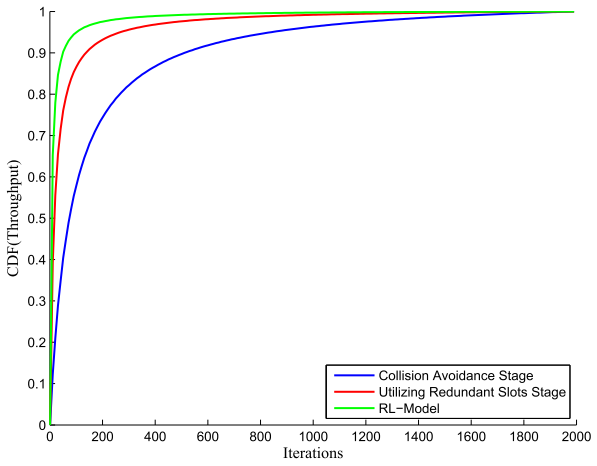


FIGURE 6. Convergence performance in different stages of the reinforcement learning algorithm.

node degree indicates that the network is denser and that more nodes will encounter more severe conflicts. The two learning stages are simulated in MATLAB simulation platforms.

Fig. 4 presents the performance comparison in terms of the achieved normalized average node throughput versus the maximum node degree. The green line represents the improved topology-transparent scheduling policy without the learning model. The red line represents the improved topology-transparent scheduling policy with the reinforcement learning model. It can be observed that 1) the proposed learning-based approach achieves the best throughput performance; 2) the performance of the improved scheduling approach without learning is close to that of Liu's algorithm when the maximum node degree exceeds 40, whereas the performance gap widens when the maximum node degree increases because the performance of the latter algorithm depends on the numbers of neighbors and reservation packets, with more neighbors resulting in more severe interference; and 3) the performance gap for the improved scheduling approach with and without the RL model is independent of the maximum node degree because the learning policy relies on the updated optimal slot selection behavior based on the proposed approach without learning.

Fig. 5 shows the variation of the system throughput with the number of nodes and the maximum node degree under the RL model. The system throughput dramatically changes with both the number of nodes and the maximum node degree. With the RL model, the system throughput is significantly enhanced. High-level performance is maintained in terms of the system throughput. This finding can be explained as follows: 1) the sample space becomes larger as the total number of nodes and the maximum node degree increase, and 2) the nodes have accumulated sufficient historical observations and decision experience with which to train the model.

Fig. 6 presents the cumulative distribution functions (CDFs) of the system throughput during the different learning stages. This figure shows that the learning model exceeds the two learning stages in terms of throughput. In addition, the convergence performance of the redundant slot utilization stage is higher than that of the collision avoidance stage. Specifically, the convergence of the RL model exhibits a gradual performance improvement. This behavior is explained by the following three considerations: 1) during the collision avoidance stage, the temporal difference learning approach helps to reduce transmission collisions, but redundant slots are not eliminated; 2) during the redundant slot utilization stage, the experience replay approach helps to optimize slot utilization based on the previous stage; and 3) the RL model further improves the convergence performance by using real experience to update the scheduling policy.

VI. CONCLUSION

In this paper, we present a hybrid topology-transparent scheduling policy that combines the advantages of polynomial based schemes with those of RL-model-based schemes. The presented policy has two main characteristics. First, by employing the TD learning approach, it allows time slots to be allocated to individual nodes in a distributed manner while avoiding collisions. Second, by employing the RL model approach, it eliminates redundant time slots and thus

increases the overall utilization of time slots. The proposed learning approaches are trained via self-play reinforcement learning without incurring communication overhead for the exchange of reservation information exchange and therefore are suitable for self-organized wireless networks, especially for next-generation large-scale cognitive wireless networks.

REFERENCES

[1] N. C. Luong, D. T. Hoang, P. Wang, D. Niyato, D. I. Kim, and Z. Han, "Data collection and wireless communication in Internet of Things (IoT) using economic analysis and pricing models: A survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 4, pp. 2546–2590, 4th Quart., 2016.

[2] Z. Ning *et al.*, "A cooperative quality-aware service access system for social Internet of vehicles," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2017.2764259.

[3] C. Zhu, J. J. P. C. Rodrigues, V. C. M. Leung, L. Shu, and L. T. Yang, "Trust-based communication for the industrial Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 16–22, Feb. 2018.

[4] C. Zhu, L. Shu, V. C. M. Leung, S. Guo, Y. Zhang, and L. T. Yang, "Secure multimedia big data in trust-assisted sensor-cloud for smart city," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 24–30, Dec. 2017.

[5] C. Zhu, V. C. M. Leung, L. Shu, and E. C.-H. Ngai, "Green Internet of Things for smart world," *IEEE Access*, vol. 3, pp. 2151–2162, 2015.

[6] X. Zhai, X. Guan, C. Zhu, L. Shu, and J. Yuan, "Optimization algorithms for multi-access green communications in Internet of Things," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2018.2792300.

[7] H. A. Omar, W. Zhuang, and L. Li, "VeMAC: A TDMA-based MAC protocol for reliable broadcast in VANETs," *IEEE Trans. Mobile Comput.*, vol. 12, no. 9, pp. 1724–1736, Sep. 2013.

[8] M. Alinaghian, M. Ghazanfari, N. Norouzi, and H. Nouralizadeh, "A novel model for the time dependent competitive vehicle routing problem: Modified random topology particle swarm optimization," *Netw. Spatial Econ.*, vol. 17, no. 4, pp. 1185–1211, 2017.

[9] B. Wang, Q. Yang, L. T. Yang, and C. Zhu, "On minimizing energy consumption cost in green heterogeneous wireless networks," *Comput. Netw.*, vol. 129, pp. 522–535, Dec. 2017.

[10] J. Zhang *et al.*, "Energy-latency trade-off for energy-aware offloading in mobile edge computing networks," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2017.2786343.

[11] Y. Guo, X. Hu, B. Hu, J. Cheng, M. Zhou, and R. Y. K. Kwok, "Mobile cyber physical systems: Current challenges and future networking applications," *IEEE Access*, vol. 6, pp. 12360–12368, 2017, doi: 10.1109/ACCESS.2017.2782881.

[12] I. Chlamtac and A. Farago, "Making transmission schedules immune to topology changes in multi-hop packet radio networks," *IEEE/ACM Trans. Netw.*, vol. 2, no. 1, pp. 23–29, Feb. 1994.

[13] J.-H. Ju and V. O. K. Li, "An optimal topology-transparent scheduling method in multihop packet radio networks," *IEEE/ACM Trans. Netw.*, vol. 6, no. 3, pp. 298–306, Jun. 1998.

[14] K. Oikonomou and I. Stavrakakis, "Analysis of a probabilistic topology-unaware TDMA MAC policy for ad hoc networks," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 7, pp. 1286–1300, Sep. 2004.

[15] Z. Cai, M. Lu, and C. N. Georghiades, "Topology-transparent time division multiple access broadcast scheduling in multihop packet radio networks," *IEEE Trans. Veh. Technol.*, vol. 52, no. 4, pp. 970–984, Jul. 2003.

[16] G. Zhang, J. Li, W. Zhang, and L. Zhou, "Topology-transparent schedule with reservation and carrier sense for multihop ad hoc networks," *IEEE Commun. Lett.*, vol. 10, no. 4, pp. 314–316, Apr. 2006.

[17] Q. Sun, V. O. K. Li, and K.-C. Leung, "A framework for topology-transparent scheduling in wireless networks," in *Proc. IEEE Veh. Technol. Conf.*, May 2010, pp. 1–5.

[18] C. N. Xu, "An algorithm for improving throughput guarantee of topology-transparent MAC scheduling strategy," *Wireless Sensor Netw.*, vol. 2, pp. 801–806, Oct. 2010.

[19] Y. Liu, V. O. K. Li, K.-C. Leung, and L. Zhang, "Topology-transparent scheduling in mobile ad hoc networks with multiple packet reception capability," *IEEE Trans. Wireless Commun.*, vol. 13, no. 11, pp. 5940–5953, Nov. 2014.

[20] Y. Liu, V. O. K. Li, K.-C. Leung, and L. Zhang, "Performance improvement of topology-transparent broadcast scheduling in mobile ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 9, pp. 4594–4605, Nov. 2014.

[21] V. D. P. Marius, and M. F. Singer, "Galois theory of linear differential equations," *Grundlehren Der Math. Wissenschaften*, vol. 328, no. 3, pp. 1–82, 2003.

[22] X. Hu, T. H. S. Chu, V. C. M. Leung, E. C.-H. Ngai, P. Kruchten, and H. C. B. Chan, "A survey on mobile social networks: Applications, platforms, system architectures, and future research directions," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1557–1581, 3rd Quart., 2015.

[23] M. Qiao, H. Zhao, S. Huang, L. Zhou, and S. Wang, "Optimal channel selection based on online decision and offline learning in multichannel wireless sensor networks," *Wireless Commun. Mobile Comput.*, vol. 2017, Nov. 2017, Art. no. 7902579, doi: 10.1155/2017.7902579.

[24] X. Hu, T. H. S. Chu, H. C. B. Chan, and V. C. M. Leung, "Vita: A crowdsensing-oriented mobile cyber-physical system," *IEEE Trans. Emerg. Topics Comput.*, vol. 1, no. 1, pp. 148–165, Jun. 2013.

[25] Y. Lin, C. Wang, J. Wang, and Z. Dou, "A novel dynamic spectrum access framework based on reinforcement learning for cognitive radio sensor networks," *Sensors*, vol. 16, no. 10, p. 1675, 2016.

[26] N. Morozs, T. Clarke, and D. Grace, "Distributed heuristically accelerated Q-learning for robust cognitive spectrum management in LTE cellular systems," *IEEE Trans. Mobile Comput.*, vol. 15, no. 4, pp. 817–825, Apr. 2016.

[27] J. N. Tsitsiklis and B. Van Roy, "An analysis of temporal-difference learning with function approximation," *IEEE Trans. Autom. Control*, vol. 42, no. 5, pp. 674–690, May 1997.

[28] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Mach. Learn.*, vol. 3, no. 1, pp. 9–44, 1988.

[29] N. N. Schraudolph, P. Dayan, and T. J. Sejnowski, "Temporal difference learning of position evaluation in the game of go," in *Proc. Adv. Neural Inf. Process. Syst.*, 1994, pp. 817–824.

[30] X. Guo, S. Singh, H. Lee, R. Lewis, and X. Wang, "Deep learning for real-time Atari game play using offline Monte–Carlo tree search planning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3338–3346.

[31] H. Johannes and S. David, "Deep reinforcement learning from self-play in imperfect-information games," in *Proc. Adv. Neural Inf. Process. Syst. Workshop*, 2016, pp. 1–10.

[32] J. Foerster, N. Nardelli, G. Farquhar, T. Afouras, P. H. S. Torr, P. Kohli, and S. Whiteson, "Stabilising experience replay for deep multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1–10.

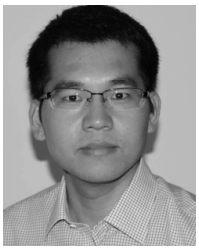
[33] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1–9.

[34] D. Zhao, H. Wang, K. K. Shao, and Y. Zhu, "Deep reinforcement learning with experience replay based on SARSA," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Dec. 2016, pp. 1–6.

[35] D. Zhao, Q. Zhang, D. Wang, and Y. Zhu, "Experience replay for optimal control of nonzero-sum game systems with unknown dynamics," *IEEE Trans. Cybern.*, vol. 46, no. 3, pp. 854–865, Mar. 2016.

[36] S. Yasini, A. Karimpour, and M. B. Naghibi Sistani, "Data-based reinforcement learning algorithm with experience replay for solving constrained nonzero-sum differential games," in *Proc. Int. Conf. Math. Theory Netw. Syst.*, Jul. 2014, pp. 701–707.

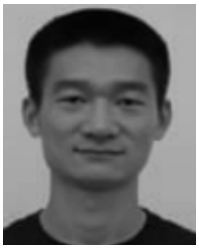
MU QIAO received the B.Sc. degree in information engineering from Hunan Agricultural University, Changsha, China, in 2011, and the M.Sc. degree in software engineering from the National University of Defense Technology, Changsha, China, in 2013, where he is currently pursuing the Ph.D. degree with the Department of Communication Engineering, College of Electronic Science. His research interests include MAC protocol, machine learning, and cognitive radio networks.



HAITAO ZHAO received the M.S. and Ph.D. degrees in information and communication engineering from the National University of Defense Technology (NUDT), Changsha, China, in 2004 and 2009, respectively. He is currently an Associate Professor with the College of Electronic Science, NUDT. Prior to this, from 2008 to 2009, he was a Visiting Research Associate with the Institute of ECIT, Queen's University of Belfast, U.K. From 2014 to 2015, he conducted Post-Doctoral Research at Hong Kong Baptist University. His main research interests include cognitive radio networks, self-organized networks, and cooperative communications. He is currently a member of ACM, a Mentor Member of the IEEE 1900.1 Standard Group, and a member of World-Wide University Network Cognitive Communications Consortium. He has served as a TPC member of the IEEE ICC'14-17, Globecom'16, and APCC'16, and guest-edited several international journal special issues on cognitive radio networks.

LI ZHOU received the B.S., M.S., and Ph.D. degrees from the National University of Defense Technology (NUDT), China, in 2009, 2011, and 2015, respectively. From 2013 to 2014, he was a Visiting Scholar with The University of British Columbia, Canada. He is currently an Assistant Professor with the College of Electronic Science, NUDT. His research interests are in the area of software-defined radios, software-defined networks, and heterogeneous networks. He served as a TPC member of the IEEE CIT 2017, a Keynote Speaker in ICWCNT 2016, and Co-Chair in ITA 2016. His research contributions have been published and presented in over 20 prestigious journals and conferences, such as the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, *Ad Hoc Networks*, WInnComm 2017, the IEEE INFOCOM 2015, and the IEEE GLOBECOM 2014.

CHUNSHENG ZHU received the Ph.D. degree in electrical and computer engineering from The University of British Columbia, Canada. He is currently a Post-Doctoral Research Fellow with the Department of Electrical and Computer Engineering, The University of British Columbia. He has authored over 100 publications published by refereed international journals (e.g., the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, the IEEE TRANSACTIONS ON CLOUD COMPUTING, *ACM Transactions on Embedded Computing Systems*, and *ACM Transactions on Cyber-Physical Systems*), magazines (e.g., the *IEEE Communications Magazine*, the *IEEE Wireless Communications Magazine*, and the *IEEE Network Magazine*), and conferences (e.g., IEEE INFOCOM, IEEE IECON, IEEE SECON, IEEE DCOSS, IEEE ICC, and IEEE GLOBECOM). His research interests mainly include Internet of Things, wireless sensor networks, cloud computing, big data, social networks, and security.



SHENGCHUN HUANG received the B.S. and M.S. degrees from the National University of Defense Technology, China, in 2005 and 2008, respectively. From 2009 to 2011, he was a Visiting Ph.D. Student with The University of British Columbia, Canada. His research interests include broadband wireless access network and radio resource management.