IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 27, 2020, accepted September 10, 2020, date of publication September 18, 2020,
date of current version September 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3025043

Q-Learning Based Scheduling With Successive Interference Cancellation

EZGI METE AND TOLGA GIRICI, (Senior Member, IEEE)
Department of Electrical and Electronics Engineering, TOBB University of Economics and Technology, 06560 Ankara, Turkey
Corresponding author: Ezgi Mete (emete@etu.edu.tr)

ABSTRACT This work studies the problem of scheduling using Q-learning, which is a reinforcement learning algorithm, in a Successive Interference Cancellation (SIC) - enabled wireless ad hoc network. Distributed Q-learning algorithm tries to find the best schedule for the transmission of maximum number of packets in the presence of the SIC technique. Performance of the algorithm is compared to the case where Q-learning is applied to a wireless network without SIC. In addition to that, the number of successful transmissions of our algorithm is compared to the optimal solution with and without SIC. Numerical results reveal that Q-learning based scheduling with SIC shows an improved performance compared to Q-learning scheduling without SIC and the optimal solution without SIC. Also, Q-learning scheduling with SIC shows similar performance to optimal scheduling with SIC when transmitting a reasonable number of packets. Thus, combining Q-learning and the SIC technique in wireless ad hoc networks is an effective approach to increase the number of transmitted packets.

INDEX TERMS Q-learning, successive interference cancellation, scheduling, wireless ad hoc network.

I. INTRODUCTION

Link scheduling, which decides the links that will transmit at a given time slot, is an important problem of wireless networks due to the nature of wireless media and the problem of interference. As the number of users in wireless networks grow, so does the demand for higher throughput services. Interference significantly limits the performance of wireless networks, which is why scheduling algorithms with different interference management techniques have been investigated in many works. In classical link scheduling algorithms, a node can transmit to only one node or receive from only one node at a given time and other simultaneous transmissions are treated as interference [1]. These algorithms have limited performance due to restricted number of transmissions that can be made in a single time slot.

Successive Interference Cancellation (SIC) is another method that is used to manage interference. It exploits the interference rather than avoiding it. Simultaneous reception of signals and interference rejection capabilities of this technique increase the system performance [1]. In SIC, before decoding the wanted signal at the receiver, the stronger signals must be decoded [2]. SIC uses the differences in the transmitting users' channel gains to order the received signals and decoding occurs successively according to this order. Decodings are successful if the received signal-to-interference plus noise ratios (SINR) are above a specific threshold. SIC has been shown to improve the performance of wireless networks [3]–[13]. It has been initially studied under cellular networks. In [3], the authors propose an uplink Code-Division Multiple Access (CDMA) system where the base station receiver has SIC capability. Optimizing the total transmit power and order of user detection and cancellation, it is shown that the proposed algorithm achieves reduced transmit power compared to the traditional ordering approach. The authors in [4] study a downlink Non-Orthogonal Multiple Access (NOMA) system where SIC is implemented at the receiver. Decoding times of signals for the cases of correct and incorrect decoding order of signals at the receiver are presented. Results reveal that the decoding time of a user equipment (UE) signal decreases as the UE distance from the base station increases. In [5], a decentralized full- duplex NOMA-based vehicle-to-everything (V2X) system is introduced. It is shown that when the NOMA and the Orthogonal Multiple Access (OMA) schemes of the system are compared, half-duplex NOMA scheme has a better achievable throughput performance than half-duplex OMA scheme. However, using SIC in wireless ad hoc networks requires advanced coordination and excessive number of control messages, which limits the feasibility of this technique in such

The associate editor coordinating the review of this manuscript and approving it for publication was Xujie Li.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

scenarios. In [6], joint routing and scheduling in a wireless network is investigated. It is shown that SIC can increase the throughput performance of a network by 300%. [7] focuses on joint routing, scheduling and rate control in a wireless network. According to the results, SIC and bit rate adaptation capabilities can increase throughput gains by at least 20% in comparison to a wireless network that does not have SIC capability. In [8], joint scheduling and routing is considered. A 20 node network shows 47% and a 50 node network shows 43.5% increase in throughput when SIC is used in wireless reception. In [9], cooperative transmission, maximum link activation and SIC are studied together. It is shown that SIC provides more improvement than cooperative transmission to the baseline algorithm that does not use SIC or cooperative transmission in the wireless network. In [10], a joint interference exploitation and avoidance algorithm improves throughput performance by 47% compared to basic interference avoidance model. In [11], a joint study of mobile base station movement and SIC is presented. It is evaluated that using SIC decreases the minimum scheduling time for a 50 node network by an average of 34% compared to the case that does not use SIC. Reference [12] considers link scheduling and power allocating for a SIC based underwater acoustic network. Simulation results show that the proposed algorithm improves throughput by 100% over traditional Time-Division Multiple Access (TDMA). Reference [13] investigates joint routing, network coding and scheduling with SIC in energy harvesting networks. According to the study, compared to the case where only network coding is used, combining network coding with SIC decreases the number of time slots required for transmission.

For the purpose of finding algorithmic solutions to complex problems, machine learning is an option that can be used instead of traditional approaches [14]. Machine learning examines the data that is given to it, which is a training set or is a result of the experience gained within the environment, and foresees the next task to execute [15]. Automatically becoming aware of the dynamics of the environment and changing how they act according to these dynamics in short amount of time are the objectives of the algorithms that belong to machine learning [16]. Recent research has used machine learning methods extensively in communications systems. For example, the work in [17] has used genetic algorithms in determining vehicles that disseminate program codes in a smart city. The authors in [18] utilize reinforcement learning for resource allocation in vehicular networks. The work in [19] addresses the data collection problem in an underwater sensor network and uses clustering, which is also a machine learning method.

There are three main types of machine learning: supervised learning, unsupervised learning and reinforcement learning. In supervised learning, system model is developed by using training data which is made up of inputs and the corresponding correct outputs (a labeled data set) [20]. The objective of supervised learning is to learn the input and output relationship [14]. Labels, past and present information acquired by the system are used to form a function that predicts outputs by correlating patterns in the data set with the intended outputs. The algorithm adapts the system model according to the errors between the achieved and wanted results. With enough training, the system can provide the output to a new input data [21]. Unsupervised learning training data set consists of inputs without any designated outputs (inputs are unlabelled) [14]. In unsupervised learning, the system discovers the hidden patterns in the data on its own since there is no output vector [21]. The fundamental objective is to categorize the data into separate groups by looking at their similarities [20].

Reinforcement learning [22] belongs neither to supervised learning nor to unsupervised learning. It involves supervision but compared to supervised learning, supervision is not in the structure of an identified output for the input. The environment, which can be seen as an external trainer, sends a reinforcement signal as feedback to the reinforcement learning algorithm, provided that an output is selected for a given input. This output is the action that the learner takes and the feedback received is the reward that implies how much the objective of the learner is achieved [14]. Reinforcement learning, is a learning process where an agent learns how to act in an environment such that it achieves a goal. The environment does not tell the agent to select a specific action, so the agent has to figure out which actions result in the highest reward by trying them [23]. The immediate reward, the next state and all the rewards that follow are influenced by the actions [22]. During the learning process, the agent chooses an action among other possible actions and receives a reward that evaluates the quality of this selection, which is based on whether that action brings the agent closer to the goal or not. By interacting with its environment this way, it tries many actions and with the knowledge it gains, after some time, it learns to pick the actions that give the best rewards. Q-learning [24], is a reinforcement learning algorithm that is known for its simple implementation. The agent learns to take the actions with best rewards either by exploring the environment or using the action that was learned that gives the best reward.

In communication networks, wireless nodes can make optimal or near-optimal scheduling decisions using reinforcement learning [25]. Scheduling in wireless networks using Q-learning has been studied in some works in literature. In [26], using Q-learning, it was shown that a high number of packets can be successfully transmitted and the average transmission power can be kept at minimum at the same time in a wireless ad hoc network. In [27], using Q-learning, packet loss was reduced by %60 and network goodput was increased by 67% compared to algorithms that do not use Q-learning. In [28], it is shown that using Q-learning can reduce energy consumption and average delay in a wireless sensor network. Although Q-learning's application to wireless ad hoc networks is being investigated in these works, the combination of Q-learning and SIC technique's effects on system performance is not present in these studies. There are a few works in literature that study the effects of using SIC and Q-learning in

wireless networks. In [29], Q-learning and SIC are used to allocate resources to maximize normalized throughput. In [30], an algorithm that uses SIC, Q-learning and a power control scheme is proposed to improve throughput. In [31], Q-learning and SIC based power allocation is proposed to increase the sum data rate of users. The algorithms in these studies involve the incorporation of SIC and Q-learning but they are applied to cellular networks, some of them do not involve scheduling and do not have realistic channel models.

In this work, we apply Q-learning both to an SIC-enabled wireless network and to an interference avoidance wireless network that assumes interference as noise. As benchmarks, two optimal solutions based on [8] are used to find the upper bounds for the achievable number of successfully transmitted packets. First optimal solution assumes SIC in the wireless reception and the second one assumes interference as noise in the wireless network.

Our contribution is the study of scheduling by applying Q-learning to an SIC-based wireless distributed ad hoc network with a realistic channel model and to show that using Q-learning and SIC together results in an increase in performance in the number of scheduled packets compared to the performance of Q-learning applied to the network without SIC, an improved performance compared to that of optimal scheduling without SIC and similar performance to optimal scheduling with SIC up to a certain load.

The rest of the paper is organized as follows. Section II describes the system model. Section III formulates the optimal solution. Section IV provides the details of the proposed Q-learning scheduling with SIC algorithm and the Q-learning scheduling without SIC algorithm. Section V presents the simulation results. Section VI concludes the paper.

II. SYSTEM MODEL

N nodes are distributed uniformly random and node locations are fixed. There are a number of time slots per frame, during which multiple packets can be scheduled for transmission. There are L links with a total number of B packets to be transmitted. All receiving nodes can perform SIC. N_o is the additive white gaussian noise. A node can not transmit and receive at the same time. Channel gains remain constant. A combined path loss, shadowing and fading model is used for calculating channel gains [32].

$$g_{i,j}^{dB} = -PL_{d0} - 10 \times \gamma \times \log 10(d_{i,j}/d0) - \psi^{dB} - r^{dB} \quad (1)$$

where $g_{i,j}^{dB}$ is the channel gain between nodes i and j, γ is the path loss coefficient, $d_{i,j}$ is the distance between nodes i and j, $d0$ is a reference distance for the far antenna field [32], ψ^{dB} is the shadowing loss, r^{dB} is the fading loss and PL_{d0} is the path loss at $d0$ in decibels (dB). Node transmit powers are constant. For the transmission from node i to j, received power at node j is denoted by $P_{i,j}$.

A. SIC-BASED TRANSMISSION

In SIC, a node that receives multiple signals in the same time slot has the ability to decode them iteratively. First, received signal powers at a node from intended and unintended transmissions are ordered in a decreasing manner. Then one by one, starting from the strongest signal, a signal is decoded and subtracted from the total received signal until all signals are decoded or decoding of the remaining signals is stopped when the next signal to be decoded is not decodable [8].

A signal from node i to node j in time slot t is decodable if $SINR_{i,j}[t]$ is greater than or equal to β [8]:

$$SINR_{i,j}[t] = \frac{P_{i,j}}{\sum_{k\neq i}^{P_{k,j}\leq P_{i,j}} \sum_{l\in I_k} P_{k,j} \cdot x_{k,l}[t] + N_o} \geq \beta \quad (2)$$

$P_{k,j}$ are the received signal powers from other nodes k that are transmitting in the time slot t with values less than $P_{i,j}$, index l represents the intended receiving nodes of transmitting nodes k, I_k are the nodes in node k's neighborhood, $x_{k,l}[t]$ is a binary scheduling variable which is set to 1 if node k is set to successfully transmit data to node l in time slot t or is set to 0.

B. TRANSMISSION ASSUMING INTERFERENCE AS NOISE

In an interference avoidance wireless network, when there is a node i transmitting an intended signal to node j, if the SINR at receiver j is greater than or equal to β, the signal from node i to node j can be decoded successfully and the transmission is successful. Receiver j treats all other simultaneous interfering transmissions as noise. In this case, SINR for successful decoding of an intended signal from node i to node j is given as [8]:

$$SINR_{i,j}[t] = \frac{P_{i,j}}{\sum_{k\neq i} \sum_{l\in I_k} P_{k,j} \cdot x_{k,l}[t] + N_o} \geq \beta \quad (3)$$

$P_{k,j}$ are the interference powers from other nodes k that are transmitting in the time slot t. Symbols l, I_k and $x_{k,l}[t]$ have the same definitions as the ones stated below (2) in Section II.A.

III. OPTIMAL SOLUTION

The problem is to determine which links to activate in each time slot t of a frame where $0 \leq t \leq T$. The objective of maximizing the number of transmitted packets is expressed as,

$$max_{\mathbf{x}} \sum_i \sum_j \sum_t x_{i,j}[t] \quad (4)$$

where $\mathbf{x} = \{x_{i,j}[t]\}$ is the vector of binary scheduling variables, and $x_{i,j}[t]$ is set to 1 if node i is set to successfully transmit data to node j in time slot t or is set to 0, otherwise [8].

For the optimal solution with SIC, when node i successfully transmits data to node j (i.e., $x_{i,j}[t] = 1$), this means that the SINRs of the stronger simultaneous transmissions of nodes m received at node j are greater than or equal to β.

It also means that, the SINR of the signal from node i to node j is greater than or equal to β. Node m transmitting to node n is given by $\sum_{n\in I_m} x_{m,n}[t] = 1$. The following statement can be made [8]:

$$If\ x_{i,j}[t] = 1\ and \sum_{n\in I_m} x_{m,n}[t] = 1,\ then\ SINR_{m,j}[t] \geq \beta \quad (j \in \mathcal{N}, i \in I_j, m \neq i, P_{m,j} > P_{i,j}, 1 \leq t \leq T) \tag{5}$$

$\lambda_m[t]$ is introduced to simplify (5):

$$\lambda_m[t] = \sum_{n\in I_m} x_{m,n}[t] \quad (m \in \mathcal{N}, 1 \leq t \leq T) \tag{6}$$

For the half-duplex transmission formulation (7), $\lambda_j[t]$ is 0 when $x_{i,j}[t]$ is greater than 0, meaning that j is receiving and not transmitting. $\lambda_j[t]$ is 1 when $x_{i,j}[t]$ is 0, meaning that j is transmitting and not receiving:

$$\lambda_j[t] + \frac{1}{|I_j|}\sum_{i\in I_j} x_{i,j}[t] \leq 1 \quad (j \in \mathcal{N}, 1 \leq t \leq T) \tag{7}$$

The conditions $x_{i,j}[t] = 1$ and $\lambda_m[t] = 1$ are combined into one condition, and a new variable is formed which is $y_{(i,j)(m)}[t]$:

$$y_{(i,j)(m)}[t] \geq x_{i,j}[t] + \lambda_m[t] - 1 \quad (j \in \mathcal{N}, i \in I_j, m \neq i, P_{m,j} > P_{i,j}, 1 \leq t \leq T) \tag{8}$$

$$x_{i,j}[t] \geq y_{(i,j)(m)}[t] \quad (j \in \mathcal{N}, i \in I_j, m \neq i, P_{m,j} > P_{i,j}, 1 \leq t \leq T) \tag{9}$$

$$\lambda_m[t] \geq y_{(i,j)(m)}[t] \quad (j \in \mathcal{N}, i \in I_j, m \neq i, P_{m,j} > P_{i,j}, 1 \leq t \leq T) \tag{10}$$

Using $\lambda_m[t]$ definition, (5) can be rewritten as:

$$If\ x_{i,j}[t] = 1\ and\ \lambda_m[t] = 1,\ then\ SINR_{m,j}[t] \geq \beta \quad (j \in \mathcal{N}, i \in I_j, m \neq i, P_{m,j} > P_{i,j}, 1 \leq t \leq T) \tag{11}$$

Since $x_{i,j}[t] = 1$ and $\lambda_m[t] = 1$ are combined to form the condition $y_{(i,j)(m)}[t] = 1$, (11) can be rewritten as:

$$If\ y_{(i,j)(m)}[t] = 1,\ then\ SINR_{m,j}[t] \geq \beta \quad (j \in \mathcal{N}, i \in I_j, m \neq i, P_{m,j} > P_{i,j}, 1 \leq t \leq T) \tag{12}$$

Using (6) and substituting m in place of i in $SINR_{i,j}[t]$ in (2), the following equation is found for $SINR_{m,j}[t]$:

$$SINR_{m,j}[t] = \frac{P_{m,j}}{\sum_{k\neq i}^{P_{k,j}\leq P_{m,j}} P_{k,j}\cdot\lambda_k[t] + N_o} \tag{13}$$

Using (12) and (13), (14) is obtained:

$$P_{m,j} - \sum_{k\neq m}^{P_{k,j}\leq P_{m,j}} \beta P_{k,j}\lambda_k[t] - \beta N_o \geq (1 - y_{(i,j)(m)}[t])D_{i,j,m} \quad (j \in \mathcal{N}, i \in I_j, m \neq i, P_{m,j} > P_{i,j}, 1 \leq t \leq T) \tag{14}$$

The constraint corresponding to the SINR of the signal from node i to node j given $x_{i,j}[t] = 1$ is:

$$If\ x_{i,j}[t] = 1,\ then\ SINR_{i,j}[t] \geq \beta \quad (j \in \mathcal{N}, i \in I_j, 1 \leq t \leq T) \tag{15}$$

(16) is obtained from (15) and (6):

$$P_{i,j} - \sum_{k\neq i}^{P_{k,j}\leq P_{i,j}} \beta P_{k,j}\lambda_k[t] - \beta N_o \geq (1 - x_{i,j}[t])H_{i,j} \quad (j \in \mathcal{N}, i \in I_j, 1 \leq t \leq T) \tag{16}$$

In (17), $B_{i,j}$ is the number of packets to be transmitted from node i to node j. The sum of number of packets scheduled from node i to node j must be less than or equal to the total number of packets to be transmitted from node i to node j:

$$B_{i,j} \geq \sum_t x_{i,j}[t] \quad (j \in \mathcal{N}, i \in I_j, 1 \leq t \leq T) \tag{17}$$

In (6), $\lambda_m[t]$ is a binary variable that is set to 1 if node m is transmitting in time slot t irrespective of the node that it is transmitting to [8]. In (8), $y_{(i,j)(m)}[t]$ is a binary variable that is set to 1 when $x_{i,j}[t]$=1 and $\lambda_m[t]$=1. In (9) and (10), when $y_{(i,j)(m)}[t]$=1, $x_{i,j}[t]$ and $\lambda_m[t]$ are equal to 1 as well. Constraints (14) and (16) are the received SINR constraints for the intended and unintended transmissions that need to be decoded. In (14), $D_{i,j,m}$ is a lower bound of $P_{m,j} - \sum_{k\neq m}^{P_{k,j}\leq P_{m,j}} \beta P_{k,j}\lambda_k[t] - \beta N_o$ and is set to $P_{m,j} - \sum_{k\neq m}^{P_{k,j}\leq P_{m,j}} \beta P_{k,j} - \beta N_o$. In (16), $H_{i,j}$ is a lower bound of $P_{i,j} - \sum_{k\neq i}^{P_{k,j}\leq P_{i,j}} \beta P_{k,j}\lambda_k[t] - \beta N_o$ and is set to $P_{i,j} - \sum_{k\neq i}^{P_{k,j}\leq P_{i,j}} \beta P_{k,j} - \beta N_o$. For the optimal solution with SIC, (6), (7), (8), (9), (10), (14), (16) and (17) are used. For the optimal solution without SIC, constraints (6), (7), (17) and,

$$P_{i,j} - \sum_{k\neq i} \beta P_{k,j}\lambda_k[t] - \beta N_o \geq (1 - x_{i,j}[t])M_{i,j} \quad (j \in \mathcal{N}, i \in I_j, 1 \leq t \leq T) \tag{18}$$

are used to solve (4) since constraints (8), (9), (10), (14), and (16) are SIC constraints. In (18), $M_{i,j}$ is a lower bound of $P_{i,j} - \sum_{k\neq i} \beta P_{k,j}\lambda_k[t] - \beta N_o$ and is set to $P_{i,j} - \sum_{k\neq i} \beta P_{k,j} - \beta N_o$ [8]. (18) is obtained from (15) and (6) by applying the SINR equation of the interference as noise case.

IV. ALGORITHM

A. Q-LEARNING SCHEDULING WITH SIC

Q-learning [24], is a model-free method of reinforcement learning where agents do not need to have knowledge about the environment that they are in. In Q-learning, the goal is

to observe an agent's actions, noting the reward that is going to be received for these actions and picking the actions that will maximize the received rewards. In this work, we propose a method in which each node determines the time slots to transmit each of its packets using Q-learning and where the corresponding receivers can use SIC in order to be able to receive multiple packets simultaneously. Components of Q-learning are as follows [22]:

1) Agents: Agents learn to choose and perform the actions that provide the best rewards by interacting with their environment.
2) Reward: Reward determines whether an action is desirable or not. Agents base their choice of actions on these rewards at a given state.
3) Actions: Actions change the agent's state and the rewards it can receive.
4) State: the situation that the agent is in, in the environment.

In Q-learning, more than one episode of the agent's interaction with the environment needs to take place to choose actions and receive rewards, and in order to learn which actions to choose for maximum rewards. Q-learning estimates the quality of state-action pairs (s,a), which are known as the Q-values [26]. All the experience that the agent gains during the episodes are recorded in a Q-Table where each entry (Q-value) corresponds to a state-action pair. Q-values are updated using the equation [24]:

$$Q(s,a) = (1-\alpha)Q(s,a) + \alpha(r(s,a) + \gamma \max_{a'} Q(s',a')) \tag{19}$$

where s is the state, a is the action, s' is the next state and a' is the next action. r is the reward. α is the learning rate which determines how fast learning occurs and is given a value between 0 and 1. γ is the discount factor which determines how much importance is given to the future rewards in the current state and is given a value between 0 and 1.

In our model, to schedule B packets in T time slots we define the Q-learning components as follows:

1) Agents: Nodes
2) States: Transmitting nodes
3) Actions: Receiving nodes and corresponding time slots in the frame ($\{i,j\}[t]$)
4) Reward: If a transmission is successful then the reward value is set to 1 or to 0 if unsuccessful. We assume error-free feedback in case of successful transmission.

We define an episode as the frame that consists of T time slots. After each episode Q-values are updated according to the following equation:

$$Q_{i,j}[t] = (1-\alpha)Q_{i,j}[t] + \alpha(r_{i,j}[t]) \tag{20}$$

where i is the transmitting node, j is the receiving node, t is a time slot in the frame, α is the learning rate. The reward r is set to 1 if a transmission from node i to node j in time slot t is successful or to 0 if unsuccessful. Please note that we assume an ad hoc network, where the nodes have a very limited information of the network. Each node only knows the ID's of its neighbors and a positive feedback if the corresponding transmission is successful.

ϵ-greedy approach [33] is being used to determine the next action where ϵ is the exploration probability. In this method, a random action is selected with probability ϵ for exploration or the action that gives the maximum Q-value is selected with probability 1-ϵ for exploitation. A node must choose the actions that it has previously tried which resulted in high rewards, and in order to do that, the node should be able to find such high reward delivering actions by trying the ones that it has not tried yet. So, the node needs to exploit from experience to acquire the highest rewards and explore to find the actions that gives the highest rewards [22].

In **Algorithm 1**, **Q** is the Q-Table and is initialized to zero. ϵ_i is the exploration coefficient of each node i, α is the learning rate and is constant, N_e is the number of episodes (i.e. time frames, iterations) used by the algorithm for the learning process to take place. $P_{i,j}$ is the received power from node i at node j. $lp_{i,j}$ is the number of packets waiting to be transmitted at link (i,j), which is initially $lp_{i,j} = B_{i,j}$. np_i is the number of packets to be transmitted from node i, which is initially $np_i = \sum_{j \in I_i} B_{i,j}$.

$P_{i,j}$ denotes the received signal strength of transmission $\{i,j\}$ and $B_{i,j}$ denotes the number of available packets to transmit from each link. The algorithm gives the schedule $x_{i,j}[t], \forall i,j \in [1,N]$ as the output (Lines 1-2). Initializations are done in (Line 3). Episodes of the algorithm are shown (Lines 4-38). At each episode either exploration or exploitation is performed randomly (Lines 7-11). Each node assigns a packet to each time slot in (Lines 6-23). This is done either randomly (exploration) or by choosing the links and time slots with better Q-values (exploitation). Then, the transmissions are performed according to the determined schedule. Each node tries to decode the received signals. In order to decode its intended packet, a node has to decode not only that signal but all the other ones with greater received power (Lines 29-32). Note that a node cannot decode a packet if it's also transmitting (Line 27). Q-Table is updated according to the rewards (i.e. successes) in (Line 36). Exploration probability is updated in (Line 37). Each node has its own exploration probability. As the ratio of successfully transmitted packet increases, exploration probability decreases. If a node transmits every available packet in an episode, then it does not explore in the next episode. (Line 39) returns the successfully transmitted packets (i.e. $i,j,t\ s.t.\ s_{i,j}[t] = 1$) as the resulting schedule.

B. Q-LEARNING SCHEDULING WITHOUT SIC

The algorithm for Q-learning Scheduling without SIC is similar to the algorithm of Q-learning Scheduling with SIC, except that the part where SIC is applied is changed. Starting from the section of the code that starts with the initialization of **s** and **r** to 0, the part of the code in Q-learning Scheduling with SIC (Lines 24-37) are replaced with the code in

Algorithm 1 Q-Learning Scheduling With SIC

1: **Input:** $P_{i,j}, B_{i,j}, \forall i,j \in [1,N]$
2: **Output:** Schedule $x_{i,j}[t]$
3: Initialize: $\mathbf{Q} = 0$, $\epsilon_i = 0.1$, $\alpha = 0.1$, N_e, $lp_{i,j} = B_{i,j}$, $np_i = \sum_j lp_{i,j}$, $\forall i,j \in [1,N]$
4: **for** $e = 1 : N_e$ **do**
5: Initialize: $lp_{i,j} = B_{i,j}$, $np_i = \sum_j lp_{i,j}$, $\forall i,j \in [1,N]$;
6: **for** $i = 1 : N$ **do**
7: **if** `rand` $< \epsilon_i$ **then**
8: `exploration`;
9: **else**
10: `exploitation`;
11: **end if**
12: **for** $t = 1 : T$ **do**
13: **if** `exploration` **then**
14: randomly choose a receiver node j^* such that $lp_{i,j} > 0$ and assign to time slot t;
15: **else**
16: $[j^*] = \arg\max_{j:lp_{i,j}>0} Q(i,j)[t] \forall t$;
17: **end if**
18: $x_{i,j^*}[t] = 1$, $lp_{i,j^*} = lp_{i,j^*} - 1$, $np_i = np_i - 1$;
19: **if** $np_i = 0$ **then**
20: `break`;
21: **end if**
22: **end for**
23: **end for**
24: Initialize $\mathbf{s} = 0$, $\mathbf{r} = 0$;
25: **for** $t = 1 : T$ **do**
26: **for** $\forall (i,j)$ s.t. $x_{i,j}[t] = 1$ **do**
27: **if** $\sum_{l \in I_j} x_{j,l}[t] = 0$ **then**
28: Set $\mathcal{K} = \{k \in N | \sum_{l \in I_k} x_{k,l} = 1, P_{kj} \geq P_{ij}\}$
29: $SINR_{k,j}[t] = \frac{P_{kj}}{\sum_{k' \in \mathcal{K} \neq k}^{P_{k'j} \leq P_{kj}} \sum_{l \in I_{k'}} P_{k'j} x_{k'l}[t] + N_o}$, $\forall k \in \mathcal{K}$
30: **if** $SINR_{k,j}[t] > \beta, \forall k \in \mathcal{K}$ **then**
31: Set $s_{i,j}[t] = 1$ and $r_{i,j}[t] = 1$;
32: **end if**
33: **end if**
34: **end for**
35: **end for**
36: $Q_{i,j}[t] = (1-\alpha) Q_{i,j}[t] + \alpha r_{i,j}[t]$, $\forall i,j \in [1,N], \forall t \in [1,T]$;
37: $\epsilon_i = 0.9\epsilon_i + 0.1 \times 0.1 \left(1 - \frac{\sum_{j \in I_i} \sum_t s_{i,j}[t]}{\sum_{j \in I_i} B_{ij}}\right)$, $\forall i \in [1,N]$;
38: **end for**
39: Return: $\mathbf{x} = \mathbf{s}$;

Algorithm 2. For all time slots (Line 2) and transmitting links (Line 3) intended transmissions are checked. In (Line 4), the half duplex constraint is checked. (Lines 5-8) check the minimum SINR threshold. As seen in the algorithm, SIC is not applied. Reward definition, Q-Table update and exploration probability update are the same.

Algorithm 2 Q-learning Scheduling Without SIC (First Part is the Same)

Initialize $\mathbf{s} = 0$, $\mathbf{r} = 0$;
for $t = 1 : T$ **do**
  for $\forall (i,j)$ s.t. $x_{i,j}[t] = 1$ **do**
    if $\sum_{l \in I_j} x_{j,l}[t] = 0$ **then**
      $SINR_{i,j}[t] = \frac{P_{i,j}}{\sum_{k \neq i} \sum_{l \in I_k} P_{k,j} x_{k,l}[t] + N_o}$
      if $SINR_{i,j}[t] > \beta$ **then**
        Set $s_{i,j}[t] = 1$ and $r_{i,j}[t] = 1$;
      end if
    end if
  end for
end for
$Q_{i,j}[t] = (1-\alpha) Q_{i,j}[t] + \alpha r_{i,j}[t]$, $\forall i,j \in [1,N], \forall t \in [1,T]$;
$\epsilon_i = 0.9\epsilon_i + 0.1 \times 0.1 \left(1 - \frac{\sum_{j \in I_i} \sum_t s_{i,j}[t]}{\sum_{j \in I_i} B_{ij}}\right)$, $\forall i \in [1,N]$;

TABLE 1. Simulation parameters.

Parameter	Definition	Value
N	Number of nodes	10
D_{max}	Network area radius	100 m
γ	Pathloss exponent	2.5
N_o	AWGN Noise Power	10^{-5} W
P_t	Node transmit power	0.3
β	SINR threshold	1.5
ϵ	Exploration coefficient	0.1
α	Learning rate	0.9
$episodes$	Number of episodes	5000

V. SIMULATION RESULTS

We perform simulations for a range of values of B, the total number of packets to be transmitted, T, the total number of time slots, and *episodes*, the total number of iterations. Simulation parameters are listed in Table 1. We compare four algorithms, which are 1) Q-learning scheduling with SIC 2) Q-learning scheduling without SIC, 3) Optimal scheduling with SIC and 4) Optimal scheduling without SIC. Presented results are produced from the results of 100 independent trials. For Q-learning scheduling with and without SIC, the maximum number of packets scheduled in all episodes and the 95^{th} percentile values of the number of packets scheduled in all episodes are plotted. The values chosen for the Q-learning parameters ϵ and α are the commonly used parameters in the literature. These typical values prove to be suitable for this scenario.

Fig. 1 shows the average number of packets transmitted vs. the number of time slots. As the number of time slots increases, for each algorithm the number of packets transmitted also increases and converges to 30 packets. Optimal scheduling with SIC performs the best with the highest number of packets transmitted. Not as many links can be scheduled for a given time slot value T for lower values of T due to interference. The following improvement percentage analyses were done for the maximum and 95^{th} percentile values of the number of packets scheduled at the low T value, $T = 10$. This is because as T increases, the performances

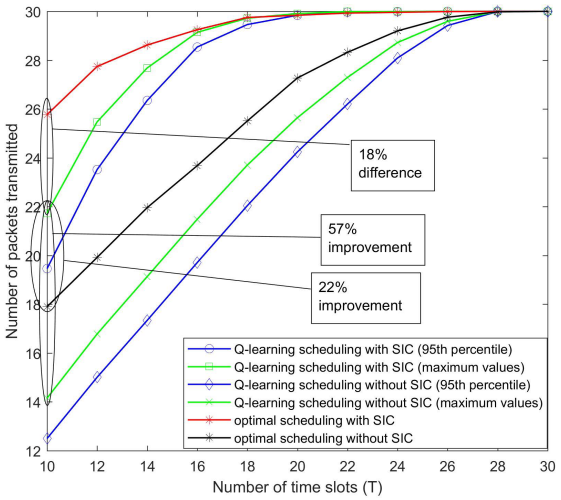

FIGURE 1. Number of packets transmitted vs Number of time slots (N = 10, B = 30, episodes = 5000).

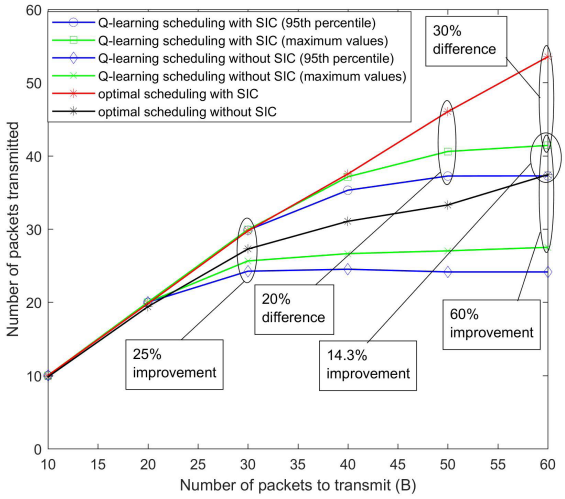

FIGURE 2. Number of packets transmitted vs Number of total packets (N = 10, T = 20, episodes = 5000).

of the algorithms converge. However, at lower values of T, the performance differences are apparent. The improvement percentage values for the maximum values of the number of packets scheduled are shown in Fig. 1. Looking at the maximum values of the number of packets scheduled using Q-learning scheduling with SIC algorithm and Q-learning scheduling without SIC algorithm, the Q-learning scheduling with SIC algorithm can transmit up to 57% more packets than the Q-learning scheduling without SIC algorithm. Maximum values of Q-learning scheduling with SIC show up to a 22% improvement over optimal scheduling without SIC. Q-learning scheduling with SIC (maximum values) and optimal scheduling with SIC have an 18% difference in performance. From $T = 16$ to $T = 30$, Q-learning scheduling with SIC (maximum values) performs similar to optimal scheduling with SIC. For the 95^{th} percentile values of the number of packets scheduled, up to 57.6% improvement can be seen in Q-learning scheduling with SIC over Q-learning scheduling without SIC. Q-learning scheduling with SIC (95^{th} percentile) shows 10% improvement over optimal scheduling without SIC. There is a 32% performance difference between Q-learning scheduling with SIC (95^{th} percentile) and optimal scheduling with SIC. From $T = 20$ to $T = 30$, Q-learning scheduling with SIC (95^{th} percentile) performs similar to optimal scheduling with SIC.

Fig. 2 shows the average number of packets transmitted vs the number of packets to transmit. For all algorithms, as the number of packets to transmit increases, congestion increases and the number of packets transmitted are less than the initial number of packets set for transmission. For optimal scheduling with SIC and Q-learning scheduling with SIC, the number of transmitted packets are less than the initial number of packets starting from $B = 30$. For optimal scheduling without SIC and Q-learning scheduling without SIC, the number of transmitted packets are less than the initial number of packets starting from $B = 20$. Q-learning scheduling with SIC begins to saturate at $B = 30$ and Q-learning scheduling without SIC begins to saturate at $B = 20$. The following analyses were done for a high load of $B = 60$. For the maximum values of Q-learning scheduling with SIC, the algorithm shows a 60% increase in performance compared to the maximum values of Q-learning scheduling without SIC algorithm. The maximum values of Q-learning scheduling with SIC show 14.3% better performance with respect to optimal scheduling without SIC. There is a difference in performance by 30% between Q-learning scheduling with SIC (maximum values) and optimal scheduling with SIC. From $B = 10$ to $B = 40$, Q-learning scheduling with SIC (maximum values) performs similar to optimal scheduling with SIC. For the 95^{th} percentile values of the number of packets scheduled, Q-learning scheduling with SIC shows 25% improvement compared to Q-learning scheduling without SIC at a lower load of $B = 30$. Between $B = 20$ and $B = 60$, Q-learning scheduling with SIC (95^{th} percentile) shows improvement over optimal scheduling without SIC but at $B = 60$ they transmit the same number of packets. Optimal scheduling with SIC and Q-learning scheduling with SIC (95^{th} percentile) have a performance difference of 20% at a higher load of $B = 50$ but from $B = 10$ to $B = 30$, Q-learning scheduling with SIC (95^{th} percentile) performs similar to optimal scheduling with SIC. The improvement percentage values for the maximum values and the 95^{th} percentile values of the number of packets scheduled are shown in Fig. 2.

Fig. 3 shows the number of packets transmitted vs the number of episodes used in the Q-learning algorithms. In only around 100 episodes Q-learning with SIC quickly learns to transmit almost 24/30 packets. It took around 200 episodes to learn to transmit 26 packets. Q-learning did not get stuck at a local optimum and continued to get better to reach 29 packets before it reaches 2000 episodes. Q-learning without SIC learns to transmit 18 packets then converge towards 22 packets before reaching 2000 episodes. High number of

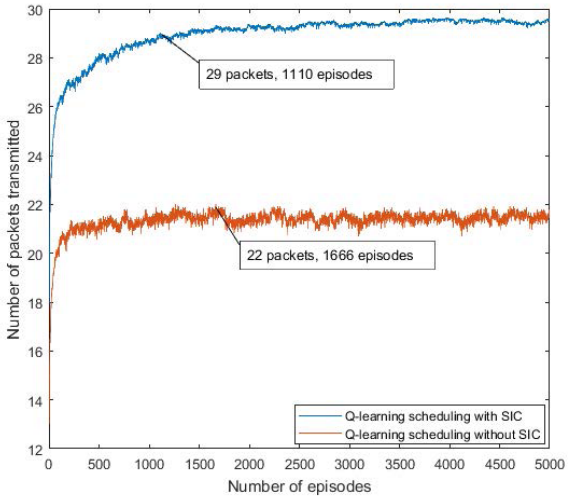

FIGURE 3. Number of packets transmitted vs Number of episodes.

packets transmitted by Q-learning scheduling with SIC in short amount of time makes it suitable for real time applications.

A brief discussion on Q-learning is in order. The main limitation of Q-learning is that it works in discrete and finite state and action spaces. Fortunately, in our case, the neighbors of a node and time slots are discrete and finite sets. Another limitation is the complexity. The time required to look up the Q-Table increases linearly with the number of neighbors and the number of slots in a frame [34]. Furthermore, as the network gets denser, the collisions increase (especially in the initial episodes), the algorithm may get stuck at a local optimum and result in a waste of resources. However, the major advantage of Q-learning is that it requires minimal feedback, no coordination among nodes, and no assumptions about the network and channel model. The only feedback required is the information whether a transmission is successful or not. Secondly, the learning algorithm creates its own experience and learns on-the-fly. As seen in Fig. 3 quite a lot of packets can still be transmitted even when the algorithm is far from convergence.

VI. CONCLUSION

In this work, we studied distributed scheduling using Q-learning algorithm with successive interference cancellation (SIC) at the receivers in order to achieve the highest number of successfully transmitted packets in a wireless ad hoc network. We compared our results to the scheduling case where Q-learning is applied to a network that assumes interference as noise. Furthermore, the performance of optimal scheduling solutions with and without SIC are also used to compare the performance of the Q-learning algorithms. Results show that the Q-learning scheduling with SIC algorithm outperforms the Q-learning without SIC and the optimal scheduling without SIC algorithms. Furthermore, it performs similar to the optimal scheduling with SIC algorithm in a scenario of reasonable load.

Possible future directions for research include minimum length scheduling based on Q-learning with SIC and power domain multiplexing. In addition to that, joint routing and scheduling based on Q-learning in the presence of SIC can be studied.

REFERENCES

[1] M. Kontik and S. Coleri Ergen, "Scheduling in successive interference cancellation based wireless ad hoc networks," *IEEE Commun. Lett.*, vol. 19, no. 9, pp. 1524–1527, Sep. 2015.
[2] S. Lv, W. Zhuang, X. Wang, and X. Zhou, "Scheduling in wireless ad hoc networks with successive interference cancellation," in *Proc. IEEE INFOCOM*, Shanghai, China, Apr. 2011, pp. 1287–1295.
[3] N. Benvenuto, G. Carnevale, and S. Tomasin, "Optimum power control and ordering in SIC receivers for uplink CDMA systems," in *Proc. IEEE Int. Conf. Commun.*, Seoul, South Korea, May 2005, pp. 2333–2337.
[4] T. Manglayev, R. C. Kizilirmak, Y. H. Kho, N. Bazhayev, and I. Lebedev, "NOMA with imperfect SIC implementation," in *Proc. 17th Int. Conf. Smart Technol.*, Ohrid, Macedonia, Jul. 2017, pp. 22–25.
[5] D. Zhang, Y. Liu, L. Dai, A. K. Bashir, A. Nallanathan, and B. Shim, "Performance analysis of FD-NOMA-based decentralized V2X systems," *IEEE Trans. Commun.*, vol. 67, no. 7, pp. 5024–5036, Jul. 2019.
[6] L. Shi, Y. Shi, Y. Ye, Z. Wei, and J. Han, "An efficient interference management framework for multi-hop wireless networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Shanghai, China, Apr. 2013, pp. 1434–1439.
[7] L. Qu, J. He, and C. Assi, "Understanding the benefits of successive interference cancellation in multi-rate multi-hop wireless networks," *IEEE Trans. Commun.*, vol. 62, no. 7, pp. 2465–2477, Jul. 2014.
[8] C. Jiang, Y. Shi, X. Qin, X. Yuan, Y. T. Hou, W. Lou, S. Kompella, and S. F. Midkiff, "Cross-layer optimization for multi-hop wireless networks with successive interference cancellation," *IEEE Trans. Wireless Commun.*, vol. 15, no. 8, pp. 5819–5831, Aug. 2016.
[9] Q. He, D. Yuan, and A. Ephremides, "Maximum link activation with cooperative transmission and interference cancellation in wireless networks," *IEEE Trans. Mobile Comput.*, vol. 16, no. 2, pp. 408–421, Feb. 2017.
[10] C. Jiang, Y. Shi, Y. T. Hou, W. Lou, S. Kompella, and S. F. Midkiff, "Squeezing the most out of interference: An optimization framework for joint interference exploitation and avoidance," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, Mar. 2012, pp. 424–432.
[11] L. Shi, Y. Hu, J. Xu, Y. Shi, and X. Ding, "The mobile base station strategy for wireless networks with successive interference cancellation," *IEEE Access*, vol. 7, pp. 88570–88578, Jun. 2019.
[12] C. Li, Y. Xu, Y. Xu, B. Diao, and Z. An, "Mlops: A sic-based minimum frame length with optimized power scheduling for Uans," *IEEE Access*, vol. 7, pp. 21133–21146, Jan. 2019.
[13] J.-S. Liu, C.-H.-R. Lin, and J. Tsai, "Delay and energy tradeoff in energy harvesting multi-hop wireless networks with inter-session network coding and successive interference cancellation," *IEEE Access*, vol. 5, pp. 544–564, 2017.
[14] O. Simeone, "A very brief introduction to machine learning with applications to communication systems," *IEEE Trans. Cognit. Commun. Netw.*, vol. 4, no. 4, pp. 648–664, Dec. 2018.
[15] R. Choudhary and H. K. Gianey, "Comprehensive review on supervised machine learning algorithms," in *Proc. Int. Conf. Mach. Learn. Data Sci. (MLDS)*, Noida, India, Dec. 2017, pp. 37–43.
[16] A. Forster, "Machine learning techniques applied to wireless ad-hoc networks: Guide and survey," in *Proc. 3rd Int. Conf. Intell. Sensors, Sensor Netw. Inf.*, Melbourne, QLD, Australia, 2007, pp. 365–370.
[17] T. Li, M. Zhao, and K. K. L. Wong, "Machine learning based code dissemination by selection of reliability mobile vehicles in 5G networks," *Comput. Commun.*, vol. 152, pp. 109–118, Feb. 2020.
[18] M. Chen, T. Wang, K. Ota, M. Dong, M. Zhao, and A. Liu, "Intelligent resource allocation management for vehicles network: An A3C learning approach," *Comput. Commun.*, vol. 151, pp. 485–494, Feb. 2020.
[19] M. Huang, K. Zhang, Z. Zeng, T. Wang, and Y. Liu, "An AUV-assisted data gathering scheme based on clustering and matrix completion for smart ocean," *IEEE Internet Things J.*, early access, Apr. 15, 2020, doi: 10.1109/JIOT.2020.2988035.
[20] M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1996–2018, 4th Quart., 2014.

[21] R. Saravanan and P. Sujatha, "A state of art techniques on machine learning algorithms: A perspective of supervised learning approaches in data classification," in *Proc. 2nd Int. Conf. Intell. Comput. Control Syst. (ICICCS)*, Madurai, India, Jun. 2018, pp. 945–949.
[22] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
[23] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," in *Emerging Artificial Intelligence Applications in Computer Engineering*, I. Maglogiannis, K. Karpouzis, B. A. Wallace, and J. Soldatos, Eds. Amsterdam, The Netherlands: IOS Press, 2007, pp. 3–24.
[24] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, 1992.
[25] K.-L.-A. Yau, K. H. Kwong, and C. Shen, "Reinforcement learning models for scheduling in wireless networks," *Frontiers Comput. Sci.*, vol. 7, no. 5, pp. 754–766, Aug. 2013.
[26] M. Bourenane, "Adaptive scheduling in mobile ad hoc networks using reinforcement learning approach," in *Proc. Int. Conf. Innov. Inf. Technol.*, Abu Dhabi, United Arab Emirates, Apr. 2011, pp. 392–397.
[27] K. Li, W. Ni, M. Abolhasan, and E. Tovar, "Reinforcement learning for scheduling wireless powered sensor communications," *IEEE Trans. Green Commun. Netw.*, vol. 3, no. 2, pp. 264–274, Jun. 2019.
[28] J. Niu, "Self-learning scheduling approach for wireless sensor network," in *Proc. 2nd Int. Conf. Future Comput. Commun.*, Wuha, China, 2010, pp. 253–257.
[29] A. Jacquelin, M. Vilgelm, and W. Kellerer, "Grant-free access with multipacket reception: Analysis and reinforcement learning optimization," in *Proc. 15th Annu. Conf. Wireless On-demand Netw. Syst. Services (WONS)*, Wengen, Switzerland, Jan. 2019, pp. 83–90.
[30] M. V. da Silva, R. D. Souza, H. Alves, and T. Abrao, "A NOMA-based Q-learning random access method for machine type communications," *IEEE Wireless Commun. Lett.*, early access, Jun. 16, 2020, doi: 10.1109/LWC.2020.3002691.
[31] L. Xiao, Y. Li, C. Dai, H. Dai, and H. V. Poor, "Reinforcement learning-based NOMA power allocation in the presence of smart jamming," *IEEE Trans. Veh. Technol.*, vol. 67, no. 4, pp. 3377–3389, Apr. 2018.
[32] A. Goldsmith, *Wireless Communication*. New York, NY, USA: Cambridge Univ. Press, 2005.
[33] J. Vermorel and M. Mohri, "Multi-armed bandit algorithms and empirical evaluation," in *Proc. 16th Eur. Conf. Mach. Learn. (ECML)*, Porto, Portugal, Oct. 2005, pp. 437–448.
[34] E. S. Low, P. Ong, and K. C. Cheah, "Solving the optimal path planning of a mobile robot using improved q-learning," *Robot. Auton. Syst.*, vol. 115, pp. 143–161, Oct. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889018308285

EZGI METE received the B.S.E. degree in computer engineering from the University of Michigan, USA, in 2006, and the M.S. degree in electrical engineering from the Polytechnic Institute of NYU, USA, in 2009. She is currently pursuing the Ph.D. degree with the TOBB University of Economics and Technology, Ankara, Turkey. From 2010 to 2011, she worked as an Assistant Systems Design Engineer with Turkish Aerospace Industries. From 2012 to 2014, she worked as a Lecturer with the University of Turkish Aeronautical Association. Her research interests include resource allocation, scheduling, routing, and machine learning for wireless networks.

TOLGA GIRICI (Senior Member, IEEE) received the B.S. degree from Middle East Technical University, Ankara, Turkey, in 2000, and the Ph.D. degree from the University of Maryland, College Park, MD, USA, in 2007, both in electrical engineering. In 2005, he has spent six months as an Intern with Intelligent Automation Inc., Rockville, MD, USA. He was a Research Assistant with the Fujitsu Laboratories, College Park, from 2006 to 2007. He is currently an Associate Professor with the TOBB University of Economics and Technology, Ankara. His research interests include resource allocation and optimization in next generation cellular wireless access networks, wireless ad hoc networks, smart grid, and tactical networks. He received the TUBITAK Career Award, in 2010, and actively collaborates with industry.